1-1-2015

# Density maximization for improving graph matching with its applications

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

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

Lingqiao Liu
*University of Adelaide*

# Density maximization for improving graph matching with its applications


## Abstract

Graph matching has been widely used in both image processing and computer vision domain due to its powerful performance for structural pattern representation. 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 its objective function prefers more matches; and 3) it works poorly when handling many-to-many object correspondences, due to its assumption of one single cluster of true matches. In this paper, we address these challenges with a unified framework called density maximization (DM), which maximizes the values of a proposed graph density estimator both locally and globally. DM leads to the integration of feature matching, outlier elimination, and cluster detection. Experimental evaluation demonstrates that it significantly boosts the true matches and enables graph matching to handle both outliers and many-to-many object correspondences. We also extend it to dense correspondence estimation and obtain large improvement over the state-of-the-art methods. We further demonstrate the usefulness of our methods using three applications: 1) instance-level image retrieval; 2) mask transfer; and 3) image enhancement.



## Keywords

matching, its, applications, maximization, density, improving, graph


## Disciplines

Engineering | Science and Technology Studies


## Publication Details

Wang, C., Wang, L. & Liu, L. (2015). Density maximization for improving graph matching with its applications. IEEE Transactions on Image Processing, 24 (7), 2110-2123.

# Density Maximization for Improving Graph Matching with Its Applications

Chao Wang*, Lei Wang, and Lingqiao Liu

***Abstract*—Graph matching has been widely used in both image processing and computer vision domain due to its powerful performance for structural pattern representation. 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 its objective function prefers more matches; (3) It works poorly when handling many-to-many object correspondences, due to its assumption of one single cluster of true matches. In this paper, we address these challenges with a unified framework called Density Maximization (DM) which maximizes the values of a proposed graph density estimator both locally and globally. DM leads to the integration of feature matching, outlier elimination and cluster detection. Experimental evaluation demonstrates that it significantly boosts the true matches and enables graph matching to handle both outliers and many-to-many object correspondences. We also extend it to dense correspondence estimation and obtain large improvement over the state-of-the-art methods. We further demonstrate the usefulness of our methods by using three applications: instance-level image retrieval, mask transfer and image enhancement.**

***Index Terms*—Graph Matching, Sparse Feature Matching, Dense Correspondence, Image retrieval, Mask Transfer, Image Enhancement.**

## I. Introduction

Sparse feature matching (SFM) is a fundamental problem for a wide range of applications in both image processing and computer vision domain, such as image retrieval, object recognition, 3D reconstruction, and motion estimation [1] [2]. Since the image sparse feature sets for matching have meaningful internal structure, they are often considered as two separate graphs, but not simply as point sets. As a result, SFM can be modeled 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 distortion of the two graphs. Compared to the parametric models (e.g. Thin-Plate Spline [3]) and the methods with geometric constraints (e.g. RANSAC [4] with affine 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 [1]. Those proposed 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 [5]–[12]. 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 [13]. 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 SFM: (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 [14], at a relatively low cost. Then only a small number of candidate 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 [13]. Cho et al. [13] proposed a progressive framework to update candidate matches based on pair-wise geometric relationships between new matches and the current graph matching result. It greatly boosts the objective function value 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 impose higher-order constraints (e.g. projective invariance) [15]–[17] on hyper graph, global constraints on all nodes [18], and the locally affine-invariant constraints on neighboring nodes [19]. The latest work [20] adopts a max-pooling approach to eliminate outlier matches. Those methods successfully filter out most outliers for single object correspondence. Unfortunately, they work poorly for many-to-many object correspondences due to the assumption of a single cluster of true matches.

Chao Wang and Lei Wang are with the School of Computer Science and Software Engineering, University of Wollongong, Australia. Lingqiao Liu is with the School of Computer Science, University of Adelaide, Australia. The authors' emails are: chaow@uow.edu.au, leiw@uow.edu.au and liulq83@gmail.com. This work is supported by Australian Research Council (ARC) Linkage Grant LP0991757.

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 [21] or authorities [22]. Cho et al. [21] and Zhang et al. [23] 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. [24] introduced a graph shift algorithm to detect dense subgraphs with iterative shrinking and expansion. Jouili et al. [25] 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 methods largely depend on the initialization and are prone to local minima. Cho et al. [22] 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 to obtain.

In this paper, we try to solve those challenges with a unified framework—Density Maximization (DM) which is complementary to graph matching methods. 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. [26] who observed that the geometric transformations associated with neighboring true matches of a same object are smoothly varying even for significant displacements. The basic idea of $DLE$ is to measure the quality of a match by using only the matches from a local smooth neighborhood, in order to avoid being cluttered by outliers and the matches from other objects. DM 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 the Shrink-and-Expansion (SAE) method [24], the median graph shift (MGS) method [25], and the Authority-shift clustering (ASC) method [22], but is much more robust to background clutters than those three methods. Furthermore, our $DAS$ is much faster than those methods because it does not require iterations while those methods 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. [13] which updates matches in a progressive way, but is more than one order of magnitude faster than [13] while introducing much less outliers.

Our DM 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 true matches, thus leading to a high precision. Similar to the progression method [13], our DM is orthogonal to specific graph matching algorithms and can be used to improve any of them. Experimental evaluation on extensive natural images demonstrates that our DM significantly increase the true matches and enables graph matching to better handle outliers and many-to-many object correspondences.

Compared to the state-of-the-art methods, our DM 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.

In addition to its high performance in sparse feature matching, our DM can be easily extended to estimate dense correspondences.

Traditional dense correspondence methods for image stitching [27] and stereo matching [28] only consider relatively simple geometric deformations (e.g. 1D disparity for stereo matching and parametric motion for image stitching). The methods [29] for optical flow estimate 2D translations in both horizontal and vertical directions for each pixel. DeepFlow [30] handles the large translations in optical flow by using the deep network. SIFTflow [31] significantly improves the robustness to intra-category variations by using SIFT feature distance, and also model the geometric deformation as 2D translations. When complex deformations (e.g., scaling) exist, the above methods might completely fail. The deformable spatial pyramid (DSP) [32] and PatchMatch [3] alleviate this problem by searching over a small pre-defined affine set. Although they can produce nice results for small deformations, the results might become far from being satisfying when the deformations are beyond the pre-defined affine set. Our method addresses this issue by building dense correspondence directly from the sparse matches obtained by our DM. Benefiting from the robustness of DM, our method is able to handle significant transformations which are beyond the capability of the state-of-the-art methods [33]–[35].

Our dense correspondence method based on DM is similar to the smoothly varying affine stitching field (SVASF) model [26] and the locally affine sparse-to-dense matching (LASM) method [31], all of which solve dense correspondence based on sparse feature matches. SVASF solves the stitching field based on the noisy SIFT matches while our method solves the stitching field based on much more accurate matches obtained by our DM. LASM differs from our method in three aspects: (1)the features in LASM is located on a uniform grid while ours adopts the popular sparse features [36] [37], (2)LASM only addresses translations while ours addresses more complex deformations such as affine combined with non-rigid motions, (3)LASM explicitly detects occlusion by using binary classification while ours implicitly propagates affine transformations to occluded regions by using the SVASF model.

Therefore, our DM can benefit a variety of applications that currently rely on previous sparse feature matching and dense correspondence methods. We demonstrate this with three applications: instance-level image retrieval, mask transfer, and image enhancement.

This paper is the extended version of our conference paper [38]. The extension includes: (1) a novel scene-level dense correspondence method, (2) a novel object-level dense correspondence method, and (3) three novel applications.

The remainder of the paper is organized as follows: Section II describes the technical background, and Section III proposes our DM. Sections IV evaluates the performance of DM for sparse feature matching. The extension to dense correspondences is given in Section V. Section VI develops three applications of our methods, and Section VII draws the final conclusions.

## II. Background

For clarity, we list in Table I the notations of the graphs used in this paper. In this Section, we first introduce the integer quadratic programming (IQP) formulation, and then analyze its limitations.

### A. 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 $v_i^P \in V^P$ matches node $v_a^Q \in V^Q$. Let $x \in \{0,1\}^{n^P n^Q}$ denote the column-wise vectorized replica of $X$, the integer quadratic programming (IQP) formulates graph matching as

$$x^* = \arg\max_x x^T W x, \tag{1}$$

$s.t. \sum_{a=1}^{n^Q} x_{ia} \leq 1, \forall i, \sum_{i=1}^{n^P} x_{ia} \leq 1, \forall a, x \in \{0,1\}^{n^P n^Q}$

The two-way constraints of (1) refer to the one-to-one matching from $G^P$ to $G^Q$. In sparse feature matching, the nodes represent features extracted from each image while the edges denote relationships between features. The pre-defined symmetric affinity matrix $W$ encodes both the unary and pairwise similarities. A diagonal entry $W_{ia;ia}$ represents a unary similarity of a match $(v_i^P, v_a^Q)$, and an off-diagonal entry $W_{ia;jb}$ refers to a pairwise similarity of two matches $(v_i^P, v_a^Q)$ and $(v_j^P, v_b^Q)$. Every entry of $W$ is non-negative.

### B. Analysis of IQP

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

Firstly, the combinatorial nature makes the computation of $W$ intractable [13]. A real-world image of a common size like $1000 \times 1000$ pixels contains more than $n = 1000$ sparse features by using the popular affine or scale invariant detectors such as SIFT [14], MSER [36] and Harris Affine [37]. The number of possible matches amount to $n \times n = 1000^2$ and this results in a huge affinity matrix $W$ of dimension $(n \times n)^2 = 1000^4$. To build such a huge matrix is intractable. Most graph matching methods reduces the number of candidate matches by using unary descriptors of discriminative features, such as SIFT, at a relatively low cost. Such a simple scheme often removes many true matches and therefore leads to the loss of useful information hidden in the full matching space [13].

Secondly, IQP prefers more matches. Let $v_i^P$ and $v_a^Q$ denote two noisy features which have no matching ones, setting $x_{ia} = 1$ non-decreases the objective function $x^T W x$ because every entry of $W$ is non-negative. This means that IQP prefers to include the matches of all the features, even if they might be outliers. To alleviate this problem, many methods [5]–[7], [9], [11] relax the integer constraint on $x$ such that its elements can take real values in [0,1], and then remove the matches with very small $x$ values. However, the results still contain many outliers.

Finally, IQP assumes that all the true matches compose a strongly connected cluster. To detect which match should be included in the cluster (thus taken as a true match), the relations between the match and all the other matches are evaluated. The measure for the relations of the match $(v_l^P, v_m^Q)$ can be derived by isolating from (1) the components involving $(v_l^P, v_m^Q)$:

$$M(l,m) = x_{lm}\Big(\sum_{i \neq l, a \neq m} W_{ia;lm} x_{ia} + \sum_{j \neq l, b \neq m} W_{lm;jb} x_{jb}\Big) \tag{2}$$

If $M(l,m) > M(l,s)$, IQP prefers $(v_l^P, v_m^Q)$ to $(v_l^P, v_s^Q)$, where $s$ denotes any other node in $G^Q$. So $M(l,m)$ is the quality measure for the match $(v_l^P, v_m^Q)$. Note that $M(l,m)$ contains the similarities between $(v_l^P, v_m^Q)$ and all the other matches. If many-to-many object correspondences exist between two images, the true matches often compose several isolated clusters. Then the above quality measure for matches become problematic because the matches in one object correspondence might clutter those in others. This is the latent reason that IQP-based graph matching algorithms cannot handle many-to-many object correspondences. To avoid this, each object correspondence should be considered independently.

## III. Density Maximization (DM)

**Algorithm 1: Density Maximization**
**Input**: image P and Q, number of candidates $N_C$
**Output**: clean graph $G^C$ and an indicator $I_S$ for clusters
1 $(G^T, G^I) \leftarrow Findinitialcandidates(P, Q)$
2 $(G^V, x) = GraphMatching(G^I)$
3 $(G^C, I_S) = DAS(G^V, x)$
4 While $\sum_{v_i \in V^C} DLE(i)$ increase do
$(G^U, x) = DAU(G^T, G^C, x, N_C)$
$(G^C, I_S) = DAS(G^U, x)$
end

In this Section, we first introduce the framework of our Density Maximization (DM), and then propose the graph density local estimator (DLE). Finally, we detail the two components of DM: Density Ascent Shift (DAS) and Density Ascent Update (DAU).

Our DM is performed on an association graph $G^{ag} = (V^{ag}, E^{ag}, A^{ag})$, similar to [5] [39] [7] [14]. To construct $G^{ag}$, we need to define graph nodes $V^{ag}$, edges $E^{ag}$ and attributes $A^{ag}$. We take each candidate match $(v_i^P, v_a^Q)$ as a node $v_{ia} \in V^{ag}$. In the affinity matrix $W$, the entry $W_{ia;jb}$ measures the mutual consistency between the candidate matches $(v_i^P, v_a^Q)$ and $(v_j^P, v_b^Q)$. We take $W_{ia;jb}$ as the attribute $a_{ia;jb} \in A^{ag}$ of the edge $e_{ia;jb} \in E^{ag}$. The edge $e_{ia;jb}$ connects node $v_{ia}$ and $v_{jb}$. Then we have constructed the association graph $G^{ag}$, and the original graph matching problem between $G^P$ and $G^Q$

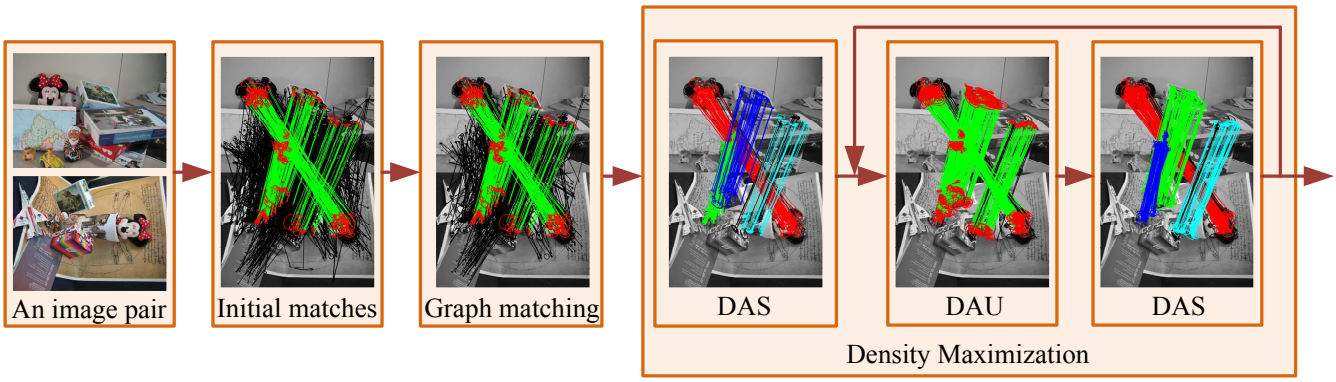



Fig. 1. Overview of our Density Maximization (DM) framework for improving graph matching methods. The Graph Matching result contains 283 true matches together with 315 outliers. DM improves Graph matching by iterating between Density-Ascent Shift ($DAS$) and Density-Ascent Update ($DAU$). $DAS$ eliminates most outliers and detects four clusters of true matches. $DAU$ boosts the number of true matches to 416 and introduces only 63 outliers. The final step $DAS$ further removes 19 outliers. True matches are shown with color lines and outliers are shown with black lines.

TABLE I
NOTATIONS OF GRAPH.

| $G^P$ | $G^Q$ | $G^I$ | $G^V$ | $G^C$ | $G^U$ | $G^T$ |
|---|---|---|---|---|---|---|
| The attributed graph built on the features of image $P$ | The attributed graph built on the features of image $Q$ | The initial graph built on SIFT matches | The valid graph produced by graph matching | The clean graph by our DAS | The updated graph by our DAU | The potential graph built on $Z$ SIFT matches |

becomes a node selection problem in the graph $G^{ag}$, which is essentially the problem in (1) [5] [39] [7] [14]. 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 $W_{i;j}$ denotes the entry of $W$ at the $i$th row and the $j$th column.

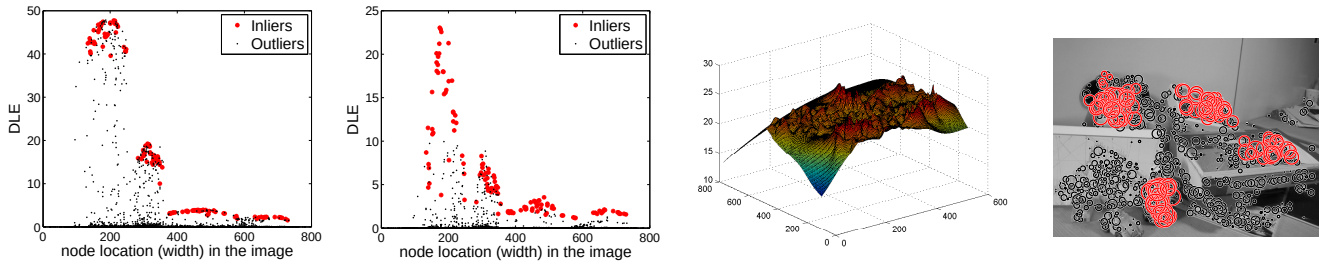
Algorithm 1 shows the framework of our Density Maximization (DM) and Fig.1 gives the immediate result for each step. We can see that our DM is complementary to graph matching methods. Given an image pair, the salient features are firstly extracted from each image and then $N_C$ candidate matches are readily established by using descriptors of the features at relatively low cost as [13] [16] [21]. $N_C$ is a user input constant and equals 3000 in this paper. Those matches are taken as the nodes to build an initial association graph $G^I$. We also build a much larger association graph $G^T$ and will detail it later. We denote the process to build both $G^I$ and $G^T$ by using function $Findinitialcandidates()$ as shown in Algorithm 1. Since our goal is to improve graph matching methods, one graph matching method is firstly adopted to select nodes from $G^I$. Then the selected set of nodes and their edges are used to construct a new graph which is called valid graph $G^V$ in this paper. Our DM improves $G^V$ by iterating between Density-Ascent Shift ($DAS$) and Density-Ascent Update ($DAU$). Based on $G^V$, $DAS$ finds the clusters of nodes as well as removes the outliers by local maximization of the $DLE$ values, producing a clean graph $G^C$. Based on $G^C$, $DAU$ produces an updated graph $G^U$ with $N_C$ nodes by global maximization of the $DLE$ values via exploring the much larger graph $G^T$. At each iteration, the result of $DAS$ is the starting point of $DAU$. This ensures that the updates of matches are mainly based on true matches. The iteration continues until the total $DLE$ value for graph $G^C$ no longer increases. In the framework of our DM, any graph matching algorithm can be adopted as the graph matching module. So our DM is orthogonal to specific graph matching algorithms.

### A. Density local estimator

Recently, graph density [24] [39] [22] has shown its potential to identify true matches 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. [24], the random walk density of Cho et al. [39], and the personalized PageRank density of Cho et al. [22]. Now we define our density local estimator ($DLE$). The main difference between $DLE$ and the above definitions lies in its novel local smooth domain.

The intuitive of $DLE$ is to estimate the graph density at a node in one object by using only the nodes within the same object. By doing so, it can avoid the clutter problem caused by outliers and the nodes in different objects. However, it is difficult to determine whether two nodes belong to the same object. Fortunately, it has been observed that the geometric transformations associated with neighboring matches in a same object are smoothly varying even for significant displacements [26]. Based on this observation, we propose to approximately identify the nodes within a same object by using a local smooth neighborhood $\Omega$. $\Omega(i)$ should satisfy two criteria: (1) Locality: the neighbors are within a close proximity to node $v_i$ in Euclidean space. (2) Smoothness: the neighbors should have similar geometric transformations and similar probabilities of node selection to $v_i$. These criteria prevent the scope of neighbors from extending into outliers and the nodes in the 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




(a) (b) (c) (d)

Fig. 2. Graph density estimation for the true matches and outliers from the image pair in Fig.1. In (a) and (b), $x-axis$ denotes the width of the target image in Fig.1 and $y-axis$ denotes the graph density value. (a) Without $\Omega$ constraint, the $DLE$ values of the true matches are mixed with those of the nearby outliers. Red stars denote true matches and black dots denote outliers. (b) With $\Omega$ constraint, the true matches' $DLE$ values are almost consistently larger than those of outliers at nearby locations. (c) 3D plot of $DLE$ values. (d) Red circles denote the features for the true matches and black circles denote the features for outliers. The diameters of the circles are proportional to the $DLE$ values of the corresponding matches.

selecting node $v_i$. Suppose we sample the distribution $N($ $N \to \infty$) times, then the number of selecting $v_i$ is $Nx_i$. The graph density at $v_i$ is

$$DLE(i) = \frac{\sum_{j\in\Omega(i)} Nx_j K(i,j)}{N} = \sum_{j\in\Omega(i)} x_j K(i,j) \quad (3)$$

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 $\Omega$.

Now we define $\Omega$ by using its two criteria: Locality and Smoothness. The Locality indicator function $L(v_i, v_j)$ of $v_j$ with respect to $v_i$ is defined by using the k-nearest neighbour function $kNN(\cdot, k)$:

$$L(v_i, v_j) = 1 \quad if \quad v_j \in kNN(v_i, k), \quad 0 \quad otherwise. \quad (4)$$

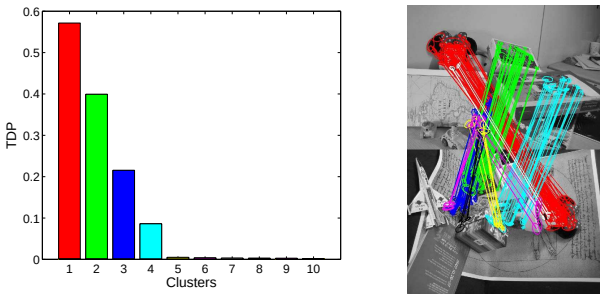
where $k = 50$ in this paper. The Smoothness of $v_j$ with respect to $v_i$ is defined on both the geometric transformation and the probability of node selection. The Smoothness of geometric transformation between node $v_i$ and $v_j$ is defined as $W_{i;j}$. The Smoothness of probability is measured by $\exp(-(x_i - x_j)^2/\sigma^2)$ with a parameter $\sigma = 0.2$. Then $\Omega(i)$ is defined as a $\varepsilon - neighbourhood$

$$\Omega(i) = \{v_j \in V^V | \Phi(i,j) > \varepsilon\} \bigcup \{v_i\} \quad (5)$$

where $\Phi(i,j) = L(v_i, v_j) W_{i,j} \exp(-(x_i - x_j)^2/\sigma^2)$ and the parameter $\varepsilon = 10$ controls the size of $\Omega(i)$. $V^V$ is the node set of graph $G^V$ produced by a graph matching method.

Figure 2 demonstrates the impact of $\Omega$ by the matches from the image pair in Fig.1. With the constraint of $\Omega$, the true matches' $DLE$ values are almost consistently larger than those of outliers at nearby locations. This means that $DLE$ is a good quality measure for matches.

The node selection probability $x$ can be produced by any graph matching method. Many graph matching methods solve (1) by relaxing the constraints on $x$ such that its elements can take continuous values in [0,1]. Then $x$ can be viewed as the confidence that the matches are true in [7] or as the probability of visits by random walks in [5] [16] [39]. In this paper we consider $x$ as the node selection probability. For other graph matching methods in which $x$ are binary, we divide each element by the sum of all the elements to obtain a uniform distribution. Therefore our $DLE$ is orthogonal to a specific graph matching algorithm whether $x$ are continuous or not.




(a) (b)

Fig. 3. (a)Top 10 max total-density-percentage ($TDP$) values. (b)The clusters for top 10 max TDP values. The true clusters of the four object pairs (denoted with red, green, blue and cyan lines) have TDP values significantly larger than those of outlier clusters.

### B. Density Ascent Shift

**Algorithm 2: Density Ascent Shift** ($DAS$)
**Input**: matching result $(G^V, x)$ by some graph matching method
**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 $v_i \in G^V$ do
$$DA(i) = \arg\max_{j\in\Omega(i)} \frac{K(i,j)}{\sum_{j\in\Omega(i)} K(i,j)} \Delta DLE(j)$$
end
3 assign each node $v_j$ to its mode by a tree traversal along $DA(i)$, and compute the total-density of each cluster
4 compute the $TDP$ for each cluster, and remove clusters with $TDP < t$
5 produce final graph $G^C$ by using the left clusters, set $I_S(i) = m$ if node $v_i$ belongs to the $m$th cluster

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 in this paper are defined as follows.

**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

$$DA(i) = \arg\max_{j \in \Omega(i)} \frac{K(i,j)}{\sum_{j \in \Omega} K(i,j)} \Delta DLE(i,j) \qquad (6)$$

where $\Delta DLE(i,j) = DLE(j) - DLE(i)$. $\frac{K(i,j)}{\sum_{j \in \Omega} K(i,j)}$ can be taken as the probability of jumping from $v_i$ to $v_j$, and then $DA(i)$ refers to the neighboring node of $v_i$ with the highest expected $DLE$ 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 [22], [25], [39], $DAS$ is guaranteed to converge, as proved as follows.

**Theorem 1** A finite sequence of density-ascent shifts from any node converge 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 [40].

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 value and the sum of the total-density values of all the clusters. It has been observed in [21] [24] [39] that the outlier clusters usually have very small total-density values based on their graph density definitions. We find that this observation also hold for our $DLE$. We test it on three popular benchmark data sets [21] [41] [42] and find no failure example. Based on this observation, we can use $TDP$ to detect and eliminate outlier clusters since they often have much less $TDP$ values than the clusters of true matches. Figure 3 shows the top 10 max $TDP$ values of the match clusters of the image pair in Fig.1. The four clusters of true matches have $TDP$ values significantly larger than those of the outlier clusters. Therefore the outliers can be easily eliminated by using a small threshold $t$ for $TDP$. Algorithm 2 gives the details about our $DAS$ method. The output of $DAS$ is a set of match clusters which compose the clean graph $G^C$ with a node set $V^C$.

Our $DAS$ is more robust than other clustering methods because of three reasons: 1) it is based on our $DLE$ which is a robust measure for the quality of matches, 2) it is a mode-seeking method which imposes no constraints on deformation shapes, and 3) it is guaranteed to converge.

### C. Density Ascent Update

Given the clean graph $G^C$ produced by our $DAS$, the aim of our $DAU$ is to update $G^C$ by increasing the total $DLE$ value. 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 evaluates the $DLE$ values of the nodes in $G^T$, and then select the $N_C$ nodes with largest $DLE$ values to construct the updated graph $G^U$. $N_C$ is a user input constant and generally we have $N_C > \|V^C\|$ with $\|V^C\|$ denoting the node number of $G^C$. Since $G^C \subset G^T$, this global maximization scheme ensures that $DAU$ non-decreases the total $DLE$ value.

**Algorithm 3: Density Ascent Update ($DAU$)**
**Input**: potential graph $G^T$, clean graph $G^C$,$x$ and $N_C$
**Output**: a updated graph $G^U$

```
1   nx(i) ← 0, dx(i) ← 0, ∀v_i ∈ G^T
    for each node v_j ∈ G^C do
        for each node v_i ∈ Ω'(j) do
            nx(i) ← nx(i) + x_j K(j,i)
            dx(i) ← dx(i) + K(j,i)
        end
    end
    x ← nx./dx
2   DLE(i) ← 0, ∀v_i ∈ G^T
    for each node v_j ∈ G^C do
        for each node v_i ∈ Ω(j) do
            DLE(i) ← DLE(i) + x_j K(j,i)
        end
    end
3   G^U ← N_C nodes with the largest non-zero DLE values
```

To compute the $DLE$ value for each node $v_i$ in $G^T$, we need to identify $\Omega(i)$ firstly according to (5). However, the node selection probability $x_i$ produced by graph matching methods might be unavailable if $v_i$ does not belong to $G^C$. For an unknown $x_i$, we estimate it by

$$x_i = \frac{\sum_{j \in \Omega'(i)} x_j K(i,j)}{\sum_{j \in \Omega'(i)} K(i,j)} \qquad (7)$$

which is a weighted average of the selection probabilities over a local smooth neighborhood $\Omega'(i)$. $\Omega'(i)$ is similar to $\Omega(i)$ but does not consider the Smoothness of probability 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. For example, 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 $x_i = nx(i)/dx(i) = \sum_{i \in \Omega'(j)} x_j K(j,i) / \sum_{i \in \Omega'(j)} K(j,i)$. The contribution of each node $v_j$ in $G^C$ to the numerator $nx(i)$ is $x_j K(j,i)$, and that to the denominator $dx(i)$ is $K(j,i)$ if $i \in \Omega'(j)$. Therefore all $x_i$ can be now 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 $DLE(i) = \sum_{i \in \Omega(j)} x_j K(j,i)$ which means that the contribution of each node $v_j$ in $G^C$ to $DLE(i)$ is $x_j K(j,i)$. Therefore all $DLE(i)$ for graph $G^T$ can be efficiently calculated by traversing the nodes of $G^C$. Our $DAU$ select the $N_C$ nodes with the largest Non-zero $DLE$ values to construct the updated graph $G^U$. Our $DAU$ guarantees one-to-one correspondences. For all the one-to-many matches which share a common feature, $DAU$ only retains the one which has the largest DLE value. Algorithm 3 gives the details about our $DAU$ method.

The potential graph $G^T$ is constructed using $Z$ matches for

each feature based on the SIFT similarity. We test on the image pairs in the intra-class dataset [21], and find that $G^T$ covers more than $90\%$ true matches when $Z = 40$. This suggests that exploring the whole matching space like Cho et al. [13] might be unnecessary. Then the number of candidate matches is significantly reduced from about $n \times n = 1000^2$ to only $40n = 40000$.

### D. Analysis of time complexity

As shown in Algorithm 1, our DM includes four building blocks: the function $Findinitialcandidates$, a graph matching method and our $DAU$ and $DAS$. For each image feature, $Findinitialcandidates$ finds its nearest neighbour to build $G^I$, and $Z$ nearest neighbors to build $G^T$. Using the approximate nearest neighbour (ANN) search, the computational complexity of $Findinitialcandidates$ reaches $O(Zn^P \log(Zn^Q))$. The computational complexity of current graph matching methods ranges from $O(n^P \times n^Q)$ to $O(|V^V|^4)$. We adopt the RRWM [5] of which the computational complexity is only $O(n^P \times n^Q)$.

By using the 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 [13] 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 than the progression method [13] 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 only $G^T$ 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.

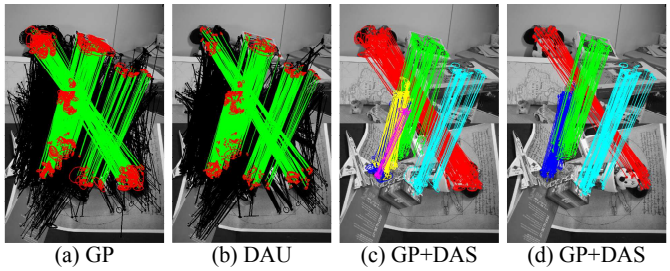
The computational complexity of $DAS$ is $O(k|V^V|\log|V^V|)$, more than one order of magnitude faster than most mode-seeking methods. The high efficiency benefits from its non-iteration scheme. More importantly, both $DAU$ and $DAS$ are much faster than most graph matching methods [5]–[11], indicating that we can improve graph matching without introducing too much computational cost.

## IV. Sparse feature matching and evaluation

In this section, we use our DM to solve sparse feature matching (SFM). First, the candidate matches are generated by using the SIFT descriptor. To measure the similarity between two matches $(v_i^P, v_a^Q)$ and $(v_j^P, v_b^Q)$, we adopted the symmetric transfer error $d(ia; jb)$ used in [13] [16] [21] [39]. The affinity matrix $W$ is calculated by $W_{ia;ib} = max(50 - d(ia; jb), 0)$. In Density Maximization, we set the threshold $t$ for match clusters to 0.03.

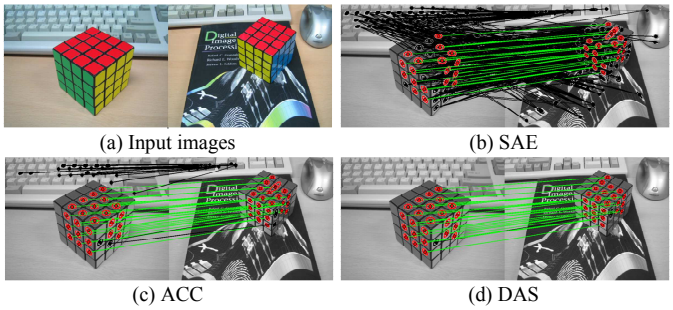
We test our DM on three challenging benchmark datasets: Intra-class dataset [21], ETHZ toys dataset [41], and Co-recognition dataset [42]. Intra-class dataset consists of 30 image pairs with large transformations and intra-class variation. ETHZ toys dataset includes nine different rigid/non-rigid objects together with the test images of significant clutter. Co-recognition dataset contains six image pairs with complex many-to-many object correspondences. The ground truth feature correpondences are manually constructed for each image pairs to enable quantitatively evaluation. We use the MSER [36] and the Harris affine [37] detectors with the SIFT descriptor [14], and set $N_C = 3000$. Our testing environment is MS Windows 7 Professional with Intel Core i5-3550 CPU 3.3GHz, 16GB RAM.

We first compare our DM with the related work and then show the improvement by our DM on several state-of-the-art graph matching methods.



(a) GP (b) DAU (c) GP+DAS (d) GP+DAS

Fig. 4. (a)The result by the graph progression (GP) [13] based on the graph matching result in Fig.1. (b)The result by our $DAU$ based on the graph matching result in Fig.1. (c)The result by GP together with our $DAS$. The outlier clusters are denoted by yellow and magenta lines. (d)The result by our $DAU$ together with our $DAS$. There is no outlier clusters.



(a) Input images (b) SAE (c) ACC (d) DAS

Fig. 5. (a)A image pair. (b)The result by SAE [24]. (c)The result by (ACC) [21]. (d)The result by our $DAS$. True matches are shown with green lines and outliers are shown with black lines.

### A. Our DM vs related work

Our DM 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 steps as well as a whole. We adopt the graph match algorithm RRWM [5] in our DM.

Firstly we compare our $DAU$ with the graph progression (GP) [13] 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 match updating iteratively. Since the aims of both $DAU$ and GP are to boost the true matches, we evaluate the Recall on the three datasets. The results of GP contain many overlapping matches. To compute Recall more accurately, we count the overlapping

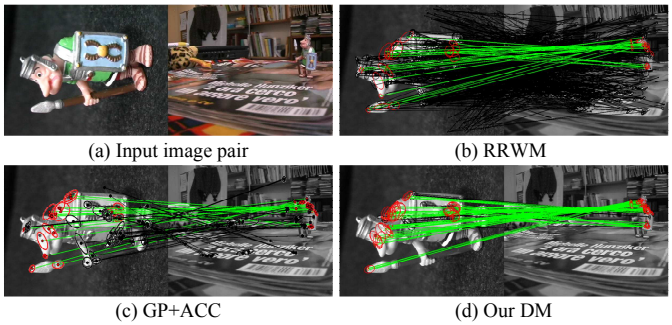
TABLE II
RECALL (%) OF PG AND OUR $DAU$, PRECISION (%) OF ACC, SAE AND OUR $DAS$, RECALL/PRECISION/RUNNING TIME OF PG+ACC AND OUR DM. 'NO' DENOTES THE FAILURE OF SAE.

| | Recall | | Precision | | | Recall/ | Precision/ | running time(seconds) |
|---|---|---|---|---|---|---|---|---|
| Data sets | PG | $DAU$ | ACC | SAE | $DAS$ | PG+ACC | AAS | DM |
| Intra-class | 81 | 83 | 71 | 43 | 83 | 71/70/15 | 22/52/3 | 73/81/4 |
| ETHZ toys | 69 | 77 | 63 | No | 85 | 62/69/86 | 23/66/5 | 72/88/8 |
| Co-recognition | 66 | 81 | 67 | No | 91 | 61/74/117 | 75/69/7 | 75/92/10 |

matches only once. The overall results are given in Table II. Compared to our $DAU$, GP tends to introduce more outliers which are very difficult to remove. Figure 4(a) gives the result by the GP method for the image pair in Fig.1. The outliers cannot be eliminated by our $DAS$ as shown in Fig.4(c). In contrast, our $DAU$ introduces much less outliers which can be easily removed by our $DAS$ as shown in Fig.4(d).

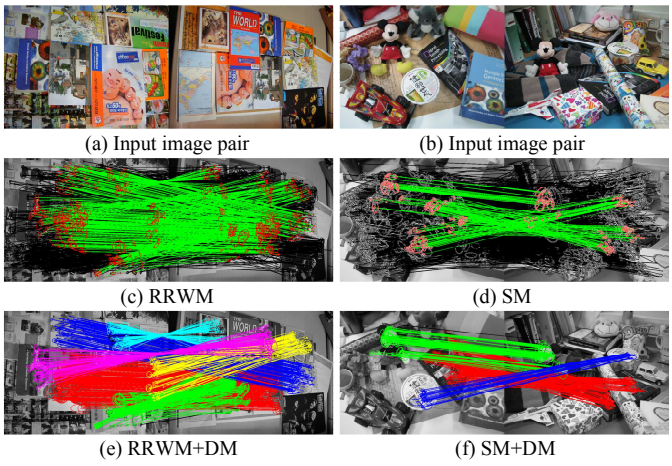
Secondly, we compare our $DAS$ with two state-of-the-art methods: the agglomerative correspondence clustering (ACC) [21] and the Shrink-and-Expansion (SAE) [24]. Since SAE cannot handle both ETHZ toys and Co-recognition datasets (the source code provided by the authors online reports 'out of memory' 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 use precision as the evaluation criterion on the three datasets. The overall results are given in Table II and an example is shown in Fig.5. As seen, SAE tends to include many 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.

Finally, we compare our DM with a combined method—GP+ACC. GP+ACC is performed in a similar way of our DM: GP and ACC are performed iteratively till convergence. We measure both Recall and Precision on the three datasets. As shown in Fig.6, 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 DM solves this problem effectively by avoiding outliers from source. It largely outperforms GP+ACC in both precision and recall. The average running time in Table II shows that our DM is much faster than GP+ACC. In Table II. We also give the quantitative results of the authority ascent shift (AAS) [39] which is a mode-seeking method. We can see that our DM produces much higher recall than AAS.



(a) Input image pair (b) RRWM (c) GP+ACC (d) Our DM

Fig. 6. (a)A image pair. (b)Graph matching result by [5]. (c)The result by GP+ACC. (d)The result by our DM True matches are shown with green lines and outliers are shown with black lines.



(a) Input image pair (b) Input image pair (c) RRWM (d) SM (e) RRWM+DM (f) SM+DM

Fig. 7. (a) and (b) are two input image pairs. (c)Result by RRWM [5] for (a). (d)Result by SM [5] for (b). (e)The result by our DM with RRWM as the graph matching module. (f)The result by our DM with SM as the graph matching module. True matches are shown with color lines and outliers are shown with black lines.

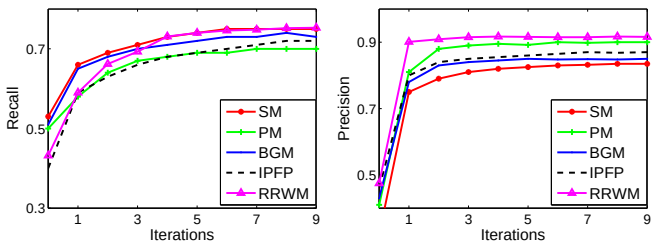
### B. DM with different graph matching methods

In this paper, our original aim is to improve graph matching methods. Now, we show the improvement of our DM on several state-of-the-art graph matching methods: SM [7], PM [17], BGM [6], IPFP [8] and RRWM [5]. The quantitative results are summarized in Table III, and some examples are shown in Fig.7. The graph matching methods themselves cannot distinguish true matches from outliers, and fail to separate matches of one object from those of others. Our DM solves these problems effectively by detecting clusters of true matches. The precision is boosted by $36\% \sim 67\%$, and the recall by $18\% \sim 47\%$.

To give a better picture to show the performance improvement over graph matching methods, we plot in Fig.8 the increase of Recall and Precision at each iteration of DM on the Co-recognition data set. The result shows a significant improvement even after single iteration step of our DM, and the maximum performance can be achieved within about five steps.

## V. EXTENSION TO DENSE CORRESPONDENCE

In this section, we extend our DM to estimate both scene-level and object-level dense correspondence.




(a) (b)

Fig. 8. Performance growth on the Co-recognition dataset by our DM over several state-of-the-art graph matching methods: SM [7], PM [17], BGM [6], IPFP [8] and RRWM [5]. The plot shows the recall and precision w.r.t the iteration steps. Note that the step 0 denotes the result by graph matching. (a)Recall. (b)Precision.

TABLE III
PERFORMANCE IMPROVEMENT (%) OVER RECALL/PRECISION BY DM

| Data sets | SM | PM | BGM | IPFP | RRWM |
|---|---|---|---|---|---|
| Intra-class | 43/47 | 39/44 | 32/36 | 47/39 | 45/41 |
| ETHZ toys | 23/64 | 27/52 | 18/45 | 33/67 | 29/53 |
| Co-recognition | 27/51 | 25/49 | 21/42 | 33/41 | 31/44 |

### A. Scene-level dense correspondences

Based on the sparse feature matches produced by our DM method, the affine transformations for each pixel can be easily estimated according to the smoothly varying affine stitching field (SVASF) model [26]. We first compute the SVASF, and then solve for the dense correspondences by optimizing an energy function with the SVASF as a prior term.

In the SVASF model, each point $j$ has an associated affine transformation $a_j$ which is biased towards a pre-computed global affine transformation $a_G$. The SVASF is a set of affine parameters for all the sparse features. Recall that our DM method produces $|V^C|$ sparse feature matches. For each match $j$, its associated affine transformation $a_j$ can be easily obtained as in [37] [21] [13] because a sparse feature can be represented by an elliptical region with its orientation being estimated by the dominant orientation of the gradient histogram. Then we obtain a $|V^C| \times 6$ matrix $A = [a_1, ..., a_{|V^C|}]^T$, and set $a_G = a_1$. Our aim is to solve the stitching field at each pixel based on the matrix $A$. Let $p_z = (x_z, y_z)$ denote the coordinates of pixel $z$, the stitching field $a_z = a_G + \Delta a_z$ at any pixel $z$ can be obtained from $A$ using a weighted sum of Gaussian functions:

$$\Delta a_z = \sum_{i=1}^{|V^C|} w_i g(\|p_z - p_i\|, \gamma) \tag{8}$$

with $[w_1, ..., w_{|V^C|}]^T = G^+ \Delta A$ and $\Delta A = [\Delta a_1, ..., \Delta a_{|V^C|}]^T$. $\gamma = 1$ and $G^+$ is the pseudo-inverse of $G$ with $G(i,j) = g(\|p_i - p_j\|, \gamma)$. $\|p_i - p_j\|$ denotes the Euclidian distance between pixel $i$ and $j$. For more details please refer to [26].

Based on the stitching field $a_j = a_G + \Delta a_j$, the initial translation $t_j^I$ for pixel $j$ can be calculated by $t_j^I = a_j p_j - p_j$. We further refine $t_j^I$ by optimizing

$$t_j = \arg\min_t (F_j^t(t) + \lambda_1 P_j^t(t, t_j^I)) \tag{9}$$

$F_j(t)$ is the data fidelity term to measure the appearance matching cost at pixel $j$ for translation $t$. It is defined as the distance between the SIFT descriptor at $p_j$ in image $P$ to that located at $p_j + t$ in image $Q$: $F_j^t(t) = \min(\|SIFT_P(p_j) - SIFT_Q(p_j + t)\|_1, \beta)$. We use a truncated $L_1$ norm for SIFT descriptor distance with a threshold $\beta$ for robustness to outliers. $\beta$ constrains that the refined translation for most outliers can not be far away from $t_j^I$. The prior term $P_j^t(t, t_j^I) = \|t - t_j^I\|_1$ regularizes the solution by penalizing large discrepancies from the initial translation $t_j^I$. Here $\lambda_1$ is a constant weight to control the importance of the prior term. Large values bias the refined translation toward $t_j^I$, while small values make the refined translation mainly depend on the appearance fidelity.

So our scene-level dense correspondence method includes two steps: computing the stitching field for each pixel and then refining it by optimizing (9). In the first step, direct pseudo-inverse of $G$ takes $O(|V^C|^3)$ time. We use the low-rank matrix approximation [43] to reduce the computational complexity to be linear of $|V^C|$. In the second step, (9) can be efficiently optimized by searching a small window around $p_j + t_j^I$ [32]. More specifically, a coarse-to-fine two-step scheme is adopted for searching a $25 \times 25$ window. The coarse step searches the $25 \times 25$ window with 5 pixels as the stride and the fine step searches the $5 \times 5$ window with one pixel as the stride. Therefore the time complexity is only $O(50N)$ with $N$ denoting the pixel number. To process an image pair of common size like $480 \times 640$, our method takes about 9 seconds. Feature extraction takes about 2 seconds and feature matching by our DM takes about 6 seconds. So our dense correspondence method takes about 17 seconds in total, compared to state-of-the-art methods: 15 seconds for SIFTflow [35], 6 seconds for DSP [32], 13 minutes for HSVAF [44], 13 minutes for SSID [45],and 25 seconds for DFF [46].

Figure 9 shows the results by our method and two state-of-the-art methods [35] [32]. As can be seen, our method is much more robust to significant geometric transformations than the other methods. The performance on large data sets is often quantitatively measured by the label transfer accuracy [35] [32] which will be given in Section VI.B.

### B. Object-level dense correspondences

Based on the sparse feature matches produced by our DM method, we can solve both the dense correspondence and object segmentation simultaneously. Similar to the above scene-level dense correspondence, we first estimate the initial translation $t_j^I$ for each pixel $j$, and then compute the binary mask $b_P(p_i)$ of image $P$ and $b_Q(p_i)$ of image $Q$ while refining $t_j^I$, where $b(p_i) = 1$ indicates the common object and $b(p_i) = 0$ indicates background at pixel $i$. For each pixel $j$, we optimize

$$\begin{aligned} E_j = \sum_{d=P,Q} \{ & F_j^t(t) + \lambda_1 P_j^t(t, t_j^I) + \lambda_2 F_j^b(b_d(p_j)) \\ & + \lambda_3 P_j^{b1}(b_d(p_j)) + \lambda_4 P_j^{b2}(b_P(p_j), b_Q(p_j + t_j)) \} \end{aligned} \tag{10}$$

$F_j^b(b(p_j)) = -\log h(L_j; b(p_j))$ is the data fidelity term as in [47] to measure the fit of the distribution of $b$ to the pixel value $L_j$ given the histogram model $h$. $P_j^{b1}(b(p_j))$, a prior

term of $b$ to measure compatibility between adjacent pixels in an image, is defined as in [48]:

$$P_j^{b1}(b(p_j)) = \sum_{k \in N_8(j)} [b(p_j) \neq b(p_k)] \exp(-\frac{1}{\sigma_L^2}\|L_j - L_k\|^2) \tag{11}$$

where $N_8(j)$ denotes 8-neighbors of pixel $j$. $[E]$ is the indicator function of $E$ which takes 1 if $E$ holds and 0 otherwise. $P_j^{b2}(b_P(p_j), b_Q(p_j + t_j))$, another prior term of $b$ to measure compatibility between correspondence pixels cross images, is defined as

$$P_j^{b2}(b_P(p_j), b_Q(p_j + t_j)) = [b_P(p_j) \neq b_Q(p_j + t_j)] \times \exp(-\frac{1}{\sigma_{SIFT}^2}\|SIFT_P(p_j) - SIFT_Q(p_j + t_j)\|^2) \tag{12}$$

Note that (10) is non-convex and its global minimum cannot be guaranteed to be obtained. We use the coordinate descent method as [49] which already produces good results in our experiments. More specifically, at each step we optimize for one image by fixing the segmentation mask for the other image. For each image, we alternate between calculating the histogram model $h$ and optimizing (10), which is similar to Grabcut [48]. The alternation is repeated for a few iterations until convergence. The standard deviation $\sigma_L$ of $\|L_j - L_k\|$ in (11) is calculated based on 1000 randomly sampled $\|L_j - L_k\|$ values in the testing data set. $\sigma_{SIFT}$ is calculated similarly. The prior weights $\lambda_1$, $\lambda_2$, $\lambda_3$ and $\lambda_4$ are fixed in this paper.

So our object-level dense correspondence method includes two steps: calculating the histogram model and optimizing (10). The first step takes $O(N)$ time with $N$ denoting the number of pixels [48]. In the second step, the minimum cut algorithm [47] is adopted. Although in the worst case its complexity reaches $O(mN^2)$ with $m$ denoting the number of edges between two pixels, its observed running time is linear of $N$ on many typical problem instances in computer vision [50]. To process an image pair of common size like $480 \times 640$, our method takes less than 15 seconds.

Figure 10 gives the segmentation results for the image in Fig.6. As can be seen, Grabcut [48] fails to segment the object accurately even if an accurate bounding box of the object is given. By using another image in Fig.6 as reference, the state-of-the-art co-segmentation method [51] cannot produce good result due to background clutters, as shown by Fig.9(b). From the warped results shown in Fig.9(c), we can seen that our method is much more accurate, benefiting from the accurate sparse matches (Fig.6(d)) obtained by our DM method.

## VI. Applications

To demonstrate the power of our proposed methods, in this section we show three applications: instance-level image retrieval based on sparse feature matching, mask transfer based on scene-level dense correspondence, and image enhancement based on object-level dense correspondence.

### A. Instance-level image retrieval

Image retrieval has become an important application of sparse feature matching, and can also be used to evaluate the performance of sparse feature matching methods [24], [52], [53]. Our experiment is conducted on the Kentucky database [54] which contains 10200 images for 2550 groups of 4 images each. Similar to [24] [52] [53], we first rank all images by using traditional image retrieval techniques (here we adopt VLAD [55]), and then re-rank the images in the top 100 based on the number of sparse feature matches. The performance measure is the top-4 precision, i.e., the average number of relevant images in the query's top 4 retrieved images as in [54]. We compare our method with the SEA method [24], the NIM method [52], and the 5dof method [53]. For fair comparison, we use the SIFT feature for all the four methods. For each image we obtain about 1200 sparse features. The top-4 precision produced by the VLAD system is 3.31. After re-ranking, the top-4 precisions by our DM, the SEA method, the NIM method and the 5dof method become 3.55, 3.35, 3.42 and 3.36 respectively.

Figure 11 illustrates the reason why our method outperforms the other ones. SIFT feature can tolerate only a small range of affine [14] and significant transformations could sharply

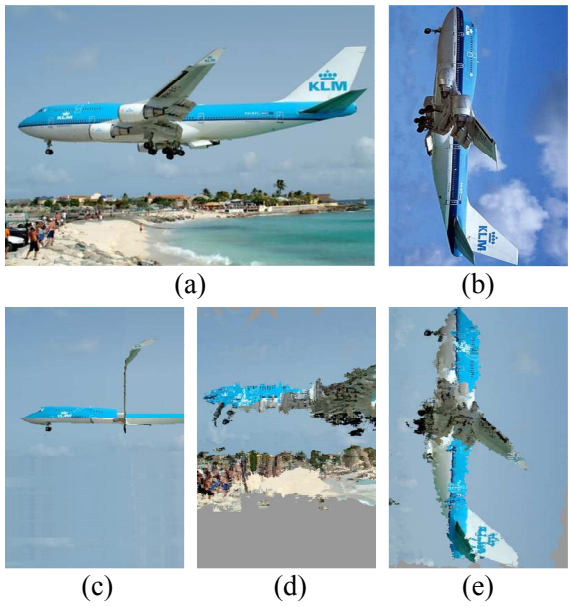



Fig. 9. Scene-level dense correspondence results. (a)The reference image. (b)The target image. (c)The result by SIFT Flow [35]. (d)The result by DSP [32]. (e)Our result.

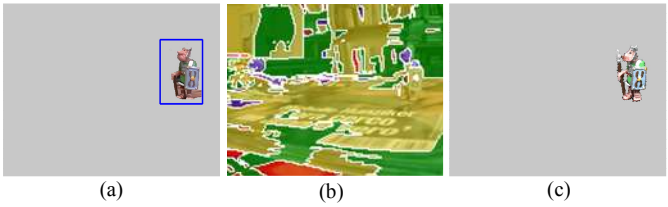



Fig. 10. Object-level dense correspondence for image segmentation of the second image in Fig.6(a). (a)The result by Grabcut [48] with a bounding box. (b)The result by the co-segmentation method [51] taking the first image in Fig.6(a) as the reference image. (c)Our segmentation and warped result by taking the first image in Fig.6(a) as the reference image.

reduce the true matches. So the number of the true matches obtained by SEA, NIM and the 5dof methods based on SIFT feature for such challenging example in Fig.11 are very limited, thus hurting re-ranking performance. In contrast, our DM produces much more true matches because it explores a very large matching space including most true matches. So image re-ranking by our DM is very robust to outlier images.

### *B. Mask transfer*

In this section, we utilize our scene-level dense correspondence method to solve mask transfer. We fix the parameters as $\lambda_1 = 0.005$ and $\beta = 500$ in (9). In the experiment, we randomly pick 5 pairs of images for each object class in the Caltech-101 [56] to obtain 505 pairs of images for testing. For each image we extract about 800 sparse features by using MSER [36] and Harris Affine [37]. We adopted the metric used in [31] [4] to evaluate the performance. For each pixel of the source image, its correspondence is considered correct if it falls within 15 pixels from the ground truth location in the target image. The metric in [31] [4] is the percent of correct matches (PCM) relative to total number of input pixel with matches. We compare our approach with four state-of-the-art methods, SIFTflow [35], Deformable Spatial Pyramid (DSP) [32], SSID [45] and DFF [46], using the authors' publicly available code or the executable.

Benefiting from both the sparse features and our DM matching method, our dense correspondence method is very robust to affine transforms. According to [57], any affine can be decomposed as:

$$\lambda \begin{vmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{vmatrix} \begin{vmatrix} t & 0 \\ 0 & 1 \end{vmatrix} \begin{vmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{vmatrix} + \begin{vmatrix} e \\ f \end{vmatrix}$$

with $t = 1/\cos\theta$. $e$ and $f$ are the translations in image plane, $\phi$ and $\theta$ are the viewpoint angles, $\psi$ parameterizes the camera spin, and $\lambda$ corresponds to the scale. There are 6 parameters in total. Since SIFTflow, DSP, SSID, DFF and ours methods all address translations $e$ and $f$, we need to test only the four parameters $\psi$, $\lambda$, $\theta$ and $\phi$ to show the improvement by our method. We first transform each source image in the testing data set by varying each of the four parameters $\psi$, $\lambda$, $\theta$ and $\phi$, while keeping the target image unchanged. Then we obtain a large dataset including 7070 image pairs. We solve the dense correspondence between each transformed source image and the target image.

Figure 12 shows the percent of correct matches (PCM) as a function of each parameter. We can see that our method is more robust than other methods for all the four parameters. Figure 13 shows the results for four cases $\lambda = 2\sqrt{2}$, $\psi = 90°$, $\theta = 60°$ and $\phi = 60°$. From our results we can see that our method works robustly under significant transforms, non-rigid motions and background clutters.

### *C. Image enhancement*

Over the years, there has been much work on the image enhancement based on a reference example. For a good survey of recent approaches, see [59]. Many methods [58] modify a target image by globally matching the color statistics of a reference image. This kind of methods often fail because even common content between two images may have widely varying appearance, as shown in Fig.14(a) and (b). The NRDC method [34] addresses this problem by solving a mapping function based on image dense correspondences. That method can produce much more pleasing results than the global matching methods when dense correspondences are accurately estimated. However, if the dense correspondences are not accurate enough, the results become far from being satisfying. Figure 14(d) gives the results by the NRDC method for two challenging examples. Since the NRDC method fail to identify the dense correspondences for the image pairs in Fig.14, the target images remain untouched. It solves for dense correspondences by searching for similar patches of a predefined affine set in the reference image. The large transformations shown in Fig.14 are beyond its predefined affine set, thus resulting in the failure results.

We first solve the object-level dense correspondences by using our method. For each image about 2000 sparse features [36] [37] can be detected. We fix the parameters as $\lambda_2 = 2$, $\lambda_3 = 15$ and $\lambda_4 = 1$ in (10). Then we compute the mapping function similar to the NRDC method [34]. From the results in Fig.14(e) we can see that our method is able to accurately produce the dense correspondences and therefore produce much more pleasing results, as shown in Fig.14(f).

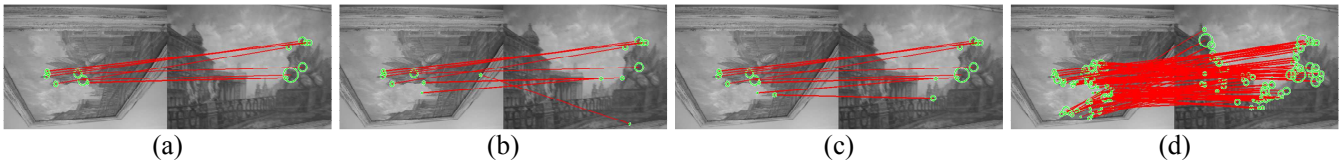
## VII. Discussion of robustness

It is interesting to analyze robustness of our DM method to the extreme cases: highly deformable objects like clothes, repetitive textures, and significant perspective transformations. Figure 15 shows the results for those cases.

From Fig.15(a) we can see that our DM method is very robust to highly deformable objects. This is because non-rigid deformations can be regarded as smoothly varying affine fields [34] [44], and our DM method is robust to a large range of affine transforms. Fig.15(b) demonstrates that repetitive textures might introduce many outliers. Our DM method cannot eliminate those outliers because they are quite similar to the correct matches. Figure 15(c,d) show that significant perspective transforms could make our DM method completely fail. This is because our DM method is based on SIFT matching which is not robust to dramatic perspective transforms [60]. Experiments reveal that our DM method might fail for the perspective transforms: $\theta > 70°$, $\phi > 70°$, $\psi > 120°$ and $\lambda > 4$. Since the sparse features do not work for extreme intra-class variations [35], our DM method cannot match extremely different intra-class objects. These issues will be the topics of our future work.

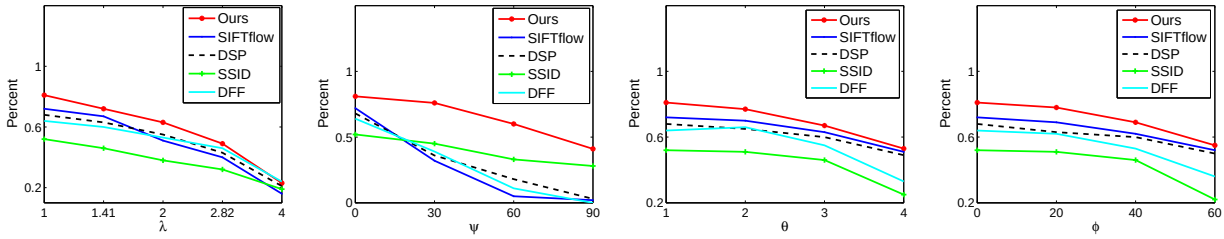
## VIII. Conclusion

We have introduced a unified framework, called Density Maximization, which effectively resolves the three limitations of conventional graph matching methods and achieves impressive performance improvement. We point out that the key to the high performance is twofold: a well-defined local smooth neighborhood to avoid clutter and an iteration scheme to ensure that match updating is mainly based on true matches.



(a) (b) (c) (d)

Fig. 11. The true matches detected by several algorithms for an image pair with significant view point difference. Only several true matches can be estimated by the related work. (a)The result by NIM [52]. (b)The result by 5dof method [53]. (c)The result by SEA [24]. (d) Our DM produces much more true matches than other methods.




(a) (b) (c) (d)

Fig. 12. The dense correspondence results measured by the percent of correct matches for three methods: ours, SIFTflow [31], DSP [32], SSID [45] and DFF [46]. Each of the four parameters $\lambda$, $\psi$, $\theta$ and $\phi$ is varied to test the robustness of those methods. (a) Scale $\lambda$. (b) Rotation $\psi$. (c) Latitude viewpoint $\theta$. (d) Longitude viewpoint $\phi$.

Experiments demonstrate that Density Maximization is adequate for very challenging real-world images which contain many-to-many object correspondences and significant outliers. We have extended our method to dense correspondences, and shown that our method is widely applicable for instance-level image retrieval, mask transfer and image enhancement. We believe that our method may also prove useful for a variety of applications that currently rely on previous sparse feature matching and dense correspondence methods.

## REFERENCES


[1] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty years of graph matching in pattern recognition," *IJPRAI*, pp. 265–298, 2004.

[2] A. Berg, T. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondences," *CVPR*, 2005.

[3] A. Bartoli, M. Perriollat, and S. Chambon, "Generalized thin-plate spline warps," *IJCV*, vol. 88, no. 1, pp. 85–110, 2010.

[4] O. Chum and J. Matas, "Matching with prosac - progressive sample consensus," *CVPR*, 2005.

[5] M. Cho, J. Lee, and K. M. Lee, "Reweighted random walks for graph matching," *ECCV*, 2010.

[6] T. Cour, P. Srinivasan, and J. Shi, "Balanced graph matching," *NIPS*, 2007.

[7] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," *ICCV*, 2005.

[8] M. Leordeanu and M. Herbert., "An integer projected fixed point method for graph matching and map inference," *NIPS*, 2009.

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

[10] J. Maciel and J. Costeira, "A global solution to sparse correspondence problems," *TPAMI*, vol. 25, no. 2, pp. 187–199, 2003.

[11] M. Gori, M. Maggini, and L. Sarti, "Exact and approximate graph matching using random walks," *TPAMI*, vol. 27, no. 7, pp. 1100–1111, 2005.

[12] Y. Tian, J. Yan, H. Zhang, Y. Zhang, X. Yang, and H. Zha, "On the convergence of graph matching graduated assignment revisited," *ECCV*, 2012.

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

[14] D. G. Lowe, "Object recognition from local scale-invariant features," *ICCV*, 1999.

[15] O. Duchenne, F. Bach, I. Kweon, and J. Ponce, "A tensor-based algorithm for high-order graph matching," *CVPR*, 2009.

[16] J. Lee, M. Cho, and K. M. Lee, "Hyper-graph matching via reweighted random walks," *CVPR*, 2011.

[17] R. Zass and A. Shashua, "Probabilistic graph and hypergraph matching," *CVPR*, 2008.

[18] F. Zhou and F. Torre, "Deformable graph matching," *CVPR*, 2013.

[19] H. Li, E. Kim, X. Huang, and L. He, "Object matching with a locally affine-invariant constraint," *CVPR*, 2010.

[20] M. Cho, J. Sun, O. Duchenne, and J. Ponce, "Finding matches in a haystack: A max-pooling strategy for graph matching in the presence of outliers," *CVPR*, 2014.

[21] M. Cho, J. Lee, and K. M. Lee, "Feature correspondence and deformable object matching via agglomerative correspondence clustering," *ICCV*, 2009.

[22] M. Cho and K. M. Lee, "Authority-shift clustering: Hierarchical clustering by authority seeking on graphs," *CVPR*, 2010.

[23] W. Zhang, X. Wang, D. Zhao, and X. Tang, "Graph degree linkage: agglomerative clustering on a directed graph," *ECCV*, 2012.

[24] H. Liu, L. J. Latecki, and S. Yan, "Fast detection of dense subgraph with iterative shrinking and expansion," *TPAMI*, 2013.

[25] S. Jouili, S. Tabbone, and V. Lacroix, "Median graph shift: A new clustering algorithm for graph domain," *ICPR*, 2010.

[26] W. Lin, S. Liu, Y. Matsushita, and T. Ng, "Smoothly varying affine stitching," *CVPR*, 2011.

[27] R. Szeliski, "Image alignment and stiching: A tutorial," *Foundations and Trends in Computer Graphics and Computer Vision*, vol. 2, no. 1, pp. 1–10, 2006.

[28] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International Journal of Computer Vision (IJCV)*, vol. 47, no. 1, pp. 7–42, 2002.

[29] A. Bruhn, J. Weickert, and C. Schnorr, "Lucas/kanade meets horn/schunk: combining local and global optical flow methods," *International Journal of Computer Vision (IJCV)*, vol. 3, no. 61, pp. 211–231, 2005.

[30] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "Deepflow: Large displacement optical flow with deep matching," *ICCV*, 2013.

[31] M. Leordeanu, A. Zanfir, and C. Sminchisescu, "Locally affine sparse-to-dense matching for motion and occlusion estimation," *ICCV*, 2013.

[32] J. Kim, C. Liu, F. Sha, and K. Grauman, "Deformable spatial pyramid matching for fast dense correspondences," *CVPR*, 2013.

[33] C. Barnes, E. Shechtman, D. Goldman, and A. Finkelstein, "The generalized patchmatch correspondence algorithm," *ECCV*, 2010.

[34] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski, "Non-

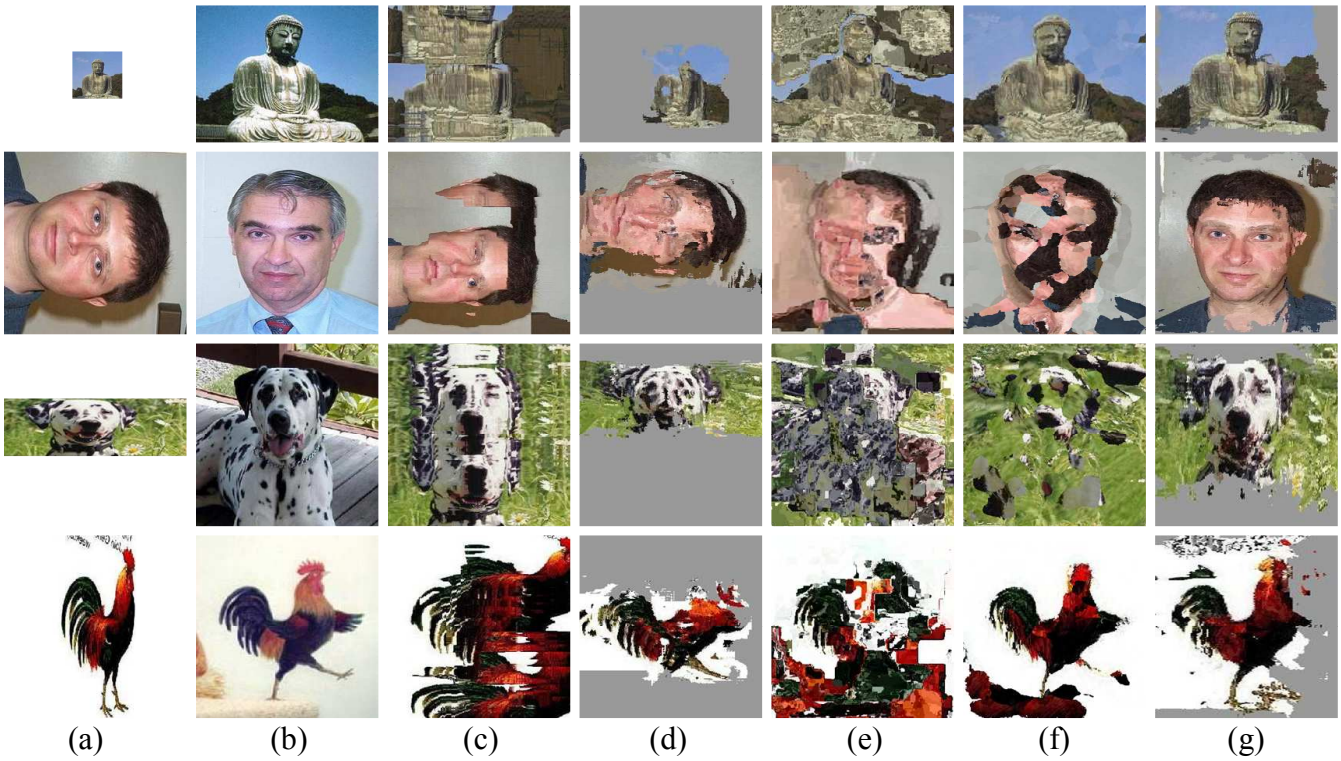



Fig. 13. (a)The source images. From top to bottom are the four challenging cases: $\lambda = 2\sqrt{2}$, $\psi = 90°$, $\theta = 60°$, and $\phi = 60°$. (b)The target images. (c)The results by SIFTflow [31]. (d)The results by DSP [32]. (e) The results by SSID [45]. (f) The results by DFF [46]. (g)Our results.


Fig. 14. (a)The reference images. (b)The target images. (c)The result by the globally matching method [58]. (d)The result by the NRDC method [34]. (e)Our segmentation and warped result. (f)Our enhancement result.

rigid dense correspondence with applications for image enhancement," *SIGGRAPH*, 2011.
[35] C. Liu, J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across different scenes and its applications," *TPAMI*, vol. 33, no. 5, pp. 978–994, 2011.
[36] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," *BMVC*, 2002.
[37] K. Mikolajczyk and C. Schmid, "Scale and affine invariant interest point detectors," *IJCV*, 2004.
[38] C. Wang, L. Wang, and L. Liu, "Improving graph matching via density maximization," *ICCV*, 2013.
[39] M. Cho and K. M. Lee, "Mode-seeking on graphs via randomwalks," *CVPR*, 2012.
[40] Y. A. Sheikh, E. A. Khan, and T. Kanade, "Mode-seeking by medoid shifts," *ICCV*, 2007.
[41] V. Ferrari, T. Tuytelaars, and L. V. Gool, "Simultaneous object recognition and segmentation from single or multiple model views," *IJCV*, vol. 67, no. 2, pp. 159–188, 2006.
[42] M. Cho, Y. M. Shin, and K. M. Lee, "Co-recognition of image pairs by data-driven monte carlo image exploration," *ECCV*, 2008.
[43] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *TPAMI*, vol. 32, no. 12, pp. 2262–2276, 2010.
[44] W. Lin, L. Liu, Y. Matsushita, and K. Low, "Aligning images in the wild," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
[45] E. Trulls, I. Kokkinos, A. Sanfeliu, and F. Noguer, "Dense segmentation-aware descriptors," *CVPR*, 2013.
[46] H. Yang, W. Lin, and J. Lu, "Daisy filter flow: A generalized discrete approach to dense correspondences," *CVPR*, 2014.
[47] Y. Boykov and M. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images," *CVPR*, 2001.
[48] C. R. V. Kolmogorov and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *SIGGRAPH*, 2004.
[49] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu, "Unsupervised joint object

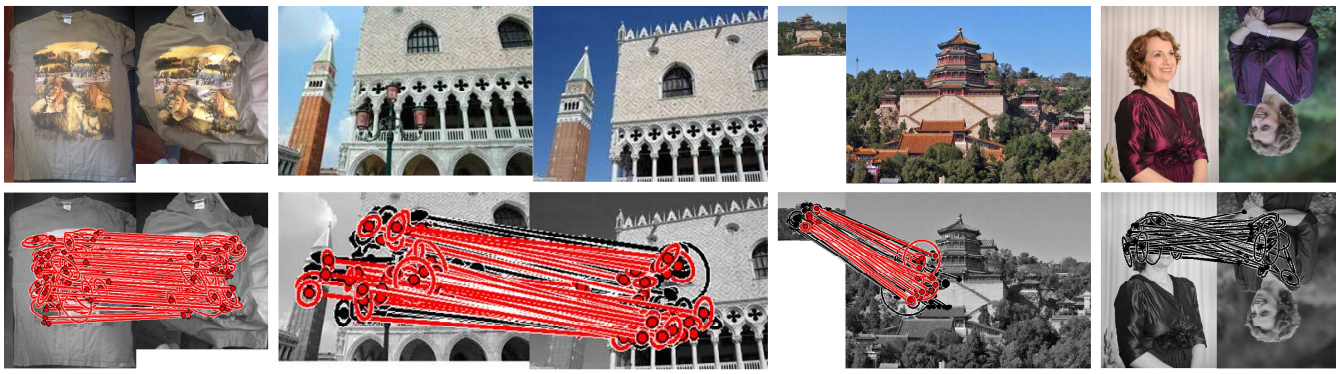
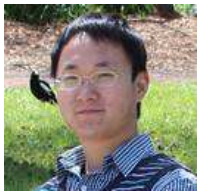



Fig. 15. The top row of images are input image pairs and the bottom images are our matching results. Red lines denote true matches and black lines denote outliers. (a)A highly deformable non-rigid object pair. 315 true matches with 15 outliers are detected. (b)A image pair with repetitive textures. 48 true matches with 205 outliers are detected. (c)A image pair with scale $\lambda$ larger than four. 29 true matches with 165 outliers are detected. (d)A image pair with $\psi = 180°$. No true matches are detected.

discovery and segmentation in internet images," *CVPR*, 2013.
[50] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *TPAMI*, vol. 26, no. 9, pp. 1124–1137, 2004.
[51] A. Joulin, F. Bach, and J. Ponce, "Discriminative clustering for image co-segmentation," *CVPR*, 2010.
[52] J. Zhu, S. Hoi, M. Lyu, and S. Yan, "Near-duplicate keyframe retrieval by nonrigid image matching," *ACM Multimedia*, 2008.
[53] J. Philbin1, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," *CVPR*, 2007.
[54] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," *CVPR*, 2006.
[55] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," *CVPR*, 2010.
[56] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *TPAMI*, vol. 28, no. 4, pp. 594–611, 2006.
[57] J.-M. Morel and G. Yu, "Asift: A new framework for fully affine invariant image comparison," *SIAM Journal on Imaging Sciences archive*, vol. 2, no. 2, pp. 438–469, 2009.
[58] F. Pitie, A. Kokaram, and R. Dahyot, "Automated colour grading using colour distribution transfer," *Computer Vision Image Understanding*, vol. 107, no. 5, pp. 123–137, 2007.
[59] X. An and F. Pellacini, "User-controllable color transfer," *Computer Graphics Forum*, vol. 29, no. 2, pp. 263–271, 2010.
[60] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Gool, "A comparison of affine region detectors," *IJCV*, vol. 65, no. 2, pp. 43–72, 2005.

Dr. Chao Wang received the B.S. degree from Harbin Institute of Technology of China in 2001, the M.S. degree from Nanjing University of China in 2004, and the Ph.D. degree from Tsinghua University of China, in 2010. He joined the school of computer science and software engineering at the University of Wollongong in August 2012. Before that, he worked in the Research Center for Computer Graphics and Visualization (CCGV) at the Department of Computing and Information Systems, University of Bedfordshire from 2010 to 2012. His research interests are in the areas of Computer Vision, Computer Graphics and Machine Learning. In particular he works on image retrieval, image restoration and enhancement and graph matching.

Dr. Lei Wang received the B.Eng and M.Eng from Southeast University, China in 1996 and 1999, respectively, and the Ph.D. from School of EEE in Nanyang Technological University, Singapore in 2004. He worked as research associate and research fellow in Nanyang Technological University from 2003 to 2005. After that, he joined the Department of Information Engineering, RSISE, The Australian National University as research fellow. In Jan 2007, he was awarded the Australian Postdoctoral Fellowship by the Australian Research Council and worked as APD research fellow from 2007 to 2009. In May 2009, he was awarded the Early Career Researcher Award by Australian Academy of Science and Australian Research Council. From Jan 2010, he works as Fellow in School of Engineering of the College of Engineering and Computer Science. Now he is with Faculty of Informatics of University of Wollongong as Associate Professor.



Dr. Lingqiao Liu (Graduated in Oct 2013) obtained his Bachelor degree and Masters degree from University of Electronic Science and Technology of China (UESTC) in 2006 and 2009, respectively, and the Ph.D. from Australian National University in 2012. Currently he is a research fellow of the School of Computer Science, University of Adelaide, Australia. His research interest includes computer vision and machine learning.