Instantiating deformable models with a neural net

Christopher K. I. Williams*, Michael Revow, Geoffrey E. Hinton

Department of Computer Science, University of Toronto

Toronto, Ontario, Canada M5S 1A4

To appear in: Computer Vision and Image Understanding

Submitted August 1995, revised April 1996, accepted August 1996

Address for correspondence:

Dr. C. K. I. Williams

Department of Computer Science and Applied Mathematics

Aston University

Birmingham B4 7ET, UK

tel: +44 121 359 3621 x 4382

fax: +44 121 333 6215

email: c.k.i.williams@aston.ac.uk

*Current address: Department of Computer Science and Applied Mathematics, Aston University, Birmingham B4 7ET, UK.

Abstract

Deformable models are an attractive approach to recognizing objects which have considerable within-class variability such as handwritten characters. However, there are severe search problems associated with fitting the models to data which could be reduced if a better starting point for the search were available. We show that by training a neural network to predict how a deformable model should be instantiated from an input image, such improved starting points can be obtained. This method has been implemented for a system that recognizes handwritten digits using deformable models, and the results show that the search time can be significantly reduced without compromising recognition performance.

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

Figure 1: Illustration of the importance of the starting position in a complex search space. (a) is the initialization obtained using the neural network, and (b) is the final fit after using the short search schedule 3 discussed in section II(B). (c) shows the initalization without the neural network (described in section I(B)) and (d) is the final fit from this starting point, again using search schedule 3. If the search from starting position (c) were continued for longer the loop of the 2 model would unfold to produce a final fit very much like that shown in (b).

Deformable models have been used widely to characterize the shape variability of objects. Their main use has been to obtain a "best fit" match between the model and given data, as in [25], [10], [8], [9], [1], [11] and [28]. This framework can also be extended to tracking objects over several frames (e.g. [5]) and to object recognition ([13], [23], [20]). Deformable models are closely related to the work on active contour models or "snakes" in [18].

A major problem with this framework is that the procedure for fitting a model to an image is very computationally intensive and typically iterative, because there is no efficient algorithm (like dynamic programming) for this task. The key idea of this paper is to use fast methods (such as feedforward neural networks) to obtain a better starting point so that the search time can be reduced significantly. This can be done by "compiling down" some of the knowledge gained while fitting models to data in the slow, iterative fashion. The idea is demonstrated on a handwritten digit recognition task for which we had previously developed a recognizer using deformable models ([13], [23])[1], and is illustrated in Figure 1. This work can be seen as a specific example of "caching" or "compiling down" the results of previous searches to speed up running time.

This paper is structured as follows: Section I describes our deformable models of handwritten

[1]This paper is an expanded and revised version of [27].

digits and shows how they can be used for a digit recognition task. Section II shows how deformable models can be instantiated using neural nets and reviews relevant previous work. Results are presented showing that similar recognition performance can be obtained using the neural network instantiations, but with more than a factor-of-two reduction in the search time required. Section III discusses the results and outlines possible extensions to the method.

I. Deformable models for digit recognition

The basic idea in using deformable models for digit recognition is that each digit has a model, and a test image is classified by finding the model which is most likely to have generated it. The quality of the match between model and test image depends on the deformation of the model and how close the inked pixels lie to the model.

More formally, the two important terms in assessing the fit are the prior probability distribution for the instantiation parameters of a model (which penalizes very distorted models), and the imaging model that characterizes the probability distribution over possible images given the instantiated model[2]. Let I be an image, M be a model and $\boldsymbol{z}$ be its instantiation parameters (such as the translation, rotation and deformations of the model). Then the evidence for model M is given by

$$P(I|M) = \int P(\boldsymbol{z}|M)P(I|M,\boldsymbol{z})d\boldsymbol{z} \tag{1}$$

The first term in the integrand is the prior on the instantiation parameters and the second is the imaging model i.e., the likelihood of the image given the instantiated model. $P(M|I)$ is directly proportional to $P(I|M)$, as we assume a uniform prior on each digit.

Equation 1 is formally correct, but if $\boldsymbol{z}$ has more than a few dimensions the evaluation of this integral is computationally intensive. However, it is often possible to assume that the integrand is strongly peaked around a (global) maximum value $\boldsymbol{z}^*$, and in this case the evidence can be approximated by the highest peak of the integrand times a volume factor $\Delta(\boldsymbol{z}|I,M)$, which measures the sharpness of the peak; this is known as Laplace's approximation.

$$P(I|M) \simeq P(\boldsymbol{z}^*|M)P(I|\boldsymbol{z}^*,M)\Delta(\boldsymbol{z}|I,M) \tag{2}$$

[2]This framework has been used by many authors, e.g. [11].

By Taylor expanding around $\boldsymbol{z}^*$ to second order it can be shown that the volume factor depends on the determinant of the Hessian of $\log P(\boldsymbol{z}, I|M)$. Define E_{def} as the negative log of $P(\boldsymbol{z}^*|M)$, and E_{fit} as the corresponding term for the imaging model. In order to use Laplace's approximation we need to find a peak of $P(\boldsymbol{z}|M)P(I|\boldsymbol{z}, M)$, or equivalently a minimum of $E_{tot} = E_{def} + E_{fit}$. This is done by iteratively refining $\boldsymbol{z}$ until a minimum of E_{tot} is reached at $\boldsymbol{z}^*$. $\Delta(\boldsymbol{z}|I, M)$ is evaluated at $\boldsymbol{z}^*$. Of course the total energy will probably have local minima; for the character recognition task we aim to find the global minimum by using a continuation method (see section I(B)).

A. Splines, affine transforms and imaging models

This section presents a brief overview of our work on using deformable models for digit recognition. For a fuller treatment, see [23].

Each digit is modelled by a cubic B-spline whose shape is determined by the positions of the control points in the object-based frame. The models have eight control points, except for the one model which has three, and the seven model which has five. To generate an ideal example of a digit the control points are positioned at their "home" locations. Deformed characters are produced by perturbing the control points away from their home locations. The home locations and covariance matrix for each model were adapted in order to improve the performance.

A model is transformed from the object-based frame to the image-based frame by an affine transformation which allows translation, rotation, dilation, elongation, and shearing. All possible affine transformations are permitted during the search.

The data we used consists of binary-pixel images of segmented handwritten digits. The general flavour of an imaging model for this problem is that there should be a high probability of inked pixels close to the spline, and lower probabilities further away. This can be achieved by spacing out a number of Gaussian "ink generators" uniformly along the contour; we have found that it is also useful to have a uniform background noise process over the area of the image that is able to account for pixels that occur far away from the generators. The ink generators and background process define a mixture model. Using the assumption that each data point is generated independently given the instantiated model, $P(I|z^*, M)$ factors into the product of the probability density of each black pixel under the mixture model.

Number of generators	Maximum number of iterations	Variance
8	5	0.04
15	3	0.02
	3	0.01
	35	0.008
30	35	0.0025
60	45	0.0006

Table 1: The annealing schedule. The maximum number of iterations refers to the fact that it is wasteful to repeatedly calculate updates if the algorithm has converged at a particular variance. The fractional change in the total energy $\Delta E_{tot}/E_{tot}$ was monitored, and if it fell below a threshold (0.001) the set of iterations was terminated and we moved on to the next variance step. The scale was set so that in the object-based frame each model was one unit high.

B. Recognizing isolated digits

For each model, the aim of the search is to find the instantiation parameters that minimize E_{tot}. The standard (i.e. non-neural) search starts with zero deformations and an initial guess for the affine parameters which scales the model so as to lie over the data with zero skew and rotation. A small number of generators with the same large variance are placed along the spline, forming a broad, smooth ridge of high ink-probability along the spline. We use a search procedure similar to the (iterative) Expectation Maximization (EM) method of fitting an unconstrained mixture of Gaussians, except that (i) the Gaussians are constrained to lie on the spline (ii) there is a deformation energy term and (iii) the affine transformation must be recalculated on each iteration. During the search the number of generators is gradually increased while their variance decreases according to a predetermined "annealing" schedule shown in Table 1[3].

After fitting all the models to a particular image, we wish to evaluate which of the models best "explains" the data. The natural measure is the sum of E_{fit}, E_{def} and the volume

[3]It is also possible to let the EM algorithm effectively determine its own annealing schedule; this approach is described in [23].

factor. However, we have found that performance is improved by including four additional terms which are easily obtained from the final fits of the model to the image. These are (i) a measure which penalizes matches in which there are ink-generators far from any inked pixels, and (ii) the rotation, shear and elongation of the affine transform. It is hard to decide in a principled way on the correct weightings for all of these terms in the evaluation function. We estimated the weightings from the data by training a simple postprocessing neural network. These inputs are connected directly to the ten output units. The output units compete using the "softmax" function [6] which guarantees that they form a probability distribution, summing to one.

II. Predicting the instantiation parameters

The search procedure described above is very time consuming. However, given many examples of images and the corresponding instantiation parameters obtained by the slow method, it is possible to train a neural network to predict the instantiation parameters of novel images. These predictions provide better starting points, so the search time can be reduced.

A. Previous work

Previous work on hypothesizing instantiation parameters can be placed into two broad classes, correspondence based search and parameter space search. In correspondence based search, the idea is to extract features from the image and identify corresponding features in the model. Using sufficient correspondences the instantiation parameters of the model can be determined. The problem is that simple, easily detectable image features have many possible matches, and more complex features require more computation and are more difficult to detect. For example, Grimson's work [12] on matching polyhedral objects to models uses edge features; as every image feature can potentially match every object feature it is necessary to search the space of possible correspondences using an interpretation tree. Other authors [21], [19], [30], [3] have emphasized the grouping of lower-level image elements into higher level features.

An alternative approach is to work directly in parameter space. A simple example of this can be found in [15], where Horn shows how to instantiate a model of a binary image using image

moments. Another parameter space approach is the Hough transform, which was originally designed for the detection of straight lines in images. Subsequently it was extended to cover a number of geometric shapes, notably conic sections and Ballard [2] further extended the approach to arbitrary shapes with the Generalized Hough Transform. The parameter space for each model is divided into cells ("binned"), and then for each image feature a vote is added to each parameter space bin that could have produced that feature. After collecting votes from all image features we then search for peaks in the parameter space accumulator array, and attempt to verify pose. The Hough transform can be viewed as a crude way of approximating the logarithm of the posterior distribution $P(\boldsymbol{z}|I, M)$ (e.g. [16], [24]).

However, these techniques have only been used on problems involving rigid models, and are not readily applicable to the digit recognition problem. For the Hough space method, binning and vote collection is impractical in the high dimensional parameter space, and for the correspondence based approach there is a lack of easily identified and highly discriminative features. Furthermore, the strengths of these two techniques, namely their ability to deal with arbitrary scalings, rotations and translations of the data, and their tolerance of extraneous features, are not really required for a task where the input data is fairly well segmented and easily normalized.

Our approach is to use a neural network to predict the instantiation parameters for each model, given an input image. Zemel and Hinton [29] used a similar method with simple 2-d objects, and more recently, Beymer *et al* [4] have constructed a network which maps from a face image to a 2-d parameter space spanning head rotations and a smile/no-smile dimension. However, their method does not directly map from images to instantiation parameters; they use a computer vision correspondence algorithm to determine the displacement field of pixels in a novel image relative to a reference image, and then use this field as the input to the network. This step limits the use of the approach to images that are sufficiently similar so that the correspondence algorithm functions well.

B. Instantiating digit models using neural networks

The network which is used to predict the model instantiation parameters is shown in figure 2. The binary images are normalized to give 16×16 8-bit greyscale images. The greyscale level of each pixel is computed from the proportion of its area which was inked in the binary

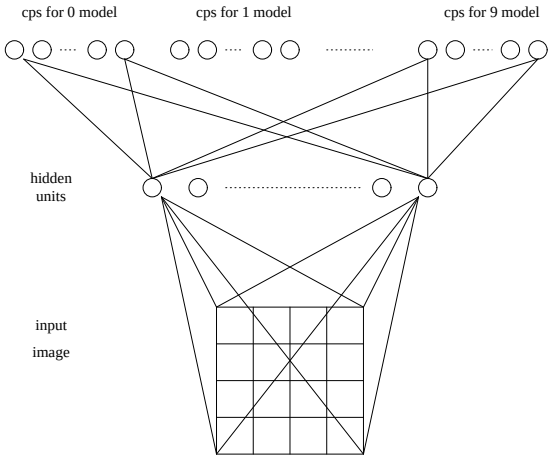

Figure 2: The prediction network architecture. "cps" stands for control points.

image. The network uses a standard three-layer architecture; each hidden unit computes a weighted sum of its inputs, and then feeds this value through a sigmoidal nonlinearity $\sigma(x) = 1/(1 + e^{-x})$. The output values are a weighted linear combination of the hidden unit activities plus output biases. The targets are the locations of the control points in the normalized image, found from fitting models as described in section I(B). Thus there are 16 outputs for each of the ten models ((x, y) locations for each of the 8 control points), except for the one and seven models which have 6 and 10 outputs respectively.

The network was trained with backpropagation to minimize the squared error, using 900 training images and 200 validation images of each digit drawn from the *br* set of the CEDAR CDROM 1 database of Cities, States, ZIP Codes, Digits, and Alphabetic Characters[4]. Two test sets were used; one was obtained from other data in the *br* dataset, and the other was the (official) *bs* test set. These test sets contained 2000 and 2711 images respectively. When training a neural network it is important to control the complexity of the net if a good generalization performance is to be obtained on novel inputs. In our case the complexity was adjusted by changing the number of hidden units and also be using a weight penalty $\lambda \sum_j w_j^2$, which prevents the weights from becoming too large. We experimented with nets using different numbers of hidden units and λ values and chose $\lambda = 1$ and 20 hidden units based on the performance on the validation set. The net took 440 epochs to train to convergence using the default conjugate gradient search method in the Xerion neural network simulator version 3.1[5]. Targets were only set for the correct digit at the output layer; nothing was

[4]Made available by the Unites States Postal Service Office of Advanced Technology.

[5]Xerion was designed and implemented by Drew van Camp, Tony Plate and Geoffrey Hinton at the

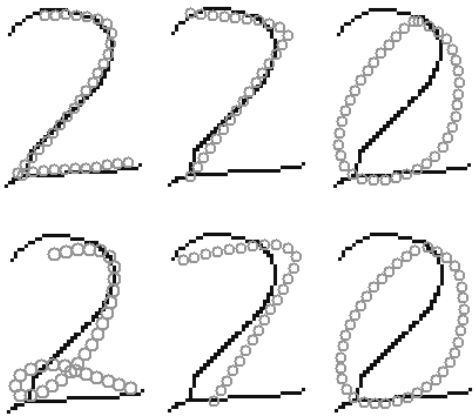

Figure 3: A comparision of the initial instantiations due to the prediction net (top row) and the trivial net (bottom row) on an image of a 2 (which has been thinned for display purposes). Notice that for the two model the prediction net is much closer to the data. The other digit models may or may not be greatly affected by the input data; for example, the predictions from both nets seem essentially the same for the zero, but for the seven the prediction net puts the model nearer to the data.

backpropagated from the other output units. It would be possible to construct ten separate networks to carry out the same task as the net described above, but this would intensify the danger of overfitting, which is reduced by giving the network a common pool of hidden units which it can use as appropriate.

For comparison with the *prediction net* described above, a trivial network which just consisted of output biases was trained; this network simply learns the average value of the control point locations. On a validation set the squared error of the prediction net was over three times smaller than the trivial net. Although this is encouraging, the acid test is to compare the performance of elastic models settled from the predicted positions using a shortened annealing schedule; if the predictions are good, then only a short amount of settling will be required.

The feedforward net predicts the positions of the control points in the normalized image. By inverting the normalization process, the positions of the control points in the un-normalized image are determined. The model deformation and affine transformation corresponding to these image control point locations can then be determined by running a part of one iteration

University of Toronto. It is available by anonymous ftp from `ftp.cs.utoronto.ca/pub/xerion`

of the search procedure. Experiments were conducted with a number of shortened annealing schedules; for each one, data obtained from settling on a part of the training data was used to train the postprocessing net. The performance was then evaluated on the *br* test set.

The full annealing schedule has six stages. The shortened annealing schedules are:

1. No settling at all
2. Two iterations at the final variance of 0.0006
3. One iteration at 0.0025 and two at 0.0006
4. The full annealing schedule (for comparison)

The results on the *br* test set are shown in table 2. The general trends are that the performance obtained using the prediction net is consistently better than the trivial net, and that longer annealing schedules lead to better performance. A comparison of schedules 3 and 4 in table 2 indicates that the performance of the prediction net/schedule 3 combination is similar to (or slightly better than) that obtained with the full annealing schedule, and is more than a factor of two faster. The results with the full schedule are almost identical to the results obtained with the default "box" initialization described in section I(B). Figure 3 compares the outputs of the prediction and trivial nets on a particular example. Judging from the weight vectors and activity patterns of the hidden units, it does not seem that some of the units are specialized for a particular digit class.

A run on the *bs* test set using schedule 3 gave an error rate of 4.76 % (129 errors), which is very similar to the 125 errors obtained using the full annealing schedule and the box initialization. A comparison of the errors made on the two runs shows that only 67 out of the 129 errors were common to the two sets. This suggests that, if accuracy is more important than speed, it would be very sensible to run both methods and reject cases where they do not agree.

III. Discussion

The prediction net used above can be viewed as an interpolation scheme in the control point position space of each digit. For each digit model the predicted position in the control point

Schedule number	Trivial net	Prediction net	Average time required to settle one model (s)
1	427	200	0.12
2	329	58	0.25
3	160	32	0.49
4	40	36	1.11

Table 2: Errors on the internal test set of 2000 examples for different annealing schedules. The timing trials were carried out on a R-4400 machine.

space $\boldsymbol{\zeta}(I)$ is given by

$$\boldsymbol{\zeta}(I) = \boldsymbol{\zeta}_0 + \sum_i^N a_i(I)\boldsymbol{\zeta}_i \tag{3}$$

where $\boldsymbol{\zeta}_0$ is the contribution due to the biases, a_i is the activity of hidden unit i and $\boldsymbol{\zeta}_i$ is the weight vector from hidden unit i to the linear output units[6]. Exactly the same equation would apply to a Radial Basis Function (RBF) network [7], [22], except in this case the hidden unit activities are computed as a Gaussian function of the distance between the input pattern and a centre *in the input space*. Indeed, any function approximator (such as sigmoidal networks or RBFs) which has "universal approximation" capabilities could be used for this task.

If there are more hidden units than output dimensions, then for any particular image there are an infinite number of linear combinations of the weight vectors $\boldsymbol{\zeta}_1, \ldots, \boldsymbol{\zeta}_N$ so that their sum will match the target vector. However, the network will tend to find solutions so that the $a_i(I)$'s in equation 3 will vary smoothly as the image is perturbed.

The output representation of the prediction net could be enhanced, so that not only $\boldsymbol{\zeta}(I)$ but also the variance of each control point is predicted, allowing the posterior uncertainty of the control point positions to be represented. This network could be trained in a similar fashion as before, except that the negative log likelihood of the target data would be minimized, rather than the squared error[7]. A large predicted variance for a control point may imply that

[6] We have used $\boldsymbol{\zeta}$ to denote a location in control point position space to distinguish it from $\boldsymbol{z}$ used for the instantiation parameters. This distinction is necessary because $\boldsymbol{\zeta}$ gives the coordinates of the control points in the normalized image, whereas $\boldsymbol{z}$ stands for the deformations and affine transformation parameters described in section I. $\boldsymbol{\zeta}$ and $\boldsymbol{z}$ can be related by inverting the normalization process.

[7] Minimizing the squared error is equivalent to minimizing the negative log likelihood if the variance is

there are two or more competing hypotheses as to where the control points should be, and the length of the search could be increased in this case relative to cases where the predictions are less uncertain. Control point uncertainty can be thought of as inducing variance on the ink generators (as each generator position is a linear combination of the control point positions) so that the effective variance of a generator is the sum of the imaging-model variance and the variance contributed by uncertainty in the control point positions.

The nets described above output just one set of instantiation parameters for a given model. However, it may be preferable to be able to represent a number of guesses about model instantiation parameters; one way of doing this is to train a network that has multiple sets of output parameters, as in the "mixture of experts" architecture of Jacobs *et al* [17]. The outputs can be interpreted as a mixture distribution in the control point position space, conditioned on the input image.

The mixture of experts approach models the posterior distribution with a fixed number of Gaussians whose means, variances and mixing proportions are adaptive. An alternative approach is to use a fixed set of basis functions $\{\phi_i\}$, and to approximate the posterior distribution (or some function of it) by the linear combination $\sum_i a_i(I)\phi_i(\boldsymbol{z})$ [14]. For example, the Hough transform uses indicator functions of the bins as basis functions, although it would be possible to use other classes of functions, e.g. Gaussians. When using fixed basis functions, a further step is necessary in order to decode the pattern of activities; we have to go "bump hunting" in the instantiation parameter space to find the (local) maxima of the posterior density. There are many issues involved with the details of this kind of scheme, concerning the density of units in the space (which in turn affects the minimum resolvable separation of two instantiations), and the method used to decode the activity patterns.

The strategies described above directly predict the instantiation parameters in parameter space. It is also possible to use neural networks to hypothesize correspondences, i.e. to predict an inked pixel's position on the spline given a local window of context in the image. With sufficient matches it is then possible to compute the instantiation parameters of the model. We have conducted some preliminary experiments with this method (described in [26]), which indicate that good performance can be achieved for the correspondence prediction task.

---

constant.

We have shown that we can obtain significant speedup using the prediction net. The schemes outlined above which allow multimodal predictions in instantiation parameter space may improve performance and deserve further investigation. We believe that using machine learning techniques like neural networks to help reduce the amount of search required to fit complex models to data may be useful for many other problems.

Acknowledgements

This research was funded by Apple and by the Ontario Information Technology Research Centre. We thank Allan Jepson, Richard Durbin, Rich Zemel, Peter Dayan, Rob Tibshirani and Yann Le Cun for helpful discussions. Geoffrey Hinton is the Noranda Fellow of the Canadian Institute for Advanced Research.

References

[1] R. Bajcsy, R. Lieberson, and M. Reivich. A computerized system for the elastic matching of deformed radiographic images to idealized atlas images. *Journal of Computer Assisted Tomography*, 7(4):618–625, 1983.

[2] D. H. Ballard. Generalizing the Hough transfrom to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[3] J. S. Beis and D. G. Lowe. Learning Indexing Functions for 3-D Model-Based Object Recognition. In *AAAI Fall 1993 Symposium on Machine Learning in Computer Vision*, pages 50–54, 1993. Proceedings available as AAAI Tech Report FSS-93-04.

[4] D. Beymer, A. Shashua, and T. Poggio. Example Based Image Analysis and Synthesis. AI Memo 1431, AI Laboratory, MIT, 1993.

[5] A. Blake, R. Curwen, and A. Zisserman. A Framework for Spatiotemporal Control in the Tracking of Visual Contours. *International Journal of Computer Vision*, 11(2):127–145, 1993.

[6] J.S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fougelman-Soulie and J. Herault, editors, *NATO ASI series on systems and computer science*. Springer-Verlag, 1990.

[7] D. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1988.

[8] D. J. Burr. A dynamic model for image registration. *Comput. Graphics Image Process.*, 15:102–112, 1981.

[9] D. J. Burr. Elastic matching of line drawings. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 3(6):708–713, 1981.

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

[11] U. Grenander, Y. Chow, and D. M. Keenan. *Hands: A pattern theoretic study of biological shapes*. Springer-Verlag, 1991.

[12] W. E. L. Grimson. *Object recognition by computer.* MIT Press, Cambridge, MA, 1990.

[13] G. E. Hinton, C. K. I. Williams, and M. D. Revow. Adaptive elastic models for hand-printed character recognition. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*. Morgan Kauffmann, 1992.

[14] G. E. Hinton, C. K. I. Williams, and M. D. Revow. Combinining two methods of recognizing hand-printed digits. In I. Aleksander and J. Taylor, editors, *Artificial Neural Networks 2*. Elsevier Science Publishers, 1992.

[15] B. K. P. Horn. *Robot Vision.* MIT Press, Cambridge, MA, 1986.

[16] D. J. Hunt, L. W. Nolte, and W. H. Ruedger. Performance of the Hough Transform and its Relationship to Statistical Signal Detection Theory. *Computer Vision, Graphics and Image Processing*, 43:221–238, 1988.

[17] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1), 1991.

[18] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *Proceedings of the First International Conference on Computer Vision*, Washington, D. C., 1987. IEEE Computer Society Press.

[19] Y. Lamdan and H. J. Wolfson. Geometric Hashing: A General and Efficient Model-Based Recognition Scheme. In *Proceedings of the Second International Conference on Computer Vision.* IEEE, 1988.

[20] A. Lanitis, C. J. Taylor, and T. F. Cootes. A generic system for classifying variable objects using flexible template matching. In J. Illingworth, editor, *Proceedings of the British Machine Vision Conference*, volume volume 1, pages 329–338. BMVA Press, 1993.

[21] D. G. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer, Boston, 1985.

[22] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of IEEE*, 78:1481–1497, 1990.

[23] M. D. Revow, C. K. I. Williams, and G. E. Hinton. Using generative models for hand-written digit recognition, 1994. Submitted to *IEEE Transactions on Pattern Analysis and Machine Intelligence* August 1994, accepted for publication March 1996. Available from `http://www.cs.toronto.edu/~revow/pami.ps.Z`.

[24] R. S. Stephens. Probabilistic approach to the Hough transfrom. *Image and vision computing*, 9(1):66–71, 1991.

[25] B. Widrow. The 'rubber-mask' technique–I. Pattern Measurement and Analysis. *Pattern Recognition*, 5:175–197, 1973.

[26] C. K. I. Williams. *Combining deformable models and neural networks for handprinted digit recognition.* PhD thesis, Dept. of Computer Science, University of Toronto, 1994.

[27] C. K. I Williams, M. D. Revow, and G. E. Hinton. Using a neural net to instantiate a deformable model. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems, Vol. 7.* MIT Press, 1995.

[28] A. L. Yuille. Deformable templates for face recognition. *Journal of Cognitive Neuroscience*, 3(1):59–70, 1991.

[29] R. S. Zemel and G. E. Hinton. Discovering viewpoint-invariant relationships that characterize objects. In R. P. Lippmann, J. E. Moody, and D. S. Touretzky, editors, *Advances In Neural Information Processing Systems 3*, pages 299–305. Morgan Kaufmann Publishers, 1991.

[30] R. S. Zemel, M. C. Mozer, and G. E. Hinton. TRAFFIC: Recognizing objects using hierarchical reference frame transformations. In D. S. Touretzky, editor, *Neural Information Processing Systems, Vol. 2*, pages 266–273. Morgan Kaufmann, San Mateo, CA, 1990.