Shape Matching and Object Recognition

Alexander C. Berg and Jitendra Malik

U.C. Berkeley
{aberg,malik}@cs.berkeley.edu
http://www.cs.berkeley.edu/Research/Projects/vision

Abstract. We approach recognition in the framework of deformable shape matching, relying on a new algorithm for finding correspondences between feature points. This algorithm sets up correspondence as an integer quadratic programming problem, where the cost function has terms based on similarity of corresponding geometric blur point descriptors as well as the geometric distortion between pairs of corresponding feature points. The algorithm handles outliers, and thus enables matching of exemplars to query images in the presence of occlusion and clutter. Given the correspondences, we estimate an aligning transform, typically a regularized thin plate spline, resulting in a dense correspondence between the two shapes. Object recognition is handled in a nearest neighbor framework where the distance between exemplar and query is the matching cost between corresponding points. We show results on two datasets. One is the Caltech 101 dataset (Li, Fergus and Perona), a challenging dataset with large intraclass variation. Our approach yields a 45% correct classification rate in addition to localization. We also show results for localizing frontal and profile faces that are comparable to special purpose approaches tuned to faces.

1 Introduction

The problem of visual object recognition is really a family of inter-related problems. If we consider spatial extent, the notion of "object" extends downwards to parts (faces, eyes, propellers, wings), and upwards to scenes (kitchens, cityscapes, beaches). On the generalization dimension, we have categorization at varying levels all the way to identification of individuals (mammals,primates, humans, "Fred"). Sometimes, even the term "object" is questionable, when we consider visual recognition of materials such as sand or cornflakes.

What computational architecture would support a solution to all these problems in a common framework? In addition to the functional requirements above, processing must be fast, a large number of categories need to be handled, and the approach should be trainable with very few examples.

We propose a three stage architecture:

– *Initial Retrieval:* Retrieving a shortlist of potentially matching models for a query image based on feature descriptors. At this stage the spatial configuration of the feature locations can be ignored in order to facilitate rapid indexing.

- *Shape Matching:* Aligning template views of stored exemplars to the support of an unknown object in the query image. In face recognition this would be the stage where the corners of eyes, nose, lips and other landmarks would be “lined up”.
- *Discriminative Classification:* Given alignments of models to images we can now compare corresponding features as well as their configurations. Discriminative classifiers can give more weight to the characteristics that best distinguish examples of one category from other related categories.

This chapter outlines a solution to the second of these stages, shape matching. Since we wish to deal with intra-category variability, all shape matching is for us necessarily deformable shape matching. Back in the 1970s, at least three different research groups working in different communities initiated such an approach: in computer vision, Fischler and Elschlager [12], in statistical image analysis, Grenander ([14]and earlier), and in neural networks, von der Malsburg ([17] and earlier). The core idea that related but not identical shapes can be deformed into alignment using simple coordinate transformations dates even further back, at least to D’Arcy Thompson, in the 1910’s with *On Growth and Form* [34].

The basic subroutine in deformable matching takes as input an image with an unknown object (shape) and compares it to a model by first aligning the two and then computing a *similarity* based on both the aligning *transform* and the *residual difference* after applying the aligning transformation. Searching for an alignment can be quite difficult. We show that the search can be approximated by an easier discrete matching problem, *the correspondence problem*, between key points on a model and a novel object.

Practically speaking, the basic difficult question for the correspondence problem is, “How do we algorithmically determine which points on two shapes correspond?” The correspondence problem in this setting is more difficult than in the setting of binocular stereopsis, for a number of reasons:

1. Intra-category variation: the aligning transform between instances of a category is not a simple parameterized transform. It is reasonable to assume that the mapping is smooth, but it may be difficult to characterize by a small number of parameters as in a rigid or affine transform.
2. Occlusion and clutter: while we may assume that the stored prototype shapes are present in a clean, isolated version, the shape that we have to recognize in an image is in the context of multiple other objects, possibly occluding each other.
3. 3D pose changes: since the stored exemplars represent multiple 2D views of a 3D object, we could have variation in image appearance which is purely pose-related, the 3D shapes could be identical

The principal contribution of this work is a novel algorithm for solving the correspondence problem for shape matching.

We represent shape by a set of points sampled from contours on the shape. Typically 50-100 pixel locations sampled from the output of an edge detector are used; as we use more samples we get better approximations. Note that there is nothing special about these points – they are *not* required to be keypoints such as those found using a Harris/Forstner type of operator or scale-space extrema of a Laplacian of Gaussian operator, such as used by Lowe [21].

We exploit three kinds of constraints to solve the correspondence problem between shapes:

1. Corresponding points on the two shapes should have similar local appearance. For this purpose we develop geometric blur to measure rough shape similarity.
2. Minimizing geometric distortion: If i and j are points on the model corresponding to i' and j' respectively, then the vector from i to j, $\boldsymbol{r}_{ij}$ should be consistent with the vector from i' to j', $\boldsymbol{r}_{i'j'}$. As examples: If the transformation from one shape to another is a translation accompanied by pure scaling, then these vectors must be scalar multiples. If the transformation is a pure Euclidean motion, then the lengths must be preserved. etc.
3. Smoothness of the transformation from one shape to the other. This enables us to interpolate the transformation to the entire shape, given just the knowledge of the correspondences for a subset of the sample points. We use regularized thin plate splines to characterize the transformations.

The similarity of point descriptors and the geometric distortion is encoded in a cost function defined over the space of correspondences. We purposely construct this to be an integer quadratic programming problem (cf. Maciel and Costeira [22]) and solve it using fast-approximate techniques.[1]

We address two object recognition problems, multi-class recognition and face detection. In the multiple object class recognition problem, given an image of an object we must identify the class of the object and find a correspondence with an exemplar. We use the Caltech 101 object class dataset consisting of images from 101 classes of objects: from accordion to kangaroo to yin-yang, available at [7]. This dataset includes significant intra class variation, a wide variety of classes, and clutter. On average we achieve **45%** accuracy on object classification with quite good localization on the correctly classified objects.

It is important to point out that these results are achieved with a simple generative model based solely on coarse shape. Better recognition can be achieved by building class specific discriminative models combining shape with other cues such as color and texture. The point here is experimental evidence of simple generative shape models proving useful for both localization and recognition.

We also consider face detection for large faces, suitable for face recognition experiments. Here the task is to detect and localize a number of faces in an image. The face dataset we use is sampled from the very large dataset used in [6] consisting of news photographs. With only 20 exemplar faces our generic system provides a ROC curve with slightly better generalization, and slightly worse false detection rate than the quite effective specialized face detector of Mikolajczyk [24] used in [6].

[1] It is worth noting that this formulation is amenable to various probabilistic models, maximum likelihood estimation for a product of Gaussians among others, but we do not address this further here.

2 Shape Descriptor

A simple descriptor based on blurred edge maps is used to compare shapes locally. These descriptors do not cover an entire object and so we refer to them as local shape descriptors, nevertheless they have a broader spatial support than typical local descriptors. The consequence of broad spatial support is variation between views of similar local structures. We use a spatially varying *geometric blur* over the positions of edges to provide the necessary robustness.

Descriptors are computed using geometric blur on edge maps. Oriented edge maps are computed and then blurred. Keypoints are then located along edges and sample points are drawn from the blurred edge maps. The vector of these samples is the descriptor as shown in Figure 8.

Geometric blur is an average over geometric transformations of a signal representing the spatial distribution of features in an image. The objective of the averaging is to make comparison of signals robust to typical geometric distortions within a bounded range. We will use a descriptor based on geometric blur to evaluate similarity between shapes.

First we motivate basing local shape descriptors on edge maps and the need for spatial uncertainty. This is followed by the mathematical definition of geometric blur. Two motivations for using a simple family of blurs — linearly increasing blur with distance from the center feature point — are presented, and a relatively low dimensional descriptor based on geometric blur is defined. A brief comparison to alternate descriptors concludes this section.

2.1 Motivation

The two helicopters shown in Figure 1 are easily recognizable as helicopters and a young child could indicate positions for the nose and tail of each. The crops below indicate the difficulty faced by a computer. Analogous structures in the images are only very roughly similar. In order to find a correspondence and then an alignment between the two objects it is necessary to find some way to get at this rough similarity. We approach this problem by first representing the rough spatial pattern of edges.

2.2 Simple Example

Before beginning the formal development of geometric blur a simple example of comparing distorted signals is presented to make concrete some of the mathematics to follow. Here we begin to show how geometric blur can provide robustness to spatial variation.

In Figure 2, which signal, A or C, is most similar to the signal B? The question is ambiguous, and we need to take into consideration some type of accepted variation, say small affine transformations. Note that here we mean spatial affine transformations, not transforms in intensity. Making robust comparison of signals with variation in intensity is rather better studied than the variation in signals due to distortions in geometry. Even with this added information, the

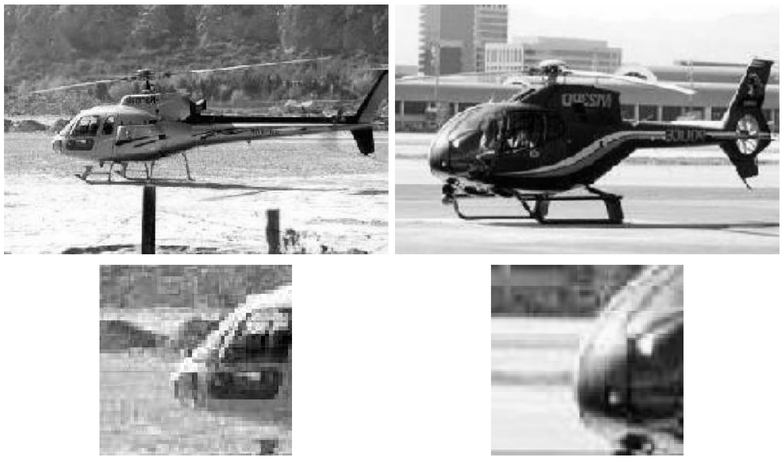


Fig. 1. In the **top row** are two images showing similar objects, helicopters. The **bottom row** shows that the local structure of the objects is only very roughly similar. Geometric blur is motivated by the goal of identifying rough similarity between bits of shapes.

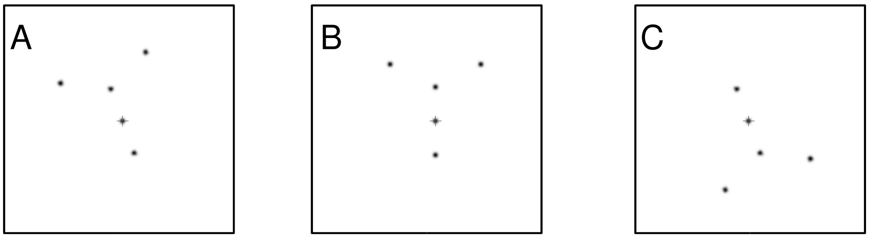


Fig. 2. Three similar signals composed of impulses. They represent the spatial location of features in an image. The goal is to recognize that a small transformation brings **A** and **B** into alignment, but not so for **B** and **C**.

correlation between either the left (A & B) or right (B & C) pair of signals is low and quite similar, providing no information about which are more similar. This can be seen in the first row of Figure 3 where the insets show the point-wise products of the signals on either side. Note that smoothing the signals with a uniform Gaussian does not quite solve the problem, as can be seen in the second row of the Figure 3. After blurring the signals with a uniform Gaussian the correlation between either pair of signals is similar, missing the clear differences. The basic idea of geometric blur is to blur or average the signals over the range of acceptable transformations (small affine transformations in this case), as shown in the third row of Figure 3. This will turn out to be mathematically equivalent to convolving the signal with a spatially varying kernel. Roughly speaking,

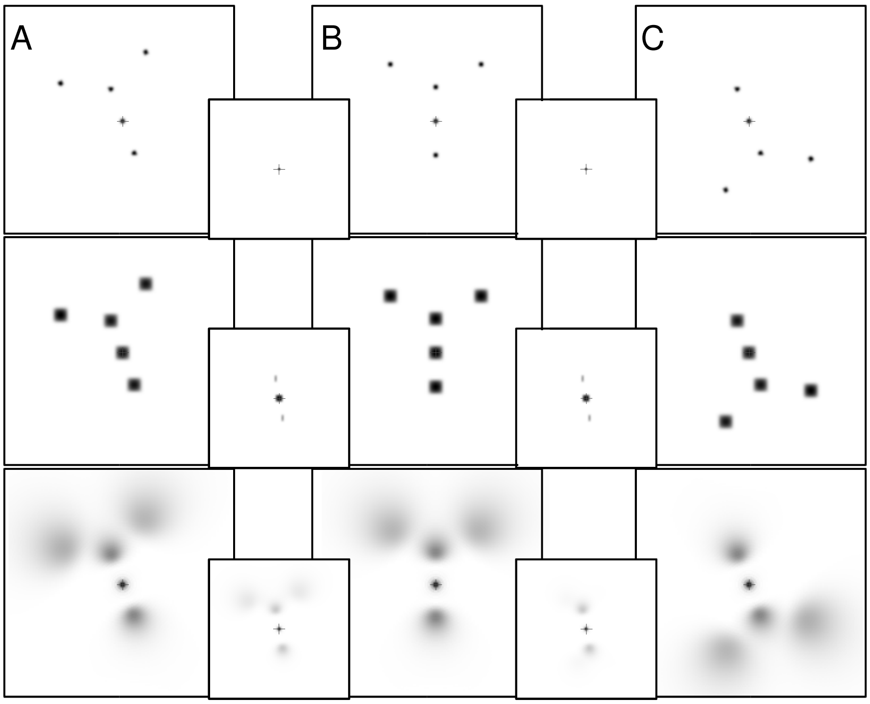


Fig. 3. The **top row** shows three signals, A, B, and C. The **top row insets** show the point-wise product of the signals on either side, each results in correlation 0.2. the **second row** shows the result of applying a Gaussian blur to the signals. Note that more context is now included, but the correlations are still equal (0.22). The **third row** shows the result of applying geometric blur, a spatially varying blur replicating the effect of averaging over small affine transforms of the signal. Now the insets indicate a difference between the correlations: 0.63 for the correct match versus 0.4 for the incorrect match.

parts of the signal farther from the center are blurred more because they have the opportunity to move more. After this type of blur, correlation can correctly identify the more similar pair, as can be seen on the bottom row of Figure 3.

2.3 Definition

We define geometric blur and show that it can be written as a spatially varying convolution.

The geometric blur $GB_I(x)$ of a signal $I(x)$ over coordinate x is the integral over a range of distorted versions of the signal:

$$GB_I(x) = \int_T I(T(x))d\mu \tag{1}$$

Where T are spatial transforms and μ is a measure on the space of transforms. Under appropriate conditions[2] there is a density ρ such that:

$$GB_I(x) = \int_p I(T_p(x))\rho\,(T_p)\,dp \tag{2}$$

Where T_p is a transform specified by parameters p in $\mathbb{R}^k$ and the integral is computed with respect to the Lebesgue measure on $\mathbb{R}^k$. The density ρ is determined by the measure on transforms. In order to reduce notational clutter we will usually drop the subscript p and assume that the transform T is parameterized by p.

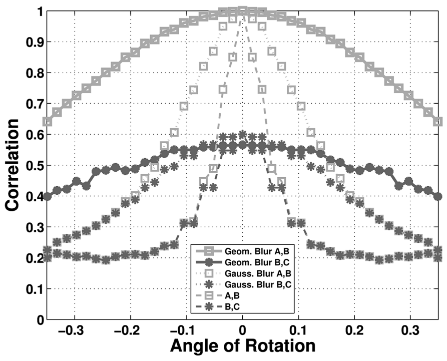


Fig. 4. Correlation of signal B and rotated (A) or rotated and flipped (C) versions of itself with no blur, uniform blur, or geometric blur. The far right end of the graph, rotation by 0.34 radians, corresponds to the signals shown in Figure 3.

Equation 1 is an integration over warped versions of the signal. We rewrite this to integrate over the range (spatial coordinates of I) of the transforms and change variables:

$$GB_I(x) = \int_z I(z) \int_{T:T(x)==z} \rho\,(T) d\tilde{p}dz \tag{3}$$

$$= \int_y I(x-y) \int_{T:(x-T(x))==y} \rho\,(T)\,d\tilde{p}dy \tag{4}$$

$$= \int_y I(x-y) K_x(y) dy \tag{5}$$

The geometric blur is then a convolution with a spatially varying kernel, $K_x(y) = \int_{T:(x-T(x))==y} \rho\,(T) d\tilde{p}$.[3]

[2] Additional details and derivations concerning geometric blur, including motivation as an approximation to Bayesian estimation, can be found in [4].

[3] In Equations 3 and 4 $d\tilde{p}$ indicates integration with respect to the measure on the "slice", $\{T : T(x) == z\}$ and $\{T : x - T(x) == y\}$ respectively.

An Example: Returning to the example signals in Figures 2 and 3. We now consider comparing signal **B** to rotations of itself, and rotations of its vertical mirror image. The green dashed lines in Figure 4 show the correlation between **B** and rotated versions of itself, and the red dashed line shows correlations between **B** and rotated versions of its vertical mirror image. As a reference, the signals shown in Figure 2 would correspond to the signals used for a rotation of 0.35 radians as shown on the far right of Figure 4.

In this and all other examples in this section the kernel function is $K_x(y) = f(\alpha|x| + \beta)G_{\alpha|x|+\beta}(y)$, where G is a Gaussian with the specified standard deviation, and f is a normalization factor so that the K_x is L^2 normalized.

2.4 Empirical Measurement for Blur

By construction geometric blur with the kernel used above is appropriate in the case of signals undergoing small affine distortions. In general if we have enough examples of patches that are known to correspond we can actually find an optimal blur pattern. We illustrate this with an example using wide base-line stereo pairs.

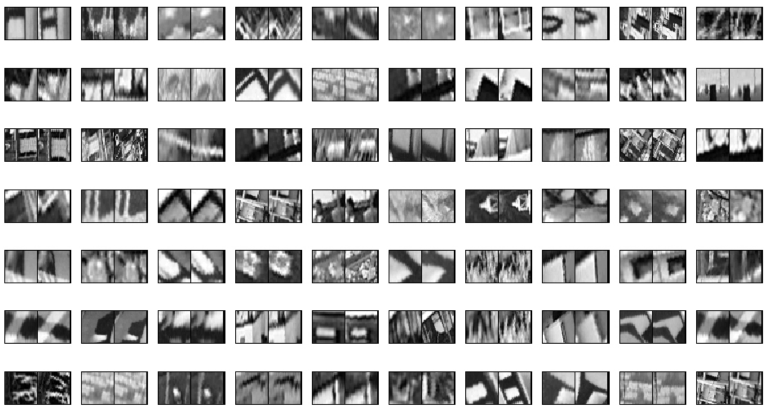


Fig. 5. Rectified paired patches found by the Harris-Affine detector. Note that the centers of the patches are usually on edges or at corners, and that the orientations and scales of matched patches are often slightly different.

We use images from different camera positions looking at the same object or scene[4]. The correspondence between images is known. A region of interest operator is applied, and where it works correctly, producing corresponding regions on images, the corresponding patches are used. Figure 5 shows pairs of corresponding patches. The raw patches are replaced by edge maps and the covariance between corresponding patches of edge maps is shown in Figure 6. Each

[4] Images are from work by Mikolajczyk and Schmid [25] on region of interest operators and descriptors for wide-baseline matching.

small block in the figure represents the covariance of all the pixels in one patch of edge map with respect to a particular pixel in the corresponding patch of edge map. The general structure shows a more concentrated covariance near the center, and a more diffuse covariance near the periphery. Plotting this shows the nearly linear pattern in Figure 7. While these examples support the linearly increasing blur kernel, they are restricted to images of the same object or scene. Replicating this study on images of categories of objects it is necessary to find correspondences in the face of intra-category variation.

2.5 Descriptor

Creating a descriptor using geometric blur involves design choices for the region of interest operator, underlying features, blur kernel, and subsampling.

Region of Interest Operator. Descriptors and region of interest operators are the head and tail respectively of a thorny beast indeed. The two are coupled

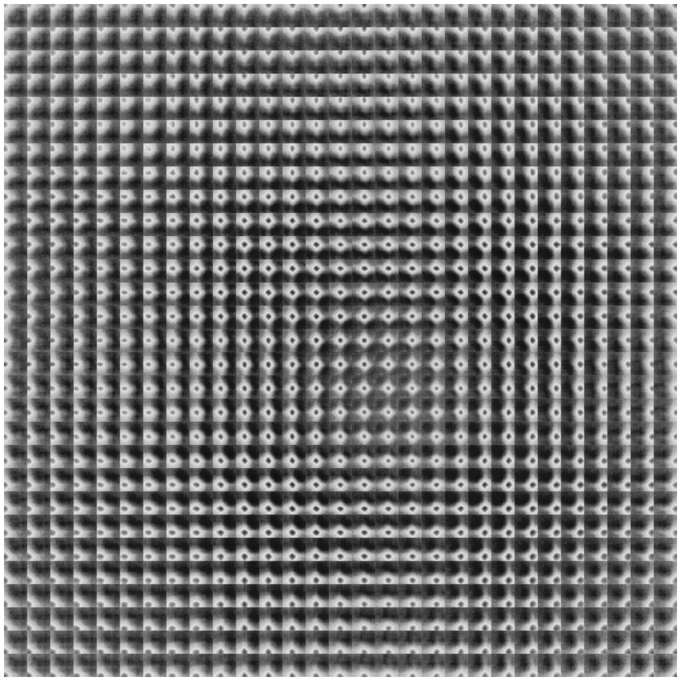


Fig. 6. Covariance of edge response between corresponding patches using a Harris-Affine detector. These have been reshaped so that each small block represents the covariance of all the pixels in one patch of edge map with respect to a particular pixel in the corresponding patch of edge map. The location of the small block specifies the pixel in the corresponding patch of edge map. For example the block at the lower right of the image shows the covariance of the all the pixels in a patch of edge map with the pixel in the lower right of the corresponding patch of edge map.

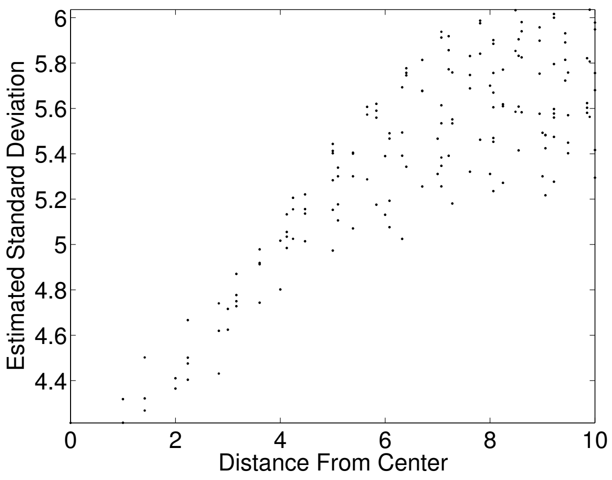


Fig. 7. Results of fitting Gaussians to the blur patterns shown in Figure 6 of covariance of edge response between corresponding patches. The estimated standard deviation is plotted against the distance from the center. The amount of blur in the covariance increases almost linearly.

because the choice of interest point operator effects the type of variation the descriptor must tolerate [4].

One benefit of the spatially varying blur is that geometric blur can be used for localization. The work by Berg & Malik [5] concentrates mainly on this aspect of geometric blur. This is quite different from other contemporary descriptors such as SIFT [20] that rely on an interest point operator to select similar locations for potential matches. As a result a somewhat promiscuous interest point operator can be used in conjunction with geometric blur, and the localization of the best match can be left up to the descriptor itself. We will place interest points anywhere in an image where there is a strong edge response, using sampling with repulsion to spread interest points throughout the image.

Choosing the scale for the descriptor can also be a complex problem. In this case we duck the issue by tying the scale of the descriptor to the scale of the object model. This means that if the object scale varies, multiple sets of descriptors must be used. Luckily geometric blur is designed to handle affine distortion including scale, and tolerates scale variation relatively well. For instance the multi-category recognition results shown later use a single scale of descriptor despite variation in scale for some of the categories.

Feature Channels. Motivated by wide ranges of appearance we base the feature channels on a coarse scale edge detector. The best results are obtained using the boundary detector of [23]. This boundary detector is constructed not to respond to texture, and produces relatively consistent boundary maps. In addition a simple and computationally less expensive edge detector based on elongated

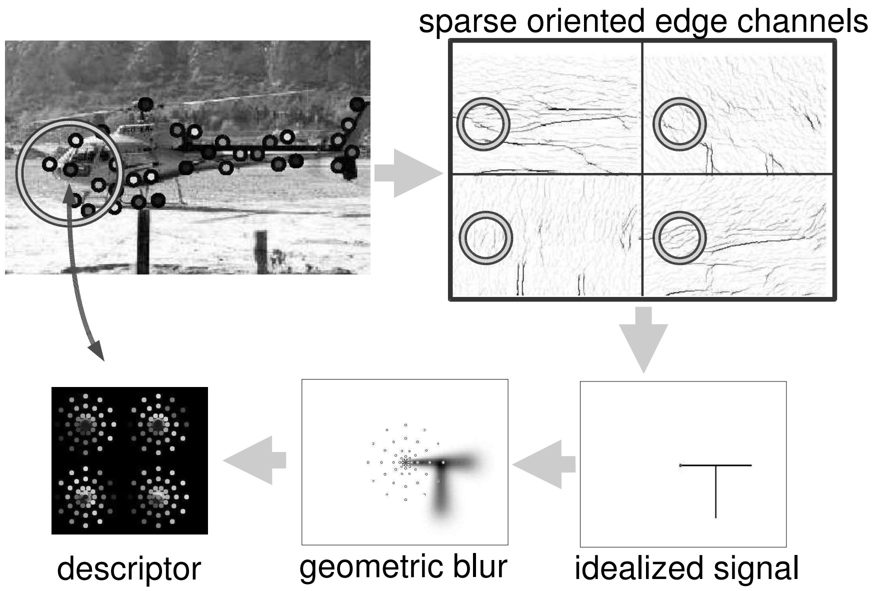


Fig. 8. The steps to compute a geometric blur descriptor. Starting with a feature point on an image (eg the point at the center of the circle in the **upper left**) and a region of interest (the circle). The sparse feature channels are cropped out as shown in the **upper right**. Geometric blur is applied to each channel (shown here with an idealized signal for clarity) and the signal is sub-sampled. The final descriptor is the vector of values indicated by dots with differing intensity at **lower left**.

derivative of Gaussian filters is used for comparison [27]. In both cases edge detection results are split up by orientation and oriented non-max suppression is applied producing multiple sparse channels as shown in Figure 8.

Blur Kernel. As before we use a simple blur kernel based on a Gaussian. If $G_a(x)$ is a Gaussian with standard deviation a then:

$$K_x(y) = G_{\alpha|x|+\beta}(y)$$

is our blur kernel. The kernel is normalized with respect to the L^2 norm.

Sub-sampling. The geometric blur of a signal should be sub-sampled using a pattern that matches the amount of blur introduced. In particular in the periphery fewer samples are required. For the kernel we consider above this implies a density of samples decreasing linearly with distance from the origin. The sampling pattern used in these experiments is shown in Figure 8.

A quick summary of steps for computing geometric blur descriptors for an image follows:

1. Detect feature locations: oriented edge detectors.
2. Choose interest points: random sampling with repulsion on points with high edge energy.

3. Compute multiple blurred versions of the channels: using the spatially varying Gaussian kernel described above.
4. Around each interest point, for each channel, sample points according to the dart-board pattern in Figure 8. These samples should be drawn from the appropriate blurred version of the channel.
5. These samples form the geometric blur descriptors.

The descriptors are compared using normalized correlation.

2.6 Comparison to SIFT

The geometric blur descriptor in this work is used to measure rough similarity in structure. There are currently a number of somewhat similar descriptors for local structure. We will use SIFT as an example to illustrate the differences.[5]

The first difference is the region of interest operator. SIFT is usually used in conjunction with a region of interest operator based on finding local maxima of a scale space operator based on the difference of Gaussians applied to pixel values. For views of the same object this works quite well, providing repeatable regions, but in the presence of intra-category variation this is no longer the case.[6] The region of interest operator we use is based simply on edge response, which is more repeatable across intra-category variation. The scale of the descriptor is tied to the object scale as described section 2.5. It is worth noting that in general the scale relative to the edge features is much *larger* that commonly found with the SIFT region of interest operator. This larger scale allows more context to be used.

The relatively large context of the geometric blur based descriptors requires more tolerance of change in the signal, which is accomplished by the radially increasing blur. One way to think of the SIFT descriptor is as constant blur with a grid subsampling pattern (4x4) instead of the dart-board pattern used for the geometric blur descriptor. As the relative size of the patch considered increases the difference between constant blur and geometric blur increasing linearly with distance becomes larger.

Finally the underlying features for the geometric blur based descriptor described in this chapter and SIFT are both based on oriented edge maps, with slightly different details in engineering. In particular the number of orientations, non-max suppression, and sometimes use of the *pb* detector from Martin *et al*[23].

[5] Shape contexts [3] are also quite similar in spirit to geometric blur based descriptors. The main differences are the hard decision about feature presence and location with shape contexts vs soft decision for both using geometric blur. Work on geometric blur introduced the connection between blur increasing linearly with distance and robustness to affine distortion, which was later used to justify the sampling pattern in shape contexts.

[6] It is important to note that some features will be reliably found even in the presence of intra-category variation, in order to find good alignment we require more matches, and so must tolerate more variation. The trick is to maintain discriminative information while being tolerant of more variation.

It is the smoothing and subsampling of the edge maps along with the region of interest operator where most differences arise.

Generally the SIFT type descriptors are suited to identifying parts of the same object from multiple views, while the geometric blur based descriptor described here is designed to evaluate potential similarity under intra-class variation exploiting larger support and spatially varying blur. There are many choices for descriptors, the experiments later in the chapter indicate that a generic geometric blur based descriptor fares well as part of a correspondence algorithm for a wide variety of object categories.

3 Geometric Distortion

Local shape similarity measurements are not sufficient to identify similar shapes. In order to combine local shape similarity measurements using geometric blur descriptors we need some way of measuring changes in the entire shape. This is accomplished be measuring the distortion in the configuration of feature points induced by a correspondence.

We consider correspondences between feature points $\{p_i\}$ in model image P and $\{q_j\}$ in image Q. A correspondence is a mapping σ indicating that p_i corresponds to $q_{\sigma(i)}$. To reduce notational clutter we will sometimes abbreviate $\sigma(i)$ as i', so σ maps p_i to $q_{i'}$.

The quality of a correspondence is measured in two ways: how similar feature points are to their corresponding feature points, and how much the spatial arrangement of the feature points is changed. We refer to the former as the match quality, and the later as the distortion of a correspondence.

We express the problem of finding a good correspondence as minimization of a cost function defined over correspondences. This cost function has a term for the match quality and for the geometric distortion of a correspondence: $\text{cost}(\sigma) = \omega_{\text{m}} C_{\text{match}}(\sigma) + \omega_{\text{d}} C_{\text{distortion}}(\sigma)$

Where constants ω_{m} and ω_{d} weigh the two terms. The match cost for a correspondence is:

$$C_{\text{match}}(\sigma) = \sum_i c(i, i') \tag{6}$$

Where $c(i, j)$ is the cost of matching i to j in a correspondence. We use the negative of the correlation between the feature descriptors at i and j as $c(i, j)$.

We use a distortion measure computed over pairs of model points in an image. This will allow the cost minimization to be expressed as an integer quadratic programming problem.

$$C_{\text{distortion}}(\sigma) = \sum_{ij} H(i, i', j, j') \tag{7}$$

Where $H(i, j, k, l)$ is the distortion cost of mapping model points i and j to k to l respectively. While there are a wide variety of possible distortion measures, including the possibility of using point descriptors and other features, in addition to location, we concentrate on geometric distortion and restrict ourselves to measures based on the two offset vectors $r_{ij} = p_j - p_i$ and $s_{i'j'} = q_{j'} - q_{i'}$.

$$C_{\text{distortion}}(\sigma) = \sum_{ij} \text{distortion}(r_{ij}, s_{i'j'}) \tag{8}$$

Our distortion cost is made up of two components:

$$C_{\text{distortion}}(\sigma) = \sum_{ij} \gamma \mathrm{d_a}(\sigma) + (1-\gamma)\mathrm{d_l}(\sigma) \tag{9}$$

$$d_a(\sigma) = \left(\frac{\alpha_d}{|r_{ij}| + \beta_d}\right) \left|\arcsin\left(\frac{s_{i'j'} \times r_{ij}}{|s_{i'j'}||r_{ij}|}\right)\right| \tag{10}$$

$$d_l(\sigma) = \frac{||s_{i'j'}| - |r_{ij}||}{|r_{ij}| + \mu_d} \tag{11}$$

where d_a penalizes the change in direction, and d_l penalizes change in length.[7] A correspondence σ resulting from pure scale and translation will result in $d_a(\sigma) = 0$, while σ resulting from pure translation and rotation will result in $d_l(\sigma) = 0$. The constants α_d, β_d, μ_d, are all terms allowing slightly more flexibility for nearby points in order to deal with local "noise" factors such as sampling, localization, etc. They should be set relative to the scale of these local phenomena. The constant γ weighs the angle distortion term against the length distortion term.

Outliers. Each point p_i, in P, is mapped to a $q_{\sigma(i)}$, in Q. This mapping automatically allows outliers in Q as it is not necessarily surjective – points q_j may not be the image any point p_i under σ. We introduce an additional point q_{null} and use $\sigma(i) = \text{null}$ to allow a point p_i to be an outlier. We limit the number of points p_i which can be assigned to q_{null}, thus allowing for outliers in both P and Q.

4 Correspondence Algorithm

Finding an assignment to minimize a cost function described by the terms in Equations 7 and 6 above can be written as an Integer Quadratic Programming (IQP) problem.

$$\text{cost}(x) = \sum_{a,b} H(a,b)x_a x_b + \sum_a c(a)x_a \tag{12}$$

Where the binary indicator variable x has entries x_a, that if 1, indicate $\sigma(a_i) = a_j$. We then have $H(a,b) = H(a_i, a_j, b_i, b_j)$, and $c(a) = c(a_i, a_j)$ from Equations 7 and 6.

[7] It is possible to construct a pairwise distortion measure based on bending energy which is compatible with the thin plate spline we use alter for interpolation [29], however we are interested in more structured transformations such as rotation and scaling, resulting in the simple distortion measure presented here.

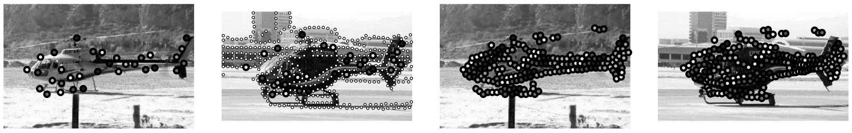


Fig. 9. An exemplar with a subset of feature points marked (**left**), the novel "probe" image with all feature points in white, and the feature points found to correspond with the exemplar feature points marked in corresponding colors (**left center**), the exemplar with all its feature points marked in color, coded by location in the image (**right center**), and the probe with the exemplar feature points mapped by a thin plate spline transform based on the correspondences, again colored by position in the exemplar (**far right**). See Figure 10 for more examples.

We constrain x to represent an assignment. Write x_{ij} in place of $x_{a_i a_j}$. We require $\sum_j x_{ij} = 1$ for each i. Furthermore if we allow outliers as discussed in Section 3, then we require $\sum_i x_{i\text{null}} \leq k$, where k is the maximum number of outliers allowed. Using outliers does not increase the cost in our problems, so this is equivalent to $\sum_i x_{i\text{null}} = k$. Each of these linear constraints are encoded in a row of A and an entry of b. Replacing H with a matrix having entries $H_{ab} = H(a,b)$ and c with a vector having entries $c_a = c(a)$. We can now write the IQP in matrix form:

$$\begin{aligned} \min \text{cost}(x) =& x'Hx + c'x \quad \text{subject to,} \\ & Ax = b, \quad x \in \{0,1\}^n \end{aligned} \tag{13}$$

4.1 Approximation

Integer Quadratic Programming is NP-hard, however specific instances may be easy to solve. We follow a two step process that results in good solutions to our problem. We first find the minimum of a linear bounding problem, an approximation to the quadratic problem, then follow local gradient descent to find a locally minimal assignment. Although we do not necessarily find global minima of the cost function in practice the results are quite good.

We define a linear objective function over assignments that is a lower bound for our cost function in two steps. First compute $q_a = \min \sum_b H_{ab} x_b$. Note that from here on we will omit writing the constraints $Ax = b$ and $x \in \{0,1\}^n$ for brevity.

If x_a represents $\sigma(i) = j$ then q_a is a lower bound for the cost contributed to any assignment by using $\sigma(i) = j$. Now we have $L(x) = \sum_a (q_a + c_a) x_a$ as a lower bound for $cost(x)$ from Equation 13. This construction follows [22], and is a standard bound for a quadratic program. Of note is the operational similarity to geometric hashing.

The equations for q_a and L are both integer linear programming problems, but since the vertices of the constraint polytopes lie only on integer coordinates, they can be relaxed to linear programming problems without changing the optima, and solved easily. In fact due to the structure of the problems in our setup they can be solved explicitly by construction. If n is the length of x, each problem takes $O(n)$ operations with a very small constant. Computing q_a for $a = 1 \dots n$ requires $O(n^2)$ time.

We then perform gradient descent changing up to two elements of the assignment at each step. This takes $O(n^2)$ operations per step, and usually requires a very small number of steps (we put an upper bound on the number of steps). In practice we can solve problems with $m = 50$ and $n = 2550$, 50 possible matches for each of 50 model points with outliers, in less than 5 seconds.

4.2 Example Correspondences

Given a model image P of an object, and a target image Q, possibly containing an instance of a similar object we find a correspondence between the images as follows:

1. Extract sparse oriented edge maps from each image.
2. Compute features based on geometric blur descriptors at locations with high edge energy.
3. Allow each of m feature points from P to potentially match any of the k most similar points in Q based on feature similarity and or proximity.
4. Construct cost matrices H and c as in Section 3.
5. Approximate the resulting Binary Quadratic Optimization to obtain a correspondence. Store the cost of the correspondence as well.
6. Extend the correspondence on m points to a smooth map using a regularized thin plate spline [28].

See Figures 9 and 10 for a number of examples. In the leftmost column of the figures is the image, P, shown with m points marked in color. In the middle left column is the target image Q with the corresponding points found using our algorithm. A regularized thin plate spline is fit to this correspondence to map the full set of feature points on the object in P, shown in the middle right column, to the target, as shown on the far right column. Corresponding points are colored the same and points are colored based on their position (or corresponding position) in P – in P colors are assigned in uniform diagonal stripes, the distortion of these striped in the far right column of the figure gives some idea of the distortion in the correspondence.

5 Object Recognition

The Caltech 101 [10] dataset consists of images from 101 categories of objects: from accordion to kangaroo to yin-yang[8]. Example images from 100 of the categories can be seen in Figure 13. There are a wide variety of image categories: man-made objects, animals, indoor images and outdoor images, drawings, etc.

[8] Available from `http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html`

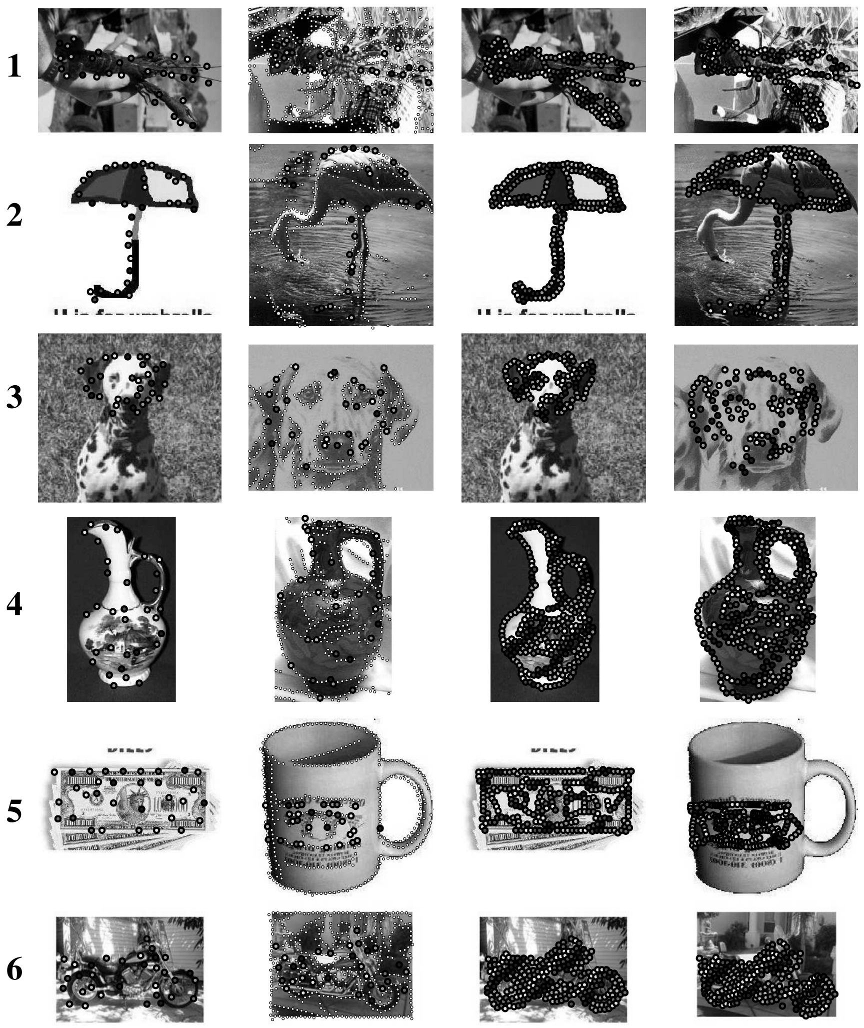


Fig. 10. Each row shows an alignment found using our technique described in section 4. Leftmost is an exemplar with some feature points marked. Left center is a probe image with the correspondences found indicated by matching colors (all possible feature matches are shown with white dots). All of the feature points on the exemplar are shown center right, and their image using a thin plate spline warp based on the correspondence are shown in the right most image of the probe. Note the ability to deal with clutter (1,6), scale variation(3), intraclass variation (all), also the whimsical shape matching (2), and the semiotic difficulty of matching a bank note to the image of a bank note painted on another object (5).

In addition many of the images have background clutter. There are up to 800 images in a category, although many categories contain 50 or fewer images. Some categories offer more variation and clutter than others.

6 Recognition Experiments

Our recognition framework is based on nearest neighbors.

Preprocessing: For each object class we store a number of exemplars, possibly replicated at multiple scales, and compute features for all of the exemplars. Features are only computed on the support of the objects. At this point object supports are marked by hand. Section 9 shows how to find them automatically.

Indexing: Extract features from a query image. For each feature point in an exemplar, find the best matching feature point in the query based on normalized correlation of the geometric blur descriptors. The mean of these best correlations[9] is the similarity of the exemplar to the query. We form a shortlist of the exemplars with highest similarity to the query image.

Correspondence: Find a correspondence from each exemplar in the shortlist to the query as described above. Pick the exemplar with the least cost.

We address two object recognition problems, multi-class recognition and face detection. In the multiple object class recognition problem, given an image of an object we must identify the class of the object and find a correspondence with an exemplar. We use the Caltech 101 object class dataset consisting of images from 101 classes of objects: from accordion to kangaroo to yin-yang, available at [7]. This dataset includes significant intra class variation, a wide variety of classes, and clutter. On average we achieve **45%** accuracy on object classification with quite good localization on the correctly classified objects.

We also consider face detection for large faces, suitable for face recognition experiments. Here the task is to detect and localize a number of faces in an image. The face dataset we use is sampled from the very large dataset used in [6] consisting of news photographs collected from yahoo.com. With only 20 exemplar faces our generic system provides a ROC curve with slightly better generalization, and slightly worse false detection rate than the quite effective specialized face detector of Mikolajczyk [24] used in [6].

For each image, edges are extracted at four orientations and a fixed scale. For the Caltech dataset where significant texture and clutter are present, we use the boundary detector of [23] at a scale of 2% of the image diagonal. With the face dataset, a quadrature pair of even and odd symmetric Gaussian derivatives suffices. We use a scale of $\sigma = 2$ pixels and elongate the filter by a factor of 4 in the direction of the putative edge orientation.

Geometric blur features are computed at 400 points sampled randomly on the image with the blur pattern shown in Figure 8. We use a maximum radius of 50 pixels (40 for faces), and blur parameters $\alpha = 0.5$ and $\beta = 1$.

For correspondence we use 50 (40 for faces) points, sampled randomly on edge points, in the correspondence problem. Each point is allowed to match to any of the most similar 40 points on the query image based on feature similarity. In addition for the Caltech 101 dataset we use $\gamma = 0.9$ allowing correspondences

[9] Some normalization is necessary to deal with relatively smaller or larger objects with fewer or more descriptors.

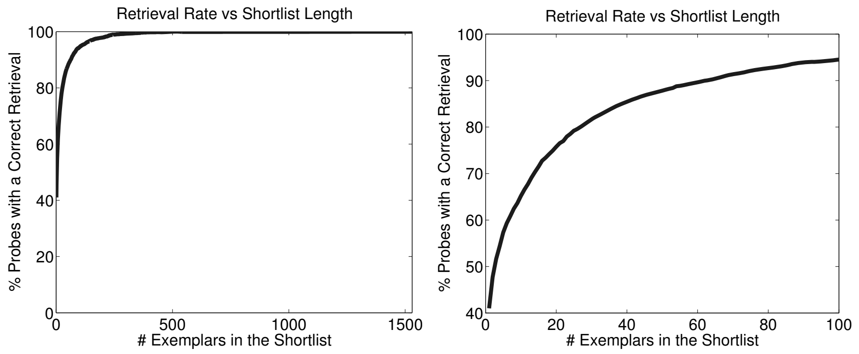


Fig. 11. For a probe or query image exemplars are ranked according to feature similarity. We plot the percentage of probes for which an exemplar of the correct class was found in the shortlist. Here the first exemplar is correct 41% of the time. **Left** Full curve. **Right** Curve up to shortlist length 100 for detail.

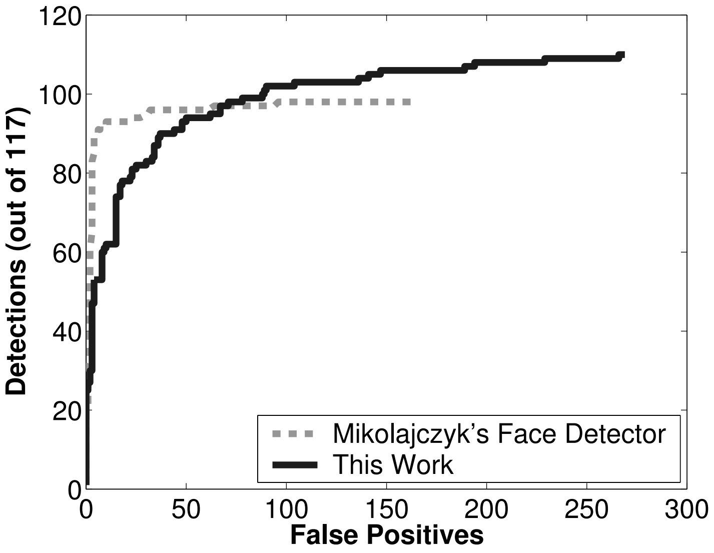
with significant variation in scale, while for the faces dataset we handle scale variation partly by repeating exemplars at multiple scales and use $\gamma = 0.5$.

7 Caltech 101 Results

Basic Setup: Fifteen exemplars were chosen randomly from each of the 101 object classes and the background class, yielding a total 1530 exemplars. For each class, we select up to 50 testing images, or "probes" excluding those used as exemplars. Results for each class are weighted evenly so there is no bias toward classes with more images.

The spatial support of the objects in exemplars is acquired from human labeling. The top entry in the shortlist is correct 41% of the time. One of the top 20 entries is correct 75% of the time. (Figure 11).

Recognition and Localization. Using each of the top ten exemplars from the shortlist we find a good correspondence in the probe image. We do this by first sampling 50 locations on the exemplar object and allowing each to be matched to its 50 best matching possibilities in the probe with up to 15% outliers. This results in a quadratic programming problem of dimension 2550. We use a distortion cost based mainly on the change in angle of edges between vertices ($\gamma = 0.9$). This allows matches with relatively different scales (Figure 10 line 3). The exemplar with the lowest distortion correspondence gives **45%** correct classification, at the same time providing localization. Note that this is using a simple nearest neighbor classifier and generative models. A baseline experiment comparing gray scale images using SSD and 1-nearest neighbor classification gives 16%. At press, the best results from the Caltech group are 40% using discriminative methods [15]. No other techniques have addressed correspondence at the level of detail presented here.



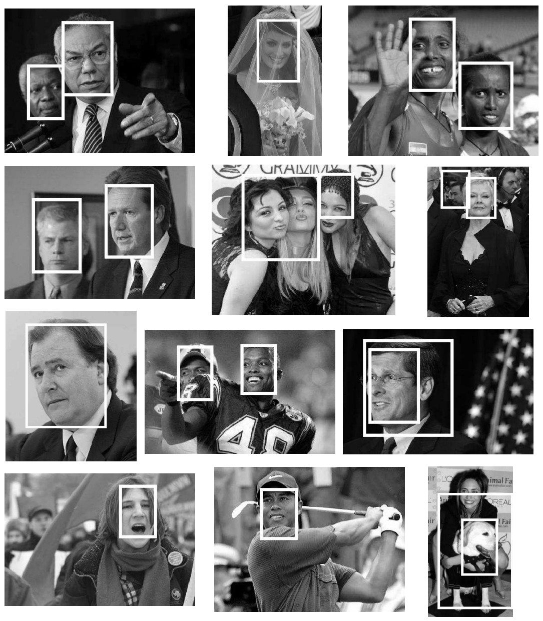


Fig. 12. Top ROC curves for our face detector using 20 exemplar images of faces (split between frontal and profile) and the detector of Mikolajczyk. Mikolajczyk's detector has proven to be effective on this dataset. simply finding sets of feature points in an image that have a good correspondence, based on distortion cost, to an exemplar. Good correspondences allow detection and localization of faces using a simple generative model, no negative examples were used. **bottom** Detections from our face detector marked with rectangles.

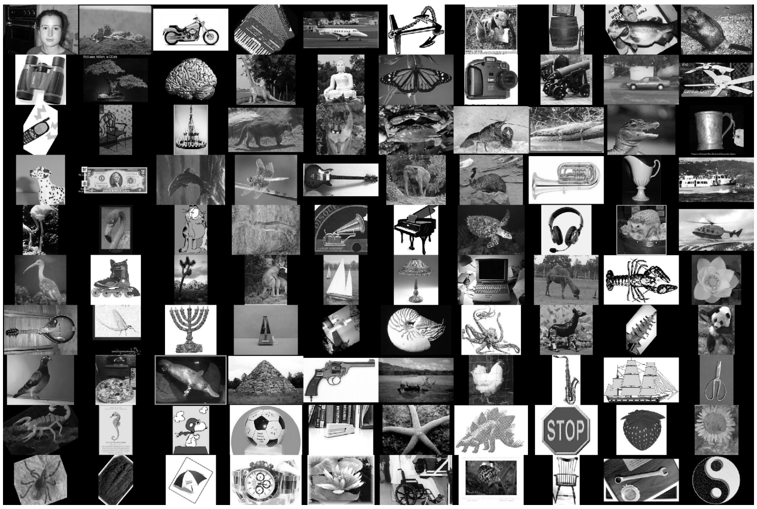


Fig. 13. Example images from 100 of the categories in the Caltech 101 dataset

Multiscale. We compute exemplar edge responses and features at a second scale for each exemplar resulting in twice as many exemplars. This improves shortlist performance by 1% or less, and does not change recognition performance. This illustrates the general lack of scale variation in Caltech 101. The face dataset exhibits a large range of scale variation.

8 Face Detection Results

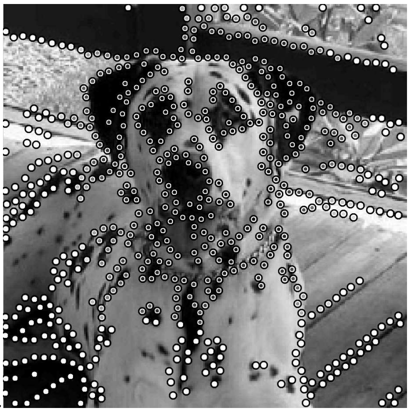
We apply the same technique to detecting medium to large scale faces for use in face recognition experiments. The face dataset is sampled from the very large dataset in [6] consisting of A.P. news photographs. A set of 20 exemplar faces split between front, left, and right facing, was chosen from the database by hand, but without care. The test set was selected randomly from the remaining images on which the face detector of [24] found at least one 86×86 pixels or larger face. We use the generic object recognition framework described above, but after finding the lowest cost correspondence we continue to look for others. A comparison of the ROC curves for our detector and that of [24] is found in Figure 12. Our detector has an advantage in generalization, while producing more false positives. While not up the the level of specialized face detectors, these are remarkably good results for a face detector using 20 exemplars and a generative model for classification, without any negative training examples.

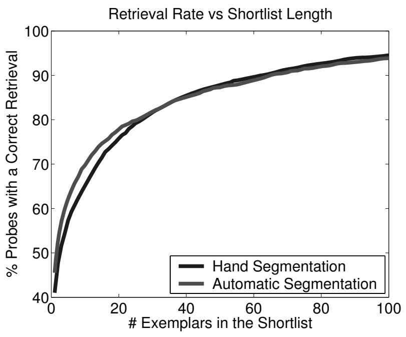
9 Automatic Segmentation

In the recognition experiments above, exemplar objects were hand segmented from their backgrounds. This can be automated by finding the repetitive aspects

a.

b.


c.



d.

Fig. 14. Illustrating automatic model segmentation: One training image (**a.**) the remaining 14 training images (**b.**) darkness of the central circle for each feature indicates how well on average the area around the feature point matched after aligning transforms to each of the other training images (**c.**) At lower right, the percentage of probes for which an exemplar of the correct class was found in the shortlist. The **darker** curve shows performance with hand segmented exemplars, the **lighter** curve shows performance with *automatically* segmented exemplars. For hand segmented exemplars the first exemplar is correct 41% of the time, for automatically segmented exemplars 45%. (**d.**)

of objects in the example images. Starting with a set of example images $\{I_i\}$ from an object class find the support of the object in an image I_{i_0} as follows. For each image I_j where $j \neq i_0$:

1. Find a correspondence from I_{i_0} to I_j.[10]
2. Use a regularized thin plate spline to map all of the feature points in I_{i_0} to I_j.
3. For each mapped feature from I_{i_0}, the quality of the match is the similarity to the best matching nearby feature in I_j.

[10] Here we allow 40% outliers instead of 15% as used in the recognition experiments.

The median quality of match for a feature is the measure of how common that feature is in the training images.

Feature points with median quality within 90% of the best for that image are considered part of the object. Repeating the recognition experiments in Section 7, the shortlist accuracy improves by 1-4% (Fig. 14). While the estimated support is usually not perfect, recognition performance is similar to that using hand segmented images, 45%.

The learned models of support reflect a region of the image that is consistent across training images, as opposed to individual discriminative features. For instance the cheek on a face is not by itself discriminative for faces, but when considering faces transformed into alignment the cheek is usually consistent.

References

1. Y. Amit, D. Geman, and K. Wilder. Joint induction of shape features and tree classifiers. *IEEE Trans. PAMI*, 19(11):1300–1305, November 1997.
2. S. Belongie, J. Malik, and J. Puzicha. Matching shapes. In *Proc. 8th Int. Conf. Computer Vision*, pages I.454–461, 2001.
3. S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. PAMI*, 24(4):509–522, April 2002.
4. A. C. Berg. *Shape Matching and Object Recognition.* PhD thesis, U.C. Berkeley, December 2005.
5. A. C. Berg and J. Malik. Geometric blur for template matching. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 607–614, 2001.
6. T. L. Berg, A. C. Berg, J. Edwards, M. Maire, R. White, Y. W. Teh, E. Learned-Miller, and D. A. Forsyth. Names and faces in the news. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 848–854, 2004.
7. Caltech 101 dataset
   www.vision.caltech.edu/feifeili/101_ObjectCategories .
8. H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Comp. Vision and Image Underst.*, 89:114–141, 2003.
9. L. Fei-Fei, R. Fergus, and P. Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *Proc. 9th Int. Conf. Computer Vision*, pages 1134–1141, 2003.
10. L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *Workshop on Generative-Model Based Vision*, 2004.
11. R. Fergus, P. Perona, and A Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 264–271, 2003.
12. M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. Computers*, C-22(1):67–92, 1973.
13. D. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *Proc. 7th Int. Conf. Computer Vision*, pages 87–93, 1999.
14. U. Grenander, Y. Chow, and D.M. Keenan. *HANDS: A Pattern Theoretic Study Of Biological Shapes.* Springer, 1991.
15. A. Holub, M. Welling, and P. Perona. Combining generative models and fisher kernels for object recognition. In *Proc. 10th Int. Conf. Computer Vision*, pages 136– 143, 2005.

16. D.P. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Trans. PAMI*, 15(9):850–863, Sept. 1993.
17. M. Lades, C.C. Vorbrüggen, J. Buhmann, J. Lange, C. von der Malsburg, R.P. Wurtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Trans. Computers*, 42(3):300–311, March 1993.
18. Y. Lamdan, J.T. Schwartz, and H.J. Wolfson. Affine invariant model-based object recognition. *IEEE Trans. Robotics and Automation*, 6:578–589, 1990.
19. T.K. Leung, M.C. Burl, and P. Perona. Finding faces in cluttered scenes using random labeled graph matching. In *Proc. 5th Int. Conf. Computer Vision*, pages 637–644, 1995.
20. D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. 7th Int. Conf. Computer Vision*, pages 1150–1157, 1999.
21. D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2):91–110, 2004.
22. J. Maciel and J Costeira. A global solution to sparse correspondence problems. *IEEE Trans. PAMI*, 25(2):187–199, 2003.
23. D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. PAMI*, 26(5):530–549, 2004.
24. K. Mikolajczyk. *Detection of local features invariant to affines transformations.* PhD thesis, INPG, 2002.
25. K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 257–263, 2003.
26. G. Mori, S. Belongie, and J. Malik. Shape contexts enable efficient retrieval of similar shapes. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, volume 1, pages 723–730, 2001.
27. M. Morrone and D. Burr. Feature detection in human vision: A phase dependent energy model. *Proc. Royal Soc. of London B*, 235:221–245, 1988.
28. M. J. D. Powell. A thin plate spline method for mapping curves into curves in two dimensions. In *CTAC*, Melbourne, Australia, 1995.
29. A. Rangarajan, H. Chui, and E. Mjolsness. A relationship between spline-based deformable models and weighted graphs in non-rigid matching. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, volume 1, pages 897–904, December 2001.
30. F. Rothganger, S. Lazebnik, C Schmid, and J Ponce. 3d object modeling and recognition using affine-invariant patches and multi-view spatial constraints. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages II:272–275, 2003.
31. H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 746–751, 2000.
32. H. Schneiderman. Feature-centric evaluation for efficient cascaded object detection. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 29–36, 2004.
33. C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. PAMI*, 19(5):530–535, May 1997.
34. D'Arcy Wentworth Thompson. *On Growth and Form.* Dover, 1917.

35. A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 762–769, 2004.
36. S. Ullman, M. Vidal-Naquet, and E Sali. Visual features of intermediate complexity and their use in classification. *Nat. Neur.*, 13:682–687, 2002.
37. P. Viola and M. Jones. Robust real-time object detection. *2nd Intl. Workshop on Statistical and Computational Theories of Vision*, 2001.