Tina Memo No. 1993-002
Proc. of IEE conference on neural networks, Brighton, 1993.

# A Practical View Based 3D Object Recognition System.

A.C.Evans, N.A.Thacker and J.E.W.Mayhew.

Last updated
5 / 12 / 2003


Imaging Science and Biomedical Engineering Division,
Medical School, University of Manchester,
Stopford Building, Oxford Road,
Manchester, M13 9PT.

# A PRACTICAL VIEW BASED 3D OBJECT RECOGNITION SYSTEM

AC Evans NA Thacker JEW Mayhew

University of Sheffield

## ABSTRACT

This paper discusses the application of a self-organizing neural network to the problem of constructing adequate, $2D$ appearence-based models of $3D$ objects. The problem of basing recognition on such models is characterised as that of approximating the multivariate function mapping shapes of an object to its identity. The domain of this function is described as a set of hypersurfaces in the space of possible shape representations. A novel representational scheme is introduced based on storing values of geometric relationships between pairs of shape features in a histogram. A network architecture capable of constructing function approximations based on such shape representations is presented. Finally, methods for assessing the accuracy of approximation are detailed.

## 1 INTRODUCTION

This paper discusses the application of a self-organizing neural network to the problem of constructing adequate, $2D$ appearence-based models of $3D$ objects. Such models are then used to recognise objects from the $2D$ shapes found in single monochrome images. The difficulty in performing recognition on such data is in dealing with the variations in shape caused by object transformations and image noise. This problem is made more difficult if the models used do not explicitly represent the $3D$ structure of an object. Basing recognition on models composed only of a small number of $2D$ shapes requires the system to generalise its recognition to shapes that do not exactly match those stored. Given the significant scope for variation, and the impracticality of storing all possible shapes, this represents a major problem.

Recent work by Ullman & Basri[1] and Poggio & Edelman[2] has provided a mathematical basis for the $2D$ approach by characterising the generalisation problem as that of approximating the multivariate function mapping the appearence of an object to its identity. Specifically, [2] explores the application of a self-organizing neural network to the problem of learning such approximations from small numbers of example shapes. While demonstrating the potential of the approach, both systems make a number of simplifications which limit their application to artificially generated data. The present work is an attempt to extend the idea of multivariate function approximation to deal with real image data, and therefore provide a practical, $2D$ view-based recognition system.

## 2 SHAPE VARIATION

We require an approximation to the function that maps the appearence of an object to its identity. The domain of this function is therefore the set of shapes produced by viewing an object under all possible viewing conditions. It will be useful to examine the source of this variation by considering the process by which $2D$ shapes are formed. It will then be possible to discuss representation of shape and the formation of the function domain in the space of possible shape representations.

### 2.1 Projection

The $2D$ shape found in an image is the result of the projection of a set of object points onto the image plane of the camera. This process can be approximated by the general projective transformation $p$, taken here to be a perspective projection. For a given object $o$ the shape $s$ produced by $p$ depends on two factors, the viewing parameters $\boldsymbol{\lambda}$ and a noise term $\eta$.

$$p : (o, \boldsymbol{\lambda}, \eta) \rightarrow s \tag{1}$$

The vector $\boldsymbol{\lambda}$ describes the position and orientation of $o$ relative to the viewing camera. If we restrict the camera to be foveated on the object, then this relationship can be fully described by 4 parameters; 3 orientation and 1 distance. The noise term $\eta$ describes variations in $s$ caused by changes in lighting and/or occlusion by other objects. It will be useful at this stage to distinguish between the set of shapes $S_\lambda$, obtained by applying projection $p$ to $o$ for all possible values of $\boldsymbol{\lambda}$, and $S_\eta$, the set of shapes produced by applying the noise model $\eta$ to $S_\lambda$.

### 2.2 Shape Representation

We have said that the domain of the required function is the set of $2D$ shapes produced by an object. Actually, given that $s \in S_\lambda$ is simply a collection of $2D$ features, we require an encoding of $s$ in the form of a vector. The construction of such an encoding can be described in general terms by the representational

function $r$, which takes a set of $2D$ features $s$ and returns a vector $\mathbf{d}$, the components of which should be related to some characteristic of $s$.

$$r : s \rightarrow \mathbf{d} \tag{2}$$

For example, if $s$ is a set of $2D$ point features then $r$ could simply produce an ordered vector of their $x-y$ coordinates, $(x1, y1, x2, y2....xn, yn)$, [1][2]. However, given that this requires full featural correspondences to be known its application to real imaging situations is obviously limited.

### 2.3 Hypersurface Generation

We now define a composite function $f = r \circ p$, that defines a mapping between $\Lambda^4$, the space of possible viewing parameters, and the representational space $D^n$. If we now consider a fixed object $o$, viewed under conditions of zero noise, ie $\eta = 0$, then $f$ maps each point in $\Lambda^4$ to a point in $D^n$ **Eq. 3**. The application of **Eq. 3** to all points in $\Lambda^4$ generates, via the intermediate set of shapes $S^\lambda$, a set of points $H$ in $D^n$ that can be said to lie on a hypersurface, **Eq. 4**.

$$f : (o, \lambda) \rightarrow \mathbf{d} \tag{3}$$

$$H = f(o, \Lambda^4) \tag{4}$$

## 3 HYPERSURFACE BEHAVIOUR

A hypersurface can be thought of as a model of an object in the space of possible shape representations. The form of each hypersurface greatly affects the ease with which recognition can be performed. Before addressing the problem of recognition based on such models it will be useful to expand on our description of hypersurface generation by considering a number of hypersurface characteristics, together with the factors affecting them.

**Smoothness.** The smoothness of a hypersurface depends on the behaviour of the representational function $r$. If $r$ is well-behaved, in the sense that a small change in $s$ produces a small displacement of $\mathbf{d}$, then the hypersurface will be locally smooth. This is essential for accurate approximation.

**Dimensionality.** Although $H$ exists in the high-dimensional representation space $D^n$, it need not itself be n-dimensional. Provided that $r$ is well-behaved, $H$ can be considered as a low-dimensional manifold embedded in the high-dimensional space of possible shape representations[1]. We can actually be more precise than this and say that, in general, $H$ will locally have as many dimensions as there are degrees of freedom in the viewing parameters $\lambda$, minus any invariances that $r$ may have.

**Invariances.** The use of a function $r$ which is invariant to changes in $\lambda$ along some dimension, eg rotation about the camera axis, will locally reduce the dimensionality of the hypersurface. An extreme example of this observation is provided in Forsyth et al.[3]. This shows that for planar objects there exists a function $r$ which is invariant to changes in all 4 viewing parameters. The use of such a function would produce a "hypersurface" consisting of a zero dimensional point.

**Symmetry.** The possibility of symmetries within an object implies that **Eq. 1** may represent a many-to-one mapping. In the projection of a cube, for instance, all possible shapes can be obtained from a sub-region of $\Lambda^4$ corresponding to views from a single octant of the space surrounding the cube. Additionally, emergent symmetries may be created by the behaviour of $r$. For example, if $r$ is invariant to rotations about the camera axis then certain views of a cube from *within* an octant are treated as being symmetric and mapped to a common point in $D^n$. The effect of these real and emergent symmetries is to collapse the hypersurface and cause it to intersect with itself.

**Self-Occlusion.** If objects are constrained to be transparent, [2], then all object points are visible for all values of $\lambda$. Provided $r$ is well-behaved the hypersurfaces of such objects will be continous. The use of opaque objects on the other hand means that $r$ must deal with rapid changes in the set of visible object features caused by self-occlusion. If $r$ is sensitive to the presence of shape features in $s$ then the hypersurface may well be quite disjoint and grouped into regions corresponding to the notion of an aspect, Koenderink & van Doorn[4]. Within these regions, where the set of visible object features is quantitatively unchanging, the hypersurface should be smooth.

**Noise.** Our description of hypersurface generation has thus far been limited to the set $S_\lambda$. We now address the effect of noise on this process by considering the mapping by $r$ of $S_\eta$ to $D^n$. Let $s \in S_\lambda$ be mapped to a point $\mathbf{c}$ in $D^n$. Let $A \subset S_\eta$ be the set of shapes produced by applying $\eta$ to $s$. What is the mapping of $A$ to $D^n$? If $r$ is well-behaved under noise, in the sense that a small corruption of $s$ produces a small displacement of $\mathbf{c}$, then $A$ is mapped to a "cloud" of points $C$, centred on $\mathbf{c}$. The variance of $C$ is obviously related to the degree to which $r$ is well-behaved under noise, ideally $C \equiv \mathbf{c}$. The expansion of each point $c \in H$ into a cloud of points will therefore cause $H$ to "thicken".

We can now say that depending on the object and the behaviour of $f$, each hypersurface will be a locally smooth, low-dimensional manifold which may be self-intersecting, disjoint and possibly thickened by noise. We now turn to the problem of recognition based on such hypersurfaces.

## 4 RECOGNITION

The application of **Eq. 4** to the set of known objects $O : \{o_1...o_m\}$ generates a set of hypersurfaces $\mathbf{H} : \{H_1...H_m\}$ in $D^n$. We now define a function $q$,

the inverse of $f$, which maps each vector $\mathbf{d} \in H_i$ to the identity $o_i$ of an object, **Eq. 5**. The set $\mathbf{H}$ therefore defines the domain of $q$.

$$q : \boldsymbol{d} \in H_i \rightarrow o_i \tag{5}$$

An important factor in the ability of $q$ to perform recognition is the degree to which individual hypersurfaces in $\mathbf{H}$ are seperated. If two hypersurfaces intersect at a point $\mathbf{k} \in D^n$, ie $f(o_i, \lambda) \equiv f(o_j, \lambda') \equiv \mathbf{k}$, then given an approximation to $q$, recognition based on shapes whose representations fall in the region of $\mathbf{k}$ must necessarily be ambigous. This implies that $q$ should be revised to provide probabilistic, rather than absolute, classifications. If we consider recognition as a 1 from M problem then the distribution of each $H_i$ provides an indicator of the probability density function of $o_i$ in $D^n$. We therefore require, for each object $o_i$, the Bayesian *a posteriori* probability $P(o_i|\mathbf{d})$ **Eq. 6**.

$$q_p : \mathbf{d} \rightarrow \begin{pmatrix} P(o_0|\mathbf{d}) \\ . \\ P(o_m|\mathbf{d}) \end{pmatrix} \tag{6}$$

The advantage of this probabilistic approach over that proposed in [2] is that it provides principled behaviour in the presence of object symmetries, ambiguities between objects and thickening of hypersurfaces by image noise.

## 5 SYSTEM ARCHITECTURE

The proposed recognition system requires two components: a representational scheme $r$ for robustly representing the 2D shapes on which both models and recognition are based and a neural network classifier for approximating the probabilistic function $q_p$.

### 5.1 Pairwise Geometric Histograms

We require a representational scheme $r$ which takes a set of $2D$ shape features $s$ and produces a vector $\mathbf{d}$ that encodes some characteristic of $s$. The proposed scheme is based on recording the geometric relationships between pairs of $2D$ shape features in a histogram, Evans et al.[5]. A description of shape in terms of a set of $2D$ line segments is obtained by performing a linear approximation of the edge strings detected in the image. The geometric relationship between line segments can be defined to a chosen degree of accuracy by a set of local geometric constraints. Two binary constraints are used, one based on angle and the other on the distance between line segments. We now define a function $g$ which returns the angle $\theta$ and distance $\delta$ between two line segments $l_i$ and $l_j$.

$$g : (l_i, l_j) \rightarrow \theta_{ij}, \delta_{ij} \tag{7}$$

The set of angles formed between all possible pairs of lines is denoted by $\Theta$, **Eq. 8**. Similarly for the set of distances $\Delta$. We now define a function $e$ that takes the sets $\Theta$ and $\Delta$ and returns a histogram $\mathbf{b}$ of these values, **Eq. 9**. Since two constraints are used to define geometric relationships the histogram $\mathbf{b}$ must have two axes. The importance of the lines in forming the shape is taken into account by setting the value of an entry equal to $|l_i|.|l_j|$. Uncertainty in the exact position and orientation of lines is encoded by blurring entries along both axes of the histogram.

$$\Theta = \{\theta_{ij} : \forall i, j\} \tag{8}$$

$$e : (\Theta, \Delta) \rightarrow \mathbf{b} \tag{9}$$

The constraint values used in this scheme are invariant to rotation of the camera about its axis. This effectively reduces the complexity of the hypersurfaces generated. The advantage of recording constraint values in a histogram is that once all entries have been made it provides statistical evidence for the presence of pairwise geometric relationships within a shape. This ensures a fair degree of robustness to image noise [5]. Although we have talked of $\mathbf{b}$ as being a histogram, it will be convenient now and in later sections to treat it as a vector. Accordingly the final stage of the scheme is to normalise the length of $\mathbf{b}$. This has the effect of collapsing hypersurfaces onto the unit hypersphere and ensures that decisions made by the subsequent network are based purely on shape.

### 5.2 Network Structure

The network used to perform probabilistic classification has two layers, a *shape representation layer* ($V_l$) and an *object recognition layer* ($O_l$), **Fig 1**. The purpose of $V_l$ is to distribute units among the domain of the recognition function $q_p$, ie the hypersurfaces $\mathbf{H}$, in the space of possible shape representations $D^n$. The object layer $O_l$ then encodes the value or *height* of $q_p$ in the region of $D^n$ covered by each unit in $V_l$. We shall now look at the behaviour of each layer in more detail.

**Shape Representation Layer.** The input to $V_l$ is a set of unlabelled shape vectors $I$ taken randomly, but with equal probability, from the set of hypersurfaces $\mathbf{H}$. Units in $V_l$ are trained in an unsupervised manner using a winner-takes-all mechanism common to many systems, Thacker & Mayhew[6]. If the current normalised input vector is $\mathbf{b} \in I$ then the activation $a_i$ of unit $u_i$ in $V_l$ is given by **Eq. 10**.

$$a_i = \textstyle\sum_{j=1}^{n} \mathbf{w}_{ij}.\mathbf{b}_j \tag{10}$$

$$\Delta \mathbf{w}_{ij} = \alpha(\mathbf{b}_j - \mathbf{w}_{ij}) \tag{11}$$

The unit with maximum $a_i$ is taken to be the winning unit and is updated according to **Eq. 11**, where $\alpha$ is a variable that determines the learning rate of the

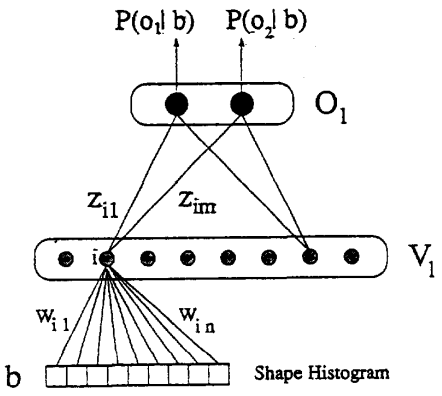


Figure 1: Network Architecture

system. This is gradually reduced throughout presentation of the training data. Weights do not need to be re-normalised after training as they will tend to converge to the normalised input patterns. A conscience mechanism is employed to ensure that during training, units in $V_l$ distribute themselves throughout $D^n$ so as to cover the statistical variation in the hypersurfaces $\mathbf{H}$, as sampled by the examples $I$. Once trained, the units in $V_l$ divide the projection of $\mathbf{H}$ on the unit hypersphere into a Voronoi tesselation, with each unit responding to inputs from a fixed region. The size of each region is inversely proportional to the number of units in $V_l$. The importance of this relation will become clear later.

**Object Recognition Layer.** Given that hypersurfaces in $\mathbf{H}$ are distributed throughout $D^n$, and may even intersect, there is the possibility that input points falling within the region of a unit in $V_l$ may represent shapes from more than one object. The purpose of $O_l$ is to encode the value or *height* of the probabilistic function $q_p$ in the regions of $D^n$ defined by the weight vectors of trained units in $V_l$. This can be achieved by creating a connection $\mathbf{z}_{ij}$ between shape unit $u_i$ and object unit $x_j$ and setting its strength equal to the probability that object $o_j$ is being viewed, given that unit $u_i$ has *fired*. These weights are trained by presenting a second, possibly different, set of example shapes to $V_l$. In order for the response of units in $V_l$ to be calibrated these shapes must now be labelled with the identity of the viewed object. This is achieved by the supervisory vector $\mathbf{t}$, where $\mathbf{t}_j = 1$ for object $o_j$ being viewed and $t_j = 0$ for all other objects. On each trial the output weights of the winning unit in $V_l$ are updated according to **Eq. 12**.

$$\mathbf{z}_{ij} = \mathbf{z}_{ij} + \mathbf{t}_j \tag{12}$$

Once training is complete the output vector of a shape unit $u_i$ can be thought of as a "freguency of wins" histogram for each object. From this the estimated probability $P(o_j|\mathbf{b})$ can be encoded in connection $\mathbf{z}_{ij}$ simply by computing the ratio of the number of times $u_i$ responded to a view of object $o_j$, to the total number of times that $u_i$ has won, **Eq. 13**.

$$\mathbf{z}_{ij} = \frac{\mathbf{z}_{ij}}{\sum_{j=1}^{m} \mathbf{z}_{ij}} \tag{13}$$

Although training is achieved by a winner-takes-all mechanism, during recognition the output of the network is an interpolated value obtained by taking the weighted average of the response of the top $k$ responding units. The combination of these layers therefore provides a network which is capable of learning the probabilistic function mapping projected shapes to objet identity.

## 6 DISCUSSION

This paper is concerned with the application of self-organizing neural networks to the problem of constructing adequate $2D$ models of $3D$ objects. In the previous section we proposed a network architecture for estimating the probability of recognition based on such models. The accuracy of this estimation will depend on the number of units used in $V_l$ and the degree to which the training set $I$ is representative of the true distribution of shapes on each hypersurface. These issues are now addressed.

### 6.1 Number of Stored Shapes

Presentation of an unknown shape $\mathbf{d}$ to the network causes a unit $u^*$ in $V_l$ to fire. This generates the set of probabilities $\{P'(o_1|\mathbf{d})...P'(o_m|\mathbf{d})\}$ at the output layer $O_l$. These values represent the networks estimate of the probabilities defined in **Eq. 6**. We would obviously like to know, for a given number of stored shapes, how accurate this estimate is. Unfortunately, although the true probability $P(o_i|\mathbf{d})$ is defined by the distribution of shapes in $\mathbf{H}$, we have no way of computing this value analytically. Consequently it is difficult to determine, for a given number of units, the accuracy to which the network is estimating true probabilities.

We can overcome this problem by thinking intuitively about the way in which the network estimates probabilities. The accuracy of the networks approximation to **Eq. 6** depends on the coarseness of the Voronoi *tiling* of $D^n$ produced by units in $V_l$. This is, as we have said, inversely proportional to the *number* of units in $V_l$. If insufficient units are used then the area occupied by each *tile* will be too large, increasing the liklihood of it including areas of more than one hypersurface. This causes recognition based on each tile to be falsely ambigous. If sufficient units have been used then each tile should be small enough to capture the *true* ambiguity in the probability distribution. Therefore, by examining the effect of the number of units on the amount of ambiguity in the network we can thereby obtain an idea of how many units must be used.

This will be made clearer by an example. A good measure of the ambiguity of a unit $u_i$ in $V_l$ is the value

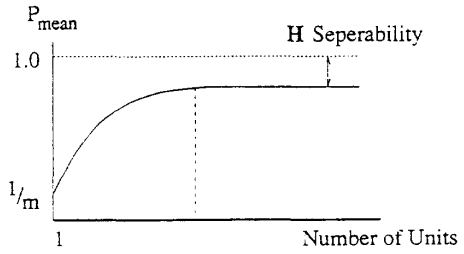


Figure 2: $P_{mean}$ Relationships

of $P_i^{max}$, defined as the most probable classification $P(o_j|\mathbf{d})$ encoded on the connection $z_{ij}$. As $n$ is increased the size of each tile should get smaller and so, on average, the value of $P_{max}$ for a given unit should increase. By taking the average value of $P_{max}$ in the network we get the value $P_{mean} = \sum_{i=1}^{c} P_i^{max}/c$, which provides a good measure of the degree of ambiguity in the network as a whole. We can now make some predictions about the relationship between $P_{mean}$ and $c$, the number of units in $V_l$.

In the extreme case, where $c = 1$, $P_{mean} = 1/m$ must be true, since the unit represents views of all objects. As $c$ is increased the value of $P_{mean}$ should increase until for some value of $c$ the tiles formed in $D^n$ are small enough to capture the *true* ambiguity of the probability distribution. At this point $P_{mean}$ should stop increasing as the fundamental limit of the distribution has been reached. Therefore the point at which $P_{mean}$ stops rising indicates the number of units that must be used to accurately estimate the true probability distribution of $\mathbf{H}$ in $D^n$. It should be noted that $P_{mean}$ may not have reached 1 at this point. In fact this must be the case if intersections exist between hypersurfaces in $\mathbf{H}$, as no matter how many units are used, shapes at the point of intersection are ambigous. The *height* of this limit therefore provides an indication as to the degree of separability of the hypersurfaces in $\mathbf{H}$.

### 6.2 Number of Training Shapes

We now address the question of how many training shapes have to be presented to the network in order for it to accurately estimate probabilities? Given that the network is to form the basis of a practical recognition system, the fewer images that have to be captured and processed the better. We consider two set of training data; the set of shapes $I$, used to train layer $V_l$ and the set $K$, used to calibrate the response of units in $V_l$. The number of examples in $I$ is not critical, for as long as there are sufficient inputs to accurately represent the distribution of shapes on the hypersurfaces in $\mathbf{H}$ then the network will distribute units adequately. Of greater importance to the accuracy of approximation is the number of shapes in $K$.

The method of approximation in the network relies on taking ratios as estimates of true probabilities. The accuracy of this estimate depends upon the number of examples involved in the ratio, ie the size of $K$. However, since we do not have the true probability distribution it is not easy to relate number of examples to accuracy of approximation. We can again overcome this problem by considering the relationship between $k$, the number of shapes in $K$, and $P_{mean}$. If $k = 1$ then $P_{mean} = 1$ must be true, since the network has only been told about a single object. As $k$ is increased the artificially high certainty of the network should fall, as the full variation in the identity of shapes falling within a tile is more accurately sampled. The value of $P_{mean}$ should therefore stop falling at the point at which the variation has been sufficiently sampled.

Obviously, the number of units used, $c$ and the size of the training set, $k$ both play a part in determining the accuracy of the network approximation to $q_p$. By analysing the relationship between the value of $P_{mean}$ and those of $j$ and $k$ we should able to determine the point at which the network is accurately approximating $q_p$.

## 7 SUMMARY

This paper has detailed an approach to 3D object recognition based on using a self-organizing neural network to learn an approximation to the function mapping 2D projected shape to object identity. The domain of this function has been characterised as a set of hypersurfaces in the space of possible shape representations. As an extention to previous systems in this field a representational scheme capable of handling real image data has been proposed together with a network architecture capable of providing the probabilistic recognition necessary in real imaging situations. Finally a method for determining the required number of stored shapes and training examples has been proposed.

### References

[1] Ullman, S. & Basri, R. 1991. Mach. Intell. 13/10 992-1005

[2] Poggio, T. & Edelman, S. 1990. Nature, 343, 263-266.

[3] Forsyth, DA., Mundy, JL., Zisserman, AP. 1990. Proc. ECCV1 427-436.

[4] Koenderink JJ., & van Doorn AJ. Biol. Cybern. 32 211-217.

[5] Evans, AC, Thacker, NA., Mayhew, JEW. 1992. Proc. 11th Int. Conf. Patt. Rec 133-137.

[6] Thacker, NA. & Mayhew, JEW. 1989. Neural Networks, 3/3, 291-300.