Graph Based Shape Representations for Object Detection

Jeffrey Byrne
University of Pennsylvania
GRASP Laboratory, 3330 Walnut Street
Philadelphia, PA 19104-6389
jebyrne@cis.upenn.edu

Abstract

Shape is arguably the most important property in visual perception, since unlike other properties such as texture, color, motion or depth, shape alone can be used for visual tasks such as object categorization and object detection. In this report, we survey graph based shape representations, which are well suited for modelling part based compositions of objects. We organize the survey by grouping graph based representations into geometric methods and topological methods, where each method is organized by representations of increasing abstraction. For geometric methods, we use the unifying framework of weighted graph matching posed as relaxations of a quadratic assignment problem, and we describe invariant shape properties maintained during various tree, bipartite and general graph matching approximations. Topological methods for shape representation are less well established for object detection, but they provide the potential for global constraints such as interior, surrounded and connected to augment geometric representations. We use the unifying framework of simplicial homology, and describe the persistent homology, a technique for recovering the homology given noisy data, and optimal homologous cycle matching for matching topologically invariant cycles. Finally, we compare and contrast these methods both in a task independent analysis based on graph representational power and computational tractability, and a task dependent analysis for suitability of this representation for object detection.

Contents

Figure 1: Master draughtsmen and the use of shape in art. (left) “Woman Covering Her Face with her Hand” by Matisse, (middle) “A Woman Sleeping” by Rembrandt, (right) “Three Studies of a Dancer” by Degas.

1 Introduction

Shape is arguably the most important property in visual perception [1]. Shape is the primary property used for visual categorization, and unlike other properties such as texture, color, motion or depth, shape alone can be used to predict other category properties. Central to the discussion of shape in a computational vision context, is the *shape representation*. The issue of representation is a fundamental problem in vision, leading many to argue that representation and generalization of shape is *the* problem in vision [2][3][4][5].

Artists have long known the power of shape for capturing visual form. Master draughtsmen such as Matisse, Rembrandt and Degas have captured such intangible qualities as strength, solitude or elegance in only a few well chosen strokes. Figure 1 shows some examples of the masters at work, where using only contours, they are capture the subtleties of the human form. Our perception of these qualities would not be any clearer by adding color to Matisse’s “Woman Covering Her Face With Her Hand”, or adding texture to Rembrandt’s “A Woman Sleeping”, or knowing the range to arms and legs of “Three Studies of a Dancer” by Degas. The artists knew that shape alone can capture these qualities, and that it is a powerful cue to inform our perception of the world.

What is a shape? Shape is an intuitive concept for most people since we perceive shapes every moment of the day, however it is difficult to define unambiguously, which has led to different operational definitions in different fields. For example, cognitive scientists define shape in terms of *objective shape* and *shape equivalence* [6]. Objective shape refers to the concept that objects in the world have a measurable volume, they have a surface boundary with a measurable surface orientation independent of any observer. Marr defines objective shape as “the geometry of an object’s physical surface” [7] and Palmer observes “objective shape is no different in principle from the well established belief that each object has an objectively definable size, position, orientation” [1]. These objects have 3D extent that is the same for all observers, and therefore objectively definable independent of perception. Intuitively, one can think of the objective shape in terms

of computer graphics, where the objective shape of an object is that which is captured by a 3D model, used to render an image independent of the camera viewing it. *Shape equivalence* refers to those shapes that are perceived to be the same object by observers. For example, viewing a 2D square that undergoes a translation is still perceived to be a square. The same holds for a scaling or small in-plane rotation. Observers viewing such objects that undergo transformations of translation, rotation or scaling perceive the same object following the transformation. The shape before the transformation is equivalent to the shape after. In general, shape equivalence refers to the variation due to the pose of the viewer relative to a fixed objective shape. A viewer can change position (translation) move towards or away from (scale) or tilt their head (rotation), and "the shape" perceived remains equivalent, even though the retinal image may change significantly.

Statisticians define shape in terms of a *statistical shape model* [8]. Different views of a shape are assumed to have common keypoints such that these *landmarks* can be put into correspondence. Given a set of shape images, a shape model is that which remains after optimally aligning landmarks, when the translation, rotation and scale effects are removed. Alignment proceeds using *Procrustes analysis*, which performs an optimal estimate of the aligning transform for a set of known landmarks to a reference coordinate system. This assumes correspondence of a known set of shape landmarks, which have been extracted from an image, such as points of high curvature. Then, once landmarks are in correspondence, deformation models can be learned from the remaining (non-similarity) alignment errors. In computer vision, this type of analysis has led to active shape models [9][10] for shape based models of faces.

Geometry defines shape in terms of an equivalence class under a group of transformations. A *group* is a finite or infinite set of elements along with a binary group operation that satisfies closure, associativity, identity and inverse properties. In the context of shape, the group is the set $\mathbb{R}^2$ under similarity transformations. A similarity transformation is a group operator $T : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ such that for any set of points $p \subset \mathbb{R}^2$, the points $T(p)$ are *similar* or related by a similarity. Formally, $||p_i - p_j|| \Leftrightarrow \alpha||T(p_i) - T(p_j)||$ for all $p \in \mathbb{R}^2, \alpha \in \mathbb{R}$. In other words, for a similarity transform, angles are preserved and distances are preserved up to a scalar scale factor.

Computer vision does not provide one definition of shape, rather the literature provides many working definitions of shape, each with experimental evaluation of the proposed representation. Shape representations can be broadly organized into two main categories: template based and graph based. *Template based* approaches represent shape in terms of a fixed or deformable template, where a template is a fixed, relative spatial distribution of features [11, 12, 13, 14, 15, 16, 17, 18, 9, 19]. *Graph based* approaches represent shape in terms of graphs, where features or parts have a variable spatial distribution organized in a graphical structure [20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31]. Both template and graph based approaches represent the geometry of an object in terms of the relative position of features, however a template assumes a fixed relative position with small allowable deformations while a graph assumes a variable relative position of parts encoded by the graph. It is arguable that due to this parts based representation, a graph based representation provides more expressive power for shape representation, and the cost of more expensive inference.

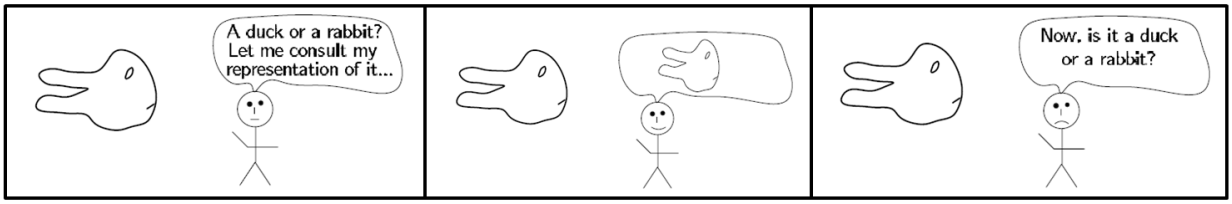


Figure 2: Wittgenstein's Joke [4]. This highlights the need for interpretation to create a useful representation of a stimulus, or on this context, why not use a 3D representation of a duck/rabbit?

In the past fifteen years, there has been significant progress in shape representations in the literature, as described in surveys [32][33][34]. In this review, we focus on graph based shape representations for object detection. This differentiates this review from the existing literature by reviewing a specific representation (graph based shape) for a specific task (object detection).

1.1 Scope

What is the scope of this review? As the title alludes, our goal is to perform a survey of graph based shape representations for object detection, however this goal is still quite a broad. Are we interested in 2D or 3D shape? Local shape representations in terms of invariant local features? Global representations, part based or category structure? Are we interested in shape registration, matching or shape similarity algorithms, and data structures associated with efficient solutions? How about classification and learning of shape within a category? What about other non-shape or non-visually grounded features such as texture, motion and color? What about the shape of a scene?

We limit the scope of the review to the following design choices:

Graph based: We focus on graph based representations rather than template based representations. The line between these two representations is often indistinct, as many graph based presentations use local templates for representation of parts, and some deformable template representations can be modelled as a graph such as a Markov random field. Where appropriate, we discuss the overlap between these broad categorizations.

Graph matching: We focus on graph matching, also called registration, or alignment correspondence, rather than graph similarity. Graph matching provides alignment between a reference model and an observation such that edge relations (e.g. geometry or topology) are preserved. This matching may include matching constraints such as one-to-one, many-to-one or many to many matching to reflect abstract representations of shape [5]. Recent work on *graph kernels* [35] for structured prediction compute the similarity of two general graphs by considering the similarity of polynomial comparable substructures such as trees, cycles, walks or paths. These approaches are motivated

by the intractability of subgraph isomorphism for general graphs (see section 3), and the need for an efficient computation of similarity of two graphs suitable for use in a kernel machine for discriminative classification. These approaches do not provide correspondence and therefore do not represent global geometric shape during matching. These approaches do consider local geometric shape in terms of matching substructures, but the final similarity is analogous to a "bag of substructures" representation which is not global. Therefore, we focus on geometric and topological matching rather than non-geometric comparisons.

Exemplar similarity: We focus on similarity computations to exemplars [2][3][4][36], rather than category models. We will argue in section 5.4 that any fixed category model, even category models learned at training time, is classical categorization revisited. Therefore, since the classical theory of categorization has been widely discredited, a category model of shape stands on a questionable foundation. As a result, we will not consider graph based shape representations for categories such as recursive compositional models [25, 37], hierarchical generative models [38], hierarchical object parsing [18, 39], stochastic image grammars [23], composition systems [24, 21] or probabilistic graphical models [40, 41, 42]. This effectively descopes the entire literature of generative models for visual category representation. Note that this does not mean that categorization cannot be performed, it simply states that categorization that relies on a fixed category model is questionable.

2D: We focus on 2D shape or view based representations rather than representation of 3D object shape. A large body of literature exists on "shape from X" recovery of 3D shape from imagery. Edelman [4] traces the history of 3D reconstruction as a shape representation, back to the influential work of Marr [7] and the 2.5D sketch. The motivation for 3D representations was to provide *object constancy* or viewpoint invariance for specific object identification across views, by first reconstructing the 3D geometry from multiple views. However, Edelman argues that 3D shape representations still suffer from drawbacks namely: (i) the difficulty of recovering correspondence, (ii) the need for task specific representations and (iii) the fact that 3D representations do not aid in further higher level processing such as categorization or detection. As observed by many researchers, a 3D representation still needs to be interpreted. Figure 2 shows an old joke due to Wittgenstein. A 3D representation requires a "homonculus" or little man sitting inside our vision box looking at the 3D representation for interpretation [4]. In short, a 3D reconstruction still requires *similarity* to an internal labeled 3D representation, on top of the challenge of 3D reconstruction itself. Therefore, we focus on 2D representations and the challenge of similarity.

Object detection: Optimal shape representations are task dependent [5][4]. We focus on the representation for the task of object detection. Object detection is the problem of assigning a bounding box at a position and scale that surrounds all each instance of a given object category in an image. Object detection is a localization problem, and can be compared with other related tasks. Object classification is the labelling of images that contain at least one instance of a given object category without localization. Object recognition [43], also called object identification or specific object recognition is object classification for a known unique object instead of an object

category. Object segmentation is the grouping and labelling of pixels corresponding to foreground and background for each instance of a given object category in an image.

Global representations: There are excellent surveys available for local feature descriptors [44, 45], which capture the local shape of an object. Many of the approaches in this survey rely on local feature descriptors as a foundation for composing a global graph based representation, however we will focus more on the global graph structure rather than the local descriptors composing them.

Basic level categories: We focus on shape representations for object detection of basic level categories [46, 47, 5]. Basic level categories are the highest level category for which members have similar shape, and does not include such categories as functional objects or scenes. There have been surveys for shape representations of popular object categories including faces [48] and human body [49], however a general shape representation for basic level categories should not be specialized for any one category.

1.2 Outline

There are many different ways of organizing and categorizing the literature on shape representations, such that each focuses on a different unifying theme. For example, we could focus on shape matching, comparing and contrasting algorithmic complexity and data structures associated with different frameworks. We could focus shape representations only, comparing features and parts extracted from imagery used to compose shapes. We could compare and contrast graph based methods with template based methods or texture based methods.

The structure of this review is graphical representations of shape of increasing abstraction and invariance. We group the problem into two major categories of graph based shape representations: geometric and topological graph based representations. *Geometric graph based representations* represent shape in terms of distances and angles between parts, such that nodes encode parts and edges encode geometric relations. This approach can be organized in terms of the underlying graph structure. We describe methods based on trees, bipartite graphs and general graphs, where the underlying graphical structure allows for more expressive representations of shape, at the cost of more expensive matching. *Topological representations* represent shape in terms of topological invariants such as connectivity or holes in a graph. These representations are less well evaluated for shape representations in computer vision, so we survey the computational topology literature and describe approaches based on persistent homology and homologous cycle matching that have the potential to provide invariant features for shape representation.

In this review, section Section 2 describe graph based representations which represent shape in terms of a graphs and describes common components to geometric and topological shape representations. Section 3 describes three graphical structures: trees, bipartite graphs and general graphs and the graph constructions and graph matchings used for representation and detection. Section 4 describes topology based representations which represent shape in terms of homologies. Finally,

we compare and contrast the different categories and draw conclusions for about the representations in sections 5 - 6.

This survey is also structured to satisfy the requirement for a WPE-II. Each major section has at least one paper that is described in detail which serves as a representative approach for the section. These papers serves as the primary references for the WPE-II critical review. For graph matching, we describe the implicit shape model which uses a star graph representation and generalized hough transform for shape matching [26, 50]. For arbitrary graphs, we describe graduated assignment algorithm [51] for convex relaxation of graph matching. For shape representations using topological methods, we describe persistent homology and optimal homologous cycle matching [52]. These cited papers are the primary references for the WPE-II.

2 Graphical Representations of Shape

Graphical representations of shape refer to the abstraction of an image into an *attributed graph*, such that the image encodes a *graph embedding*. A grayscale image I may be defined as a function $I : \mathbb{Z}^2 \rightarrow \mathbb{R}$, such that I is a mapping between integer valued pixel coordinates (i, j) and pixel intensity $I(i, j)$. A graph embedding in an image is a graph $G = (V, E)$ (see defn 2.1) such that each node is associated with a pixel coordinate (i, j). An attributed graph is a graph $G = (V, E, \alpha)$ that has been augmented with a set of node and edge attributes $\alpha_V(v)$ $\alpha_E(u, v)$ for all $v \in V, (u, v) \in E$. These attributes encode local image properties from the graph embedding. Recent work has considered attributes for higher order simplexes in hypergraphs [53], however in this section we consider pairwise node and edge attributes only, and postpone the discussion of higher order simplex attributes to section 4.

Graph based shape representations for can be described in terms of attributes, structure, construction and matching. Attributes refer to those local image properties associated with each node and edge as determined from the graph embedding. Node attributes may be organized in order of increasing support in an image, centered at the embedding coordinates. *Pixel support* refers to attributes such as pixel intensity, oriented gradient filter response or corner response at only the embedding coordinate. *Patch support* refers to attributes derived from a fixed, local region of interest centered at the embedding coordinate, such as a grayscale patch, intensity histogram or oriented gradient histogram. *Region or contour support* refers to attributes derived from perceptual organization of an image, such as segmentations or boundary detections. Finally, *part support* refers to attributes derived from part responses, such that a part is itself an object with shape and is used in compositional models to compose shape in terms of simpler component shapes.

Edge attributes capture pairwise geometric or topological relationships between nodes. For example, geometric edge attributes may include length between two nodes, the angle between a node and a reference orientation, or the scale normalized length between two nodes. As discussed in section 1.1:similarity, we consider only attributed graphs in this review. Other methods such as probabilistic graphical models enable efficient inference techniques by encoding *conditional*

dependence in edge relations, however as discussed we will not consider these cases. We will revisit this issue in section 5.

Graph construction is the process of graph embedding and attribute extraction, and graphs may be explicitly or implicitly constructed. Explicit construction is the computation of a graph from an image, prior to any processing or matching. Implicit construction postpones the graph constructing to the matching phase. Implicit construction follows Marr's principle of least commitment [7], such that graph embedding decisions for nodes and edges are delayed until there is further information from the matching process, rather than committing to mistaken embedding and corrupting the match. Recent work in graph matching has transitioned from explicit to implicit graph construction, and we will use this as a comparison criterion in section 5.

Graph matching is the problem of finding correspondences between two graphs such that relational structure is preserved. We will discuss this problem in context of the unifying framework of the *quadratic assignment problem* in section 2.2 following standard definitions of graphs and graph properties in section 2.1.

2.1 Graph Definitions

We begin with standard definitions from graph theory [54]. Intuitively, a *graph* is an abstract representation of a set of objects where some objects are connected by links or edges. More formally,

Definition 2.1. A graph G is a pair (V,E) of sets satisfying $E \subseteq V \times V$

Elements E are 2-element subsets of V formed from the Cartesian product of V. The elements of set V are *vertices* and elements of E are *edges*. Two vertices (u,v) are *adjacent* if there exists an edge $(u,v) \in E$. Vertices (u,v) are *incident* with an edge e if $(u,v) \in E$. The *order* of G is $|V|$ or the number of vertices. The *degree* of a node $v \in V$ is the number of incident edges. A graph G' is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$.

Definition 2.2. An *attributed graph* is a graph $G = (V,E,\alpha)$ that has been augmented with a set of node and edge attributes $\alpha = \{\alpha_V, \alpha_E\}$ such that $v \in V, (u,v) \in E$ there exists node attributes $\alpha_V(v)$ and edge attributes $\alpha_E(u,v)$.

Definition 2.3. A graph $G = (V,E)$ is *bipartite* if and only if V can be partitioned into two disjoint subsets V_1 and V_2 such that $V = V_1 \bigcup V_2$, $V_1 \cap V_2 = \emptyset$ and adjacency vertices (u,v) are in different subsets $u \in V_1$ and $v \in V_2$.

Definition 2.4. A *path* in G is a subgraph $P = (V,E)$ such that $V = \{u_0, u_1, \ldots, u_k\}$ and $E = \{(u_0,u_1),(u_1,u_2),\ldots,(u_{k-1},u_k)\}$

A graph is *connected* if there exists a path *linking* any two vertices. The *length* of a path P is the order of P.

Definition 2.5. A *cycle* in G is a path P such that $u_0 = u_k$.

A graph is *acyclic* if it contains no cycles.

Definition 2.6. A graph $G = (V,E)$ is a tree T if and only if any of the following properties hold.

- Any two vertices of T are linked by a unique path in T
- T is minimally connected, such that removing any edge e results in a disconnected graph
- T is maximally acyclic, such that T contains no cycles and adding an edge e between any two non-adjacent vertices will introduce a cycle

Note that definition 2.6 is in fact a standard theorem of graph theory and not a definition, however we state it here without proof and refer the reader to [54]. The *diameter* of a tree is the maximum length of the shortest path between any two vertices.

Definition 2.7. A tree is a *star* if and only if either the following two properties holds:

- The tree has diameter 2.
- The tree $T = (V,E)$ is a complete bipartite graph such that nodes are partitioned into disjoint sets $V_1 = \{v_0\}$ and $V_2 = V \backslash v_o$, and the cardinality $|V_1| = 1$.

Definition 2.8. Graphs $G = (V,E)$ and $G' = (V',E')$ are *isomorphic* if and only if there exists a bijection $\phi : V \to V'$ with $(u,v) \in E \Leftrightarrow (\phi(u),\phi(v)) \in E'$ for all $u,v \in V$

where ϕ is a functional mapping from nodes in G to nodes in G' that is one to one and onto, and preserve incidence.

Definition 2.9. Graphs $G = (V,E)$ and $G' = (V',E')$ are *homomorphic* if and only if there exists an surjection $\phi : V \to V'$ with $(u,v) \in E \Leftrightarrow (\phi(u),\phi(v)) \in E'$

where ϕ is a functional mapping from nodes in G to nodes in G' that is onto, and preserve incidence.

2.2 Quadratic Assignment Problem

Graph matching is the problem of finding correspondences between two graphs such that relational structure is preserved. This is a fundamental problem in computer vision, machine learning and pattern recognition since structured data in widespread in such forms as part based object recognition, structured prediction, and shape representations. For recent surveys of graph matching, see [55, 56]. In general, graph matching can be posed as a weighted graph matching problem, with special cases of subgraph isomorphism and maximum common subgraph.

Weighted graph matching can be posed as follows. Given two attributed graphs $G = (V,E,\alpha)$, $G' = (V',E',\alpha')$, let X be an $|V| \times |V'|$ permutation matrix, such that $X(i,i') = 1$ if nodes (i,i') are matched and zero otherwise. Let W be an $|V||V'| \times |V||V'|$ weight matrix determined from attributes (α,α') such that $w_{ii',jj'} \in \mathbb{R}$ encodes the compatibility of matching (i,i') and (j,j'). Let

x be an an $|V||V'| \times 1$ columwise vector representation of X such that $x_{ij} = X(i,j)$ and x_i^T is the ith row and x_j is the j^{th} column. Then,

$$\begin{aligned} x^*_{QAP} = \arg\max \quad & x^T W x \\ \text{s.t.} \forall (i,j) \quad & \mathbb{1}^T x_j = m \\ & x_i^T \mathbb{1} = n \\ & x_{ij} \in \{0,1\} \end{aligned} \tag{1}$$

The constraints $\mathbb{1}^T x_j = m$ and $x_i^T \mathbb{1} = n$ are *mapping constraints*. Let $\mathbb{1}$ be a vector of ones, then $\mathbb{1}^T x_j$ is a column sum for columns x_j of X, and $x_i^T \mathbb{1}$ is a row sum of X. If $m = n = 1$, then the mapping constraints are *one-to-one* such that each node in G must be mapped to exactly one node in G. If $n \geq 1$ and $m = 1$, then the mapping constraints are *many-to-one* for many nodes in G' to one node in G'. Similarly, if $n \geq 1$ and $m \geq 1$ then the mapping constraints are *many-to-many*. These mapping constraints enable the graph matching to encode constraints such as isomorphism or homomorphism, and allows the graph matching to be robust to imperfect graph construction.

The optimization in (1) is an instance of a *quadratic assignment problem*, such that x^*_{QAP} is a maximum weight edge preserving matching where $(u,v) \in E \Leftrightarrow (X(u),X(v)) \in E'$ [57]. The quadratic assignment problem is a classic problem in combinatorial optimization that can be motivated as a *facilities localization* problem. Consider the problem of assigning a given set of facilities to locations, where there are costs for a given assignment due to the cost of the flow of goods between facilities and the costs of assigning a facility at a given location. The goal is to determine an optimal assignment given these costs. Formally, given N facilities and M locations, let A be an $N \times N$ matrix defining the weight between facilities, and B be an $M \times M$ matrix for weights between locations. Solve for an assignment x that minimizes (1) such that the cost of assigning facility i to location j is W_{ij}.

The optimization in (1) is an integer quadratic program which is NP-complete, so approximate solutions are necessary. Approximation algorithms that have been explored in the literature include combinatorial search [57], graduated assignment [51], spectral [58, 59], semidefinite programming [60], and graph edit distance [61]. These approaches require a construction of the quadratic objective weights W which is quadratic in the size $|V||V'|$. Robust performance has been demonstrated [51, 59], but practical problem sizes are limited to hundreds of nodes due to the quadratic objective, and weights are limited to pairwise interactions.

The optimization in (1) can be made efficient for constrained graph structures. For example, if the graph G is bipartite, then the quadratic assignment problem reduces to the *linear assignment problem* for which polynomial time integer solutions are available [28]. Similarly, if the graph G is a tree, then there exist tree edit distance algorithms [62] based on dynamic programming [63][64].

3 Geometric Graph Representations

Graph representations model an object or object category using a *graph*, such that object detection is isomorphic to *graph matching*. In general, graph matching approaches can be described in terms of the structure being preserved. Given two attributed graphs $G = (V, E, \alpha)$, $G' = (V', E', \alpha')$ a structure preserving matching $f : V \rightarrow V'$ is an optimal solution $f^* = argmin\, c(f)$, subject to structure preserving matching constraints and assignment costs c. Exact structure preserving methods, such as subgraph isomorphism, maximum common subgraph and weighted graph matching preserve edge relations, such that $(u, v) \in E \Leftrightarrow (f(u), f(v)) \in E'$. In contrast, inexact graph matching, such as graph edit distance problem [61, 65] require approximate solutions.

Recent work in the vision literature has focused on geometry preserving linearizations [27, 66, 67] of the quadratic assignment problem in (1). Similarity invariant matching [66] solves for the optimal permutation matrix X and linearized similarity transformation parameters θ to minimize an assignment cost and an $L1$-norm linear deformation cost. Locally affine invariant matching [67] solves for the optimal assignment X given $L1$-norm barycentric coordinate preservation costs for each node, where barycentric coordinates are locally affine invariant and defined in terms of neighboring graph nodes. Both approaches use an $L1$-norm in the objective, and exhibit linear constraints ([66] includes a linearization of the similarity constraints) resulting in a linear programming relaxation.

These geometric approaches provide fast and efficient matching, but they can suffer from ambiguity when the input graph does not satisfy the assumptions of the geometric transform model, such as cases of non-similarity transformations or degenerate triangulations. Furthermore, assignment weights are limited to node assignment weights only, ignoring informative assignment weights for edges and other higher order structures [53]. Finally, these methods must discretize X to a final binary permutation matrix for valid and invalid matches. Poor geometric alignments with large deformation costs may still be valid (e.g. articulated objects), and good geometric alignments with small deformations may be invalid. These issues will be revisited in the topological methods in section 4.

Approaches to graph matching can be compared using graph construction and graph topology. Graph construction refers to the approach used to create an attributed graph that represents an image, including constructing nodes and edges and assigning attributes. Graph topology refers to the structure of a graph, such as trees, bipartite or general graphs. The graph structure is closely related to the efficiency and optimality of the matching, so there are tradeoffs between the fidelity of the representation vs. the optimality of the matching. In this section, we will describe three different graph structures and describe representative approaches for each.

3.1 Trees

Trees provide a useful abstraction to provide *part based representations* of shape. A part based representation of shape decomposes an object into a discrete set of component subshapes or *parts*

that are configured to create an object. Parts may be large and sparse such as the decomposition of a face into semantic parts such as eyes, nose and mouth. Alternatively, parts may be small and dense such as contour fragments composed into a holistic shape representation. However, in all part based representations, local parts are *composed* provide a structural decomposition of a global representation of shape into a configuration of local parts. Recall from section 2.1, that a graph $G = (V, E)$ is a tree if it is connected and acyclic. The nodes of a tree represent parts, the edges of the tree represent a composition of discrete and independent parts into a whole.

The motivation for a part based representation is invariance, compositionality, reuse, and computational tractability [5][1]. By decomposing an object into a set of parts that can be recomposed into multiple different objects in different configurations, parts can be reused to represent many object shapes. Similarly, by decomposing parts into a tree based or hierarchical representation or hierarchy of parts, the tree structure can enable efficient matching by taking advantage of the acyclic graphical structure.

Part based representations can be described in terms of part representations, structural representations and detection framework. Part representations capture the local appearance or geometry of a small subset of an object such that each part captures some local property of an object. Part representations may be appearance based or contour based, local or global, invariant (affine, rotation) or non-invariant (patches). Structural representations describe the tree structure, which may be flat as in the case of constellation or star trees, or hierarchical with multiple levels of part interactions. Finally, the detection framework considers how the part representations are detected in an image, localized and composed into an object. Optimization approaches include dynamic programming, belief propagation, generalized hough transform and graph matching.

In this section, we discuss in detail an approach to part based representation of shape that are representative of *star graphs* structural representations and *generalized hough transform* detection methods. These approaches represent shape using star graphs, a special case of trees, as the part based shape representations. First, we discuss the *implicit shape model*, and approach to part based object categorization that decomposes an object into a constellation of parts around an object center. This approach learns appearance and relative position of part prototypes and uses a generalized hough transform for detection. Second, we describe the max margin hough transform, an extension of the implicit shape model to learn discriminative part weights for improved voting.

3.1.1 Implicit Shape Model

The *implicit shape model* is an example of part based object categorization that is representative of approaches based on the generalized Hough transform. The implicit shape model was developed for categorization and top down segmentation [26, 50] and extended to a more principled probabilistic foundation [31] and improved discriminative part weights [68, 69]. This approach has been influential in developing and deploying part based models and is among the highest performing shape based detection and recognition approaches in the literature. This approach is well known, popular and effective, and is representative of other GHT-based approaches in the literature

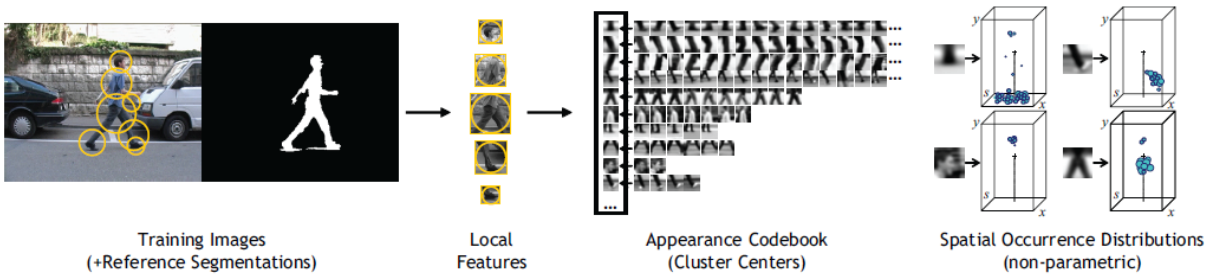


Figure 3: Implicit shape model [50]: Training

[70, 71, 72]. In this section, we sill describe the implicit shape model in more detail.

3.1.1.1 Overview The implicit shape model (ISM) is concerned primarily with object detection for basic level categories. As will be discussed in section 5.4, categorization at the basic level assumes that all elements in the same category share visually grounded attributes, including common shape features. During training, categories are labelled such that similar views of an object class, such as profile views of cars, are labelled together. This means that each category contains examples of similar shape features that can be used for representation of the object category. This observation serves as the foundation for the implicit shape model. Objects in a category are analyzed to determine a finite set of *part prototypes*, such that each prototype describes the local appearance or geometry of an object. For example, in a basic category or "profile views of cars", one part prototype may capture a wheel and another may capture the hood. Since the objects share shape features, the learned prototypes can be used to capture the variability of the basic category in terms of the spatial distribution of prototypes.

The implicit shape model uses a generalized hough transform for object detection. As discussed below, shape prototypes are computed by clustering to capture the local shape of an image, then each interest point is assigned a prototype that best captures the local shape. Finally, each prototype has a known offset to the object centroid, which serves as the foundation for generalized hough transform as discussed below..

In this section, we provide an overview of the implicit shape model algorithm. This approach is split into a training and testing phase. Figure 3 shows an overview of the training phase. Assume that there exists a labelled training set with ground truth segmentation masks for a given object category. For each training image, first extract interest points for each image. Next, compute a local descriptor for the patch centered at each interest point, such that the local descriptor captures the spatial layout of the local contrast nearby the interest point. These descriptors may be simple appearance representations of the grayscale values of a patch or oriented gradient histograms capturing the distribution of local contour fragments. Next, the local descriptors are clustered to find a set of N prototype descriptors forming a *codebook* $\mathcal{C}$. This codebook or *dictionary* partitions a large set of local descriptor observations into a finite set of N groups, each represented by a mean

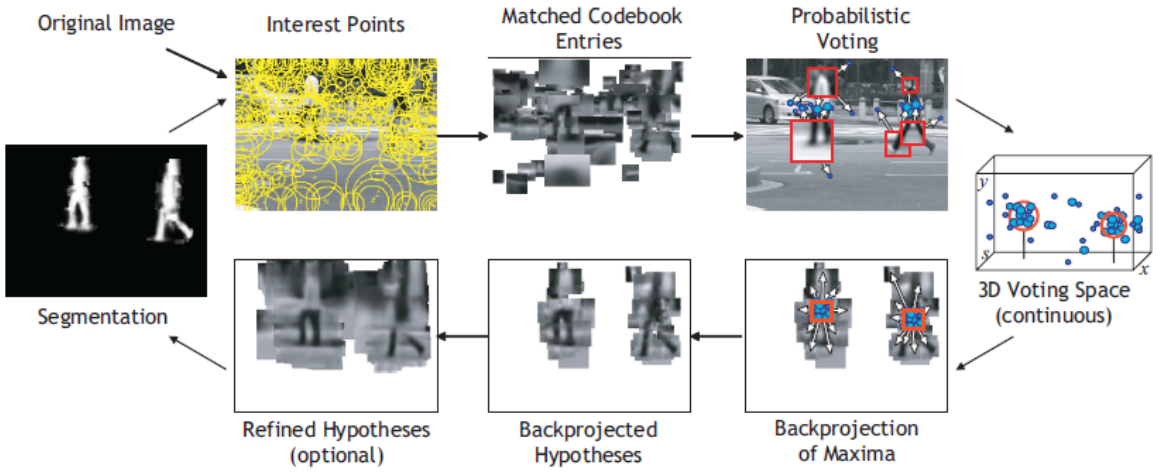


Figure 4: Implicit shape model [50]: Testing

central member or *prototype*. Next, each interest point location is assigned to a prototype which best matches the local patch descriptor, such that the match is greater than an activation threshold. Next, the position offsets P_c from each matched prototype to the object centroid are recorded. The "implicit shape model" is the set $(\mathcal{C}, P_c)$ of codebook and offsets.

Figure 4 shows an overview of the testing phase. The output of the training phase is an implicit shape model $S_i = (\mathcal{C}_i, P_{(i,c)})$ for each object category $i \leq N$. Recognition proceeds by extracting interest points and matching prototype entries $\mathcal{C}_i$ at interest points. Next, perform generalized hough transform to vote for object centers using the spatial probability distributions P_c for each prototype. Local maxima in this Hough voting space are extracted using the mean-shift procedure [73], and those local maxima at x^* with accumulation greater than a threshold θ are hypotheses $h_j = (o_i, x)$ for object i at position x. These hypotheses are the hypothesized centroids of detected objects, and these centroids are backprojected into the image to determine the subset of prototypes that are consistent with each centroid. Finally, associated with each prototype is a binary segmentation mask corresponding to foreground/background, and this is used to determine an object segmentation.

Finally, the term "implicit" shape model refers to the fact that the shape representation does not explicitly state the allowable shape deformations of prototypes relative to an object centroid. Instead, the offset of each prototype relative to the object center in the training set is recorded, and these offsets are used during *testing* for object detection. This is in contrast to part based approaches which learn an explicit deformation model of parts during training, and require a large training set for accurate deformation parameter estimation [22, 20, 74].

3.1.1.2 Generalized Hough Transform The Generalized Hough transform (GHT) [75] is a non-parameteric extension of the classic Hough transform for detecting parameterized shapes in

imagery. In the Hough transform, simple shapes are assumed described by a given parameterization (e.g. the two parameters defining a 2D line). Given this parameterization, a Hough space is constructed such that each dimension of this space corresponds to an independent parameter of the shape under consideration. Points in this Hough space correspond to a specific configuration of parameters. For example, a 2D hough space for a line has points $p = (\rho, \theta)$ for a line at a specific orthogonal distance ρ and orientation θ from a given origin in the image. The goal of the Hough transform to accumulate measurements or *votes* for each configuration of parameters based on evidence from an image that is consistent with that parameterization. For example, a given point p in Hough space to a line in the image with orthogonal distance ρ from the origin, and orientation θ. This parameterization can be realized and the evidence to support this parameterization can be measured in the image (e.g. construct a line at this position and orientation and compute a fitness measure for this line with support in the image). The Hough space is quantized, and evidence is accumulated exhaustively for every configuration of parameters. Finally, local maxima for accumulated evidence in the Hough space correspond to parameters of an hypothesized shape detection.

The generalized Hough transform [75] extends the classic Hough transform to the detection of arbitrary shapes undergoing an unknown but parameterized transformation. Given an image that contains an fixed shape or *template* that has been distorted by applying an unknown geometric transform, the goal of the generalized hough transform is to recover the parameters of the geometric transform. This approach has been used for parameterized transformations such as Euclidean, similarity, affine and projective groups. However, in the simplest case, the transformation is simply an unknown translation. Then, each point in the quantized Hough space corresponds to a possible shape centroid, and features in an image *vote* for the likelihood of a centroid position relative to their current position. Local maxima in this Hough space correspond to detection hypotheses for positions of a template with the most support in the image.

3.1.1.3 Training The training procedure for the implicit shape model is simple. As discussed in section 3.1.1.1 and shown in figure 3, the output of the training procedure is a list (C, P_c) of prototype codebook C and relative offsets of prototypes P_c. The codebook generation requires design choices for interest point detection, part representation and clustering algorithm. In this paper, the authors analyze the effects of four different interest point detections (Harris, Harris-Laplace, Hessian-Laplace and difference of Gaussian), three different part representations (grayscale patch, SIFT descriptor, shape context descriptor) and three different clustering algorithms (K-means, agglomerative clustering, Reciprocal Nearest Neighbor based agglomerative clustering). Results are shown that describe the effect of these choices on recognition performance.

3.1.1.4 Detection Object detections are computed using a probabilistic interpretation of the generalized hough transform. Let o_n be the object category label, x is the detected object image position, f is the observed feature at location l, and C_i is a codebook prototype. The probability of an object at position x is

$$p(o_n,x) = \sum_i p(o_n,x,f_i,l_i) \tag{2}$$

$$p(o_n,x) = \sum_i p(o_n,x|f_i,l_i)p(f_i,l_i) \tag{3}$$

$$p(o_n,x) \propto \sum_i p(o_n,x|f_i,l_i) \tag{4}$$

$$p(o_n,x) \propto \sum_{i,j} p(o_n,x|C_i,f_j,l_j)p(C_i|f_j,l_j) \tag{5}$$

$$p(o_n,x) \propto \sum_{i,j} p(x|o_n,C_i,l_j)p(o_n|C_i,l_j)p(C_i|f_j) \tag{6}$$

where equation 2 is the marginal over observed features, equation 3 follows from the chain rule for conditional probabilities, equation 4 assumes a uniform prior over observed features, equation 5 includes a marginal over codebook entries, and equation 6 follows from the chain rule, and the assumption that the conditional probability of codebook matches given features are independent of position $p(C|f,l) = p(C|f)$.

Equation 6 is the joint probability of an object detection at location x which encodes the generalized hough transform. The term $p(x|o_n,C_i,l_j)$ is a vote weight for a detected object at x given the object label, and marginalized over codebook entries and codebook match positions. The term $p(o_n|C_i,l_j)$ is a probability for an object label given the matched codebook prototype, and can be estimated as the inverse frequency that a codebook entry is matched to an object during training. The term $p(C_i|f_j)$ is error weighting of the prototype to feature matching.

Vote generation operates as follows. Every matched prototype has an associated centroid offset for each object label which captures the position of the centroid relative to the prototype for all training images. Vote weights w are determined from (6), where $w = p(x|o_n,C_i,l_j)p(o_n|C_i,l_j)$ is the occurrence weight for a prototype for a given class, and $p(C_i|f_j)$ is the prototype match likelikelihood. Each term is normalized to integrate to one.

Finally, detections are localized using scale adaptive hypothesis search. This approach uses mean shift to determine local maxima in a continuous hypothesis space, initialized from discrete local maxima from the generalized hough transform. The mean shift is a kernel density estimation procedure for the position of the object center, using weights determined from the generalized hough transform. The kernel used in the mean shift procedure is the balloon density estimator (7)

$$\hat{p}(o_n,x) = \frac{1}{V_b(x)} \sum_k \sum_j p(o_n,x_j|f_k,l_k) K(\frac{x-x_j}{b(x)}) \tag{7}$$

This density estimator includes a scale dependent kernel bandwidth $b(x)$ that captures the intuition that as the object scale increases, the voting uncertainty should be spread over a larger area.

Dataset	Training (#)	Testing (#)	Equal Error Rate (%)
UIUC cars, single scale	50	200	97.5
UIUC cars, multiscale	50	139	95
Caltech cars	126	1896	70.9
TUD motorbikes	153	125	87
VOC motorbikes	153	227	48
TUD pedestrians	210	595	80

Table 1: Performance evaluation of the implicit shape model

3.1.1.5 Segmentation The implicit shape model can also be used for top down segmentation. Each prototype determined during training also has an associated foreground/background mask determined from ground truth annotation. Intuitively, once we have an object detection hypothesis that includes a matching of features to part prototypes, we also have an hypothesized foreground/background mask at each interest point. A soft segmentation can be estimated for each pixel by marginalizing over all foreground/background masks that contain that pixel. Consider:

$$p(u_{fig}|o_n,x) = \sum_{p\in(f,l)} p(u_{fig}|o_n,x,f,l)p(f,l|o_n,x) \tag{8}$$

$$p(u_{fig}|o_n,x) = \sum_{p\in(f,l)} \sum_i p(u_{fig}|o_n,x,C_i,l)\frac{p(o_n,x|C_i,l)p(C_i|f)p(f,l)}{p(o_n,x)} \tag{9}$$

$$p(u_{fig}|o_n,x) \propto \sum_{p\in(f,l)} \sum_i p(u_{fig}|o_n,x,C_i,l)p(o_n,x|C_i,l)p(C_i|f) \tag{10}$$

Equation 8 is a marginalization over all features which contain a given pixel u. Equation 9 ... Equation 10 assumes uniform priors on $p(o_n,x)$ and $p(f,l)$ The final segmentation is computed using a likelihood ratio test for both figure and ground, resulting in a final soft segmentation. An example of the segmentation is shown in figure 4.

3.1.1.6 Results Object detection performance was evaluated on UIUC cars, CalTech cars, TUD motorbikes, VOC'05 motorbikes, Leeds cows, TUD pedestrians. Correct detections must lie within an ellipse with major and minor axis length equal to 25% of the bounding box dimensions, centered at the true detection, with at least 50% bounding box overlap for true detection. Object detection performance was evaluated using recall vs. 1-precision plots, with equal error rates summarized in table 1.

The results in table 1 are competitive with the state of the art as of 2004. Segmentation performance was not evaluated quantitatively, and only qualitative soft segmentation results are shown. The most difficult dataset is the VOC motorbikes, which exhibits low foreground contrast, strongly

textured backgrounds, target occlusion and in-plane rotation which results in a large number of false part detections in the background, and missed detection of prototypes.

Next, the authors analyzed the effect of different design choices on object detection performance. First, the authors considered two clustering algorithms for computing prototypes: k-means and agglomerative clustering. Results showed that performance degrades gracefully as the number of prototypes is decreased, and that agglomerative clustering results in more compact clusters and therefore more selective prototypes. Next, the authors evaluated the effect of training set size on performance for the UIUC single scale cars dataset, and results showed that diminishing returns appear at approximately 30 training images. Next, the authors considered the effect of three different interest point operators: hessian, harris, and difference of Gaussian on scale changes. Finally, the authors considered three different patch descriptors (SIFT, shape context, and raw grayscale patches), and results showed that SIFT and shape context outperformed grayscale patches.

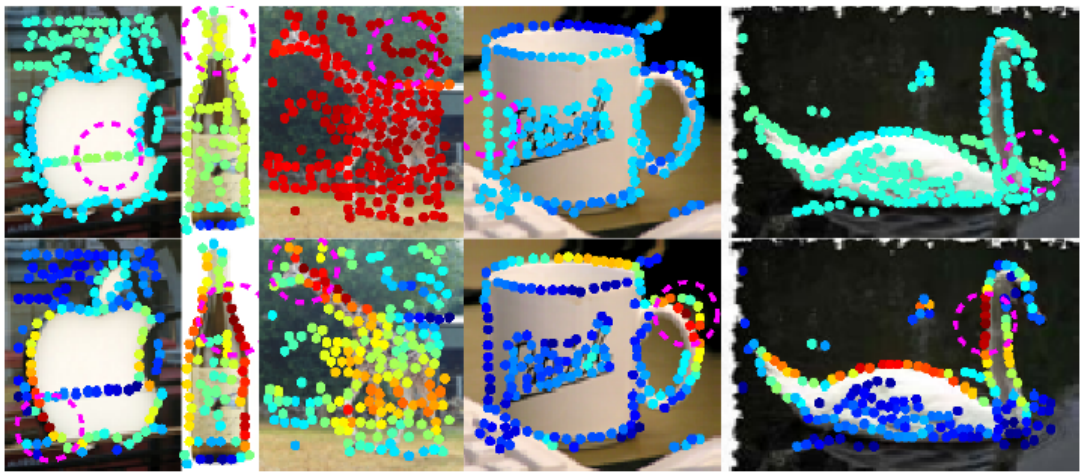
Finally, the authors describe an information theoretic approach to hypothesis verification using minimum description length hypotheses. This step is used for verification of hypotheses from competing objects. This approach has been replaced in the literature by discriminative training based verification for more principled verification, so as a result we do not spend time on this component, and rather focus on the discriminative verification. This will be described in more detail in the next section.

3.1.2 Max Margin Hough Transform

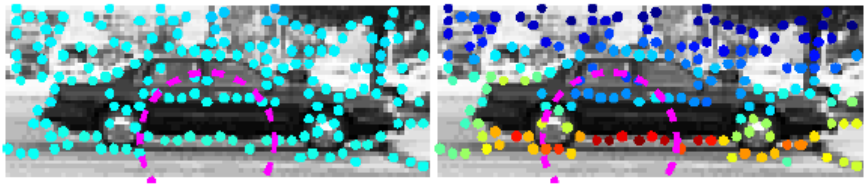
The *max margin hough transform* [68] is an extension of the implicit shape model to a discriminative hough tranform. This approach combines the benefits of the shape representation of the implicit shape model with part weights learned in a discriminative setting to show improved recognition performance.

This approach is motivated by the observation that parts voting in the Hough transform should be *repeatable* and *stable*. Repeatable parts are those that commonly occur within an object category. Stable parts refer to those parts whose position is consistent relative to the object center across training examples in the object category. The implicit shape model does extract parts that are repeatable as part of the prototype learning phase, however the requirement that parts are also stable are not included in the optimization. The benefit of this approach is that the local maxima in the Hough voting space are more "peaked", as the votes are only generated from discriminative parts rather than accumulated over all parts.

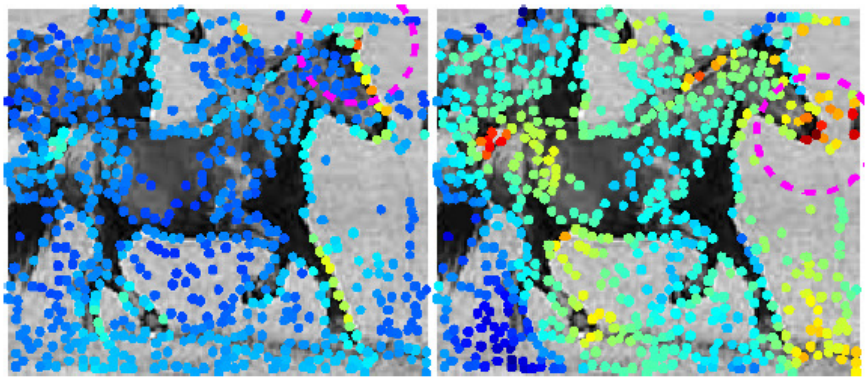
The part weight learning can be summarized as follows. First, we describe the probabilistic hough transform and decompose this into terms which can be analysed to motivate the part weight learning framework. In this work, the authors set up a framework where part weights are optimized using supervised learning. This approach provides high vote weights to those prototypes that are repeatable and stable, which results in improved Hough voting performance.



(a) ETHZ Shape Dataset



(b) UIUC Cars



(c) INRIA Horses

Figure 5: Max Margin Hough Transform [68]: Learned part weights as compared to naïve Bayes based weighting. (a)(top) is naïve Bayes (bottom) MMHT (b and c)(left) naïve Bayes (right) MMHT. Observe that the part weights learned are highly weighted at discriminative parts for the object class, such as the neck of the swan, the handle of the mug, and the nose of the horse, and is low weighted at random parts such as the writing on the mug or the label of the wine bottle.

$$S(O,x) \propto \sum_{(i,j)} p(x|O,C_i,l_j)p(C_i|f_j)p(O|C_i,l_j) \tag{11}$$

$$= \sum_i p(O|C_i) \sum_j p(x|O,C_i,l_j)p(C_i|f_j) \tag{12}$$

$$= w^T A(x) \tag{13}$$

where $A^T = [a_1\ a_2 \ldots a_K]$ and $a_i(x) = \sum_j p(x|O,C_i,l_j)p(C_i|f_j)$. In this formulation, O is the object index, x is the detected pose, C_i is the codebook, f_j is the feature extracted from the image at position l_j.

The part weight optimization is similar in formulation to a support vector machine. Equation 14 shows a quadratic program for the optimal part weights.

$$\begin{aligned} (w^*,b^*,\xi^*) = \arg\min \quad & \tfrac{1}{2}w^T w + C\textstyle\sum_{i=1}^T \xi_i \\ \text{s.t.} \quad & y_i(w^T A_i + b) \geq 1 - \xi_i \\ & w \geq 0, \xi_i \geq 0 \end{aligned} \tag{14}$$

In this formulation, $\{(x_1,y_1),(x_2,y_2),\ldots,(x_N,y_N)\}$ are a set of labelled training examples such that $y_i \in \{-1,1\}$. Optimal weights w^*,b^* encode the maximum margin separating hyperplane for training observations, with ξ^* as the optimal soft margin variables to handle outliers.

An example of the weights learned from this approach is shown in figure 5. In this figure, the maximum margin hough transform (MMHT) part weight learning is compared to naïve Bayes weights. The naïve Bayes weights consider only appearance rather than appearance and spatial distribution of a part. The naïve Bayes weights are set independently for each codebook entry C_i as $p(C_i|O)/p(C_i)$, where $p(C_i|O)$ is the likelihood of a codebook entry given an object index and $p(C_i)$ is the likelihood of a codebook entry across all training examples. Figure 5 shows that the weights learned by MMHT are higher for discriminative parts for an object class, such as the nose of the horse or the handle of the mug, and lower for those parts that are not discriminative such as text on the mug or uninformative contours on the swan.

Quantitiative results show that these learned weights result in improved detection rates on ETHZ shapes, increasing the average detection rate at 1 false positive per image from 0.54 using the naïve Bayes to 0.61 using MMHT weights. This paper also includes an improved verification stage following object detection that replaces the information theoretic MDL based hypothesis verification in the original ISM paper. However, this approach uses existing work on spatial pyramid matching and a support vector machine (SVM) with a pyramid kernel to provide verification of the final detections. This is a more principled approach to verification and provides improved recognition performance on challenging datasets such as ETHZ shapes and Inria horses, however this is not the primary contribution of this paper and so this approach and the corresponding results are not addressed here.

3.2 Bipartite Graphs

A graph $G = (V, E)$ is a bipartite graph if and only if the vertex set V can be partitioned into two disjoint subsets V_1 and V_2 such that $V = V_1 \bigcup V_2$ and no edge in the edge set E has endpoints in the same subset [54]. Bipartite graphs are more general that the star graphs described in section 3.1, since a star graph is a special case of a complete bipartite graph. To see this, observe that the star graph nodes can be partitioned into two subsets containing the central node and the leaf nodes forming a bipartite graph, and that every pair (v_i, v_j) such that $v_i \in V_1$ and $v_j \in V_2$ has an associated edge $e = (v_i, v_j) \in E$, forming a complete bipartite graph.

Bipartite graphs are useful for representing one to one matching. A perfect matching M for a bipartite graph G is a subset of edges ($M \subset E$) such that each vertex is incident to at most one edge of M. A minimum weight perfect matching is a perfect matching of minimum cost where the cost of a matching is given by $c(M) = \sum_{(i,j)\in M} c_{ij}$. The minimum weight perfect matching problem can be posed as a linear assignment problem which is formulated as an integer linear program as follows

$$\begin{aligned} x^*_{LAP} = \arg\max \quad & Wx \\ \text{s.t.} \quad & \mathbb{1}^T x_j = 1 \\ & x_i^T \mathbb{1} = 1 \\ & x_{ij} \in \{0, 1\} \end{aligned} \tag{15}$$

It can be shown that the constraint matrix A in equation (15) capturing the matching constraints (after dropping the integer constraints) is totally unimodular, so the integer linear program has an efficient integer solution. Minimum weight perfect matching can also be solved efficiently by using special purpose algorithms such as the Hungarian algorithm or reducing to a maximum network flow problem [76].

Shape contexts [28] are a representative approach to shape representations that uses bipartite matching. Shape contexts use a descriptor to capture the local shape, and bipartite graph matching to align a reference shape with an observation. Complete bipartite graphs are constructed by first performing edge detection in an image, then subsampling the resulting edge map. The subsampled locations provide a graph embedding for V_1, and the attributes of each node are a shape context descriptor. This descriptor counts the number of nearby edges using a log-polar histogram centered at the embedding coordinate and concatenates this log-polar histogram into a vector representation d_i. The bipartition V_2 is a reference shape, and a complete bipartite graph is constructed with edges between image nodes V_1 and reference nodes V_2. Minimum weight perfect matching is performed by setting the assignment weights W according to kernel weight between shape context descriptors $w_{ij} = K(d_i, d_j)$, where a common kernel is the Gaussian kernel.

Since shape contexts using bipartite matching, there are no edges between nodes within V_1 and V_2. This means that the matching is performed on nodes only. It is assumed that the geometry is encoded in the shape context descriptors, since each log-polar histogram has large support often covering a large fraction of the object. However, since there are no explicit edges to preserve in the

matching, this approach cannot guarantee preserving geometric relationships. In practice, bipartite matching has been subsumed by many to one matching on general graphs.

3.3 General Graphs

A general graph is an arbitrary graph $G = (V, E)$ without additional constraints on edges or nodes. Unlike the graphs considered in previous sections for which the special graph structure enabled efficient matching algorithms, graph matching in general is NP-hard problem as it is isomorphic to the quadratic assignment problem outlined in section 2.2.

In this section, we focus on two convex relaxations of the quadratic assignment problem. First, we describe *graduated assignment* which is a relaxation using graduated non-convexity. Second, we describe a spectral relaxation.

3.3.1 Graduated Assignment

Graduated assignment [51] refers to relaxation of the quadratic assignment problem in (1) based on *graduated non-convexity*. This approach relaxes the integer constraint $x \in \{0, 1\}$, then performs an iterative linearization of the quadratic objective in a deterministic annealing framework, resulting in a sequence of optimal solutions to *linear assignment problem* for which efficient solutions exist. This work is a classic in the literature, and is representative of graph matching approaches using deterministic annealing optimization of the intractable quadratic assignment problem.

Graduated assignment is based on the key concepts of softassign and graduated non-convexity. *Softassign* is Sinkhorn bistochastic normalization. Sinkhorn showed that any square matrix with positive elements will converge to a doubly stochastic matrix by iteratively normalizing rows and columns. The goal of one-to-one graph matching is finding an optimal assignment of each node in one graph to a node in the other, such that the assignment is encoded in a binary permutation matrix (see section 3.3.1.1). A bistochastic or doubly stochastic matrix has columns and rows that sum to one, making is analogous to a continuous relaxation of a permutation matrix. Hence, the bistochastic matrix can be considered a soft-assignment or softassign. Graduated non-convexity refers to an iterative deterministic annealing approach to convex relaxation, where the softassign is exponentiated with a gradually increasing exponent parameter, which results in a softassign that is gradually more like a permutation matrix. In other words, the control parameter is increased at each step to move the real valued bistochastic matrix closer to binary valued permutation matrix.

3.3.1.1 Problem Definition A permutation matrix Π is a square orthogonal matrix with binary entries such that each row and column has exactly one entry $\Pi_{i,j} = 1$ and zero elsewhere. A permutation matrix has the following properties:

- *Orthogonality*: $\Pi^T \Pi = I$

- *Doubly Stochastic*: $\Pi^T \mathbb{1} = \mathbb{1}$ and $\mathbb{1}^T \Pi = \mathbb{1}$, such that $\mathbb{1} = [1\ 1 \ldots\ 1]^T$ is a vector of ones. A doubly stochastic matrix with each row sum ($\Pi^T \mathbb{1}$)and column sum ($\mathbb{1}^T \Pi$) that equals one.
- *Assignment*: $y = \Pi x$, such that $y_i = x_{\Pi(i)}$ is a permutation of the elements of x.

A permutation matrix encodes an assignment or *matching*. Given an indicator vector $\mathbb{1}_i = [0\,0 \ldots\ 1\ \ldots\ 0]^T$ with zeros everywhere except at element i then an assignment of i to j is given by $\mathbb{1}_j = \Pi^T \mathbb{1}_i$. It is assumed that "dummy nodes" can be added to graph G or G' so that each graph has equal number of nodes to enable a one to one matching.

The graduated assignment algorithm will be described in the context of weighted graph matching. Given two attributed graphs G, G', the goal of weighted graph matching is to find an matching (permutation) matrix M that optimally assigns nodes $\Pi : V \rightarrow V'$, given an weighted assignment cost C. The objective function, subject to permutation matrix constraints on M, is as follows:

$$E_{ga}(M) = -\frac{1}{2} \sum_{a=1}^{A} \sum_{i=1}^{I} \sum_{b=1}^{A} \sum_{j=1}^{I} M_{ai} M_{bj} C_{aibj} \tag{16}$$

$$E_{ga}(M) = -\frac{1}{2} m^T C m \tag{17}$$

M is a permutation matrix of size $A \times I$, M_{ij} is the ij^{th} element of M, m is a vectorized permutation matrix, C is a real valued quadratic matching cost, and C_{aibj} is the matching cost for edges (a,b) in G and (i,j) in G'. Observe that this objective in vector notation (16) is equivalent to the matrix notation (17), and this matrix notation is equivalent to the quadratic assignment problem described in (1), up to a constant factor of -0.5. The matrix notation is a more compact representation for this problem and is useful notation to show a common framework for describing and comparing different weighted graph matching algorithms.

As discussed in section 2, the quadratic assignment problem is intractable. The graduated assignment uses a sequence of optimal solutions to a *linear assignment problem*. A linear assignment problem is a combinatorial optimization problem which can be stated as a linear program as was defined in equation (15). The graduated assignment uses a linear assignment problem that is justified from a linearization of the quadratic objective. The linearization is justified as follows. Given an initial match M_0, the objective (16) can be expanded using a Taylor series approximation as follows:

$$E_{ga}(M) = m^T C m \tag{18}$$

$$E_{ga}(M) \approx m_0^T C m_0 - q^T (m - m_0) \tag{19}$$

$$q = -\left.\frac{\partial E_{ga}}{\partial M_{ai}}\right|_{M=M_0} = \sum_{b=1}^{A} \sum_{j=1}^{I} M_{bj}^0 C_{aibj} \tag{20}$$

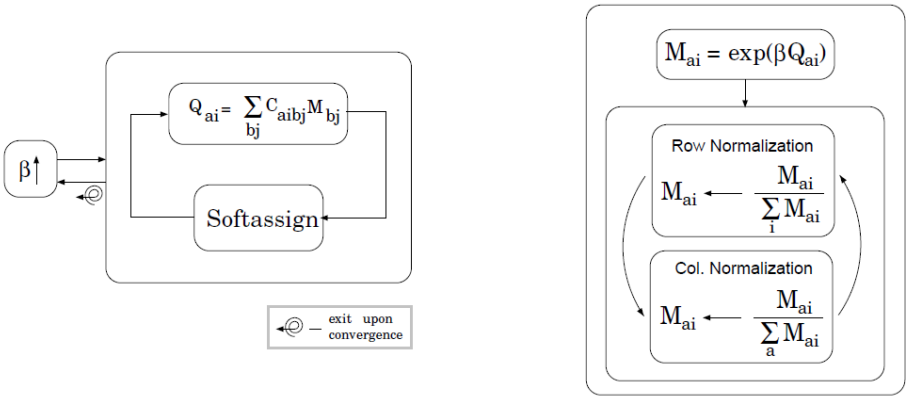


Figure 6: Graduated Assignment [51]: Algorithm Overview

where the linearization in (19) is minimized by maximizing $Q^T m$ which is exactly a linear assignment problem. This observation will be used in the optimization algorithm in section 3.3.1.2.

3.3.1.2 Algorithm The algorithm for graduated assignment is summarized in pseudocode as algorithm 1 and graphically in figure 6. β is the graduated non-convexity control parameter with initial value β_0 and maximum final value β_f. The maximum number of iterations for deterministic annealing is I_0 and the maximum number of iterations for Sinkhorn normalization is J_0. The Jacobian Q is given by (20). Finally the output requires a final "cleanup" heuristic function f_h to enforce that the final match matrix is a permutation matrix. In this paper, the heuristic chosen is to assign the maximum value in each column to one, and the other entries to zero.

The algorithm operates by iterated graduated non-convexity and soft-assign. Given a uniform initial condition for the matching, first compute the Jacobian $Q_{ai} = \sum_{bj} C_{aibj} M_{bj}$ as in (20) using the current estimate of M and the known assignment costs C. Next, perform a softassign such that $M = exp(\beta Q)$. This enforces that elements of the match matrix M is strictly positive, and that the maximum elements are exponentially amplified. Following from (20), Q encodes a linear assignment problem, which is maximized by the maximum permutation matrix. So, the $exp(\beta Q)$ sets Q to be a soft-assignment matrix with maximum elements, and then this matrix is Sinkhorn normalized to set it to a bistochastic matrix. This is the maximum bistochastic matrix, and therefore minimizes the linearization in (19). Increase the annealing parameter β by an annealing rate β_r and repeat until convergence of M or the maximum number of iterations is reached. The final bistochastic matrix M is projected to a permutation matrix by a heuristic function f_h which sets the maximum element in each row and column to one, and all other elements to zero.

The computational complexity for this algorithm is $O(lm)$ were $l = |E|$ and $m = |E'|$ for the number of edges in each graph.

Algorithm 1: Graduated assignment for weighted graph matching

Input: $\beta \leftarrow \beta_0$, $\hat{M}_{ai} \leftarrow (1+\varepsilon)$
Output: $\Pi = f_h(M)$
while $\beta \leq \beta_f$ **do**
  while $I < I_0$ **do**
    $Q_{ai} \leftarrow -\frac{\partial E}{\partial M_{ai}}$
    $M^0_{ai} \leftarrow exp(\beta Q_{ai})$
    while $J < J_0$ **do**
      Sinkhorn normalization:
      $\hat{M}^1_{ai} \leftarrow \frac{\hat{M}^0_{ai}}{\sum_{i=1}^{I+1} \hat{M}^0_{ai}}$
      $\hat{M}^0_{ai} \leftarrow \frac{\hat{M}^1_{ai}}{\sum_{i=1}^{A+1} \hat{M}^1_{ai}}$
    end
    $\beta \leftarrow \beta_r \beta$
  end
end

3.3.1.3 Results Evaluations of this approach were performed on random graphs and on object matching in natural imagery. The object matching task considered an attributed graph extracted from a reference image of a wrench and an observed image containing a wrench in clutter. The reference graph contained 7 nodes, and the observed graph 27 nodes, and the approach successfully matched these graphs with no errors. Next, the authors hand constructed attributed graphs of a reference coffee mug and a coffee mug in clutter, matched the resulting graphs with no errors. These object matching results were state of the art at time of publication, however they are less than compelling today due to the assumption of hand extracted graphs rather than automatically constructed and due to the results on only two images due to the computational restrictions of the day. So, we include these results for completeness, however the more informative results that are still relevant today are the results on random graphs. We focus on these results in the remainder of this section.

The random graph experiments included randomly generating 100 node graphs and perturbing both the sparse structure and attributes. First, an experiment was performed to test robustness of subgraph isomorphism to percent connectivity in the graph, and results showed that less than one percent of the nodes were mis-assigned at 16 percent connectivity and 10 percent deleted. Next, an experiment was performed perturbing the weights of an attributed graph and results showed that the graph matching exhibited graceful degradation in errors up to 10 percent uniform noise on the weights over various graph structures (5 percent connected, 5 percent deleted/added to 60 percent deleted/added, 15 percent connectivity). Finally, performance was evaluated on attributed graphs with similar results.

Algorithm 2: Discretization for spectral graph matching [58]

Input: $L = V \times V'$, $x = [0\ 0 \ldots 0_{|L|}]$, x^* from (22)
Output: x
while $|L| > 0$ **do**
  $a^* = \arg\max_{a \in L} x^*(a)$
  if $x^*(a^*) = 0$ **then**
    return
  else
    $x(a^*) = 1$
    $L = L \backslash \{A^*, a^*\}$
  end
end

This work also discusses results on attributed relational graphs and constrained optimization using Lagrange multipliers, however these extensions are not discussed in detail here, as we focus on the primary contribution of the graduated assignment algorithm itself.

3.3.2 Spectral Graph Matching

Spectral graph matching [58, 59] is an alternative approach to graph matching which provides a spectral relaxation to the quadratic assignment problem in (1). Rather than an iterative solution based on deterministic annealing, this approach poses a convex relaxation of an integer quadratic program.

Spectral graph matching with linear matching constraints [59] is shown as an integer quadratic program in equation 21. Like previous approaches, x is a vectorization of an assignment matrix, and the linear constraints encode the mapping constraints of one to one or many to one.

$$\begin{aligned} x^*_{sgm} = \arg\max \quad & x^T W x \\ \text{s.t.} \quad & Cx = b, x \in \{0,1\} \end{aligned} \tag{21}$$

Observe that by dropping the mapping and integrality constraints in (21), the result is an unconstrained quadratic program. Observe that since x encodes an assignment matrix, only the relative values of x matter rather, so the norm of x can be set to one after optimization $||x|| = 1$. This means that the solution to the unconstrained (21) is equivalent to the solution

$$x^* = \arg\max_x \frac{x^T W x}{x^T x} \tag{22}$$

Equation (22) is of the form of a *Rayleigh quotient*, such that the optimal x^* is given by the leading eigenvector of W [77]. This spectral solution to (22) provides an optimal solution to the uncon-

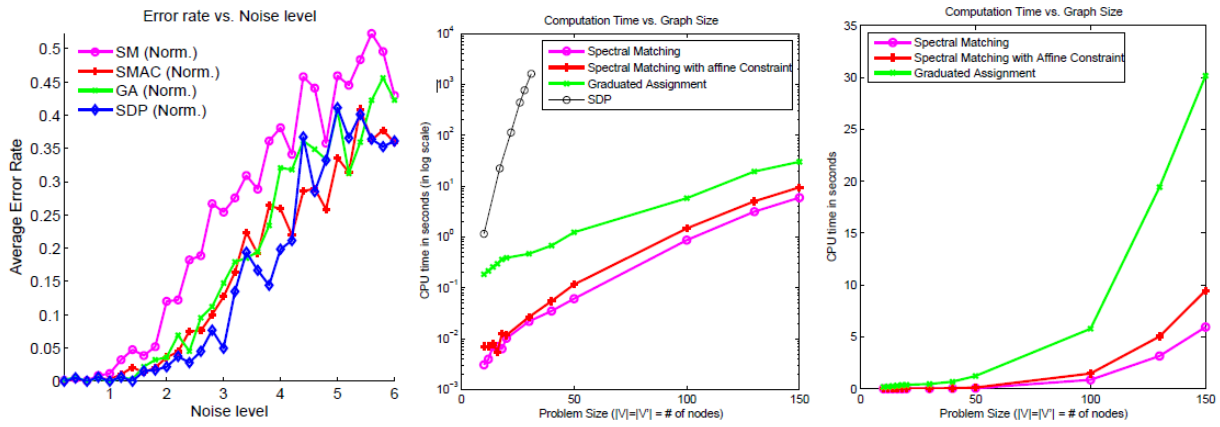


Figure 7: Spectral graph matching [59]: Performance Comparison on random graphs

strained quadratic program in (21), however this solution must still be discretized to satisfy the assignment and mapping constraints.

The discretization algorithm is described in algorithm 2. Let the vectorized x^* be reshaped into original assignment matrix form X^*. Real valued entries X^*_{ij} are interpreted as a confidence that nodes (i, j) should be assigned, however since columns and rows of X are real valued, the mapping constraints and integrality constraints are not satisfied. The goal of the discretization is to use the confidence of assignments to convert X into a binary assignment matrix that satisfies the mapping constraints. This iteration is initialized by a list L of all possible assignments of nodes in G and nodes in G', and a vector x of zeros of the same size as x^*. The iteration proceeds by finding the optimal assignment a^* from the confidence x^* and setting this as an assignment in x. This assignment a^* and all those assignments A^* that conflict with a^* from the mapping constraints in (21) are removed from L. Iteration proceeds until the assignment list is empty. The result is an assignment matrix x that satisfies integrality and mapping constraints and is generated using an optimal confidence of assignment x^* from spectral relaxation.

The computational complexity of the spectral matching is $O(n^{3/2})$ which follows from the computational of the leading eigenvector of W. The complexity is dominated by the eigen-decomposition and not the discretization, and it assumes efficient iterative solutions for eigen-decomposition of large sparse matrices using the power method, such as the Lanczos algorithm. The main computational gain of this approach comes from dropping both the linear mapping constraints and the integrality constraints during the optimization and using them during the iterative discretization step. This is contrasted with graduated assignment which computes the optimal constrained solution at each iteration.

Figure 7 shows results [59] comparing graduated assignment (GA) with spectral matching (SM). These results are from a closely related spectral graph matching paper by Cour et. al [59] which considers affine matching constraints, and also compares performance to this approach and

a semidefinite programming relaxation. Figure 7(left) shows matching error rate averaged over 100 trials for increasing noise on the attributes used to construct the weight matrix. Results show that spectral matching performance is comparable to graduated assignment, for a significantly reduced computational requirement, that provides scalability to larger graph instances.

4 Topological Graph Representations

4.1 Computational Topology

Computational topology is the study of the algorithmic questions in topology and topological questions in algorithms [78][79]. For example, topological problems are those about invariant properties of connectivity and continuity, without requiring spatial notions such as straightness, distance or convexity. A topological question may ask about the number of "holes" in a topological space, allowing deformations but not cutting or gluing, whereas an algorithmic question in topology may be the computation of the number of holes in a discrete representation of this space. Similarly, a topological question in algorithms may be, given a discrete representation of a topological space, does it preserve the topology of the underlying continuous space? Computational topology grew out of the desire to extend discrete results in computational geometry for point sets, polygons and polyhedra in to continuous domains, curved surfaces and higher dimensions. The success of computational geometry and the interesting overlap of computer science and topological questions hold promise of furthering both fields.

However, a challenge for collaboration between computer science and topology is the lack of common language. Motivated computer scientists without appropriate training in topology may find the topological literature unapproachable. Topology has an occasionally complex notation, and requires a significant number of definitions and accumulated theory to be grasped to understand the literature, and motivated readers may be unsure if the effort to learn this common language will be worthwhile.

Fortunately, recent work has demonstrated the power of computational topology to justify this learning curve. Topological concepts have been applied to a wide range of application areas including shape acquisition for solid modelling using computer aided design, shape representations for interoperability, portability and simplification in computer graphics, mesh generation for physical simulation and finite element analysis, configuration spaces in robotics, molecular biology, computer vision and databases. These application areas all rely on topological algorithms, and collaboration between topologists and computer scientists have analyzed the computational complexity. Typically, planar topological problems are polynomially solvable, problems in $\mathbb{R}^3$ are exponentially solvable and are thought to be *NP*-complete, while problems in $\mathbb{R}^4$ and above are known to be undecidable [79]. Such analysis and the wide range of applications, many in high dimensional spaces, highlights the need for further investigation into efficient algorithms for computational topology.

In this section, we summarize a survey on computational topology [78]. This survey focuses on applications and methods, and provides an introductory set of definitions in topology to facilitate the language barrier between topology and computer science. Next, we provide a more in depth review of one application of computational topology to the qualitative analysis of the space of natural images in computer vision [80][81]. This approach uses *persistent homology* to characterize the topology of the space of 3x3 patches in natural images. This is an example application of computational topology which addresses an approach to suitably preprocess noisy data to enable topological analysis of global structure.

The survey by Dey, Edelsbrunner and Guha [78] provides three main contributions: (i) a description with motivating examples of six application areas for computational topology including image processing, cartography, computer graphics, solid modelling, mesh generation and molecular modelling, (ii) a description of six topological methods and (iii) an appendix with technical definitions to aid the reader without a background in topology. In this section, we focus on intuitive descriptions of the topological applications and methods, then provide an in depth description of one application in section 4.3.

4.1.1 Applications

Image processing can be posed as a topological problem by considering questions of connectivity and skeletonization of binary pixels. Connectivity is the problem of determining paths of adjacent pixels with the same value, such that adjacency is defined using either 4-neighbors or 8-neighbors in a grid. Using such a definition of paths, one can use basic results in plane topology to provide existence proofs for connected components. For example, the Jordan curve theorem which states that every simple closed curve in $\mathbb{R}^2$ decomposes the plane into two disjoint open connected components. This allows the computation of connected components in an image using topological computations. Similarly, skeletonization is the replacement of a set of foreground pixels with its "skeleton", which is a thin subset of the foreground that maintains connectivity. Topological concepts of connectivity can be used to aid in the computation of skeletons.

Cartography exhibits topological problems when considering deformations of geographic maps. When considering two or more maps with different shapes, a specific visualization of map information requires bringing these maps into correspondence. The smooth deformation required for registration of these maps can be modelled as a homeomorphism, or when representing a map in terms of a decomposition into simplices, then the simplicial map providing the registration is a simplicial homeomorphism.

Computer graphics can use topological concepts to pose problems of warping and morphing. Warping is the distortion or deformation of a 2D image, which can be used to deform a texture map to suitably model the foreshortening distortion of a texture based on viewpoint during the rendering pipeline. As with cartography, simplicial maps can be used to map triangles defining the object surface to deform to the rendered image. Morphing is the problem of gradually deforming one shape into another in a visually pleasing way. Morphing can be posed as a topological problem

of registration using simplical maps. Given two objects with isomorphic simplical complexes, the simplical map can be constructed to slowly move each vertex in one simplical map to the corresponding vertex in the morphed map.

Solid modelling is the process of representation, import and analysis of 3D object models in a computer. Model representation is the specification of a discrete representation of a continuous object for use and storage in a computer. There are many possible representations for 3D object that have been developed, many of which are domain specific. Describing the relationships between all of these representations, providing a unified representation for them all, and converting between them all require topological ideas and methods. For example, using geometry alone is often insufficient for representing an object model, as this does not guarantee topological properties such as watertightness is preserved during conversion [79]. Model import or surface reconstruction is the process of measuring surface properties for reconstruction of the surface in the computer. Surface measurements may be collected using some form of range sensing resulting in a discrete point cloud which must be processed to reconstruct a surface representation. A suitable reconstruction is one that is topologically valid and consistent with the original object. Finally, analysis of 3D models to suitably simplify or compress an object model requires topological concepts, so that the model simplification results in a complex that is homeomorphic to the original, otherwise the simplification could mistakenly introduce self-intersections, often known as "bubbling" [79].

Mesh generation is the process of decomposing a space or shape of interest into a complex for scientific analysis. A challenge with mesh generation is creating "quality" meshes, which in the case of structured meshes, can be posed as a topological problem of constructing a mesh homeomorphic between the shape or space and the unit cube.

Molecular modelling is the study of the 3D structure and chemical conformation of molecules. Of particular interest is the protein folding problem, which is the simulation and prediction of the natural binding of protein molecules to uncover protein function. Such questions about folding can be posed as topological questions by abstracting a protein to a simplicial complex, which can then be analyzed to determine such topological properties as Betti number, which counts and characterizes holes, as well as determining such properties as voids, cavities and pockets using concepts in Morse theory.

4.1.2 Methods

Topological methods are theoretical topics in topology, which in this survey paper focus on: decompositions, fixed points, embedding, three-manifolds, and homology computation. Given that definitions and theorems are often their own best summaries, in this section, we will focus on the main ideas for each method rather than reiterating the definitions, theorems and proofs themselves.

Decompositions are the process of decomposing a shape into simple pieces. A classic result for complexes defined by the boundary of polyhedra states that $v - e + f = 2$, where v, e and f are the number of vertices, edges and faces in the polyhedron. More generally, the Euler characteristic of a space is the alternating simplex count, which is determined by counting vertices, edges and

triangles in a suitably oriented triangulation. The construction of a decomposition requires considering the cover of a topological space, which is a collection of subsets whose union is the space, and the nerve which is cover with non-empty common intersection. Such theorems have motivated the design of automatic triangulation algorithms for a topological space. Shelling is the process of constructing a complex by adding one cell at a time, and if a complex is shellable, then shelling can be used in such applications as computing the convex hull of a set of n points in $O(log(n))$ per face, with an initial $O(n^2)$ preprocessing.

Fixed points are those points of a function where the point is its own image. A classical result on contracting maps is that they have a unique fixed point. Continuous Brouwer's fixed point theorem is one of the most basic facts about topological spaces and generalizes the metric fixed point theorem stating that every map $f : \sigma^d \to \sigma^d$ has a fixed point, where σ^d is a d-simplex homeomorphic to $\mathbb{B}^d$. This theorem can be used to show the existence of a fixed point on a simplicial map, which is useful for such applications as finding centerpoints and equipartitions.

An *embedding* of one topological space into another is an injection whose restriction to the image is a homeomorphism. A classic result states that every abstract simplicial complex $\mathcal{A}$ has a special embedding in $\mathbb{R}^d$ called a geometric realization, provided d is large enough, and is always possible if $d = 2k+1$. For $\mathbb{R}^2$, a well studied problem is the embedding of planar graphs, which can always be embedded in $\mathbb{R}^3$, but can only be embedded in $\mathbb{R}^2$ if and only if it does not contain a subgraph homeomorphic to K_5 (complete graph with 5 vertices) or $K_{(3,3)}$ (complete bipartite graph with 6 vertices). Also, a classic result for $\mathbb{R}^2$ for more general spaces is that a 2-manifold can be embedded in $\mathbb{R}^2$ iff it is orientable (e.g. projective plane, Klein bottle)

Three-Manifolds are topological spaces that is locally Euclidean ($\mathbb{E}^3$), and many specialized results exist for 3-manifolds that do not generalize to higher dimensions. Unfortunately, this survey shows its age in that the Poincaré conjecture is described as an open problem, which has since been solved (proved 2002-2003, confirmed in 2006). A *knot* is an embedding of a closed curve in space $K : \mathcal{S}^1 \to \mathbb{R}^3$. A remarkable result due to Seifert states that every knot is the boundary of an orientable 2-manifold embedded in $\mathbb{R}^3$.

Homology computation offers a formal algebraic framework for studying and counting holes in a topological space by computing homology groups. Formally, these holes are characterized by Betti numbers, and since Betti numbers are invariant to triangulation, computing the Betti numbers of a simplicial complex is equivalent to computing the Betti number of the underlying space. Betti numbers can be computed for higher dimensional simplices by computing smith normal forms, using incremental algorithms or combinatorial laplacians, or using special cases of solids.

4.2 Simplicial Homology

In this section, we first provide a brief introduction to those definitions and results in simplicial homology necessary for describing the topological approaches in later sections. For detailed discussion of homology and algebraic topology, see [82, 83, 84].

We begin with general definitions. Practical vision applications are typically limited to dimension ≤ 3, however we introduce general definitions for completeness.

Definition 4.1. A p-dimensional simplex or p-simplex is the convex hull of $p+1$ affinely independent vertices $v \in \mathbb{R}^D$

Definition 4.2. A *face* of a p-simplex σ is a non-empty subset of vertices of σ.

Definition 4.3. A simplicial complex K is a set of simplices that satisfies the following closure conditions

(i) Any face of a simplex in K is a simplex in K

(ii) The intersection of any two simplexes $\sigma_i, \sigma_j \in K$ is a face of both σ_i, σ_j

Let K be a finite simplicial complex of dimension p, such that all simplexes $\sigma \in K$ have dimension at most p. A simplicial k-chain is a finite formal sum of k-simplices

$$\sum_{i=1}^{N} c_i \sigma_i,\ c_i \in \mathbb{Z}_2,\ \sigma_i \in K \tag{23}$$

where $c_i \in \{-1, 0, 1\}$ are binary valued coefficients. For each $k \geq 0$, k-chains along with the modulo-2 addition operator form the *chain group* $C_k(K)$.

Definition 4.4. The boundary operator $\partial_k \colon C_k \to C_{k-1}$ is a homomorphism between chain groups such that

$$\partial_k(\sigma) = \sum_{i=0}^{k} (-1)^i \langle v^0, \ldots, v^{i-1}, \hat{v^i}, v^{i+1}, \ldots, v^k \rangle \tag{24}$$

The notation $\hat{v^i}$ denotes that the vertex should be dropped. The boundary homomorphism is a linear operator and commutes with addition $\partial_k(c_1 + c_2) = \partial_k(c_1) + \partial_k(c_2) \forall c_1, c_2 \in C_k(K)$ Observe that $\partial_k(\sigma) = \sum c_i \partial_k(\sigma_i)$, and the boundary homomorphism is a map from k-simplexes to a sum of its $(k+1)$ faces.

The boundary homomorphism has a unique matrix representation with respect to a choice of basis. Let $\{\sigma_i\}$ and $\{\tau_j\}$ be the sets of k-simplexes and $(k-1)$-simplexes of size $|M-1|$ and $|N-1|$ that represent the elementary chain bases for C_k and C_{k-1}. Then, ∂_k is represented as an $M \times N$ boundary matrix with entries $a_{ij} \in \{-1, 0, 1\}$, such that $|a_{ij}| = 1$ if $i \in \tau$ is a face of $j \in \sigma$ with sign determined from (24), and zero otherwise. Some authors distinguish the boundary operator ∂_k with the matrix form $[\partial_k]$, however in this review, we assume that the context will make this distinction clear.

Definition 4.5. The chain complex

$$C_k \xrightarrow{\partial_k} C_{k-1} \xrightarrow{\partial_{k-1}} C_{k-2} \ldots C_0 \xrightarrow{\partial_0} 0$$

is a sequence of chain groups connected by boundary homomorphisms.

The boundary homomorphism has several useful properties, which we state but do not prove [82].

Lemma 4.6. *Given a boundary homomorphism* ∂*,*

(i) The boundary of a boundary is zero, $\partial_k \partial_{k+1} d = 0$, for every integer k and every $(k+1)$-chain d.

(ii) A k-cycle c is a k-chain with zero boundary $\partial_k c = 0$

(iii) The boundary of every 0-simplex is zero.

(iv) The cycle group $Z_k = ker(\partial_k) = \{x \in C_k(K) : \partial_k x = 0\}$

(v) The boundary group $B_k = im(\partial_{k+1}) = \{x \in C_k(K) : \exists\, y\ s.t.\ x = \partial_{k+1} y\}$.

Elements of the cycle group Z_k are k-chains called k-cycles, elements of the boundary group B_k are k-chains which called k-boundaries, which are boundaries of a $(k+1)$-chain and are also cycles ($B_k \subset Z_k$).

Definition 4.7. An *homology class* is an equivalence class of cycles such that for a fixed representative cycle z_0, $\{z | z = z_0 + \partial_{k+1} c, c \in C_{k+1}(K)\}$, where equivalent cycles of the same homology class are *homologous* and denoted $c \sim c'$.

Definition 4.8. The *homology group* $H_k(K) = Z_k / B_k$ is a quotient group formed on the set of homology classes.

Definition 4.9. The kth betti number $\beta_k = rank(H_k(K)) = rank(L_k)$ where $L_k = \partial_k^T \partial_k + \partial_{k+1} \partial_{k+1}^T$ is the rank of the kth homology group or equivalently the rank of the kth combinatorial laplacian L_k [85].

Definition 4.10. A chain map $M_k \colon K \to K'$ is a homomorphism mapping k-simplexes of simplicial complexes K and K'. The chain map must satisfy boundary commutativity.

$$\partial'_k \circ M_k = M_{k-1} \circ \partial_k$$

This requirement follows from the requirement $\partial_k \circ \partial_{k+1} = 0$. A chain map between chain complexes maps boundaries to boundaries and cycles to cycles, and induces homomorphisms between homology groups of the two complexes [82].

The definitions for simplicial homology were introduced in general for k-simplexes, however these concepts have intuitive low dimensional interpretations in the context of graph theory. Given a graph $G = (V, E)$, a 0-simplex is a vertex in V, a 1-simplex is an edge in E, a 2-simplex is a triangle which forms a three node clique, a 3-simplex as a tetrahedron or four node clique, and so on. The faces of a edge (1-simplex) are the two incident vertex endpoints (0-simplexes), and trivially the edge itself. The faces of a triangle (2-simplex) are the triangle itself (trivially), three edges (1-simplexes) and three nodes (0-simplexes). The simplicial complex closure constraints in (defn 4.3) states intuitively that if two edges are incident on a common vertex, then both edges

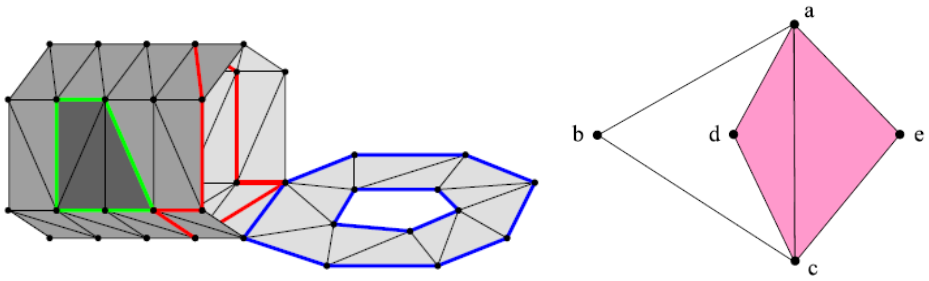


Figure 8: Examples of cycle and boundary groups [84]

must contain the vertex as a face. A 1-chain (23) is any subset of edges, not necessarily connected. The boundary map ∂_1 (24) is the oriented node-edge incidence matrix, and the boundary map ∂_2 is the edge-triangle incidence matrix. For node-edge incidence only, the combinatorial laplacian $L = \partial_1 \partial_1^T = D - W$, which is the classic graph laplacian for unit weights W. Betti numbers β_1 (defn 4.3) capture the number of "holes" in the graph, and the homology group H_1 contains equivalence classes such that each homology class is the set of all cycles that differ by a boundary from a cycle surrounding this hole. Finally, traditional graph matching is an estimation of $\hat{M}_0$, the permutation matrix between graph nodes (0-simplexes) that preserves edges (1-simplex intersections).

Homology groups (defn 4.8) are the key concept in this section, so let's deconstruct it further and give a simple example to build intuition. Figure 8 (left) shows a simplicial complex of maximum dimension $k = 2$ that has the shape of a cylinder with a lid. The green edges (c_g) form the boundary of the three 2-simplexes (c) shown as dark grey triangles. Formally, ($c_g = \partial_2 c$) and observe that the "boundary" of the three dark triangles is what one would expect, it is formed from the non-overlapping faces of the triangles. The two blue cycles are homologous since they differ by a boundary, but are not homologous to the red or green cycles.

Figure 8 (right) shows a different (smaller) simplicial complex for which we can construct and example to explicitly compute homologous cycles. This simplicial complex contains five 0-simplexes (a,b,c,d,e), seven 1 simplexes (ab,ac,ad,ae,bc,cd,ce) and two 2-simplexes (acd,ace) shown in pink. The boundary matrices are constructed using defn 4.4 and are given by:

$$\partial_1 = \begin{bmatrix} -1 & -1 & -1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & -1 & -1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \quad \partial_2 = \begin{bmatrix} 0 & 0 \\ -1 & -1 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \tag{25}$$

Entries $\partial_1(i,j)$ correspond to whether the i^{th} 0-simplex shares a face with the j^{th} 1-simplex. For example, $\partial_1(1,1) = 1$ since the edge ab shares a face with point a, $\partial_2(3,2) = 0$ since triangle

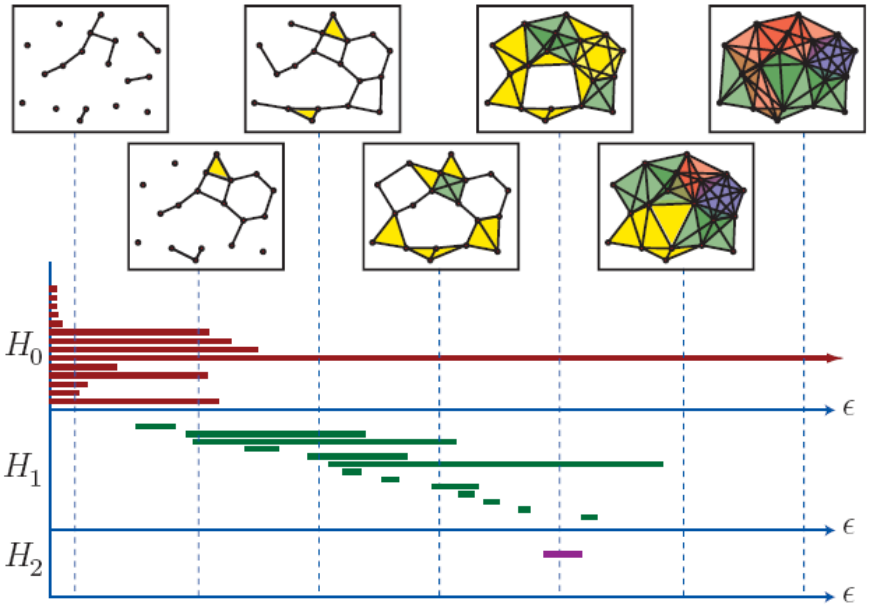


Figure 9: Persistent Homology [81]. (top) Rips complexes $\mathcal{R}_\varepsilon$ for increasing ε. Colors correspond to k-simplices. (bottom) "Barcode" representation of persistent homology groups for increasing ε. The vertical dotted lines correspond to the Rips complex at a specific ε, and the homology groups present with this representation.

ace does not share a face with ad. Observe that the cycle c_{abc} defined by the faces (ab, bc, ac) is represented by a 1-chain $c_{abc} = [1\ 1\ 0\ 0\ 1\ 0\ 0]^T$ and $\partial_1 c_{abc} = 0$ which satisfies definition 4.6 for a 1-cycle. Remember that all addition is modulo-2. An homologous cycle $c_{abcd} \sim c_{abc}$ surrounds the same hole, such that the homologous cycle c_{abcd} differs by a boundary from a representative cycle c_{abc}. In other words, there exists a chain $c \in C_2(K)$ such that

$$c_{abcd} = c_{abc} + \partial_2 c \tag{26}$$

Where the boundary of c is $\partial_2 c$ and this boundary is added (modulo-2) to the representative cycle to result in the homologous cycle. In this example, $c = [1\ 0]^T$ is a 2-chain that contains only the 2- simplex acd. The boundary $\partial_2 c = [0\ 1\ -1\ 0\ 0\ 1\ 0]^T$ is the set of faces $\{ac, da, cd\}$, and the modulo-2 addition $c_{abc} = [1\ 1\ 0\ 0\ 1\ 0\ 0]^T + [0\ 1\ -1\ 0\ 0\ 1\ 0]^T = [1\ 0\ -1\ 0\ 1\ 1\ 0]^T = c_{abcd}$. Notice that the addition of the cycle and boundary cancelled out the face shared by both, namely ac.

4.3 Persistent Homology

Persistent homology is a tool from applied algebraic topology that characterizes the homology groups that persist over scale variations for noisy data. Understanding the global properties of a

topological space includes characterizing it's homeomorphism type, which involves understanding the geometry of the space up to stretching and bending, but not tearing and gluing. Determining the homeomorphism type is difficult, so an alternative is to characterize the *homology* of a discrete approximation of the space, then computing its simplicial homology groups. Informally, the homology of a topological space provides knowledge of number and type of holes in the space, such that the characterization of these holes provides a description of the global topological structure.

A fundamental challenge of computing topological features of real datasets is to determine which features are real ("signal"), which which features are artifacts of the discrete representation ("noise"). For example, given a collection of points x in Euclidean space $\mathbb{E}^n$, the *Rips complex* $\mathcal{R}_\varepsilon$ [86] is the abstract simplicial complex whose k-simplicies correspond to unordered $(k+1)$-tuples of points that are pairwise within distance ε. Figure 9 (top) shows an example of the Rips complex for a set of points in $\mathbb{E}^2$ for increasing ε. For this abstract simplicial complex, homology generators H_j can be computed, such that the rank of H_0 reflects the number of connected components, the rank of H_1 reflects the number of one dimensional holes and so on. Which is the "right" ε for this dataset? In other words, as ε increases, holes appear and disappear, which begs the question which holes are "real"? In general, without prior knowledge of the dataset generation, the true ε cannot be determined. However, observe that those topological features which persist across large changes in scale ε are unlikely to be due to topological noise. So, informally, let the persistent homology be the homology that persists across large changes in ε.

The persistent homology is computed as follows. Figure 9 (bottom) shows an example for a representation of the persistent homology called a *barcode* [81]. Informally, a barcode can be considered to be the persistence analogue of a Betti number. Recall that a Betti number captures the rank of a homology group, however a homology group is dependent on the abstract simplicial complex, which is dependent on a scale parameter. As the scale increases, some Betti numbers decrease and others increase, which correspond to the changing topology of the abstract simplicial complex. A barcode captures the changes in Betti numbers as a scale or persistence is changed. Figure 9 (bottom) shows a scale parameter ε as horizontal lines, the homology groups are shown ordered vertically, and the rank of $H_k(\mathcal{R}_{\varepsilon_i})$ for the k^{th} homology group H_k for a given Rips complex $\mathcal{R}_{\varepsilon_i}$ is the number of intervals intersecting the dotted lines. The rank of the homology group H_k is equal to the k^{th} Betti number β_k of the complex, which is a quantitative measure of the global topological structure. For example, β_0 is the number of connected components of the complex. At the second dotted line, the Rips complex contains six connected components, but as ε increases at the third dotted line, the Rips complex becomes fully connected. So, the rank of H_0 changes from $\beta_0 = 6$ to $\beta_0 = 1$, as shown by the intersection of the dotted lines with the red intervals. Similarly, at the fifth dotted line, the Rips complex is connected and there is only one hole remaining, so $\beta_0 = 1$ and $\beta_1 = 1$ as shown by the intersection of the dotted line with the red and green intervals.

One application of persistent homology is the topological analysis of the space of natural images [80][81]. In [80], the authors extended the approach of [87] to characterize the topological space $\mathcal{M}$ of projected 3x3 patches of natural images onto the seven dimensional unit sphere. Understanding the statistics of natural images is motivated by the need to model prior probability

distributions of local image patches for use in object recognition, object localization, segmentation, denoising, and compression. For example, in [87] the authors model the full probability distribution of high contrast 3x3 binary patches from natural images, by projecting suitably preprocessed 3x3 patches onto $\mathcal{S}^7$, a 7 dimensional unit sphere in $\mathbb{R}^8$. Results show that the projected distribution of patches in $\mathcal{S}^7$ are strongly clustered, and that a majority of the patches are concentrated in submanifolds in the unit sphere. This result shows that natural images are composed of basic image primitives which generate low dimensional non-linear structures where intrinsic dimension of these manifolds is fixed and independent of the embedding space dimension. Such analysis focuses on the probabilistic and geometric model of basic image primitives in natural images. Results of this analysis over a dense subset of points in $\tilde{M}$ showed that this topological space has $\beta_1 = 5$.

4.4 Homologous Cycle Matching

Optimal homologous cycle matching is the problem of finding the shortest cycle in the same homology class as a given cycle. Intuitively, in a 2D graph, given a cycle surrounding a hole in the graph, the optimal homologous cycle is the shortest cycle that surrounds the same hole. The primary result of this paper is that finding optimal cycles in a given homology class can be solved in polynomial time since it can be shown that the integer linear program for homology cycle matching has totally unimodular constraints, and therefore the linear programming formulation provides an integer solution.

In the context of object detection, homologous cycles and chains have the potential to elegantly capture global matching constraints. These constraints include surrounded, interior, connected which cannot be captured by only local methods. Using these tools from algebraic topology has not been demonstrated convincingly to provide improvements for object detection performance, however it may provide an additional set of tools for characterizing shape to augment geometric methods.

In this section, we will summarize the construction of an integer linear program for optimal homologous cycle matching.

4.4.1 Problem Definition

Optimal homologous cycle matching [52][88] is formulated as follows. Let c be a p-chain in a simplicial complex K, with n simplexes of dimension $p+1$ and m simplexes of dimension p. The optimal homologous chain problem is to find a p-chain x^* which has the minimum weighted l_1-norm $||Wx^*||_1$ among all chains homologous to c. Equation (27) shows the weighted l_1 optimization for homologous chains.

$$\begin{aligned} (x^*, y^*) = \arg\min \quad & ||Wx||_1 \\ \text{s.t.} \quad & x = c + \partial_{p+1} y \\ & x \in \mathbb{Z}^m, y \in \mathbb{Z}^n \end{aligned} \tag{27}$$

In this optimization, $\partial_{p+1} : C_{p+1} \to C_p$ is a boundary matrix mapping $(p+1)$-chains to p-chains and y is a $(p+1)$-chain.

The key constraint in this optimization is $x = c + \partial_{p+1} y$. Recall from section 4.2, that a homology group H_k is defined as $H_k(K) = \{z | z = z_0 + \partial_{k+1} c, c \in C_{k+1}(K)\}$ (defn 4.8), where an homologous cycle z differs by a boundary from a fixed representative cycle z_0. In this case, the known representative cycle is c, and the optimal homologous cycle x must differ by a boundary of a $(p+1)$-chain y. Therefore, the resulting p-cycle x is *homologous* to c, and there exists a $(p+1)$-chain y such that x and c differ by the boundary of y.

4.4.2 Integer Linear Program

Equation 27 includes an l_1 norm in the objective which is non-linear in x due to the absolute value of the l_1 norm. However, it is well known that an l_1 minimization can be rewritten as an linear program by adding slack variables and additional constraints.

$$\begin{aligned} (x^+, x^-, y^+, y^-)^* = \arg\min \quad & \textstyle\sum_i w_i (x_i^+ + x_i^-) \\ \text{s.t.} \quad & x^+ - x^- = c + \partial_{p+1}(y^+ - y^-) \\ & y^+, y^- \geq 0 \\ & x^+, x^- \leq 1 \\ & x^+, x^- \geq 0 \end{aligned} \tag{28}$$

Observe that the constraint $x \in \{-1, 0, 1\}$ is equivalent to the relaxed constraint $0 \leq (x^+, x^-) \leq 1$ if x^+ and x^- are restricted to integer solutions. This constrains x to have a natural geometric meaning for chain coefficients without explicitly requiring that $x \in \mathbb{Z}_2$, which leads to an intractable optimization.

The integer linear program in equation 28 can be written with linear inequality constraints of the form $Ax \geq b$, such that $x = [x^+ \; x^- \; y^+ \; y^-]^T$ and

$$A = \begin{bmatrix} -I & I & \partial_{p+1} & -\partial_{p+1} \\ I & -I & -\partial_{p+1} & \partial_{p+1} \\ -I & 0 & 0 & 0 \\ 0 & -I & 0 & 0 \\ I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \quad b = \begin{bmatrix} -c \\ c \\ -1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{29}$$

It can be shown that if the boundary matrix ∂_{p+1} is totally unimodular, then the constraint matrix A is totally unimodular, and therefore the linear program in (28) has an integer solution and can be solved exactly in polynomial time.

4.4.3 Total Unimodularity and Boundary Matrices

An integer linear program can be solved in time polynominal in the dimensions of the constraint matrix A if and only if the constraint matrix is totally unimodular. The following lemma is proved in [89].

Lemma 4.11. *If a matrix A is totally unimodular (TUM) then a matrix A' obtained from A by any of the following operations is also TUM*

- $A' = A^T$
- $A' = [A, I]$
- *A' is obtained from gauss jordan pivoting*
- *Adding one or more rows or columns with all zeros and a single one.*
- *Removing a row or column from A*
- *Adding to A one or more rows or columns already in A*
- *Multiplying a row or column by -1*
- *Permuting rows or columns*

Lemma 4.12. *If ∂_{p+1} is totally unimodular, then the constraint matrix A in (29) is totally unimodular.*

Proof: The proof uses the properties for totally unimodularity described in lemma 4.11. Let B be of size $M \times N$. If B is TUM, then $[B\ B]$ is TUM by applying property five to repeat columns of B. If $[B\ B]$ is TUM, then $[B\ -B]$ is TUM by applying property seven to multiply the first N columns by -1. If $[B\ -B]$ is TUM then $[I\ I\ B\ -B]$ is TUM by applying property four. If $[I\ I\ B\ -B]$ is TUM then $[-I\ I\ B\ -B]$ is TUM by applying property seven. Observe that this form is the same as the first block row of A in (29). Let this block row be A_1. If A_1 is TUM, then $[A_1^T\ -A_1^T]^T$ is TUM by applying properties one, six and seven. Let this matrix be A_{12}. If A_{12} is TUM, then A is TUM by applying properties four and six. Therefore, since $B = \partial_{p+1}$, if ∂_{p+1} is TUM, then A is TUM. □

Lemma 4.13. *For a finite simplicial complex triangulating a $(p+1)$-dimensional compact orientable manifold, ∂_{p+1} is totally unimodular.*

Proof: Every p-face is a face of either one or two $(p+1)$-simplex. For example, in graph theory, every node (0-face) has an endpoint at either one or two edges (1-simplex). Therefore, the boundary matrix ∂_{p+1} will have row entries of at most two non-zeros. Since it is known that a consistent orientation of $(p+1)$-simplexes always exists for a finite triangulation of a compact orientable manifold, the row entries for ∂_{p+1} with two nonzeros always contain a +1 and -1.

It is known that a matrix A is totally unimodular if rows of A can be partitioned into two disjoint sets A' and A'' such that the following four properties hold (i) every column of A contains at most two nonzeros (ii) every entry is $\{0,1,-1\}$, (iii) If two nonzero entries of A have the same sign, then

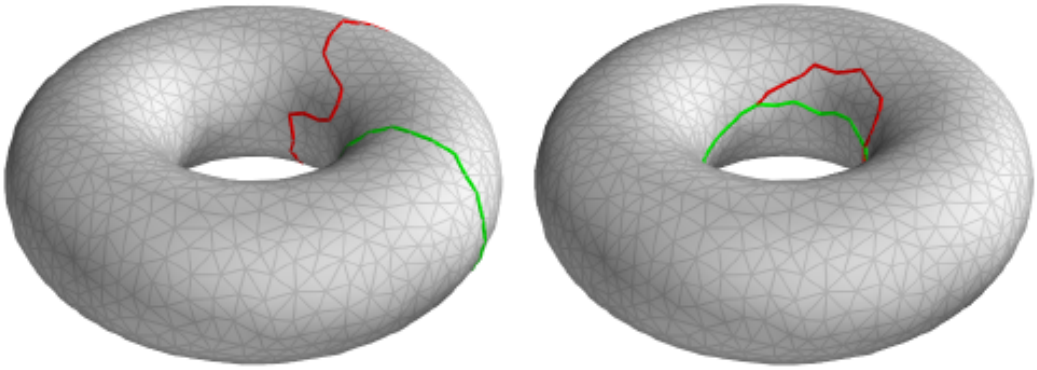


Figure 10: Optimal Homologous Cycle Matching [52]. Given a reference cycle (red) for a given homology group, optimal homologous cycle matching finds the minimum weight homologous cycle (green).

the row of one is in A' and the other in A" (iv) If two nonzero entries of A have opposite signs, then both rows are in either A' or A". Consider $A = \partial_{p+1}^T$. We have shown (i), (ii) follows from construction of a boundary matrix, we have shown that (iii) for orientable manifolds and (iv) we can construct matrices $A' = A$ and $A'' = \emptyset$ [89]. Therefore, A is TUM, and if A is TUM, then A^T is TUM from property one of lemma 4.11. Therefore, ∂_{p+1} is TUM. □

Finally, lemma 4.13 shows that the boundary matrix ∂_{p+1} is totally unimodular, therefore from lemma 4.12 the constraint matrix A in (29) is totally unimodular, and the linear program in (28) provides an integer solution.

4.4.4 Results

The main result is the linear program in equation 28, and a characterization of those simplicial complexes for which the boundary matrix is totally unimodular. This characterization enables practical use of this result. An example result is shown in figure 10, which shows a reference cycle in red, and an optimal homologous cycle in green for the simplicial complex shown.

The total modularity for boundary matrices that are useful for shape representations were summarized in lemma 4.13. The paper also presents extensions for more general cases such as non-orientable manifolds and abstract simplicial complexes, however since these results are not directly relevant for shape representations (shape representations can assume an embedding in $\mathbb{R}^2$), we do not summarize them here.

5 Analysis

In this survey, we organized graph based shape representations into geometric and topological methods. We gave an overview of geometric methods with an organization by graph structure, describing trees, bipartite graphs and general graphs, and graph construction and graph matching approaches for each. We showed graph matching results and described the computational complexity for representational approaches for each graph structure. We described topological methods for shape representation using a unifying framework of simplicial homology, where homology was computing using persistent homology and matched using homologous cycle matching. We showed how this type of topological representation could be used to capture global topological properties for use in shape representations to complement geometric methods.

In this section, we analyze the performance of the graph based shape representations described so far in both a task independent and task dependent manner. *Task independent* analysis refers to comparing and contrasting the representational power and matching efficiency of different graph based approaches independent of the computer vision task that this representation would be applied to. Section 5.1 shows a comparison table and describes the tradeoff of representational power vs. computational tractability inherent with graph based representations. We compare different shape representations using a unifying criteria of structure, attributes, construction and matching, and we compare shape representations using graph structures and matching algorithms, by considering representational power of the graph structure and the efficiency of the matching algorithm.

Task specific analysis considers performance for graph based shape representations in a particular vision task, such as object detection. Unfortunately, the dirty little secret of graph based representations is that they do not perform as well as discriminative template based methods for object detection, recognition or categorization. Note that this is the case also for generative methods, such as probabilistic graphical models, but as discussed in section 1.1, such methods are not included in the scope of this analysis. In fact, all the methods discussed so far when applied to object detection augment graph matching for detection using either (i) a final discriminative classifier, commonly an SVM or randomized forest [70, 68, 30, 22] or (ii) local distance learning [90, 91] to optimize a distance metric for nearest neighbor classification. These classifiers are practical since they result in improved detection performance by reducing the lower false alarm rate for spurious detections on background features. However, they have a highly questionable foundation for shape representation. In section 5.4 we will describe why these methods inherit the warts of classical categorization and should be treated with skepticism.

In the remainder of this section, we ask broader questions. We seek to understand if graphs and graph matching are an appropriate shape representation for practical vision tasks such as object detection, or if they have fundamental limitations or if there are theoretical problems with the representational power. Specifically we ask:

- **Is object detection geometric?** If we have perfect graph matching that optimally preserves geometry and topology of a reference graph during matching, have we solved the object detection problem? We show simple counter-examples where graph matching alone does not

Graph	Matching	Attributes	Explicit?	Reoresentative example
star	GHT	prototypes	no	ISM [26, 50]
star	DP	discriminative templates	no	[20]
star	linear assignment	contours	no	[30]
star	GHT	regions	no	[70]
bipartite	perfect matching	local descriptors	yes	shape context [28]
tree	tree edit distance	weight matrix	yes	[64][63][62]
tree	tree isomorphism	inflection points	yes	Shock graph [29]
general	QAP	weight matrix	yes	Annealing [51], spectral [59]
general	TPS	local descriptors	yes	[27]
simplex	Homologous cycles	weight matrix	yes	[52]

Table 2: Comparison of graph based shape representations

capture the similarity of two objects. In general, graph matching is an example of a representation based on a *first order isomorphism*, but a more powerful and expressive representation is based on *second order isomorphisms*. We describe work by Shepard [92] and Edelman [4] outlining this idea. This is addressed in section 5.2.

- **Is object detection topological?** If we have perfect topological shape representations such as simplicial homology and optimal matching of topological representations that preserves topological invariants, have we solved object detection? Clearly, this answer is no since topological invariants alone do not capture perceptual similarity. We describe a simple counter example, but sketch out how topological invariants can be useful to augment geometric based representations. This is addressed in section 5.3.
- **Are discriminative classifiers classical categorization revisited?:** Does the final discriminative classifier step introduced by most graph matching approaches to reduce false alarm rates and handle the dirty little secret come at a cost? Are discriminative classifiers the way forward for shape representations, or does it make an implicit assumption on a discredited theory from cognitive science? In section 5.4, we argue that the final discriminative step is on a shaky foundation, and that this is not a credible path forward, even though it looks to improve performance.

5.1 Task Independent Comparison

These methods presented in this survey can be compared by considering the following criteria: graph structure, attributes, construction and matching. The graph structure refers to the underlying graphical structure, where in this survey we described trees, bipartite graphs, general graphs and simplicial complexes. The attributes refer to the edge and node features used to describe the local shape of the image at the graph embedding coordinates. Some work considers specific attributes to define the local shape, and other consider a graph structure that requires only a weight matrix to

be defined. The graph construction can either be explicit or implicit, where explicit construction means that the graph is embedded prior to matching and implicit means that the construction is coupled with the matching. Finally, the graph matching defines the algorithm used for alignment of the reference graph with the current observed image.

Table 2 shows a comparison of graph based shape representations. This table includes the approaches reviewed in this survey, as well as other approaches that were not discussed in detail, but are representative of graph based shape representations.

The primary tradeoff in comparing in graph based representations is representational power versus computational tractability. General graphs encode all relevant pairwise geometry and global topology that capture shape, however matching for a general graph is equivalent to a quadratic assignment problem which is computationally intractable. Section 3.3 discussed two relaxations of the QAP, a graduated assignment algorithm based on graduated non-convexity, and a spectral relaxation. These algorithms match general graphs, however the final result is a fractional assignment, which must be discretized to a final matching output. The graduated assignment provides an iterative discretization using softassign and annealing, however it is unclear how to choose the annealing schedule in general to avoid being trapped in local minima. The spectral relaxation relaxes the matching constraints to perform a convex quadratic optimization, then enforcing the constraints during the discretization step. The final result is an assignment matrix, but it is not an optimal assignment matrix, since the discretization is a convenient heuristic to recover a discrete solution from a fractional optimization, unrelated to the original objective.

Sections 3.1 and 3.2 described bipartite graphs and trees, graph structures which enable efficient matching algorithms. However, these graphs do not capture the full geometry of a shape, and therefore lack expressive representational power. For example, section 3.1.1 described a star graph used in the implicit shape model. This tree structure cannot capture pairwise geometry between parts, and can only capture the geometry between parts and the centroid. This limits the representational power, which can be seen in figure 4. The probabilistic voting assumes that each codebook entry is independent, and each casts a separate vote for the object centroid according to the star model. This independence assumption clearly does not hold in general since two parts, say the legs and torso, are correlated. The star graph and independence assumption is a convenient fiction to enable efficient matching using the generalized Hough transform. Similarly, a bipartite graph cannot capture any pairwise geometry, and as described in section 3.2, it is assumed that the shape context attribute will capture this pairwise interaction rather than having it expressed as geometric constraints during matching. However, these weights are not constraints to be enforced, and the result is that each node is independently matched according to weight, resulting in the same lack of representational power as star graphs.

A similar problem of representational power vs. computational tractability exists in the generative model community. Stuart Geman describes this as the *Markov dilemma*. How do we model constraints on attributes such as poses of pairs eyes while having a Markov network that is modelling eyes as conditionally independent given a face, without giving up computational tractability? [24]. The fact that probabilistic graphical models also share this representational limitation hints

that it is universal graph representations, which fundamentally limits the representational power.

5.2 Is Object Detection Geometric?

Object detection is the problem of localizing the position and scale of a bounding box surrounding an instance of an object category in an input image. Clearly geometry plays a role in object detection, as can be shown by dramatic performance improvements of methods that use shape representations that include even weak geometry [12] versus texture-only classification [93]. However, is object detection just alignment to a reference graph that preserves geometry during matching?

Figure 11 shows two examples of where alignment fails to capture perceptual similarity. Figure 11 (left) shows two reference images, a dotted square and a triangle, and an observed image of a square. Assume that these reference images and observation have an explicit graph embedding, such that the attribute weights for graph matching are proportional to geometric deformation. In both cases, the reference models will match to 50% of the observation, which from alignment would result in these triangle model and the dotted square as equally similar to the observed square. Clearly, we perceive the dotted square as more similar to the observation that the triangle, so alignment is not capturing this perceptual similarity. Figure 11 (right) again shows two models, a handwritten "O" and a typographic "Q", and an observation of a typographic "O". The handwritten "O" does not align anywhere with the observation, while the "Q" aligns everywhere except the most important, yet small, discriminative stroke. Alignment alone would define the "Q" more similar to the "O", while perceptually we would consider the handwritten "O" more similar. So, clearly there is more to perceptual similarity than geometric alignment alone. However, alignment does provide a cue for similarity, since the dotted square or the handwritten "O" are more similar to the observation than say a cat or a motorcycle.

These examples show that similarity is more than alignment, it requires *relevant alignment*. The same issue of representation has been explored in the categorization literature. For example, Murphy and Medin make the following observation for the use of common attributes for object categorization as described by Edelman [4]:

> *The number of attributes shared by plums and lawn-mowers could be infinite: both weigh less than 1000 kilograms (and less than 1001 kilograms), both cannot hear well, both have a smell, etc. Any two entities can thus be arbitrarily similar or dissimilar, depending on what is to count as a relevant property.*

This observation is an example of the *ugly duckling theorem* which states informally that similarity requires bias, since otherwise is if similarity is judged in terms of number of predicates shared then any two objects are equally similar. The same holds for shape representations. In figure 11, the relevant alignments are not all precise deformations of the handwritten O, but rather discriminative properties of the holistic shape. This begs the question, what is important? This seems to imply that discriminative classifiers are necessary, since these classifiers are designed to weight features that

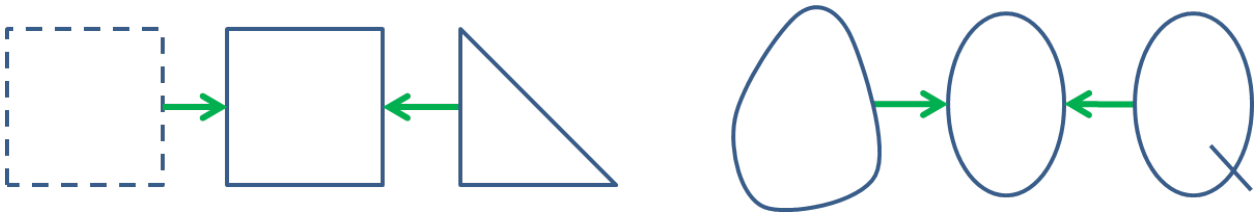


Figure 11: Alignment does not capture all similarities

are important for classification, but we will see that they also inherit the deficiencies of classical categorization. So, alignment is useful, but it does not tell the whole story.

5.3 Is Object Detection Topological?

Section 4 described using homology and homologous cycles for shape representation, but is this sufficient for object detection? Clearly, topology alone does not capture a representation of objective shape. Topology can be described as qualitative geometry, and the invariants that remain once geometric concepts are removed. However, objective shape is a geometric concept, as shown by our clear perception of bagels and coffee cups as different objects, even though as topological shapes they have the same Betti numbers, and therefore the same topological invariants.

However, homologous cycles and chains do have the potential to elegantly capture global matching constraints in perceptual organization, such as "surrounded", "interior" and "connected". These constraints are *global* which capture the holistic properties of a shape which cannot be captured by local node or edge interactions only, and these constraints are independent of geometric deformation. These topological invariants may provide robustness to intraclass variability than geometric attributes only. For example, topological features can be used for recognition of people and objects, such as pedestrians carrying a bag or pushing a stroller may exhibit significant variation in geometric configuration, however the topological invariants remain stable (e.g. the loop of a bag being held, a baby surrounded by and interior to a stroller). These topological attributes provide an invariant shape representation that may provide improved generalization.

So, while use of topology to represent these global properties has not yet been demonstrated convincingly, there is the potential to provide a combined geometric and topological framework that is more powerful that either independently. So, topology may be useful, but it does not tell the whole story.

5.4 Are We Revisiting the Classical Theory of Categorization?

The classical theory of categorization states that categories are defined by necessary and sufficient conditions for membership [46, 47]. For example, the category of "triangle" can be defined unambiguously, such that any subset of line segments can be checked for consistency with the definition.

So, if a subset of line segments contains three segments, and if each pair of line segment endpoints intersect at a unique point, then the subset is categorized as "triangle". The conditions are *necessary* in that they must be satisfied to make a categorization decision, and they are *sufficient* in that even if there are other properties present such as the lines colored red or dotted lines, the minimum properties are enough to render a categorization decision. The necessary and sufficient conditions define the category, so this is often called a *definitional* approach to categorization. This approach can be traced back to Aristotle who discussed the essence of a category as those properties shared by all members.

The definitional approach has a number of implied properties. First, an object is either in the category or not. If a subset of line segments satisfies the definition of a triangle, then either it is a triangle or it is not, there are no "in betweens". This is often called the property of the excluded middle. Second, there are no distinctions between category members. As long as the necessary and sufficient conditions are satisfied, then the object is assigned membership to a category. There are no "better" or "worse" examples of a category, since all members are equal in satisfying the definition.

The classical view appears intuitive, and it does provide unambiguous categorization for some objects with a definitional nature, but has been widely discredited as a general representation. Consider the category of "dogs", as described by Murphy [46]:

> *The definition for dogs...namely things that have four legs, bark have fur, eat meat and sleep is obviously not true. Does something have to have four legs to be a dog? Indeed, there are unfortunate dogs who have lost a leg or two. How about having fur? Although most dogs do have fur, there are hairless varieties like chihuahuas that don't. What about barking? Almost all dogs bark, but I have known a dog that lost its bark as it got older. This kind of argument can go on for some time when trying to arrive at necessary features...Wittgenstein urged his readers not to simply say 'there must be something in common' but to specify the things in common. Indeed, it turns out to be very difficult to specify necessary and sufficient conditions of* most *real world categories.*

This problem is not limited to animals. Murphy describes the practical problem of categorization in material science by as described by distinguished metallurgist Robert Pond [46]

> *You really don't know what a metal is, and there's a big group of people that don't know what is a metal is. Do you know what we call them? Metallurgists! ... Here's why metallurgists don't know what a metal is. We know that a metal is an element that has metallic properties. So, we start to enumerate all these properties: electrical, conductivity, thermal conductivity, ductility, malleability, strength, high density. Then you say, how many of these properties does an element have to have to classify as a metal?...We can't get the metallurgists to agree!...So, we just proceed along presuming that we are all talking about the same thing.*

These anecdotal examples of the problems facing definitional approach to categorization can be traced back to Wittgenstein, who was the first to question the assumption that categories could be defined. For example, Wittgenstein describes the category of *games* and argues that there are no common definitional properties common to all games. Rather, games share family resemblances such as amusement, rule following or competition. From Lakoff, "Games, like family members, are similar to one another in a wide variety of ways. That, and not a single well defined collection of common properties is what makes game a category." [47].

Eleanor Rosch and her work on prototype theory provided a set of empirical evidence to challenge and eventually discredit classical categorization [94]. She made two key observations. First, she observed that if categories are definitional, then no member should be any better example of the category than any other. Rosch provided experimental evidence that all categories have best examples called *prototypes*, such as a robin being a prototype of a bird and not an ostrich. These central members are more commonly associated with the category, are more quickly labelled than non-central members, children learn them more quickly, and they exhibit more family resemblances to other members of the category than non-central members. Second, she observed that if category membership is definitional, then there should be no ambiguity as to category membership and categorization should not be observer dependent. However, experimental evidence exists for categories that are ambiguous (is swampwater water or poison?) and categorization that changes from person to person (is a tomato a fruit or vegetable?).

These observations are not explained by classical categorization theory, which puts this theory on shaky foundation. Murphy summarizes the arguments against classical theory as follows [46]:

> *It has simply ceased to be a serious contender in the psychology of concepts...First, it has been extremely difficult to find definitions for most natural categories...Second, the phenomenon of typicality and unclear membership are both unpredicted by the classical view...Third, the existence of intransitive category decisions (car seats are chairs, chairs are furniture, but car seats are not furniture).*

In short, classical categorization using abstract absolute definitions does not explain experimental evidence for how we perform categorization. It is unlikely that there are "essences" that define objects or "natural kinds" that define animals or plants. As Darwin said, "We shall have to treat species as artificial combinations made for convenience in order to be free from the vain search for the undiscovered and undiscoverable essence of the term 'species' ".

However, even though classical categorization has been discredited, the literature on object detection implicitly relies on it. Object detection requires both localization and categorization, however for the moment, consider only categorization part. Let's assume that object categorization is limited to basic level categories, then we ask the question, is object categorization equivalent to feature space classification? Feature space classification refers to embedding a labelled training set into a common feature space, then learning a discriminative classifier. Every training image is represented as a point in a common feature space, and a learning framework optimizes a discrim-

inative classification function to partition this space. We will show that this assumption *implicitly assumes the classical view of categorization*, and inherits its warts.

Consider the popular "bag of words" approach to object classification [93]. In this approach, an image is represented by the number of times a set of local prototypes or *visual words* appear in an image, independent of the relative position of these words. Each training image is represented by a histogram which captures the frequency of this unordered "bag" of visual words. Each training image is represented by a histogram, and the set of histograms from the training set serve as observations input to a support vector machine (SVM) classifier. The weights learned by this classifier represent the category. What does this approach say about a category? The category is defined in a *feature space*, such that each training image is represented as a point in this feature space. For bag of words models, the dimensions of the feature space correspond to the frequency of prototypes occurring in a given image. The weights learned for a (one vs. all) linear support vector machine defines an optimal hyperplane in this feature space which separates category members from non-members. Points one one side of the hyperplane are declared category members and points on the other side are not members. The SVM weights which define the hyperplane parameters define a "rule" that can be used to define the category in terms of the weights in each of the feature space dimensions. For a given observation, the SVM weights define necessary and sufficient conditions for category membership, and either an object is a category member or not. In other words, classical categorization.

This argument is not limited to SVMs. Any discriminative classifier that learns a classification function in a fixed feature space is learning a categorization rule. This is true by construction, since a classification function is a mapping from feature space to label, which encodes a (complicated and opaque and non-obvious) rule for assigning observations to categories. However, as described in section 5.4 the classical theory has been widely discredited as a category representation by cognitive scientists. Why then are they so popular? Critics of the classical view claim that the definitional approach is flawed since they themselves cannot write down a definition for a natural category such as dogs. This does not mean that a category definition does not exist, since the definition may be hidden in the data, and can only be uncovered by statistical analysis. However, if experts cannot agree on those features that are needed for categorization, and if a discriminative classifier can capture the essence of a category, then why don't the experts use a discriminative classifier to settle the debate? Perhaps the classifier cannot capture an optimal statistical rule because a rule does not exist.

So, object detection approaches that use feature space classification appears to be a revisitation of the classical theory of categorization. What are the alternatives? Edelman argues that shape representation should be a *second order isomorphism* [2][4]. An isomorphism refers to a functional mapping that is one to one and onto. Formally, a function f is an isomorphic map if and only if it is a bijection (see also section 2.1). In the context of shape representations, a first order isomorphism refers to correspondence between a model representing distal properties and an internal representation or proximal representation. For example, 3D reconstruction attempts to create an internal 3D representation for an object that is isomorphic with 3D shape of an object in the

scene, or graph matching attempts to create an internal graphical representation of an object that is isomorphic to a reference. In contrast, a second order isomorphism [92] refers to the relationships between representations and not the representations themselves. From Shepard [92]:

> *Although the internal representation for a square need not itself be square, it should (whatever it is) at least have a closer functional relation to the internal representation for a rectangle than to that, say, for a green flash or the taste or persimmon.*

Second order isomorphism and first order isomorphism or representation *of* similarity instead of representation *by* similarity. "The idea of second order isomorphism translates to the notion that only certain relations between the objects - not the shape of individual objects themselves - need be represented" [4]. Second order isomorphism neatly bypasses the question of the "right" representation. Instead of trying to find the ideal representation that best captures an object category, second order isomorphism asks for a given representation, do the proximal relations preserve the distal relations? In other words, I may not know how to represent the category of "dog", but for any choice I make for representing shape of an individual dog, the shape of a golden retriever should be more similar to a Doberman than a bus. This focuses on the representation of similarity rather than the representation of an object category. This sentiment is shared in recent work in object categorization on exemplar based similarity [95, 96].

An approach like second order isomorphisms neatly bypass the limitation of classical categorization, since there are no global rules defining categories, there are only similarity relations among data. Similarity computations are delayed until test time, which delays the classification decision until test time, which does not require a predefined category definition learned during training. In essence, the data is it's own definition. In other words, in a complex world, perhaps the best model of the world is the world itself [97].

6 Conclusions

In this report, we surveyed graph based shape representations by grouping into geometric methods and topological methods, where each method is organized by invariances of increasing abstraction. For geometric methods, we used the unifying framework of weighted graph matching posed as a quadratic assignment problem as a unifying framework for discussion, and we described invariant properties maintained during various tree, bipartite and general graph matching approximations. For topological methods, we used the unifying framework of simplicial homology, and describe the persistent homology, a technique for recovering the homology given noisy data, and optimal homologous cycle matching for matching topologically invariant cycles. Finally, we analyzed these methods both in a task independent and object detection task dependent context.

The conclusions are that graph based shape representations for object detection based on graph matching to an exemplar, do provide both a measure of geometric and topological similarity. However, they do not provide equivalence to perceptual similarity, they do not embody Edelman's sec-

ond order isomorphism, and they cannot provide the same performance as a discriminative classifier. However, since discriminative classifiers have their own representational problems, more work is needed to cross the representational gap.

David Weinberger in "Everything is Miscellaneous" [98] observes the same representational problems with knowledge that we outlined with shape representations. The Dewey decimal system for categorizing knowledge is good for for searching for books by author, but not for searching for a birthday present. Alphabetization is good for categorizing knowledge in printed encyclopedias, but only if you know the name of what you are looking for. These and other problems of categorization he attributes to the fact that there is no one categorization model for knowledge. He observes, "The world is too diverse for any single classification system to work for everyone in every culture at every time...The best representation depends on the task.". To compensate, he outlines four strategic principles for organizing knowledge: (i) Filter on the way out not on the way in (ii) Put each leaf on as many branches as possible, (iii) Everything is meta-data and everything can be a label and (iv) Give up control. The next generation of shape representations may be down a similar path.

References

[1] S. Palmer, *Vision Science: Photons to Phenomenology*, MIT Press, 1999.

[2] S. Edelman, "Representation, similarity, and the chorus of prototypes," *Minds and Machines*, pp. 45–68, 1995.

[3] S. Edelman, "Representation is representation of similarities," *Behavioral and Brain Sciences*, vol. 21, no. 4, pp. 449–467, 1998.

[4] S. Edelman, *Representation and Recognition in Vision*, The MIT Press, 1999.

[5] S.J. Dickinson, A. Leonardis, B. Schiele, and M. Tarr, *Object Categorization*, chapter The Evolution of Object Categorization and the Challenge of Image Abstraction, Cambridge University Press, 2009.

[6] S. Palmer, *Object perception: Structure and Process*, chapter Reference frames in the perception of shape and orientation, pp. 121–163, Lawrence Erlbaum Associates, 1989.

[7] D. Marr, *Vision: A computational investigation into the human representation and processing of visual information*, MIT Press, 1982.

[8] I.L. Dryden and K.V. Mardia, *Statistical Shape Analysis*, Wiley, 1998.

[9] T. F. Cootes, D. H. Cooper, C. J. Taylor, and J. Graham, "A trainable method of parametric shape description," *Image and Vision Computing*, vol. 10, no. 5, pp. 289–294, June 1992.

[10] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE TPAMI*, vol. 23, no. 6, pp. 681–685, 2001.

[11] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *In CVPR*, 2005, pp. 886–893.

[12] C. Schmid, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *In CVPR*, 2006, pp. 2169–2178.

[13] J. Mutch and D. Lowe, "Object class recognition and localization using sparse features with limited receptive fields," *International Journal of Computer Vision (IJCV)*, vol. 80, no. 1, pp. 45–57, October 2008.

[14] T. Serre, L. Wolf, and T. Poggio, "Object recognition with features inspired by visual cortex," in *CVPR*, 2005.

[15] P. Viola and M. Jones, "Robust real-time object detection," vol. 57, no. 2, pp. 137–154, 2001.

[16] U. Grenander and M. Miller, *Pattern Theory: From Representation to Inference*, chapter , Oxford University Press, 2007.

[17] N. Pinto, Y. Barhomi, DD Cox, and JJ DiCarlo, "Comparing state-of-the-art visual features on invariant object recognition tasks," in *EEE Workshop on Applications of Computer Vision (WACV 2011)*, 2011.

[18] L. Zhu, Y. Chen, and A. Yuille, "Learning a hierarchical deformable template for rapid deformable object parsing," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 32, no. 6, pp. 1029–1043, 2010.

[19] Y. Amit and P. Trouve, "Patchwork of parts models for object recognition," *IJCV*, vol. 75, no. 2, November 2007.

[20] P.F. Felzenszwalb and D.P. Huttenlocher, "Pictorial structures for object recognition," *IJCV*, vol. 61, pp. 2005, 2003.

[21] *Object Categorization: Computer and Human Vision Perspectives*, chapter Learning Hierarchical Compositional Representations of Object Structure, Cambridge University Press, 2009.

[22] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *CVPR*, 2008.

[23] S.C. Zhu and D. Mumford, "A stochastic grammar of images," *Found. Trends. Comput. Graph. Vis.*, vol. 2, pp. 259–362, January 2006.

[24] Y. Jin and S. Geman, "Context and hierarchy in a probabilistic image model," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, 2006, pp. 2145–2152.

[25] L. Zhu, Y. Chen, A. Torralba, W. Freeman, and A. Yuille, "Part and appearance sharing: Recursive compositional models for multi-view multi-object detection," in *CVPR*, 2010.

[26] B. Leibe, A. Leonardis, and B. Schiele, "Combined object categorization and segmentation with an implicit shape model," in *In ECCV workshop on statistical learning in computer vision*, 2004, pp. 17–32.

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

[28] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 24, pp. 509–521, April 2002.

[29] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker, "Shock graphs and shape matching," *International Journal of Computer Vision*, vol. 35, no. 1, pp. 13–32, 1999.

[30] P. Srinivasan, Q. Zhu, and J. Shi, "Many-to-one contour matching for describing and discriminating object shape," in *CVPR*, 2010.

[31] A Lehmann, B. Leibe, and L. Van Gool, "Prism: Principled implicit shape model," in *BMVC*, 2009.

[32] S. Loncaric, "A survey of shape analysis techniques," *Pattern Recognition*, vol. 31, no. 8, pp. 983–1001, August 1998.

[33] Remco C. Veltkamp and Michiel Hagedoorn, "Principles of visual information retrieval," chapter State of the art in shape matching, pp. 87–119. Springer-Verlag, London, UK, UK, 2001.

[34] B. Munsell, P. Dalal, and S. Wang, "Evaluating shape correspondence for statistical shape analysis: A benchmark study," *PAMI*, vol. 30, no. 11, pp. 2023–2039, November 2008.

[35] S.V.N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt, "Graph kernels," *Journal of Machine Learning Research*, vol. 11, pp. 12011242, April 2010.

[36] S. Ullman, *High Level Vision*, MIT Press, 1996.

[37] L. Zhu, C. Lin, H. Huang, Y. Chen, and A. L. Yuille, "Unsupervised structure learning: Hierarchical recursive composition, suspicious coincidence and competitive exclusion," in *ECCV*, 2008.

[38] A. Leonardis, "Evaluating multi-class learning strategies in a generative hierarchical framework for object detection," in *NIPS*, 2009.

[39] I. Kokkinos and A.L. Yuille, "Hop: Hierarchical object parsing," in *CVPR*, June 2009.

[40] B. Ommer and J. Buhmann, "Learning the compositional nature of visual object categories for recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 99, 2010.

[41] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories," in *IEEE CVPR04 Workshop on Generative-Model Based Vision*, 2004.

[42] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 1527-1554, 2006.

[43] K. Grauman and B. Leibe, "Visual object recognition," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 5, no. 2, pp. 1–181, April 2011.

[44] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 27, no. 10, pp. 1615–1630, 2004.

[45] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors - survey," *CVG*, vol. 3, no. 1, pp. 1–110, 2008.

[46] G. L. Murphy, *The Big Book of Concepts*, MIT Press, 2002.

[47] G. Lakoff, *Women, Fire and Dangerous Things: What categories reveal about the mind*, University of Chicago Press, 1987.

[48] M. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24, no. 1, pp. 34–58, 2002.

[49] N. Werghi, "Segmentation and modeling of full human body shape from 3-d scan data: A survey," *IEEE transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2007.

[50] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *International Journal of Computer Vision Special Issue on Learning for Recognition and Recognition for Learning*, vol. 77, pp. 259–289, 2008.

[51] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, 1996.

[52] T. K. Dey, A. Hirani, and B. Krishnamoorthy, "Optimal homologous cycles, total unimodularity, and linear programming," in *42nd ACM Sympos. Comput. Theory (STOC 2010)*, 2010, pp. 221–230.

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

[54] R. Diestel, *Graph Theory*, Springer-Verlag, 2010.

[55] H. Bunke, "Graph matching: Theoretical foundations, algorithms, and applications," in *In Proc. Vision Interface 2000*, 2000, pp. 82–88.

[56] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty years of graph matching in pattern recognition," *IJPRAI*, vol. 18, no. 3, pp. 265–298, May 2004.

[57] J. R. Ullman, "An algorithm for subgraph isomorphism," *J. Assoc. Comput. Mach*, vol. 31, no. 42, 1976.

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

[59] T. Cour, P. Srinivasan, and J. Shi, "Balanced graph matching," in *Advances in Neural Information Processing Systems (NIPS)*, 2006.

[60] C. Schellewald and C. Schnorr, "Probabilistic subgraph matching based on convex relaxation," in *In Energy Minimization Methods in Computer Vision and Pattern Recognition*, 2005.

[61] H. Bunke, "Error correcting graph matching: On the influence of the underlying cost function," *IEEE Trans. PAMI*, vol. 21, 1999.

[62] P. Bille, "A survey on tree edit distance and related problems," *Theoretical Computer Science*, vol. 337, pp. 217–239, 2005.

[63] D. Shasha and K. Zhang, "Simple fast algorithms for the editing distance between trees and related problems," *SIAM J. Comput*, vol. 16, no. 6, pp. 1245–1262, 1989.

[64] P.N. Klein, "Computing the edit-distance between unrooted ordered trees," in *In Proceedings of the 6th annual European Symposium on Algorithms (ESA)*. 1998, pp. 91–102, Springer-Verlag.

[65] X. Gao, B. Xiao, D. Tao, and X. Li, "A survey of graph edit distance," *Pattern Analysis and Applications*, vol. 13, January 2010.

[66] H. Jiang, S. X. Yu, and D. R. Martin, "Linear scale and rotation invariant matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

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

[68] S. Maji and J. Malik, "Object detection using a max-margin hough transform," in *CVPR*, 2009, pp. 1038–1045.

[69] A. Opelt, A. Pinz, and A. Zisserman, "A boundary-fragment model for object detection," in *ECCV*, 2006, p. 575588.

[70] C. Gu, J. Lim, P. Arbelez, and J. Malik, "Recognition using regions," in *CVPR*, 2009.

[71] J. Gall and V. Lempitsky, "Class-specic hough forests for object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*, 2009.

[72] O. Barinova, V. Lempitsky, and P. Kohli, "On detection of multiple object instances using hough transforms," *PAMI*, 2012.

[73] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, May 2002.

[74] R. Fergus, P. Perona, and A. Zisserman, "Object class recognition by unsupervised scale-invariant learning," in *CVPR*, 2003, pp. 264–271.

[75] D.H. Ballard, "Generalizing the hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, pp. 111–122, 1981.

[76] J. Kleinberg and E. Tardos, *Algorithm Design*, Addison Wesley, 2005.

[77] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2000.

[78] T. Dey, H. Edelsbrunner, and S. Guha, "Computational topology," *Advances in Discrete and Computational Geometry*, pp. 109–143, 1999.

[79] Bern et. al, "Emerging challenges in computational topology," in *Results of the NFS Workshop on Computational Topology*, 1999.

[80] G. Carlsson, T. Ishkhanov, V. de Silva, and A. Zomorodian, "on the local behavior of spaces of natural images," *International Journal of Computer Vision*, vol. 76, pp. 1–12, January 2008.

[81] R. Ghrist, "Barcodes: the persistent topology of data," *Amer. Math. Soc. Current Events Bulletin, Jan. 2007. Revised version in Bull. Amer. Math. Soc*, vol. 45, pp. 61–75, 2008.

[82] A. Hatcher, *Algebraic Topology*, Cambridge University Press, 2002.

[83] H. Edelsbrunner and J. Harer, "Persistent homology a survey," *AMS*, 2007.

[84] D. Freedman and C. Chen, "Algebraic topology for computer vision," *Nova Science*, 2011.

[85] T. E. Goldberg, "Combinatorial laplacians of simplicial complexes," Tech. Rep., Bard University, 2002.

[86] A. Zomorodian, "Fast construction of the vietoris-rips complex," in *Computers and Graphics, Shape Modeling International, Aix-en-Provence, France*, 2010.

[87] A. Lee, K. Pedersen, and D. Mumford, "The nonlinear statistics of high-contrast patches in

natural images," *International Journal of Computer Vision*, vol. 54, pp. 83–103, 2003.

[88] A. Tahbaz-Salehi and A. Jadbabaie, "Distributed coverage verification algorithms in sensor networks without location information," *IEEE Transactions on Automatic Control*, vol. 55, 2010.

[89] A. Schrijver, *Theory of linear and integer programming*, Wiley-Interscience series in discrete mathematics and optimization, 1986.

[90] A. Frome, Y. Singer, F. Sha, and J. Malik, "Learning globally-consistent local distance functions for shape-based image retrieval and classification," in *ICCV*, 2007.

[91] D. Ramanan and S. Baker, "Local distance functions: A taxonomy, new algorithms, and an evaluation," *IEEE Pattern Analysis and Machine Intelligence (PAMI)*, vol. 33, no. 4, pp. 794–806, April 2011.

[92] R. Shepard and S. Chipman, "Second order isomorphism of internal representations: Shapes of states," *Cognitive Psychology*, vol. 1, no. 1, pp. 1–17, 1970.

[93] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *CVPR*, 2005.

[94] E. Rosch, *Cognition and Categorization*, chapter Principles of Categorization, pp. 27–48, Erlbaum, Hillsdale, NJ, 1978.

[95] T. Malisiewicz, A. Gupta, and A. Efros, "Ensemble of exemplar-svms for object detection and beyond," in *ICCV*, 2011.

[96] T. Malisiewicz and A. Efros, "Beyond categories: The visual memex model for reasoning about object relationships," in *NIPS*, 2009.

[97] R. Brooks, "Intelligence without representation," *Artificial Intelligence*, vol. 47, pp. 139–159, 1991.

[98] D. Weinberger, *Everything is Miscellaneous: The Power of the new digital disorder*, Times Books Henry Holt and Company, 2007.