Int J Comput Vis (2008) 78: 47–66
DOI 10.1007/s11263-007-0077-5

Region-Based Hierarchical Image Matching

Sinisa Todorovic · Narendra Ahuja

Received: 16 September 2006 / Accepted: 17 July 2007 / Published online: 12 September 2007

Abstract This paper presents an approach to region-based hierarchical image matching, where, given two images, the goal is to identify the largest part in image 1 and its match in image 2 having the maximum similarity measure defined in terms of geometric and photometric properties of regions (e.g., area, boundary shape, and color), as well as region topology (e.g., recursive embedding of regions). To this end, each image is represented by a tree of recursively embedded regions, obtained by a multiscale segmentation algorithm. This allows us to pose image matching as the tree matching problem. To overcome imaging noise, one-to-one, many-to-one, and many-to-many node correspondences are allowed. The trees are first augmented with new nodes generated by merging adjacent sibling nodes, which produces directed acyclic graphs (DAGs). Then, transitive closures of the DAGs are constructed, and the tree matching problem reformulated as finding a bijection between the two transitive closures on DAGs, while preserving the connectivity and ancestor-descendant relationships of the original trees. The proposed approach is validated on real images showing similar objects, captured under different types of noise, including differences in lighting conditions, scales, or viewpoints, amidst limited occlusion and clutter.

Keywords Image matching · Edit-distance graph matching · Many-to-many matching · Maximum subtree isomorphism · Segmentation trees · Transitive closures · Association graphs · Maximum weight cliques

S. Todorovic (✉) · N. Ahuja
Computer Vision and Robotics Laboratory, Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana-Champaign, 405 N. Mathews Ave., Urbana, IL 61801, USA
e-mail: sintod@vision.ai.uiuc.edu

N. Ahuja
e-mail: ahuja@vision.ai.uiuc.edu

1 Introduction

Image matching is a long-standing problem in computer vision. While most approaches use point features and/or curve fragments for image matching, there is also a significant amount of work on region-based matching. For example, matching using region properties is done to address problems from stereo matching (e.g., Medioni and Nevatia 1985; Cohen et al. 1989a; Randriamasy and Gagalowicz 1991), and motion/optical-flow analysis (e.g., Xuguang and Ramchandran 1999; Ming-Hsuan et al. 2002; Fuh and Maragos 1989) to object recognition (e.g., Basri and Jacobs 1997; Keselman and Dickinson 2005). This work demonstrates several advantages of using regions over interest points or edge fragments for image matching. The higher dimensionality of regions makes them richer descriptors of target objects' geometric properties, such as size and shape, and photometric properties such as gray-level contrast. The higher dimensional character of regions also makes their matching more stable to small illumination and viewpoint changes across given images. This work, however, uses only region geometry and appearance for matching. The motivation behind the work reported in this paper is to extend the matching criteria to include hierarchical region properties, which should improve the robustness of image matching. Specifically, this paper proposes a robust solution to the following problem: given two images, image 1 and image 2, find in image 2 a match for each region in image 1 such that the matched regions have similar geometric and photometric properties, the same holds recursively for their subregions, and the set of matches maximize a combination of similarity and total image area covered by the matched regions.

Similarity between regions is expressed in terms of intrinsic geometric and photometric properties (e.g., region area, boundary shape, and color), as well as and the proposed extension involving topological properties (e.g., region layout, and the number of subregions, and the relationships between the properties of the subregions and the regions). To compactly capture these requirements of the proposed matching algorithm, each image is represented as a segmentation tree. Each node in the tree denotes an image segment whose contrast with the surround is significantly larger than its interior variability. The root of the tree represents the entire image, and the children of any node represent subsegments contained within the parent segment. The leaf nodes correspond to segments that are truly uniform in brightness, or have an inhomogeneity level that is acceptable. Successive levels of the tree thus capture smaller details completely contained within the parent regions, and subtrees correspond to subimages. With this representation at hand, our matching problem becomes a graph matching problem, namely, finding all similar subtrees across the two image trees.

In real life, however, due to lighting, viewpoint, or scale variations successive images of the same scene do not always yield the same tree structure. For instance, a region in image 1 may often be segmented into two or more smaller, low-mutual-contrast regions in image 2 due to a slightly different illumination direction. Similarly, the reverse may happen and two regions may merge. While such minor changes change the image appearance only slightly, they will significantly alter the tree topologies. Our solution must thus allow region matching despite the significant differences in the associated tree structure. We present a tree matching algorithm that addresses this need.

We conduct tree matching by explicitly accounting for many-to-many, one-to-many, and one-to-one node correspondences at the same time. To this end, the segmentation trees of a given image pair are modified by inserting and appropriately connecting new nodes, referred to as mergers. Each merger node is the union of a few children of a node. It instantiates the hypothesis that the children were incorrectly formed due to, for example, lighting changes etc., and therefore their union should be restored as a separate node. To cover all possibilities, mergers correspond to all members of the power set of *adjacent* children under each node. The adjacent sibling nodes that merged are called the source nodes. Mergers do not eliminate their source nodes in a given tree. Instead, each merger is inserted as a sibling of its source nodes (i.e., a merger is connected to its sources' parent), and inherits the children of its source nodes. Thus, the addition of mergers converts the tree into a directed acyclic graph (DAG). Then, the transitive closure of the DAG is constructed by adding new edges between each node and its descendants in the DAG. The reason for constructing transitive closures is that their matching is more flexible than matching DAGs, allowing matches of all descendants under a visited node. The tree matching algorithm is then presented as a graph matching algorithm that finds a bijection between nodes of the two transitive closures. The bijection must satisfy the following consistency constraints whose reasons are obvious: (1) each merger is disallowed to match if its source nodes get matched, and (2) matching transitive closures of DAGs should preserve the original connectivity, and ancestor-descendant relations of the segmentation trees. These consistency constraints disallow many node-pairs from being candidates for matching, thereby eliminating the inefficiency that would otherwise result from our formulation of the tree matching problem in terms of the more general problem of graph matching.

1.1 Literature Review and Relationship to Previous Work

Image matching by using graph abstractions has been the focus of sustained research activity in computer vision for more than two decades now. The two key issues investigated in the graph-matching literature are how to measure the similarity of two graphs in the presence of structural noise, and how to search efficiently for the best match. Early work includes Barrow and Burstall's idea (Barrow and Burstall 1976) to match two graphs by searching for the maximum common subgraph. This problem can be reformulated as finding a maximum clique of the association graph, whose vertices represent all pairs of nodes in the two original graphs. Pelillo et al. (1999) use their idea to solve the problem of attributed tree matching. Their method finds a match by specifying an association graph whose maximum weighted clique is in one-to-one correspondence with the maximum common subtree. Although the maximum clique problem is known to be NP-hard, powerful heuristics and theoretical results exist that provide for good approximate solutions (Pardalos and Xue 1994). Here, the most relevant is the Motzkin-Straus theorem (Motzkin and Straus 1965), which allows us to transform the maximum clique problem into a continuous quadratic programming problem that can be efficiently solved, for example, by replicator dynamics (Pelillo 1999; Bomze et al. 2000; Pelillo 2002).

A different approach to graph matching uses the spectral graph theory. The structural properties of graphs are captured by the eigenvectors of the adjacency matrix, or the Laplacian matrix. Examples include Umeyama's idea (Umeyama 1988) to perform singular value decomposition on the adjacency matrices of two same-size graphs, and a more recent work of Shokoufandeh et al. (2005), where similar graph structures are efficiently retrieved from a database of graphs by exploiting the upper bounds on the stability of graphs' spectra with respect to minor topological changes. While recent work on spectral graph matching puts a significant effort to improve the resilience of

these methods to structural noise (Fowlkes et al. 2004; Richard et al. 2005), there are two major fundamental weaknesses of the spectral representations: (1) structurally different graphs may have the same spectrum, and (2) noise added by a few spurious nodes or edges in a graph may significantly change its spectral representation.

Another group of approaches to error-tolerant graph matching involves directly minimizing the differences between the graph structures. Specifically, these approaches minimize a measure of difference called the graph edit-distance introduced by Bunke and Allermann (1983), and independently by Fu and his collaborators (Tsai and Fu 1979; Sanfeliu and Fu 1983; Eshera and Fu 1986). Edit-distance associates a cost with basic edit operations on nodes and edges—such as insertion, deletion, merging, splitting and relabeling—the sequence of which would make the two graphs isomorphic. By minimizing the cost of modifications needed in the two graphs to match them, it is possible to compute a measure of similarity between them. However, finding an optimal sequence of edit operations that minimizes the specified cost is in most cases computationally prohibitive. Recently, Bunke and Kandel (2000) have established the relationship between the size of the maximum common subgraph and the edit-distance. They have shown that under certain constraints on edit-costs—namely, that deletions and re-insertions of nodes and edges are not more expensive than the corresponding node and edge relabeling—computing the maximum common subgraph and the minimum-cost edit-distance are computationally equivalent. This is an important result, because under the given constraints one can optimize the sequence of edit operations only via insertions and deletions instead of using a larger set of edits, thus reducing computational complexity.

Recent work considers improving the robustness of edit-distance based matching to the amount of structural noise. In particular, Sebastian et al. (2004) have developed a variational method to measure the distance between two graphs as the minimum extent of deformation necessary for one graph to match the other. To extract shape skeletons of objects in video, Golland et al. (2000) have proposed to minimize a suitably defined cost of small perturbations of graph nodes. These approaches are relatively robust to deformations in graph structure; however, while the former requires that the two graphs are initially aligned in order to be matched, the latter is computationally very expensive, which significantly narrows the domain of their application. For the case of matching trees, Torsello and Hancock (2002, 2003) propose that instead of matching the original trees, one should match their transitive closures. In transitive closures of trees, the original set of tree nodes remains intact, whereas new edges are added between every tree node and its descendants. The transitive closures allow that a search for a maximally matching node pair is conducted over all descendants under a visited ancestor node pair, rather than stopping the search if the ancestors' children do not match. This makes matching more flexible and robust. Also, Torsello and Hancock (2002) redefine their tree-matching formulation so that it could be applied to the tree-to-DAG matching problem; however, they only informally state that the theoretical findings previously proved for their tree matching approach generalize to this new setting. Our approach is most closely related to this work. The main differences are that they augment only the initial set of edges, while we augment both the set of edges and set of nodes to form transitive closures on trees or DAGs, and that they assume only one-to-one node correspondences, while we account for one-to-one, many-to-one, and many-to-many correspondences, all at the same time.

The assumption of one-to-one node correspondences in matching is usually too restrictive. Recently, several many-to-many and many-to-one graph-matching approaches have been proposed to address this problem. For example, Keselman et al. (2003) and Demirci et al. (2004) propose to embed a tree into a normed space by partitioning the tree into N mutually exclusive paths, corresponding to its N leaf nodes. These N paths can be viewed as defining an N-dimensional normed space, in which every tree node represents a vector with entries corresponding to the paths necessary to traverse to reach the node from the root. To embed two trees with a different total number of leaf nodes into a unique normed space, they use the PCA. This transforms tree matching to the problem of many-to-many matching of points in a low-dimensional normed space, for which they use the Earth Mover's Distance algorithm (Rubner et al. 1998; Cohen and Guibas 1999). Such embedding, however, may produce different normed spaces for the same tree, since the choice of distinct paths in the tree is not unique. Liu and Geiger (1999) have studied many-to-many graph matching in the context of edit-distance. Due to the high computational cost of searching for the minimum-cost edit-sequence, they use a sub-optimal, heuristic A* algorithm. Sebastian et al. (2004) have also considered one-to-many node correspondences. In their approach, matching a single node with another single node is favored, while matching a single node with many is penalized with a high cost, which may not be justified in many cases.

Relevant to our approach is many-to-many tree matching using the notion of ϵ-morphism proposed by Pelillo et al. (2001). They merge n-tuplets of nodes, replacing them with a single node, referred to as merger, and try to match the combined node to another merger in the other tree. Merging of nodes is allowed only if their weights differ by less than ϵ. Since our approach also merges nodes, it is worthwhile to note the differences: (1) they allow merging among only those nodes that satisfy the threshold constraint,

(2) their mergers replace the source nodes, whereas ours augment them, (3) their mergers are formed from all possible nodes along the ascendant-descendant path, whereas ours are formed only among sibling nodes representing contiguous image regions, and (4) their merging transforms the original tree into another tree, whereas ours, into a directed graph.

There is very limited past work on matching segmentation hierarchies in which the recursive containment of regions is explicitly used as a cue for matching. For the most part, existing methods use greedy approaches, where, for example, matching is done top-down only between regions at the same tree level, such that a bad match between two regions penalizes attempts to match their respective descendants (Cohen et al. 1989b; Perrin et al. 1998). In Glantz et al. (2004), a sequence of planar graphs, representing multiscale image segmentations, is defined as segmentation hierarchy, and ten topological relationships between any two regions v_1 and v_2 in the segmentation hierarchy are defined as a combination of the following basic relationships: (i) v_1 may be adjacent to v_2, (ii) v_1 may enclose but not contain v_2, (iii) v_1 may contain v_2, and (iv) v_1 and v_2 may be apart. Then, given two segmentation hierarchies, their matching is formulated as a one-to-one mapping between regions of one hierarchy and regions of the other, which preserves the ten topological properties associated with each region. In case two regions from the two hierarchies differ in a single topological property (e.g., adjacency) their pair is discarded, which yields poor matching results for two images with variations arising, for example, from partial occlusion of target objects. Also, as other one-to-one matching approaches, this method suffers from low robustness to small topological changes in graph structure.

1.2 Contributions

The main criticism that can be leveled at the existing many-to-many, or many-to-one image matching approaches is that image graphs are treated as general data structures, disregarding the fact that nodes represent structures in images. For example, thresholding methods, which involve node merging, specify a heuristic, single, global threshold applied to the whole graph structure. In real images, mergers are needed to handle the loss of borders between regions, e.g., due to illumination changes, and therefore, the optimal threshold for merging nodes varies across the image. Further, the magnitude of node attribute disparities in two images to be addressed by mergers is a priori unknown. Consequently, these approaches are very sensitive to threshold selection. An arbitrarily small threshold on the allowed disparity may result in a wasted effort when merging is attempted, while large threshold values may lead to excessive merging, and, hence, to an unjustified loss of structural information. Moreover, the node merging methods are primarily concerned with graph transforms for forming mergers, such that the resulting matching would be less sensitive to structural noise, without accounting for the plausibility of these transforms in the image space or for the costs of the corresponding region mergers. Another characteristic of the existing methods is that they are not capable of considering one-to-one, many-to-one, and many-to-many node correspondences at the same time.

Our approach allows enforcement of image constraints and one-to-one, many-to-one, and many-to-many node correspondences, at the same time, in matching two similar image trees that contain structural noise, which may be local (e.g., a few nodes/edges missing) or global (e.g., children nodes of a node in one tree become great-grandchildren of that node in the other tree). We augment the image segmentation tree with mergers representing the power set of only those nodes that represent neighboring image regions. This addition of mergers transforms the segmentation tree into a graph. The extra nodes present in the graph represent the redundancy needed to combat realistic noise, since only adjacent regions are plausible candidates for mergers. Image matching is formulated as graph matching, subject to the consistency constraints that node ascendant-descendant relations of the original segmentation trees must be preserved, and matching of the mergers must be disallowed in case their source nodes already got matched.

Recall that Torsello and Hancock (2002, 2003) have showed that, given two trees, better performance is obtained by matching their transitive closures than the original trees. We advance their idea by matching the transitive closures on directed graphs, obtained by augmenting the original trees with mergers. In addition, our graph matching is subject to the aforementioned consistency constraints. Hence, we generalize their work, by giving the necessary and sufficient conditions for finding a sequence of edit-operations, which induces a consistent subtree isomorphism between the two transitive closures on DAGs.

For experimental validation we match images in a set to find those containing similar objects. We use a carefully prepared database consisting of 700 natural-scene images showing similar objects, including flowers, animals, musical instruments and buildings. By choosing similar resolution, viewpoints, illumination and other imaging conditions, we ensure that object occurrences consist of image regions that have similar low-level properties, such as color, size, and shape across the database. At the same time, we also ensure that image regions occupied by a specific object type also exhibit significant low-level differences that are expected to be encountered in real life. To this end, we allow flowers, animals, musical instruments to vary by selecting them from the same class instead of being exact replications of the same

objects. This results in low-level variations, for example, in color by selecting horses of different color, or in size and shape of different sunflowers. Another cause of low-level variations in the database comes from a certain degree of occlusion present between objects and clutter. Also, there exist pairs of images that have significantly different lighting conditions. These variations result in geometric and photometric differences between regions that should be matched, and thus test the robustness of our matching criteria and algorithm. The matching performance is measured through (1) appropriately defined pixel- and region-matching errors, and (2) clustering the database images into sets containing similar objects. Such validation on real images offers more challenges than synthetic data (e.g., in Richard et al. 2005), binary images of object silhouettes (e.g., in Pelillo et al. 1999; Torsello and Hancock 2002, 2003), or simple scenes of a single object in front of a uniform background, obtained in a lab-controlled environment (e.g., in Keselman and Dickinson 2005). Our experimental results show that accounting for one-to-one, many-to-one, and many-to-many node correspondences at the same time, and a local regulation of image graph transformation do indeed yield successful matching performance on real images, and demonstrate the advantages of our approach over prior work.

This paper is organized as follows. The multiscale image segmentation algorithm, used in this paper for representing images as trees, is reviewed in Sect. 2. The image matching problem is formulated within the graph-theoretic framework in Sect. 3. Section 4 presents our divide-and-conquer algorithm for matching transitive closures on DAGs. Section 5 is concerned with the computational complexity of our approach. The experimental validation is reported in Sect. 7, while our conclusions and final remarks are presented in Sect. 8.

2 Multiscale Image Segmentation

In this paper, images are represented by trees obtained by a multiscale segmentation algorithm discussed in Ahuja (1996), Tabb and Ahuja (1997), Arora and Ahuja (2006). The segmentation algorithm partitions an image into homogeneous regions of a priori unknown shape, size, gray-level contrast, and topological context. Here, a region is considered to be homogeneous if variations in intensity within the region are smaller than intensity change across its boundary, regardless of its absolute degree of variability. Consequently, image segmentation is performed at a range of homogeneity values, i.e., intensity contrasts, referred to as photometric scales. A segment at any photometric scale may be recursively segmented to extract finer scale segments, characterized by increasing degree of homogeneity, until one obtains strictly constant intensity regions. As a result, this process yields a multiscale segmentation of the image.

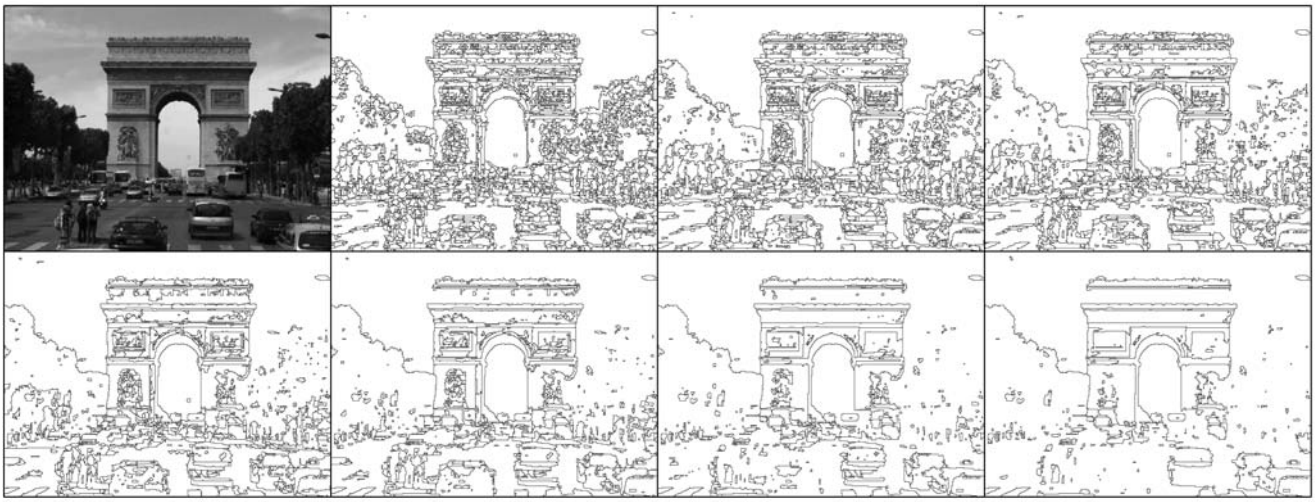
Specifically, the algorithm models the segmentation process in physical terms by treating image pixels as charged particles attracted to other pixels (Ahuja 1996). The force of attraction of a particle to another particle is proportional to the similarity of their gray levels and closeness of their locations. The net force of attraction exerted on a pixel thus tends to be in the direction in which similar pixels lie close by. Pixels belonging to a region form a smoothly varying, convergent, force flow field, which exhibits a force divergence across the region border. Since the photometric scale parameters of image regions are unknown a priori, all the regions can be detected by varying exhaustively the photometric scale across the entire range (e.g., gray-level intensity 1–255), as illustrated in Fig. 1a. As the photometric scale varies, the force signatures of regions emerge when the scale parameters approach the values defining the regions. As the photometric scale increases, regions with smaller gray-level contrasts than the current scale strictly merge. A sweep of the scale parameter values thus results in the extraction of all the segments present in the image.

The segmentation tree is then derived by organizing the segmented regions into a tree structure with respect to their size and location in the image. In the segmentation tree, the root represents the whole image, nodes at upper levels, closer to the root, represent large regions, while their children nodes capture smaller details completely contained within the corresponding parent region, as depicted in Fig. 1b. The number of tree levels and the homogeneity values associated with regions are dynamically determined by the algorithm for each image at hand. Thus, for example, the segmentation tree may be highly unbalanced, with leaf nodes occurring at any level, and with each node having arbitrary many children, as dictated by the image topology. In the sequel, we will use the notion of adjacent sibling regions. Region v_1 and region v_2 having the same parent region u in the segmentation tree are called adjacent siblings if the distance, $d_{v_1 v_2}$, between the two closest points on the boundaries of regions v_1 and v_2 is smaller than 5% of the square-root of u's area, $d_{v_1 v_2} < 0.05 \cdot \sqrt{\text{area}(u)}$.

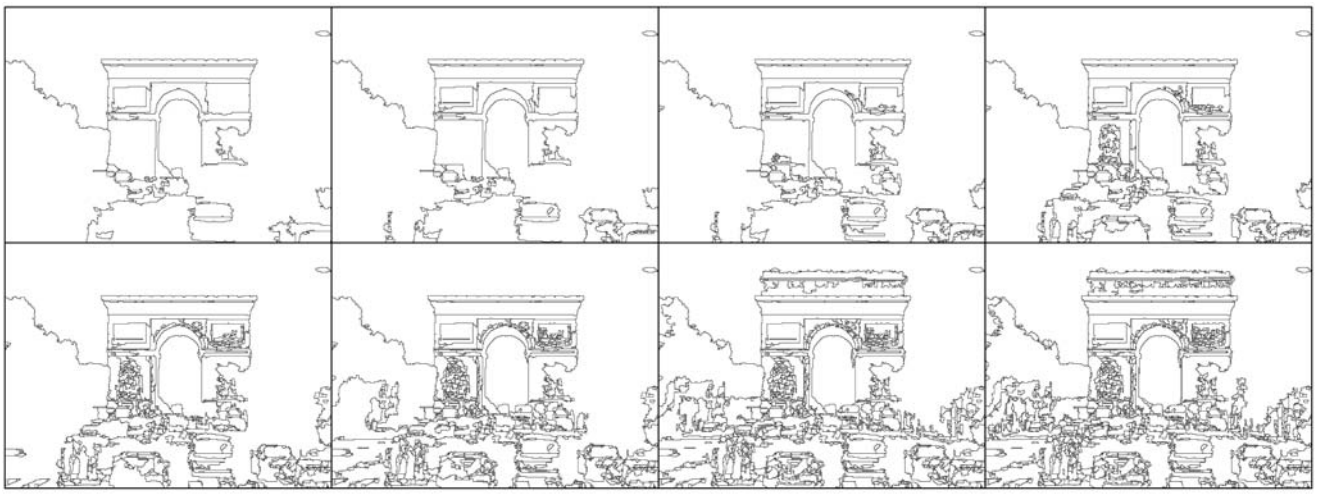
Each node of the segmentation tree is characterized by the intrinsic geometric and photometric properties of the corresponding region. The geometric properties may include the area and shape of the region, while the photometric properties may encompass the gray level mean and variance of the region. The choice of region properties is dictated by the underlying application for which the proposed region-based image matching is used.

3 Tree Matching

In this section, our objective is to formulate image matching as the tree matching problem. To this end, below, we first present necessary graph-theoretic concepts.



(a) Original image and example segmentations for photometric scales: 4, 6, 8, 10, 12, 16, 20 (in a raster scan).



(b) Segmentation tree: projections of nodes at tree levels 2, 3, 4, 5, 6, 7, 8, 9 out of 16 tree levels onto the image plane (in a raster scan), where the root is at level 1.

Fig. 1 Images are represented by trees obtained from the multiscale segmentation algorithm (Ahuja 1996; Tabb and Ahuja 1997; Arora and Ahuja 2006)

3.1 Definitions

Let $G = (V_G, E_G, \psi_G)$ denote an attributed, directed graph, where V_G is the set of vertices (nodes), $E_G \subseteq V_G \times V_G$ is the set of directed edges, and $\psi_G : V_G \to \Psi$ is a function that assigns an attribute vector $\boldsymbol{\psi}_v \in \Psi$ (e.g., geometric and photometric properties of the corresponding region) to each node $v \in V_G$.[1] For any pair of nodes $u, v \in V_G$ connected by an edge, denoted as $u \sim v$, u is said to be the *parent* of v, and v is said to be the *child* of u. Children of the same parent are called *siblings*. A directed *path* between nodes $v_1, v_n \in V_G$, denoted as $v_1 \rightsquigarrow v_n$, is any sequence of distinct nodes $v_1 v_2 \ldots v_n \in V_G$, such that $v_i \sim v_{i+1}$, for $i = 1, \ldots, n-1$. If $v_1 = v_n$ the path is called a cycle. A rooted tree, $T = (V_T, E_T, \psi_T)$, is an attributed, connected, directed graph with no cycles, in which each node has one and only one parent, except for the one node called *root* that has no parents. *Connectivity* means that for each pair of nodes in T there is an undirected path between them. The *induced subgraph* $G[U]$ of G is a graph with $U \subseteq V_G$ as its node set, and where $u \sim v$ in $G[U]$ if and only if $u \sim v$ in G.

Let $G_1 = (V_{G_1}, E_{G_1}, \psi_{G_1})$ and $G_2 = (V_{G_2}, E_{G_2}, \psi_{G_2})$ be two directed acyclic graphs (DAGs). Any bijection $f : U_{G_1} \to U_{G_2}$, where $U_{G_1} \subseteq V_{G_1}$ and $U_{G_2} \subseteq V_{G_2}$, is called *subgraph isomorphism* if $G_1[U_{G_1}]$ is the induced subgraph

[1] In the general definition of a weighted graph, weights are assigned to both edges and nodes. Since in our graphs only vertices are weighted, we call them attributed graphs to distinguish them from weighted graphs.

of G_1 and $G_2[U_{G_2}]$ is the induced subgraph of G_1. That is, a bijection between two subsets of nodes in G_1 and G_2, respectively, is subgraph isomorphism if it preserves the node adjacency and ancestor-descendent relations of G_1 and G_2. Subgraph isomorphism can be associated with a cost function of matching the induced subgraphs $G_1[U_{G_1}]$ and $G_2[U_{G_2}]$ by bijection f. One frequently used formulation of the cost function is *edit-distance* (Barrow and Burstall 1976). One can think of constructing the induced subgraphs $G_1[U_{G_1}]$ and $G_2[U_{G_2}]$ as removing nodes from V_{G_1} and V_{G_2} by the edit-operation called *remove*, after which the remaining nodes in U_{G_1} are matched with those in U_{G_2} by the edit operation called *match*. Let w_v be the cost of removing node v from a graph, and let $m_{v_1v_2}$ be the cost of matching two nodes v_1 and v_2. Both edit-costs can be defined in terms of the attribute vectors $\boldsymbol{\psi}_v$, as will be discussed in Sect. 6. Then, the edit-distance of a given subgraph isomorphism $f : U_{G_1} \to U_{G_2}$ is defined as

$$D \triangleq \sum_{v_1 \in V_{G_1} \setminus U_{G_1}} w_{v_1} + \sum_{v_2 \in V_{G_2} \setminus U_{G_2}} w_{v_2} + \sum_{(v_1,v_2) \in U_{G_1} \times U_{G_2}} m_{v_1v_2} \tag{1}$$

$$= \sum_{v_1 \in V_{G_1}} w_{v_1} + \sum_{v_2 \in V_{G_2}} w_{v_2} - \sum_{(v_1,v_2) \in U_{G_1} \times U_{G_2}} [w_{v_1} + w_{v_2} - m_{v_1v_2}]. \tag{2}$$

The goal of a graph matching algorithm can be formulated as finding the subgraph isomorphism that is characterized by the minimum cost function. Alternatively, a matching algorithm can be formulated as a search for the maximum subgraph isomorphism, characterized by the maximum similarity measure (also referred to as utility Torsello and Hancock 2002, 2003) W defined as

$$W \triangleq \max_f \sum_{(v_1,v_2) \in f} [w_{v_1} + w_{v_2} - m_{v_1v_2}]_+, \tag{3}$$

where $[x]_+ \triangleq \max(0, x)$. If $[w_{v_1} + w_{v_2} - m_{v_1v_2}] \geq 0$, $\forall (v_1, v_2)$, the minimum-cost edit-sequences of *removes* and *matches* on G_1 and G_2 induce the maximum subgraph isomorphism between G_1 and G_2.

3.2 Problem Formulation

Tree matching can be formulated as finding two optimal sequences of *edit-operations*, called *edit-sequences*, on trees T_1 and T_2, respectively, which produce the maximum subtree isomorphism (Torsello and Hancock 2003; Bunke and Kandel 2000). In this paper, we consider the following set of edit-operations:

(1) *Remove*: Removes a node, and links its children to its parent.
(2) *Insert*: Adds a new node between an existing node u and its children $v_1, \ldots, v_n$, so that the inserted node becomes the single child of u, and the parent of $v_1, \ldots, v_n$.
(3) *Merge*: Coalesces n-tuples of adjacent sibling nodes having the same parent, referred to as *source* nodes, into a new node, called *merger*, which is then added to the graph. In the n-tuple, all source nodes represent regions are adjacent to each other. The merger does not remove the sources from the graph. The merger is connected to the sources' parent, and inherits all the sources' children.
(4) *Split*: Generates two or more nodes from a single source node. Each new node thus obtained is then connected to the parent and children as the source node.
(5) *Match*: Matches a pair of nodes.

For the purpose of matching T_1 and T_2, we treat *remove* as dual to *insert*, and *merge* as dual to *split*, because the result of *remove* (*split*) in T_1 can be achieved by using *insert* (*merge*) in T_2. Consequently, edit-sequences on T_1 and T_2 may consist of only *remove*, *merge* and *match* operations.

In this paper, we take one of many possible solutions to finding the optimal edit-sequences on T_1 and T_2. Specifically, we first conduct all possible *merges* on T_1 and T_2, which yields directed graphs G_1 and G_2, respectively, and then perform necessary *removes* and *matches* on nodes of G_1 and G_2, which would induce maximum subtree isomorphism. This reduces edit-sequence optimization to selecting the optimal sequence of *removes* and *matches*. An example of performing one *merge* operation on a tree is illustrated in Fig. 2. Note that *merge* transforms the tree into a DAG in which the connectivity and ancestor-descendent relations between nodes of the original tree are preserved by definition. The two acceptable sequences of *removes* and *matches*

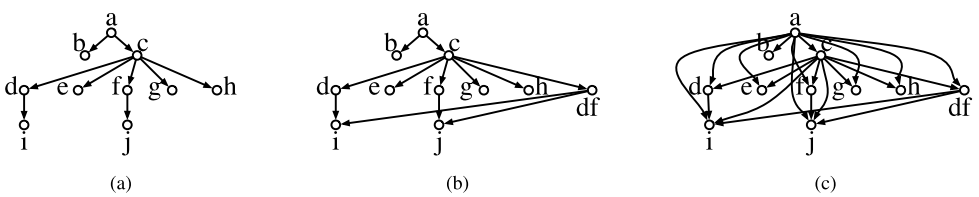


Fig. 2 Suppose in the tree depicted in (**a**) only nodes d and f represent adjacent sibling regions embedded in a larger parent region c. Then, *merge* can be applied only to d and f, which results in a new node df, depicted in (**b**). The transitive closure of the DAG in (**b**) is illustrated in (**c**)

on nodes of G_1 and G_2 must ensure that the resulting maximum subtree isomorphism between G_1 and G_2 satisfies the following consistency constraint: if a merger is matched then its source nodes cannot be matched and vice versa.

In the following section, we discuss our approach to finding the optimal sequence of *removes* and *matches*.

3.3 Transitive Closures

We propose to conduct matching between two given DAGs by matching their transitive closures. The transitive closure is constructed from a DAG by establishing additional parent-child connections between each pair of nodes that are connected with a directed path in the DAG, as illustrated in Fig. 2c. Formally, a transitive closure of a DAG G, denoted as $\Omega(G)$, is a DAG with the same node set as G, and with edges satisfying $u \sim v$ in $\Omega(G) \Leftrightarrow u \rightsquigarrow v$ in G. Note that multiple distinct paths $u \rightsquigarrow v \in G$ correspond to the unique edge $u \sim v$ in $\Omega(G)$. Note that our approach generalizes Torsello and Hancock's (2002, 2003) idea of using transitive closures of trees by extending it to DAGs. More specifically, we give necessary and sufficient conditions for finding the sequences of *removes* and *matches* on two DAGs G_1 and G_2 which induce a subtree isomorphism between transitive closures $\Omega(G_1)$ and $\Omega(G_2)$ subject to the consistency constraints. To this end, we begin with the following lemma: the operations of transitive closure, $\Omega(\cdot)$, and *remove* of node v from a DAG, $R_v(\cdot)$, commute.

Lemma 1 $R_v(\Omega(G)) = \Omega(R_v(G))$.

Proof Since $R_v(G)$ and $R_v(\Omega(G))$ operate on the same node v, two graphs $R_v(\Omega(G))$ and $\Omega(R_v(G))$ have the same node sets. Next, we show that edge $a \sim b$, $a \neq v$, $b \neq v$ is in $R_v(\Omega(G))$ if and only if it is in $\Omega(R_v(G))$. Distinct paths between a and b in G that do not contain v will not be effected by $R_v(G)$ leaving the corresponding edges in $R_v(\Omega(G))$ and $\Omega(R_v(G))$ intact. Now, suppose v is on multiple distinct paths $a \overset{v}{\rightsquigarrow} b$ in G, then by definition we have $a \rightsquigarrow b \in R_v(G) \Leftrightarrow a \overset{v}{\rightsquigarrow} b \in G$. Thus, $a \sim b \in \Omega(R_v(G)) \Rightarrow a \rightsquigarrow b \in R_v(G) \Rightarrow a \overset{v}{\rightsquigarrow} b \in G \Rightarrow a \sim b \in \Omega(G) \Rightarrow a \sim b \in R_v(\Omega(G))$. Similarly, we have $a \sim b \in R_v(\Omega(G)) \Rightarrow a \sim b \in \Omega(G) \Rightarrow a \overset{v}{\rightsquigarrow} b \in G \Rightarrow a \rightsquigarrow b \in R_v(G) \Rightarrow a \sim b \in \Omega(R_v(G))$. □

Recall that DAGs G_1 and G_2 are obtained by applying all possible *merge* operations to the original trees. Lemma 1 allows us to relate the edit-based matching of G_1 and G_2, using *remove* and *match* operations, to a subtree isomorphism between $\Omega(G_1)$ and $\Omega(G_2)$ which must satisfy the consistency constraints, as stated in the following theorem.

Theorem 1 *Suppose subtree t is obtained from a DAG G by applying a sequence of* removes. *Then, G and t are subtree isomorphic subject to the consistency constraints if and only if for each two siblings v_1 and v_2 in t there is no edge $v_1 \sim v_2$ in $\Omega(G)$.*

Proof Note that Theorem 1 is a generalization of its counterpart in Torsello and Hancock (2002, 2003), wherein every sequence of *removes* on tree T induces an isomorphic subtree of $\Omega(T)$. In contrast, our Theorem 1 deals with DAGs, where some sequences of *removes* on G may not induce an isomorphic subtree of $\Omega(G)$, and may not satisfy the consistency constraints.

Suppose $t = \mathcal{R}(G)$, where $\mathcal{R}(G) = R_{v_1} \circ \cdots \circ R_{v_n}(G)$ is a sequence of *removes*. Then, by virtue of Lemma 1, we have $\Omega(t) = \mathcal{R}(\Omega(G))$, i.e., t is a subtree of $\Omega(G)$. It follows that for any two siblings v_1 and v_2 in t there are no edges $v_1 \sim v_2$ in $\Omega(G)$, because by definition there are no directed paths between siblings v_1 and v_2 in t.

Now, to prove the converse, suppose that t is a subtree of $\Omega(G)$, and that for any two siblings v_1 and v_2 in t there is no edge $v_1 \sim v_2$ in $\Omega(G)$. Our goal is to prove that there exists a sequence of *removes* on G resulting in isomorphic subtree t. Note that $a \sim b \in t \Rightarrow a \sim b \in \Omega(G)$, which means that there are possibly multiple distinct paths $a \rightsquigarrow b \in G$. It is therefore necessary to show that for every edge $a \sim b \in t$, $\mathcal{R}(G)$ must operate on all nodes $\{v_i\} \in G$ that are on paths $a \overset{v_i}{\rightsquigarrow} b \in G$.

Suppose that node v is not removed by $\mathcal{R}(G)$, and v is on path $a \overset{v}{\rightsquigarrow} b \in t$, and $a \sim b \in t$. Let a node u in t denote the minimum common ancestor of a and v in t, as illustrated in Fig. 3. Since t is connected, there exist two distinct children of u in t, denoted as u_a and u_v, such that u_a is ancestor of a in t, and u_v is ancestor of v in t. Node u_v cannot be ancestor of a, because u is the minimum common ancestor of v and a. Therefore, there is a directed path $u_a \rightsquigarrow u_v \in G$, which contradicts the initial assumption that edge $u_a \sim u_v$ cannot be in $\Omega(G)$. Consequently, given $a \sim b \in t$, *all* nodes v in G that are on *all* paths $a \overset{v}{\rightsquigarrow} b \in G$ must be removed by $\mathcal{R}$.

By definition, $\mathcal{R}$ preserves connectivity and ascendant-descendant relations of nodes in G. Also, note that either the inserted mergers or their source nodes remain in t, since $\mathcal{R}(G)$ must remove *all* nodes v in G that are on *all* paths $a \overset{v}{\rightsquigarrow} b \in G$. Consequently, t satisfies the consistency constraints. □

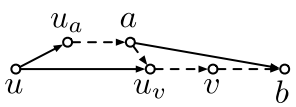


Fig. 3 Example of a hypothetical path $a \overset{v}{\rightsquigarrow} b \in t$ that coexists with edge $a \sim b \in t$: u, u_a, u_v, and v are hypothesized nodes in subtree t; *dashed lines* represent paths and full lines indicate edges. The figure shows that there must exist a directed path between u_a and u_v, which contradicts the initial assumption

In conclusion, by virtue of Theorem 1, it is possible to find two sequences of *removes* on G_1 and G_2 that would induce t_1 and t_2, which are subtree isomorphic to $\Omega(G_1)$ and $\Omega(G_2)$, respectively, and satisfy the consistency constraints. Since minimizing the edit-cost D, given by (2), and maximizing W, given by (3), are equivalent in our formulation, the minimum-cost sequence of *removes* on G_1 and G_2 yields the maximum consistent subtree isomorphism between $\Omega(G_1)$ and $\Omega(G_2)$.

4 Recursive Computation of Similarity Measure Bottom-Up

To compute the maximum consistent subtree isomorphism between $\Omega(G_1)$ and $\Omega(G_2)$, we use the following strategy. For each pair of nodes (u_1, u_2), where $u_1 \in \Omega(G_1)$ and $u_2 \in \Omega(G_2)$, the maximum consistent subtree isomorphism is found for the two transitive closures rooted at u_1 and u_2. In this fashion, the matching problem is solved recursively through a set of sub-matching problems. Let us assume that for all the children v_1 of u_1 in $\Omega(G_1)$ and all the children v_2 of u_2 in $\Omega(G_2)$ we have previously computed $W(v_1, v_2)$, given by (3), of the transitive closures rooted at v_1 and v_2. Then, our goal is to find the optimal set of (v_1, v_2) that do not violate the consistency constraints, and at the same time yield the largest $W(u_1, u_2)$. From (3), we have

$$W(u_1,u_2) = [w_{u_1} + w_{u_2} - m_{u_1u_2}]_+ + \sum_{(v_1,v_2)\in\mathcal{C}(u_1,u_2)} W(v_1,v_2), \tag{4}$$

where $\mathcal{C}(u_1, u_2)$ denotes the set of selected consistent pairs of children of u_1 and u_2. Once computed, $W(u_1, u_2)$ is further propagated to ancestors of u_1 and u_2.

As shown in Barrow and Burstall (1976), Pelillo et al. (1999), Torsello and Hancock (2003), finding $\mathcal{C}(u_1, u_2)$ is equivalent to solving the maximum clique problem. That is, $\mathcal{C}(u_1, u_2)$ is the maximum weighted clique of the association graph constructed from all possible children pairs (v_1, v_2), where the v_1's are the children of u_1 and the v_2's are the children of v_2. Each node (v_1, v_2) of thus constructed association graph is characterized by weights $W(v_1, v_2)$. A clique $\mathcal{C}$ is a subset of graph vertices in which all pairs of nodes are connected by an edge. The maximum weighted clique has the largest total weight of nodes included in the clique. Thus, we decompose the graph-matching problem into a series of maximum weighted clique problems, following a long-track record of similar approaches reported in the literature since the seminal paper by Barrow and Burstall (1976).

The maximum weighted clique formalism allows us not only to find the maximum similarity measure associated with the maximum consistent subtree isomorphism, but also to account for the consistency constraints in a principled manner, as explained below. For each node pair $(u_1, u_2) \in \Omega(G_1) \times \Omega(G_2)$, we first construct the association graph $A = (V_A, E_A, W_A)$, where $V_A = \{i = (v_1, v_2) : u_1 \sim v_1, u_2 \sim v_2$ and $u_1, v_1 \in \Omega(G_1), u_2, v_2 \in \Omega(G_2)\}$, E_A is the set of undirected edges, and $W_A : V_A \to \mathbb{R}^+$ is a function that assigns positive real weights $W(i)$ to nodes $i = (v_1, v_2) \in V_A$, where $W(i) = W(v_1, v_2)$ is given by (4). When specifying E_A, node pairs (v_1, v_2) and (v_1', v_2') in A should not be connected if v_1 and v_1' or v_2 and v_2' violate the consistency constraints. Hence, imposing the consistency constraints is done in a simple manner during the construction of A.

There are many approaches to solving the maximum weighted clique problem (see Bomze et al. 1999 for a review). In this paper, we use the game (replicator) dynamics approach thoroughly discussed in Pelillo et al. (1999), Pelillo (1999), Bomze et al. (2000), Pelillo (2002). This method uses the Motzkin-Straus theorem (Motzkin and Straus 1965) to transform the maximum clique problem, known to be NP-hard, into the following continuous quadratic programming problem. Consider a non-negative, symmetric matrix $Q = (q_{ij})_{|V_A|\times|V_A|}$, where for nodes i, j in the association graph A we have: (i) $q_{ij} \triangleq \frac{1}{2W(i)}$ if $i = j$; (ii) $q_{ij} \triangleq 0$ if $i \neq j$, and $i \sim j$; and (iii) $q_{ij} \triangleq \frac{1}{2W(i)} + \frac{1}{2W(j)}$, otherwise. Also, consider a characteristic vector $\boldsymbol{x}^{\mathcal{C}} = (x_i^{\mathcal{C}})_{|V_A|\times 1}$ of a subset of vertices $\mathcal{C}$ in A, where $\boldsymbol{x}^{\mathcal{C}}$ is a point in the standard simplex $S_{|V_A|} = \{\boldsymbol{x} \in \mathbb{R}^{|V_A|} : \boldsymbol{e}^{\mathrm{T}}\boldsymbol{x} = 1, \boldsymbol{x} \geq \boldsymbol{0}\}$, defined as $x_i^{\mathcal{C}} \triangleq W(i)/\sum_{j\in\mathcal{C}} W(j)$ if $i \in \mathcal{C}$, or $x_i^{\mathcal{C}} \triangleq 0$ otherwise. Then, $\mathcal{C}$ is a maximum weighted clique of A if and only if $\boldsymbol{x}^{\mathcal{C}}$ is a global solution of the following optimization problem: $\boldsymbol{x}^{\mathcal{C}} = \max_{\boldsymbol{x}\in S_{|V_A|}} \boldsymbol{x}^{\mathrm{T}}(\xi\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} - Q)\boldsymbol{x}$ (Bomze et al. 2000), where $\xi \triangleq \max q_{ij}$. This quadratic programming problem can be solved iteratively by using the following replicator equations in iteration step $(t + 1)$:

$$x_i(t+1) = x_i(t)\frac{((\xi\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} - Q)\boldsymbol{x}(t))_i}{\boldsymbol{x}(t)^{\mathrm{T}}(\xi\boldsymbol{e}\boldsymbol{e}^{\mathrm{T}} - Q)\boldsymbol{x}(t)}, \quad i = 1,\ldots,|V_A|. \tag{5}$$

Starting from an arbitrary initial state (in our case $x_i(0) = 1/|V_A|, i = 1, \ldots, |V_A|$), it can be shown that the trajectory of $\boldsymbol{x}(t)$ under the replicator dynamics of (5) converges to a strict local optimizer of the above quadratic programming problem. The stopping criterion is set at iteration step t_c for which $\forall i, x_i(t_c) \notin [0.05, 0.95]$. Then, nodes i in the association graph A whose corresponding $x_i(t_c) > 0.95$ are taken as nodes of the maximum weighted clique of the association graph A.

The total weight of nodes in the maximum weighted clique of A is inserted in (4), which gives us $W(u_1, u_2)$. In this manner, we compute the similarity measure of all subtrees rooted at node pairs $(u_1, u_2) \in \Omega(G_1) \times \Omega(G_2)$.

Then, the subtree pair $(u_1, u_2)^*$ with the largest similarity measure determines the maximum consistent subtree isomorphism, whose set of matched nodes is equal to $\{(u_1, u_2)^*\} \cup \mathcal{C}(u_1, u_2)^*$.

In the following section we present the computational complexity of the proposed approach.

5 Computational Complexity of Matching

There are two major steps in our approach which contribute to computational complexity: augmentation of given trees with merger nodes, and actual matching of thus obtained graphs. The segmentation algorithm of Ahuja (1996), Tabb and Ahuja (1997) typically produces trees with approximately $|V| = 30$ to $|V| = 100$ nodes, for the image database described in Sect. 7. Let s denote the average number of adjacent sibling nodes under a visited parent. Then, the complexity of transforming a given tree into a graph by applying *merge* operations is $O(2^s|V|)$. Note that s is considerably smaller than the average total number of a node's children. Typically, we have $0 \leq s \leq 4$.

Once the transitive closures on the two DAGs with approximately $2^s|V|$ nodes are constructed, our next step is to solve for $2^s|V| \times 2^s|V|$ maximum weighted clique problems. The replicator dynamics algorithm of Torsello and Hancock (2003), Pelillo et al. (1999, 2001) that we use converges in such problems after only a few iterations. Each iteration involves $O(|G|^2)$ multiplications, $|G|$ being the total number of nodes in a graph whose maximum clique is computed (Pelillo et al. 1999). Thus, complexity of the second step is $O(16^s|V|^4)$.

Overall, computational complexity of our approach is $O(16^s|V|^4)$, which typically amounts to $O(10^{10})$ computations, performed in less than 20 seconds on a 2.8 GHz, 2 GB RAM PC for all pairs of images in the database presented in Sect. 7. This does not include the processing time of the segmentation algorithm. Compared to Torsello and Hancock's approach (Torsello and Hancock 2002, 2003), ours increases computational complexity $O(16^s)$ times. This increase is justified by significant improvements in matching performance as reported in Sect. 7.

6 Specification of Edit-Costs

The problem of defining the optimal edit-costs has received considerable attention in the literature (see, e.g., Torsello and Hancock 2006 for a review). In this paper, the costs of node removal and matching, w_{v_1} and $m_{v_1 v_2}$, are specified as a function of intrinsic geometric and photometric properties of the image regions, represented by nodes v_1 and v_2. Requirements of a particular application inform the selection of optimal region properties. Below, we present one of many possible sets of region properties, which we use for defining the edit-costs.

Let μ_v and σ_v^2 denote the mean and variance of the gray-level pixel values in segmented region v. Also, let a_v denote v's area, whose centroid is located at image coordinates (x_v, y_v). To describe the boundary shape of v, we parse the image into $L = 40$ pie slices, each of which begins at (x_v, y_v), and subtends the same angle $2\pi/L$. Next, we compute the normalized histogram $h_v(l)$, $l = 1, \ldots, L$, of the number of pixels of region v that fall in pie slice l. The histogram is made rotation invariant, with respect to rotations by angles that are integer multiples of $2\pi/L$, by assigning $l = 1$ to the slice having the largest histogram value. In case the assumption of rotation invariance is invalid, or rotation invariance not required, label $l = 1$ is assigned to the slice containing the image's "x" axis. The entropy of such normalized histogram is defined as

$$H_v \triangleq -\sum_{l=1}^{L} h_v(l) \log h_v(l). \tag{6}$$

For a sufficiently high value of L ($L > 20$), the matching algorithm is not sensitive to changes in L values. Note that rotation invariance might be violated by the presence of noise in cases where the two largest slices are of similar sizes. Therefore, in addition to the shape histograms, we alternatively use four of the well known shape affine moment invariants (Sonka et al. 1999), denoted as I_1, I_2, I_3, and I_4, and given by

$$\begin{aligned}
I_1 &= \frac{\nu_{20}\nu_{02} - \nu_{11}^2}{\nu_{00}^4},\\
I_2 &= (\nu_{30}^2\nu_{03}^2 - 6\nu_{30}\nu_{21}\nu_{12}\nu_{03} + 4\nu_{30}\nu_{12}^3\\
&\quad + 4\nu_{03}\nu_{21}^3 + 3\nu_{21}^2\nu_{12}^2)/\nu_{00}^{10},\\
I_3 &= (\nu_{20}(\nu_{21}\nu_{03} - \nu_{12}^2) - \nu_{11}(\nu_{30}\nu_{03} - \nu_{12}\nu_{21})\\
&\quad + \nu_{02}(\nu_{12}\nu_{30} - \nu_{21}^2))/\nu_{00}^7,\\
I_4 &= (\nu_{20}^3\nu_{03}^2 - 6\nu_{20}^2\nu_{11}\nu_{12}\nu_{03} - 6\nu_{20}^2\nu_{02}\nu_{21}\nu_{03}\\
&\quad + 9\nu_{20}^2\nu_{02}\nu_{12}^2 + 12\nu_{20}\nu_{11}^2\nu_{12}\nu_{03} + 6\nu_{20}\nu_{11}\nu_{02}\nu_{03}\nu_{30}\\
&\quad - 18\nu_{20}\nu_{11}\nu_{02}\nu_{21}\nu_{12} - 8\nu_{11}^3\nu_{30}\nu_{03} - 6\nu_{20}\nu_{02}^2\nu_{30}\nu_{12}\\
&\quad + 9\nu_{20}\nu_{02}^2\nu_{21}^2 + 12\nu_{11}^2\nu_{02}\nu_{30}\nu_{12}\\
&\quad - 6\nu_{11}\nu_{02}^2\nu_{03}\nu_{21} + \nu_{02}^3\nu_{30}^2)/\nu_{00}^{11},
\end{aligned} \tag{7}$$

where the shape moments, ν_{pq}, are computed as $\nu_{pq} \triangleq \sum_{xy} x^p y^q F(x, y)$, and where x and y are the image coordinates, and $F(x, y) = 1$ if pixel at location (x, y) belongs to the region, and $F(x, y) = 0$ otherwise. The four shape affine moment invariants of region v are grouped in a vector $\mathcal{I}_v = [I_1, I_2, I_3, I_4]$.

The vector of intrinsic properties of region v, $\boldsymbol{\psi}_v$, comprises the following components:

$$\boldsymbol{\psi}_v = [\mu_v, \sigma_v^2, a_v, x_v, y_v, h_v, \mathcal{I}_v]. \tag{8}$$

Together, the intrinsic region properties are used to compute the cost of removing node v, w_v, as

$$w_v \triangleq \gamma\left[\frac{|\mu_v - \mu_u|}{\max(\mu_v, \mu_u)} + \frac{|\sigma_v^2 - \sigma_u^2|}{\max(\sigma_v^2, \sigma_u^2)}\right] + (1-\gamma)\left[\frac{a_v}{a_u} + H_v\right], \tag{9}$$

where u is the parent of v, and $\gamma \in [0, 1]$ represents the significance given to region photometric properties relative to geometric properties.

The cost of matching shape histograms h_{v_1} and h_{v_2} of regions v_1 and v_2 is specified as the χ^2 test statistic:

$$\rho_{v_1 v_2} \triangleq \frac{1}{L}\sum_{l=1}^{L}\frac{(h_{v_1}(l) - h_{v_2}(l))^2}{h_{v_1}(l) + h_{v_2}(l)}. \tag{10}$$

Then, the cost of matching two nodes v_1 and v_2, $m_{v_1 v_2}$, is defined as

$$m_{v_1 v_2} \triangleq \gamma\frac{(\mu_{v_1} - \mu_{v_2})^2}{\sigma_{v_1}^2 + \sigma_{v_2}^2} + (1-\gamma)\left[\left|\frac{a_{v_1}}{a_{u_1}} - \frac{a_{v_2}}{a_{u_2}}\right| + \rho_{v_1 v_2}\right], \tag{11}$$

where u_1 and u_2 are the parents of v_1 and v_2, respectively, and $\gamma \in [0, 1]$ is a desired weight of the photometric properties relative to geometric properties.

Alternatively, when the affine moment invariants are used to represent region boundary shape then the term H_v in (9) is replaced with $\|\mathcal{I}_v\|_1 = I_1(v) + I_2(v) + I_3(v) + I_4(v)$, and the term $\rho_{v_1 v_2}$ in (11) is replaced with $\|\mathcal{I}_{v_1} - \mathcal{I}_{v_2}\|_1 = |I_1(v_1) - I_1(v_2)| + |I_2(v_1) - I_2(v_2)| + |I_3(v_1) - I_3(v_2)| + |I_4(v_1) - I_4(v_2)|$.

From (9) and (11), for any pair of nodes v_1 and v_2, expression $[w_{v_1} + w_{v_2} - m_{v_1 v_2}]$ may not be positive. Recall that to compute W we discard pairs (v_1, v_2) whose cost of node matching is larger than the sum of their removal costs, i.e, we use $[w_{v_1} + w_{v_2} - m_{v_1 v_2}]_+$ in (3). From our experiments, such node pairs make less than one percent of the total number of node pairs to be matched.

7 Experiments

Given two images, the proposed region-based image matching approach identifies regions in image 1 and their matches in image 2 such that each matched pair represents the largest regions having the maximum similarity in low-level properties given by (8). For experimental validation we choose a setting in which the goal is to match images in a set containing similar objects. To this end, we use a carefully prepared database consisting of 700 natural-scene images with similar objects, including flowers, animals, musical instruments and buildings. Similar objects occupy similar subimages, which may differ to a certain degree in low-level properties of their constituent regions, such as color, size, and shape across the database. These variations in low-level region properties, which should be tolerated by the matching algorithm, arise from: (1) differences among target objects (e.g., black and brown horses whose legs appear in slightly different poses); (2) different imaging conditions (e.g., images are captured under different lighting conditions, and similar objects are viewed at different scales, or from different viewpoints); and (3) a certain degree of occlusion and clutter unavoidable in natural scenes.

The database is publicly available at http://vision.ai.uiuc.edu/~sintod/datasets.html. There are 30 distinct types of objects appearing in the database images. The types of objects include flowers, animals, musical instruments, and buildings. The database is divided into three datasets with respect to the scene complexity. Dataset I consists of images each of which shows a single object well contrasted from a homogeneous background, and each target object appears in only 10 images. One sample pair from the set of 10 images for each of the 30 objects in Dataset I is shown in Fig. 4. The objects are labeled by their position in Fig. 4 in a raster scan. For example, *snail* is the 7th object. Dataset II contains only the first 20 objects of Dataset I, i.e., flowers, animals, and musical instruments. Again, there are 10 images per object in Dataset II, examples of which are shown in Figs. 7 and 8. As can be seen, Dataset II is more complex than Dataset I in that the images are captured under greater variation in lighting conditions, the background contains clutter, and the objects are viewed at different scales or from different viewpoints. Finally, the 20 objects of Dataset II also appear in Dataset III, with 10 images per object. However, Dataset III increases the complexity over Dataset II as each image may contain multiple occurrences of similar target objects, as shown in Figs. 9–11.

We report the results of two types of experiments. The first type concerns matching of all possible pairs of images containing a target object, in a specific dataset. Matching performance is evaluated through the pixel- and region-matching errors. The ground truth for evaluating these errors is obtained by hand-labeling regions comprising to the objects of interest in given images. Given two images, let G denote the total ground-truth area of target objects in the images, and let M denote the total area of all matched regions in the two images. Then, the pixel-matching error, e_p, is computed as the XOR of G and M, expressed as a percentage of the union of G and M, $e_p \triangleq [(\mathrm{G} \cup \mathrm{M}) \setminus (\mathrm{G} \cap \mathrm{M})]/(\mathrm{G} \cup \mathrm{M})$. Note that in Dataset III any two images may contain a

Fig. 4 One sample pair of the 10 images for each of the 30 objects in Dataset I. The objects are labeled by their position in a raster scan. For example, the statue of liberty is the 22nd object

different total number of target objects. Since the image with fewer target objects constrains possible matches, the ground-truth area in both images should be labeled so as to include the maximum possible number of matches. The region-matching error is computed as the total number of matched node pairs that do not correspond to the same part of the target object in the two images, expressed as a percentage of the total number of matched node pairs. The evaluation of region-matching error is done by inspecting each matched node pair, and visually comparing the object parts associated with the two nodes. The pixel- and region-matching errors are averaged over all possible pairs of images containing the same target object within a specific dataset.[2]

In the second type of experiments, we assess the ability of the proposed approach to separate images containing different target objects from those showing similar ones. This is done by clustering images with respect to the quality of their match, where ideally each cluster should consists of only those images that contain a specific target object. Image tree clustering is conducted by using the normalized cuts algorithm (Shi and Malik 2000) with a distance between two trees, T_1 and T_2, defined as (Torsello et al. 2005)

$$d(T_1, T_2) \triangleq 1 - \frac{W(T_1, T_2)}{4|V_{T_1}||V_{T_2}|}, \tag{12}$$

where $|V_T|$ denotes the number of nodes in tree T, $W(T_1, T_2)$ is the maximum similarity measure between T_1 and T_2, given by (3). To show that $d(T_1, T_2)$ is positive, suppose $|V_{T_1}| \leq |V_{T_2}|$, then from (3) and (9), the upper limit of $W(T_1, T_2)$ is given by $W(T_1, T_2) \leq \sum_{v \in V_{T_1}} (w_v + w_{f(v)}) \leq 4 \cdot |V_{T_1}|$, from which we have $d(T_1, T_2) \geq 0$. The normalized cuts algorithm considers both the total dissimilarity between the different clusters, as well as the total similarity within the clusters with respect to $d(T_1, T_2)$ (Shi and Malik 2000).

Each cluster is labeled according to a majority vote over the labels of target objects present in the cluster. An image in a cluster that shows objects with a different label from that of the cluster is declared a clustering error. The average clustering error is computed as the total number of clustering errors, expressed as a percentage of the total number of images. In addition, we compute the confusion matrices of each dataset.

In both types of experiments, we compare our approach with the following tree-matching methods. The first method is Torsello and Hancock's edit-distance matching by using

[2]Matching is evaluated on images showing objects of the same type, but our approach does not prevent matching images that contain different objects (e.g., roosters and hens).

transitive closures on trees (Torsello and Hancock 2003). In their formulation merging and splitting of nodes is not considered, and, hence, we refer to this method as *One-to-one*. The second approach is the work of Pelillo and his collaborators (Pelillo et al. 2001), in which many-to-many node correspondences are considered by introducing the notion of ϵ-morphism. This method is a representative of those approaches in which the merging of nodes is controlled by thresholding a measure of their match, and, therefore, we refer to it as *Thresholding*. In our implementation of *Thresholding*, we set $\epsilon = 0 : 0.02 : 0.1$, and allow merging of all those nodes u and v in a given tree, along a unique directed path, whose cost of matching $m(u, v)$, given by (11), is less than ϵ. The third approach concerns the work of Keselman et al. (2003), which is a representative of the group of algorithms that achieve many-to-many matching by embedding graphs into a low-dimensional vector space. Therefore, we refer to this algorithm as *Embedding*. In our implementation of *Embedding*, each vector representing node v in the original tree is associated with weight w_v, given by (9). Note that *Embedding* was originally designed to match only entire images (e.g., for image retrieval), and may not be appropriate for matching their parts. Specifically, *Embedding* uses the Earth Mover's Distance algorithm to match several vectors obtained from one tree (which may not represent contiguous image parts) to a single vector obtained from another tree, or vice versa. We will denote the quality of matching between two such vectors, v_1 and v_2, as $W(v_1, v_2)$, in order to have a consistent notation for all the algorithms considered. Finally, the fourth matching approach concerns the brute-force variant of ours, where merging and splitting of nodes is not constrained but includes all sibling nodes. As in our approach, the mergers augment the initial trees, which in this case leads to an exponential increase in complexity. We refer to this method as *Brute-force*.

7.1 First Type of Experiments: Image Matching

The image-matching experiment consists of the following steps. First, images are processed by the multiscale segmentation algorithm (Ahuja 1996), which detects all regions in an image, and then represents their recursive containment structure as a tree. Each node in the tree represents a region, with its children corresponding to the subregions contained within the parent region. Each node also stores a list of region's properties as desired, e.g., those concerning gray level, area, shape and size. Next, all possible pairs of image trees, T_1 and T_2, showing the same target object are matched. For each pair of nodes $(v_1, v_2) \in T_1 \times T_2$, the similarity measure $W(v_1, v_2)$, given by (4), is computed. Then, the pair that has the maximum similarity measure $W^{\max}$ is determined, as well as the standard deviation of $W(\cdot)$ values with respect to $W^{\max}$, denoted as σ_f. All the node pairs (v_1, v_2) for which $W(v_1, v_2) \geq W^{\max} - \sigma_f$ are selected as the solution, i.e., the matched image regions in the two images.

Note that the optimal choice of threshold of $W(v_1, v_2)$ values is dictated by the requirements of a particular application (e.g., supervised vs. unsupervised settings, availability of cross-validation data, etc.), and could be a challenging research topic on its own. The aforementioned heuristic threshold appears reasonable for our experiments due to the following reasons. We hypothesize that $W(v_1, v_2)$ values of node pairs (v_1, v_2) representing the background are much smaller than the $W(v_1, v_2)$ values of node pairs representing the target object. In addition, it is desirable to select more than one node pair as maximally matching image regions, since two images may contain several occurrences of the target object. In our experiments, the proposed heuristic threshold of one standard deviation from the maximum similarity measure yields good results.

Figures 5–10 illustrate our matching results on Datasets I, II, and III. Each figure shows only a subset of the matched image regions, marked white. For the experiments presented in Figs. 5–9, the edit-costs are computed as specified by (9) and (11), and are not invariant to scale. The results shown in Fig. 10 are obtained for region boundary shapes represented by the four affine moment invariants, which are scale invariant. As can be seen in the figures, the number of unmatched

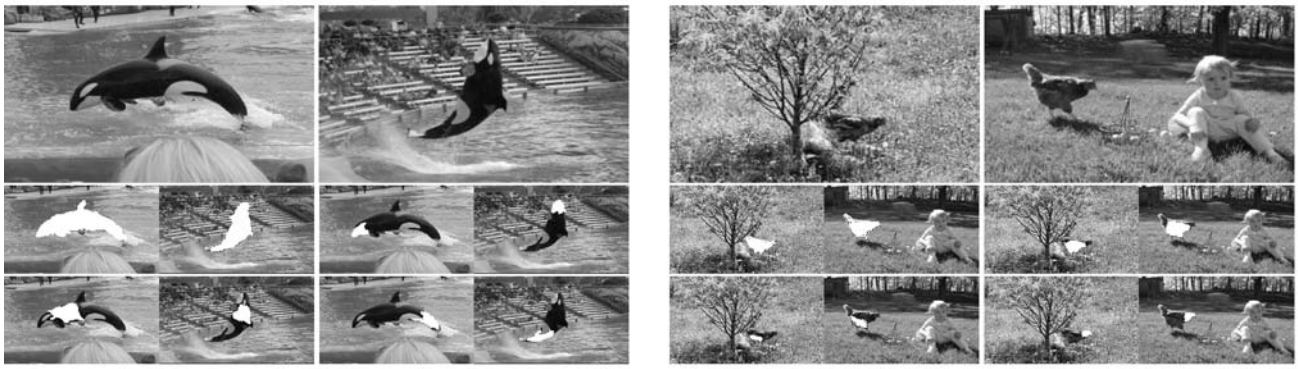
(a) two original images

(b) the result of matching the two images in (a)

Fig. 5 Matching in Dataset I: the tree representing *left* image is matched to the tree representing *right* image; *left* and the corresponding *right white* regions illustrate the matched nodes whose quality of match is above the specified threshold; as can be seen, some corresponding parts of the building in the two images are significantly different due to capturing the scene under different imaging conditions, and therefore their quality of match is below the threshold

(a) (b) (c)

Fig. 6 Examples of tree matching in Dataset I: the tree representing *left* image is matched to the tree representing *right* image; a few examples of the matched nodes are given for each experiment (**a**), (**b**), and (**c**); *left* and the corresponding *right white* regions illustrate the matched nodes



(a) (b)

Fig. 7 Examples of tree matching in Dataset II: see caption for Fig. 6

(a) (b)

Fig. 8 Examples of tree matching in Dataset II: see caption for Fig. 6

(a) (b)

Fig. 9 Examples of tree matching in Dataset III: see caption for Fig. 6

(a) (b) (c)

Fig. 10 Examples of tree matching in Dataset III, where shapes are represented by the four affine moment invariants: see caption for Fig. 6

pixels belonging to the target object is small. Also, note that the matched nodes in most cases do correspond to the same object parts in the two images.

The results shown in Fig. 11 allow us to point out the main characteristics of the competing algorithms. The result of *Thresholding*, shown in Fig. 11c, is the best for $\epsilon = 0.06$ over the range $\epsilon = 0 : 0.02 : 0.1$. Ours, shown in Fig. 11f, is obtained for $\gamma = 0.5$ and with the scale-invariant region shape representation. Regions marked white are *all* the matched regions for which $W(v_1, v_2) \geq W^{\max} - \sigma_f$. Note that among the matched regions are those representing non-target objects (e.g., grass patches), which are not counted toward error, according to the definitions of pixel- and region-matching errors. Typically, *One-to-one*, *Thresholding*, and *Embedding* yield matched regions with many "holes" in them, which represent unmatched, recursively embedded image subregions. Further, for all the algorithms on Dataset III, it may happen that different parts of one target object in image 1 are matched with parts of several target objects in image 2. For example, in Fig. 11e, the body and head of the left rhino in the right image matched the body of the leftmost rhino and the head of the middle rhino in the left image, respectively. This does not influence the region-matching error, as long as the matched regions represent semantically the same parts of the target objects.

The quantitative comparison is shown in Fig. 12. The plots depict the average pixel- and region-matching errors over Datasets I, II, and III. Note that the two entries in the legend, referred to as our approach, differ in region shape representation, as discussed in Sect. 6. The matching errors presented for *Thresholding* (Pelillo et al. 2001) are the best results found over the range of ϵ values, obtained for $\epsilon = 0.04, 0.06, 0.06$ on Datasets I, II, and III, respectively. The results of *Thresholding* are very sensitive to the choice of the threshold. The results of our approach are obtained for $\gamma = 0.5$. From Fig. 12, our approach significantly outperforms the competing methods with respect to the both matching errors. For example, the average pixel- and region-matching errors of our approach on Dataset I are 9.2% and 14.3%, respectively, while the corresponding errors of the second best *Thresholding* are 17.8% and 25.2%. When the scale invariant region shape representation is used, our approach yields improvements on Datasets II and III, as compared to when the histogram shape representation is used: 14.4% and 17.5% vs. 16.3% and 19.1% pixel-matching errors on Datasets II and III, respectively.

Brute-force exhibits the worst performance on all three datasets, with respect to both matching errors, despite the extra redundancy present due to mergers being the power sets of *all* sibling nodes. Our analysis of the nodes that got matched by this method indicates that *Brute-force* is biased towards matching larger merger nodes. The larger the number of source nodes forming a merger, the more likely it is for that merger to get matched.

The results presented in Fig. 13 demonstrate the impact of different γ values on the performance of our algorithm. For small γ values, $0 \leq \gamma < 0.5$, the region geometric properties are a dominant factor in computing W, given by (3). Conversely, for large γ values, $0.5 < \gamma \leq 1$, the photometric properties become more important in computing W. From Fig. 13, the optimal γ value for all three datasets, for which the pixel- and region-matching errors are minimum, is $\gamma = 0.5$. While these are average results, the experiments show that the optimal γ value is very close to 0.5 for each of the 30 target objects, too. That is, the strategy that uses equally weighted geometric and photometric properties as cues for matching outperforms the method that uses either of the properties. This observation is supported by each of Datasets I, II, and III.

Table 1 details the average processing time it takes our algorithm and the competing ones to match two images in Datasets I, II, and III. It is worth noting that the code for *One-to-one*, *Thresholding*, and *Embedding* is not publicly available, and that our implementation of these algorithms

(a) Two images from Dataset III

(b) *Brute-force*

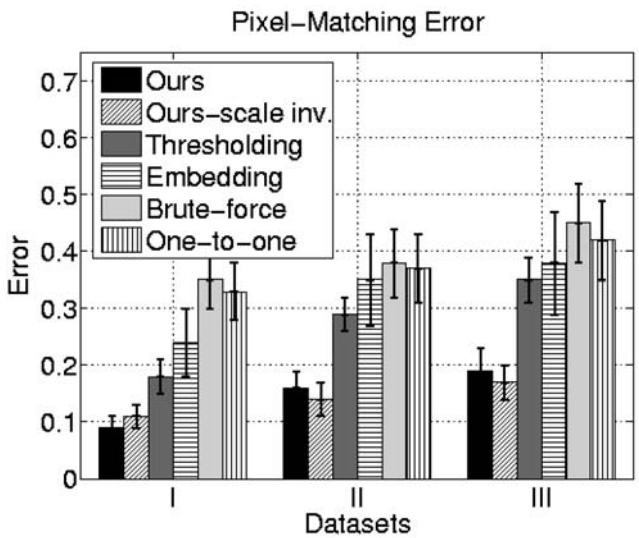
(c) *Thresholding* (Pelillo et al. 2001)

(d) *Embedding* (Keselman et al. 2003)

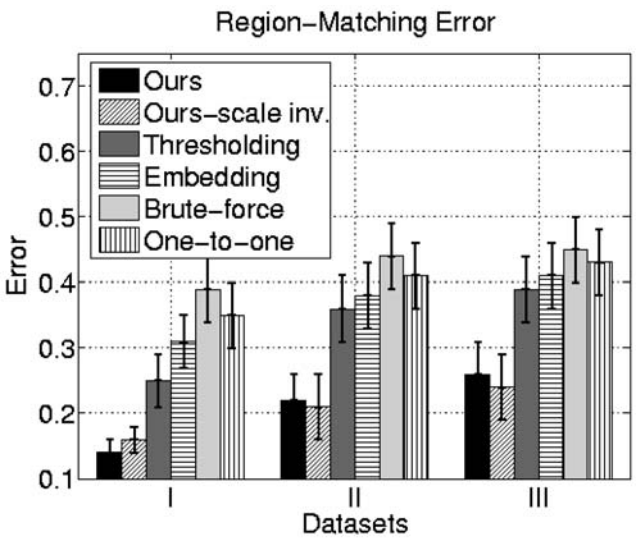
(e) *One-to-one* (Torsello and Hancock 2003)

(f) Ours

Fig. 11 Comparison of different matching approaches; tree T_1 representing *left* image is matched to tree T_2 representing *right* image; *left* and the corresponding *right white* regions illustrate *all* selected matched node pairs (v_1, v_2), $v_1 \in T_1$, and $v_2 \in T_2$, for which $W(v_1, v_2) \geq W^{\max} - \sigma_f$



(a)



(b)

Fig. 12 Comparison of the average matching errors of six tree-matching approaches on Datasets I, II, and III: (**a**) pixel-matching error, (**b**) region-matching error. The results of our approach are obtained for $\gamma = 0.5$, while those of *Thresholding*, for $\epsilon = 0.04,\ 0.06,\ 0.06$ for Datasets I, II, and III, respectively

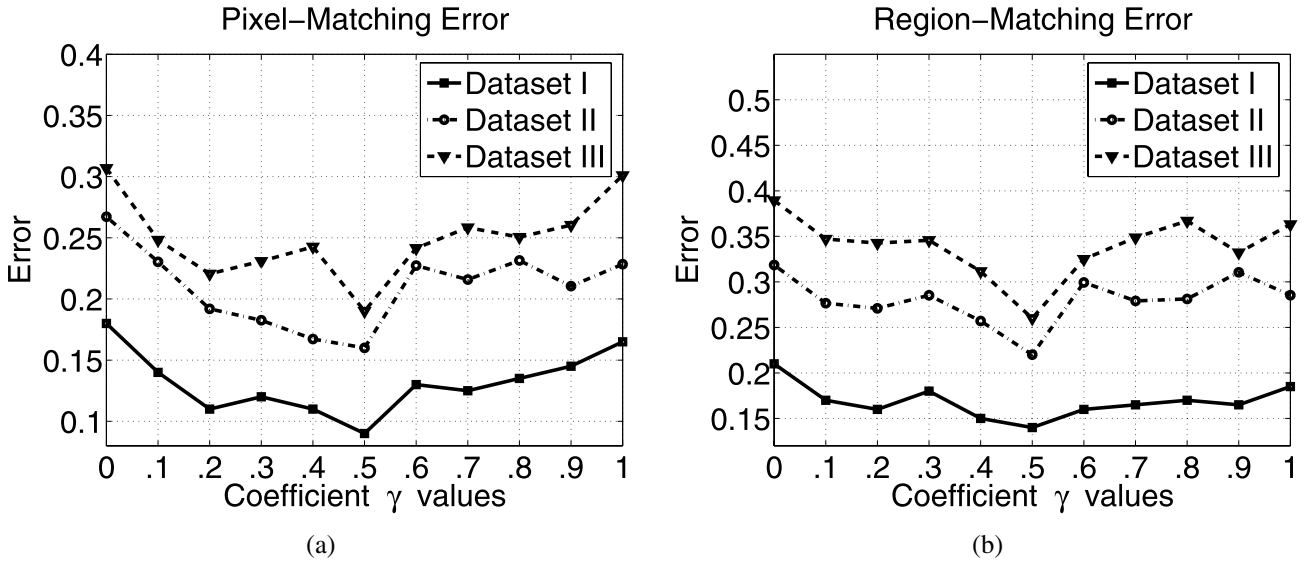


Fig. 13 Average (**a**) pixel-matching error, and (**b**) region-matching error for Datasets I, II, and III over a range of γ values

Table 1 Average processing time of matching two images

	Ours	*One-to-one*	*Thresholding* $\epsilon = 0.04$	*Embedding*
Dataset I	15.4 s	8.3 s	7.9 s	
Dataset II	16.9 s	9.2 s	8.8 s	<5 s
Dataset III	18.4 s	13.7 s	12.8 s	

may not yield optimal execution times. However, since our algorithm shares a number of steps with *One-to-one* and *Thresholding* (e.g., replicator dynamics algorithm), the comparison with these two algorithms is sufficiently fair. The code for *Embedding* is implemented in MATLAB, while the code for the other three approaches is implemented in C, and run on a 2.8 GHz 2 GB RAM PC. The presented times do not include multiscale image segmentation/tree generation. The increase in processing time from Dataset I to III is due to the increase in the size of maximum clique problems to be solved for images with several occurrences of the target object. *Embedding* has the best runtime, while *Thresholding* is the second best, as expected, by reducing the number of nodes in the original tree via ϵ-node merging.

7.2 Second Type of Experiments: Image Clustering

The experimental set-up for image clustering is similar to that of image matching, described in Sect. 7.1. After segmenting the images and finding their tree representations, all the trees within a specific dataset are clustered by using the normalized cuts algorithm (Shi and Malik 2000), with respect to the distance metric, given by (12).

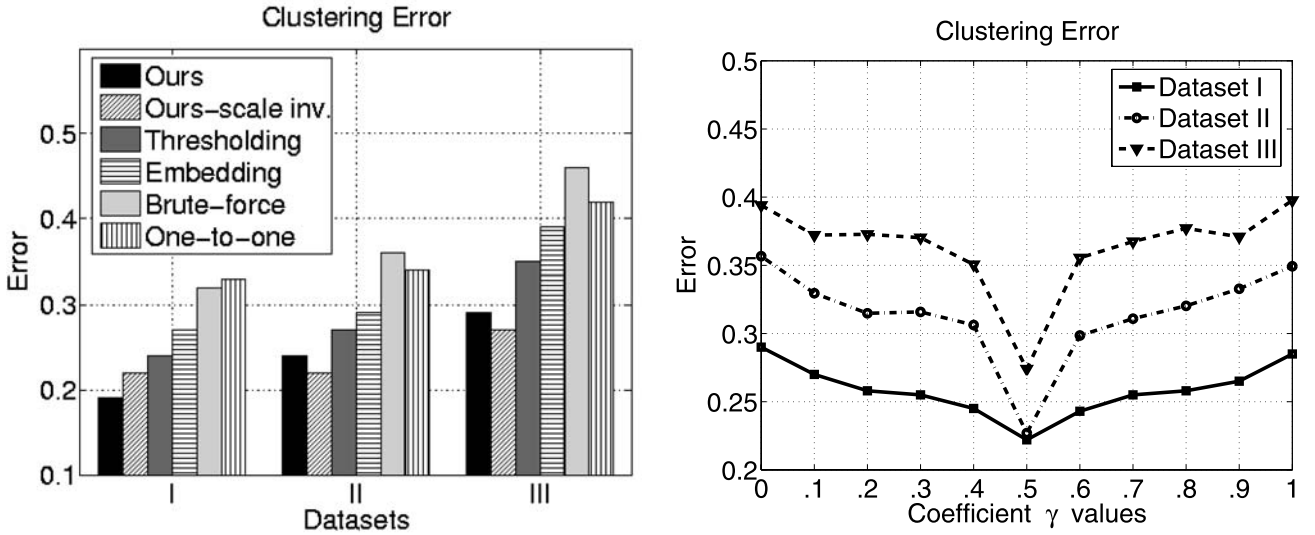
Figure 14 shows the clustering error of the competing algorithms and ours. The clustering errors of our approach for $\gamma = 0.5$ with the scale invariant region shape representation are 22.2%, 22.7%, and 27.4% on Datasets I, II, and III, respectively. The second best method is *Thresholding* (Pelillo et al. 2001) with clustering errors of 24.4%, 27.1%, and 35.6% obtained for $\epsilon = 0.02,\ 0.04,\ 0.06$ on Datasets I, II, and III, respectively. *Brute-force* has the worst performance. The results presented in Fig. 14b, show that the best performance of our approach is again obtained for equal contributions of photometric and geometric properties to W, i.e., when $\gamma = 0.5$.

Finally, Fig. 15 presents two confusion matrices on Datasets I and III, respectively, for our approach with the scale invariant region shape representation and $\gamma = 0.5$. Rows of the confusion matrices indicate the labels of target objects, and columns represent the cluster labels. From Fig. 15, the most confused objects have labels 1 and 2, and 4 and 5, the appearances of which are also difficult to correctly classify for a human (see Fig. 4).

8 Conclusion

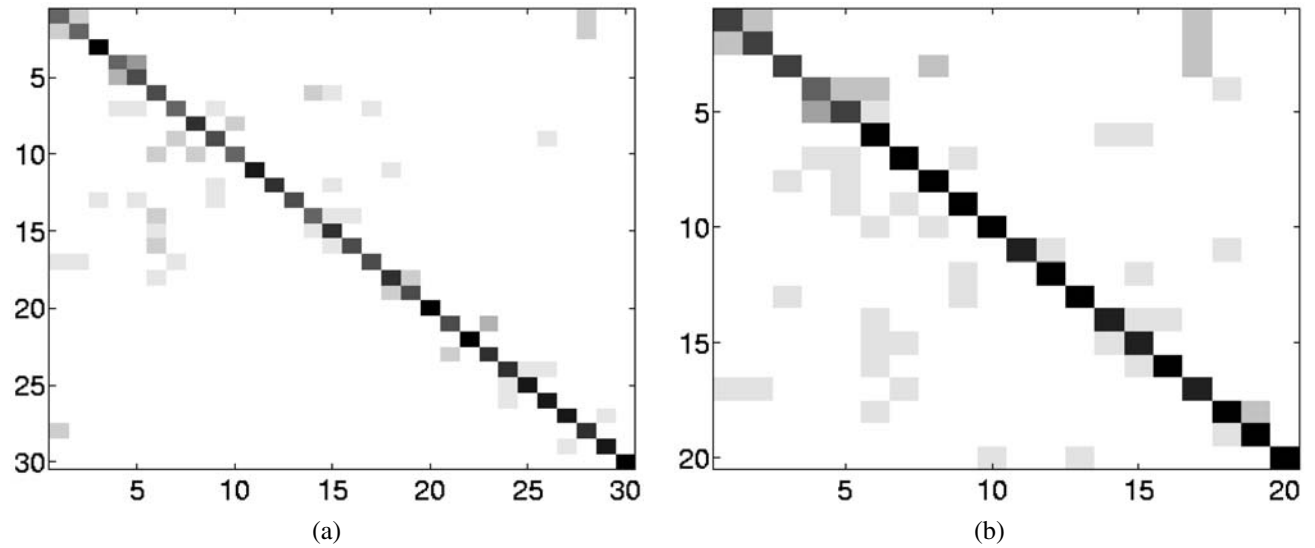
Image matching is of special interest to a wide range of computer vision problems, from stereo matching to object categorization. While most such work uses point or edge features, there are also approaches that use regions for image matching. The motivation behind the work reported in this paper is that the use of hierarchical region properties should significantly extend the robustness of image matching to noise.

One of the main difficulties in region matching is that low-contrast contiguous regions may easily merge into a large, less homogeneous region, and the reverse may happen



(a) (b)

Fig. 14 (**a**) Comparison of the clustering errors of five tree-matching approaches on Datasets I, II, and III; the results of our approach are obtained for $\gamma = 0.5$, while those of *Thresholding*, for $\epsilon = 0.02, 0.04, 0.06$ for Datasets I, II, and III, respectively. (**b**) The average clustering error of our approach with the scale invariant shape representation for Datasets I, II, and III over a range of γ values



(a) (b)

Fig. 15 Clustering using our approach with the scale invariant shape representation for $\gamma = 0.5$; (**a**) Confusion matrix of 30 object categories in Dataset I; (**b**) Confusion matrix of 20 object categories in Dataset III. Object categories are enumerated as explained in the caption of Fig. 4. Rows of the confusion matrix indicate object category, and columns represent clusters. The shade of matrix entries (i, j) indicates the number of images from category i in cluster j, where this number ranges from 0 (*white*) to 10 (*black*)

when a large region splits into several smaller, more homogeneous regions. These phenomena occur when the images showing the same scene are captured under different lighting conditions, or when the scene is viewed at different scales or from varying viewpoints. In this paper, we have proposed an algorithm that addresses these problems.

Images are represented as trees whose nodes correspond to segmented image regions at various scales, obtained by a multiscale segmentation algorithm. In the segmentation tree, large regions appear at upper tree levels closer to the root, while their children nodes capture smaller details completely contained within the corresponding parent region. The two trees are augmented with new nodes, called mergers, which represent the power sets of contiguous regions that are embedded in the same parent region. This transforms the trees into directed acyclic graphs (DAGs) that

preserve connectivity and ancestor-descendent relations of nodes in the original trees. Then, transitive closures on the DAGs are constructed, whereby ascendant-descendant relationships in the DAGs are transformed into parent-child node connections. Thus, image matching is formulated as a search for the maximum subtree isomorphism between the transitive closures of DAGs. This search is constrained, because each inserted merger node must be disallowed to match if its source nodes got matched, and the obtained maximum similarity common subtree must respect ascendant-descendant relationships of the initial trees.

The proposed approach is validated on three datasets, each introducing a higher degree of scene complexity. Experimental results of image matching and clustering are reported. In addition, the performance of our approach is contrasted to that of the competing methods, which (1) threshold possible node merges, (2) embed the trees into a low-dimensional vector space, and there specify many-to-many matching, (3) do not consider splitting/merging of nodes, and (4) allow for all possible merges of children nodes under a visited parent. Selection of optimal region properties for image matching is discussed. It is shown that the matching strategy with equally weighted region photometric and geometric properties outperforms those strategies where either photometric properties or geometric properties are favored. The results demonstrate that our approach outperforms the conventional methods with respect to suitably defined pixel, region and clustering errors, at a price of a reasonably increased complexity. Contrary to intuition, the brute-force method, where all sibling nodes of the original trees are allowed to merge, introducing exponential complexity, yields the worst performance.

Acknowledgements Himanshu Arora provided the segmentation code. The support of the Office of Naval Research under grant N00014-03-1-0107 is gratefully acknowledged.

References

Ahuja, N. (1996). A transform for multiscale image segmentation by integrated edge and region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(12), 1211–1235.

Arora, H., & Ahuja, N. (2006). Analysis of ramp discontinuity model for multiscale image segmentation. In *ICPR*.

Barrow, H. G., & Burstall, R. M. (1976). Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters*, *4*(4), 83–84.

Basri, R., & Jacobs, D. (1997). Recognition using region correspondences. *International Journal of Computer Vision*, *25*(2), 145–166.

Bomze, I. M., Budinich, M., Pardalos, P. M., & Pelillo, M. (1999). The maximum clique problem. In D. Z. Du & P. M. Pardalos (Eds.), *Handbook of combinatorial optimization (supplement Vol. A)* (pp. 1–74). Boston: Kluwer Academic.

Bomze, I. M., Pelillo, M., & Stix, V. (2000). Approximating the maximum weight clique using replicator dynamics. *IEEE Transactions on Neural Networks*, *11*(6), 1228–1241.

Bunke, H., & Allermann, G. (1983). Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, *1*(4), 245–253.

Bunke, H., & Kandel, A. (2000). Mean and maximum common subgraph of two graphs. *Pattern Recognition Letters*, *21*(2), 163–168.

Cohen, S., & Guibas, L. (1999). The Earth Mover's Distance under transformation sets. In *Proc. IEEE Int. Conf. Computer Vision* (Vol. 2, pp. 1076–1083).

Cohen, L., Vinet, L., Sander, P., & Gagalowicz, A. (1989a). Hierarchical region based stereo matching. In *Proc. IEEE Conf. Computer Vision Pattern Rec.* (pp. 416–421).

Cohen, L., Vinet, L., Sander, P., & Gagalowicz, A. (1989b). Hierarchical region based stereo matching. In *Proc. IEEE Conf. Computer Vision Pattern Rec.* (pp. 416–421).

Demirci, M. F., Shokoufandeh, A., Dickinson, S., Keselman, Y., & Bretzner, L. (2004). Many-to-many feature matching using spherical coding of directed graphs. In *Lecture notes in computer science: Vol. 3021. Proc. European Conf. Computer Vision* (pp. 322–335).

Eshera, M. A., & Fu, K. S. (1986). An image understanding system using attributed symbolic representation and inexact graph-matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *8*(5), 604–618.

Fowlkes, C., Belongie, S., Chung, F., & Malik, J. (2004). Spectral grouping using the Nystrom method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(2), 214–225.

Fuh, C. S., & Maragos, P. (1989). Region-based optical flow estimation. In *Proc. IEEE Conf. Computer Vision Pattern Rec.* (pp. 130–135).

Glantz, R., Pelillo, M., & Kropatsch, W. G. (2004). Matching segmentation hierarchies. *International Journal of Pattern Recognition and Artificial Intelligence*, *18*(3), 397–424.

Golland, P., Eric, W., & Grimson, L. (2000). Fixed topology skeletons. In *Proc. IEEE Conf. Computer Vision Pattern Rec.* (Vol. 1, pp. 10–17).

Keselman, Y., & Dickinson, S. (2005). Generic model abstraction from examples. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(7), 1141–1156.

Keselman, Y., Shokoufandeh, A., Demirci, M., & Dickinson, S. (2003). Many-to-many graph matching via metric embedding. In *Proc. IEEE Conf. Computer Vision Pattern Rec.* (Vol. 1, pp. 850–857).

Liu, T. L., & Geiger, D. (1999). Approximate tree matching and shape similarity. In *Proc. IEEE Int. Conf. Computer Vision* (Vol. 1, pp. 456–462).

Medioni, G., & Nevatia, R. (1985). Segment-based stereo matching. *Computer Vision, Graphics, and Image Processing*, *31*(3), 2–18.

Ming-Hsuan, Y., Ahuja, N., & Tabb, M. (2002). Extraction of 2d motion trajectories and its application to hand gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(8), 1061–1074.

Motzkin, T. S., & Straus, E. G. (1965). Maxima for graphs and a new proof of a theorem of Turan. *Canadian Journal of Mathematics*, *17*(4), 533–540.

Pardalos, P., & Xue, J. (1994). The maximum clique problem. *Journal Global Optimization*, *4*, 301–328.

Pelillo, M. (1999). Replicator equations, maximal cliques, and graph isomorphism. *Neural Computation*, *11*(9), 1935–1955.

Pelillo, M. (2002). Matching free trees, maximal cliques, and monotone game dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(11), 1535–1541.

Pelillo, M., Siddiqi, K., & Zucker, S. W. (1999). Matching hierarchical structures using association graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *21*(11), 1105–1120.

Pelillo, M., Siddiqi, K., & Zucker, S. W. (2001). Many-to-many matching of attributed trees using association graphs and game dynamics. In *Lecture notes in computer science: Vol. 2059. Int. Workshop Visual Form.* (pp. 583–593).

Perrin, B., Ahuja, N., & Srinivasa, N. (1998). Learning multiscale image models of 2D object classes. In *Proc. Asian Conf. Computer Vision* (Vol. 2, pp. 323–331).

Randriamasy, S., & Gagalowicz, A. (1991). Region based stereo matching oriented image processing. In *Proc. IEEE Conf. Computer Vision Pattern Rec.* (pp. 736–737).

Richard, W. C., Hancock, E. R., & Luo, B. (2005). Pattern vectors from algebraic graph theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(7), 1112–1124.

Rubner, Y., Tomasi, C., & Guibas, L. J. (1998). A metric for distributions with applications to image databases. In *Proc. IEEE Int. Conf. Computer Vision* (pp. 59–66).

Sanfeliu, A., & Fu, K. S. (1983). A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, *13*(3), 353–362.

Sebastian, T. B., Klein, P. N., & Kimia, B. B. (2004). Recognition of shapes by editing their shock graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(5), 550–571.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 888–905.

Shokoufandeh, A., Macrini, D., Dickinson, S., Siddiqi, K., & Zucker, S. W. (2005). Indexing hierarchical structures using graph spectra. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(7), 1125–1140.

Sonka, M., Hlavac, V., & Boyle, R. (1999). *Image processing, analysis, and machine vision* (2nd ed.). Pacific Grove: Brooks/Cole.

Tabb, M., & Ahuja, N. (1997). Multiscale image segmentation by integrated edge and region detection. *IEEE Transactions Image Processing*, *6*(5), 642–655.

Torsello, A., & Hancock, E. R. (2002). Matching and embedding through edit-union of trees. In *Lecture notes in computer science: Vol. 2352. Proc. European Conf. Computer Vision* (pp. 822–836).

Torsello, A., & Hancock, E. R. (2003). Computing approximate tree edit distance using relaxation labeling. *Pattern Recognition Letters*, *24*(8), 1089–1097.

Torsello, A., & Hancock, E. R. (2006). Learning shape-classes using a mixture of tree-unions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(6), 954–967.

Torsello, A., Rowe, D. H., & Pelillo, M. (2005). Polynomial-time metrics for attributed trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(7), 1087–1099.

Tsai, W. H., & Fu, K. S. (1979). Error-correcting isomorphism of attributed relational graphs for pattern analysis. *IEEE Transactions on Systems, Man, and Cybernetics*, *9*(12), 757–768.

Umeyama, S. (1988). An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *10*(5), 695–703.

Xuguang, Y., & Ramchandran, K. (1999). A low-complexity region-based video coder using backward morphological motion field segmentation. *IEEE Transactions on Image Processing*, *8*(3), 332–345.