Multi-Stage Contour based Detection of Deformable Objects

Saiprasad Ravishankar, Arpit Jain, Anurag Mittal

Indian Institute of Technology Madras, sairavi45,arpitjain1@gmail.com, amittal@cse.iitm.ernet.in

Abstract. We present an efficient multi stage approach to detection of deformable objects in real, cluttered images given a single or few hand drawn examples as models. The method handles deformations of the object by first breaking the given model into segments at high curvature points. We allow bending at these points as it has been studied that deformation typically happens at high curvature points. The broken segments are then scaled, rotated, deformed and searched independently in the gradient image. Point maps are generated for each segment that represent the locations of the matches for that segment. We then group k points from the point maps of k adjacent segments using a cost function that takes into account local scale variations as well as inter-segment orientations. These matched groups yield plausible locations for the objects. In the fine matching stage, the entire object contour in the localized regions is built from the k-segment groups and given a comprehensive score in a method that uses dynamic programming. An evaluation of our algorithm on a standard dataset yielded results that are better than published work on the same dataset. At the same time, we also evaluate our algorithm on additional images with considerable object deformations to verify the robustness of our method.

1 Introduction

Object detection using edge information is an important problem in Computer Vision that has received considerable attention from many researchers. The popularity of such methods is due to the fact that edges encode the object shape and are fairly invariant to color and illumination changes.

In this paper, we present a computationally efficient and robust multi-step approach for localizing objects in real, cluttered scenes given a single or few object sketches. Simple hand drawn models input by the user are first obtained for different object classes. The model is broken into segments at points of high curvatures. In the coarse search stage, these segments are passed through a series of deformations such as rotation, scale and bend and searched over the gradient image. Point maps are obtained for each segment which represent the possible locations of the matches for that segment. We then connect k points from the point maps of k adjacent segments (forming a k-segment group) using a cost function that takes into account local scale variations as well as inter-segment orientations. Each matched k-segment group is used to vote for local object centroids.

Since a correct object is likely to be identified by several k-groups, maxima in centroid densities (centroid boosting) are used to obtain bounding boxes that represent likely locations of objects in the image. The contour through each matched k-segment group is built from the gradient image using bending energy and edge strength formulations. In the fine stage of matching, the contours of the whole object are matched in a sequence using dynamic programming and a score is given for the matches in order to differentiate between correct detections and false ones.

We work on the gradient image which is rich in edge information and hence, as opposed to many other methods proposed in the literature, our approach does not encounter the problems associated with the use of edge detectors such as Canny [1] or the berkeley edge detector [2] that try to reduce the clutter in images by using spatial gradient information but unfortunately also miss out on many important edges. Dynamic programming on this gradient image allows us to be efficient while searching for the object segments.

1.1 Related Work

Several approaches which use edge information for object detection have been proposed in the past. Early methods include the Hausdorff distance [3] and the Chamfer distance [4] measures between an edge-based object model and the edge image. Such methods do not allow for much change in the shape of the model, but may be used for matching different parts separately for deformable objects.

Ferrari et al. [5] have shown that objects can be detected accurately in images using simple model sketches. They build a contour segment network and find paths that resemble the model chains. The method relies on the berkeley edge detector [2] which is able to remove a lot of clutter from the edge map and detects mostly object boundaries. This method of using contour segments has been further used in [6] to group contour segments into groups of k straight segments (called kAS) which are then matched in the berkeley edge image to detect objects.

In further work [7], Ferrari et. al. learn the shape model automatically from a set of training images and use a combination of Hough-style voting with a non-rigid point matching algorithm (thin-plate splines) in order to localize the model in cluttered images. The thin-plate spline model allows for a global affine transformation of the shape while allowing some local deviations from the affine model for each straight contour segment (PAS).

Opelt et. al. [8] use similar ideas of combining boundary segments but learn discriminative combinations of boundary fragments (weak detectors) to form a strong Boundary-Fragment-Model (BFM) detector. Their segments encode information about position relative to the centroid and they use a hough-style voting technique to find the centroids of the objects. [9] also use probabilistic extension of the Generalized Hough Transform to determine the object's centroid during recognition. Shotton et. al. [10, 11] learn discriminative contour fragments but look at these segments in isolation rather than in groups. Their fragments, however, are much larger than those of others. Similarly, Wu and Nevatia [12]

use edgelet features in order to build a classifier for detection and segmentation of specific objects.

Wu et. al. [13] use Gabor wavelet elements at different locations and orientations as deformable templates to learn an active basis from a set of images. Each of these elements is allowed a small amount of variation in their location and orientation while matching.

While the above mentioned methods are effective for many kinds of shape deformations, all of these methods measure deviation of each segment with respect to the object centroid or a global shape and can thus handle only relatively small local deformations of the shape from an overall shape. In contrast, we develop an approach where we use cost functions that take into account the deformations/orientations of adjacent segments. Thus, we can handle larger deformations of the object while still maintaining the overall sense of the object shape.

Felzenszwalb and Schwartz [14] use shape trees that use a hierarchical structure such that every segment is divided at different levels in a tree structure. Noise is added to each node in the shape tree and this yields the set of possible deformations of an object. An efficient algorithm based on dynamic programming was used. While this is an interesting approach, it is not clear whether such shape trees capture all possible deformations of an object.

Basri et. al. [15] proposed interesting analytical functions which can be used to match deformable shapes but the results were shown only on segmented objects. We borrow many of the ideas for shape deformation from this paper, while applying them in a more general and difficult setting.

In the rest of the paper, sections 2 and 3 describe the course and fine stages of our object recognition algorithm while section 4 summarizes the results.

2 Coarse Match

2.1 Basic Segment Match

The first step in our approach involves breaking simple hand drawn models into segments. A single model is usually sufficient for most objects (eg. bottle, applelogo etc.) but if the object appears drastically different from some view points, a model can be obtained for each of those view points (eg. side and rear views of a car). The model is broken at points of high curvatures (sharp turns or bends) into segments of low curvature. The breaking of the model into low curvature segments is done to allow more bending deformations at the points of high curvature (similar to [15]). Figure 1 shows the segment breakup for a model.

We permit bending deformation at the points of high curvature by allowing the two low curvature segments on either side of the high curvature point to rotate with respect to each other. Each segment is allowed rotation in steps in the range $[-\delta, \delta]$ (we use a δ of 30 degrees) to account for bending deformation at the high curvature points. Segments are also scaled to sizes in a range: $\alpha, 2\alpha, 3\alpha, ..., p\alpha$ where p is the size of the image and α is a fraction. Model

segments which have small curvature typically deform the least. We allow for small bending deformations of these model segments by scaling them along the direction perpendicular to their curvature (Figure 1(a)).

Line-like segments are easier to match but give numerous matches whereas slightly curved segments impart some degree of discriminability in the matching stage. Each rotated, scaled and deformed model segment is independently matched in the gradient image by finding paths that satisfy a normalized edge score (N).

$$N = \sum_{i,j \in C} G(i,j)/\sigma_C \tag{1}$$

where C represents the path, G is the gradient magnitude image and σ_C is the variance of the gradient values in the bounding box of the path. The maxima of N for each model segment for all its allowed scales, rotations and deformations are obtained. These can be computed quite efficiently using a sliding window mechanism that is implemented using dynamic programming.

These maxima (we take the top 15%) are represented by the mid points of the particular matched paths. Thus, the basic match step gives a point map for every segment that indicates the strong presence of that segment at those locations. The margins for bending deformations (Figure 1(a)) and scaling of the model segments allow us to capture most of the candidate match locations. Object segments in the image which could still not be correctly matched at this stage due to their larger deformation are efficiently detected in our final detection stage.

2.2 k-Segment Grouping

In the basic segment match step, point maps indicating matched segment locations are generated for every model segment. The point maps encode the midpoints of these segment matches. We search for k-segment groups in the point maps to localize the objects accurately. k matched points corresponding to k adjacent segments are searched jointly and costs are enforced on these k-segment groups to obtain those that might belong to the object. The k-segment groups are obtained for all possible k-adjacent segment combinations of the model.

At high values of k, we get a higher order shape perspective but the method becomes vulnerable to occlusions while at lower k values, we get a large number of matches which increases the computational cost. Hence, we choose an intermediate value of $k = 3$.

All the model segments are numbered such that adjacent segments get consecutive numbering. Adjacent segments have local scale and orientation information and hence can be used efficiently in the search process. The model segment midpoints are chosen to form the model k-segment groups.

A particular k-segment group obtained from the image point maps (the k^{th} point is extracted from the point map of the k^{th} segment) is required to satisfy local scale and orientation constraints. The k points are first linked pairwise (1-2, 2-3,.....,$(k-1)$-k) using line segments. This produces a graph (G) with k nodes,

(a) (b) (c)

Fig. 1. Illustration of model segments extraction: (a) Bending deformation through anisotropic stretching (b) The Applelogo model from the ETHZ dataset [5] (c) The various segment end points are indicated as green dots

$k-1$ adjoining line segments and $k-2$ node angles. A corresponding graph is also obtained for the model. Figure 2 (a) illustrates this procedure for a model with $k = 3$.

Cost function: The scales of the line segments (obtained by joining adjacent pair of points) in the graph G with respect to the corresponding line segments in the model graph are obtained. The scale ratio $\sigma_{m,t}$ of a particular linking line segment is the ratio of the lengths of the line segment in the model (m) and the image (t): $\sigma_{m,t} = \frac{d_t}{d_m}$. The scale cannot change drastically from one line segment to the next adjoining line segment (i.e. local scale variations should be small). But far away segments can have dissimilar scale variations. The node angles (represented by α) also indicate the inter-segment structural information. A cost function for the k-segment groups involving both local scale changes and node angle changes is formulated as follows:

$$C(s,a) = w_s C_s(m,t,k) + w_a C_a(m,t,k) \tag{2}$$

where s represents scale and a the node angle. We empirically determined $w_s = 0.4$ and $w_a = 0.6$). Furthermore, we compute the cost of scale changes as

$$C_s(m,t,k) = \frac{1}{k-2}\sum_{i=1}^{k-2}(max(\frac{\sigma^i_{m,t}}{\sigma^{i+1}_{m,t}}, \frac{\sigma^{i+1}_{m,t}}{\sigma^i_{m,t}}) - 1) \tag{3}$$

where $\sigma^i_{m,t} = \frac{d_{i,i+1,t}}{d_{i,i+1,m}}$ is the scale ratio of the i^{th} segment. Cost of orientation changes is taken as:

$$C_a(m,t,k) = \frac{1}{k-2}\sum_{i=1}^{k-2}(1 - e^{-c(\frac{\Delta\alpha^i_{m,t}}{\pi})^2}) \tag{4}$$

where $\Delta\alpha^i_{m,t} = \alpha^i_m - \alpha^i_t$ is the change in the i^{th} node angle and this angle difference is normalized by π. c is a factor that controls the amount of bending allowed by penalizing the change in the node angle. The scale and angle scores are averaged over the group. We use a threshold $C <= 0.25$ in our implementation for considering the k-segment cluster as matched.

Centroid Boost: Each point of a matched k-segment group contributes to a local object centroid (Figure 2(d)). The centroid is obtained from a matched group point (p) as follows: The vector joining the corresponding model group point to the model center is scaled by the local scale calculated at p and is used to determine the centroid. Each point of a matched group contributes a candidate centroid.

Matched k-segment groups of the object in the image would provide a high centroid density about the object centroid. The centroids which are not contributed by object groups are usually scattered. This leads to centroid density boosting at correct object locations which is used to localize objects. A centroid density map (ρ) is generated for the image. Points of maxima in this map are obtained by a summation over a circular window W:

$$F = \sum_{i,j \in W} \rho(x - i, y - j) \tag{5}$$

where (x, y) is a point on the centroid density map and the summation over the circular window W allows for uncertainty about the exact centroid. The points in the map which maximize F are taken as possible object locations. Once the centroids are identified, bounding boxes are computed for each of them. For computing the scale of a bounding box with respect to the model bounding box, the average (μ) and standard deviation (σ) of the scales of the k-segment clusters which contribute to centroids in the window W (about the strong centroid) are considered. The scale of the box is taken as $\mu + 2\sigma$. The matched k-segment groups lying inside the bounding boxes are considered for the fine matching stage.

In our coarse stage, we also match single stepped segment groups (eg. $1, 3, 5$) along with the adjacent segment groups as single stepped segments impart a higher order shape perspective. This is also useful for certain classes of objects where some adjacent segments are more deformable (eg. variations in the shape of a bottle) than single stepped segments.

The principle of centroid density boosting can efficiently localize multiple object instances in an image. The coarse segment group match accounts for local bending deformations in the object but does not take care of rotations of the object. To account for object rotation, we rotate our model and perform the coarse match. Rotated objects will provide maximal centroid density at the corresponding model rotation. The coarse matching stage effectively reduces the search space for the object contour detection to a few bounding boxes thus making our method computationally efficient. The effect of clutter is also drastically reduced in the final object contour search.

3 Fine Matching and Contour Completion

The coarse matching stage localizes objects with bounding boxes. The object contours are obtained in these localized regions by searching for contours in the gradient image that connect the points of the matched k-segment groups.

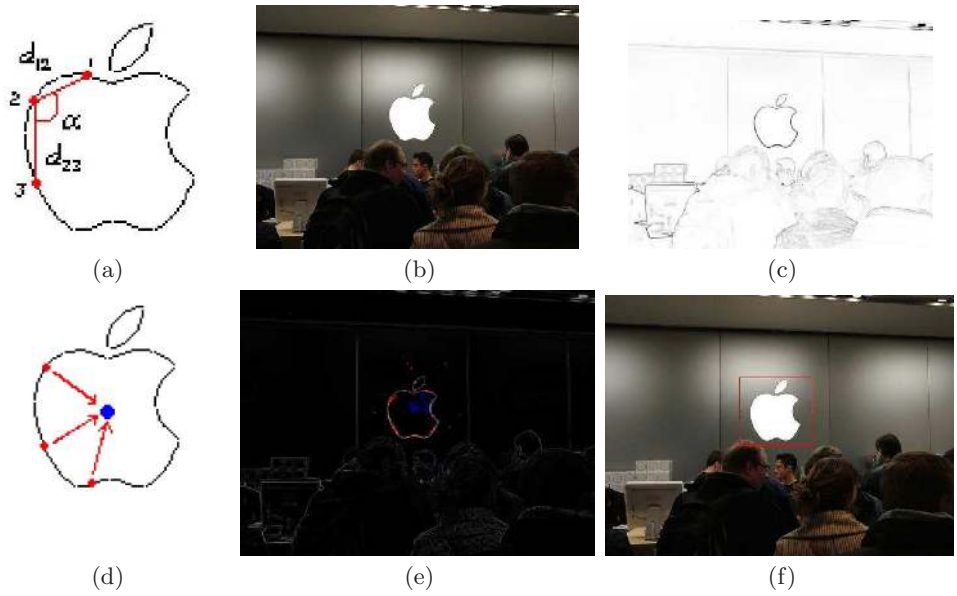


Fig. 2. Illustration of the steps of our object recognition algorithm. (a) Applelogo model with one 3-segment group (linking line segments and node angle α) shown (b) A test image from the database (c) The gradient magnitude image (d) Example of Centroid boosting (e) Matched k-segment groups in red with the centroids in blue (f) Object localization with bounding box.

Contour Search in Oriented Boxes: The object contour is built starting with adjacent pairs of points in the matched k-segment groups (i.e. 1-2, 2-3,....,$(k-1)$-k) and completing the contour between them. An oriented box is placed between every point pair (Figure 3). The orientation of the box is given by the orientation of the line segment in the model that joins the two points. The width of the box depends on the local scale (obtained from the coarse match) and the amount of bending of the contour that can be tolerated.

The contour of a k-segment group is built progressively from point 1 to point k using oriented boxes which greatly simplifies the computational complexity of obtaining matched contours. Paths are taken between the two points in each oriented box based on continuity of gradient magnitudes and directions. A cost is

formulated to obtain the best matching path that takes into account the bending energy and edge strength of the path.

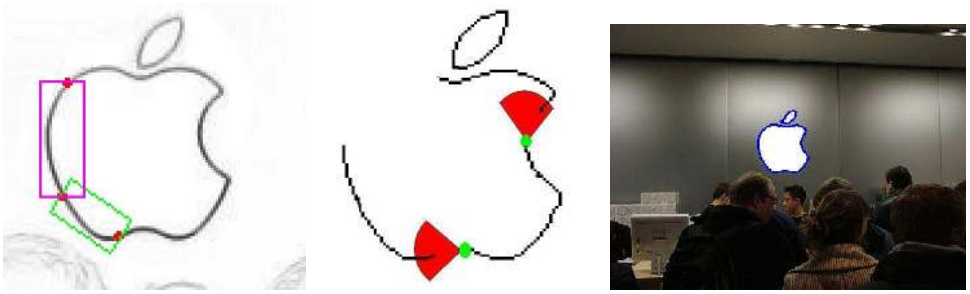
The cost function for a contour path is as follows:

$$q = w_b C_b + w_g C_g \tag{6}$$

where C_b and C_g are the costs for the bending energy and edge strength respectively (we empirically determined $w_b = 0.7$ and $w_g = 0.3$). The bending energy cost depends on the angles between consecutive tangents along the path. A vector (ϕ_c) of these angles is obtained for the contour path in the image. A similar vector (ϕ_m) of the angles between consecutive tangents along the corresponding segment of the model is also obtained. The larger of the two vectors is downsampled and the bending energy cost is computed as follows:

$$C_b = \frac{1}{L}\sum_{i=1}^{L}(1 - e^{-}(\frac{\phi_c^i - \phi_m^i}{\pi})^2) \tag{7}$$

where L is the minimum of the vector lengths of ϕ_m and ϕ_c. The difference $\phi_c^i - \phi_m^i$ (the difference in the i^{th} elements of ϕ_c and ϕ_m) in the cost function is normalized by π. The edge strength cost C_g is obtained from the gradient magnitude image in a similar manner as the normalized edge score Eq. (1) that was used in the basic segment match stage . The path between the two points in the oriented box which minimizes the cost q is taken as the matched contour. This contour extraction procedure is iterated for all adjacent pairs of points in the matched k-segment groups.



a b c

Fig. 3. (a) The oriented boxes for a particular 3 point cluster are shown on the gradient image of the example used in figure 2. (b) An example of the neighborhood search for contour completion using oriented sectors (c) Final object contour detection result shown in blue for the example used in figure 2.

Fine Match using Neighborhood Search: The contours of k-segment groups (k-segment contour) obtained from the gradient image are now stitched together

to obtain the full object contour in the localized boxes. The entire contour is built in steps and a cost function is updated at each step. The following are the steps of our fine object contour detection and completion algorithm:

1. The best k-segment contour (best in terms of the object contour cost discussed below) is used to start the contour build up in each of the localized bounding boxes.
2. The neighborhood at both ends of the best contour is searched to obtain the next k-segment contour candidates (refer to Figure 3(b)).
3. The contour in the gradient image that connects the two k-segment contours is obtained. A comprehensive object contour cost (discussed below) is assigned to this extended contour. In the case of multiple candidates for contour extension, the one that gives the least total cost is chosen.

Steps 2 and 3 are iterated (with the extended matched contour updated as the best contour at step 2) till either the entire object contour is completed or the local neighborhood at both ends of the contour has no more candidate segment groups that satisfy the cost. The object contour completion algorithm is also started from a few other locations other than the best contour to obtain contours missed out in the first search.

The search for candidate segment groups looking from both ends of the current contour is implemented as follows: A sector with its origin placed at the end point of the contour and radius determined by the maximum search distance is used (Figure 3). The orientation of the sector is along the tangent direction at the endpoint of the contour. We use a sector of radius 6 pixels and angle of 60^o for an image of size 357×216 pixels.

Object Contour Cost: The cost function for the extended contour at each iteration of the algorithm is computed as follows:

$$Q = w_s C_s(m,t) + w_a C_a(m,t) + w_b C_b(m,t) + w_g C_g(m,t) \tag{8}$$

where C_s is the scale cost Eq. (3) , C_a the angle cost Eq. (4), C_b the bending energy cost Eq. (7) and C_g the edge strength cost of the updated contour Eq. (1). We empirically determined $w_s = 0.2$, $w_a = 0.2$, $w_b = 0.5$ and $w_g = 0.1$.

The first two components of the cost function Q constrain the local scales and node angles of the extended segment group (obtained by combining the current and candidate k-segment groups). They help account for inter-segment scale and orientation variations. The last two components of Q are those associated with the extended contour (bending energy and edge strength) and account for intra-segment bending deformations.

All the components of Q have already been pre-computed except at the links between the two contours (obtained at step 3 of the algorithm). We use dynamic programming to efficiently update the comprehensive contour cost at each iteration of the algorithm.

The cost at the end of each iteration is compared with a threshold to determine if we should continue matching. All the matched paths are retained in the

next iteration. This type of dynamic programming helps in finding the best contour in a given bounding box efficiently and accurately. The full object contour in each bounding box is progressively built. Wrong contours or matched clutter falling inside the bounding boxes will be discarded by the cost function. Thus, the fine matching stage gives the object contours accurately.

4 Results of Experiments and Discussion

We tested our algorithm on the ETHZ database [5] which has 5 object classes namely Applelogo, Swan, Bottle, Mug and Giraffe. It contains a total of 255 images divided among object classes as apple logos (40), bottles (48), giraffes(87), mugs (48) and swans (32). The dataset contains objects of various scales, orientations, deformations and images with multiple object instances which makes it highly challenging for object detection. We use the simple hand drawn models of [5] for our experiments. We also evaluated the algorithm on 50 additional images obtained from Flickr, each having instances of one of the object classes with considerable shape variations and clutter.

Figure (4) and Figure (5) show some of our results for the database. Images 1-c, 2-c, 4-c, 4-d and 5-g are from the additional dataset while the rest are from the ETHZ database. The results of object contour detection are shown as white bordered lines in the image. Images 4-c, 4-e, 6-h and 7-k show the detection of object contours in very cluttered scenes by our algorithm. The object contour forms only a fraction of the scene in these cases. Images 3-e, 5-k, 7-h and 7-i show our detection results for images with multiple object instances. Images 3-e and 5-k also show detection results of multiple objects across scales present in the same image. In image 5-k, two swans of drastically different sizes are detected by our algorithm. Two of the other swans in image 5-k are near reflections of the model and hence require a reflection of the model to be detected.

Images 2-f, 4-b and 5-i show our detection of applelogos which are heavily deformed compared to the model. Image 4-b is obtained by applying an affine transformation to image 3-b. The results for swans (2-b, 2-d, 1-e, 2-e, 5-k, 6-i) also indicate that our method can handle intra-class variability very well. Image 6-j shows our detection result for a substantially rotated applelogo. We can efficiently handle multiple object instances and object rotation using the centroid boosting principle. Our algorithm successfully extracts the contour (silhouette) of giraffe in test images where there is substantial pose change (1-b, 1-c, 1-f, 3-a, 3-c, 4-a, 5-j, 7-j). Image 4-d shows an example where our algorithm correctly discriminates between an apple and an applelogo based on curvature.

A detection is counted as correct if its bounding-box overlaps more than 50% with the ground-truth one. Our system achieves a very promising average detection rate of 92% at a low value of 0.2 FPPI (False Positives Per Image obtained over all the 305 images). In contrast to [5], we obtain the object bounding box very accurately using the mean and standard deviation of local scales contributing to the object centroid.

Table 1 shows the comparison of results of Ferrari et. al [7] and Ferrari [5] against our method for the ETHZ database. The detection rates at 0.4 FPPI and 0.3 FPPI averaged over the entire ETHZ dataset are shown in the table. The results illustrate that our algorithm performs remarkably well on all the object classes. Huge improvements with respect to [5] and [7] are seen for the giraffe and applelogo classes since we efficiently account for object deformation. Our method is more robust to false positives. The false positives are systematically filtered out at each step of our multi stage approach. Our cost functions are also more comprehensive compared to [5] and [7] since they take into account various factors such as bending energy, intra segment orientations, local scales and edge strength. The computational cost is also quite low (9-10 seconds on an average). We also measure how accurately the output shapes differ from the true object boundaries in a manner similar to [7]. Our method performs quite well in this respect achieving an average error rate of about 2% on the ETHZ dataset.

Table 1. Comparison of detection rates of objects at 0.4 FPPI / 0.3 FPPI and accuracy at 0.4 FPPI

	Apple	Bottle	Giraffe	Mug	Swan
Ferrari et al. [5]:	72.7/56.8	90.9/89.1	68.1/62.6	81.8/68.2	93.9/75.8
Ferrari et al. [7]:	86.4/84.1	92.7/90.9	70.3/65.9	83.4/80.3	93.9/90.9
our system :	97.7/95.5	92.7/90.9	93.4/91.2	95.3/93.7	96.9/93.9
accuracy:	1.2	2.1	2.3	2.9	1.8

5 Conclusion

In this paper, an efficient multi-stage approach to object recognition in real, cluttered images that is robust to scale, rotation and intra class variability is presented. Shape information from simple model sketches is used to localize objects and detect their contours. Experiments confirm that k-segment grouping together with centroid boosting can localize the objects accurately in an image. Finally the object contours are extracted in the fine matching stage using a comprehensive score and dynamic programming. Separation of the matching into two stages allows us to detect objects fast while maintaining accuracy and matching of only k-segment groups initially allows to detect objects that may be partially occluded or cluttered. Thus, the method handles local scale variations, bending deformations, clutter, multiple object instances and rotations of the object in an efficient manner and achieves results that are quite promising. Future work would involve combining other object properties like color and texture along with edge information to detect objects.

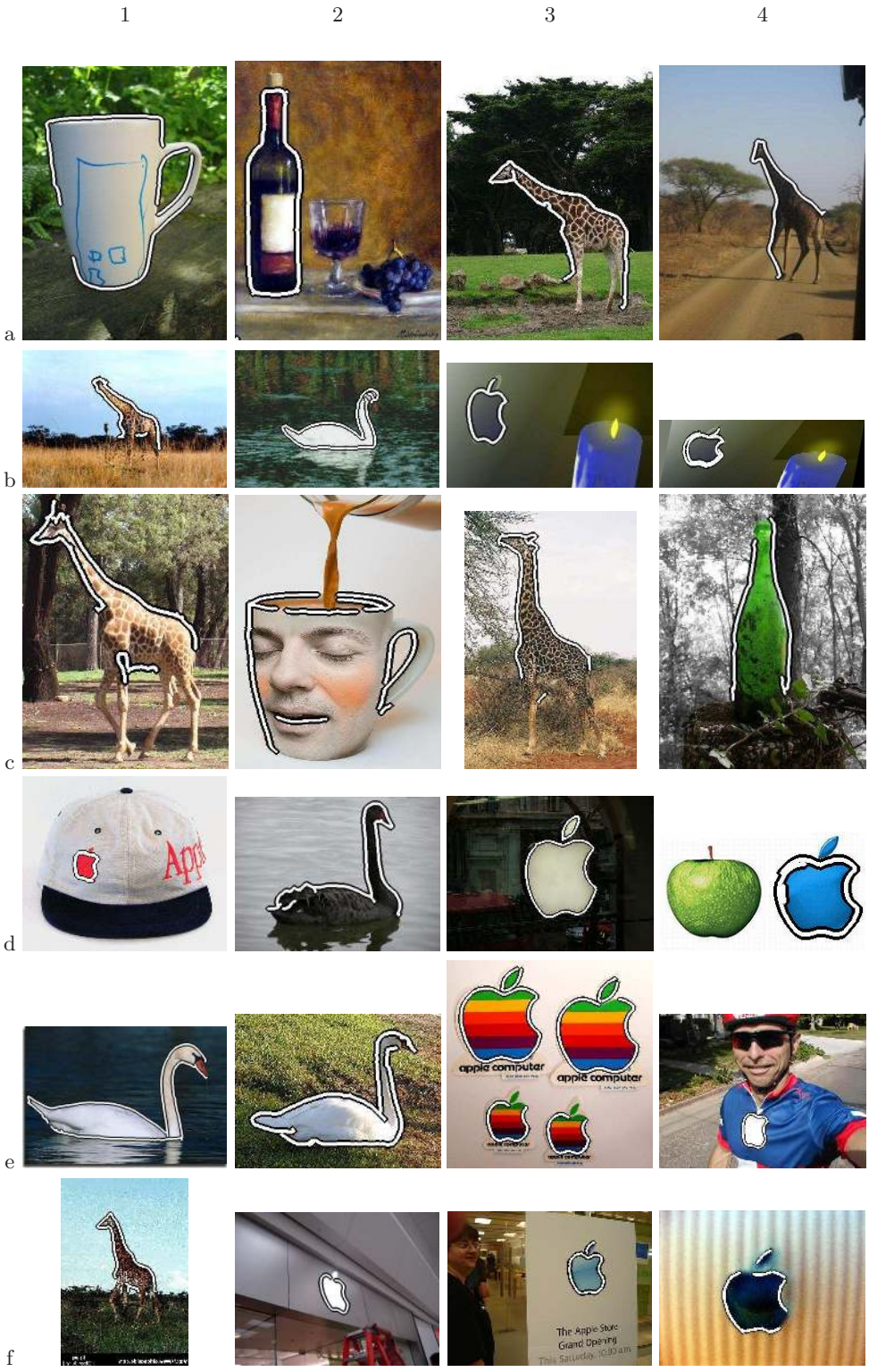

1
2
3
4
a
b
c
d
e
f
apple computer
apple computer
The Apple Store
Grand Opening

Fig. 4.

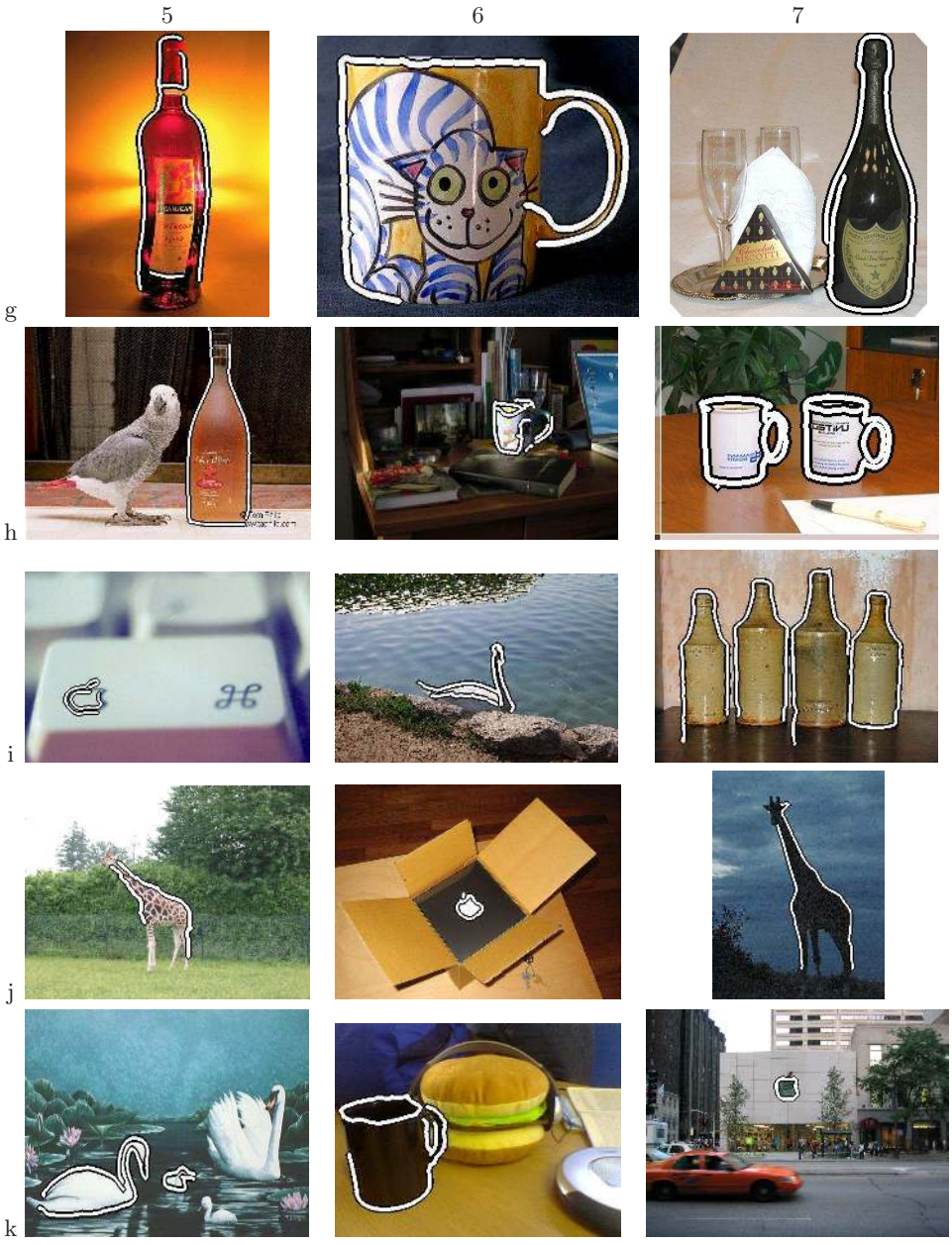


Fig. 5.

References

1. Canny, J.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **8** (1986) 679–698
2. Martin, D., Fowlkes, C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004) 530–549
3. Huttenlocher, D., Klanderman, G., Rucklidge, W.: Comparing images using the hausdorff distance. IEEE Transactions on Pattern Analysis and Machine Intelligence **15** (1993) 850–863
4. Thayananthan, A., Stenger, B., Torr, P., Cipolla, R.: Shape context and chamfer matching in cluttered scenes. In: IEEE Conference on Computer Vision and Pattern Recognition. (2003) 127–133
5. Ferrari, V., Tuytelaars, T., Gool, L.V.: Object detection by contour segment networks. In: European Conference on Computer Vision. (2006) III: 14–28
6. Ferrari, V., Fevrier, L., Jurie, F., Schmid, C.: Groups of adjacent contour segments for object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **30** (2008) 36–51
7. Ferrari, V., Jurie, F., Schmid, C.: Accurate object detection with deformable shape models learnt from images. In: IEEE Conference on Computer Vision and Pattern Recognition. (2007) 1–8
8. Opelt., A., Pinz, A., Zisserman., A.: A boundary-fragment-model for object detection. In: European Conference on Computer Vision. (2006) 575–588
9. Leibe, B., Schiele, B.: Scale invariant object categorization using a scale-adaptive mean-shift search. In: DAGM04. (2004) 145–153
10. Shotton, J., Blake, A., Cipolla, R.: Multi-scale categorical object recognition using contour fragment. IEEE Transactions on Pattern Analysis and Machine Intelligence (2008)
11. Shotton, J., Blake, A., Cipolla, R.: Contour-based learning for object detection. In: IEEE International Conference on Computer Vision. (2005) 503– 510
12. Wu, B., Nevatia, R.: Simultaneous object detection and segmentation by boosting local shape feature based classifier. In: IEEE Conference on Computer Vision and Pattern Recognition. (2007) 1–8
13. Wu, Y., Si, Z., Fleming, C., Zhu, S.: Deformable template as active basis. In: IEEE International Conference on Computer Vision. (2007) 1–8
14. Felzenszwalb, P., Schwartz, J.: Hierarchical matching of deformable shapes. In: IEEE Conference on Computer Vision and Pattern Recognition. (2007) 1–8
15. Basri, R., Costa, L., Geiger, D., Jacobs, D.: Determining the similarity of deformable shapes. Vision Research **38** (1998) 2365–2385