

# Object Categorization and the Need for Many-to-Many Matching

Sven Dickinson<sup>1</sup>, Ali Shokoufandeh<sup>2</sup>, Yakov Keselman<sup>3</sup>, Fatih Demirci<sup>2</sup>, and Diego Macrini<sup>1</sup>

<sup>1</sup> Department of Computer Science, University of Toronto,

<sup>2</sup> Department of Computer Science, Drexel University,

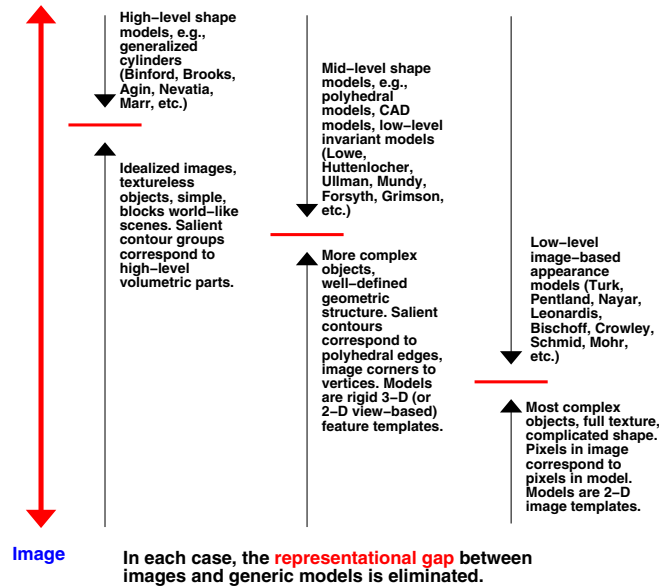
<sup>3</sup> School of CTI, DePaul University,

**Abstract.** Object recognition systems have their roots in the AI community, and originally addressed the problem of object categorization. These early systems, however, were limited by their inability to bridge the representational gap between low-level image features and high-level object models, hindered by the assumption of one-to-one correspondence between image and model features. Over the next thirty years, the mainstream recognition community moved steadily in the direction of exemplar recognition while narrowing the representational gap. The community is now returning to the categorization problem, and faces the same representational gap as its predecessors did. We review the evolution of object recognition systems and argue that bridging this representational gap requires an ability to match image and model features many-to-many. We review three formulations of the many-to-many matching problem as applied to model acquisition and object recognition.

## 1 Introduction

The evolution of object recognition over the past thirty years has witnessed a shift from shape-based category recognition to appearance-based exemplar recognition. As seen in Figure 1, early object recognition systems modeled object categories as collections of high-level, volumetric parts, such as generalized cylinders. These models represented intuitive, parts-based abstractions of objects that offered invariance to minor shape deformations, part articulation, and occlusion (due to the locality of the representation). Unfortunately, the vision community lacked the tools to recover such shape abstractions from real images of real objects, leaving a representational gap that remains a challenge to this day. Instead, category-level object recognition systems were presented with images of simple, textureless objects in which image features, such as edges, mapped directly to the surface discontinuities and occluding boundaries of the abstract parts comprising categorical models. In effect, the representational gap was artificially eliminated by bringing the images closer to the models. However, the simplistic nature of the resulting scenes left many unsatisfied.

The 1980's witnessed a movement toward geometric CAD models that captured the exact geometry of an object. Rather than defining object categories,



**Fig. 1.** Evolution of object recognition. In each period, the representational gap was avoided by either bringing the images closer to the models, bringing the models closer to the images, or both. Figure taken from [3], copyright IEEE, 2005.

such models were effectively 3-D shape templates of object exemplars. Once again, simple image features, such as edges, mapped directly to surface discontinuities and occluding boundaries of the model. This time, however, the representational gap was artificially eliminated by bringing the models closer to the image. Although the models were not categorical, they could be used to recognize object exemplars in real images, provided that a detailed geometric model could be acquired and that the imaged objects were textureless. These two restrictions meant that not only did a user have to construct or acquire a faithful 3-D geometric model of the object, the object had to be smoothly shaded, resulting yet again in an unrealistic recognition setting.

The 1990's saw a significant shift in object recognition philosophy from object-centered modeling and recognition to viewer-centered modeling and recognition. Whereas object-centered systems recognize 3-D models from 2-D images, viewer-centered systems modeled an object as a set of 2-D views, thereby reducing "3-D from 2-D" recognition to simple 2-D recognition, at the cost of having to store/match potentially many views. Within the viewer-centered framework, the same representational gap exists between low-level image features and categorical, view-based models. Instead of bridging this gap, the community eliminated the gap by bringing the models even closer to the image, storing the actual images of an object exemplar as its (appearance-based) model views. This paradigm gained tremendous popularity, as 3-D object modeling was no longer required

and an object could have arbitrarily complex structure and surface texture. An object database could be constructed by simply placing an object on a turntable and spinning it in front of a camera to acquire a dense set of views.

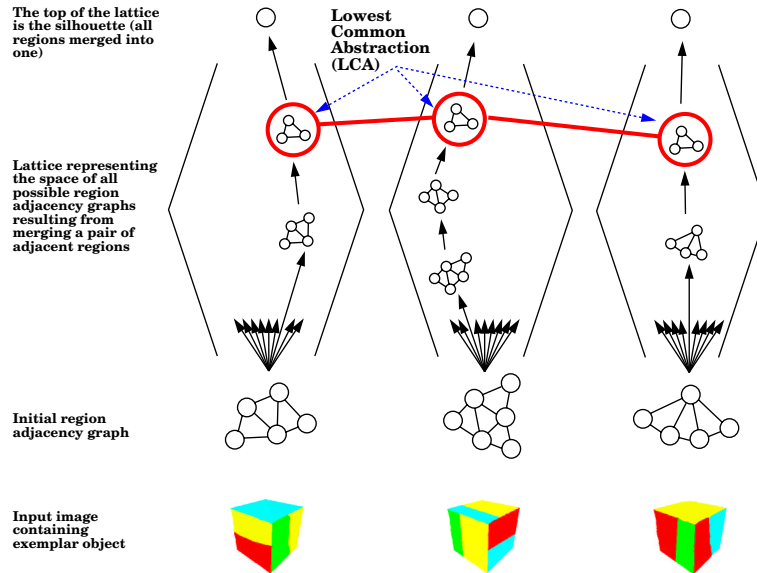
For the first time, real images of real objects could be recognized. However, it is important to note that through this evolution, the recognition community redefined the recognition problem from category recognition (the predominant goal of recognition systems in the 70's/80's) to exemplar recognition. In each of the three periods above, the difficult problem of image abstraction, i.e., bridging the representational gap by mapping low-level image features to high-level shape primitives, was avoided by either moving the images up or moving the models down. Underlying this movement is a fundamental assumption that for every salient image feature (e.g., region, line, pixel, pixel neighborhood, etc.) there exists a corresponding salient model feature. However, due to noise, segmentation errors, articulation, scale difference, within-class variation, etc., a feature in one image may correspond to a collection of features in another image.

The community is now beginning to return to its categorization roots, armed with new machine learning techniques and invariant image features. But the one-to-one feature correspondence assumption remains, and today's models still tend to encode the appearance (i.e., discriminating texture) of specific exemplars rather than the prototypical shapes of object categories. Hence, two exemplars with different texture/appearance that belong to the same shape category will have little in common with respect to their low-level image structure (e.g., neighborhood-encoded interest points or image patches). The success of object categorization depends on solving two problems. First, we must strive to recover the coarse, high-level shape features that define the prototypical shape of an object, with such features representing abstractions/groupings of low-level image features. Second, we must relax the restrictive, one-to-one correspondence assumption and develop mechanisms for matching image features *many-to-many*.

In this paper, we review three formulations of the many-to-many matching problem and its application to both shape category acquisition and recognition. We assume that an image can be processed to yield a structured collection of image features, conveniently represented as a graph whose nodes encode attributed features and whose edges encode feature relations. In each formulation, one-to-one feature (i.e., one-to-one node) correspondence between exemplars in a given category may not exist at the level of extracted features, but may exist at the level of groups of features. This sets up an intractable many-to-many graph matching problem, for which approximation methods must be sought. Each formulation takes a different approach to the problem, and we review preliminary progress on each front.

## 2 Model Abstraction from Examples

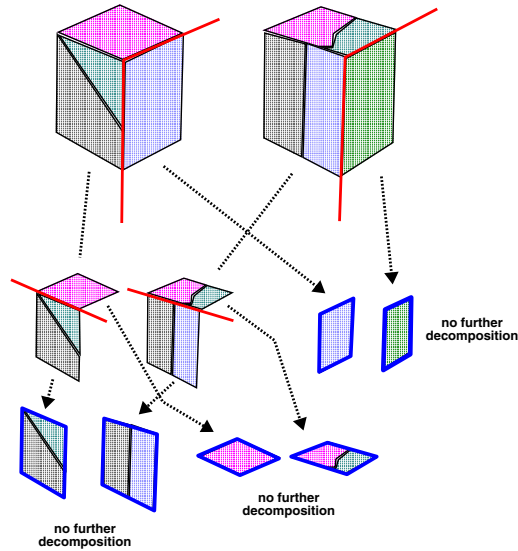
Our first formulation of many-to-many matching addresses the problem of trying to recover a structural model from a set of exemplars belonging to a known class. Consider the simple example shown in Figure 2, in which three different



**Fig. 2.** The Lowest Common Abstraction (LCA) of a set of input exemplars (blocks). In this case, the LCA of the three input exemplars would be the canonical view of the block, in which three surfaces are visible. Figure taken from [3], copyright IEEE, 2005.

exemplar images are presented to the system, each region segmented to yield a region adjacency graph. Although the structure of the input graphs may be different – in fact, not a single one-to-one node (feature) correspondence may exist between the input exemplar graphs – the exemplars are similar at a higher level of abstraction. If we consider the space of all possible graphs formed by merging two adjacent regions, then each input exemplar gives rise to a lattice of region adjacency graphs, representing all possible region groupings. Finding a model abstraction that best accounts for the input exemplars consists of finding the most complex (maximum cardinality equals most informative) graph that lies in the intersection of the lattices. Since more complex abstraction graphs are lower in the lattice, we call this graph the *lowest common abstraction (LCA)*, representing a category-level, view-based description of one view class; additional view classes are acquired by rotating each of the exemplars and repeating the process. Note that the LCA may not be unique.

The above formulation is intractable, for neither the lattices nor their intersection can be realistically generated. Instead, we first narrow the scope of the problem by finding the lowest common abstraction of two lattices, and exploit the fact that we know one member of the intersection lattice, namely the silhouette. As shown in Figure 3, we work top-down from the silhouette, searching for a pair of cuts, one per graph, in the two exemplars' region adjacency graphs, such that corresponding shapes (each cut yields two shapes) and their relations



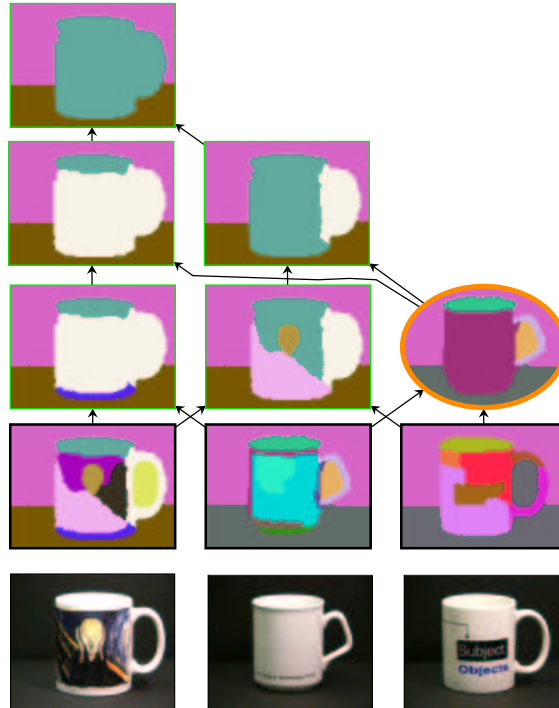
**Fig. 3.** Finding the lowest common abstraction between two exemplars through a coordinated, recursive decomposition of their silhouettes. Figure taken from [3], copyright IEEE, 2005.

are similar. When a successful cut is found, the corresponding shapes represent a many-to-many matching of component regions.

The process continues recursively down the intersection lattice until no further decomposition is possible. The union of primitive shapes forms the LCA of the two graphs. This procedure is applied to all pairs of input exemplar graphs, leading to a weak approximation of the intersection lattice for all inputs. Finally, we search this approximation for the global lowest common abstraction. Figure 4 shows the LCA computed for three different coffee cup exemplars and the resulting intersection lattice; the computed global LCA is highlighted in orange and represents a good abstraction of the cup, including a handle, body, top, and hole (which it can't distinguish from a surface). Details of the algorithm can be found in [3].

### 3 Matching Structural Abstractions

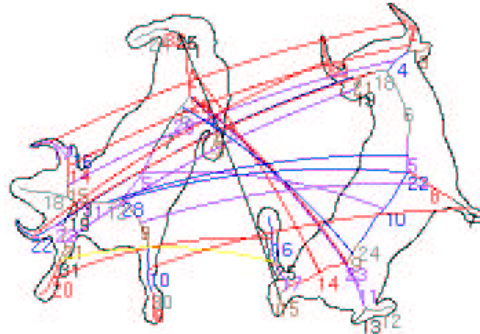
The previous formulation of categorical model acquisition from examples searched for abstractions (groupings) of regions (features) that were similar in shape and common to many input lattices. The intractable complexity of the resulting many-to-many graph matching problem led to an effective approximation method that yielded matching abstractions among pairs. However, the abstractions were still graphs, and isomorphism had to be effectively computed. One way of reducing the complexity of graph matching in the presence of noise (spurious nodes and edges), occlusion (both extraneous and missing structure), and



**Fig. 4.** Computed LCA (orange border) of 3 examples. From a set of three region adjacency graphs representing the appearance of three different cup exemplars, the algorithm extracts the salient surfaces of the cup, including the body, inside surface (ellipse at top), handle, and hole (indistinguishable from a real surface). Figure taken from [3], copyright IEEE, 2005.

minor deformation (leading to graphs of different size) is not to match the graphs themselves but rather low-dimensional vector abstractions of the graphs, each of which encodes the “shape” of a subgraph. Drawing on the domain of spectral graph theory, we have developed a low-dimensional abstraction of directed acyclic graph structure that is robust to noise and perturbation yet rich enough to distinguish structural differences between graphs. It assigns a “structural signature” vector to each node, encoding the “shape” of the underlying subgraph rooted at that node. In a hierarchical structure, where nodes reflect a coarse-to-fine feature decomposition, these vectors provide us with a powerful tool for coarse-to-fine object recognition.

Our vector abstraction of graph structure is a function of the magnitudes of the eigenvalues of a directed graph’s underlying adjacency matrix. The eigenvalues encode the degree distribution of the graph and are provably robust to minor perturbation of the graph. Moreover, the dimensionality of the resulting vectors is bounded by the maximum branching factor of the graph and not by



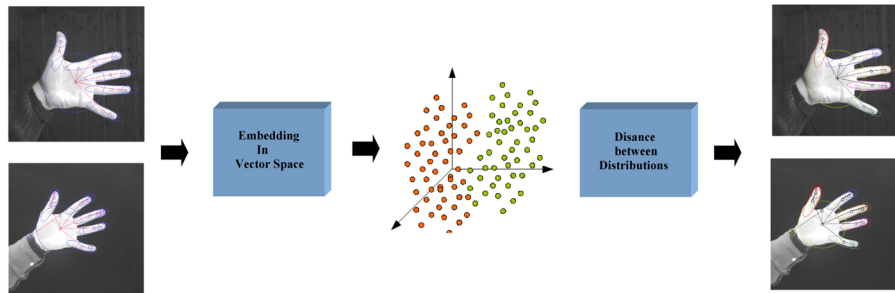
**Fig. 5.** Part correspondences in two Shock Graphs. Each colored branch in the shape's skeleton represents a node in the shape's hierarchical shock graph. Node correspondences are shown, with correspondences at coarser levels of the hierarchy implicitly defining many-to-many correspondences among substructures at finer levels.

the number of nodes in the graph. Not only do these structural signature vectors offer an effective mechanism for rapid indexing from large databases of graphs [7], but they offer a mechanism for matching graph abstractions [8,6,5]. If the vectors computed for two nodes of two directed acyclic graphs are similar, then their underlying subgraphs have similar structure, i.e., the collection of nodes forming the subgraph rooted at a node in one graph matches a collection of nodes forming the subgraph rooted at a node in a second graph, effectively yielding a many-to-many node (and structure) correspondence. Figure 5 indicates the part correspondence computed for two silhouettes, each represented using a *shock graph* [8], a qualitative decomposition of a silhouette's medial axis structure in terms of a vocabulary of qualitatively defined parts.

#### 4 Structural Matching as Weighted Point Matching

Our first problem formulation could be seen as computing a set of many-to-many node correspondences across a set of input graphs. However, the graphs were known to belong to the same class, i.e., isomorphic at some level of abstraction. Our second problem formulation was more general in that it computed correspondences between nodes whose underlying (rooted) subgraphs have similar structure. The vector abstraction of graph structure implicitly defined a many-to-many matching of the underlying nodes without explicitly computing node correspondences. In this section, we look at the most general formulation of the many-to-many matching problem. Specifically, how do we match two graphs whose structure may be different (as in our first problem) but which may not belong to the same class (as in our second problem)?

Many-to-many graph matching is intractable, for any subset of nodes in one graph may correspond to any subset of nodes in the other graph. Our approach, depicted in Figure 6, transforms the many-to-many graph matching problem

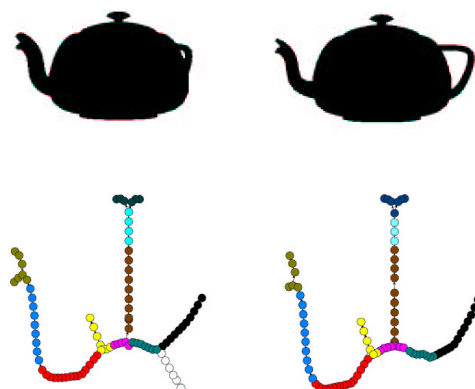


**Fig. 6.** Many-to-Many Graph Matching as Weighted Geometric Point Matching. Two graphs to be matched are embedded with low distortion into a low-dimensional Euclidean space, the two weighted sets of points are matched many-to-many using the Earth Mover’s Distance (EMD) algorithm, and the solution defines a many-to-many node correspondence in the original graphs.

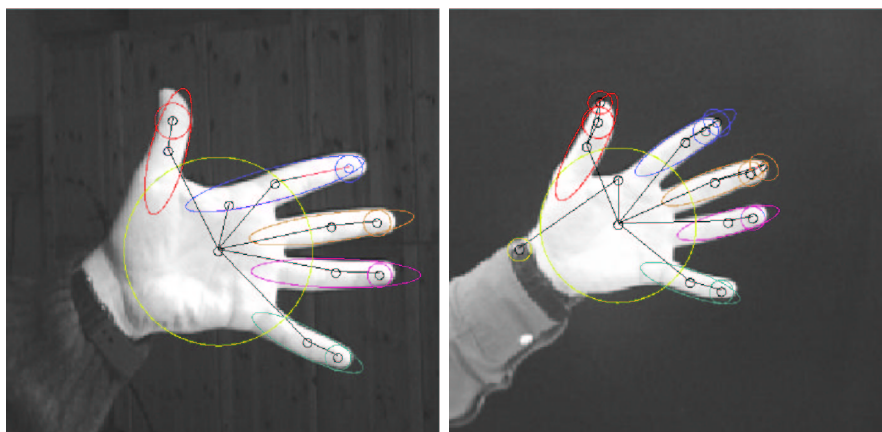
into a domain in which many-to-many matching is easier. Specifically, in working with edge-weighted, attributed graphs, each node maps to a point in a low-dimensional geometric space, with the node’s attributes mapping to a mass vector assigned to the point. The structure of the graph is effectively mapped (embedded) into the geometric space by ensuring that the shortest path distances (along graph edges, summing edge weights) between pairs of nodes in the original graph are preserved (with low distortion) as Euclidean distances between their corresponding points in the transformed space; the edges of the graph are effectively discarded.

The above graph embedding framework allows us to translate our original many-to-many graph matching problem to a many-to-many point matching problem. If we let one graph’s points (embedded nodes) be represented as piles of dirt, each of whose volume is proportional to the node’s mass, and the other graph’s points be represented as holes, each of whose volume is proportional to the node’s mass, the many-to-many weighted point matching problem can be formulated as finding the assignment of dirt to holes that minimizes the work required to move the dirt. The Earth Mover’s Distance (EMD) algorithm [1] provides an efficient solution, allowing a pile of dirt to be spread across multiple holes and allowing a hole to receive dirt from multiple piles. Moreover, the computed flows between piles and holes can be translated back to the original problem, yielding a many-to-many node correspondence between the original graphs. Details can be found in [4,2].

Figures 7 and 8 illustrate two example domains in which we have successfully applied the approach. In Figure 7, we compute many-to-many node correspondences between two silhouettes represented as skeleton graphs, in which nodes represent medial axis points, edge weights represent Euclidean distances between connected points, and node masses represent the radii of the maximally inscribed circles. The algorithm computes the many-to-many correspondence shown below the silhouettes, with corresponding points colored similarly. In Figure 8, we com-



**Fig. 7.** Many-to-Many Graph Matching Applied to Skeleton Graphs. Corresponding groups of nodes (whose cardinalities may be different) are colored the same, with white nodes unmatched. Figure taken from [4], copyright IEEE, 2003.



**Fig. 8.** Many-to-Many Graph Matching Applied to Blob Graphs. Many-to-many feature correspondences have been colored the same. Corresponding groups of nodes (whose cardinalities may be different) are colored the same. Figure taken from [2] (pg. 332), copyright, Springer, 2004, used with kind permission of Springer Science and Business Media.

pute many-to-many node correspondences between two hand images represented as hierarchical blob and ridge decompositions [6]; note that the number of blobs used to model the fingers or palm differs between the two decompositions. The algorithm computes the many-to-many correspondences indicated by the similar coloring.

## 5 Conclusions

The within-class shape and appearance variation of many object categories precludes recognition strategies that assume a one-to-one correspondence between low-level image features, for it is only at higher levels of abstraction that similarity or correspondence may exist. Image abstraction is an open problem, and it's not clear what form image abstractions should take or how to compute them. In one part of this paper, we review a particular graph (structural description) abstraction and apply it to matching structural descriptions. In a second part of this paper, we use the fact that two objects belong to the same category to help search for a common abstraction. In the final part of the paper, we map structural descriptions to be matched into a geometric space in which rules (i.e., edge weights) for abstracting/grouping features in graph space can be exploited by computationally tractable strategies (e.g., EMD) in geometric space to find corresponding abstractions. Each of these formulations effectively addresses the underlying problem of many-to-many object matching, an important challenge to categorization systems.

## References

1. S. D. Cohen and L. J. Guibas. The earth mover's distance under transformation sets. In *Proceedings, 7th ICCV*, Kerkyra, Greece, 1999.
2. M. Demirci, A. Shokoufandeh, S. Dickinson, Y. Keselman, , and L. Bretzner. Many-to-many feature matching using spherical coding of directed graphs. In *Proceedings, ECCV*, Prague, 2004.
3. Y. Keselman and S. Dickinson. Generic model abstraction from examples. *IEEE PAMI*, 27(7), 2005.
4. Y. Keselman, A. Shokoufandeh, M. Demirci, , and S. Dickinson. Many-to-many graph matching via metric embedding. In *Proceedings, IEEE CVPR*, Madison, WI, 2003.
5. D. Macrini, A. Shokoufandeh, S. Dickinson, K. Siddiqi, and S. Zucker. View-based 3-D object recognition using shock graphs. In *Proceedings, Internal Conference on Pattern Recognition*, pages 24–28, Quebec City, August 2002.
6. A. Shokoufandeh, S. Dickinson, C. Jönsson, L. Bretzner, and T. Lindeberg. The representation and matching of qualitative shape at multiple scales. In *Proceedings, ECCV 2002*, pages 759–772, Copenhagen, 2002.
7. A. Shokoufandeh, D. Macrini, S. Dickinson, K. Siddiqi, , and S. Zucker. Indexing hierarchical structures using graph spectra. *IEEE PAMI*, 27(7), 2005.
8. K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 30:1–24, 1999.