Sequential Learning of Reusable Parts for Object Detection

S. Krempp
CMLA
Ecole Normale Superieure
Cachan, France 94325
krempp@cmla.ens-cachan.fr

D. Geman
Dept. of Math. Sciences
Johns Hopkins University
Baltimore, MD 21218-2686
geman@cis.jhu.edu

Y. Amit
Dept. of Statistics
University of Chicago
Chicago, IL 60637
amit@marx.uchicago.edu

Abstract

Our long-range goal is detecting instances from a large number of object classes in a computationally efficient manner. Detectors involving a hierarchy of tests based on edges have been used elsewhere and shown to be quite fast online. However, significant further gains in efficiency - in representation, error rates and computation - can be realized if the family of detectors is constructed from common parts. Our parts are flexible, extended edge configurations; they are learned, not pre-designed. In training, object classes are presented sequentially; the objective is then to accommodate new classes by maximally reusing parts. Ideally, the number of distinct parts in the system would grow much more slowly than linearly with the number of classes. Initial experiments on learning to detect several hundred LaTeX symbols are encouraging.

1. Introduction

One of the grand but largely unrealized objectives of computer vision is semantic scene labeling - identify all instances of ordinary objects in natural images or video sequences, including object poses and occlusion patterns. One obstacle is certainly the sheer number of different object categories that might appear in typical indoor and outdoor scenes. Whereas in any given application this number might be severely limited by specific goals, it is not unrealistic to imagine hundreds or even thousands of categories of interest. Add another order of magnitude if the challenge is to emulate human capabilities.

Naturally, computation then becomes the paramount issue, both for constructing the detectors (whether from models or samples) and for implementing them online. Throughout this paper a "detector" is a binary classifier for some instance of object/background separation. Fast detectors have recently been built based on some type of coarse-to-fine search. The particular ones we train are a variant of these: There is a hierarchy of individual Boolean detectors dedicated to various subsets of presentations, each of which is based on checking for a minimum number of "parts" which are chosen during training from a virtually infinite candidate pool. Our aim is to design the underlying object representations to share as many parts as possible. This opens the way to still faster online execution as well as more efficient representation. It can also reduce the error rates if the parts are extended structures which are significantly rarer in the "background" than on the objects of interest.

Finding a universal "alphabet" of features has of course intrigued researchers for a long time, both in computational and biological vision. The advantages are numerous, including simplifying object recognition and scene parsing by basing the search on the parts, some of which could be precomputed. Ideally, not too many total parts would be needed in order to represent each object in terms of a relatively small number of them together with rough geometric constraints among them [2],[5]. In addition, methods based on stochastic and structural grammars are by nature hierarchical; indeed, the concept of "parts" lies at the heart of "compositional vision" [4]. From the point of view of biological vision, this provides a model for the responses of neurons at higher level retinotopic layers such as V4 and some part of IT [11]. Previous attempts have focused on pre-designed parts (see e.g. [5] and [8]). In contrast, our approach is statistical, learning-based and recursive.

Our parts are medium-scale, binary features which are defined in terms of flexible and extended configurations of edges. The learning scenario is sequential: new object classes are continually being added to an existing library. In fact, we imagine beginning with a single object class and successively adding one new class at each iteration. The objective is then to modify the existing system in order to accommodate the new category. We simply build a detector for the new class and add it to the existing collection. When the detector for the $k+1$'st class is constructed, the parts

which constitute the previous k detectors are examined first and favored in the construction of the new one. Only if the quota of necessary parts is not reached in this way are new parts learned from data. Initially, most parts are learned and unique; eventually, many are reused in new classifiers. Ideally, the total number of distinct parts in the system after k constructions will grow slowly, for instance logarithmically, with k, which directly reduces storage and opens the way to online efficiency by basing scene parsing on the parts rather than the individual detectors.

We describe some steps along this path. The parts are described in §2 and the method for integrating them into a detector is outlined in Section 3. Obvious tradeoffs emerge among measures of reusability, discrimination and part complexity. The sequential learning algorithm is explained in Section 4 and experiments in character recognition, specifically detecting LaTeX symbols, are presented in Section 5. Finally, in Section 6, we mention some open issues, for instance the scaling behavior and exploiting redundancy to minimize online computation.

2 The Ensemble of Parts

As indicated above, the parts we explore are different from those in the cited references, being inherently discrete, learned from data rather than designed (or specified analytically) and chosen based purely on statistical criteria. Specifically, we seek:

- *A rich pool of potential parts - geometrically and statistically diverse*;
- *A modestly-sized subset which provides a balance between commonality and discrimination (object specificity)*;
- *A degree of geometric and photometric invariance*.

Global features are therefore not appropriate since they are unlikely to generalize and too sensitive to occlusion and clutter; they are also unsuitable for the particular types of coarse-to-fine detectors we design since, even for a single object category, we require features which are simultaneously likely over a wide range of poses. Local features are not sufficiently discriminating, at least if we desire to construct our detectors from a relatively small number of parts. Consequently, with all these constraints in mind, we use "mid-scale" features constructed themselves from semi-invariant local features.

The only local features we use are edges, but others might serve equally well as long as local topographic information is expressed in manner largely invariant to photometry and small geometric perturbations. We have used both a "home-grown" edge filter and converted a state-of-the-art edge-detector [6] into a binary feature after quantizing the gradient to 4 canonical directions. Fig 1 illustrates the edges found on an arc. For our purposes, the differences among edge operators is unimportant; the key operation is "spreading", which generates a cascade of local features over many "scales," and hence many levels of invariance and statistical power.

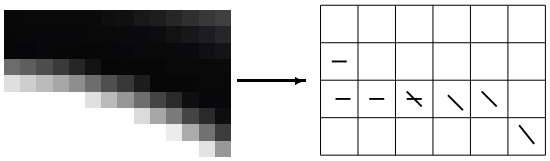


Figure 1: *The edges detected on a piece of arc.*

There is one "spread edge" ξ for each location $x \in L$ (the pixel lattice), direction (four values), polarity (light-to-dark or vice-versa) and "spread" $\eta \in \{1, 2, ...\}$:

$\xi = 1$ if there is an edge of the given direction and polarity anywhere along a line of pixels of length η centered at x, and orthogonal to the edge direction; otherwise $\xi = 0$.

We will refer to the line of pixels as the "support" of the spread edge (although the set of all pixels which participate in the definition of the component edges is of course larger). The case $\eta = 1$ corresponds to ordinary edges. We write $\mathcal{E}$ for this collection of binary features. Spread edges have arisen in a variety of forms in recent studies [1], [9], motivated by achieving a desired level of geometric invariance (see §3).

A *part* is a subset $A = \{\xi_1, ..., \xi_M\} \subset \mathcal{E}$ of M spread edges, whose positions are confined to a (reference) $K \times K$ window (centered at the origin) and whose spreads are identical, say $\eta(A)$. Given a location $x \in L$, the corresponding binary feature is

$$X_A(x) = \begin{cases} 1 & \text{if } \sum_{\xi \in A} \tau_x \xi \geq m \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $\tau_x \xi$ denotes the shifting of the spread edge ξ by the vector x. To avoid redundant representations of object boundaries we require that the supports of any two spread edges of the same direction be disjoint. The degree of invariance is controlled by η. For instance, if the subimage in Figure 1 undergoes a small translation, rotation and scaling, the set of edges found is roughly the same as before for sufficiently large η, and hence the response of X_A is exactly the same.

We take $M = K$ for convenience. As for reusability, suppose the threshold $1 \leq m \leq M$ is chosen so that A is rather likely to appear (meaning the event $X_A = 1$ has high probability) on instances of some object category over a range of poses. Clearly the potential for X_A to capture geometric properties of other object categories, and hence be

reused, depends critically on M and η: As M increases and η decreases, the part becomes more and more specialized. In all our experiments we choose $m = 5$ and $M = K = 10$ relative to objects whose scale is order 40. In order to promote reusability, we also limit A to at most two types of edges (in terms of direction and polarity).

Fig 2 shows sample parts chosen during training (see §4), all heavily reused; the bounding boxes are 10×10. The graphs in Fig 3 indicate the level of discrimination of one of the (coarse) parts in Fig 2 by showing object and background probabilities for different thresholds m. For example, if the threshold in (1) is $m = 5$, the likelihood of the event $\{X_A = 1\}$ is nearly 0.7 on objects and less than 0.1 on background.

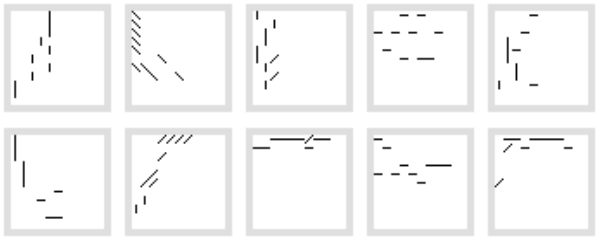


Figure 2: *A sample of parts found during training. Top row : Parts frequently used by pose-invariant detectors. Bottom row : Parts frequently used by pose-specific detectors.*

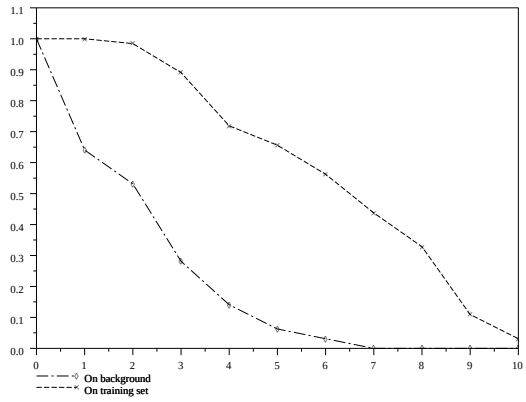


Figure 3: *Probabilities on object and background of one of the sample parts in Fig 2. The horizontal axis represents the number m of spread edges in the part; the vertical axis is the estimated probability of finding the part for different thresholds m in (1), both for object and non-object images.*

3 Object Detectors

The detectors we train are closely related to recent work ([3], [13]) in which Boolean features at different levels of invariance are combined in a hierarchy in order to accelerate scene parsing. The detectors are defined as object vs. non-object classifiers for a reference window, which are then applied at a sparse sublattice of image locations. The cascade nature of the classifier leads to quick rejection of most image regions, allowing the computation to concentrate on object-like structures.

The modular or "compositional" nature of these hierarchical detectors makes them ideal for incorporating the flexible parts we have described. In the cited work, each detector in the hierarchy is based on checking for the existence of certain number of Boolean features (exactly as in our definition of a part). In effect, we are simply grouping these primitive Boolean features (our spread edges) into clusters (our parts) in order to make the detectors more discriminating.

Given a *reference grid* G on which the detector is defined, instead of checking whether $\sum_{\xi \in B} \xi \geq l$ for some "large" family B of edges (as done before), we evaluate

$$f = \begin{cases} 1 & \text{if } \sum_{(A,x)\in\mathcal{D}} X_A(x) \geq n \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

where $\mathcal{D}$ is a family of part/location pairs (A, x) with $x \in G$. Here n is on the order of half $|\mathcal{D}|$. In the current implementation, the locations are all at least K pixels apart. Thus there is no overlap among the supports of the (translated) parts.

In this way we retain the basic advantages of the above-mentioned methods, such as very rapid processing of most portions of the scene (due to early exit from the search) while at the same time improving discrimination and allowing for efficient sequential learning (see §4).

In order to make this paper largely self-contained, and to formulate the learning problem in abstract terms, we briefly describe the structure of the detector hierarchy and the manner in which the scene is parsed to find instances from an object category. All possible poses in the scene are partitioned into (disjoint) subsets; for example, the object position is divided into non-overlapping blocks. (The scale can also be restricted to an interval commencing at the minimum detectable scale and the algorithm rerun on downsampled images to find larger objects.) The global procedure is then to visit a subimage - corresponding in size to the reference grid G - surrounding each block and check for the presence of objects with a pose in the specified subset. This is done using a hierarchy of detectors of the form (2). In the experiments below, the blocks are of size 8×8 and the initial scales run from the minimum size to twice the minimum.

More specifically, assume that *both* an object class or

category c is fixed *and* a subset of poses, denoted Θ. The goal is to build a detector (binary classifier) F_c which responds positively when there is an instance of class c with pose in Θ. The set Θ is recursively partitioned into subsets organized in a tree hierarchy $\{\Omega_{lj}, j = 1, ..., n_l, l = 1, ..., L\}$, where Ω_{lj} is the j'th member of the l'th layer. The children of Ω_{lj} result from subdividing Ω_{lj} based on one of the pose parameters. The details of the recursive partitioning are irrelevant for our purposes. In this scheme, a detector $f_{lj} \in \{0, 1\}$ of the form (2) is learned for each Ω_{lj}. The classifier F_c is defined in terms of $\{f_{lj}\}$: If the root detector $f_{11} = 0$, the search is terminated; if it responds positively, the detectors for $l = 2$ are evaluated; and so forth. In general, f_{lj} is evaluated if and only if all its ancestors are evaluated and return positive answers, and hence the object is declared to be present ($F_c = 1$) if and only if a chain of positive responses is found all the way down to the final layer. (In that case an estimate of the pose is easily provided.)

The motivation for this search design is that each detector f_{lj} is designed to have a null false negative error (no missed detections) in the sense that if there is indeed an instance of object class c with pose in Ω_{lj}, then $f_{lj} = 1$ with probability (close to) 1. This can always be (roughly) satisfied at the expense of false positive error and allows the search to be very quickly terminated in general. (For "invariants" of a very different sort see, for example, the differential ones in [7].)

4 Learning

The abstract problem is to induce a detector $f_{\mathcal{L}}$ with good error statistics from a training set $\mathcal{L}$ (of images). In our work this means that the false positive rate (or type II error) is as small as possible subject to the false negative (type I) generalization error being close to zero: $f_{\mathcal{L}}(I) = 0$ with very low probability for new samples I from whatever "class" $\mathcal{L}$ represents. This particular error tradeoff underlies the rationale for the classifiers described in §3. Needless to say, observing no missed detections on $\mathcal{L}$ is not sufficient; in the applications the threshold n, which controls the tradeoff, is chosen *smaller* than necessary to achieve no missed detections on $\mathcal{L}$, but symbols are still occasionally missed.

The process of learning new parts is simple but computationally intensive. Given $\mathcal{L}$, the first step is to identify a candidate subfamily $\mathcal{E}_0 \subset \mathcal{E}$ of spread edges. The criterion is that ξ is found on at least one-half the images in $\mathcal{L}$, but the same edge with any smaller spread is not. In other words, among "common" spread edges, take the most precise ones possible. Now pass a $K \times K$ window over the image lattice and check for the existence of a part, i.e., the existence of M elements in $\mathcal{E}^0$ meeting the constraints for a part as defined in §2 as well as a "frequency" condition that the part appear on one-half the training set.

Sequential Learning: Now consider a large library $\mathcal{C}$ of object categories, which are incrementally presented to the learning system in some order $\{c_1, c_2, ...\}$. More precisely, at iteration k we obtain a learning set $\mathcal{L}_k$ of samples from category c_k at poses in Θ. Our goal is to build detectors dedicated to various subsets $\Omega \subset \Theta$ (the subsets Ω_{lj} defined in §3). These detectors will be combined (as described in §3) into a Boolean classifier F_{c_k} dedicated to finding instances of class c_k. We can assume from here on that c_k and Ω are fixed, and we are building $f = f_{c_k,\Omega}$. The number $N = |\mathcal{D}|$ of parts in f depends on the range of scales in Ω; in practice, N ranges from about 10 to 25.

At the first step ($k = 0$) we fill our quota N entirely with newly learned parts. Even in this step we try to translate a part X_A attached to location x to new locations and check if the frequency condition is satisfied. At stage k, we wish to maximize the use of existing parts as follows.

Let $\mathcal{P}_k$ be the family of parts which appear in $F_{c_1}, ..., F_{c_{k-1}}$ and set $\mathcal{D} = \emptyset$. The detector $f = f_{c_k,\Omega}$, equivalently $\mathcal{D}$, is learned in four steps:

1. *Evaluate $X_A(x)$ for each location $x \in L$ and each $A \in \mathcal{P}_k$. Retain only those part/location pairs for which the fraction of positive responses exceeds one-half. Call this set of part/location pairs $\mathcal{W}$.*

2. *Choose a random element $(A, x_1) \in \mathcal{W}$. Of all elements in $\mathcal{W}$ with location x_1 find the one (A_1, x_1) with lowest spread $\eta(A_1)$. Add it to $\mathcal{D}$. The subsequent search is restricted to the subset $\mathcal{W}_\eta \subset \mathcal{W}$ of parts with spread η.*

3. *Add to $\mathcal{D}$ a random element (A_i, x_i) in $\mathcal{W}_\eta$ chosen from among those for which x_i is at least K pixels apart from $x_1, \ldots, x_{i-1}$. If none exist go to 4. Otherwise $i = i + 1$ and repeat 3.*

4. *If $|\mathcal{D}| < N$, add $N - |\mathcal{D}|$ new parts and make f.*

The motivation for using the minimal possible spread is discrimination: Given a part, the false positive rate is monotonically increasing with η. Of course if Ω is "large," so that f must be highly invariant, there may be no available parts with small spreads. As η increases, however, more and more parts are found which are common on $\mathcal{L}$. (In practice, only parts found at invariance levels comparable to Ω are likely to be used.) In Fig 4 we show 23 parts appearing in the detector for presentations of the symbol $c = \Theta$ within a relatively large subset Ω.

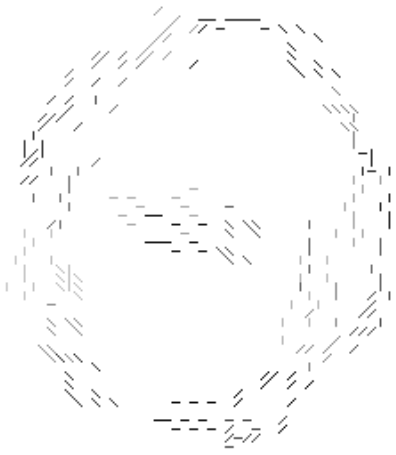


Figure 4: *Parts appearing in the detector for the symbol* Θ *over a range of poses.*

5. Experiments

We have tested the system on a rich family of two-dimensional shapes, namely the LaTeXsymbols. Nothing in the learning algorithm is adapted to this particular shape class; the process would be exactly the same for, say, fish or leaves or mixtures thereof, or for shapes with internal structure, such as faces or logos. However, for highly deformable objects or 3D objects it would clearly be necessary to revisit the basic classifiers in order to accommodate more degrees of freedom in the presentations.

Each LaTeXsymbol is first rendered in high resolution and then normalized in size so that the larger of the dimensions of a bounding box is 64 pixels. Training data for each symbol is synthetically generated for a given set of poses Ω by randomly translating, scaling and rotating the templates. The ordering of the symbols is also random. Samples of parts were shown in Fig 2 and Fig 4; the spreading of the edges is not depicted, but can easily be imagined, so the parts are actually quite flexible features. The somewhat irregular geometry is due to the purely statistical criteria for selection - no "regularity" is assumed other than restricting the edges to at most two orientations.

The degree of reuse is small at the beginning but rapidly accelerates. Indeed, even at the scale of the parts the complexity of this world of shapes is limited: All the symbols are finally composed of a limited number of sub-objects arranged in a variety of spatial configurations. The curve $k \longrightarrow |\mathcal{P}|_k$, depicted in Fig 5. The approximating function is $k \longrightarrow 197 + 84 \ln k$ (least-squares fit). The initial slope is large, but more than half of the total number of parts used for $k = 170$ classes are built in the first 6 iterations.

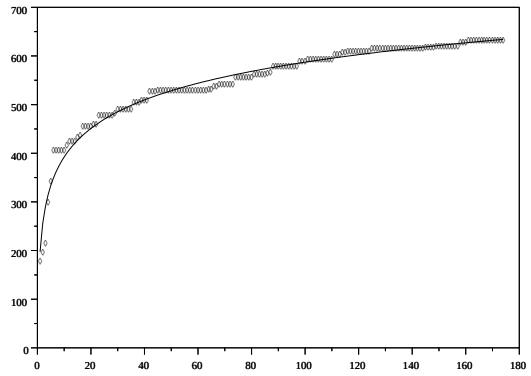


Figure 5: *Growth rate of the number of distinct parts in the system as a function of the number* k *of classes learned, for* $k = 1, ..., 170$*. The superimposed curve is logarithmic.*

The degree of usage varies considerably among parts. Shown in Fig 6 is the usage distribution: The fraction of total parts which are used *at least* j times. Recall that a part may be used in many different detectors for the *same* object class.

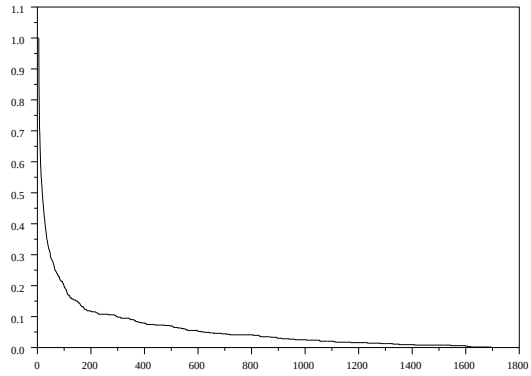


Figure 6: *Distribution of the usage of individual parts. For each* $j \geq 1$*, the graph displays the fraction of parts in the system which at used at least* j *times.*

When artificial "LaTeXscenes" are parsed, there are definitely false positives, particularly for categories learned late in the process since the corresponding detectors tend to be less discriminating than those based entirely on newly-built parts. A sample "scene" together with the detections of the

Φ symbol are shown in Fig 7. This symbol happens to be the first one learned and hence the false positive rate is quite low. Other results are shown in Fig 8 and Fig 9.

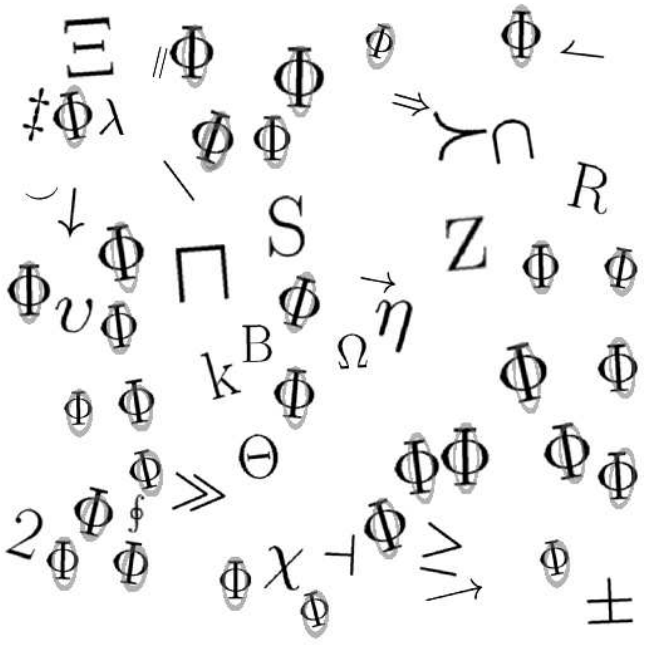


Figure 7: *Results of detecting instances of Φ on a LaTeX-scene.*

6 Discussion and Conclusions

Other sequential learning scenarios could be imagined. For instance, at each iteration only a single new instances (image) might be presented, or a sets of images representing multiple object categories, some new and some previously seen. Indeed, the single example case has been considered in the machine learning literature; see e.g. the "Incremental Tree Inducer" algorithm [12].

The learning algorithm has not been optimized; many ameliorations are possible, both for accelerating the learning and for reducing the false positive rate. In particular, the part selection process is far from optimal: Basically a greedy, recursive search, starting from a given part and randomly adding any one which is disjoint from the previous ones. A more principled approach might be to find the best detector in terms of some objective function which accounts simultaneously for reusability and an estimated false positive rate.

Finally, and most importantly, the redundancy among the detectors has not yet been exploited to reduce online computation. When multiple classes are detected we simply implement the separate detectors one by one. Although detection is quite rapid due to frequent early exit from the hierarchy, steep further gains should be possible. An interesting question, both in practice and theory, is how to exploit the overlap to maximally reduce the amount of online computation. Clearly, rather than loop over detectors, it makes sense to base the search on the parts themselves, for example sequentially evaluating them in a tree-structured decision protocol, thereby taking advantage of *both* the hierarchy *and* the commonality.

References

[1] Y. Amit, *2D Object Detection and Recognition*, M.I.T. Press, 2002.

[2] I. Biederman, "Recognition-by-components: A theory of human image understanding," *Psychological Review*, Vol. 94, pp. 115-147, 1987.

[3] F. Fleuret and D. Geman, "Coarse-to-fine face detection," *Inter. J. Computer Vision*, Vol. 41, pp. 85-107, 2001.

[4] S. Geman, D. Potter and Z. Chi, "Composition systems," *Quarterly of Applied Mathematics*, to appear.

[5] R. M. Haralick, and G. L. Shapiro, *Computer and Robot Vision*, Vol. 2, Addison Wesley, Reading, MA, 1992.

[6] P. Meer and B. Georgescu, *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 23 , pp. 1351-1365, 2001.

[7] J. L. Mundy and A. Zisserman, A., *Geometric Invariance in Computer Vision*, MIT Press, Cambridge, 1992.

[8] M. Parizeau and R. Plamondon, "A fuzzy-syntactic approach to allograph modeling for cursive script recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 17, pp.702-712, 1995.

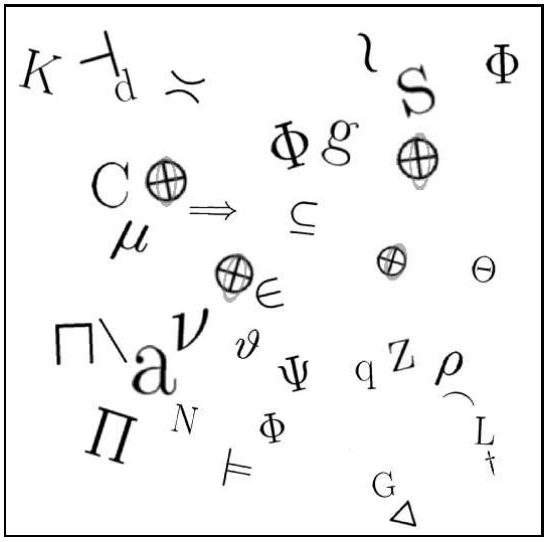
[9] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience* Vol. 2, pp. 1019-1025, 1999.

[10] L. G. Shapiro, "A structural model of shape," *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 2, pp. 111-126, 1980.

[11] K. Tanaka, "Inferotemporal corex and object vision," *Annu. Rev. Neursci.*, Vol. 19, pp. 109-139. 1996.

[12] P. E. Utgoff, N. C. Berkman and A. J. Clouse, "Decision tree induction based on efficient tree restructuring," *Machine Learning*, Vol. 29, pp.5-44, 1997.

[13] P. Viola and M. J. Jones, "Robust real-time object detection," *Intl. J. Computer Vision*, 2002.



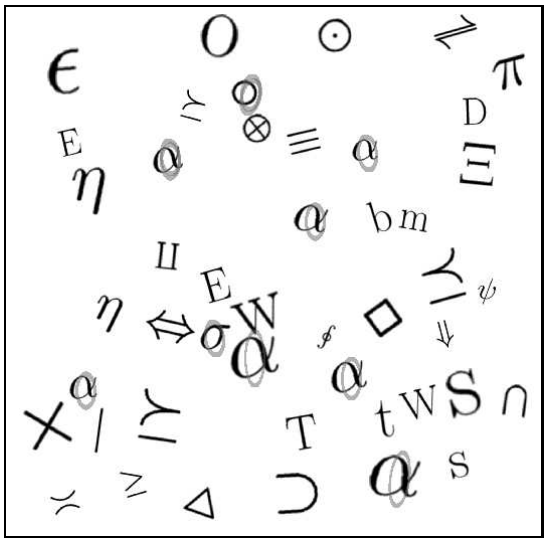
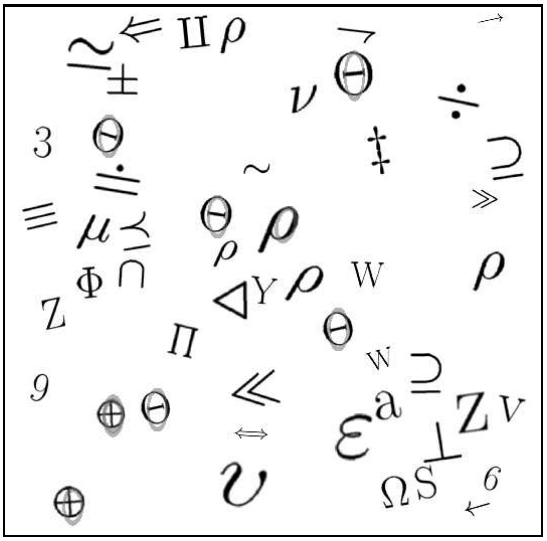


Figure 8: *Examples of detection results for other symbols, displayed to left of the corresponding image. The oval reflects the detected pose in position, scale and tilt.*



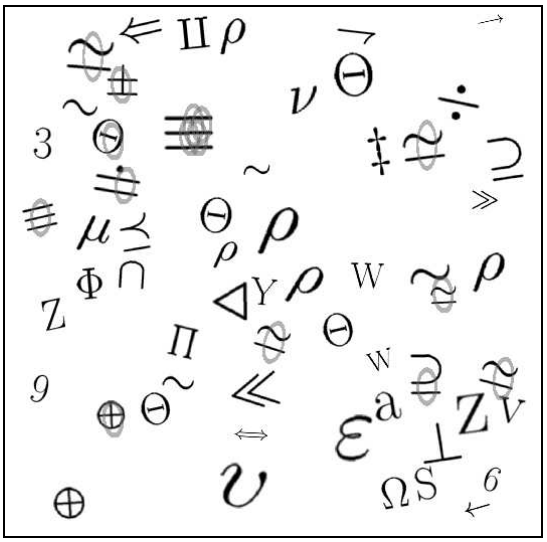


Figure 9: *Additional results.*