Shape Matching and Object Recognition using Low Distortion Correspondences

Alexander C. Berg* Tamara L. Berg † Jitendra Malik
Computer Science Division
U.C. Berkeley
UCB//CSD-04-1366

Dec. 6, 2004

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 then 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 (Fei-Fei, Fergus and Perona), an extremely challenging dataset with large intraclass variation. Our approach yields a 48% correct classification rate, compared to Fei-Fei et al's 16%. We also show results for localizing frontal and profile faces that are comparable to special purpose approaches tuned to this task.

1. Introduction

Our thesis is that recognizing object categories, be they fish or bicycles, is fundamentally a problem of 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 [10], in statistical image analysis, Grenander ([12]and earlier), and in neural networks, von der Malsburg ([14] and earlier). The core idea that related but not identical shapes can be deformed into alignment using simple coordinate transformations dates even further back, to D'Arcy Thompson, in his classic work *On Growth and Form* [31].

The setup is the following: we have stored multiple exemplars for different object categories, and in the case of 3D objects, multiple 2D views from different poses as well. Given an image which contains an unknown object (shape), we compare it to different exemplar shapes thus

1. solve the correspondence problem between the two shapes,
2. use the correspondences to estimate an aligning transform, and
3. compute the distance between the two shapes as a sum of matching errors between corresponding points.

Given these distances we can use a nearest neighbor classifier to determine the category of the hitherto unknown shape in the image.

Practically speaking, the most difficult step is the *correspondence problem*, e.g. how do we algorithmically determine which points on two shapes correspond? The correspondence problem in this setting is much harder than in the setting of, say, binocular stereopsis, for a number of reasons:

1. Intra-category variation: the aligning transform between two instances of a category is not a simple parameterized transform. It is reasonable to assume that it is a smooth mapping, but it would be difficult to characterize it 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

*Funded by ONR N00014-01-1-0890 (MURI) and NSF ITR IIS-00-85864

†Funded by a NSF graduate student fellowship

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

First, some background. 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 [18].

We exploit three kinds of constraints to solve the correspondence problem between shapes represented as point sets:

1. Matching point descriptors: Corresponding points on the two shapes should have similar local descriptors. There are several choices here: SIFT [18], Shape contexts [3], and Geometric blur[4]. We used geometric blur.

2. Minimizing geometric distortion: If point i corresponds to i', and point j corresponds to j', then the vector from i to j, $\vec{r}_{ij}$ should be consistent with the vector from i' to j', $\vec{r}_{i'j'}$. E.g. 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. And so on.

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.

We encode both the similarity of point descriptors and the geometric distortion in the form of a cost function, where the minimization is over the space of correspondences. This is then solved as an integer quadratic programming problem (cf. Maciel and Costeira [19]).

We address two object recognition problems, multiclass recognition and face detection. In the multi-class object class recognition problem, given an image of an object we must identify the class of the object and its location in the image. We use the Caltech 101 object class dataset consisting of images from 101 classes of object: from accordion to kangaroo to yin-yang, available at [1]. This dataset includes significant intra class variation, a wide variety of classes, and clutter. On average we achieve **48%** accuracy on object classification with quite good localization on the correctly classified objects. This compares favorably with the state of the art of 16% from [8].

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 [5] 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 used in [5].

2. Related Work

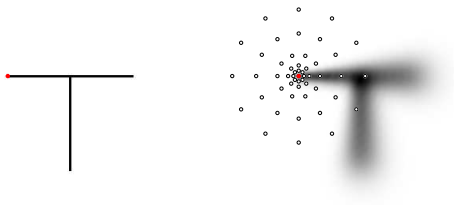
There have been several approaches to shape recognition based on spatial configurations of a small number of keypoints or landmarks. In geometric hashing [15], these configurations are used to vote for a model without explicitly solving for correspondences. Amit et al. [2] train decision trees for recognition by learning discriminative spatial configurations of keypoints. Leung et al. [16], Schmid and Mohr [28], and Lowe [17] additionally use gray level information at the keypoints to provide greater discriminative power. Lowe's SIFT descriptor [17] [18] have been shown in various studies e.g. [22] to perform very well particularly at tasks where one is looking for identical point features.

Recent work extends this approach to category recognition [9, 7, 8], and to three-dimensional objects[27].

It should be noted that not all objects have distinguished key points (think of a circle for instance), and using key points alone sacrifices the shape information available in smooth portions of object contours. Approaches based on extracting edge points are, in our opinion, more universally applicable. Huttenlocher et al. developed methods in this category based on the Hausdorff distance [13]; this can be extended to deal with partial matching and clutter. A drawback for our purposes is that the method does not return correspondences. Methods based on Distance Transforms, such as [11], are similar in spirit and behavior in practice. Work based on shape contexts is indeed aimed at first finding correspondences [3, 23] and is close to the spirit of this work. Another approach is the non-rigid point matching of [6] based on thin plate splines and "softassign".

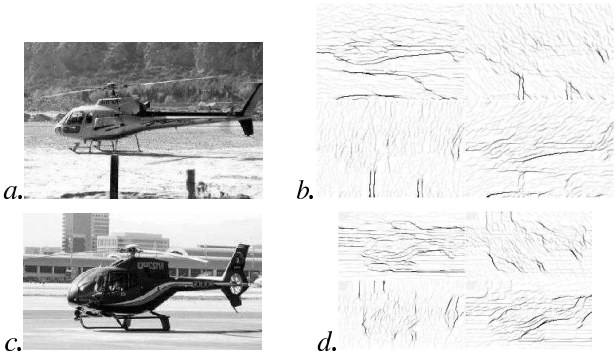
One can do without extracting either keypoints or edge points: Ullman et al propose using intermediate complexity features, a collection of image patches,[34, 33]

For faces and cars the class specific detectors of [35, 30, 29] have been very successful. These techniques use simple local features, roughly based on image gradients, and a cascade of classifiers for efficiency. Recent work on sharing features [32] has extended to multiclass problems.



a. *b.*

Figure 1: *A sparse signal S (**a.**) and the geometric blur of S around the feature point marked in red (**b.**) We only sample the geometric blur of a signal at small number of locations $\{s_i\}$, indicated in (**b.**)*



a. *b.* *c.* *d.*

Figure 2: *Two images (**a.** and **c.**) and four oriented edge channels derived from the images using the boundary detector of [20] (**b.** and **d.** respectively). A feature point descriptor is the concatenation of the subsampled geometric blur descriptor at the feature point for each of the channels.*

3. Geometric Blur Descriptor

We use features based on a subsampled version of the geometric blur descriptor of [4]. This descriptor is a smoothed version of the signal around a feature point, blurred by a spatially varying kernel. The amount of blur is small near the feature point and grows with distance from the feature point. The intuition is that under an affine transform that fixes a feature point, the distance a piece of the signal moves is linearly proportional to the distance that piece was from the feature point.

When applied to a sparse signal related to the image such as oriented edge energy channels, the geometric blur descriptor provides a comparison of the neighborhoods around feature points that is robust to affine distortion. It is important to note that the spatial support of the geometric blur features can be made quite large, and as a result the descriptors can provide useful discriminative information.

In practice the geometric blur of a signal is usually rather smooth far from a feature point, we take advantage of this by subsampling the geometric blur, as shown in figure 1.

The experiments in this paper use two types of sparse channels from which to compute geometric blur descriptors, the oriented edge detector outputs of [20] and oriented edge energy using quadrature pairs, following [24, 25]. See Figure 2 for an example. In each case the edge detector is used to produce four channels of oriented edge responses. The feature descriptor at a point is then the concatenation of the subsampled geometric blur descriptor at that point in each of the channels.

For this work we use a spatially varying Gaussian kernel to compute geometric blur. Given one of the oriented channels discussed above as the signal, S, we compute blurred versions, $S_d = S * G_d$, by convolving with a Gaussian of standard deviation d. The geometric blur descriptor around location x_0 is then

$$B_{x0}(x) = S_{\alpha|x|+\beta}(x_0 - x) \tag{1}$$

Where α and β are constants that determine the amount of blur. We sample this signal at a sparse set of points $x = s_i$ as shown in figure 1, so we need only compute S_d for a few distinct values of $d = \alpha|s_i| + \beta$. Since the Gaussian is a separable kernel and we can subsample the signal for larger standard deviations, extracting geometric blur descriptors is quite fast, taking less than a second per image in our experiments.

We compare geometric blur descriptors using (L_2) normalized correlation.

4. Geometric Distortion Costs

We consider correspondences between feature points $\{p_i\}$ in 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:

$$C_{\text{match}}(\sigma) = \sum_i c(i, i') \tag{2}$$

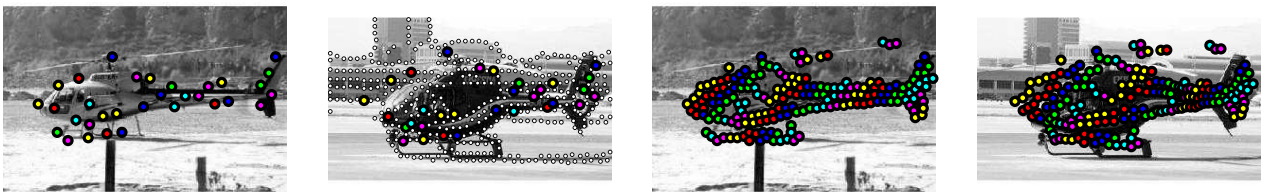


Figure 3: *An exemplar with a subset of feature points marked (***left***), the novel "probe" image with all feature points in white, and the feature points found to correspond with the exemplar feature points marked in corresponding colors (***left center***), the exemplar with all its feature points 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 9 for more examples*

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 points in an image. This allows the cost minimization to be expressed as an integer quadratic programming problem.

$$C_{\text{distortion}}(\sigma) = \sum_{ij} H(i,i',j,j') \tag{3}$$

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

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

Our distortion cost is made up of two components:

$$C_{\text{distortion}}(\sigma) = \sum_{ij} \gamma \text{d}_\text{a}(\sigma) + (1-\gamma)\text{d}_\text{l}(\sigma) \tag{4}$$

$$d_a(\sigma) = \left(\frac{\alpha_d}{|r_{ij}|} + \beta_d\right) \left|\arcsin\left(\frac{s_{i'j'} \times r_{ij}}{|s_{i'j'}||r_{ij}|}\right)\right| \tag{5}$$

$$d_l(\sigma) = \frac{|s_{i'j'}| - |r_{ij}|}{(|r_{ij}| + \mu_d)} \tag{6}$$

where d_a penalizes the change in direction, and d_l penalizes change in length. 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 related to terms allowing slightly more flexibilty 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 against the length distortion.

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 match to any point p_i under σ. We add an additional point q_{null} and use $\sigma(p_i) = q_{\text{null}}$ 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.

5. Correspondence Algorithm

Finding an assignment to minimize a cost function described by the terms in Equations 3 and 2 above can be written as the objective function for an Integer Quadratic Programming (IQP) problem.

$$\text{cost}(x) = \sum_{a,b} H(a,b)x_a x_b + \sum_a c(a)x_a \tag{7}$$

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 3 and 2.

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. Futhermore if we allow outliers as discussed in Section 4, 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 one 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:

$$\begin{aligned} \min \text{cost}(x) = & x'Hx + c'x \quad \text{subject to,} \\ & Ax = b, \quad x \in \{0,1\}^n \end{aligned} \tag{8}$$

5.1. Approximation

Integer Quadratic Programming is NP-Complete, 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 as follows:

$$q_a = \min \sum_b H_{ab} x_b \quad \text{subject to,} \qquad (9)$$
$$Ax = b, \qquad x \in \{0,1\}^n$$

If x_a represents $\sigma(i) = j$ then q_a is a lower bound for the cost contributed to any assignment by using $\sigma(i) = j$.

$$\min L(x) = \sum_a (q_a + c_a) x_a \quad \text{subject to,} \qquad (10)$$
$$Ax = b, \qquad x \in \{0,1\}^n$$

$L(x)$ is a lower bound for $cost(x)$ from Equation 8. This construction follows [19], and is a standard bound for a quadratic program. Of note is the operational similarity to geometric hashing, with allowance for non-rigidity.

Equation 9 and 10 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 and m is the number of points in image P, each problem takes $O(nm \log(m))$ operations with a very small constant. Computing q_a for $a = 1 \ldots n$ requires $O(n^2 m \log(m))$ time.

We then perform gradient descent changing one element of the assignment at each step. This takes $O(n)$ operations per step, and usually requires a very small number of steps (we limit this to 100). 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 2 seconds.

5.2. Warping

Once we have found a low distortion correspondence between points in two images we can define a smooth warp between the images. We use regularized thin plate splines [26] fit to the correspondences. The regularized thin plate spline for a univariate function is modeled on the idea of minimizing the bending energy of a thin plate deflected to approximate some values. It is widely used for data interpolation.

6. Correspondence results

Given an 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. Pick m of these features from P.
4. Allow each of the m feature points from P to potentially match the n most similar points in Q based on feature similarity and or proximity.
5. Construct the cost matrices H and c based on Section 4.
6. Approximate the resulting Binary Quadratic Optimization to obtain a correspondence. Store the cost of the correspondence as well.
7. Given the correpondence on m points extend to a smooth map for other points in P using a regularized thin plate spline.

See Figures 3 and 9 for a number of examples. In the leftmost column of the figure 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. Correponding points are colored similarly and points are colored based on their position (or corresponding position) in P.

7. Recognition Experiments

Our recognition framework is based on nearest neighbor.

- *Preprocessing*
  1. For each object class we store a number of exemplars.
  2. Possibly replicate the exemplars at different scales.
  3. Compute features for all of the exemplars as described above.
- *Indexing*
  1. For a query image, extract features as described above
  2. 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 median of these best correlations is the similarity of the exemplar to the probe.
  3. Form a shortlist of the exemplars with highest similarity to the query image.
- *Correspondence*
  1. Find a correspondence from each exemplar in the shortlist to the query. Score these correspondences by their cost.

2. Pick the exemplar with the least cost.

- *Warping:* If desired, construct a thin plate spline transformation to warp other points on the exemplar to the query image, or look for additional objects in the query points not used by the correspondence.

We apply our technique to two different data sets, the Caltech set of 101 object categories (available here [1]) and a collection of news photographs containing faces gathered from yahoo.com (provided by the authors of [5]). In the experiments that follow, we utilize the same parameters for both datasets except for those specifically mentioned.

For all images 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 [20] 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 1. 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 with significant variation in scale, while for the faces dataset we hande scale variation partly by repeating exemplars at multiple scales and use $\gamma = 0.5$.

8. Caltech 101 Results

Basic Setup: Fifteen exemplars were chosen randomly from each of the 101 object classes and the background class, yeilding a total 1530 exemplars. For each class, we select up to 50 testing images, or "probes" excluding those used as exemplars. Some of the classes have fewer than 65 elements, in which case we take as many as are available. Results for each class are weighted evenly so there is no bias toward classes with more images.

The spatial support of the objects in the exemplars is acquired from human labeling. The shortlist results are shown in Figure 4. The top entry in the shortlist is corect 41% of the time. One of the top 20 entries is correct 75% of the time.

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 20% 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 as shown in the Figure 9. Each exemplar is rated by the resulting matching cost. The class of the best exemplar based on correspondence cost gives 48% correct classification. This is an increase over the best matching exemplar not using correspondence (41%), and provides localisation of the object in the image that is approximately correct in almost all cases when the probe is classified correctly.[1]

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 lack of scale variation in Caltech 101. The face dataset exhibits a large range of scale variation.

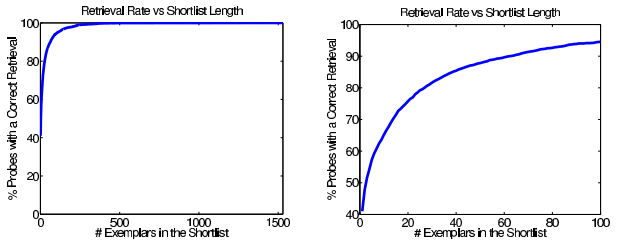


Figure 4: *For a probe or query image exemplars are ranked according to feature similarity. We plot the percentage of probes for which an exemplar of the correct class was found in the shortlist. Here the first exemplar is correct 41% of the time.* **Left** *Full curve.* **Right** *Curve up to shortlist length 100 for detail.*

9. Face Detection Results

We apply the same technique to detecting medium to large scale faces for possible use in face recognition experiments. The face dataset is sampled from the very large dataset from [5] consisting of news photographs collected from yahoo.com. A set of 20 exemplar faces split between frontal, left, and right facing, was chosen from the database by hand, but without care. We selected the testing set randomly from the remaining images on which the face detector of [21] found at least one large (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. We generate an ROC curve based on our match cost, and compare this to the ROC curve for the detector of [21]. See figure 6. Our detector has an

[1]For comparison our baseline experiments using 1-nn and color histograms achieved 12% while 1-nn with ssd on greyscale images achieved 16%, comparable to the state of the art.

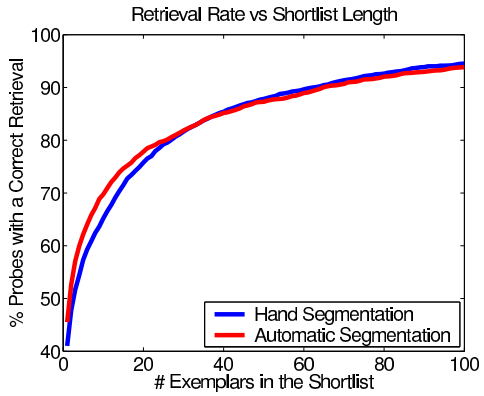


Figure 5: *The percentage of probes for which an exemplar of the correct class was found in the shortlist. The* **blue** *curve shows performance with hand segmented exemplars, the* **red** *Curve shows performance with* automatically *segmented exemplars. For hand segmented exemplars the first exemplar is correct 41% of the time, for automatically segmented exemplars 45%.*

advatage 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.

10. Automatic Model Building

In the recognition experiments above, exemplar objects were hand segmented from their backgrounds. We now show how these segmentations can be performed automatially.

We exploit repetition of objects in the example images. For a given exemple image we will identify the features that are repeated in the same configuration in the other example images. Ideally this would be computed for all images simultaneously. We show that in many cases it is sufficient to find the similar parts in pairs of images independantly.

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:

1. For each image I_j where $j \neq i_0$:
   (a) Find a correspondence from I_{i_0} to I_j [2]
   (b) Use a regularized thin plate spline to map all of the feature points in I_{i_0} to I_j
   (c) Foreach mapped feature from I_{i_0}, the quality of the match is the similarity to the best matching nearby feature in I_j.
2. The median quality of match for a feature is the measure of how common that feature is in the training images.

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

In Figures 7 and 8 the points shown in color are those with median quality within 90% of the best for that image. Using this automatically estimated support we repeat the recognition experiments in Section 8. The shortlist accuracy is actually better using the automatic segmentation, an improvement of 1-4% for the top 10 entries, as can be seen in Figure 5.

While the estimated support is not always intuitive, recognition performance is very close to that using the same techniques and hand segmented images, 48%. As can be seen in figures7 and 8 the automatically estimated support is often correct.

This is a very simple example of correspondence allowing us to build models in object coordinates. The learned models of support reflect a region of the image that is consistent across training images, as opposed to individual discriminative features.

11. Acknowledgements

We thank Charless Fowlkes for mentioning the work of Macial and Costeira, David Forsyth for excellent discussion on quadratic programming, and David Martin and Charless Fowlkes for their boundary detection code.

References

[1] Caltech 101 dataset `www.vision.caltech.edu/feifeili/101_objectcategories/`.

[2] Y. Amit, D. Geman, and K. Wilder. Joint induction of shape features and tree classifiers. *IEEE Trans. PAMI*, 19(11):1300–1305, November 1997.

[3] S. Belongie, J. Malik, and J. Puzicha. Matching shapes. In *ICCV*, pages I.454–461, 2001.

[4] A. C. Berg and J. Malik. Geometric blur for template matching. In *CVPR*, pages 607–614, 2001.

[5] 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 *CVPR*, pages 848–854, 2004.

[6] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *CVIU*, 89:114–141, 2003.

[7] L. Fei-Fei, R. Fergus, and P. Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *ICCV*, pages 1134–1141, 2003.

[8] 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 *CVPR, Workshop on Generative-Model Based Vision*, 2004.

[9] R. Fergus, P. Perona, and A Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages 264–271, 2003.

[10] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. Computers*, C-22(1):67–92, 1973.

[11] D. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *Proc. 7th Int. Conf. Computer Vision*, pages 87–93, 1999.

[12] U. Grenander, Y. Chow, and D.M. Keenan. *HANDS: A Pattern Theoretic Study Of Biological Shapes*. Springer, 1991.

[13] D.P. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Trans. PAMI*, 15(9):850–863, Sept. 1993.

[14] 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.

[15] Y. Lamdan, J.T. Schwartz, and H.J. Wolfson. Affine invariant model-based object recognition. *IEEE Trans. Robotics and Automation*, 6:578–589, 1990.

[16] 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.

[17] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, 1999.

[18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[19] J. Maciel and J Costeira. A global solution to sparse correspondence problems. *PAMI*, 25(2):187–199, 2003.

[20] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *PAMI*, 26(5):530–549, 2004.

[21] K. Mikolajczyk. *Detection of local features invariant to affines transformations*. PhD thesis, INPG, 2002.

[22] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *CVPR*, pages 257–263, 2003.

[23] G. Mori, S. Belongie, and J. Malik. Shape contexts enable efficient retrieval of similar shapes. In *CVPR*, volume 1, pages 723–730, 2001.

[24] 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.

[25] P. Perona and J. Malik. Detecting and localizing edges composed of steps, peaks and roofs. In *Proc. Int. Conf. Computer Vision*, pages 52–7, Osaka, Japan, Dec 1990.

[26] M. J. D. Powell. A thin plate spline method for mapping curves into curves in two dimensions. In *Computational Techniques and Applications (CTAC95)*, Melbourne, Australia, 1995.

[27] F. Rothganger, S. Lazebnik, C Schmid, and J Ponce. 3d object modeling and recognition using affine-invariant patches and multi-view spatial constraints. In *CVPR*, pages II:272–275, 2003.

[28] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. PAMI*, 19(5):530–535, May 1997.

[29] H. Schneiderman. Feature-centric evaluation for efficient cascaded object detection. In *CVPR*, pages 29–36, 2004.

[30] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *CVPR*, pages 746–751, 2000.

[31] D'Arcy Wentworth Thompson. *On Growth and Form*. Dover, 1917.

[32] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *CVPR*, pages 762–769, 2004.

[33] M. Vidal-Naquet and S. Ullman. Object recognition with informative features and linear classification. In *ICCV*, pages 281–287, 2003.

[34] S. Ullman M. Vidal-Naquet and E Sali. Visual features of intermediate complexity and their use in classification. *natneur*, 13:682–687, 2002.

[35] P. Viola and M. Jones. Robust real-time object detection. *2nd Intl. Workshop on Statistical and Computational Theories of Vision*, 2001.

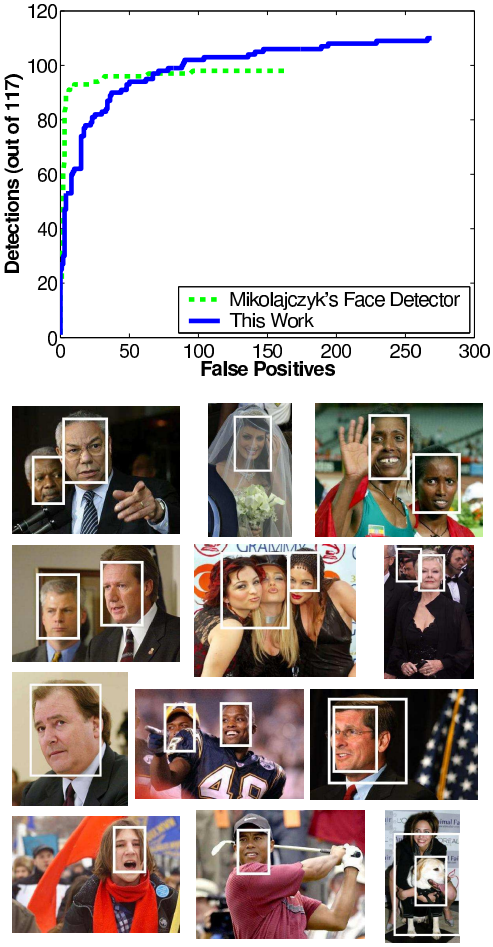


Figure 6: **Top** *ROC curves for our face detector using 20 exemplar images of faces (split between frontal and profile) and the the detector of Mikolajczyk [21] (similar to that of [30]) evaluated on a dataset of ap news photos. Mikolajczyk's detector has proven to be effective on this dataset [5]. Our detector works by simply finding sets of feature points in an image that have a good correspondence, based on distortion cost, to 20 exemplars. Good correspondences allow detection and localization of faces using a simple generative model based on nearest neighbor to 20 exemplars. No negative examples were used for our model.* **Bottom** *Detections from our face detector marked with rectangles.*

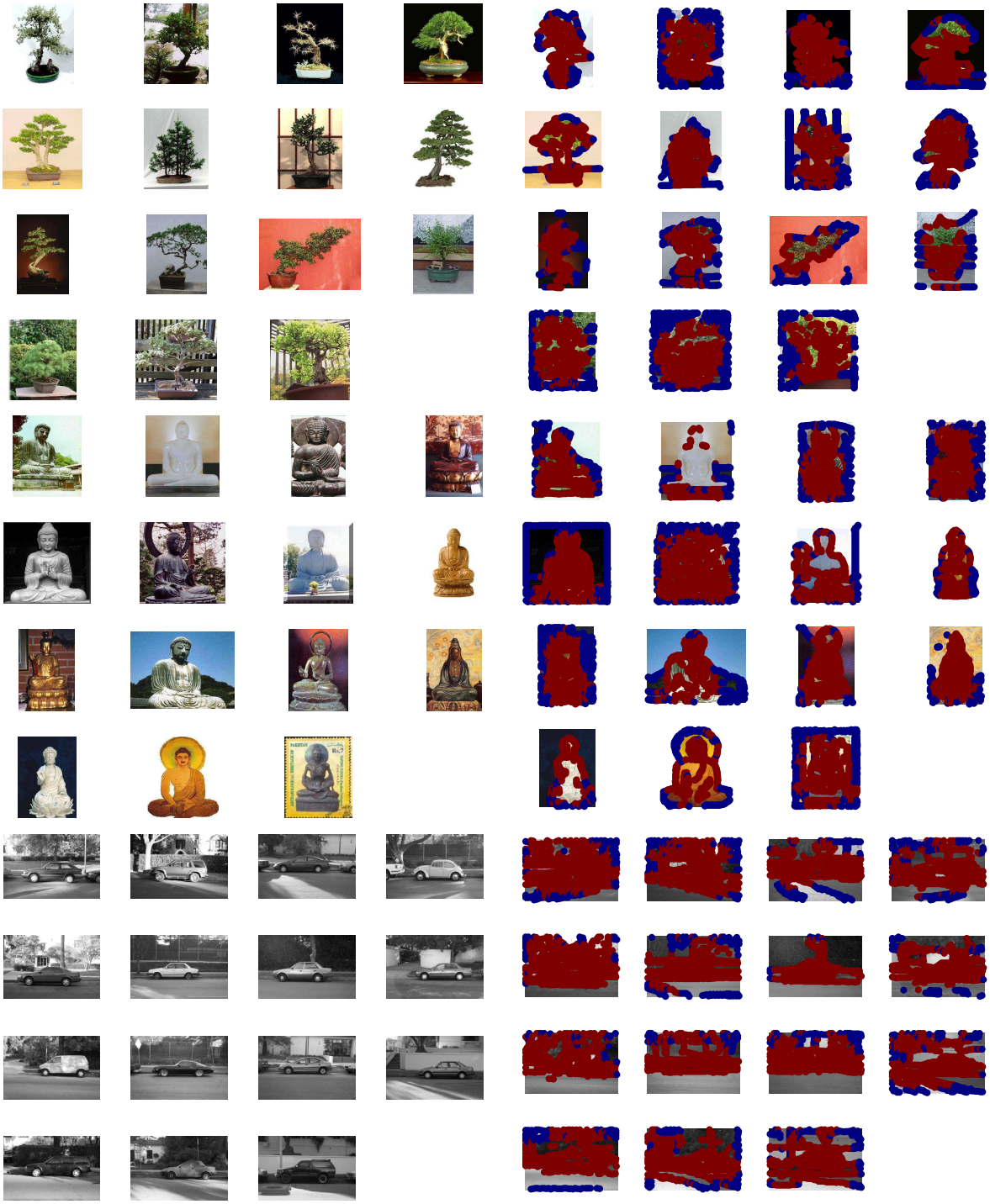


Figure 7: *Training images for a class are shown at* **Left**, *the locations of the objects are not given. For each image the feature points which are well registered by alignment (as described in Section 10) are automatically extracted and marked in red at* **right** *The car class shows an apparent failure as much of the background has been labeled as object, but significant portions of the background are actually as consistent as the object itself! (Note images are in color for clarity, all computation is done on grayscale images.)*

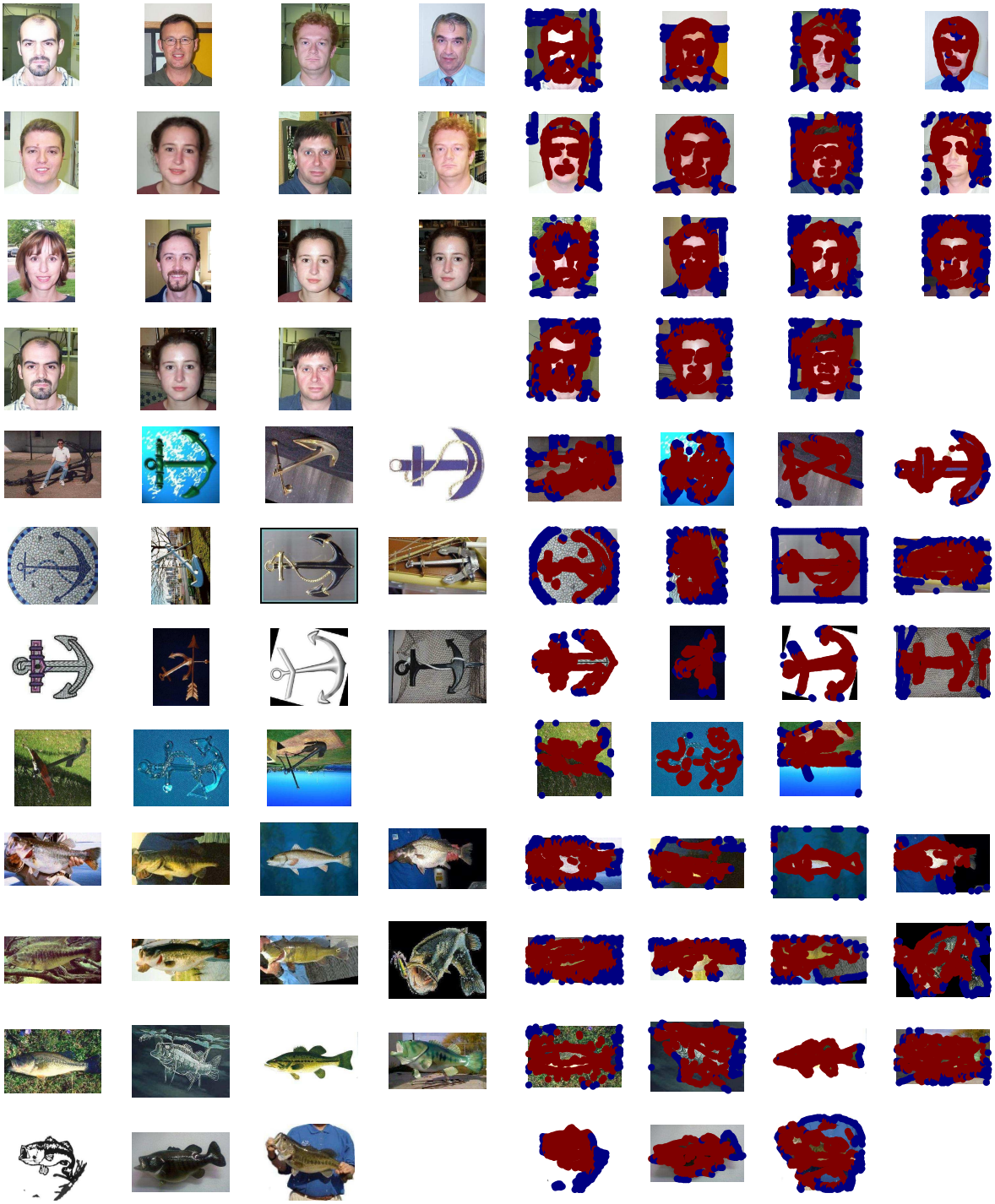


Figure 8: *Training images for a class are shown at* **Left**, *the locations of the objects are not given. For each image the feature points which are well registered by alignment (as described in Section 10) are automatically extracted and marked in red at* **right** . *(Note images are in color for clarity, all computation is done on grayscale images.)*

Figure 9: *Each row shows a correspondence found using our technique described in section 5. 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).*