Department of Electrical
and
Computer Systems Engineering

Technical Report
MECSE-28-2006

Object Detection by Global Contour Shape

K. Schindler and D. Suter

MONASH
UNIVERSITY

Object Detection by Global Contour Shape

Konrad Schindler and David Suter

Electrical and Computer Systems Engineering,
Monash University, Clayton, 3800 VIC, Australia

konrad.schindler@eng.monash.edu.au

November 14, 2006

Contents

Abstract

We present a method for object detection based on global shape. A distance measure for elastic shape matching is derived, which is invariant to scale and rotation, and robust against non-parametric deformations. Starting from an over-segmentation of the image, the space of potential object boundaries is explored to find boundaries, which have high similarity with the shape template of the object class to be detected. An extensive experimental evaluation is presented. The approach achieves a remarkable detection rate of 91% at 0.2 false positives per image on a challenging data set.

1 Introduction

The aim of this report is to investigate the potential of global shape as a cue for object detection and recognition. Both are important visual tasks, which recently have received a lot of attention in computer vision. In recent years, the dominant approach has been recognition using local appearance. An object class is represented by a collection of smaller visual stimuli, either linked by a configuration model ("part-based models" [7, 31]), or without using the relative position information ("bag-of-features models" [20, 34]). Spectacular results have been achieved, partly due to new methods of robustly describing local appearance [3, 20].

However, local appearance is clearly not the only cue to object detection, and in fact for some classes of objects the local appearance contains very little information, while they are easily recognized by the shape of their contour (see Figure 1). Detection by shape has been investigated in earlier work. The basic idea common to all methods is to define a distance measure between shapes, and then try to find minima of this distance A classical method is chamfer matching [6, 16, 27], in which the distance is defined as the average distance from points on the template shape to the nearest point on the image shape. However, chamfer matching does not cope well with shape deformations. Even if a hierarchy of many templates is used to cover deformations, the rate of false positives is rather high (typically >1 false positive per image). More sophisticated methods allow the shape template to deform, so that it can adapt to the image content, including methods such as spline-based shape matching [12], diffusion snakes [10], and active shape models [8]. The distance in this case is the deformation energy (the "stretching") of the template shape. However, all the mentioned deformable template matching techniques are local optimizers, and thus require either good initial positions or clean images to avoid local minima. They are therefore not a viable option by themselves, because we are concerned with the detection of objects, which may appear anywhere in the image, and form only a small part of the entire image content. In this report, we will propose a search strategy, which relieves the problem of false local minima, at the cost of having to re-evaluate the shape distance at each search step. We will devise a probabilistically motivated shape distance, which can be computed more efficiently. The proposed distance requires *closed* contours, but nevertheless can be interpreted as a deformable template matching method.

Recent shape matching techniques include the one of [14], which finds the optimal grid location for all vertices of a polygonal model by dynamic programming. However, the method

Figure 1: Some object classes are by their nature easier to recognize by shape than by local appearance. Examples from the *swan* and *hat* classes of our data set.

is quadratic in the number of potential locations, so the object needs to cover a large part of the image, otherwise localization becomes prohibitively expensive (quantizing the location to $\frac{1}{100}$ of the image size already leads to computation times in the order of one hour to detect a single object in an image). A powerful shape matching method has been presented in [4], which uses integer quadratic programming to match sets of points sampled from object edges. In practice, either a good initial position or relatively clean images are required, similar to deformable template matching methods, because computational demands limit the amount of outliers the method can deal with.

Recently, researchers have moved from the original models, which describe the shape of the entire object in one piece (from now on called *global* shape models), to an object model consisting of local pieces of the contour in a certain configuration [11, 28, 33], similar to earlier appearance-based models. Methods using contour fragments generally also model their relative position, since short boundary fragments are not specific enough to use in a bag-of-features model: the shape of the contour is only a useful cue when seen in the global context of the object, while the local shape of fragments contains little information. This last observation leads to a renewed interest in matching larger sections of contours [15].

Here, we will present an approach to object detection by global shape, which is based on elastic matching of contours. An edge-map with closed edge chains is produced by segmentation into super-pixels. A probabilistic measure for the similarity between two contours is derived, and combined with an optimization scheme to find closed contours in the image, which have high similarity with a template shape. The method only requires a single object template, which in our case is a hand-drawn sketch, but could also be learned from examples. An extensive experimental

evaluation is presented which compares the elastic matching approach to chamfer matching as well as an appearance-based method.

2 Detection by contour matching

In the following section an elastic shape-matching approach to object detection is described in detail. It covers the detection of potential object contours in the input image, the contour model used to describe a contour's shape, a probabilistically motivated distance measure to compare a candidate shape to an object class template, and an optimization framework to find shapes in the image, which have high similarity to the template.

2.1 Segmentation vs. edge detection

The first step towards shape-based object detection is to extract potential object contour points from the input image, which then are compared to a shape template. The short-comings of this basic edge detection has been one of the major difficulties for shape-based object detection.

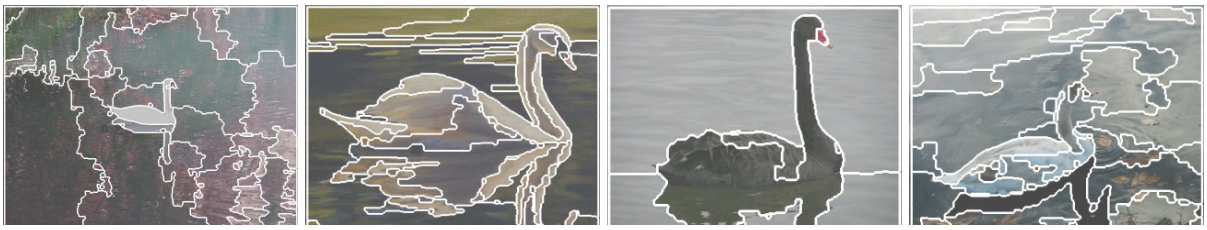
Deformable template matching techniques have largely avoided the problem – although they provide a way of measuring the distance between two shapes, they cannot be regarded as object detection methods: they generally assume that there are not many spurious edges, which will distract the search, hence they either require clean images of the objects without clutter, or an approximate solution, which assures the optimization is not mislead by the clutter.

On the other hand, earlier attempts at shape-based object detection were plagued by high rates of false positives (1-2 per image in [16]), partly because of the poor quality of the underlying edge detection, which for real images often gives broken contours, swamped by large amounts of clutter. To overcome the problem, Ferrari et al. have used the sophisticated (but computationally expensive) edge detection method of [23], and then link the resulting edges to a network of connected contour segments by closing small gaps.

Instead, we rely on a different way of avoiding clutter and obtaining good edges. The basic intuition is similar: objects shall have closed contours, and clutter edges will normally not form closed contours. We therefore prefer to work with an edge map, which *only* consists of closed contours to begin with. This cannot be easily achieved by pixel-wise edge detection, but is trivially solved by segmentation, since the boundaries of segments obviously are closed. To

make sure the boundaries are complete, segmentation is parametrized in such a way that it almost certainly leads to over-segmentation, rather than risking that parts of the object contour are missing. This strategy of segmenting into "super-pixels" has been inspired by [5, 18, 29], where the authors also rely on over-segmentation, respectively multiple segmentations with varying parameters, to make sure no important boundaries are missed.

For our purpose, we observe that region-based methods, which merge pixels satisfying some homogeneity criteria, usually give better boundaries. Spectral segmentation methods such as normalized cuts [32], which are really based on edge detection, tend to give overly smooth segmentations. This is not a problem if the boundaries only serve to delineate regions, which are then used to build an appearance-based model, but it does impair our application, where the shape of the boundary itself is the cue.



(a) (b) (c) (d)

Figure 2: Segmentation of swans in natural images. (a) An easy image – complete object boundary, object mostly covered by one segment. (b) The most frequent case – complete object boundary, object broken up into several segments. (c) A difficult case – segments do not completely trace the object boundary, but the characteristic shape is preserved. (d) A rare case, where segmentation catastrophically fails.

In this work we have used the excellent "statistical region merging" segmentation of [25], which not only produces good segmentations, but also is very efficient: on a standard PC, segmentation starting from the raw input image takes less than 1.5 seconds for a 512×512 pixel image. Some results are presented in Figure 2 to show the quality of segmentations obtainable with natural images. We note that if one accepts over-segmentation, the object contour is completely found in the majority of images. In some cases, parts of the boundary are missed, but nevertheless the contour does preserve enough of the specific object shape to perform detection. Only in rare cases does the segmentation catastrophically fail to delineate the object.

The edge map obtained by segmentation forms the basis of the entire detection chain outlined in the following: the fact that the extracted contours are naturally *closed* is an essential ingredient

for the shape matching, while the neighborhood system defined by the super-pixels guides the search in the image space.

2.2 Shape Model

We adopt the following shape model to describe an object: a shape X is approximated by a closed polygon with a fixed number of equally spaced vertices N. Since the points are equally spaced, the sequence of points can be parametrized by an integer arc length: $X = \{\mathbf{x}(u), u = 0 \dots N - 1\}$. The last vertex coincides with the first one for computational simplicity: $\mathbf{x}(N) = \mathbf{x}(0)$.

As a shape descriptor X' for the contour X, we use the sequence of tangent angles. To attach the descriptor to the contour rather than the image coordinate system, the angles are normalized by aligning the first tangent vector with the x-axis. Let the tangent vector at point $\mathbf{x}(u)$ be denoted by $\dot{\mathbf{x}}(u) = \big[\dot{x}(u), \dot{y}(u)\big]$, then

$$X'(u) = \{x'(u), u = 0 \dots N - 1\} \quad \text{, where} \quad x'(u) = \arctan \frac{\dot{y}(u)}{\dot{x}(u)} - \arctan \frac{\dot{y}(0)}{\dot{x}(0)} \,. \tag{1}$$

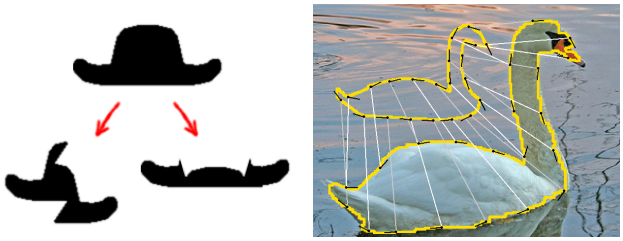
To account for the noise-sensitivity of differential quantities, it is usually recommended to smooth the raw gradients, which we do with a simple averaging filter over a 3-neighborhood.

It has been suggested that when using differential invariants to describe shapes, curvatures (sometimes called "turning angles") are more appropriate than tangents [1, 2]. While curvature is theoretically robust to articulated deformation, we found that it does not work well in practice, because the degree of invariance becomes too high: even a simple bending, which only changes the curvature in a few points, can dramatically alter the shape of an contour (see Figure 3(a)). Tangents better preserve the global shape. An experimental comparison of the two option is given in section 3.8.

The descriptor is by construction invariant to translation, and to scale including the smoothing: segmentation is always carried out at the full resolution, so the contour of an object will have more small details if it appears larger in the image, making the local tangent estimation directly from the contour unstable. Sampling a fixed number of vertices and using these to compute the tangent implicitly rescales all shapes to a common size and ensures the descriptor has the same level of detail independent of the scale.

Given the segmentation of the image into super-pixels, it is easy to compute the shape descriptor: any group of adjacent super-pixels has a closed contour, which can be extracted with a boundary-tracer, and down-sampled to the desired resolution N. In all experiments presented

here, we have set $N = 100$, which has proved to be a reasonable setting for all classes we have tested. To improve efficiency, it may be useful to separately determine the required number of vertices for each class from the shape complexity. An experimental comparison of different contour resolutions is given in section 3.7.



(a) (b)

Figure 3: Representing and matching shape. (a) Curvature is not suitable as shape descriptor – changing the curvature of the hat boundary in only four points results in a dramatically different shape. (b) Non-linear elastic matching of contours. Only a subset of the total of 100 tangents per contour has been plotted.

2.3 Matching Shapes

Given are two shapes X and Y, one for the shape template, and one for a candidate contour extracted from a test image. A matching between the two shapes is a function, which associates the point sets $\{\mathbf{x}(u)\}$ and $\{\mathbf{y}(v)\}$ (both parametrized by their arc length), such that each point on either curve has at least one corresponding point on the other curve. The same point on X can have multiple matches on Y and vice versa, as long as the ordering of both sequences is preserved. A matching is given by the sequence of arc lengths of the matching contour points.

For the moment, assume that we know one pair of matching points on the two contours (this restriction will be removed later), and let the arc lengths at these points be $(u = v = 0)$. Then a matching of the two contours is given by

$$\mathcal{V}(X,Y) = \left\{\langle u_i \Leftrightarrow v_i\rangle, i = 1 \ldots K_\mathcal{V}\right\} = \left\{\langle 0 \Leftrightarrow 0\rangle, \ldots, \langle u_i \Leftrightarrow v_i\rangle, \ldots, \langle N \Leftrightarrow N\rangle\right\} . \qquad (2)$$

A matching $\mathcal{V}$ has a variable number $K_\mathcal{V}$ of individual point matches, with $N \leq K_\mathcal{V} \leq 2N$ because of the requirement to close the loop at $\langle 0 \Leftrightarrow 0\rangle$. The probability that two points match can be decomposed into two components, one for their similarity, and one for the local stretch of the curve: $P(u_i \Leftrightarrow v_i) = P_D(u_i \Leftrightarrow v_i) P_S(u_i \Leftrightarrow v_i)$. An assignment between two points is deemed

more likely, if the local tangent angles of the two curves are similar, and if no local stretching is required to make the two points match. See Figure 3(b) for an illustration. To account for the stretching, and to ensure a complete and ordered matching, we define

$$P_S(u_i \Leftrightarrow v_i) = \frac{1}{1+e^{-E}} e^{-S(u_i,v_i)} \quad , \text{ where} \tag{3}$$

$$S(u_i, v_i) = \begin{cases} 0 & \text{if } (u_{i-1} = u_i - 1 \ , \ v_{i-1} = v_i - 1) \\ E & \text{if } (u_{i-1} = u_i \ , \ v_{i-1} = v_i - 1) \text{ or } (u_{i-1} = u_i - 1 \ , \ v_{i-1} = v_i) \\ \infty & \text{else} \end{cases} . \tag{4}$$

The first two cases assure that matching two points without stretching the curve has a higher probability than matching them with stretch. The third case assures only valid assignments are made, by forbidding assignments which skip any u_i or v_i, and assignments which do not follow the sequence.

Next, we have to provide $P_D(u_i \Leftrightarrow v_i)$, or in other words, we have to decide how to measure the dissimilarity $D(u, v)$ between two points $\mathbf{x}(u)$ and $\mathbf{y}(v)$. The dissimilarity is a non-negative function of the tangent angle difference, defined over the interval $[-\pi, \pi]$. There are different possibilities such as the absolute value, square, or cosine (see section 3.5). Whichever we decide to use, we end up with a matching probability of the form

$$P_D(u_i \Leftrightarrow v_i) = \frac{1}{H(B)} e^{-B \cdot D(u_i,v_i)} \ , \tag{5}$$

where, given the functional form of the exponent, $H(B)$ is a constant, which normalizes the probabilities so that they integrate to unity. As will be seen later, the constant B does not qualitatively change the cost function, but only leads to a different penalty for contour stretching. In our experiments (except for the comparison of different dissimilarity measures in section 3.5), we have set $B = 1$, and used a wrapped Laplace distribution, so $H = 2(1 - e^{-\pi})$ and $D(u_i, v_i) = |x'(u) - y'(v)|$.

Taking the product over all points on the contour gives the total probability of a certain matching $\mathcal{V}$. Note that the number of individual assignments $\langle u_i \Leftrightarrow v_i \rangle$ reaches its minimum $K_\mathcal{V} = N$ if no stretching occurs at all, and its maximum $K_\mathcal{V} = 2N$ if all assignments produce local stretching of one of the contours.

$$P(\mathcal{V}) = \prod_{i=1}^{K_\mathcal{V}} P_D(u_i \Leftrightarrow v_i) \prod_{i=1}^{K_\mathcal{V}} P_S(u_i \Leftrightarrow v_i) \ . \tag{6}$$

Among the combinatorial set of possible matchings between two contours, we are interested in the most likely one, which we will call the *best matching*. The best matching is the one which minimizes the negative log-likelihood

$$C(\mathcal{V}) = B\sum_{i=1}^{K_\mathcal{V}} D(u_i, v_i) + \sum_{i=1}^{K_\mathcal{V}} \big(Q + S(u_i, v_i)\big) \text{ , where } Q = -\log\big(H(1+e^{-E})\big) \ . \tag{7}$$

If we make sure that only valid matchings are compared, then the case $S = \infty$ will never happen. Furthermore there will be exactly $2(K_\mathcal{V} - N)$ individual point matches with $S = E$, and $(2N - K_\mathcal{V})$ matches with $S = 0$. Using these quantities, we can expand the second sum in equation (7), subtract the constant NQ from the cost, and multiply by B, to yield the cost for the best matching (the "distance" between the two shapes)

$$\begin{aligned} \widetilde{C}(X,Y) &= \min_{\mathcal{V}} \left[\sum_{i=1}^{K_\mathcal{V}} |D(u_i, v_i)| + B(K_\mathcal{V} - N)(Q + 2E) \right] \\ \text{s.t. } & u_1 = v_1 = 0 \ \ ; \ \ u_K = v_K = N \ \ ; \ \ \forall\, i \ : \ u_{i+1} - u_i \in \{0,1\} \ , \ v_{i+1} - v_i \in \{0,1\} \ \ . \end{aligned} \tag{8}$$

The cost function is of a form which allows one to efficiently find the best matching for a given pair of starting points by *dynamic programming*, with complexity $\mathcal{O}(N^2)$, as described in the following section. A distance defined in this way is known in the pattern recognition literature as the "non-linear elastic matching distance with stretch-cost R", abbreviated NEM_R [9, 36], where the stretch-cost is $R = \frac{1}{2}B(Q + 2E)$. The distance is formally a semi-metric, and satisfies a relaxed form of the triangle inequality [13]. More important for practical purposes, it increases gracefully with growing distortion due to viewpoint change or non-rigid deformation. We note in passing that a similar shape distance based on integral rather than differential invariants has recently been proposed by [21].

So far, we still need to know one pair of matching points in advance – in the derivation above, these points have been chosen as the starting points $\mathbf{x}(0)$ and $\mathbf{y}(0)$. This can be resolved easily by iteratively testing different relative rotations, which at the same time makes the cost invariant to rotation. To achieve a rotation of contour X relative to contour Y, the tangent angles $x'(u)$ have to be shifted in a circular manner, and then re-normalized such that $x'(0) = 0$ for the new starting point. Note that in some situations, such as for rotationally symmetric objects or objects which cannot be turned on their head, it may be better to opt for partial invariance by only testing a certain range of rotations.

2.4 Computing the shape distance

Finding the matching $\widetilde{C}(X, Y)$ between two ordered sets, which results in the minimum cost, is a classical application of dynamic programming [19, 24, 26, 30]. Formally, the matching problem is converted into a path-search problem in a directed graph, in which each node represents a matching of a certain point pair $\langle u_i \Leftrightarrow v_i \rangle$. Here, we will give a brief intuitive explanation, rather than the exact graph-theoretic formulation, which is quite cumbersome. Again, let us for the moment assume that one matching point pair $u = 0, v = 0$ is known.

The first step is to compute the $(N \times N)$ dissimilarity matrix $\mathcal{D}$, with the dissimilarities $d_{ij} = D(u_i, v_j)$ between all possible pairs of vertices as its elements (in practice, there is an upper limit to the shape deformation, so only a band of values along the diagonal is required). We know that the first point (which is the same as the last point, see above) must match: $\langle 0 \Leftrightarrow 0 \rangle, \langle N \Leftrightarrow N \rangle$. The task thus is to find the path from d_{00} to d_{NN}, which has the lowest cumulative cost due to dissimilarity and stretching. Furthermore, every vertex on either contour must have at least one match, and the vertex ordering must be preserved, so at any point $\{i, j\}$ along this path there are three possible continuations:

1. match the next pair of vertices without stretching the contours, by moving to $\{i+1, j+1\}$;
2. locally stretch X: match its next vertex to the current one on Y, by moving to $\{i+1, j\}$;
3. locally stretch Y: match its next vertex to the current one on X, by moving to $\{i, j+1\}$.

The three possible steps directly define the recursion for the cumulative matching cost matrix $\mathcal{R}$ – see Algorithm 1. The distance between the two contours is the total matching cost r_{NN}, which has been accumulated when one is back at the starting point. The pointwise assignment between the two contours is not required for our purposes, but can be easily found by back-tracking the path from r_{NN} to r_{11} – see Figure 4.

To achieve invariance to rotation, one has to add an outer loop to the shape distance computation, which moves the starting point along one of the two contours. As mentioned above, in many cases partial rather than complete rotation invariance is desired, either because an object class exhibits rotational symmetry, or because the object is rarely encountered in a certain range of orientations (for example, the *starfish* and *swan* classes in section 3). For such objects, a practical solution is to roughly align them with a simple heuristic, and only allow a restricted range of rotations around the alignment. In our experiments, we have simply set the initial orientation

Input: Shape descriptors $\{x'(u)\}$ and $\{y'(v)\}$

Output: $ShapeDistance$

allocate $\mathcal{D}_{N\times N}, \mathcal{R}_{N\times N}$;

`// compute vertex dissimilarities`

for $i = 0$ *to* N **do**

  for $j = 0$ *to* N **do**

    $d_{i,j} \leftarrow D(u_i, v_i)$;

  end

end

`// compute cumulative matching cost`

$r_{0,0} \leftarrow d_{0,0}$;

for $i = 1$ *to* N **do**

  $r_{i,0} \leftarrow r_{i-1,0} + d_{i,0} + R$;

  $r_{0,i} \leftarrow r_{0,i-1} + d_{0,i} + R$;

end

for $i = 1$ *to* N **do**

  for $j = 1$ *to* N **do**

    $h_{10} \leftarrow r_{i-1,j} + R$;

    $h_{01} \leftarrow r_{i,j-1} + R$;

    $h_{11} \leftarrow r_{i-1,j-1}$;

    $r_{i,j} \leftarrow \min(h_{10}, h_{01}, h_{11}) + d_{i,j}$;

  end

end

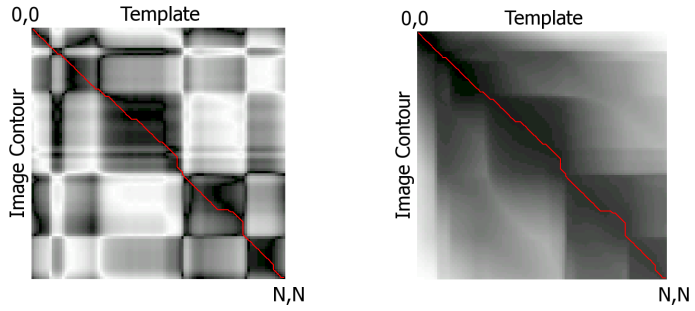
$ShapeDistance \leftarrow r_{N,N}$;

Algorithm 1: Dynamic programming recursion to find the shape distance $\widetilde{C}(X, Y)$.

by identifying the topmost vertices on both contours, i.e. the ones with the minimal y-coordinate. Other possibilities include using curvature maxima, or principal components of the contour.

2.5 Detecting objects

The non-linear elastic matching distance is a measure for the similarity between two contours. What is still missing is an optimization framework to find the contours in the edge map, which have the highest similarity with a given object template. Any group of neighboring super-pixels forms a closed contour, and the combinatorial set of all such contours is the search space for ob-



(a) (b)

Figure 4: Minimizing the shape distance by path search. Best path overlayed on (a) dissimilarity matrix $\mathcal{D}$, and (b) cumulative cost matrix $\mathcal{R}$. Brighter colors denote larger values.

ject detection. Unfortunately, the non-linear elastic matching distance over this set is essentially an oracle, and not amenable to any simple optimization procedure.

Currently, we are using a greedy multi-start gradient descent (pseudo-code is given in Algorithm 2). Each super-pixel in turn is chosen as seed region. Starting from the seed region, the method attempts to reduce the elastic matching distance as much as possible by merging one of the neighboring super-pixels into the region. This is repeated until a local minimum is reached, and no further improvement is possible. The local minima for all seed regions are considered potential detections. A detection threshold is used to weed out those with low similarity.

Since several starting regions may converge to the same or similar local minima, non-minima suppression is performed as a last step: each candidate detection is visited in decreasing order of similarity, and all candidates, which overlap with the current one by more than a certain threshold (in all our experiments set to 30%) are removed.

The described optimization scheme is rather weak – if adding two super-pixels together to a solution would lead to a better minimum, this possibility will be missed, unless adding only one of them is the best intermediate step. This may well prevent some correct detections in cases where super-pixels have complicated shapes. Replacing the greedy descent with Tabu-search [17] did not improve the results in our experiments, and the development of a more sophisticated optimization scheme is left for future research.

Input: object template, list of super-pixels, detection threshold, overlap threshold

Output: detected objects

foreach $SuperPixel$ **do**

`// greedily maximize similarity`

set object candidate: $Candidate \leftarrow SuperPixel$;

set $Distance \leftarrow NEM_R(Candidate, Template)$;

while $Distance$ *decreases* **do**

set $Neighbors \leftarrow$ adjacent Superpixels of $Candidate$;

foreach $Neighbor$ **do**

compute $NEM_R(Candidate \cup Neighbor, Template)$;

end

find $Neighbor_i$ which most decreases the $Distance$;

set $Candidate \leftarrow Candidate \cup Neighbor_i$;

end

`// only keep candidate if similarity is high`

if $NEM_R(Candidate, Template) > DetectionThreshold$ **then**

delete $Candidate$;

end

end

`// non-minima suppression in candidate list`

sort $Candidates$ in descending order of $Distance$;

for $i = 1$ *to* $\#Candidates$ **do**

for $j = i + 1$ *to* $\#Candidates$ **do**

if *Overlap*$(Candidate_i, Candidate_j) > OverlapThreshold$ **then**

remove $Candidate_j$;

end

end

end

`// return detected objects`

$Detections \leftarrow Candidates$;

Algorithm 2: Multi-start gradient descent to detect objects which are similar to a template.

3 Experimental results

We have extensively tested the proposed object detection scheme. The experimental evaluation starts with detection results on different data sets, and then presents an in-depth study of different variants of the method and different parameter settings, all on the same standard data set.

3.1 Comparison with baseline methods

As a main testbed, we use a collection of four diverse object classes, which have in common that they are mainly defined by their global shape, while they have either little texture at all, or strongly varying texture, which is difficult to use as a generic cue. The database contains 50 images for each class collected from `Google` and `Flickr`, some of which show multiple instances of the same class. Objects appear over a range of scales, with different backgrounds and large, realistic intra-class shape variation.[1]

As object model, the system is given a hand-drawn shape sketch for each class, together with the (empirically determined) stretch cost R. Note that the optimal stretch cost may differ between classes: on one hand, some shapes are more unique than others, and require a lower penalty for stretching in order to be discriminative; on the other hand, certain types of objects can deform more than others (see section 3.6 for more details on this issue).

The test images were segmented into super-pixels with the "statistical region merging" code of [25] (code available from author's web-page). All images were segmented with the same parameters: since it is not known in advance, whether a test image contains an instance of a certain object, it is not permissible to use class-specific information in the segmentation.

Figure 5 shows example detections. Each image is shown together with the segmentation into super-pixels (in white), and the detection result (in black). The last image for each class is an example of a false detection.

It can be seen that the method can cope well with shape variations including significant viewpoint changes, see for example images A2, B3, B4, images C2, C4, D2, and images E1, F1, F2. Note that a single template is able to cover wide intra-class variability, which would require a large fragment base in a contour-fragment model. As long as the characteristic shape is largely

[1] Two classes, *swans* and *applelogos*, are extensions of the same classes in the data set of [15]. We could not use all their data: our method cannot handle open templates; and, although in principle possible, we do not consider consensus voting for templates consisting of more than one contour.

preserved, the method also exhibits some robustness to imperfect segmentations, which deviate from the true contour, for example A2, A5, B4, F3. Some correct detections have imperfect contours, because the super-pixels happen to fall such that a part of the true object matches the template better than the whole object, see D2, F4, G3. Super-pixel segmentation can in many cases avoid overly strong edge clutter – B3, F2. Very fine-grained segmentation increases the chance of a false positive, such as in D4, but even in the presence of many super-pixels the chance for this is comparatively low – see E3, H1. The object does not have to cover a large image area – D3, H2. Finally, typical false detections indicate that elastic matching allows some deformations, which lead to implausible shapes - see B5, H5. This suggests that the method could be improved by learning the permissible deformations of a class from examples and making the dissimilarity cost dependent on the arc length.

Quantitative evaluation results are shown in Figure 6. Four detection methods were compared:

Chamfer matching with Canny edges. Edge pixels are extracted from the test image with the Canny edge detector, and the Euclidean distance transform is computed on the edge map. The distance transform image is then convolved with the object template at different scales and rotations. We have used 8 scales with a scale factor of 1.2 between consecutive scales, and 20 rotations equally spaced between -36° and 36°. After checking all templates, non-maxima suppression was performed by ordering the potential detections by increasing chamfer distance, and iteratively removing all detections, which overlap a detection with a lower distance by more than 30%. The results are given in Figure 6 (red dashed lines). On average, chamfer matching achieves a detection rate of 16% at 0.2 false positives per image (FPPI), and 27% at 0.4 FPPI.

Chamfer matching with Super-pixels. In order to validate our way of finding the edge map, we have used the same chamfer matching method again, but with the edge maps produced by region-based segmentation, instead of the Canny detector. The reduced amount of clutter, and more faithful edge maps, give a significant improvement in the detection results – see Figure 6 (blue dashed lines). Detection rates are 23% at 0.2 FPPI, and 38% at 0.4 FPPI. These correspond well with the results of [15], where chamfer matching with good quality edge maps achieves approximately 23% detection rate at 0.2 FPPI, and 39% at 0.4 FPPI.

Part-based recognition. This serves as a baseline for part-based recognition using local appearance. Ten representative images were picked from each class, and a five-part star-graph model was manually trained. All five parts had to be chosen on distinctive, highly curved parts of

Figure 5: Detection results with elastic matching.

the object boundary, since no characteristic parts with discriminative texture are present in the objects' interior. Similarity between parts was measured with the absolute value of the normalized cross-correlation, to account for the fact that our data contains both dark object instances on lighter background, and light instances on darker background. The relative weight between appearance and configuration of the regions was chosen empirically for each class to give the best detection results. Non-maxima suppression was performed in the same way as for chamfer matching.

The comparison is slightly biased in favor of the part-based method: firstly, the ten training images are part of the test data. secondly, we found that a multi-scale version produced large numbers of false positives, therefore the results were obtained at a fixed scale, after manually rescaling the test data to a common object size. Note that part-based recognition, as an appearance-based method, is confused by object classes with strongly varying texture. In the absence of distinctive parts such as eyes, wheels etc., distinctive regions are only available along the boundary, and part-based recognition then works best for classes which are relatively homogeneous and untextured (*swans*, *hats*) – see Figure 6 (red solid lines). Average detection results were 51% at 0.2 FPPI, and 58% at 0.4 FPPI, with much lower results for the *starfish* data, which exhibits a wide variety of object textures.

Elastic Matching. The proposed method as described in section 2.5, with $N = 100$ points per contour, and the L_1-distance as tangent similarity (see section 3.5 for other distances). The starting point on both shapes was chosen to be the point with the smallest y-coordinate, and rotations of up to ± 10 vertices were allowed, similar to the $\pm 10\%$ rotation used for chamfer matching. Note especially that the detection rates grow very quickly as the detection threshold increases, giving high detection rates already at very low numbers of false positives – see Figure 6 (solid blue lines). Average detection results were 91% at 0.2 FPPI, and 93% at 0.4 FPPI. These numbers cannot be directly compared to other methods, since they use different data sets, but the best reported results for detection with a single shape template, of which we are aware, are approximately 65% at 0.2 FPPI, and 82% at 0.4 FPPI [15].

3.2 ETHZ shape classes

In order to get a direct comparison to the state-of-the-art in contour-based detection, we have also tested our method for the *swan* and *applelogo* classes of Ferrari et al.: their data-set consists

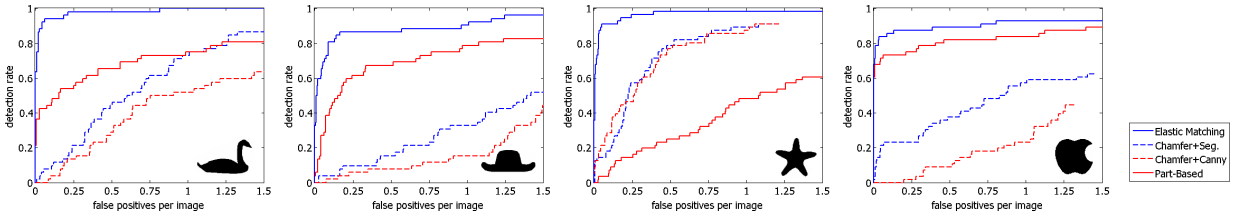


Figure 6: Comparison of detection results.

of 255 images, including 40 with Apple-logos and 32 with swans, which also form part of the previously used test database. The results are summarized in Table 1 – global elastic matching clearly outperforms piecewise contour segment matching. Note however that the latter is more general, and can handle open contours. Note also that our results for the Apple-logo are obtained without using the leaf, because we have chosen to restrict this investigation to a single contour, and have not implemented consensus voting over multiple closed polygons.

	0.2 FPPI			0.4 FPPI		
	Swan	Apple	Avg.	Swan	Apple	Avg.
NEM_R	94%	86%	89%	97%	88%	92%
[15]	73%	52%	61%	94%	73%	82%

Table 1: Detection rates of global elastic matching and piecewise contour matching.

3.3 Caltech101 classes

As a further dataset, we have selected three classes with characteristic shapes from the Caltech101 database. The classes *stopsign*, *yinyang*, and *mandolin*, together with 99 images from the *background* class, were selected, to form a dataset with a total of 266 images, including 64 with stop-signs, 60 with Yin Yang symbols, and 43 with mandolins (see Figure 7 for examples). The images are dominated by the object and usually contain little clutter, but some are of bad quality, with poor contrast, low resolution, and strong compression artifacts. The shape templates and detection results are shown in Figure 8.

While the method performs well on the first two classes, it has difficulties with some images of the *mandolin* class. About 65% of the objects are detected fine, then the detection rate flattens out. The reason is that, due to poor image contrast (see Figure 9 for examples), the super-pixel segmentation catastrophically fails, so that important parts of the contour are missing. Therefore some objects cannot be detected. Obviously, the missing contours also impair chamfer matching,

Figure 7: Example images from of Caltech101 data set.

so that the curves for elastic matching and chamfer matching based on super-pixels converge. Chamfer matching based on Canny edges copes better with the low-contrast images – partly due to the fact that the objects cover most of the image area, so that even randomly distributed edge pixels will eventually trigger a detection with sufficient overlap.

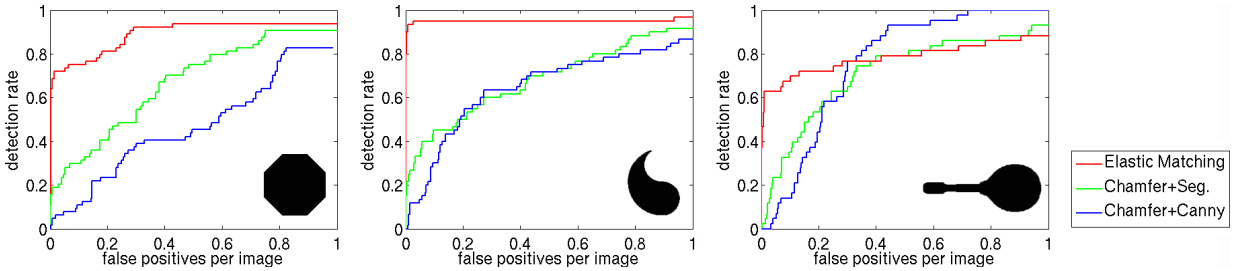


Figure 8: Detection results for Caltech images.

The inability to deal with cases, in which significant parts of the contour are missing, is inherent to a global approach. In order to overcome it, additional cues such as the shape or color of object parts are required. A plausible strategy seems to be a feedback loop ("hypothesize-and-verify"), which employs local cues to generate weak hypotheses, uses these hypotheses to guide a more sensitive search for discontinuities around the "hallucinated" contour parts, and uses the global object shape for verification. In fact a similar behavior can be seen in humans: for example, in the left most image of Figure 9, a quick glance only reveals the corpus of the instrument, and one has to actively search for the camouflaged fingerboard, in order to "complete" the mandolin and ascertain its class (and similar for the corpus in the second example). Further investigation of this issue is left for future work.

3.4 Localization accuracy

For all results reported so far we have used the same definition for a detection as in [15], in order to make results comparable: let $A(region)$ be the area of a regions bounding box, then a detection is correct, if $\frac{A(template \cap object)}{\max(A(template), A(object))} > 0.2$ (according to a personal communication with

Figure 9: Overly poor image contrast causes super-pixel segmentation to fail.

the authors). However, this threshold is rather generous, and classifies many cases as correct, in which the object is poorly localized, while one of the strengths of the proposed method is that the global contour shape enables accurate localization.

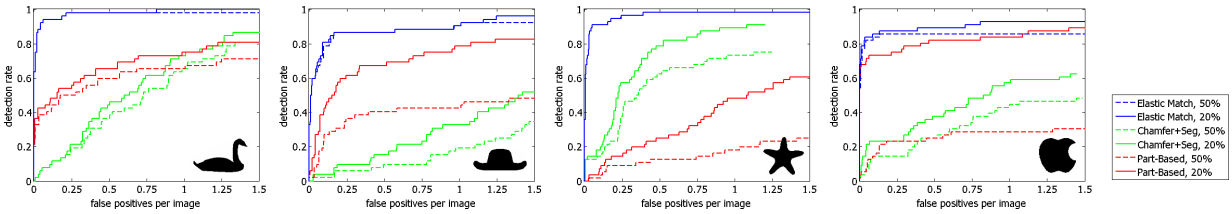


Figure 10: Localization accuracy of different methods.

To test the localization accuracy of different methods, we have therefore repeated the detection experiment of section 3.1 with a higher threshold of 50% overlap, which to us seems a more realistic definition of a successful localization. Figure 10 shows the results. It can be seen that the detection rate for the proposed method barely changes, confirming its good localization accuracy. On the contrary, the detection rate for chamfer matching is markedly lower with a tighter threshold (chamfer matching results are only given for super-pixel edges, which always outperformed Canny edges). The detection rate for the part-based method is much lower: often some parts are detected incorrectly – it seems that the shape constraints imposed by the star-graph are fine for highly distinctive appearance regions, but not strong enough to characterize the global shape of contours with some variation.

3.5 Influence of dissimilarity function

As explained earlier, given the tangent difference $\alpha = x'(u) - y'(v)$, there are many ways to define the dissimilarity function for matching points, and thus the matching probability $P_D(\alpha)$. The dissimilarity should be a symmetric, non-negative function in the interval $[-\pi, \pi]$, with $D(0) = 0$. Natural choices are

- $D = |\alpha|$, which leads to a (truncated) Laplace distribution $P_D(\alpha)$.
- $D = \alpha^2$, which leads to a (truncated) Normal distribution $P_D(\alpha)$.
- $D = (1 - \cos\alpha)$, which leads to a von Mises distribution, and would be the correct choice if the difference is considered to be the sum of a large number of random influences on the tangent direction.

Note that the the parameter B in $P_D(\alpha)$, which essentially defines the second moment (the "width") of the distribution, does not qualitatively change the matching: as shown by equations (7) and (8), changing B does not change the form of the cost function, but only the value of the stretch-cost R.

Choosing the *absolute differences*, i.e., the Laplace distribution, makes matching more robust, due to the heavier tail (this is similar to the robustness of the L_1-norm compared to the L_2-norm). The wrapped Normal and von Mises distributions are equivalent for our purposes, since in the 1-dimensional case they can be made to closely approximate one another, by setting the appropriate parameters [22]. We will therefore not consider them separately.

We have run the detection experiment with both the wrapped Laplace distribution (our standard setting also used in all other experiments), and with the wrapped normal distribution, in order to compare the functions. The stretch-cost has been set to the optimal value for each version. Experimentally, the difference between the two distance functions is small, with the absolute differences performing slightly better. On the whole the results seem to suggest that the sampled and smoothed tangent angles are already a robust representation of the contour shape, so that there is not much room for improvement using the L_1-norm rather than the L_2-norm. The detection rates for both dissimilarity functions are depicted in Figure 11.

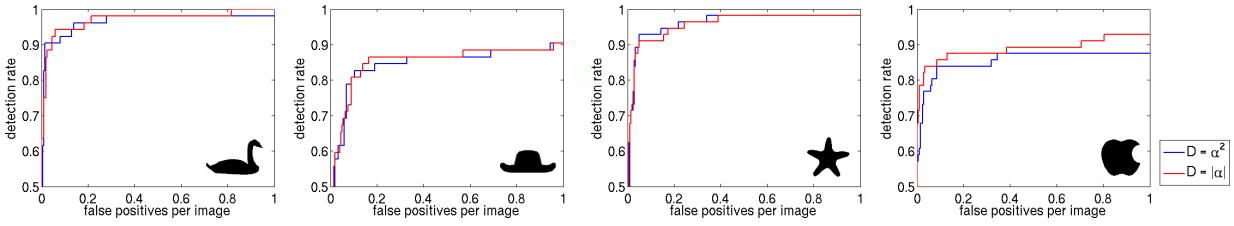


Figure 11: Detection performance with different distance functions.

3.6 Influence of stretch-cost

Another issue is how to determine the appropriate stretch-cost to penalize deformations of the template shape. It depends on several class-specific properties in a rather complex way:

- if a class has a unique shape, one can afford a lower stretch-cost, since even comparatively strong deformations will not render the contour ambiguous. For example, the *swan* class needs a low stretch-cost: the shape is unique, in the sense that it is rather uncommon and not many other scene parts will exhibit a similar contour pattern. On the contrary, the *hat* class requires a high stretch-cost, because elongated blobs with a protrusion occur frequently in the contour network.

- if a class has wide intra-class variability (such as for example *swans*), it requires large deformations and hence low stretch-cost, while other classes have only little variability, and one will select a higher stretch-cost to avoid false detections (for example *stopsigns*).

- if a class template has few inflection points, and sharp creases, then there is less chance that it can be stretched to fit an arbitrary contour, hence a low stretch-cost suffices (for example, the *applelogo* template does not require a high stretch-cost, because it can never be made to match any contour with more than one indentation). On the contrary, if there are many inflection points, and the tangent changes smoothly along the contour, a higher stretch-cost is required, because it can be deformed to approximately fit almost any shape (for example the *starfish*).

Since the stretch-cost R is a class-specific property, it needs to be determined separately for each class from examples, and it is interesting to see how sensitive the method is to the value of R. We have empirically investigated this question by repeatedly running the method on the same data with different values, while all other parameters were kept constant. Results are displayed in Figure 12. As a general trend we remark that there is a reasonable range of values, within which the exact setting is not critical (e.g., $R \in [0.2, 0.6]$ for *swans*, and $R \in [0.6, 1.0]$ for *hats*). Very low settings close to $R = 0$ are not satisfactory even for very unique shapes like *swans* or *yinyang*.

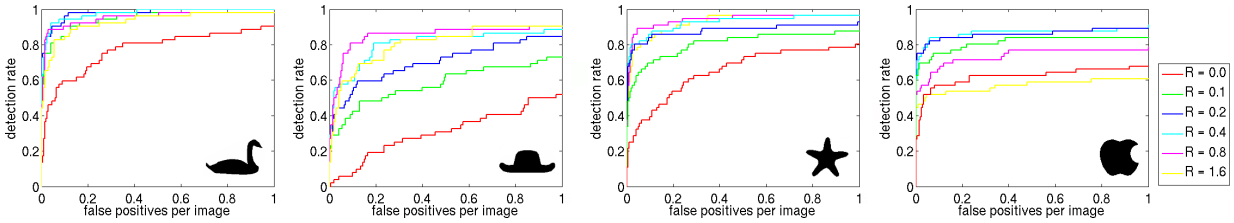


Figure 12: Detection performance with varying stretch-cost.

3.7 Influence of contour sampling

Another parameter required for the method is the number of vertices for the polygonal approximation of a contour. While in principle denser sampling should give higher accuracy, there are two arguments against overly fine quantization: firstly, the number of vertices should not be larger than the number of pixels of the original image contour, to avoid up-sampling artifacts which may distort the computation of tangent angles; and secondly, the computational cost of dynamic programming grows quadratically with the number of vertices, making too fine sampling prohibitively slow.

Again, we have repeated the detection experiment with different settings in order to empirically determine the effect of the numbers of vertices. Detection results are shown in Figure 13. Again, there is a certain tolerance region within which the exact choice does not matter. Our dataset covers a wide range of scales, with object contour lengths ranging from approximately 125 to 2000 pixels. For this data, 100 points proved to be enough to accurately represent all tested contours, while smaller numbers performed significantly worse. Note that the optimal stretch-cost depends on the number of vertices. If fewer vertices are sampled from the same contour, then a local stretch in one vertex has a larger influence on the total cost. This is mostly compensated by the fact that the stretch in that case *should* be more expensive, since it corresponds to a larger shape deformation. However, larger vertex numbers tend to require slightly higher stretch-cost for optimal performance. The curves in Figure 13 have been computed using for each sample number the stretch-cost which obtained the best performance.

3.8 Influence of curvature

In section 2.2 we have argued that tangent angles are more suitable than curvatures as shape descriptor. In this section, we support this claim by an experimental evaluation. The detection experiment was run several times, using the following descriptors: only tangent angles, only

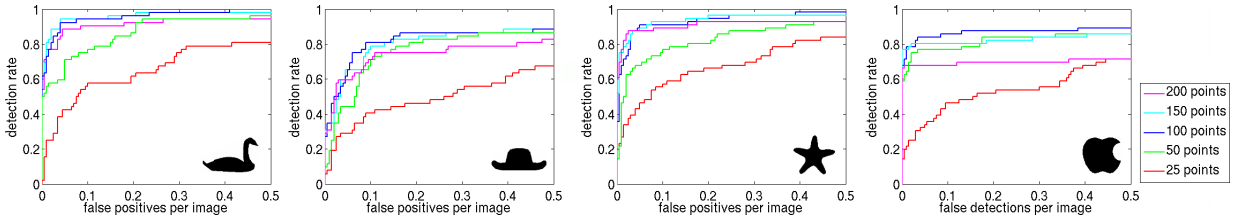


Figure 13: Detection performance with varying number of sampled contour points.

curvatures, an unweighted sum of tangent angle differences and curvature differences, and a weighted linear combination with weights $w_{tang} : w_{curv} = 4 : 1$.

Results are shown in Figure 14, and in Table 2. Tangent angles clearly outperform curvatures. When combined together with equal weights, the performance is still significantly lower than that of tangents alone, and the shape of the curve seems to indicate that adding curvatures only swamps the discriminative qualities of tangent angles: the two curves are roughly parallel over almost the whole range of thresholds, suggesting that adding curvature to the descriptor causes the algorithm to miss some correct detections, without reducing the amount of false positives. When combined such that tangent angles weigh 4 times more than curvature, the results are very similar to those obtained with tangent angles alone. Mostly, tangents alone are still slightly better, except for the *hat* class, for which the weighted combination performs best at some thresholds. With the little evidence we have at present, it is not possible to decide, whether curvature really improves the performance for less pronounced and more ambiguous shapes like the *hat*, or whether the effect is due to some unrecognized bias in the data set.

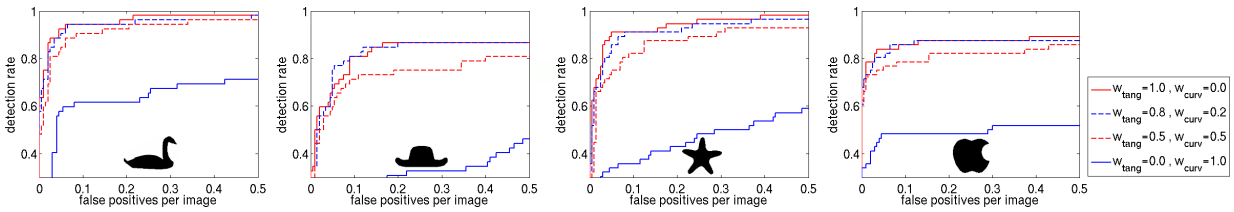


Figure 14: Detection performance for different combinations of tangent angle and curvature.

4 Discussion and Conclusion

We have revisited the problem of recognition and detection of object classes by their global shape. A probabilistic measure of shape similarity has been introduced, which can be efficiently computed with dynamic programming. The similarity measure has been integrated in a discrete

	w_{tang}	w_{curv}	Swan	Hat	Starfish	Apple	Avg.
0.2 FPPI	1.0	0.0	96	87	95	88	91
	0.8	0.2	94	87	91	88	90
	0.5	0.5	92	75	87	82	84
	0.0	1.0	62	31	43	48	46
0.4 FPPI	1.0	0.0	98	87	98	89	93
	0.8	0.2	96	87	96	88	92
	0.5	0.5	96	81	93	84	88
	0.0	1.0	69	35	54	52	52

Table 2: Comparison of detection rates for different combinations of tangent angle and curvature.

optimization framework based on merging super-pixels, in order to detect objects, which have high similarity with a template shape. Using only a single shape template per class, the method is robust to scaling, rotation, and some non-linear elastic deformation. In a detailed experimental evaluation, it has been shown to outperform previous methods, and has achieved a detection rate of 91% at 0.2 false positives per image for a challenging real-world data set. Somewhat surprisingly, it has been found that even if the object contour is not accurately recovered, robust global matching can often detect the object, because its characteristic parts (such as the neck and head of a swan) are still accurate.

Processing time depends on the image size, and on the number of super-pixels, which in turn depends on the image content. With our current MatLab/C implementation, the complete detection process for a new image takes on average 16 seconds for the *swans* (average image size $\approx 480 \times 320$ pixels), and 36 seconds for the *applelogos* (average size $\approx 640 \times 480$ pixels).

The main limitation of the presented method is the inherent inability of global methods to deal with significant occlusions. In this context the question arises, how global shape as a cue fits into a more general approach to object detection. Naturally we do not claim that global shape should replace other object descriptions, but we do believe that it has an important role for recognition and detection. Given the varying properties of different object classes, a generic object detection/recognition system will have to combine global boundary shape, local shape of typical parts, and local as well as global appearance (including color), and context. This will require weights for the different cues, depending both on the object class (for example contour and context are strong cues for hats, but appearance is more important for faces) and on the

environment conditions (for example the usefulness of color depends on the lighting).

Another natural limitation is that the method cannot deal with extreme viewpoint changes (although rotation and elastic deformation do give it some robustness). This difficulty is shared by other 2D recognition methods, and can be overcome by modeling an object class by a collection of 2D models for different viewpoints, e.g. [35]. Note however that for some viewpoints, shape may not be a strong cue (for example, a hat seen from above appears as an unspecific blob).

References

[1] E. M. Arkin, L. P. Chew, D. P. Huttenlocher, K. Kedem, and J. S. B. Mitchell. An efficiently computable metric for comparing polygonal shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):209–216, 1991. 7

[2] R. Basri, L. Costa, D. Geiger, and D. Jacobs. Determining the similarity of deformable shapes. *Vision Research*, 38:2365–2385, 1998. 7

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002. 3

[4] A. C. Berg, T. L. Berg, and Jitendra Malik. Shape matching and object recognition using low distortion correspondence. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, San Diego, California*, 2005. 4

[5] E. Borenstein, E. Sharon, and S. Ullman. Combining top-down and bottom-up segmentation. In *Proc. IEEE Workshop on Perceptual Organization in Computer Vision*, 2004. 6

[6] G. Borgefors. Hierarchical chamfer matching: a parametric edge matching algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):849–865, 1988. 3

[7] M. C. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *Proc. 5th European Conference on Computer Vision, Freiburg, Germany*, 1998. 3

[8] T. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models – their training and application. *Computer Vision and Image Understanding*, 61(1), 1995. 3

[9] G Cortelazzo, G. A. Mian, G. Vezzi, and P. Zamperoni. Trademark shapes description by string-matching techniques. *Pattern Recognition*, 27(8):1005–1018, 1994. 10

[10] D. Cremers, C. Schörr, and J. Weickert. Diffusion-snakes: Combining statistical shape knowledge and image information in a variational framework. In *Proc. Workshop on Variational and Levelset Methods*, 2001. 3

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, San Diego, California*, pages 886–893, 2005. 4

[12] A. Del Bimbo and P. Pala. Visual image retrieval by elastic matching of user sketches. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):121–132, 1997. 3

[13] R. Fagin and L. Stockmeyer. Relaxing the triangle inequality in pattern matching. *International Journal of Computer Vision*, 30(3):219–231, 1998. 10

[14] P. F. Felzenszwalb. Representation and detection of deformable shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(2):208–220, 2005. 3

[15] V. Ferrari, T. Tuytelaars, and L. Van Gool. Object detection by contour segment networks. In *Proc. 9th European Conference on Computer Vision, Graz, Austria*, volume 3, pages 14–28, 2006. 4, 15, 16, 18, 19, 20

[16] D. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *Proc. 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 87–93, 1999. 3, 5

[17] F. Glover and M. Laguna. Tabu search. In C. Reeves, editor, *Modern Heuristic Techniques for Combinatorial Problems*. Blackwell Publishers, 1993. 13

[18] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *Proc. 10th International Conference on Computer Vision, Beijing, China*, 2005. 6

[19] L. J. Latecki and R. Lakämper. Shape similarity measure based on correspondence of visual parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1185–1190, 2000. 11

[20] D. Lowe. Object recognition from local scale-invariant features. In *Proc. 7th International Conference on Computer Vision, Kerkyra, Greece*, pages 1150–1157, 1999. 3

[21] S. Manay, D. Cremers, B.-W. Hong, A. J. Yezzi, and S. Soatto. Integral invariants for shape matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1602–1618, 2006. 10

[22] K. V. Mardia. *Statistics of directional data*. Academic Press, Inc., 1972. 22

[23] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004. 5

[24] W. Niblack, W. Barber, M. Equitz, M. Flickner, E. Glasman, D. Petkovic, and P. Yanker. The QBIC project: querying images by content using color, texture and shape. In *Proc. SPIE Conference on Storage and Retrieval of Image and Video Databses*, 1993. 11

[25] R. Nock and F. Nielsen. Statistical region merging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1452–1458, 2004. 6, 15

[26] Y. Ohta and T. Kanade. Stereo by intra- and inter-scanline search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(2):139–154, 1985. 11

[27] C. Olson and D. Huttenlocher. Automatic target recognition by matching oriented edge pixels. *IEEE Transactions on Image Processing*, 6(1):103–113, 1997. 3

[28] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *Proc. 9th European Conference on Computer Vision, Graz, Austria*, volume 2, pages 575–588, 2006. 4

[29] B. C. Russell, A. A. Efros, J. Sivic W. T., Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, New York*, 2006. 6

[30] H. Sakoe and S. Chiba. Dynammic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978. 11

[31] E. Sali and S. Ullman. Combining class-specific fragments for object classification. In *Proc. 10th British Machine Vision Conference, Nottingham, UK*, pages 203–213, 1999. 3

[32] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. In *Proc. 6th International Conference on Computer Vision, Bombay, India*, pages 1154–1160, 1998. 6

[33] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. In *Proc. 10th International Conference on Computer Vision, Beijing, China*, volume 1, pages 503–510, 2005. 4

[34] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. 9th International Conference on Computer Vision, Nice, France*, 2003. 3

[35] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool. Towards multi-view object class detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, New York*, 2006. 27

[36] R. C. Veltkamp. Shape matching: Similarity measures and algorithms. Technical Report UU-CS-2001-03, Institute of Information and Computing Sciences, Utrecht University, 2001. 10