Improving graph matching via density maximization

Chao Wang
*University of Wollongong, chaow@uow.edu.au*

Lei Wang
*University of Wollongong, leiw@uow.edu.au*

Lingqiao Liu
*Australian National University*

Follow this and additional works at: [https://ro.uow.edu.au/eispapers](https://ro.uow.edu.au/eispapers)

Part of the [Engineering Commons](https://ro.uow.edu.au/eispapers), and the [Science and Technology Studies Commons](https://ro.uow.edu.au/eispapers)

**Recommended Citation**

Wang, Chao; Wang, Lei; and Liu, Lingqiao, "Improving graph matching via density maximization" (2013). *Faculty of Engineering and Information Sciences - Papers: Part A*. 2351.

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au
Improving graph matching via density maximization

Abstract
Graph matching has been widely used in various applications in computer vision due to its powerful performance. However, it poses three challenges to image sparse feature matching: (1) The combinatorial nature limits the size of the possible matches, (2) It is sensitive to outliers because the objective function prefers more matches, (3) It works poorly when handling many-to-many object correspondences, due to its assumption of one single cluster for each graph. In this paper, we address these problems with a unified framework-Density Maximization. We propose a graph density local estimator (DLE) to measure the quality of matches. Density Maximization aims to maximize the DLE values both locally and globally. The local maximization of DLE finds the clusters of nodes as well as eliminates the outliers. The global maximization of DLE efficiently refines the matches by exploring a much larger matching space. Our Density Maximization is orthogonal to specific graph matching algorithms. Experimental evaluation demonstrates that it significantly boosts the true matches and enables graph matching to handle both outliers and many-to-many object correspondences.

Keywords
era2015

Disciplines
Engineering | Science and Technology Studies

Publication Details

This conference paper is available at Research Online: https://ro.uow.edu.au/eispapers/2351
Improving Graph Matching via Density Maximization

Chao Wang, Lei Wang
School of Computer Science & Software Engineering
University of Wollongong, NSW, Australia, 2522
chaow, leiw@uow.edu.au

Lingqiao Liu
CECS, Australian National University
ACT 0200, Canberra, Australia
lingqiao.liu@cecs.anu.edu.au

Abstract

Graph matching has been widely used in various applications in computer vision due to its powerful performance. However, it poses three challenges to image sparse feature matching: (1) The combinatorial nature limits the size of the possible matches; (2) It is sensitive to outliers because the objective function prefers more matches; (3) It works poorly when handling many-to-many object correspondences, due to its assumption of one single cluster for each graph. In this paper, we address these problems with a unified framework—Density Maximization. We propose a graph density local estimator (DLE) to measure the quality of matches. Density Maximization aims to maximize the DLE values both locally and globally. The local maximization of DLE finds the clusters of nodes as well as eliminates the outliers. The global maximization of DLE efficiently refines the matches by exploring a much larger matching space. Our Density Maximization is orthogonal to specific graph matching algorithms. Experimental evaluation demonstrates that it significantly boosts the true matches and enables graph matching to handle both outliers and many-to-many object correspondences.

1. Introduction

Sparse feature correspondence (SFC) is a fundamental problem for a wide range of applications in computer vision, such as image retrieval, object recognition, 3D reconstruction, and motion estimation. Since these feature sets have meaningful internal structure, they are often considered as two separate graphs, but not simply as point sets. As a result, SFC can be modelled as graph matching in which graph nodes represent features extracted from each image while graph edges represent relationships between features. Graph matching finds a mapping between the two feature sets by minimizing the distortions of the two graphs. Compared to the parametric models (e.g. Thin-Plate Spline) and the methods with geometric constraints (e.g., RANSAC with rigid transformation assumption), graph matching provides greater flexibility for object modeling and is more robust to large non-rigid transformations.

There have been a myriad of algorithms proposed for graph matching [7]. Those before 1990s did not aim to optimize a well-defined objective function. Among recent algorithms, the Integer Quadratic Programming (IQP) has emerged as a de facto formulation of graph matching [2, 8, 14, 15, 23, 19, 11]. IQP explicitly considers both unary and pair-wise terms which reflect the compatibilities in feature appearance as well as pair-wise geometric relationships. Since IQP is NP-complete, the optimal solution is virtually unachievable and approximations are required. While recent approximate methods have led to tremendous progress, the results for many real-world images are still far from being perfect due to several factors.

Aside from its NP-complete nature, IQP owns several limitations some of which might not have been explicitly pointed out before. Firstly, the combinatorial nature of graph matching makes computation of the full affinity matrix in IQP intractable for large graphs. Secondly, due to the non-negative property of the edge attributes, the objective function of IQP prefers more matches even if they are outliers. Last but not least, IQP assumes that each graph contains only one cluster of nodes. In real-world cases, however, image pairs can have a large number of sparse features, significant clutter, multiple objects, and even many-to-many object correspondences. Therefore, graph matching poses three challenges to SFC: (1) its combinatorial nature limits the size of the possible matches; (2) it is sensitive to outliers; (3) it works poorly for many-to-many object correspondences.

To address the first challenge, most methods establish the set of candidate matches by using unary descriptors of discriminative features, such as SIFT [18], at a relatively low cost. Then only a small number of matches are utilized to build an initial graph. Their results might be unsatisfactory due to the loss of useful information hidden in the full matching space [5]. Cho et al.[5] proposed a progressive framework to update candidate matches based on pair-wise geometric relationships between new matches and the cur-
rent graph matching result. It greatly boosts the objective function of IQP. However, it tends to introduce many outliers because the current graph matching result might be noisy. Furthermore, its computational complexity is high because exploring the full matching space is required.

To address the second challenge, some popular attempts extend the general graph to the hyper-graph [9, 13, 24]. By using higher-order constraints (e.g., projective invariance) instead of the unary or pair-wise ones, such methods successfully filter out most outliers. Unfortunately, they also work poorly for many-to-many object correspondences due to the single cluster assumption.

To address both the second and the third challenges simultaneously, unsupervised clustering might be the most promising approach. Each cluster of matches naturally corresponds to one object pair, and the outliers are filtered out by eliminating the clusters with small sizes or authorities[3]. Cho et al.[1] and Zhang et al.[25] proposed two novel methods based on agglomerative clustering. Such methods are based on heuristic rules and therefore global optimum cannot be guaranteed. Other attempts perform clustering via mode-seeking in the graph domain. Liu et al.[17] introduced a graph shift algorithm to detect dense subgraphs with iterative shrinking and expansion. Jouili et al.[12] presented a median graph shift which is an extension of the medoid shift based on the concept of the median graph. Both methods perform mode-seeking by shifting from one subgraph to another subgraph, but not between nodes. As will be shown, such method largely depends on the initialization and is prone to local minima. Cho et al.[3] proposed a node-shifting scheme based on the high-order personalized PageRank (PPR) matrix. Its iterative PPR propagation scheme tends to accumulate errors on outliers, and PPR matrix is computationally expensive.

In this paper, we try to solve those challenges with a unified framework—Density Maximization. We first propose a density local estimator (DLE) which is a reliable measure for the quality of matches. Our work is inspired by Lin et al.[16] who observed that the geometric transformations associated with neighbouring true matches are smoothly varying even for significant displacements. The basic idea of DLE is to measure the quality of a match by using only the inliers from a local smooth neighbourhood, in order to avoid being cluttered by outliers and other object correspondences. Density Maximization is then modeled as maximization of the DLE values both locally and globally. Our local maximization, named Density-Ascent Shift (DAS), detects clusters of nodes as well as eliminates outliers. DAS is a mode-seeking method similar to [12, 3, 17], but is much more robust to clutter. Furthermore, DAS is much faster than [12, 3, 17] because it does not require iterations while [12, 3, 17] do. Our global maximization, called Density-Ascent Update (DAU), refines the candidate matches by efficiently exploring a much larger matching space. DAU is similar to the progression method of Cho et al.[5] which updates matches in a progressive way, but is more than one order of magnitude faster than [5] while introducing much less outliers.

Density Maximization performs DAS and DAU iteratively until convergence. At each iteration, the result of DAS is the starting point of DAU. This simple scheme ensures that updating candidate matches is mainly based on the inliers, thus leading to a high precision. Similar to the progression method [5], Density Maximization is orthogonal to specific graph matching algorithms and can be used to improve any of them. Experimental evaluation on extensive natural images demonstrates that Density Maximization significantly boosts the true matches and enables graph matching to handle outliers and many-to-many object correspondences.

Compared to the state-of-the-art methods, Density Maximization has the following advantages:

1. It addresses the three challenges of graph matching in a unified framework.
2. It is much more robust to significant clutter.
3. It is more than one order of magnitude faster.
4. Its precision is much higher.

2. Background

2.1. Graph matching formulation

Let $G^P = (V^P, E^P, A^P)$ and $G^Q = (V^Q, E^Q, A^Q)$ be two attributed graphs, where $V$ denotes a set of nodes, $E$, edges, and $A$, attributes. The objective of graph matching is to find a mapping between $V^P$ and $V^Q$, represented by a binary assignment matrix $X \in \{0, 1\}^{n^P \times n^Q}$ with $n^P$ and $n^Q$ denoting the numbers of nodes in $G^P$ and $G^Q$ respectively. $X_{i,a} = 1$ implies that node $v^P_i \in V^P$ matches node $v^Q_a \in V^Q$. Let $x \in \{0, 1\}^{n^P \times n^Q}$ denote the column-wise vectorized replica of $X$, the integer quadratic programming (IQP) formulates graph matching as

$$x^* = \arg \max_x x^T W x,$$

subject to:

$$\forall i \sum_{a=1}^{n^Q} x_{ia} \leq 1, \forall a \sum_{i=1}^{n^P} x_{ia} \leq 1$$

and

$$x \in \{0, 1\}^{n^P \times n^Q}.$$

The two-way constraints of (1) refer to the one-to-one matching from $G^P$ to $G^Q$. In sparse feature correspondence, the nodes represent features extracted from each image while the edges denote relationships between features. The symmetric affinity matrix $W$ encodes both the unary and pairwise similarities. A diagonal element $W_{i,a;ia}$ represents a unary similarity of a match $(v^P_i, v^Q_a)$, and a non-diagonal term $W_{i,a;j,b}$ refers to a pairwise similarity of two matches $(v^P_i, v^Q_a)$ and $(v^P_j, v^Q_b)$. Every element of $W$ is non-negative.
2.2. Analysis of IQP

Aside from the NP-complete nature, IQP has several other limitations.

Firstly, the combinatorial nature makes the computation of $W$ intractable. A real-world image of common size contains more than $n = 1000$ sparse features by using the popular affine or scale invariant detectors such as SIFT [18], MSER [20] and Harris Affine [21]. This results in a huge affinity matrix of dimension $n \times n = 1000^2$. Most graph matching methods reduce the size of $W$ by using matches at a relatively high unary similarity. Such a simple scheme often leads to the loss of useful information hidden in the full matching space [5].

Secondly, IQP prefers more matches. Let $v^p_i$ and $v^q_i$ denote two noisy features which have no true matching ones, setting $X_{ia} = 1$ non-decreases the objective function $x^T W x$ because every element of $W$ is non-negative. This means that IQP prefers including the matches of all the features, even if they might be outliers. To alleviate this problem, some methods sacrifice $x$ by increasing large values while smoothing out small values [2, 8, 14, 23, 11]. The results still contains many outliers.

Finally, IQP assumes that the feature set from one image belongs to a single cluster. This means that the quality of one match is measured based on all current matches. From (1) we can see that the contribution of each match $(v^p_i, v^q_m)$ to the objective function is

$$C(l, m) = x_{lm}(\sum_{i \neq l, a \neq m} W_{ia:lm} x_{ia} + \sum_{j \neq l, b \neq m} W_{lm:jb} x_{jb})$$

If $C(l, m) > C(l, s)$, IQP prefers $(v^p_i, v^q_m)$ to $(v^p_i, v^q_s)$, and vice versa. $C(l, m)$ contains the similarities between $(v^p_i, v^q_m)$ and all the other matches. This measure is problematic for many-to-many object correspondences because the matches in one object correspondence might clutter those in others. To avoid this, each object correspondence should be considered independently.

3. Density Maximization

Similar to [2, 4, 14, 18], we construct an association graph $G^ag = (V^ag, E^ag, A^ag)$ based on the affinity matrix $W$. We take each candidate match $(v^p_i, v^q_m)$ as a node $v_{ia} \in V^ag$, and its associated weight $W_{ia:jm}$ as the attribute of the edge $e_{ia:jm} \in E^ag$. Then the original graph matching problem between $G^p$ and $G^q$ becomes node selection problem in the graph $G^ag$. For brevity, we will use a single letter to index the node of $G^ag$ in the following sections, e.g., $v_i$ denotes the $i$-th node and $W_{ij}$ denotes the component of $W$ at the $i$-th row and the $j$-th column.

Figure 1 shows the framework of Density Maximization. Given an image pair, the salient features are firstly extracted from each image and then $N_C$ candidate matches are readily established using unary descriptors of the features at a relatively low cost. Those matches are taken as the nodes of an initial association graph. Graph matching selects best nodes from it in order to maximize the objective function in (1). Then a reduced set of nodes and their edges are selectively used to construct a new graph which is called valid graph $G^v$ in this paper. Density Maximization improves $G^v$ by iterating $DAS$ and $DAU$. Based on $G^v$, $DAS$ finds the clusters of nodes as well as removes the outliers by local maximization of the $DLE$ values. $DAU$ produces a updated graph $G^u$ with $N_C$ nodes by global maximization of the $DLE$ values via exploring a much larger graph called potential graph $G^p$. At each iteration, the result of $DAS$ is the starting point of $DAU$. This ensures that the updates of matches are mainly based on inliers. The iterations continue until the total $DLE$ values no longer increase. The final step $DAS$ further removes the outliers introduced by $DAU$. As can be seen from Fig.1, Density Maximization is orthogonal to specific graph matching algorithms, and any
of them can be adopted as the graph matching module in the framework.

3.1. Density local estimator

Recently, graph density has shown its potential to identify inliers and detect strongly connected node clusters in an association graph. A few attempts to define the graph density include the average kernel density of Liu et al. [17], the random walk density of Cho et al. [4], and the personalized PageRank density of Cho et al. [3]. Now we define our density local estimator (DLE). The main difference between DLE and other methods lies in its well-defined local smooth domain.

The intuitive of DLE is to estimate graph density at one node by using only the nodes within a same object, so that it can avoid the clutter problem introduced by outliers and the nodes in other objects. However, it is difficult to determine whether two nodes belong to a same object. Fortunately, it has been observed that the geometric transformations associated with neighbouring matches in a same object are smoothly varying even for significant displacements [16]. This reveals a simple method to approximately identify the nodes within a same object by using a local smooth neighbourhood $\Omega$. $\Omega(i)$ should satisfy two criteria: (1) Locality: the neighbours are within a close proximity to node $v_i$. (2) Smoothness: the neighbours should have similar values to $v_i$ for a same measure. These criteria prevent the scope of neighbours from extending into outliers and the nodes in other objects.

We adopt the popular kernel density estimation method to compute the graph density locally. We consider node selection as a distribution and use $x_i$ to denote the probability of selecting node $v_i$. Suppose we sample the distribution $N \rightarrow \infty$ times, then the number of selecting $v_i$ is $N x_i$. The density at $v_i$ is

$$DLE(i) = \frac{1}{N} \sum_{j \in \Omega(i)} N x_j K(i, j) = \frac{1}{\Omega(i)} \sum_{j \in \Omega(i)} x_j K(i, j)$$  \hspace{1cm} (3)$$

This is called DLE in this paper. $K(i, j) = W_{i,j}$ implies the similarity between $v_i$ and $v_j$. The only difference between DLE and the classical kernel density estimation lies in $\Omega$.

The Locality of $v_j$ with respect to $v_i$ is defined by using the k-nearest neighbour function $kNN(., k)$:

$$L(v_i, v_j) = \begin{cases} 1 & \text{if } v_j \in kNN(v_i, k), \\ 0 & \text{otherwise}. \end{cases}$$  \hspace{1cm} (4)$$

The Smoothness of $\Omega(i)$ is defined on two measures: the geometric transformation and the probability of node selection. The Smoothness of geometric transformation between node $v_i$ and $v_j$ amounts to $W_{i,j}$. The Smoothness of probability is measured by $exp(-(x_i - x_j)^2/\sigma^2)$ with a parameter $\sigma$. Then $\Omega(i)$ is defined as a $\varepsilon-neighbourhood$

$$\Omega(i) = \{v_j \in V^V | \Phi(i, j) > \varepsilon \} \cup \{v_i\}$$  \hspace{1cm} (5)$$

where $\Phi(i, j) = L(v_i, v_j)W_{i,j}exp(-(x_i - x_j)^2/\sigma^2)$ and the parameter $\varepsilon$ controls the size of $\Omega(i)$. Figure 2 demonstrates the impact of $\Omega$. With the constraint of $\Omega$, the inliers’ DLE values almost consistently larger than those of outliers at nearby locations.

The node selection probability $x$ naturally corresponds to the solutions to (1) since many graph matching methods solve (1) by relaxing the constraints on $x$ such that its elements can take real values in $[0, 1]$. In those methods, $x$ can be viewed as the confidence that the matches are true [14] or as the probability of visits by random walks [2, 13, 4]. In this paper we consider $x$ as the node selection probability. For other graph matching methods in which $x$ are integer, we normalize $x$ to obtain a uniform distribution. Therefore DLE is orthogonal to specific graph matching algorithms whether $x$ are continuous or not, unlike other methods.

3.2. Density Ascent Shift

As shown in Fig.1, the aim of DAS is to produce node clusters and eliminate outliers from valid graph $G^V$. DAS is a mode-seeking method and the density modes on a graph are defined as follows.
Algorithm 1: Density Ascent Shift

Input: matching \((G^V, x)\)

Output: clean graph \(G^C\) and an indicator \(I_S\) for clusters

1. compute \(DLE(i) \forall v_i \in G^V\)
2. for each node \(v_i \in G^V\)
   \[ DA(i) = \arg \max_{j \in \Omega(i)} K(i, j)\Delta DLE(j) \]
   end
3. assign each node \(v_i\) to its mode by a tree traversal along \(DA(i)\), and compute the total-density
4. compute the \(TDP\) for each cluster, and remove outliers with \(TDP < t\)
5. produce final graph \(G^C\) using the left clusters, set \(I_S(i) = m\) if node \(v_i\) belongs to the \(m\)-th cluster.

Definition 1 Density modes on a graph are local maximizers of the \(DLE\) values.

\(DAS\) performs mode-seeking along the density-ascent direction. The density-ascent \(DA(i)\) of node \(v_i\) is formulated as

\[ DA(i) = \arg \max_{j \in \Omega(i)} K(i, j)\Delta DLE(j) \quad (6) \]

which means the neighbouring node of \(v_i\) with the highest expectation of \(DLE\) increment. This density-ascent is the steepest ascent over the \(DLE\) values within \(\Omega(i)\). \(\Omega(i)\) prevents shifting into irrelevant clusters. Similar to other mode-seeking methods[12, 4, 3], \(DAS\) is guaranteed to converge.

Theorem 1 A finite sequence of density-ascent shifts from any node converges to a density mode.

Proof Since \(\Omega(i)\) of any node \(v_i\) includes itself, the \(DLE\) values of a sequence of shifts keep strictly increasing until the shifts reach a node whose density-ascent is itself. The final node, therefore, is the density mode, and the length of the sequence is \(|V^V|\) at most, with \(|V^V|\) denoting the number of nodes in \(G^V\).

For each node, we compute its density-ascent just once. Then the successive density-ascent for any node already exists. The trajectory of nodes sharing a common density mode builds a tree, and leads to a natural cluster. Then the cluster label of all nodes associated with each disjoint tree can be assigned in a single tree traversal, similar to the medoid shift[22].

We define the total-density of each cluster as the sum of the \(DLE\) values of its members, and the total-density-percentage (\(TDP\)) of each cluster as the ratio between its total-density and the sum of the total-densities of all the clusters. The clusters of outliers usually have very small total-density, so that \(TDP\) provides a reliable measure for detection and elimination of them, as shown in Fig.3. \(DAS\) is depicted in Algorithm 1.

Algorithm 2: Density Ascent Update

Input: potential graph \(G^T\), clean graph \(G^C, x\) and \(N_C\)

Output: a updated graph \(G^U\)

1. \(nx(i) \leftarrow 0, dx(i) \leftarrow 0, \forall v_i \in G^T\)
   for each node \(v_j \in G^C\)
   \[ nx(i) \leftarrow nx(i) + x_j K(j, i) \]
   \[ dx(i) \leftarrow dx(i) + K(j, i) \]
   end
   \[ x \leftarrow nx_i/dx \]
2. \(DLE(i) \leftarrow 0, \forall v_i \in G^T\)
   for each node \(v_j \in G^C\)
   \[ DLE(i) \leftarrow DLE(i) + x_j K(j, i) \]
   end
3. \(G^U \leftarrow N_C\) nodes with the largest \(DLE\) values

3.3. Density Ascent Update

Given a clean graph \(G^C\), the aim of \(DAU\) is to produce a updated graph \(G^U\) with \(N_C\) nodes by maximizing the total \(DLE\) values. To achieve this, \(DAU\) explores the potential graph \(G^T\) which contains \(G^C\) but is much larger. \(G^T\) covers most true matches and will be detailed later. \(DAU\) firstly estimates the \(DLE\) values of the nodes in \(G^T\), and then select \(N_C\) best nodes with largest values to construct \(G^U\). Since \(G^C \subset G^T\), this global maximization scheme ensures that \(DAU\) non-decreases the total \(DLE\) values.

To compute the \(DLE\) value for each node \(v_i\) in \(G^T\), we need to identify \(\Omega(i)\) at first. But the node selection probability \(x_i\) might be unavailable if \(v_i\) does not belong to \(G^C\). Here we approximate \(x_i\) by

\[ x_i = \frac{\sum_{j \in \Omega'(i)} x_j K(i, j)}{\sum_{j \in \Omega'(i)} K(i, j)} \quad (7) \]

which is a weighted average of the selection probabilities over a local smooth neighbourhood \(\Omega'(i)\). \(\Omega'(i)\) is similar to \(\Omega(i)\) but does not consider the probability Smoothness since \(x_i\) is unknown. However, \(x_j\) for \(j \in \Omega(i)\) might be unavailable. We observe that \(\Omega'(i)\) is nearly symmetric for true matches. By investigating the nodes for true matches in Fig.1 we find that if \(j \in \Omega'(i)\) the probability for \(i \in \Omega'(j)\) is above 90%. Therefore (7) can be approximately rewritten as \(x_i = \frac{\sum_{j \in \Omega'(j)} x_j K(j, i)}{\sum_{j \in \Omega'(j)} K(j, i)}\). Let \(nx(i) = \sum_{j \in \Omega'(j)} x_j K(j, i)\) and \(dx(i) = \sum_{j \in \Omega'(i)} K(j, i)\), the contribution of each node \(v_j\) in \(G^C\) to \(nx(i)\) is \(x_j K(j, i)\), and that to \(dx(i)\) is \(K(j, i)\) if \(i \in \Omega'(j)\). Therefore all \(x_i\) can be estimated very efficiently by traversing the nodes of \(G^C\).

Since \(\Omega'(i)\) is nearly symmetric, \(\Omega(i)\) is also nearly symmetric because \(\exp(-(x_i - x_j)^2/\sigma^2)\) is symmetric. Then
(3) can be rewritten as \( DLE(i) = \sum_{j \in G(j)} c_j K(j, i) \) which means that the contribution of each node \( v_j \) in \( G^C \) to \( DLE(i) \) is \( c_j K(j, i) \). Therefore all \( DLE(i) \) can be efficiently calculated by traversing the nodes of \( G^C \). \( DAU \) is summarized in Algorithm 2.

The potential graph \( G^P \) is constructed using \( Z \) matches for each feature based on the unary similarity. We test on the image pairs of the intra-class dataset[1] which own large intra-category variations, and find that \( G^T \) covers more than 95% true matches when \( Z = 40 \). This reveals that exploring the whole matching space like Cho et al.[5] is unnecessary.

3.4. Analysis

Using the approximate nearest neighbour (ANN) search, the computational complexity of \( DAU \) is \( O(k |V^V| \log(Zn^P)) \) with \( |V^V| \) denoting the node number of \( G^V \), \( n^P \) denoting the node number of graph \( G^P \) (i.e., the feature number of one image) and \( k = 50 \). As far as we know, the only work similar to \( DAU \) is the progression method[5] whose computational complexity is \( O(k_1k_2 |V^V| \log(n^P) \log(n^Q)) \) with \( k_1 = 25 \) and \( k_2 = 5 \). \( DAU \) is more than one order of magnitude faster because \( k_1k_2 \log(n^P) \log(n^Q) / k \log(Zn^P) > 10 \) for general cases with \( n^P > 1000 \) and \( n^Q > 1000 \). The main difference is that \( DAU \) explores the potential graph \( G^T \) while the progression method searches the whole matching space based on \( G^V \). Since \( G^T \) covers most true matches, exploring \( G^T \) does not degrade the performance. On the other hand, this scheme successfully avoids many outliers in the whole matching space, as will be shown in the experiments.

The computational complexity of \( DAS \) is \( O(k |V^V| \log |V^V|) \), more than one order of magnitude faster than most mode-seeking methods. The high efficiency benefits from its non-iteration scheme. More importantly, either \( DAU \) or \( DAS \) is much faster than most graph matching methods [2, 8, 14, 15, 23, 19, 11], indicating that we can improve graph matching without introducing too much computational cost.

An important trick of our Density Maximization framework is that \( DAU \) takes the result of \( DAS \) as its starting point at each iteration. This ensures that the updating matches is mainly based on inliers. In this way, Density Maximization significantly increases the precision in sharp contrast with the progression method, as shown in Fig.4.

4. Experiments

In our experiments, the candidate matches are generated using the SIFT descriptor. To measure the similarity between two matches \((v_i^P, v_j^Q)\) and \((v_i^Q, v_j^Q)\), we adopted the symmetric transfer error \( d(i; j) \) used in [5, 13, 1, 4]. The affinity matrix \( W \) is calculated by \( W_{ia;ib} = \max(50 - d(i; j), 0) \). In Density Maximization, we set \( \sigma = 0.2 \), \( k = 50 \), \( \varepsilon = 10 \) and \( t = 0.03 \).

![Figure 4. Performance growth on the image pair in Fig.1 by our Density Maximization and the progression method[5]. The plot shows the precision w.r.t the iteration steps. Note that the step 0 denotes the result by graph matching.](image)

![Figure 5. (a)The result by GP[5] based on the graph matching result in Fig.1. (b)The result by our \( DAU \). (c)The result by GP together with our \( DAS \). The false match clusters are denoted by yellow and magenta lines. (d)The result by our \( DAU \) together with our \( DAS \). There is no false match clusters.](image)

We test Density Maximization on three challenging benchmark datasets: Intra-class dataset[1], ETHZ toys dataset[10], Co-recognition dataset[6]. Intra-class dataset consists of 30 image pairs of large transformations and intra-class variation. It provides detected MSER features and initial matches. In this dataset, most images have a small number of features and only several hundreds of initial matches. For fair comparison, we adopt those features and always fix the number of candidate matches \( N_C \) to the same as the number of the given initial matches. ETHZ toys dataset includes 9 different rigid/non-rigid objects together with the test images of significant clutter. Co-recognition dataset contains 6 image pairs with complex many-to-many object correspondences. The ground truth feature correspondences are manually constructed for each image pair to enable quantitatively evaluation. For these two dataset, we use the MSER and the Harris affine detectors with the SIFT descriptor, and set \( N_C = 3000 \). Our testing environment is MS Windows 7 Professional with Intel Core i5-3550 CPU 3.3GHz, 16GB RAM.

4.1. Density Maximization vs related work

Density Maximization contains novel approaches to both updating matches (i.e., \( DAU \)) and clustering matches (i.e., \( DAS \)). We will show the effectiveness of it in both of these
Firstly we compare our DAU with the graph progression (GP)[5] since it is the only similar work to DAU as far as we know. For fair comparison, we adopt the same progressive framework as GP, which performs graph matching and updating matches iteratively. Since the aims of both DAU and GP are to boost the true matches, we access Recall on the three datasets. The results of GP contain lots of overlapping matches. To compute Recall more accurately, we count the overlapping matches only once. The overall results are given in Table 1. Compared to our DAU, GP tends to introduce more outliers which are very difficult to remove. Figure 5 shows the outliers of an example. The average times for each iteration of GP to process each image pair in the three data-sets are 0.45, 5.32 and 31.53 seconds respectively. The corresponding times for DAU are 0.27, 1.64 and 3.73 seconds respectively.

Secondly, we compare our DAS with two state-of-the-arts methods: the agglomerative correspondence clustering (ACC)[1] and the Shrink-and-Expansion (SAE)[17]. Since SAE cannot handle both ETHZ toys and Co-recognition datasets (the source code provided by the authors on the internet has ’out of memory’ problem when handling thousands of matches), we only report its result for Intra-class dataset. Since the aim of DAS is to improve Precision, we access Precision on the three datasets. The overall results are given in Table 1 and an example is shown in Fig.6. SAE tends to include lots of outliers. The results of ACC are much better, but are still noisy. In contrast, our DAS successfully detects true matches and distinguishes them from outliers. To process each image pair in the three datasets DAS takes only 0.69, 1.72 and 1.93 seconds on average, while ACC takes 8.74, 78.52 and 95.73 seconds. SAE is much slower and takes more than one minute on average to process each image pair of Intra-class dataset.

Finally, we compare our Density Maximization with a combined method—GP+ACC. GP+ACC is performed in a similar way of Density Maximization: GP and ACC are performed iteratively till convergence. We measure both Recall and Precision on the three datasets. As shown in Fig.7, the outliers introduced by GP cannot be eliminated by ACC, and result in noisy clusters. So GP+ACC increases Recall at the expense of Precision. Our Density Maximization solves this problem effectively by avoiding outliers from source. It largely outperforms GP+ACC in both precision and recall as shown by Table 1.

## 4.2. Density Maximization vs Graph Matching

In this experiment, we show the improvement of Density Maximization on several state-of-the arts graph matching methods: SM[14], PM[24], BGM[8], IPFP[15] and RRWM[2]. The quantitative results are summarized in Table 2, and some examples are shown in Fig.8. The graph matching methods cannot distinguish inliers from outliers, and fail to separate matches of one object from those of others. Density Maximization solves these problems effectively by detecting clusters of true matches. The precision
Figure 8. (a) and (b) are two input image pairs. (c) Result by RRWM[2] for (a). (d) Result by SM[2] for (b). (e) The result by our Density Maximization with RRWM as the graph matching module. (f) The result by our Density Maximization with SM as the graph matching module. True matches are shown with color lines and false matches are shown with black lines.

is boosted by 36% ∼ 67%, and the recall by 18% ∼ 47%.

5. Conclusion

We introduced a unified framework, called Density Maximization, which effectively resolves the three limitations of conventional graph matching and achieves impressive performance improvement. By globally and locally maximizing a novel proposed density estimator, i.e., density local estimator, Density Maximization leads to the integration of updating matches, eliminating outliers and cluster detection. We point out that the key to the high performance is twofold: a well-defined local smooth neighbourhood to avoid clutter and an iteration scheme to ensure that updating matches is mainly based on inliers. Experiments demonstrate that Density Maximization is adequate for very challenging real-world images which contain many-to-many object correspondences and significant outliers.

6. Acknowledgement

This work is supported by Australian Research Council (ARC) Linkage Grant LP0991757.

References