Contour Grouping and Abstraction Using Simple Part Models

Pablo Sala and Sven Dickinson

Department of Computer Science, University of Toronto, Toronto ON, Canada

Abstract. We address the problem of contour-based perceptual grouping using a user-defined vocabulary of simple part models. We train a family of classifiers on the vocabulary, and apply them to a region oversegmentation of the input image to detect closed contours that are consistent with some shape in the vocabulary. Given such a set of consistent cycles, they are both abstracted and categorized through a novel application of an active shape model also trained on the vocabulary. From an image of a real object, our framework recovers the projections of the abstract surfaces that comprise an idealized model of the object. We evaluate our framework on a newly constructed dataset annotated with a set of ground truth abstract surfaces.

Keywords: perceptual grouping, shape abstraction, part vocabulary.

1 Introduction

The problem of computational perceptual grouping received considerable attention before the advent of appearance-based recognition, when object models were typically shape-based and image features were typically contour-based. Moreover, while object databases were rather small, it was generally assumed that a linear search of a database, i.e., matching the image features against each model in succession and choosing the best-matching model, was an unacceptable strategy, for it did not scale to very large databases. In an effort to achieve sublinear scaling, much effort was devoted to the problem of object *indexing*, i.e., using a set of image features to query the database for candidate objects that might account for the image features. An effective query structure, or index, should be small enough to be reliably extracted, yet discriminative enough to aggressively prune the database down to a few promising candidates. Since image features were contour-based, perceptual grouping played a major role in grouping together contours that were unlikely to co-occur by chance. Moreover, grouping was based *not* on object-level prior knowledge, but rather on mid-level (object-independent) prior knowledge. Such grouping was essential, since local contour features were highly ambiguous, and without grouping them into more discriminative structures, effective indexing into large databases was problematic.

The object categorization community's focus on the object *detection* problem has since drawn attention away from perceptual grouping, since there is no need to construct an effective index when the candidate (target) object is known. However, there are signs that the categorization community is not only returning to the more categorical feature of object shape, but to the more general problem of recognition from a large

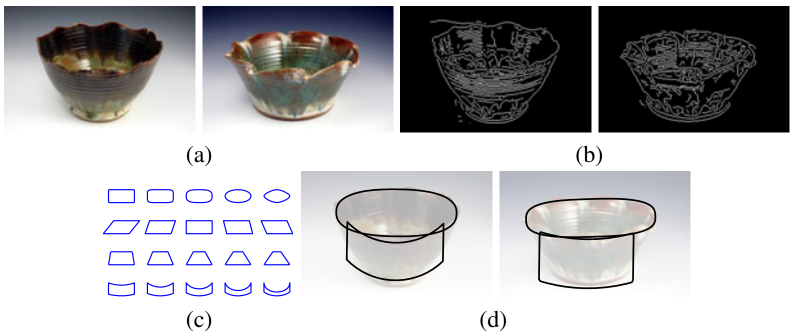


Fig. 1. Recovering abstract shape parts from an image: (a) input image of two exemplars that show considerable within-class variation; (b) extracted contours: note that corresponding contour-based features are seldom in one-to-one correspondence; (c) a simple example vocabulary of 2-D part models that will drive the perceptual grouping and shape abstraction processes; (d) the resulting abstract surfaces recovered by our framework; contour correspondence exists not at the level of individual contours, but at a much higher level of abstraction.

database. In turn, the need to group together contour features into powerful indexing structures may stimulate interest in perceptual grouping [1–3]. However, whereas simple groups of contour features may have been sufficient for indexing into a database of shape exemplars, today's interest in categorization will require not only the grouping of causally related contour features, but their *perceptual abstraction* to yield higher-order shape features that are invariant to within-class variation. This raises two important challenges: 1) how to perceptually group related contours; and 2) how to perceptually abstract the groups into high-order shape features that are more generic and lessspecific.

In this paper, we present a novel approach to the perceptual grouping and abstraction of image contours using a set of 2-D part models. We assume no object-level prior knowledge and, like the perceptual grouping community, assume only a mid-level shape prior. However, our shape prior is slightly stronger than such classical Gestalt features as symmetry, parallelism, collinearity, etc. Specifically, our mid-level shape prior takes the form of a user-defined vocabulary of simple 2-D shape models, representing a fixed set of parts from which a large database of object models can be constructed. In that sense, our vocabulary can be seen as a high-level nonaccidental regularity – a common denominator set of part shapes that can be used to model a large collection of objects in the world. Since different domains may demand different vocabularies of parts, it's essential that our framework be *independent* of the part vocabulary; therefore, the vocabulary is an input to our framework.

Figure 1(a) shows images of two object exemplars that belong to the same class (bowl), while Figure 1(b) shows their extracted contours; note that corresponding contour-based features are seldom in one-to-one correspondence. In Figure 1(c), we show sample instances from a simple vocabulary of 2-D shapes that will be used to group and abstract the contours in Figure 1(b). In Figure 1(d), we overlay the abstract shapes recovered by our algorithm. If we examine carefully the region boundaries in both images, we observe that due to within-class variation or noise, there are few corresponding contours between the two parts. As noise and within-class variation increase, methods that rely on one-to-one feature correspondences among specific contour-based features may fail. Only by examining the abstract shapes defined by these contours does commonality between the two exemplars emerge.

2 Related Work

There is a large body of work on using simple shape models to group and regularize 2-D contour data. Due to space constraints, we will not review approaches that take, as input, a silhouette, i.e., assume figure-ground segmentation, nor will we review approaches that assume knowledge of what object is in the scene, i.e., object-level shape priors. Rather, we adopt the classical perceptual grouping position and review related approaches that assume only mid-level shape priors. Such priors can range from simple smoothness to compactness to convexity to symmetry to more elaborate part models, but stop short of object models.

Jacobs [4] and Estrada and Jepson [5] explored the nonaccidental regularity of convexity to group contours into convex parts. Other researchers, e.g., Stahl and Wang [3], have explored the nonaccidental regularity of symmetry to group contours into symmetric parts, while Lindeberg [6], has explored symmetry to extract symmetric blobs and ridges directly from image data. While each of these models exploits a particular nonaccidental shape regularity, they also restrict the image domain. Moreover, each mid-level shape prior comes with its own computational model, and there is little to unify the approaches.

More powerful part models stemmed from the early recognition by parts paradigm. Pentland [7] partitioned a binary image into 2-D parts corresponding to the projections of a vocabulary of 3-D deformable superquadrics. The method focused more on the problem of part selection (from a large space of part hypotheses) than the grouping of features into parts, and the framework was never applied to contours. Pilu and Fisher [8] attempted to recover 2-D deformable parts models from image contours. However, they assumed that the correspondence between image and model contours was one-to-one, restricting the scenes to contain very simple objects. Little abstraction was achieved, and such systems were rarely applied to complex scenes.

The dual problem to fitting part models to contours is fitting part models to regions. Liu and Sclaroff ([9]) proposed a method capable of finding instances of a 2-D shape (possibly a part model) in an image. From a bottom-up region segmentation, the space of region merges and splits is explored in search of region groups with shapes similar to a 2-D statistical template model. Wang et al. [2] proposed a stochastic approach to explore the space of region merges and splits in search of region groups having a particular shape. From a bottom-up image region segmentation, their approach was capable of finding multiple occluded instances of a model shape by grouping oversegmented regions. However, these approaches typically admit a single model shape and also rely strongly on appearance homogeneity to guide the grouping process. Moreover, Wang et al.'s method employed a detailed model of the shape, and did not attempt shape abstraction. In [10], we introduced a model-based framework to detect abstract part hypotheses from a multiscale edge map. However, it was not only computationally expensive (since

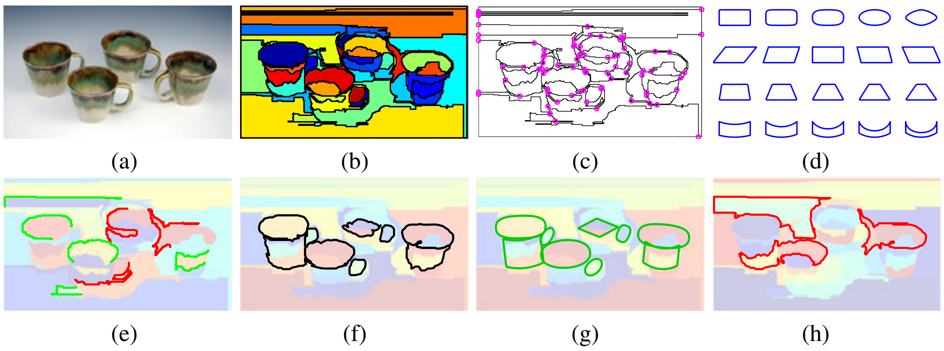


Fig. 2. Problem Formulation: (a) input image; (b) region oversegmentation; (c) region boundary graph; (d) example vocabulary of shape models (used in our experiments); (e) example paths through the region boundary graph that are consistent (green) and inconsistent (red); (f) example detected cycles that are consistent with some model in the vocabulary; (g) abstractions of cycles consistent with some model; (h) example cycles inconsistent with all models.

it exhaustively explored all possible transformations of every model), it also generated a large number of hypotheses (several hundred), thus yielding poor precision.

In summary, a diverse set of mid-level shape priors have been proposed, each with its own strengths and weaknesses. Unfortunately, most approaches are closely tied to their underlying shape priors, and the mechanism for recovering one class of parts may vary greatly from the mechanism for recovering the next class. Moreover, most part recovery schemes are rather brittle and offer little opportunity for recovering abstract parts from the noisy, irregular contours that often make up real objects. We address these two shortcomings head on with a part-based grouping framework that's independent of the parts, and a shape abstraction mechanism that can recover the abstract parts that make up a large collection of real objects.

3 Overview of the Approach

Our approach begins by computing a region oversegmentation (Figure 2(b)) of the input image (Figure 2(a)). The resulting region boundaries yield a *region boundary graph* (Figure 2(c)), in which nodes represent region boundary junctions where three or more regions meet, and edges represent the region boundaries between nodes; the region boundary graph is a multigraph, since there may be multiple edges between two nodes. Our approach can be formulated as finding simple cycles in the region boundary graph whose shape is consistent with one of the model shapes in the input vocabulary (Figure 2(d)); these are called *consistent cycles*. There is an exponential number of simple cycles in a planar graph [11], and simply enumerating all cycles (e.g., [12]) and comparing their shapes to the model shapes is intractable. Instead, we start from an initial set of starting edges and extend these paths, called *consistent paths* (or CPs), as long as their shapes are consistent with a part of *some* model. To determine whether a given path is consistent (and therefore extendable), we approximate the path at multiple scales with a set of polylines (piecewise linear approximations, providing a set of low-dimensional boundary abstractions), and classify each polyline using a one-class classifier trained on the set of training shapes (Figure 2(e)). When a consistent path is also a simple cycle, it is added to the set of output consistent cycles (Figure 2(f)).

Figure 2(d) shows the input vocabulary used in our experiments: four part classes (superellipses plus sheared, tapered, and bent rectangles, representing the rows) along with a few examples of their many within-class deformations (representing the columns). It is important to note that our approach is independent of the vocabulary of parts. While our demonstration vocabulary is ideally suited to the projected surfaces of ellipsoids, and straight and bent rectangular blocks and cylinders, representing a simple volumetric part vocabulary, the approach can accommodate any set of shapes, parameterized or otherwise. Each shape model is allowed to anisotropically scale in the x- and y- directions as well as rotate in the image plane. Since we employ scale-, rotation-, and translation-invariant features to train the classifiers, we need to generate only (approx.) 1,500 instances (by varying the aspect ratio and deformation parameters) belonging to these four shape classes. A *single* classifier is trained on all the component polylines of length (i.e., number of piecewise linear segments) k spanning the *complete* set of shape models and their deformations. Therefore, if K is the upper bound on the length of a polyline approximating a shape in the vocabulary, then K classifiers are trained. An ideal vocabulary defines a small set of "building blocks" common to a large database of objects. As such, the complexity of the vocabulary shapes is low, and even at the finest scale of polyline partitioning of a vocabulary shape's contour, K remains low; for our vocabulary, K is 13.

The algorithm outputs cycles of contours that are consistent with one of the model (training) shapes. A cycle consists of actual contours (edges in the region boundary graph) in the image, and therefore does not explicitly capture the abstract shape of the contours. Moreover, the cycle has not yet been categorized according to the shapes in the vocabulary. To abstract (or regularize) the shape of a cycle and to categorize it, we follow a standard, iterative 2-step active shape model (ASM) fitting framework [13] trained on about 600,000 model instances, generated by varying their aspect ratio, orientation, and a finer sweeping of the deformation parameters than the one used to train the polyline classifiers. We iterate over the classical two-step ASM procedure, consecutively aligning and deforming the mean training shape with the cycle until convergence. However, we depart from a standard ASM framework in two key ways.

In a standard ASM framework, the training shapes belong to a single shape class, and the allowable, often limited, deformations are typically captured (using PCA) in a low-dimensional shape space that can be approximated by a multidimensional Gaussian distribution. Moreover, at run time, the model must be properly initialized, for if the model is grossly misaligned, the deformations required to warp the model into the image may fall outside the space of allowable deformations. In our case, given a consistent cycle, we don't know which category of vocabulary shape it belongs to, and hence which ASM model to apply (if we assumed one model per category in the vocabulary). Moreover, even if we knew its category, we assume no correct or near-correct initial landmark correspondence. We overcome the first problem by having a single ASM that's trained on all instances of all the shapes in the vocabulary, and overcome the second problem by training on all possible landmark correspondences (alignments) across these shapes.

After ASM convergence, the training shape closest to the deformed model identifies the category of the cycle. In the previous step, the consistent cycle classifier's precision rate is never 100% at reasonable recall rates, and some of the recovered consistent cycles (of contours) may yield shapes that are qualitatively different from those in the vocabulary. Therefore, following ASM convergence, shapes that are still significantly different from the training shapes are discarded. Figure 2(g) illustrates the vocabulary shapes abstracted from the consistent cycles in Figure 2(f); for each detected shape, the algorithm also yields its shape category. Finally, Figure 2(h) illustrates some of the false positives discarded by the shape abstraction process.

4 Finding Consistent Cycles

In the following subsections, we elaborate on the steps of our algorithm for finding consistent cycles, i.e., cycles whose shape is consistent with one of the model shapes; Section 5 will focus on the problem of abstracting/categorizing the shape of the cycle.

4.1 Path Initialization

The goal of path initialization is to identify a minimum cardinality set of edges such that every cycle in the graph contains at least one of the edges. This can be easily achieved by computing the *feedback edge set*, i.e., the smallest set of edges whose deletion results in an acyclic graph. The feedback edge set is computed as the edge complement of a spanning tree. Favoring edges that represent longer, and thus more informative contours, we will choose as our initial edge set the edge complement of the minimum spanning tree, where edge weight equals contour length. While every cycle contains at least one of these initial edges, two or more of these edges may be part of the same cycle. This is problematic, since extending these initial paths will yield the same cycles, which is highly inefficient. We avoid this problem by imposing a total ordering on the edges, and allowing a path to be extended only by an edge whose rank is greater than that of the initial edge of that path; the edges in the minimum spanning tree are all assigned a rank of ∞. The rank of a path is the rank of its initial edge. The set of initial edges, and their ranks, form our initial set of paths, and they are added to the queue of paths to be extended.

4.2 Path Extension

At each iteration of the algorithm, one of the paths is taken off the queue. If the path is a cycle, its consistency with the vocabulary of model shapes is checked. If it's consistent with at least one shape in the vocabulary, it is added to the output list of consistent cycles. If, however, the path is not a cycle, its consistency is also checked. If the path is consistent with a portion of the boundary of at least one shape in the vocabulary, then the path's possible extensions by an edge whose rank is greater than the path are added to the queue. The algorithm continues until the queue is empty, and outputs the consistent cycles.

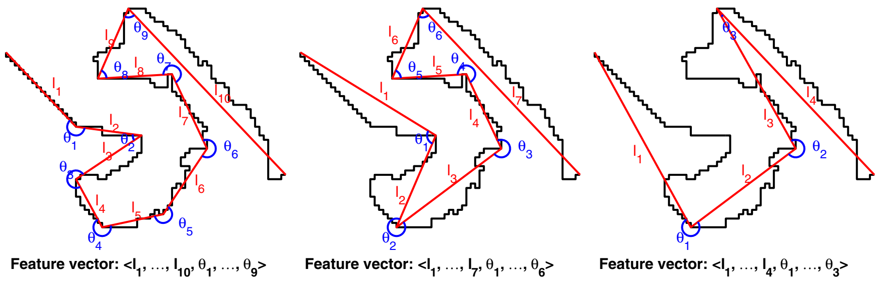

Feature vector: <l1, ..., l10, θ1, ..., θ9>
Feature vector: <l1, ..., l7, θ1, ..., θ6>
Feature vector: <l1, ..., l4, θ1, ..., θ3>

Fig. 3. Feature vectors for a multiscale polyline approximation of a contour

Checking consistency begins by approximating the shape of the cycle or path with a polyline computed at different scales using the Ramer-Douglas-Peucker algorithm [14]. For each resulting polyline, we compute a feature vector that encodes the angles and normalized lengths of the linear segments making up the polyline. As illustrated in Figure 3, the length of the feature vector is a function of the number of linear segments comprising the polyline. Each feature vector is passed to a one-class classifier (there is a classifier for each feature vector length) that determines if the feature vector is geometrically close to one of the training feature vectors. Those scales at which their corresponding polylines are consistent are associated with the path. If a path at a particular scale is not consistent, then no extension of that path can be consistent at that scale. Thus, when a path is initialized, it is associated with all scales, and when it is extended, its associated scales can only remain constant or decrease. If there is no scale at which the path is consistent, the path is discarded, and will not be extended further.

4.3 Training the Classifiers

The feature vectors used to train the classifiers are generated from contour fragments of model instances. Axis-aligned instances of within-class deformations of each model are generated at varying aspect-ratios, and Gaussian noise is added to each generated contour with a standard deviation proportional to the model size (defined as the average distance from a model contour point to the model's centroid). A number of equidistant points along each generated contour are sampled, and the two subcontours between every pair of such points (traversed in both directions) are used as a training example. A feature vector is generated for each subcontour from its polyline approximation, computed using a tolerance that is proportional to model size. Finally, the dimensionality of the feature vectors is reduced using PCA. Classification is performed on these reduced dimensionality vectors. For our model vocabulary, we observe that at least 99% of the variance is, in general, captured by the top N PCA components for the case of feature vectors of dimension $2N-1$, corresponding to polylines with N linear segments.

In our implementation, the number of linear segments comprising the longest polyline approximating a model's contour is bounded by 13. For this reason, at consistency check time, a path whose polyline approximation is longer than this value is discarded as inconsistent. Moreover, in the case of the path being a cycle, if its polyline approximation at a certain scale has less than three linear segments, the cycle is deemed inconsistent at that scale, since the scale is obviously too coarse. In the case of an open path with a polyline approximation with less than two segments, there is not enough information to decide on the path's consistency, since the path is evidently still too short at the given scale. In this case, the path is treated in the algorithm as if it had been consistent, leaving the decision to a future iteration, after it has been eventually extended to a length where a consistency decision is possible.

For our experiments, approximately 4 million contour fragments were employed to train the classifiers. Due to the difficulty in generating an adequate set of training examples of inconsistent model contour fragments, a one-class classifier was used instead of a binary classifier. Since the consistency check needs to be performed a large number of times (once per path extension), an efficient implementation calls for a method with a low classification complexity for this task. We obtain good classification performance and very fast classification rates using a Nearest Neighbor Data Description approach [15] implemented via a fast approximate nearest neighbor search data structure [16].

5 Abstracting the Shape of a Consistent Cycle

As mentioned in Section 3, we employ an ASM to both abstract the shape of a consistent cycle and to categorize it. Recall that we train a single ASM on all deformations of all vocabulary shape classes over all possible landmark correspondences. This avoids a proliferation of ASM models (one model, regardless of the size of the vocabulary), and allows the model to be initialized anywhere on the cycle. To train the Point Distribution Model, we sweep the parameter space of the shapes in the model vocabulary. Specifically, we generate contours for all models and degrees of deformation (i.e., bending, tapering, and shearing) at a dense set of discrete aspect ratios and all possible cyclical landmark alignments; in our implementation, we generate a total of approximately 600,000 such training cases. Rotation invariance is achieved by training on all model landmark alignments corresponding to all possible N cyclical rotations of the landmarks, while scale invariance is achieved during ASM fitting by the rigid transformation estimation.

The list of landmark points $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_N \in \mathbb{R}^2$ in a training example corresponds to a fixed number of equidistant points sampled along the contour ($N = 64$ in our experiments). The first landmark $\boldsymbol{m}_1$ is the one for which the vector from the centroid $\overline{\boldsymbol{m}}$ to the landmark has the lowest (absolute value) angle with respect to the x-axis, i.e., $(1, 0)^T \cdot (\boldsymbol{m}_i - \overline{\boldsymbol{m}})\ /\ ||\boldsymbol{m}_i - \overline{\boldsymbol{m}}||$ is maximum for $i = 1$; in case of a tie, we choose the point with maximum $||\boldsymbol{m}_i - \overline{\boldsymbol{m}}||$. The indices of the other landmarks in the list keep their natural order along the contour. Following the standard ASM approach, a vector is formed for each contour by rasterizing the contour's landmark coordinates. A PCA basis is computed for the training set, and the lowest-order principal components that capture most of the variance are chosen. For our training set, 99.9% of the total variance is captured by the top 21 components.

Fitting is performed by the successive iteration of two steps: one that finds the rigid transformation that best aligns the current deformed model to the query contour, and a second step that adjusts the shape parameters that deform the model to better improve the fit. This adjustment is constrained such that the deformed model shape is consistent with that of the training examples. Enforcing this constraint is accomplished by checking that the adjusted shape parameters do not fall outside of the distribution of the training shapes; if the parameters do fall outside, they are set according to the closest shape in the distribution.

We differ from a standard ASM in how a shape adjustment is constrained to lie in the space of training shapes. In a classical ASM framework, where training shapes belong to the same class and exhibit relatively minor deformations, the low-dimensional subspace where the shape points live is approximately Gaussian distributed. Under this condition, enforcing this constraint amounts to simply verifying that the adjusted shape parameters do not deviate more than a certain number of standard deviations (e.g., 3) from the mean shape; beyond that, the point is scaled down to correspond to the closest point within the distribution. In our case, the set of training shapes is quite heterogeneous (spanning multiple categories and all possible initial correspondences), yielding a complex shape space boundary whose enclosed distribution is not well approximated by a Gaussian. We therefore need a different way to constrain a given adjusted shape to fall within the shape space spanned by the training set. Since our training set densely covers the space of shapes of interest and a low-dimensional subspace captures most of the shape variance, a Nearest Neighbor Data Description method [15] provides a fast mechanism for checking if a query shape belongs to the target distribution. If an adjusted shape does not belong to the distribution, it is constrained to be in the distribution by replacing it by a near neighbor in the distribution. To avoid falling in local minima, we randomly choose the replacement from among the $k(t)$ nearest neighbors, where $k(t) \in \mathbb{N}_{>0}$ is a non-increasing function of the number of iterations. In our experiments, we obtained good results by using a linear function for $k(t)$.

Since we attempt to bridge the gap between image contours and ideal model contours, a simple distance field between image and model contour landmarks is inappropriate as a driving force to guide the model deformation process. Such an approach would give the same weight to all contour landmark correspondences and may fail to deform the model appropriately in the case of minor region undersegmentation or minor contour shape deviation from the model. In order to cope with these conditions, we define the deformation force for each landmark i as:

$$\boldsymbol{\delta_i} = (1 - \alpha(t))(\boldsymbol{q}_i - \boldsymbol{m}_i) + \alpha(t)(\mathsf{closest}_{\boldsymbol{q}}(\boldsymbol{m}_i) - \boldsymbol{m}_i), \tag{1}$$

where $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_N \in \mathbb{R}^2$ are image contour landmarks sampled at equidistant positions along the contour, $\mathsf{closest}_{\boldsymbol{q}}(\boldsymbol{m}_i)$ is the point along the image contour $\boldsymbol{q}$ that is closest to model landmark $\boldsymbol{m}_i$, and $\alpha(t) \in [0, 1]$ is a bijective monotonically increasing function of the iteration number t. In this way, at the beginning of the fitting process, the attraction forces between the image and model contours are globally driven purely by landmark correspondences. This roughly aligns the model to the cycle. As the iterations proceed, the model deformation becomes increasingly driven by local contour attraction forces, giving more weight to the consensus of the image contours that are closest to the model, and thus letting the deformation process overlook significant image contour departure from the abstract model as well as some undersegmentation. Note that our fitting problem is more constrained than a standard ASM framework, since all landmark correspondences between a consistent cycle and the model ASM are known, and the influence of outlier landmarks on a consistent cycle (and their resulting incorrect correspondences) is decreased over time (iterations). Thus, the ASM is initially fit to an *entire* closed contour (not just a portion), but it converges to fit the relevant portion.

Finally, it is is possible that inconsistent cycles are misclassified as consistent. After convergence, if the distance between the cycle and the model exceeds a threshold, or the cycle coverage by the model (i.e., proportion of cycle contour covered by model landmarks) is poor, the cycle is discarded as a false positive. We compute the scale-independent distance

$$d(\boldsymbol{q}, \boldsymbol{m}) = \frac{1}{N} \sum_{i=1}^{N} \frac{\|\boldsymbol{\delta}_i\|}{\sqrt{\frac{1}{N}\sum_{i=1}^{N} \|\boldsymbol{q_i} - \bar{\boldsymbol{q}}\|^2}} \tag{2}$$

between the fitted model $\boldsymbol{m}$ and the cycle $\boldsymbol{q}$, where the denominator is a normalizing factor corresponding to the geometric mean distance between cycle landmarks and their centroid, thus making $d(\boldsymbol{q}, \boldsymbol{m})$ a scale-independent measure with an intuitive geometric interpretation; in our experiments, we obtained good results using a threshold of 0.15.

6 Results

Unfortunately, we know of no benchmark dataset for evaluating part-based shape abstraction, nor are we aware of competing approaches for part-based shape abstraction using a vocabulary of simple part models, except for [10]. Therefore, in order to evaluate our framework, we created an annotated dataset with 67 images[1] containing object exemplars whose 3-D shape can be qualitatively described by cylinders and bent or tapered cubic prisms. The abstract visible surfaces of each 3-D shape were hand-labeled using 2-D models drawn from our vocabulary. Figure 4 illustrates the output of our system on a number of examples in the dataset: column (a) shows the input image; column (b) shows the region oversegmentation used as input to our algorithm, computed using the local variation approach by Felzenszwalb and Huttenlocher [17] with a fixed parameterization on all images; column (c) shows the consistent cycles from which the shapes in column (d) were abstracted, representing the recovered parts closest to the ground truth in column (e). The numbers inside recovered abstract parts in column (d) indicate the rank of the part among all recovered parts in that image, computed as a function of the distance to the contours of the cycles that they are abstracting. The target regions can sometimes rank low if their degree of abstraction is high compared to non-target regions in the image (whether real or segmentation artifacts) that require less abstraction. Note that in some cases, e.g., the blender body in row 8, the ideal ground truth part (e.g., corresponding to the projection of the body of a tapered cylinder) did not exist in the vocabulary.

Our ability to abstract the shape of a cycle of contours with high local irregularity (shape "noise") means that many false positive parts will be recovered. As a result, the ranks of some of the ground truth shapes among the hypotheses is poor. This is entirely due to the naive scoring mechanism (absolute fitting error) that tends to favor small,

[1] Available at `http://www.cs.toronto.edu/~psala/datasets.html`

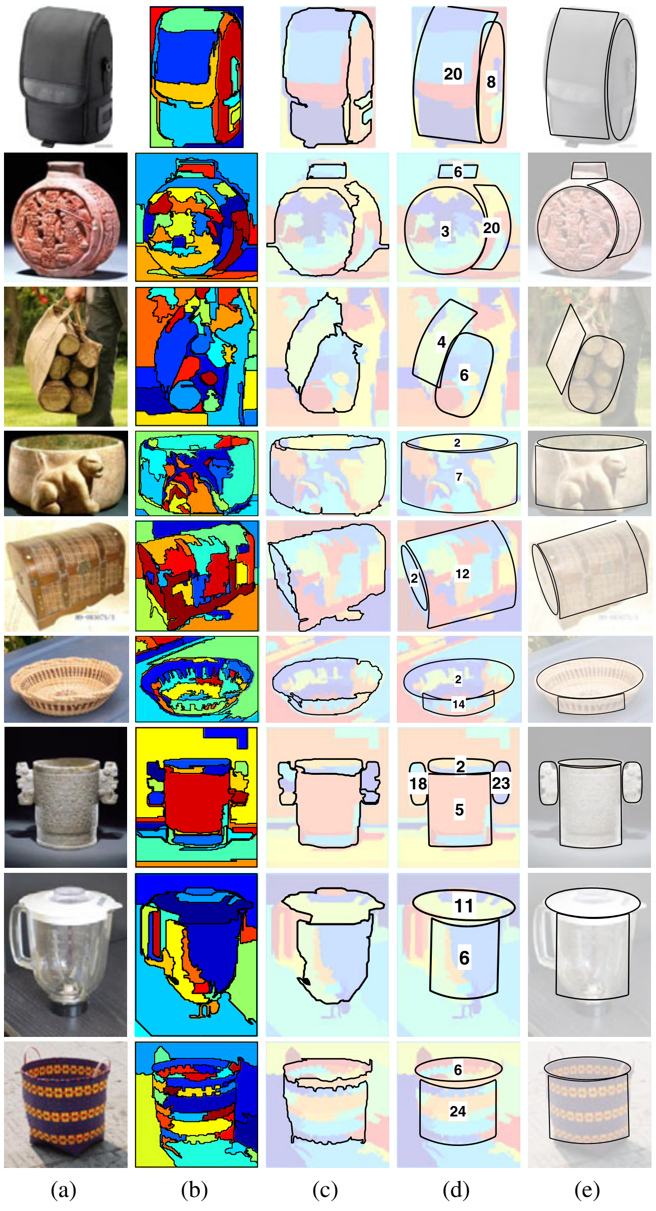
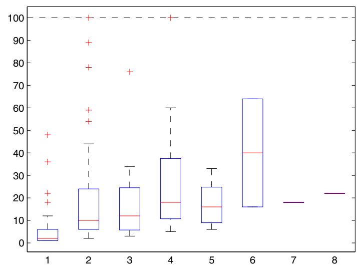
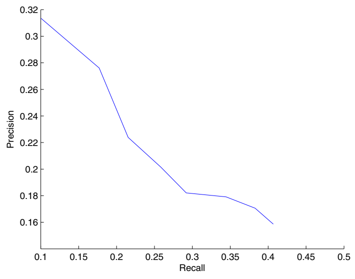


Fig. 4. Abstract Part Recovery (see text for discussion)

well-fitting shapes over larger abstractions. While more inspired scoring functions may increase the ranks of the target shapes, we mean only to illustrate the significant extent to which the target shapes are indeed generated. It is at a later stage, when contextual constraints are added, where we expect an aggressive pruning of false positives.



(a)



(b)

Fig. 5. Quantitative Evaluation: (a) Distribution of Rankings of Ground Truth Parts Among Generated Abstract Shapes. For example, the first detected ground truth part (leftmost column) in an image ranked 2nd (median rank - red bar) among the ranked hypothesis in the image. The top and bottom of the blue box defines the upper and lower quartiles, the whiskers define the furthest datum within 1.5IQR of the lower and upper quartiles, and the red "+"'s the outliers. (b) Precision-recall curve (see text for discussion).

In future work, we plan to explore powerful contextual relations, including proximity, alignment, and 3-D shape information to prune many of these false positives. For example, if the surfaces in our images can indeed be the projections of volumetric parts, such as cylinders or prisms, then there are strong constraints on the shapes and relations of the component faces (parts) of their aspects. Other constraints are also possible, such as pruning smaller surfaces that are subsumed by larger surfaces. Adding these relational constraints is beyond the scope of this paper, and here we focus only on the initial recovery of the primitive parts. As can be seen from Figure 4, our framework is able to recover and abstract many of the surfaces of the objects.

To provide a quantitative evaluation of our framework's ability to recover the correct abstractions amid the abstract shapes (hypotheses) recovered from an image, we analyze the rank of a ground truth shape among the ranked hypotheses. Figure 5(a) shows the distribution of ground truth part rankings. For example, the first detected ground truth part (leftmost column) in an image ranked 2nd (median rank - red bar) among the ranked hypothesis in the image. The second detected ground truth part (second column) ranked 10th, and so forth. The number of ground truth parts, on average, was 3, while the number of part hypotheses generated for an image, on average, was 71. No attempt was made to eliminate redundant models (i.e., models of the same category with roughly the same parameterization), and no size filtering was performed. From these results, we conclude that the target (ground truth) shapes reside in a manageable number of hypotheses, and we expect that with the application of contextual constraints, the false positive shapes can be drastically reduced. Figure 5(b) illustrates precision and recall for our database. While precision is low due to the high number of false positives (due to lack of contextual pruning and non-maximum suppression), our recall of ground truth shapes is reasonably good, typically failing in the presence of significant region undersegmentation. In terms of running time, a typical run of the consistent cycle detection algorithm requires an average of about 40,000 iterations, which takes about 3 seconds in our MATLAB/C++ implementation running on a laptop. The model

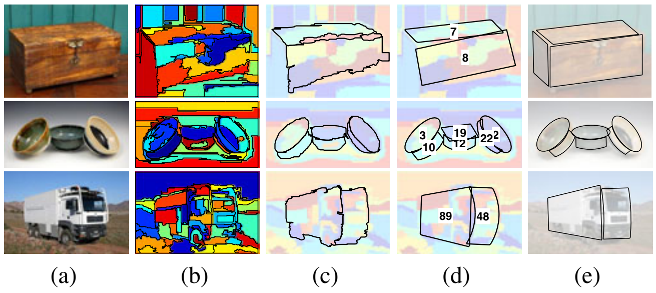


Fig. 6. Limitations of the Approach (see text for discussion)

abstraction algorithm was fully implemented in MATLAB and takes about 150 seconds to process all consistent cycles detected in an image.

Exploring the results in more detail, we see that Figure 4(d) shows the ability of our approach to abstract object surfaces that are locally highly irregular due to noise or within-class variation, but capture a model shape at a higher level of abstraction. In some cases (e.g., rows 5,6, and 8), we see misalignment with a neighboring shape. This can be due to two reasons: (1) the vocabulary may not contain the appropriate shape to model the surface; and (2) the shapes are recovered independently, with no alignment constraints exploited; such constraints, as well as other constraints, will play an aggressive role in pruning/aligning hypotheses in our future work. In all the examples, we can see that the model abstraction process is able to cope with region undersegmentation when it is restricted to a relatively small section of the contour. Figure 4 (rows 1 and 9) shows examples of cases in which, although some portions of the correct surface boundaries are missing, the models are still correctly fit due to the consensus of the correct surface contours.

Figure 6 illustrates some weaknesses of our approach. The top row shows a case in which an object's surface is missing (i.e., box's left face) due to strong undersegmentation of the input to our algorithm. Since the consistent cycle detection mechanism already keeps incremental hypotheses of partial contour matchings, in future work we plan to allow informative consistent paths to be abstracted (using a similar framework). This will not only accommodate region undersegmentation, but also region occlusion and partial part abstraction. In the second row of Figure 6, we see a case in which a consistent cycle was abstracted by a model of an incorrect category (i.e., the rim of the central bowl). This is because either the abstraction approach was trapped in a local minimum or there is an inherent shape ambiguity in the noisy contour. This can be remedied by allowing the abstraction process to return not just one model, but a list of candidate models that lie within a certain distance from the consistent cycle. We expect our future use of relational constraints to help overcome such ambiguity, and in this case "flip" the rectangle to an ellipse. Finally, the third row of Figure 6 shows a case where the ranking of the correct models is poor due to the presence of many uninteresting image region groups whose shape is consistent with vocabulary model shapes (i.e., there is a large number of region groups forming regular quadrilaterals). The use of context or non-maximum suppression can eliminate many of these false positives.

7 Conclusions

We have presented a framework for grouping contours (region boundaries) into parts according to a user-defined vocabulary of abstract parts models. Our contributions are threefold: (1) we train a classifier on all possible component fragments of a vocabulary of parts, and use the resulting set of classifiers to guide a grouping process that searches for cycles of locally irregular contours that are consistent, at some level of abstraction, with some model shape; (2) the consistent cycles are abstracted and categorized using a novel application of an ASM model which captures the entire vocabulary of shapes with a single model and which needs no proper initialization; (3) the resulting framework reports promising first steps toward part-based shape abstraction from images of real objects, and establishes a number of important directions for future work.

References

1. Zhu, Q., Song, G., Shi, J.: Untangling cycles for contour grouping. In: ICCV (2007)
2. Wang, J., Gu, E., Betke, M.: Mosaicshape: Stochastic region grouping with shape prior. In: CVPR (2005)
3. Stahl, J., Wang, S.: Globally optimal grouping for symmetric boundaries. In: CVPR (2006)
4. Jacobs, D.W.: Robust and efficient detection of salient convex groups. PAMI 18, 23–37 (1996)
5. Estrada, F., Jepson, A.: Perceptual grouping for contour extraction. In: ICPR (2002)
6. Lindeberg, T.: Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. IJCV 11, 283–318 (1993)
7. Pentland, A.P.: Automatic extraction of deformable part models. IJCV 4, 107–126 (1990)
8. Pilu, M., Fisher, R.: Model-driven grouping and recognition of generic object parts from single images. In: ISIRS, Lisbon, Portugal (1996)
9. Liu, L., Sclaroff, S.: Deformable model-guided region split and merge of image regions. IVC 22, 343–354 (2004)
10. Sala, P., Dickinson, S.: Model-based perceptual grouping and shape abstraction. In: POCV, Anchorage, Alaska, pp. 1–8 (2008)
11. Buchin, K., Knauer, C., Kriegel, K., Schulz, A., Seidel, R.: On the number of cycles in planar graphs. In: Lin, G. (ed.) COCOON 2007. LNCS, vol. 4598, pp. 97–107. Springer, Heidelberg (2007)
12. Tiernan, J.C.: An efficient search algorithm to find the elementary circuits of a graph. Commun. ACM 13, 722–726 (1970)
13. Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J.: Active shape models - their training and application. CVIU 61, 38–59 (1995)
14. Douglas, D., Peucker, T.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. CC 10, 112–122 (1973)
15. Tax, D., Duin, R.: Data description in subspaces. In: ICPR, vol. 2, pp. 672–675 (2000)
16. Mount, D.M., Arya, S.: ANN: A library for approximate nearest neighbor searching (2006), `http://www.cs.umd.edu/~mount/ANN/`
17. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV 59, 167–181 (2004)