Pulse Coupled Neural Networks: An Exploration of Parameterisation Methods

Robert Stewart

MSc by Research in Pattern Analysis and Neural Networks

The University of Aston in Birmingham

September 2000

The University of Aston in Birmingham

Pulse Coupled Neural Networks: An Exploration of Parameterisation Methods

Robert Stewart

MSc by Research in Pattern Analysis and Neural Networks

Thesis Summary

Pulse coupled neural networks (PCNNs) comprise a family of biologically motivated models originally developed to replicate the phase-synchronised pulsing behaviour observed amongst collections of neurons in the mammalian visual cortex. They have been applied to a number of applications within the image processing field: most notably image smoothing and segmentation. PCNNs are complex dynamical models with a number of adjustable parameters of reciprocal influence. As a result their behaviour is difficult to accurately predict, control or analyse. This paper follows the development and analysis of a number of parameterisation methods for the PCNN aimed at making it a more powerful and reliable image segmentation model. Experimental results are used to examine the strengths of each of these methods relative to one another in both qualitative and quantitative terms. An energy function formalism for a sub-class of the PCNN family is then proposed and analysed and a Bayesian interpretation is offered.

Keywords: PCNN, image segmentation, energy function

Acknowledgements

The author would like to thank British Aerospace for making the use of the Sowerby Image Database available for this research project.

Contents

List of Figures

1 Introduction

1.1 The Original Model

1.1.1 Development

Pulse Coupled Neural Networks (PCNNs) comprise a family of neural computing models used principally in image processing applications. Their development was originally motivated by neuroscientific exploration of the workings of mammalian primary visual cortex. In particular, studies of the cat visual cortex [1], [2] found that neurons separated by surprisingly large distances within the region would sometimes exhibit phase-synchronized pulsing behaviour. These results were subsequently replicated in studies using awake monkeys [3]. The suggestion was made [1], [4] that this behaviour was part of the mammalian brain's solution to the problem of feature binding or feature linking wherein disparate pieces of sensory information regarding a given object must be seamlessly linked together into a unified percept for that object. When we look at a cat we see not just the edges, textures and movement of the animal but also the animal as an indivisible whole.

In 1990 Eckhorn introduced the linking field model as a phenomenological model designed to replicate the observed cortical behaviour [5]. It was soon realised that this model exhibited a number of interesting properties in line with the feature-linking hypothesis. Subsequent work through the 1990s, led by Johnson [6], [7], Kuntimad and Ranganath [7], [8], Kinser [9], [10] and Lindblad [10], led to a number of modifications and simplifications being made to the linking field model. These modifications allowed what was a piece of computational neuroscience to evolve into an effective image processing tool and produced the family of algorithms now commonly referred to as PCNNs.

1.1.2 Features

1.1.2.1 Neuron Features

The model neurons of the original model are generally taken to be identical within a given network and feature two functionally different types of synapse. The *feeding synapses* receive input from the main stimulus-driven pathway that directly carries sensory information. The

linking synapses receive input from neighbouring neurons and act upon the feeding inputs in a multiplicative, modulatory manner. Both types of synapse are dynamic: being modelled as leaky capacitors whose outputs are steeply charged by a large incoming signal before decaying exponentially over time.

The linking and feeding synapses, generally denoted L and F respectively, together form the neuron's internal activation U, which corresponds to the membrane voltage of a biological cell. The internal activation passes into a pulse generator called a neuromime that operates via the interaction between the excitatory input U, an inhibitory threshold T and an excitatory step function Y. Like the linking and feeding synapses, the threshold is realised as a leaky capacitor that is charged by the output pulses Y and decays exponentially over time. Figure 1.1 schematically displays the operation of one of these model neurons.

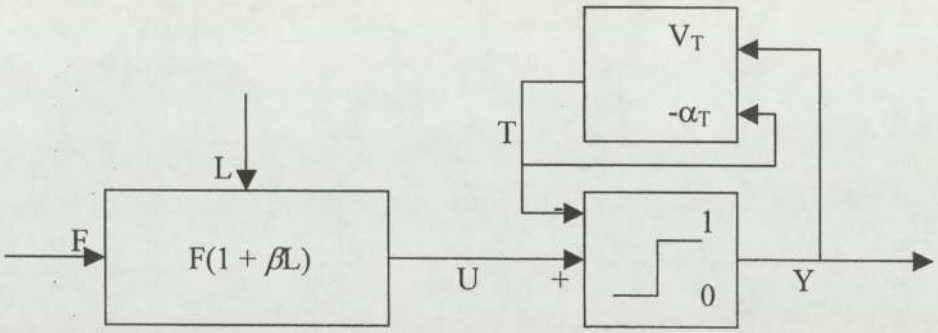


Figure 1.1. Schematic representation of one of Eckhorn's model neurons

1.1.2.2 Network Features

Early work with the linking field model focussed on one-dimensional layers of neurons but subsequent work has been largely devoted to networks featuring two-dimensional planar arrangements, usually configured as a regular, rectangular lattice. In image processing applications, a one-to-one pixel to neuron correspondence is generally employed and mathematically the operation of this arrangement is expressed via the following set of equations.

REPEAT

$$F_{ij}[n] = e^{\alpha_F} F_{ij}[n-1] + V_F \sum_{kl} M_{ijkl} Y_{kl}[n-1] + S_{ij} \tag{1a}$$

$$L_{ij}[n] = e^{\alpha_L} L_{ij}[n-1] + V_L \sum_{kl} W_{ijkl} Y_{kl}[n-1] \tag{1b}$$

$$U_{ij}[n] = F_{ij}[n]\{1 + \beta L_{ij}[n]\} \tag{1c}$$

$$Y_{ij}[n] = \begin{cases} 1 & \text{if } U_{ij}[n] > T_{ij}[n-1] \\ 0 & \text{otherwise} \end{cases} \tag{1d}$$

$$T_{ij}[n] = e^{\alpha_T} T_{ij}[n-1] + V_T Y_{ij}[n] \tag{1e}$$

UNTIL some stopping condition

The value S_{ij} appearing in (1a) is the stimulus input to the neuron N_{ij}. In image processing applications, this is generally taken to be the intensity of the corresponding pixel. The feeding and linking kernels M and W represent the connectivity pattern within the network and are traditionally taken to be local and Gaussian, although this is not a strict requirement. In the interests of computational efficiency, M and W are often set equal to one another so that the expensive spatial convolution operation is only performed once at each time step. The linking coefficient β was not present in the original linking field model but was soon developed as a control over the strength of the linking modulation [7] and has been universally employed ever since. It takes the form of a positive constant generally applied globally to control the linking strength over the entire network.

1.1.3 Properties

The linking field model is able to robustly replicate the synchronous pulsing activity observed in recordings from the visual cortex. The mechanism underpinning this is the spread of activation through the network in the form of feeding and linking autowaves. An autowave is defined as a normal propagating wave that does not reflect or refract and hence are annihilated by contact with another wave. When a neuron pulses, an excitatory autowave spreads from it in all directions. Synchronisation then occurs via *pulse capture*. This is the term used to describe the process whereby an autowave increases the internal activation of a receiving neuron sufficiently for the neuron to pulse at an earlier time step than it would have in isolation.

The pulse capture process can eventually cause large regions of the network to pulse in synchrony but what exactly this indicates about the corresponding regions of the distal stimulus depends on the specific receptive field properties of the model neurons employed and the nature of the local stimulus input to each of the neurons involved. In general we can say that neurons with similar receptive fields and similar stimulus inputs will tend to exhibit synchronous pulsing. In the image processing case, with local, Gaussian kernels and S_{ij} set to pixel intensities, this means that pulsing synchronisation depends on two main properties of the corresponding image pixels:

1. Intensity similarity

2. Spatial proximity

We can therefore view synchronisation within the network as a way of grouping the pixels themselves according to these properties and synchronous pulsing behaviour is found to spread across an image region despite small spatial and temporal discontinuities and intensity variations. This enables the PCNN to perform a number of useful image processing tasks such as smoothing [7], edge detection, and particularly segmentation [8]. In this current work we will focus on the smoothing and segmentation properties of the PCNN.

1.2 Modifications and Simplifications

The linking field model is a complex dynamical model with a large number of adjustable parameters. It is difficult to effectively control or analyse the behaviour of the model in this form and consequently a number of modifications and simplifications have been introduced. What follows is a partial list of these alterations in what is hoped is a logical rather than chronological order.

1.2.1 The Single Pass Modification

Johnson and Padgett [10] point out that when applying the PCNN "the segmentation, smoothing and grouping action occurs in the first pass through the image." The single pass modification takes advantage of this by introducing a stopping condition such that the

algorithm terminates when every neuron in the network has pulsed. Generally, the parameters of the model are set in such a way as to ensure that each neuron pulses once and only once.

1.2.2 The Direct Feeding Modification

Kuntimad and Ranganath [11] suggest that in the case of a static image, the inclusion of a leaky capacitor in the feeding pathway is of no benefit. The direct feeding modification accordingly removes the leaky capacitor. Information from other neurons is also removed from the feeding pathway since this can be communicated via the linking pathway. The feeding input to a neuron therefore simplifies to a constant value equal to the intensity of the corresponding image pixel.

$$F_{ij}[n] = S_{ij} \tag{2}$$

1.2.3 The Fast Linking Modification

Allowing linking waves to propagate more rapidly across the network has been noted to improve denoising [9] and result in a smoother regional segmentation [10] of the input image. The fast linking modification therefore allows linking waves to propagate across the entire network in a single time step by iterating (1b), (1c), (1d) within an internal loop until the output matrix $\mathbf{Y}$ stabilises to a fixed state.

1.2.4 The Stepped Linking Modification

The stepped linking modification comprises two changes to the linking field usually applied together. First the leaky capacitor is removed from the pathway so that the linking input becomes an instantaneous weighted sum of local pulsing activity. With single pass operation on static images, the capacitor confers no benefit and adds unnecessary complexity to the model. The note is then made [10] that more uniform linking values produce more uniform segmentations, which leads to the second change where the linking value is passed through a squashing or, more usually, a step function (3). Lindblad and Kinser [9] note that stepped linking tends to preserve the edges of the linking autowaves better than the original scheme, which may account for the improved performance observed.

$$L_{ij}[n] = \begin{cases} 1 & \text{if } \sum_{kl} W_{ijkl} Y_{kl} > \theta_L \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

One common practice is to use a threshold of zero on this stepping function and this is actually a rather special case. Since the kernel W does not normally contain negative values and Y_{kl} takes on values of zero and one, the linking calculation corresponds to a logical OR over the linking field.

1.2.5 Threshold Modifications

There are two main modifications to the threshold mechanism. The first of these is to replace the leaky capacitor with a linear decay system. This has been found to produce results of equal quality to the original scheme and is preferred due to simplicity. The next modification is to replace the multiplicative recharge of the threshold with a mechanism that resets thresholds in the event of pulsing, to a constant value the same for every neuron. This is computationally simpler and equally effective. With both these modifications in place, the threshold equation becomes:

$$T_{ij}[n] = \begin{cases} \Omega & \text{if } Y_{ij}[n] = 1 \\ T_{ij}[n-1] - dT & \text{otherwise} \end{cases} \quad (4)$$

1.3 Taking Stock

Looking closely at the stepped linking modification, we note that the time index of Y_{kl} in (3) has become ambiguous. We can take its value to be $[n\text{-}1]$ as in the original, but when we combine stepped linking with fast linking it becomes more natural to use a time index of $[n]$. At each time step, the linking input to every neuron now has a value of zero prior to a pulsing event. With direct feeding, we find using (1c) that the internal activation now takes on the value of the pixel intensity prior to pulsing so that (1a) to (1d) simplifies to

$$Y_{ij}[n] = \begin{cases} 1 & \text{if } S_{ij} > T_{ij}[n-1] \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

Simultaneously applying all the modifications, we then arrive at the following set of equations

REPEAT

$$Y_{ij}[n] = \begin{cases} 1 & if\ S_{ij} > T_{ij}[n-1] \\ 0 & otherwise \end{cases} \tag{6a}$$

WHILE there is any change in pulsing activity

$$L_{ij}[n] = \begin{cases} 1 & if\ \sum_{kl} W_{ijkl} Y_{kl}[n] > \theta_L \\ 0 & otherwise \end{cases} \tag{6b}$$

$$U_{ij}[n] = S_{ij}\{1 + \beta L_{ij}[n]\} \tag{6c}$$

$$Y_{ij}[n] = \begin{cases} 1 & if\ U_{ij}[n] > T_{ij}[n-1] \\ 0 & otherwise \end{cases} \tag{6d}$$

END

$$T_{ij}[n] = \begin{cases} \Omega & if\ Y_{ij}[n] = 1 \\ T_{ij}[n-1] - dT & otherwise \end{cases} \tag{6e}$$

UNTIL all neurons have pulsed

The central WHILE loop, iterating (6b) to (6d) is referred to as the fast linking loop and has become the main component of the model in terms of both computational expense and the determination of network behaviour.

We note that the eventual PCNN form displayed above is very different from the original linking field model. Recalling that even the linking coefficient β was not present in the original model we can see that every equation of the model has been considerably modified. This form of PCNN still performs the same function as its predecessor: grouping pixels according to their intensity similarity and spatial proximity, but is designed to do so in a single pass through the model and at much more limited computational expense.

Despite these simplifications however, the control and analysis of PCNN behaviour is still far from trivial. Crucial determinants of the performance of this model include the initial threshold value, the threshold decay term dT, the linking threshold θ_L, the nature of the linking kernel **W** and the value of the linking coefficient β. If this collection of values is fixed and the threshold reset term is appropriately large, then behaviour of the model is also fixed but this still leaves us with a considerable number of degrees of freedom in the operation of the network.

Previous papers have frequently suggested appropriate ranges for some of the parameters of the PCNN acquired through operating experience. While these suggestions are valuable to a point, each application of the PCNN will typically require user-tweaking of parameters and even model form in the search for better performance. This type of unprincipled tweaking was prevalent in multi-layer perceptron research prior to their statistical treatment and the hope is that the situation can also be remedied for the PCNN.

1.4 Aims

The aim of this research project was to develop effective parameterisation methods for the PCNN family. Qualifying the term effective, the following list of desirable characteristics was drawn up. Effective methods should feature most or all of these characteristics.

1. They should be fully automated.

2. They should be flexible and yet powerful.

3. They ought to produce parameter sets that are in some sense optimal for the application under consideration.

4. They should improve the performance and reliability of the PCNN as an image processing tool.

Ideally, we wanted to produce methods founded on probability theory so that they would be open to analysis and would hopefully give us a good trade-off between flexibility and power. A probability-based approach would also give us a more precise way of defining *optimal* than would a purely empirical approach.

We decided to restrict our current work to the parameterisation of PCNNs designed to perform image segmentation since this is the main area to which these models have so far been applied. Initially it was felt crucial to gain a more thorough understanding of the properties and behaviour of the PCNN family in its image segmentation role and to achieve this we aimed to seek out and explore previous work in this area with particular attention being given to any existing parameterisation methods that could be found.

2 Methods And Results

While exploring the literature for existing parameterisation methods for the PCNN, we came across an interesting paper by Kuntimad and Ranganath [12]. Here they presented a method for setting certain PCNN model parameters according to prior knowledge of the pixel intensity ranges present within different image regions. The claim is made that under certain conditions this method can be guaranteed to achieve perfect segmentation of a simple figure/ground image even when the intensity ranges of the regions in question overlap to some extent. We decided to investigate further.

2.1 Perfect Segmentation

At first glance, the perfect segmentation guarantee appears to be an impressive and significant result: the authors making the claim that such conditions have yet to be established for any other image segmentation method. However, the method also suffers from severe limitations relating to the degree of intensity range overlap and the boundary geometry between the image regions to be segmented. In the following sections, a summary of the method proposed by Kuntimad and Ranganath (K&R) is presented and the performance of the method is demonstrated as the intensity range overlap of the image regions is varied in a controlled manner. Alterations to the method; made in the hope of countering these limitations, are introduced and briefly analysed.

An alternative method for setting the model parameters is then presented that further highlights the weaknesses of the K&R method's reliance on boundary geometry conditions and serves to motivate subsequently developed parameterisation methods of more general utility.

2.1.1 Basic Model Features

The model employed by K&R differs considerably from Eckhorn's original linking field model. The modifications involved: all variants of those described in the introduction, are briefly described below:

1. *Single Pass Operation:* The K&R model is reset to its initial conditions after each pass and according to the deterministic nature of the PCNN would therefore produce

identical results with every pass. It is therefore justified to consider the first pass alone.

2. *Direct Feeding:* Only static images are considered in the K&R paper and as we noted in the introduction, the use of a leaky integrator in the feeding path confers no benefit and incurs unnecessary additional computational expense. The direct feeding modification is therefore applied.

3. *Threshold Reset:* Exponential decay of threshold values is retained but the simpler reset scheme is used in preference to multiplicative recharge and the decay is calculated relative to the reset value omega rather than the previous threshold value. The threshold mechanism is referred to as a threshold signal generator (TSG) and is applied identically to all neurons.

The size of the kernel **W**, employed in the linking field, is determined by a radius parameter r that defines the maximum linking distance: distance between neurons being defined as the Euclidean distance between the corresponding pixels in the image. The weighting values within this radius are given as $1/d^2$ where d is the distance from the centre of the kernel. For the special case of the centre of the kernel, a value of zero may be employed since self-linking has no effect on the performance of the model and requires additional computation in software implementations. K&R employed only small linking radii of $r = 1.0$ and $r = 1.5$. The corresponding kernel values are displayed below:

0	1	0
1	0	1
0	1	0

(a)

0.5	1	0.5
1	0	1
0.5	1	0.5

(b)

Figure 2.1. Linking kernels employed in the K&R model. (a) Linking radius $r = 1.0$. (b) Linking radius $r = 1.5$.

2.1.2 A Mathematical Summary

REPEAT

$$Y_{ij}[n] = \begin{cases} 1 & if\ S_{ij} > T_{ij}[n-1] \\ 0 & otherwise \end{cases} \tag{7a}$$

WHILE there is any change in pulsing activity

$$L_{ij}[n] = e^{\alpha_L} L_{ij}[n-1] + V_L \sum_{kl} W_{ijkl} Y_{kl}[n-1] \tag{7b}$$

$$U_{ij}[n] = S_{ij}\{1 + \beta L_{ij}[n]\} \tag{7c}$$

$$Y_{ij}[n] = \begin{cases} 1 & if\ U_{ij}[n] > T_{ij}[n-1] \\ 0 & otherwise \end{cases} \tag{7d}$$

END

$$T_{ij}[n] = \begin{cases} \Omega & if\ Y_{ij}[n] = 1 \\ \Omega \exp(-(n-n_1)/\alpha_T) & otherwise \end{cases} \tag{7e}$$

UNTIL all neurons have pulsed

The value n_1 used in the threshold signal generator is defined as the last time of pulsing of the neuron N_{ij}.

2.1.3 The K&R Method

Following K&R, we use the example of a figure/ground image having spatially connected object pixels forming R and background pixels forming B. We Let ($S_{R\,min}$, $S_{R\,max}$) and ($S_{B\,min}$, $S_{B\,max}$) be the intensity ranges of the object and background pixels respectively and apply the following assumptions:

$$S_{R_{\max}} > S_{B_{\max}} \tag{8}$$

$$S_{B_{\max}} > S_{R_{\min}} \tag{9}$$

Stating this slightly loosely, the object region R is brighter than the background but there is some overlap between the intensity ranges so that simple thresholding is incapable of producing perfect segmentation.

The K&R method begins by setting the threshold values of all neurons to the reset value Ω and allowing it to decay until the first neurons pulse. According to (8), the first pulsing neurons will be the brightest object neurons. We simplify the method here by setting the initial threshold value just below the maximum pixel value of the object region, $S_{R\,max}$ and this allows us to employ an arbitrary, very large threshold reset value rather than needing to calculate one just large enough to ensure that each neuron only pulses once. In Matlab implementations the system dependent constant REALMAX may be employed for Ω.

Once the first pulsing event has occurred, the algorithm passes into a fast linking loop, wherein for perfect segmentation we must ensure that the following conditions are satisfied for all object neurons N_{ij} and all background neurons N_{pq} respectively.

$$U_{ij} > T_{ij} \tag{10}$$

$$U_{pq} \leq T_{pq} \tag{11}$$

Using the simplified threshold initialisation, we know that $T_{ij} = T_{pq} = T_{init}$ and that T_{init} is set just below $S_{R\,max}$. So equivalently we can put:

$$U_{ij} \geq S_{R_{\max}} \tag{12}$$

$$U_{pq} < S_{R_{\max}} \tag{13}$$

Using (7c) we can derive the minimum value of beta required to satisfy (12) by setting S_{ij} equal to the minimum object pixel value $S_{R\,min}$ and $L_{ij}[n]$ equal to the minimum linking input from other object neurons which we will call $L_{R\,min}$. In similar fashion, we can derive the maximum value of beta able to satisfy (13) by setting S_{ij} equal to the maximum background pixel value $S_{B\,max}$ and L_{ij} equal to the maximum linking input from pulsing object neurons which we will call $L_{B\,max}$. By rearranging (7c) and using (12) and (13), the following limiting values for beta are obtained:

$$\beta_{\min} = \left((S_{R_{\max}} / S_{R_{\min}}) - 1\right) / L_{R_{\min}} \tag{14}$$

$$\beta_{\max} = \left((S_{R_{\max}} / S_{B_{\max}}) - 1\right) / L_{B_{\max}} \tag{15}$$

K&R apply a further condition on the minimum value of beta according to the requirement for all the background neurons to pulse together at the second time step. Without altering the segmenting performance of the model however, we can further simplify the threshold scheme by introducing linear rather than exponential decay. In the figure/ground case, this can be reduced to decay to zero which forces any neurons not pulsing in the first time step to pulse in the second, thus producing the two part segmentation required. Equivalently, threshold decay can be abandoned altogether by simply classing pixels whose neurons do not pulse in the first time step as background pixels. In a case with more than two regions to segment, this simple acceptance condition would be applied to the last region with no detriment to segmentation quality.

2.1.4 Boundary Geometry Conditions

While the values of $S_{R\,max}$, $S_{R\,min}$ and $S_{B\,max}$ are assumed to be known quantities, the values of $L_{R\,min}$ and $L_{B\,max}$ depend on the linking radius r and, according to K&R, the object-background boundary geometry. As will be shown below, this is not entirely correct but initially we will begin by presenting their simple example.

R is defined as a rectangular region as displayed in Figure 2.2, and the pixel/neuron positions suggested to be receiving $L_{R\,min}$ and $L_{B\,max}$ for $r = 1.0$ and $r = 1.5$ are indicated in Figure 2.2 (a) and (b) respectively. Given these positions, it can be shown that (a) for $r = 1$, $L_{R\,min} = 2.0V_L$ and $L_{B\,max} = 1.0V_L$ and (b) for $r = 1.5$, $L_{R\,min} = 2.5V_L$, $L_{B\,max} = 2.0V_L$.

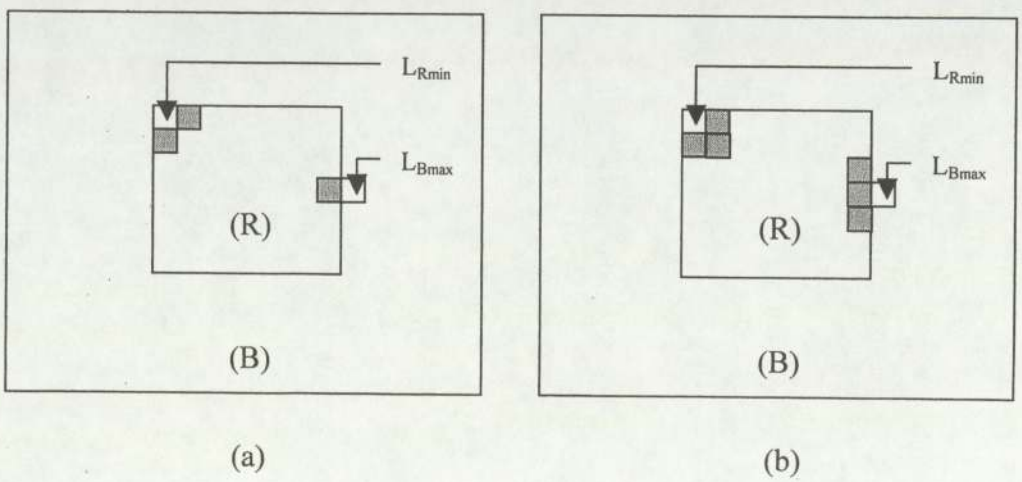


Figure 2.2. Crucial boundary positions for a rectangular region. (a) $r = 1.0$. (b) $r = 1.5$.

2.1.5 Solutions with Uniform Intensity Distributions

In order to study the performance of Kuntimad and Ranganath's perfect segmentation method in as controlled a way as possible, we examine the case of an image normalised in the range (0,1) with uniform intensity distributions symmetrically overlapping about a value of 0.5. Thus the intensity range of the background region is (0, $S_{B\,max}$) and the intensity range of the object region is (1-$S_{B\,max}$, 1). The extent of overlap is then controlled by the value of $S_{B\,max}$ alone allowing for simple analysis. Equations (14) and (15) can here be rewritten in the following form:

$$\beta_{\min} = \left((1/(1-S_{B_{\max}})) - 1\right) / L_{R_{\min}} \tag{16}$$

$$\beta_{\max} = \left((1/S_{B_{\max}}) - 1\right) / L_{B_{\max}} \tag{17}$$

By setting $\beta_{\min}$ equal to $\beta_{\max}$ and using values for $L_{R\,min}$ and $L_{B\,max}$ appropriate for the border geometry and linking radius, we can calculate the maximum value of $S_{B\,max}$ at which perfect segmentation can be guaranteed. Remaining with a rectangular object region R, we set the linking amplification factor V_L to one and consider the two cases of r = 1.0 and r = 1.5

Case1: r = 1.0, $L_{R\,min}$ = 2.0, $L_{B\,max}$ = 1.0. Setting $\beta_{\min}$ equal to $\beta_{\max}$ and rearranging, we arrive at a quadratic in $S_{B\,max}$:

$$S_{B_{\max}}^2 - 4S_{B_{\max}} + 2 = 0 \tag{18}$$

Solving this gives us a maximum value for $S_{B\,max}$ of 0.586 to 3s.f.

Case 2: r = 1.5, $L_{R\,min}$ = 2.5, $L_{B\,max}$ = 2.0. Following the same procedure we arrive at the following quadratic:

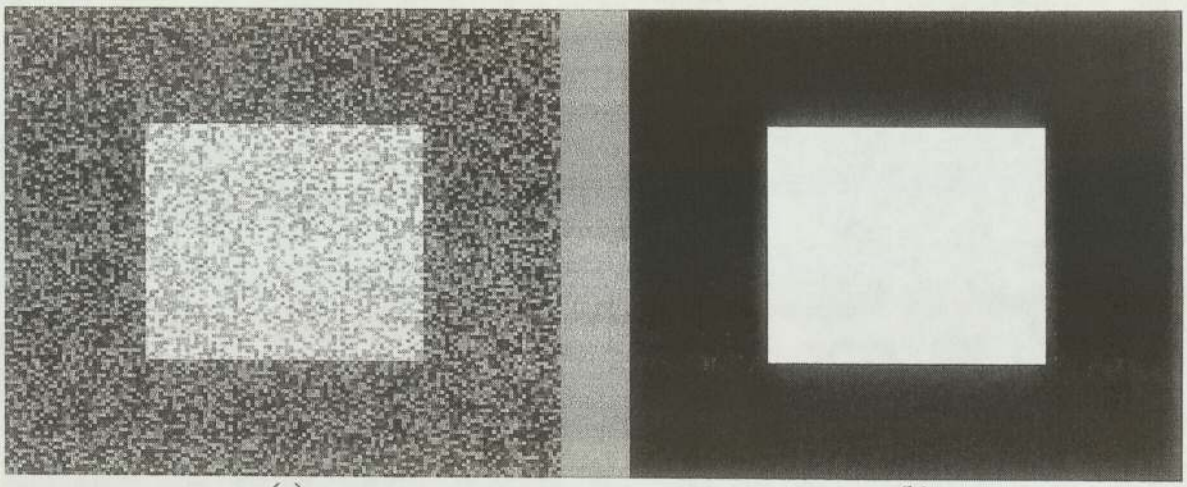
$$S_{B_{\max}}^2 - 10S_{B_{\max}} + 5 = 0 \tag{19}$$

Solving this gives us a maximum value for $S_{B\,max}$ of 0.528 to 3s.f.

Consequently, we ought to prefer the smaller linking radius r = 1.0, since this can succeed in producing perfect segmentation in a situation of greater intensity overlap between adjacent regions than the larger linking radius.

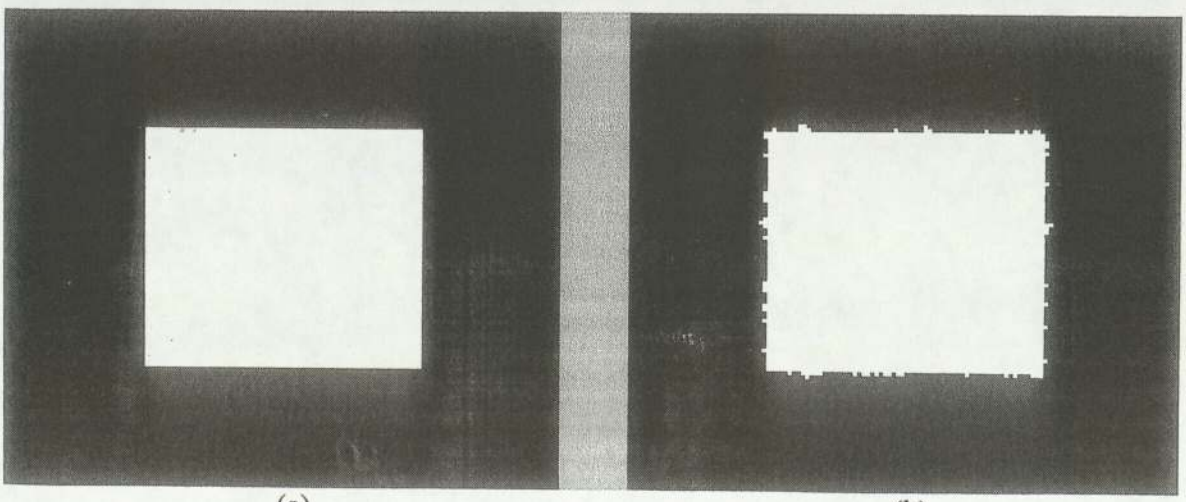
2.1.6 Results for Uniform Distributions

We test the solutions obtained above by running the algorithm using both kernel sizes and with $S_{B\,max}$ set to 0.55. This value lies between the maximum values calculated for the two kernels and ought thus to be perfectly segmented by the smaller kernel, $r = 1.0$, but not necessarily by the larger one, $r = 1.5$. Figure 2.3 displays, (a) the input image and (b) a corresponding template demonstrating perfect segmentation for this image. Figure 2.4 shows the segmentation results obtained where we set beta equal to $\beta_{\min}$ for (a) $r = 1.0$ and (b) $r = 1.5$. This arrangement should ensure complete capture of the object region but at values of $S_{B\,max}$ greater than the maximum calculated above it will typically also capture some background pixels. As expected according to the solution above, perfect segmentation is achieved in the former but not the latter case.



(a) (b)

Figure 2.3. Test image for the K&R method. (a) Input image with $S_{B\,\max} = 0.55$. (b) Template demonstrating perfect segmentation



(a) (b)

Figure 2.4. Segmentation results with the K&R method. Results obtained using $\beta = \beta_{\min}$ with (a) $r = 1.0$ and (b) $r = 1.5$.

2.1.7 Problems with the method

The main problems with the perfect segmentation method proposed by K&R are the limited intensity range overlap at which it remains effective and its reliance on exceptional border geometry conditions. The former problem has been dealt with above when solving for $S_{B\,max}$ showing that, particularly with larger linking radii, the method is restricted to very small intensity range overlap values. In reference to the latter problem, K&R admit that "boundary geometry determines the performance of the PCNN" but they fail to define the geometry required for the success of the method. Here we make the requirement explicit:

For guaranteed success of the method, $\beta_{\min}$ must be less than $\beta_{\max}$. In order for this to occur, $L_{B\,max}$ must be less than $L_{R\,min}$ and this condition only holds when R is rectangular. As soon as there is a concavity of any size in the border of R, the method fails. While it may still achieve perfect segmentation, this achievement cannot be guaranteed.

2.1.7.1 Modifying the Method

Kuntimad and Ranganath acknowledge some of the weaknesses of their parameterisation method and propose two techniques for its improvement. The first of these is external image smoothing but while this is effective to some extent, it is not favoured here because of it's external nature. It is felt that a single layer PCNN ought to be able to achieve satisfactory smoothing internally by virtue of the linking mechanism and that this method should be favoured for its elegance and faster implementation. This approach will be developed in subsequent sections.

The second technique is to modify the neurons of the network to incorporate inhibitory as well as excitatory linking fields. The inhibitory receptive fields serve to delay the pulsing of neurons that initiate a fast linking loop. This results in an effective compression of the intensity range within a region and hence also an effective reduction in the overlap between adjacent regions. Improved segmentation results may consequently be achieved. However, the addition of the inhibitory field complicates the analysis of the behaviour of the PCNN. It will therefore not be included in this work although it may be investigated in subsequent research.

2.1.7.2 Falsifying the Proof

As indicated in 2.1.4 above, the K&R method relies on boundary conditions to fix the values of $L_{R\,min}$ and $L_{B\,max}$ but while the method for setting $L_{B\,max}$ is unassailable, fault can readily be found with the asserted method for setting $L_{R\,min}$. In order for $L_{R\,min}$ to occur at the location indicated, pulsing activation must first have spread from its initial positions to the corners of the region R but this spread of activation is assumed implicitly and no guarantee for its occurrence is made. We might therefore reasonably ask under what conditions the required spread of activation will occur.

Unfortunately the spread of pulsing activity in a PCNN is governed by local patterns of pixel intensity and in this case the required conditions would rely upon local intensity patterns across the entire region R. As R becomes larger, we are faced with very high dimensional pattern spaces and an analytical solution for the conditions becomes unfeasible. We can still apply the heuristic conditions below but must bear in mind that these are necessarily imprecise.

- The intensity distribution within the region R should be fairly smooth

- The spatial distribution of intensity values within R should be fairly even

Very rough intensity distributions or uneven spatial distributions will cause activation to fail to spread to the corners of R. Consequently the K&R method will fail to achieve perfect segmentation despite the existence of an apparently tractable intensity range overlap. The following example illustrates a situation where these heuristic conditions are violated and the K&R method fails.

Example: Let R be a rectangular region with a uniform intensity of 0.45. Now let some random process set pixels within the region to a value of 1.0 with a probability p(change). Provided $0 < p(\text{change}) < 1$, the values of $S_{R\,min}$ and $S_{R\,max}$ are 0.45 and 1.0 respectively. Background intensities are once again drawn from a uniform distribution in the range (0, 0.55). We note that this size of overlap is within the range that ought to allow perfect segmentation when employing the smallest linking radius $r = 1.0$. Figure 2.5(a) shows an image of this type where $p(\text{change}) = 0.05$. Note that the high value pixels in R are sparsely distributed and that intensity distribution is highly non-smooth. Figure 2.5(b) shows the segmentation achieved by the K&R method on this image.

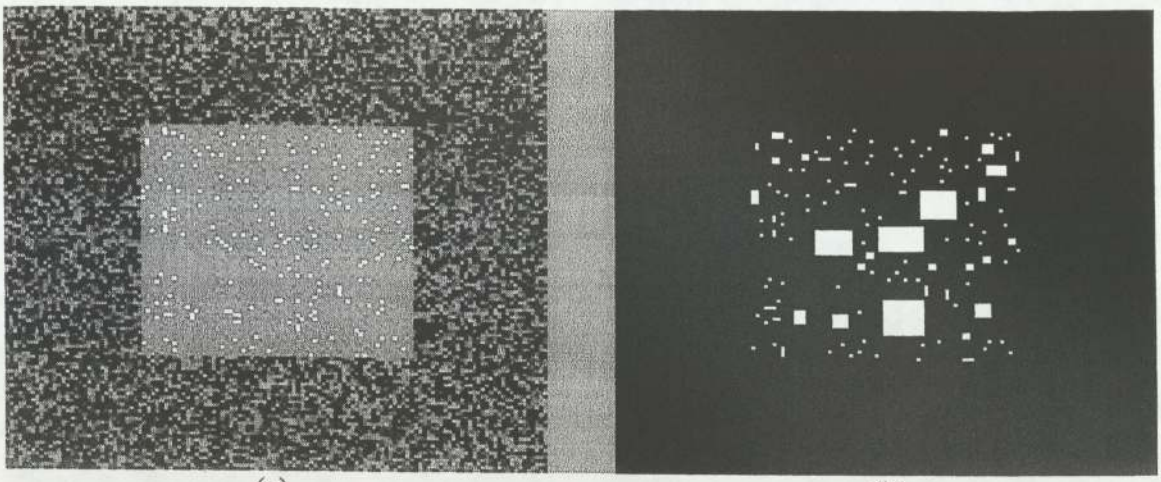


Figure 2.5.Failure of the K&R method with sparse distribution. (a) Input image with sparsely distributed bright pixels. (b) Segmentation achieved by the K&R method with $\beta = \beta_{min}$ and $r = 1.0$.

Figure 2.6 shows the performance of the method as a percentage of correctly segmented pixels as we vary the value of p(change). To display the results in order of increasing bright pixel sparseness, we use a value of 1 - p(change) for the x-axis here and we see that for most values of p(change), perfect segmentation is achieved but as this value falls below (0.05), the algorithm fails suddenly. There is no hard threshold here however. Using a different random seed may result in a slightly different value of p(change) causing failure. The method simply becomes brittle in situations of sparse or uneven bright pixel distributions.

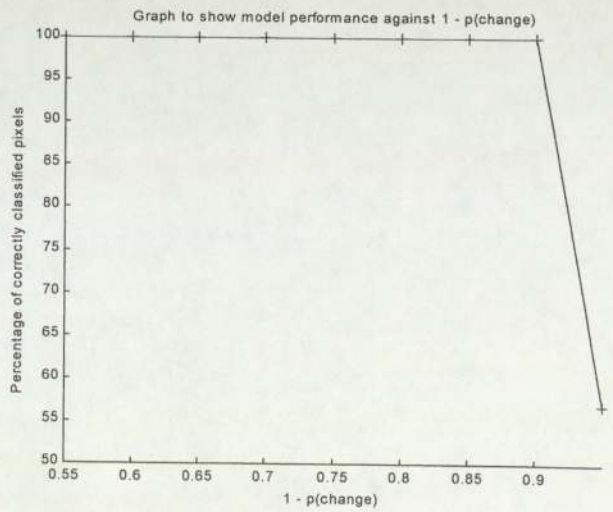


Figure 2.6. Segmentation performance of K&R method as a percentage of correctly classified pixels over 1 – p(change) with r = 1.0 and $\beta = \beta_{min}$.

2.1.8 An Alternative Method

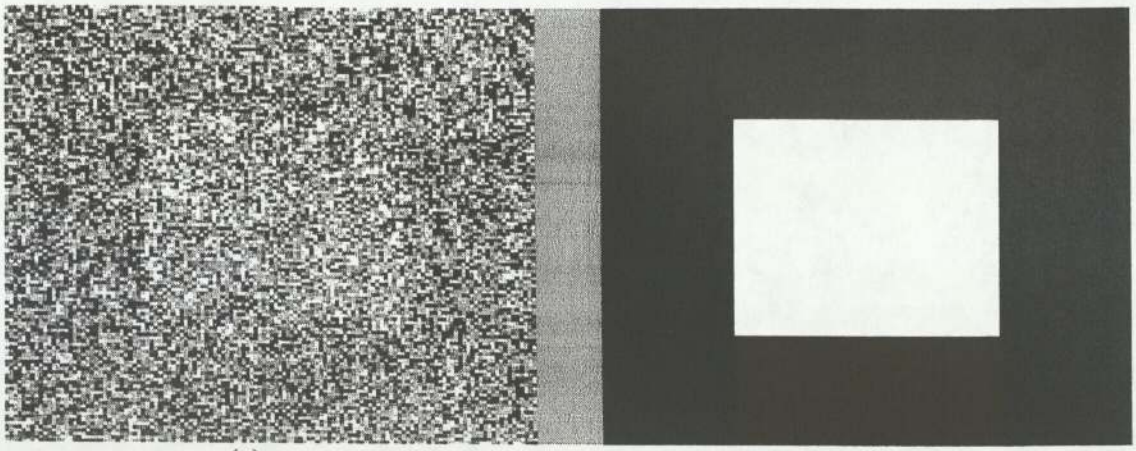
The main purposes of presenting an alternative perfect segmentation method are to further highlight the weaknesses of the K&R method, and to motivate more general and useful parameterisation methods. The method selected relies on the same border geometry conditions as the K&R method but takes the reliance to the extreme so that if R is rectangular, very large intensity range overlaps may be overcome but if R is not rectangular the algorithm will fail utterly. The same sort of heuristic conditions for success as presented in 2.1.7 will also apply here.

The model employed here will be slightly different from that employed by K&R. It features stepped linking along with the same simplified thresholding scheme introduced above. Whereas perfect segmentation was previously achieved via manipulation of the linking coefficient β, here we rely on manipulation of the linking threshold θ_L.

As demonstrated in 2.1.4, given a rectangular object region R and suitably spreading activation, we arrive at the following border linking values: (a) for $r = 1$, $L_{R\,min} = 2.0V_L$ and $L_{B\,max} = 1.0V_L$ and (b) for $r = 1.5$, $L_{R\,min} = 2.5V_L$, $L_{B\,max} = 2.0V_L$. By thresholding $\mathbf{L}$ at $L_{B\,max}$ we ensure that background pixels will always receive zero linking input from object neurons. To encourage pulsing activity to spread to the borders of R, we set β to be large enough to cause the object neuron having minimal feeding input $S_{R\,min}$ to pulse provided a linking input is present.

Unlike the (slightly modified) K&R method where the initial threshold value T_{init} was set just below the maximum object pixel intensity $S_{R\,max}$, here we set T_{init} to a value just above the maximum background pixel intensity $S_{B\,max}$. This produces the desirable result of increasing the number of initially pulsing neurons but is unavailable to the previous method because it causes $\beta_{\max}$ to take on a tiny value.

To demonstrate the effectiveness of the model in overcoming very large intensity range overlaps, we return to the example of the uniform distributions introduced in 2.1.5 above. Figure 2.7 shows results obtained on an image where $S_{B\,max}$ was set to 0.9. In (a) we see the original image and find that the central square is barely visible to the human eye while in (b) we display the perfect segmentation obtained by the new method.



(a) (b)

Figure 2.7. Perfect segmentation with the alternative method. (a) Input image. (b) Segmentation obtained with $r = 1.0$.

Figure 2.8(a) compares the performance of the alternative method with that of the K&R method as we vary the value of $S_{B\,max}$ on the original symmetrically overlapping, uniform distribution example. Performance here is measured as the mean percentage of correctly classified pixels per image region. Note the similarity in the performance of the alternative method to the results obtained in 2.1.7.2. This is not co-incidental. Failure in both cases stems from increasing sparseness of super-threshold object pixels, which eventually breaks down the spreading activation process hence preventing perfect segmentation.

Figure 2.8(b) compares the processing time for the two methods as we vary the value of $S_{B\,max}$. Times were obtained using a moderately efficient Matlab implementation running on a 500MHz AMD Athlon processor with 128MB RAM. We note that there is an upwards trend in the time required by each method as we increase the overlap between the classes. This is an expected result of the requirement for more fast-linking loops with sparser pixel distributions. We also note that the alternative method is considerably faster than the K&R method particularly at low values of $S_{B\,max}$.

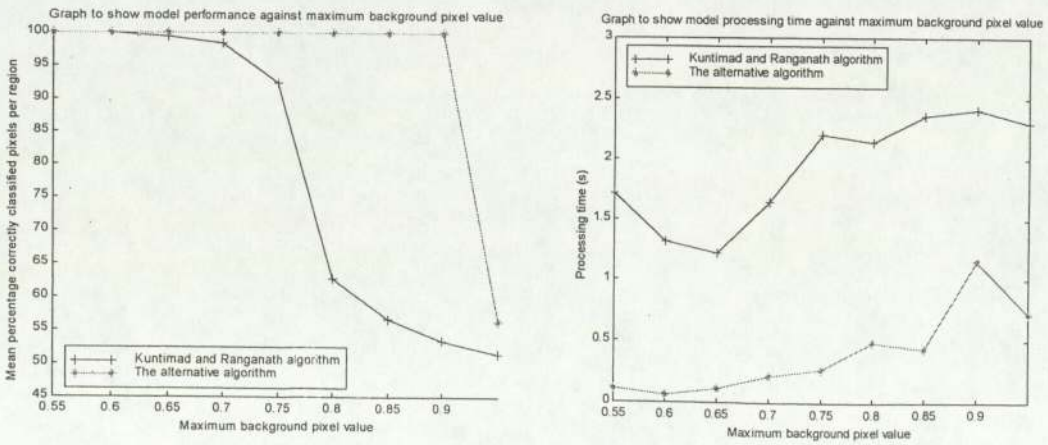


Figure 2.8 Model performance as a function of $S_{B\,max}$. (a) Segmentation performance. (b) Processing time

2.1.9 Motivating Subsequent Work

This work on perfect segmentation algorithms has made it clear that while the performance of the PCNN is affected by the border geometry between image regions, any reliance on this border geometry is a weakness in more general applications. We therefore set out to develop algorithms that place no reliance on border geometry and ought by virtue of this to be of more general utility.

Kuntimad and Ranganath conclude their paper on perfect segmentation by presenting a number of problems for future consideration. Amongst these were the following.

- The investigation of the properties of PCNNs with simplified thresholding and linking mechanisms.

- The development of multispectral PCNNs

- The development of optimal parameterisation methods based on the intensity probability density function of the image.

The first of these was dealt with to some extent in the modifications to the K&R method made above but will be pursued in further detail in subsequent sections. The second problem has not yet been addressed but will also be the focus of subsequent work presented here. The final problem has also yet to be addressed but it is felt that the process will be more tractable and any solutions more useful if we proceed within a framework that disregards border geometry as suggested above. This will be the principle motivation for the models and parameterisation methods developed below.

To begin this developmental process, we first sought a definition of the PCNN that was general enough to apply to the whole model family and simple enough to suggest an analytically trivial initial model from which to progress. The definition selected was as follows: "A dynamic thresholding model with limited local spatial context" and this led to the use of simple dynamic thresholding as the starting point from which increasingly complex PCNN model variants were incrementally developed.

2.2 The Simple Dynamic Thresholding Method

Simple dynamic thresholding will be the term used here to describe a primitive image segmentation algorithm wherein a global intensity threshold variable is used to segment our input image. Like the PCNN, the algorithm works in discrete time and at each time step the threshold is set to a new value allowing different groups of pixels to be segmented. Ideally, individual image regions or classes of image regions will be segmented at each time step but if the intensity ranges of different regions overlap then perfect segmentation will be impossible with this method.

2.2.1 Optimising Performance

Since no spatial context is available with the simple dynamic thresholding method, observations of the image itself will be of no use in optimising the performance of the algorithm. Instead, we need to focus solely on the intensity probability distribution of the image. Simple statistical procedures can then be applied to optimise model performance according to criteria such as error minimisation.

Consider an image consisting of any number of discrete regions but having only two classes of pixel, C_1 and C_2, we wish to separate from one another. Let the pixel intensity values of C_1 and C_2 be drawn from a pair of one dimensional Gaussian probability density functions of differing mean and let the intensity values of C_1 be drawn from the Gaussian of greater mean. Further, let these p.d.f.s overlap to some significant degree so that perfect segmentation cannot be achieved by global thresholding. Figure 2.9 graphically illustrates the situation with which we are presented.

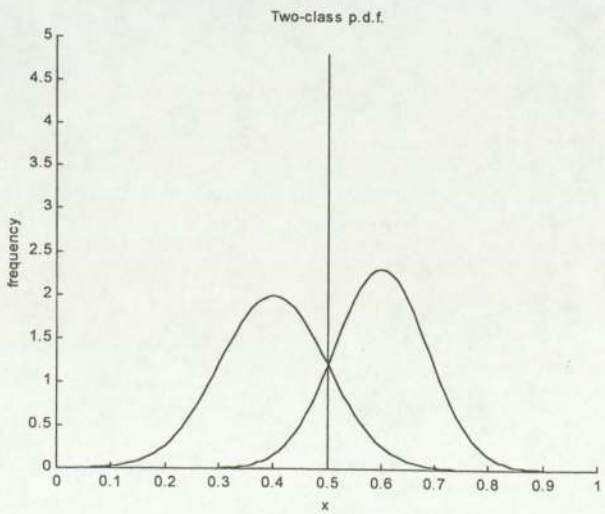


Figure 2.9 Schematic illustration of the joint probability density

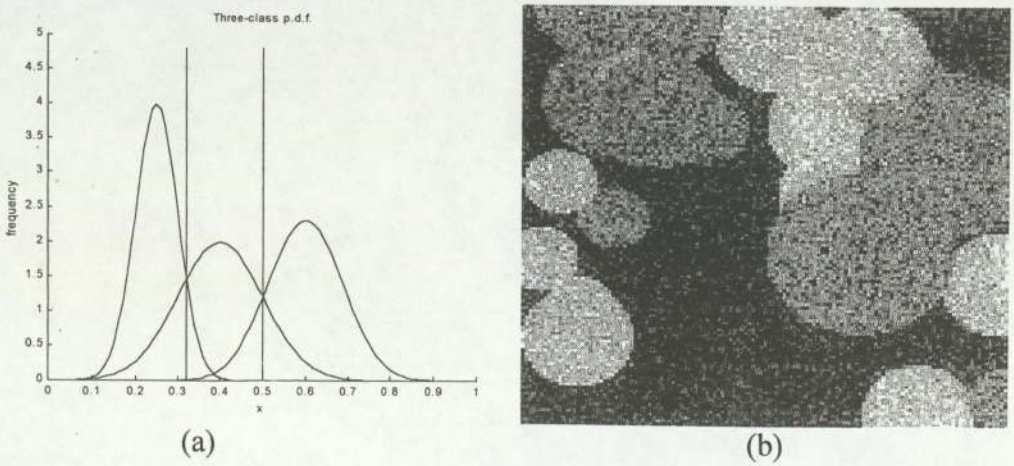
According to probability theory, the minimum error decision boundary between the two classes is the intensity at which the two Gaussian distributions have equal probability. This value is marked with a vertical line in Figure 2.9 and corresponds to the optimal threshold value for our simple dynamic thresholding algorithm according to the criterion of minimum classification error.

We can calculate this optimal value analytically by using the equation for a one-dimensional Gaussian p.d.f. (20), and setting the value to be equal for the two Gaussians in question. Disregarding constant terms, we arrive at (21) and by plugging in mean and variance values for the two distributions we arrive at a quadratic in x that we can solve for the minimum error decision boundary. Where no confusion exists, this decision boundary intensity will henceforth be referred to as S_{mid}.

$$p(x) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\} \tag{20}$$

$$\frac{(x-\mu_1)^2}{\sigma_1^2} + \ln(\sigma_1^2) = \frac{(x-\mu_2)^2}{\sigma_2^2} + \ln(\sigma_2^2) \tag{21}$$

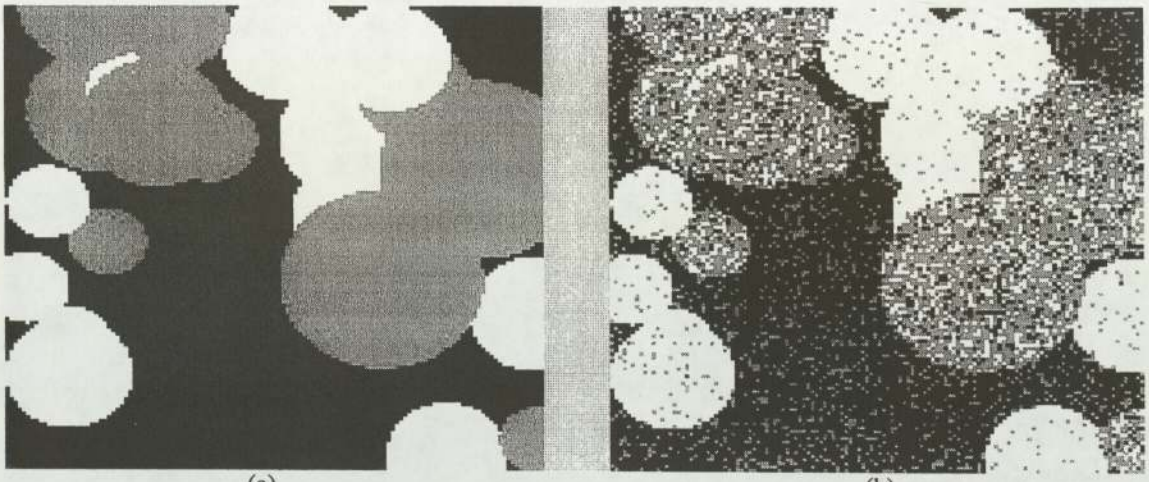
This method can easily be extended to cases of more than two classes by calculating a series of these S_{mid} values, referred to here as S_{mids}, and using each value as the threshold at one time step. Obviously we could apply all the threshold values concurrently but the temporal segmentation mirrors that of the PCNN and leads more logically into subsequent models. Figure 2.10(a) displays the joint p.d.f. of a three-class example with means of 0.25, 0.4 and 0.6 and variances of 0.0025, 0.01 and 0.0075 respectively. Two decision boundaries are marked as before. Figure 2.10(b) displays a 128 by 128 pixel image formed from these classes that will serve as a general example to which each method developed here will be applied.



(a) (b)

Figure 2.10. The three-class case. (a) Probability density function. (b) Corresponding three-class image.

Figure 2.11 displays the segmentation results obtained for the example image using simple dynamic thresholding at the optimal decision boundaries. As with the perfect segmentation methods, 2.11(a) shows a template indicating the perfect segmentation while 2.11(b) shows an image formed by inverting the time step at which the pixel was found to be super-threshold and scaled to the same intensity range as the input image. We note that the intensities correspond roughly to those found in the original image but that this correspondence is relative rather than absolute: brighter regions in the input image will correspond to brighter regions in the output image but the intensity values themselves are not preserved.



(a) (b)

Figure 2.11. Results using Simple Dynamic Thresholding. (a) Template. (b) Segmentation.

2.3 The Safe Method

The first PCNN variant motivated by the simple dynamic thresholding method will be referred to here as the safe method. It is hoped that by combining the same decision boundaries as used in thresholding with minimal spatial information we ought to be able to achieve better results with little computational overhead. In the following two subsections we first introduce the model form employed and then the parameterisation method designed to achieve optimal performance given that model form.

2.3.1 The Model

Following the nomenclature introduced in section 1.3, the model form selected here is a single pass variant featuring direct feeding, zero-thresholded stepped linking, fast linking, linear threshold decay and threshold reset. Mathematically, this corresponds to the algorithm introduced in 1.3.6 with the value of θ_L being set to zero.

2.3.2 The Method

The parameterisation method is applied to the threshold and linking coefficient values. Its application is first introduced on the same two-class example as discussed in 2.2 above, before the extension to more than two classes is dealt with.

2.3.2.1 Threshold

The initial threshold for all neurons is set to the same value T_{init}. This value is selected as a level very unlikely to be generated by the lower intensity Gaussian distribution. This value may be calculated graphically by looking at the joint probability density but we prefer to apply an analytical solution by employing a value several standard deviations above the mean of the distribution. 3.5 standard deviations was found to be quite effective across a range of images. Figure 2.12 illustrates the position of the threshold as a vertical line topped with a star, which has been added to the two-class p.d.f. of figure 2.9.

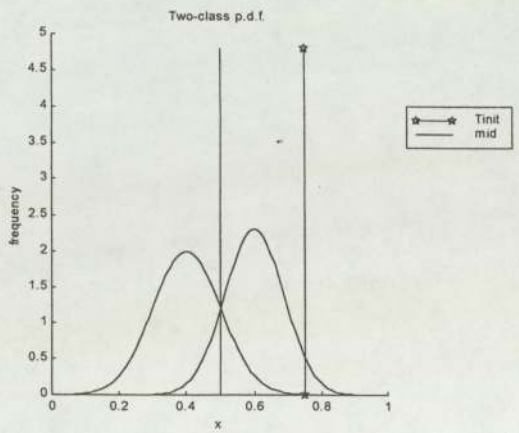


Figure 2.12. Two-class p.d.f. with T_{init}

2.3.2.2 Linking Coefficient

The linking coefficient β is set so that the minimum pixel intensity value that can cause its neuron to pulse corresponds to S_{mid}. The correct value to achieve this can be found by looking at the equation for PCNN internal activity in (6c). Since the linking is thresholded at zero, we note that this is the logical OR condition and that the maximum linking input L_{max} is one. By setting the value of U_{ij} equal to T_{init}, substituting S_{mid} and L_{max} into the equation and rearranging, we arrive at the desired prescription for β:

$$\beta = \left(T_{init} / S_{mid}\right) - 1 \tag{22}$$

2.3.3 Extending to more than two Classes

With more than two classes, we calculate a series of decision boundaries S_{mids} between our classes and also a series of threshold values T_{vals} set 3.5 standard deviations above the lower intensity mean at each time step. Figure 2.13 displays once again the joint p.d.f of our three-class example along with the required threshold values marked as lines topped with stars.

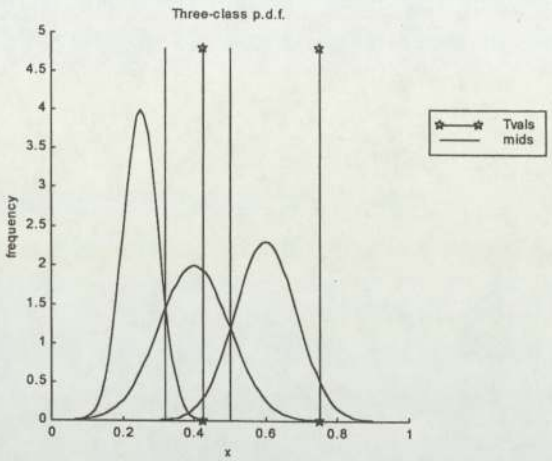


Figure 2.13. Thee-class p.d.f. showing T_{vals} as well as S_{mids}.

Given these values, we can calculate an appropriate sequence of linking coefficients according to:

$$\beta[n] = \left(T_{vals}[n] / S_{mids}[n]\right) - 1 \tag{23}$$

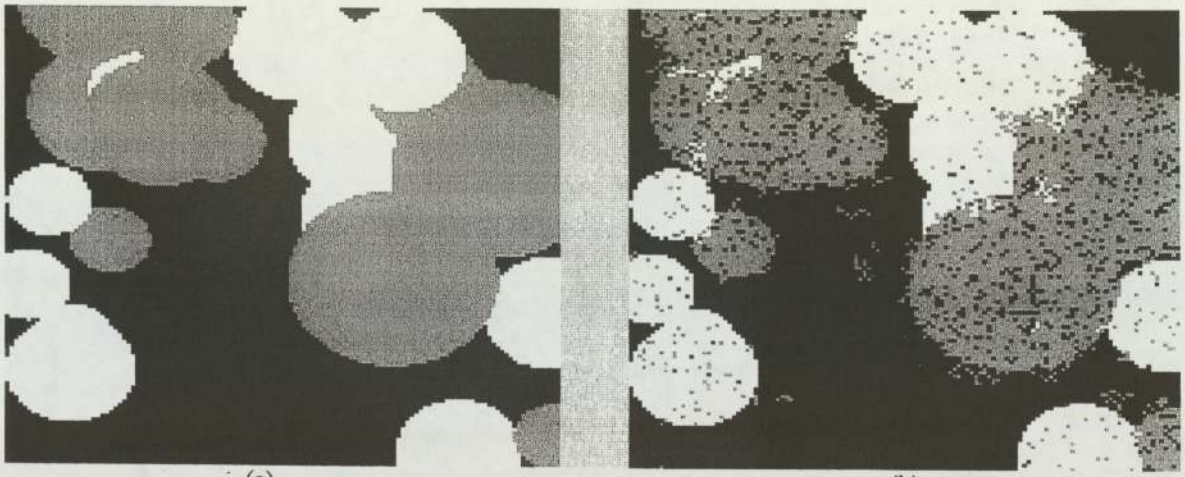
We note that the linking coefficient β is no longer constant but takes on a value at each time step dependent upon the characteristics of the neighbouring classes under consideration. Similarly, the threshold decay term dT becomes time dependent, being set so that the required T_{val} is obtained at each time step. This makes more sense than other methods that fix these values as constants since the optimal values will depend on the generating distributions for each pixel class. This results in the following forms being used for the internal activation and threshold equations:

$$U_{ij}[n] = S_{ij}\{1 + \beta[n] L_{ij}[n]\} \tag{24}$$

$$T_{ij}[n] = \begin{cases} \Omega & \text{if } Y_{ij}[n] = 1 \\ T_{ij}[n-1] - dT[n] & \text{otherwise} \end{cases} \tag{25}$$

2.3.4 Results

The results for all subsequent PCNN variants will be presented in the following manner. A pulse matrix **P** of the same size as our input image is used to store the time of pulsing of each neuron in the network. The values within this pulse matrix are then subtracted from a constant equal to the number of time steps used plus one and normalised. Finally we display the result as an image which corresponds to the output image formed earlier for simple dynamic thresholding. As in Figure 2.11, Figure 2.14 displays a template indicating perfect segmentation in (a) and the segmentation achieved by this method in (b). **W** here was a kernel of the type suggested by Kuntimad and Ranganath with radius $r = 1.5$.



(a) (b)

Figure 2.14. Results using the Safe Method. (a) Template. (b) Segmentation

2.3.5 Why The Method Works

As can be seen from the results above, the safe method performs demonstrably better on this segmentation task than simple dynamic thresholding. If we look again at the joint probability density schematic for two classes we can see exactly how this improvement is achieved. The error for the simple dynamic thresholding algorithm can be split into two components. Errors derived from pixels generated by the lower intensity distribution but having intensities greater than S_{mid} will be termed high-tail errors while errors derived from pixels generated by the higher intensity distribution but having intensities lower than S_{mid} will be termed low-tail errors.

The safe method allows us to greatly reduce high-tail errors because of the low probability of these high-tail pixels lying in proximity to the border of the higher intensity region. Note that this assumed pixel arrangement is not reliant on particular border geometry conditions as

were the perfect segmentation methods. Instead we rely on the low ratio of border pixels to total pixels, which is a general property of solid regions with reasonably smooth edges. In the case of Gaussian classes, we also take advantage of the low ratio of high-tail pixels to total pixels in the lower intensity region. It is the combination of these two factors that results in reduced high-tail errors.

Low-tail errors however are unaffected by this method since the beta value calculated cannot cause pixels below S_{mid} to pulse. This can clearly be seen when we compare Figures 2.11 and 2.14. High-tail errors here correspond to higher intensity pixels appearing on a lower intensity region of the output image while low-tail errors correspond to lower intensity pixels appearing on a higher intensity region. We see that in Figure 2.14, high-tail errors have been virtually eliminated but that low-tail errors are still present.

Obviously we want to improve on the performance of the safe method and the best way to do so seemed to be to enable our algorithm to cope with low-tail as well as high-tail errors. This was the motivation for the augmented method.

2.4 The Augmented Method

The augmented method is so called because it uses a model designed to produce a larger internal activation than the safe model under certain controlled conditions driven by the spatial context. By so doing, the model should be capable of causing the pulsing of neurons with lower feeding inputs than was previously possible, this being the necessary behaviour for overcoming low-tail segmentation errors.

We now present a modification to the model form designed to enable this augmented activation before introducing the alterations to the parameterisation method required by the new model form.

2.4.1 Model Modification

The augmented method uses a model form very similar to the safe method, the only alteration being in the linking field. The linking input to a neuron is still instantaneous but is no longer thresholded so that its value simply becomes the two-dimensional convolution of the output matrix $\mathbf{Y}$ with the kernel $\mathbf{W}$.

$$L_{ij}[n] = \sum_{kl} W_{ijkl} Y_{kl}[n] \tag{26}$$

This alteration to the linking can be interpreted as the introduction of additional spatial context information into the model. The linking input to a neuron in the safe model contained only qualitative information regarding whether any of its neighbours were pulsing. By removing the thresholding, we introduce quantitative information regarding the number of pulsing neighbours and also, if a larger kernel is employed, the distance to these pulsing neighbours.

This quantitative information enables the model to overcome low-tail as well as high-tail errors and looking back at the results from the safe method should make it clear exactly how it does so. Here we saw that low-tail errors led to lower intensity pixels appearing on a higher intensity region of the output image. These lower intensity output pixels result from a neuron with low feeding input pulsing later than its neighbours. Thus at the time step that generated the higher intensity region in the output, we would typically find these rogue neurons surrounded by several pulsing neighbours.

With thresholded linking, the only information being passed to such a neuron is that pulsing activity is occurring within its neighbourhood. Contrastingly, the augmented method allows the number of pulsing neighbours to be communicated, causing the linking input to take on a high value and forcing the low-feeding neuron to pulse with its neighbours. Thus the introduction of additional spatial context information potentially allows for much smoother and less error-prone segmentation in this situation.

2.4.2 Prescription for Beta

While the additional information introduced here is potentially valuable to our model, there is a price to be paid. The determination of appropriate linking coefficient values was obvious for the safe model: we simply applied an error-minimisation condition. Here, an optimal value becomes less certain and we are forced to employ a heuristic prescription, which we introduce below in its two-class form for ease of explanation but with no loss of generality.

$$\beta = \lambda \frac{\left((T_{init} / S_{mid}) - 1\right)}{L_{max}} \tag{27}$$

We now note the appearance of the safe method's prescription for beta as the numerator of the fraction here. The augmented prescription is a modified form of the original designed to take into account the alteration made to the linking field. The denominator of the fraction L_{max} is defined as the maximum linking input a neuron can receive. This is simply the sum of the values present in the linking kernel **W**. It makes intuitive sense for beta to be inversely proportional to this constant since the more linking weight there is, the less amplification provided by the linking coefficient we require.

However, simply dividing by L_{max} would produce a value for beta too small for effective segmentation. This can be demonstrated if we look once again at the internal activation for a neuron: $U_{ij}[n] = S_{ij}\{1 + \beta L_{ij}[n]\}$. If we divide beta by L_{max}, then only with maximal linking input, $L_{ij} = L_{max}$, would the internal activation take on the same value as it had for the safe method. Any linking input value below the maximum would result in lower internal activation than previously whereas what we want is higher activation where the linking input is large and lower activation where the linking input is low.

We therefore introduce the lambda coefficient as a positive constant greater than one to increase the value of beta in a controlled manner. Now the internal activation for the safe and augmented models becomes the same when the linking input to a neuron takes on the following value:

$$L_{ij} = L_{\max} / \lambda \tag{28}$$

Below this value, internal activity is reduced but above it, activity is increased or *augmented*, forcing pixels with feeding intensities below S_{mid} to pulse with their neighbours. In this manner, low-tail errors are greatly reduced.

We might think that a value of two for lambda would be appropriate so that half the maximum linking input would match the internal activation of the safe method and capture neurons with feeding intensities as low as S_{mid}. In practice however a value of three is generally found to be more successful in allowing pulsing activity to spread within a region while at the same time preventing it from crossing boundaries between regions. A value greater than three does more to encourage intra-region spreading but also results in more inter-region spreading. This may be more appropriate in certain circumstances and gives the method controlled flexibility.

as it generally succeeds in allowing activation to spread within a region while at the same time preventing it from crossing boundaries between regions.

The extension of this method to more than two classes works in the same way as for the safe method. We have a sequence of decision boundaries S_{mids}, a sequence of threshold values T_{vals} and consequently also dT values, and a sequence of linking coefficients calculated by plugging these values into (27).

2.4.3 Results

Figure 2.15 follows the format used in 2.11 and 2.14 for our three-class problem. The template is again displayed in 2.16(a) for comparison with the segmentation achieved, which appears in 2.16(b). The same kernel was used here as for the safe method while the lambda coefficient was set to a value of 3.0.

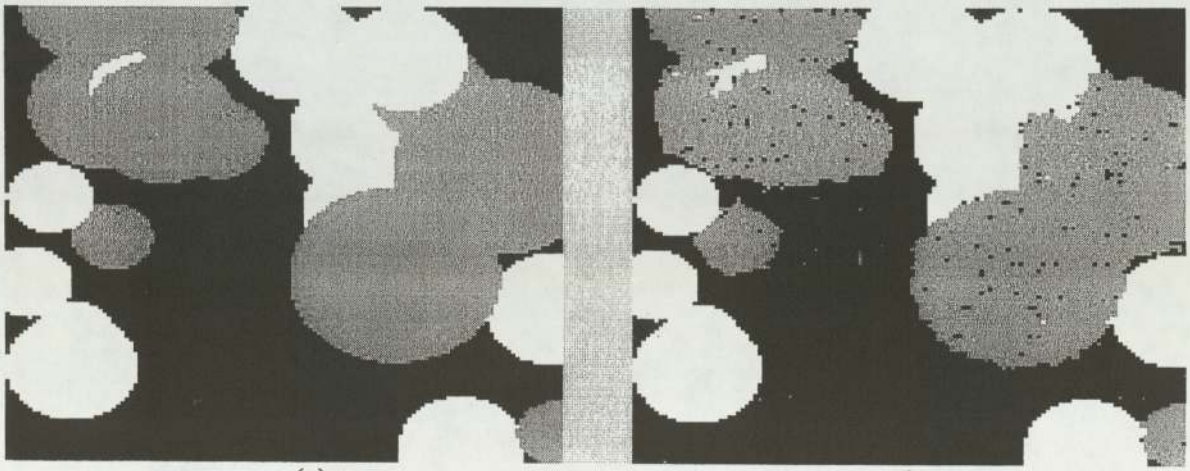


Figure 2.15. Results using the Augmented Method. (a) Template. (b) Segmentation.

2.5 The Intensity-Coded Method

While the results from the augmented method proved to be a considerable improvement over the safe method, it was still felt that we might be able to do better. Once again, we sought to incorporate additional spatial context information into the model by further modifying the linking field. In both the safe and augmented models each neuron communicates to its neighbours whether or not it is pulsing. At the receiving neuron, the safe model applies a logical OR to arrive at the linking value while the augmented model takes a weighted sum over the linking neighbourhood but the information being transmitted is limited in either case

to zeros and ones. In the subsections below we present the change made to the model to enable more sophisticated transmission before introducing the requisite alteration to the parameterisation method.

2.5.1 Model Modification

The intensity-coded method allows pulsing neurons to transmit the intensity of their feeding pixel S_{ij} rather than just a value of one. Non-pulsing neurons still transmit a value of zero. This modification is implemented by using (29) in place of (26) and we now return to the two-class example used above to explain why we might expect an improvement to performance by employing intensity coding.

$$L_{ij}[n] = \sum_{kl} W_{ijkl} Y_{kl}[n] S_{kl} \tag{29}$$

Our confidence that a given pixel belongs to the higher intensity class varies with pixel intensity and while this principle is already used in determining whether a given neuron will pulse, previous models have not allowed the communication of this confidence to a neuron's neighbours. Intensity coding effectively enables the communication of this confidence measure in an easily implemented manner thus adding sophistication to the decision process and potentially improving overall segmentation/classification.

2.5.2 Prescription for Beta

This further modification to the linking field demands another alteration to the prescription for beta. The new prescription is as follows:

$$\beta = \lambda \frac{\left((T_{init} / S_{mid}) - 1\right)}{\left(L_{max}\, S_{mean}\right)} \tag{30}$$

The new value S_{mean} introduced here is the mean intensity of the higher intensity distribution. Now the internal activation becomes the same as for the safe model when the linking input to a neuron takes on the following value:

$$L_{ij} = (L_{\max} S_{mean}) / \lambda \tag{31}$$

Thus this reference value is now scaled by the mean of the higher intensity distribution. In a perfect segmentation, the only neurons pulsing at time step one will be fed by pixels drawn from this distribution so the mean feeding intensity of pulsing neurons will approximate S_{mean} increasingly accurately during the fast linking phase of the algorithm. While perfect segmentation will not be achieved in general, model performance can be seen as an approximation to this goal and the closer the model gets to perfect segmentation, the closer the mean feeding intensity of pulsing neurons gets to S_{mean}.

Scaling by S_{mean} therefore effectively balances the introduction of S_{kl} as a multiplicative factor in the linking equation since the mean value of S_{kl} for pulsing neurons tends towards S_{mean} as we tend towards perfect segmentation.

The extension to more than two classes is identical to the previous methods and will not be repeated here.

2.5.3 Results

Figure 2.16 follows the same format as before in displaying the results for this method on our three-class example. The template appears in (a) while the segmentation appears in (b). Again **W** is an $r = 1.5$ kernel of the K&R type and the lambda coefficient is set to 3.0.

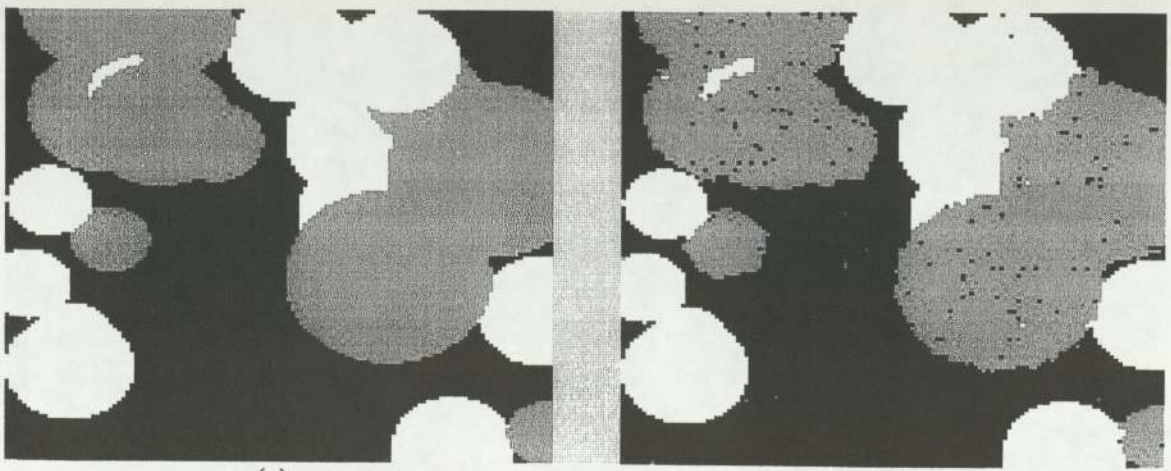


Figure 2.16. Results using the Intensity-Coded Method. (a) Template. (b) Segmentation.

2.6 Quantitative Comparisons

We now test the performance of the three parameterisation methods on a slightly different problem and using simple dynamic thresholding as a baseline measure for comparison. Again we have three Gaussian classes with the same means as before but here we set the variances of each Gaussian to be equal to one another. We then gradually increase the variance and consequently the degree of overlap between the classes and measure segmentation performance for each model as the percentage of correctly classified pixels. We also record the processing time for each model at each variance value. Figure 2.17 displays the results of this comparison, which was performed using a moderately efficient Matlab routine running on a 500MHz computer as in section 2.1.8. We should note here that apparent zero processing times for the simple dynamic thresholding method are due to Matlab's inability to register time spans smaller than 0.006 seconds. Lambda coefficients of 3.0 were used for both the augmented and intensity-coded methods and a linking radius of 1.5 was maintained throughout.

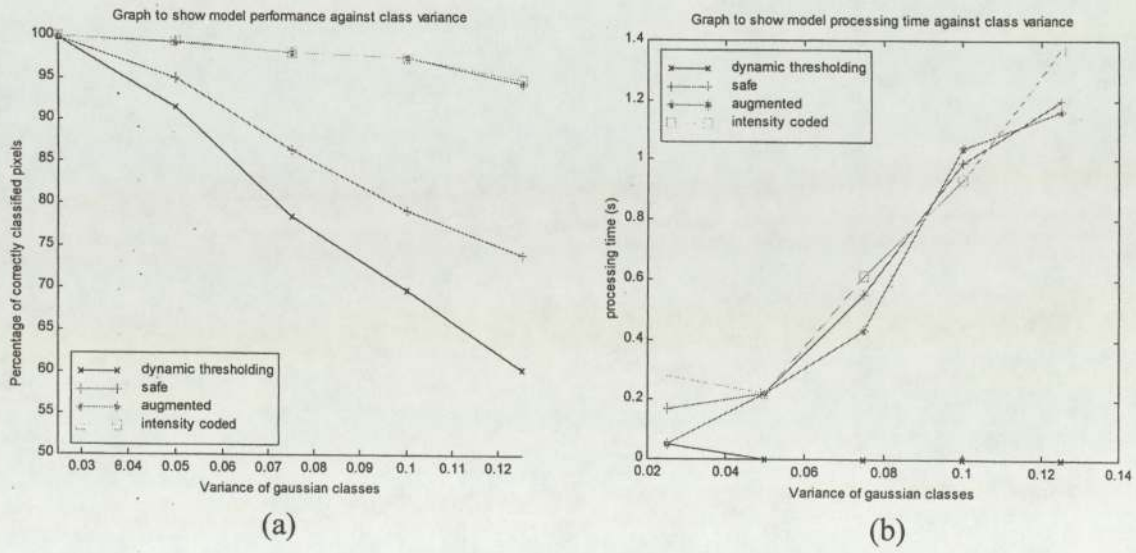


Figure 2.17. Quantitative results. (a) Performance. (b) Times.

2.6.1 Comparisons with Perfect Segmentation Methods

Having compared the performance characteristics of our newly developed parameterisation methods with one another, we felt that it was pertinent now to see how well they measured up to the perfect segmentation methods explored earlier. We chose to compare the augmented method with the perfect segmentation methods on the same example images as employed in section 2.1.8. We recall that these images featured rectangular object regions and symmetrically overlapping, uniform distributions.

A linking radius of 1.0 was used for the perfect segmentation methods while a radius of 1.5 was employed for the augmented method here. Because of the border geometry conditions present, pulsing activity was less inclined to cross between the image regions than in the previous sections. Consequently, a slightly larger lambda coefficient of 4.0 was employed in the augmented method to encourage spreading behaviour. Figure 2.18(a) displays segmenting performance of the methods while 2.18(b) displays processing time, both over the maximum background pixel value.

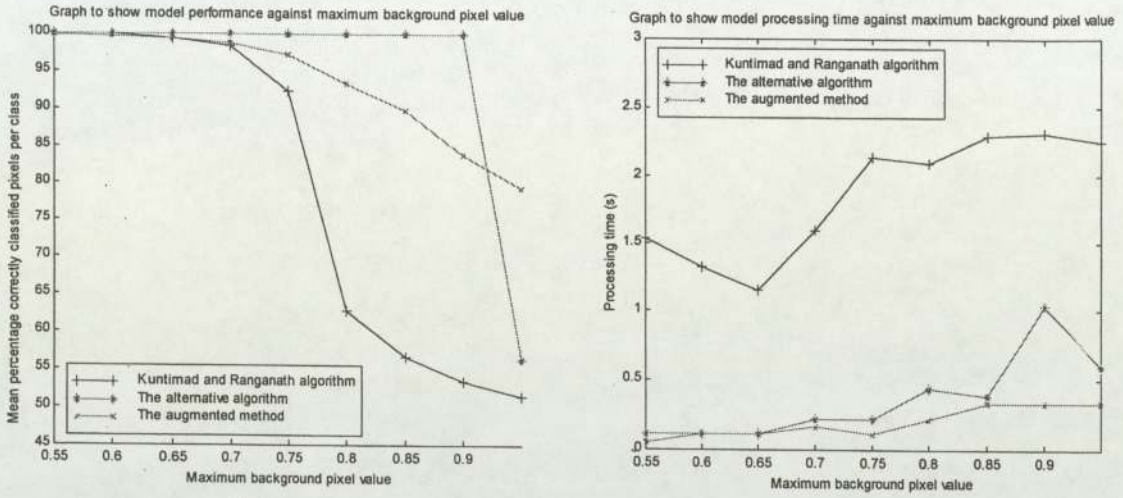


Figure 2.18. Comparison with 'perfect' methods. (a) Performance. (b) Times.

2.7 Multi-Channel Extensions

The model forms and parameterisation methods so far introduced are designed only to deal with single-channel input data: greyscale images. In general image processing applications however, we will often be presented with multi-channel data such as RGB colour images. While it is possible to convert these to greyscale images before presenting them as model inputs, information is necessarily lost during the conversion process. It is therefore desirable to modify our methods to deal with multi-channel data in a natural way and the following subsections will propose modifications suitable for this purpose.

2.7.1 An Example

The example problem used here will be based on the three-class, single-channel example introduced in section 2.2.1 and graphically illustrated in figure 2.10. Figure 2.19 displays the joint p.d.f. for the three classes across three channels: red, green and blue. Each channel features the same three Gaussian distributions but the order of the classes is different in each

case as indicated by the different traces used. For example, the class that is brightest in the red channel is of medium brightness in the green channel and is the darkest class in the blue channel. Since each class occupies a different position in the hierarchy in each channel, the mean and variance of the total intensity for each class should be equal. Decision boundaries and appropriate threshold values for the PCNN are displayed as in the single-channel case.

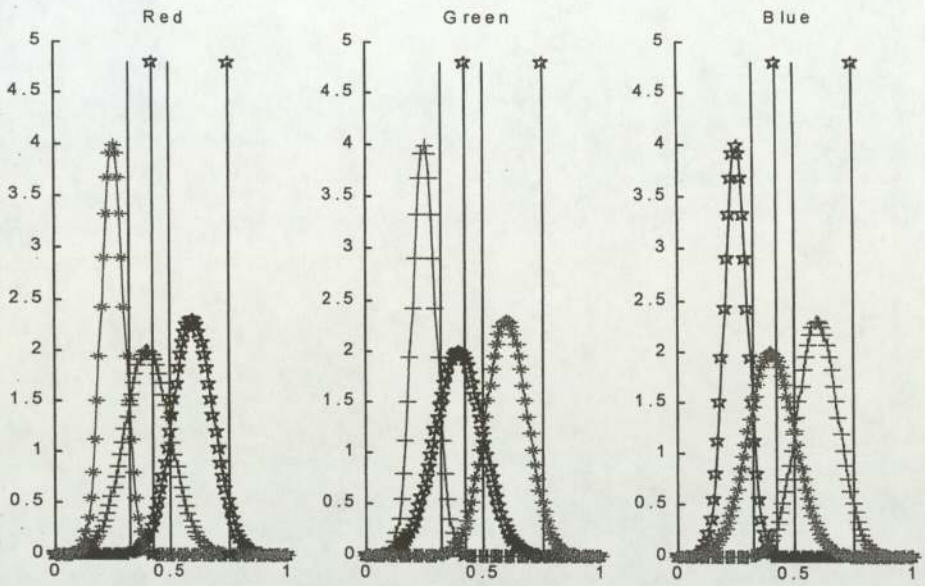
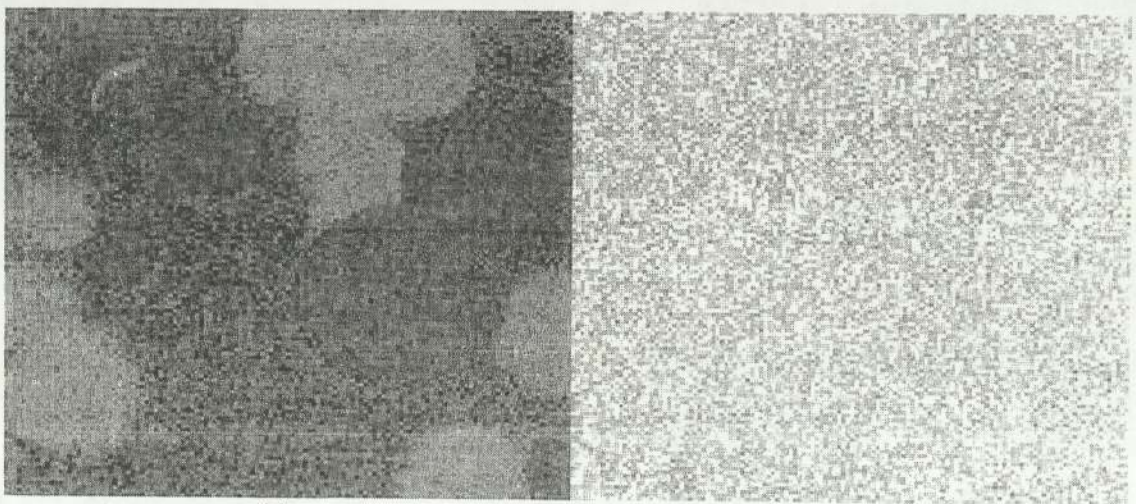


Figure 2.19. Three-channel joint p.d.f. for three classes.

Figure 2.20 displays an RGB image formed from the classes above. In (a) we see the colour image itself and find that they are clearly distinguishable from one another while in (b) we form a greyscale image by summing over the channels and see that the three classes become identical in this case and as such would be impossible to separate by any of the single-channel methods presented above. This RGB image will be used to test each of our methods as we extend them into multi-channel form.



(a) (b)

Figure 2.20. Three-channel test image. (a) RGB image. (b) Greyscale version of the image.

2.7.2 Simple Dynamic Thresholding

To extend the simple dynamic thresholding method to multiple channels we need to have a global threshold for each channel. To initialise the model we begin by selecting an order in which to segment the classes. In the current case the order is arbitrary but in general we would segment in descending order of total mean intensity. By using this form of ordering we avoid the potential problem of attempting to segment a class when another class yet to be segmented has greater mean intensity in each channel. All the methods currently developed would fail in this case.

After ordering the classes, we run through them in order setting appropriate threshold value for each channel according to our error minimisation criterion. Each class now has a set of threshold values: three in the example presented above. Addressing the first class, we apply a logical AND over the channels, classifying only pixels with channel intensities greater than all thresholds as belonging to that class. Thresholds are then moved on to the next set of values and the process is repeated until the threshold values are exhausted and all classes have been addressed.

2.7.3 The Safe Method

The safe model is altered so as to feature the following:

- Multiple direct feeding channels
- Threshold values for each channel
- Linking coefficients for each channel
- Internal activation values for each channel

As with the simple dynamic thresholding method, the output at a given time step is calculated by applying a logical AND over all channels. This AND is applied here to the comparison of intensity values with the thresholds initially and of internal activations with the thresholds within the fast linking loop. Mathematically, the three-channel version of the model is expressed as follows:

REPEAT

$$Y_{ij}[n] = \begin{cases} 1 & if\ ((S_{ij}^R > T_{ij}^R[n-1]) \,\&\, (S_{ij}^G > T_{ij}^G[n-1]) \,\&\, (S_{ij}^B > T_{ij}^B[n-1])) \\ 0 & otherwise \end{cases} \quad (32a)$$

WHILE there is any change in pulsing activity

$$L_{ij}[n] = \begin{cases} 1 & if\ \sum_{kl} W_{ijkl} Y_{kl}[n] > 0 \\ 0 & otherwise \end{cases} \quad (32b)$$

$$U_{ij}^R[n] = S_{ij}^R \left\{ 1 + \beta^R[n] L_{ij}[n] \right\} \quad (32c)$$

$$U_{ij}^G[n] = S_{ij}^G \left\{ 1 + \beta^G[n] L_{ij}[n] \right\} \quad (32d)$$

$$U_{ij}^B[n] = S_{ij}^B \left\{ 1 + \beta^B[n] L_{ij}[n] \right\} \quad (32e)$$

$$Y_{ij}[n] = \begin{cases} 1 & if\ ((U_{ij}^R > T_{ij}^R[n-1]) \,\&\, (U_{ij}^G > T_{ij}^G[n-1]) \,\&\, (U_{ij}^B > T_{ij}^B[n-1])) \\ 0 & otherwise \end{cases} \quad (32f)$$

END

$$T_{ij}^R[n] = \begin{cases} \Omega & if\ Y_{ij}[n] = 1 \\ T_{ij}^R[n-1] - dT^R[n] & otherwise \end{cases} \quad (32g)$$

$$T_{ij}^G[n] = \begin{cases} \Omega & if\ Y_{ij}[n] = 1 \\ T_{ij}^G[n-1] - dT^G[n] & otherwise \end{cases} \quad (32h)$$

$$T_{ij}^B[n] = \begin{cases} \Omega & if\ Y_{ij}[n] = 1 \\ T_{ij}^B[n-1] - dT^B[n] & otherwise \end{cases} \quad (32i)$$

UNTIL all neurons have pulsed

The superscripts on a number of variables here assume the presence of red, green and blue channels but the method can be applied to arbitrary channels in practice. Sets of decision boundaries for each class are calculated as with the simple dynamic thresholding model. Thresholds are calculated in the same way as in the single-channel case and given these values, suitable linking coefficients and threshold decay terms for each channel become available.

However, the use of the logical AND scheme to combine information between channels tends to inhibit the spread of pulsing activity required for effective segmentation/classification. As a

result, we tend to use thresholds at slightly lower value than before and linking coefficients slightly larger than our prescription suggests. In our example, thresholds were set 2.5 standard deviations above class means rather than 3.5 as in the single channel case and the linking coefficients were multiplied by 2 after the prescription had been applied. This proved to be fairly effective here but see also section 2.8. Thus the linking coefficient for a channel x at time n is calculated as follows:

$$\beta^x[n] = 2\left((T^x_{val}[n-1]/S^x_{mid}) - 1\right) \tag{33}$$

2.7.4 The Augmented Method

The extension of the augmented method into multi-channel form runs along very similar lines to the safe model. Once again, the only change to the model is in the linking equation and since linking is not multi-channel the alteration is identical to the single-channel version.

The prescription for the linking coefficients is applied to each channel of the multi-channel method exactly as it was to the single-channel case but as with the safe method we find that the values calculated need to be increased by a factor of 2 to overcome the inhibitory effect of the logical AND scheme. Threshold values were the same as for the multi-channel safe method. Here this is achieved by multiplying the value of the lambda coefficient by 2 so that for channel x at time n, the prescription becomes:

$$\beta^x[n] = \lambda \frac{\left((T^x_{val}[n-1]/S^x{}_{mid}) - 1\right)}{L_{max}} \tag{34}$$

2.7.5 The Intensity-Coded Method

The multi-channel, intensity-coded method uses a model that replaces the linking equation of the safe model with (35) so that intensity coding is multiplicative across the channels. Note that here we assume red, green and blue input channels but the equation generalises easily. Aside from this alteration, the model is identical to that employed by the multi-channel safe method.

$$L_{ij}[n] = \sum_{kl} W_{ijkl} Y_{kl}[n] S^R_{kl} S^G_{kl} S^B_{kl} \tag{35}$$

The prescription for β takes the linking modification into account by using (36). This appropriately balances the internal activation so that the method remains effective.

$$\beta^x[n] = \lambda \frac{\left((T^x[n-1]/S^x{}_{mid}) - 1\right)}{\left(L_{max}\, S^R_{mean}\, S^G_{mean}\, S^B_{mean}\right)} \tag{36}$$

As with the augmented method, the lambda coefficient is multiplied by 2.

2.7.6 Results

2.7.6.1 Qualitative Results

Figure 2.21 displays the segmentations achieved by our four parameterisation methods. These results can be compared directly to the single-channel segmentations displayed in Figure 2.11, 2.14, 2.15 and 2.16 and the multi-channel PCNNs appear to perform less well in this comparison. We believe that this is due to the high variance central Gaussian in each channel. In the single-channel case this is easily the least well segmented class because of the large intensity range overlaps with its neighbours and in the multi-channel case it appears in each class, degrading performance in each case.

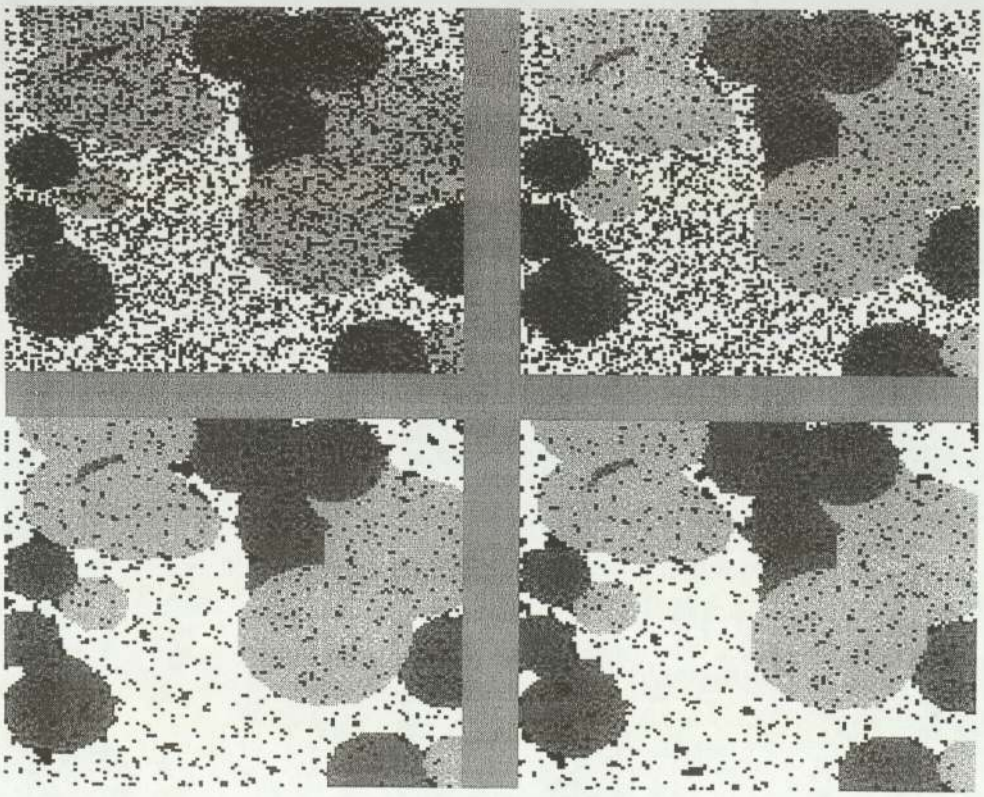
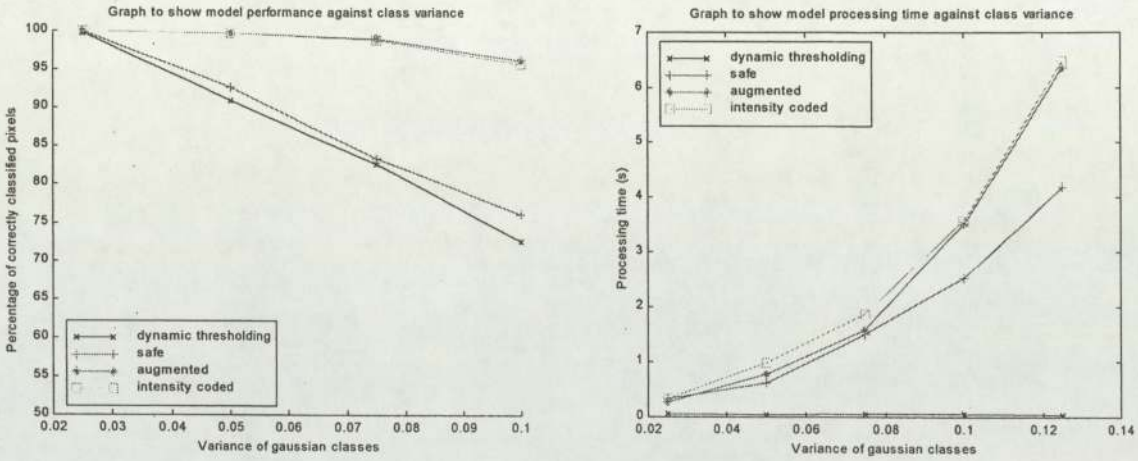


Figure 2.21. Multi-channel segmentations. Clockwise from top left, the simple dynamic thresholding method, the safe method, the intensity-coded method and the augmented method.

2.7.6.2 Quantitative Results

Figure 2.22 shows the results equivalent to those in Figure 2.17 for the single-channel methods. Once again, variances of the Gaussians were made equal to one another and gradually increased to test performance with varying levels of overlap between classes. Here, each class was drawn from a distribution of different mean for each channel as in section 2.7.1 so that the greyscale distributions would be identical. Threshold values for the PCNNs were set 2.5 standard distributions above the Gaussian means while linking coefficients were multiplied by 4/3 compared with the single-channel case, so that lambda for the augmented and intensity-coded methods was set to 4.0.



(a) (b)

Figure 2.22. Quantitative Results. (a) Performance. (b) Times.

2.8 Application To Real Images

2.8.1 The Sowerby Image Data

The Sowerby Image Database (SID) consists of a collection of images in JPEG and HIPS formats as well as various data files. The image files of principle interest here are the RGB calibrated images depicting real-world scenes and the images of segmentation interpretations that assign labels defining the type of various image regions.

The RGB calibrated images are in 24-bit colour. They began life as 35mm slides before being digitised and recalibrated to remove image acquisition errors. These images were then segmented by an unspecified method and the resulting segments were labelled by hand.

There are a total of 54 base-level labels for image regions but these are grouped in a hierarchical manner to give just the 11 parent labels listed below.

1. Atmospheric phenomenon (sky, cloud etc.)
2. Bounding object (fence, gate etc.)
3. Building
4. Electrical object (pylon, wire etc.)
5. Illumination shadow
6. Landscape (soil, water, vegetation etc.)
7. Mobile object (bike, car etc.)
8. Road surface (unmarked areas)
9. Road marking (white lines)
10. Road border (gravel, mud etc.)
11. Road sign

It is these parent labels that are used to form the images of segmentation interpretations. These too are colour images with a different colour being assigned to each label and undefined regions arising from imperfect segmentation being left black.

The RGB calibrated images and images of segmentation interpretation were each available in both the original HIPS format and a slightly degraded JPEG format. Unfortunately, Matlab does not recognise the HIPS file format so that we forced to use the degraded JPEGs. The JPEG versions of the RGB calibrated images are subsequently referred to as the input images while the JPEG images of segmentation interpretation are referred to as the labelled images.

2.8.2 Data Selection and Sampling

The initial intention was to sample for each class from the entire database of images, create class-conditional models using the sample data and use these models to parameterise our multi-channel PCNNs. However, closer inspection of the correspondence between the input images and the labelled images suggested that this would be ineffectual due to the nature of the labelling.

We can see the problems we are faced with by looking at the 'landscape' class. Our multi-channel PCNN models are colour/intensity segmentation algorithms and will ultimately be incapable of treating 'landscape' as a single unified class when it includes regions as disparate

as a ploughed field and a pond. Similar problems exist for most of the parent labels employed and as a result, more limited experiments were proposed where sampling would be from carefully selected training images and testing would occur on the training images themselves and similar test images.

2.8.2.1 Selection Criteria

Sixteen of the images within the database are specified as test images and it was felt that our input images ought to be selected from amongst these. These test images were drawn from three different films: 11, 12 and 13 and the eight images from film 13 were selected for experimental purposes because here the classes seemed to be most uniform. Two of these eight images feature dirt tracks while the remainder feature tarmac surfaces so the group was initially separated along these lines. The four images within the tarmac group found to be least similar in the classes they featured were then discarded leaving us with four images in two groups.

Within each group one image was designated for sampling purposes while the other was left purely for testing. Figures 2.23 and 2.24 display the input images from each group on the left hand side with the corresponding labelled images on the right. In each case the sampling images appear above the test images.

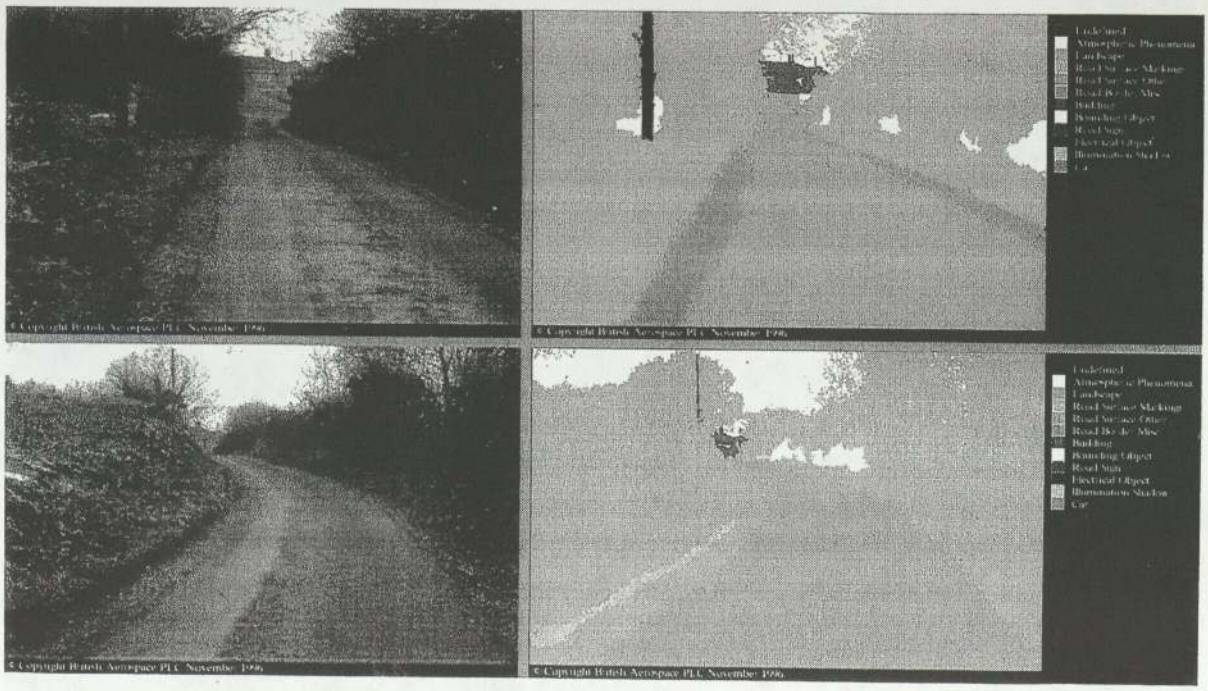


Figure 2.23. The tarmac images. Clockwise from top left: sampling input image, labelled sampling Image, labelled test image, test input image.

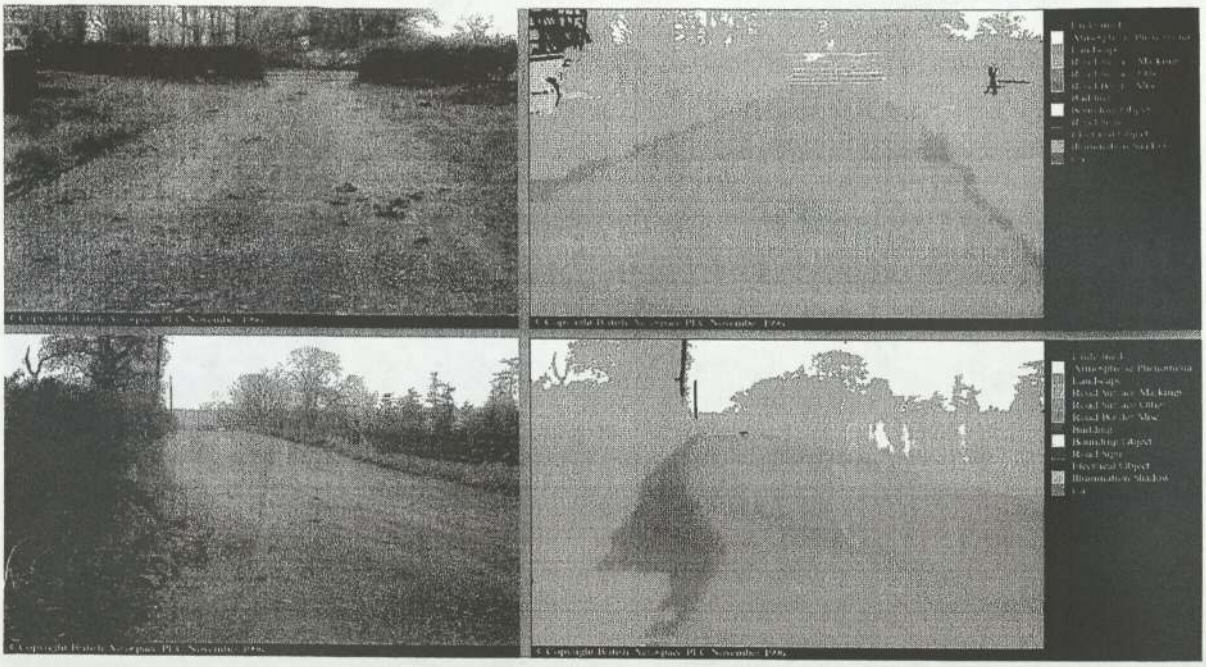


Figure 2.24. The dirt-track images. Clockwise from top left: sampling input image, labelled sampling Image, labelled test image, test input image.

2.8.2.2 Sampling

The first step in sampling from the images selected was to form greyscale template matrices as used on the synthetic images in the sections above. These were to be created using the labelled sampling images and used to select regions of the input images from which to sample for each class. Unfortunately, the extraction of our templates was subject to error due to the degradation involved in the JPEG encoding of the labelled images. It was found that thresholding was unable to perfectly separate the labelled regions and while using a PCNN for this task might have been more effective, an objective measure of segmentation performance here was unavailable.

Rather than confuse matters by employing a PCNN in the sampling process used to parameterise another PCNN, it was felt that thresholding, while error-prone, was the more sensible option here. The templates were still used as region selectors during the sampling process and the errors here were absorbed as an additional source of noise for our methods to contend with.

Mean and variance values were calculated for each channel of each class present in the sampling images and the classes were rearranged in descending order of total mean intensity. Appropriate sequences of S_{mids}, T_{vals}, dTs were then calculated for all our methods before individual beta values were assigned to each method according to the prescriptions developed in section 2.7 above.

2.8.3 Results

2.8.3.1 Qualitative Results

Figures 2.25 to 2.28 show the segmentations achieved by each method on the tarmac sampling image, tarmac test image, dirt-track sampling image and dirt-track test image respectively. Lambda values and kernels were identical to those employed in the varying variance experiment in 2.7.6.

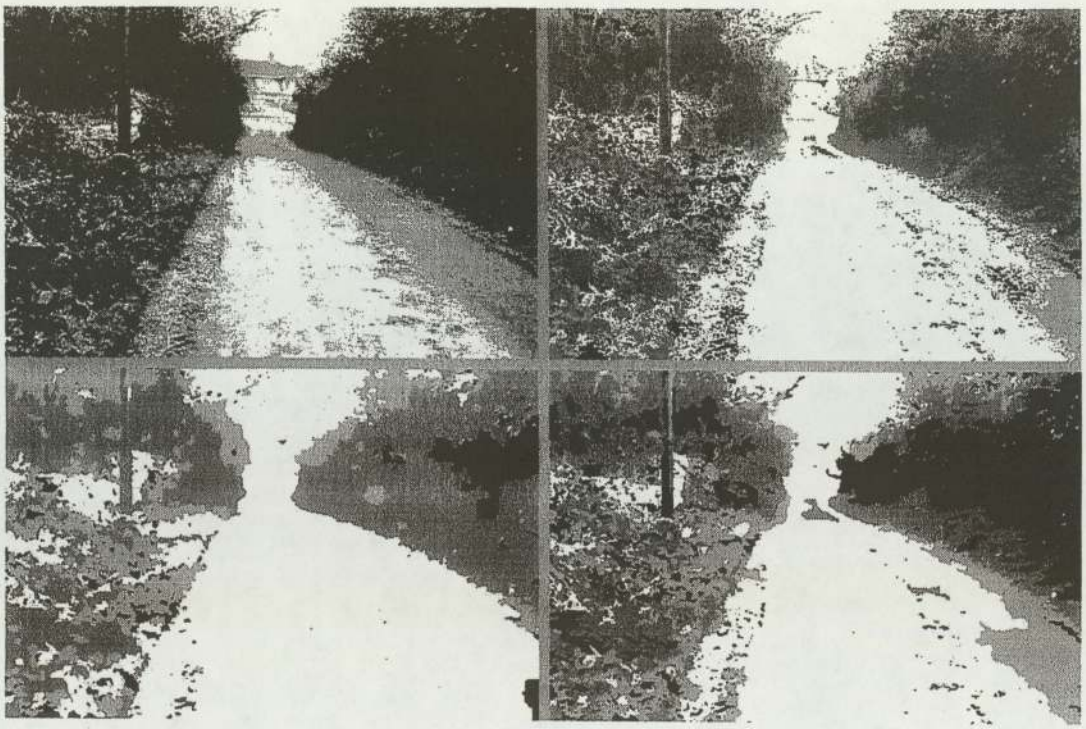


Figure 2.25. Tarmac sampling image segmentations. Clockwise from top left: simple dynamic thresholding, the safe method, the intensity-coded method, the augmented method.

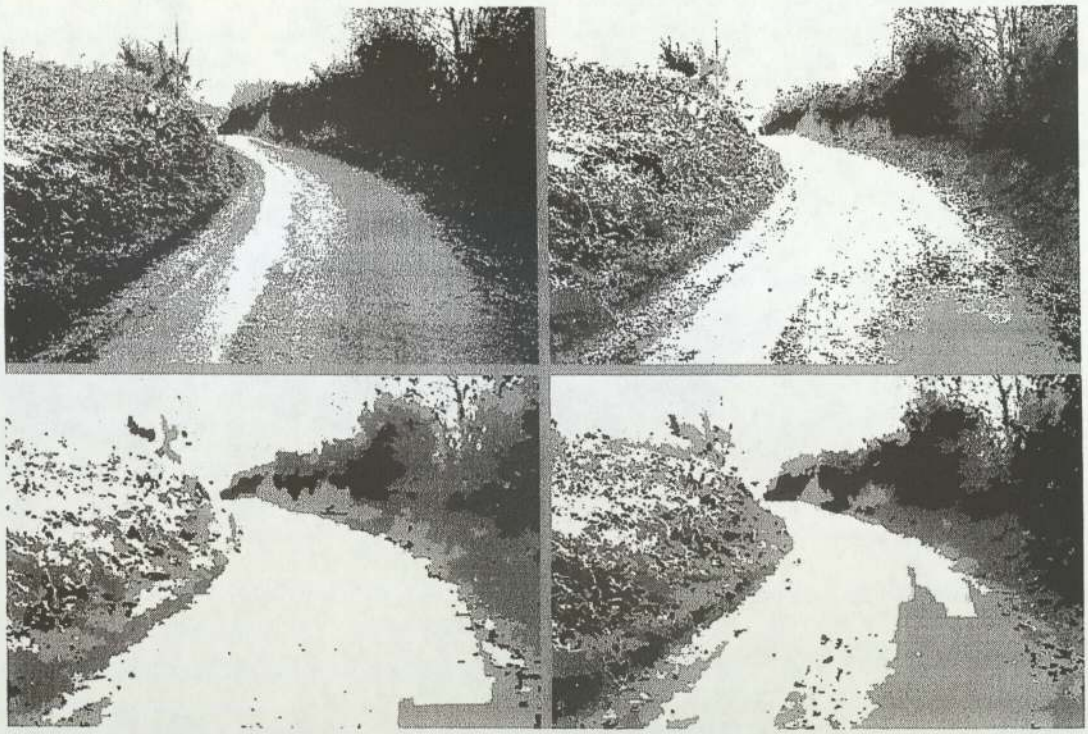


Figure 2.26. Tarmac test image segmentations. Clockwise from top left: simple dynamic thresholding, the safe method, the intensity-coded method, the augmented method.

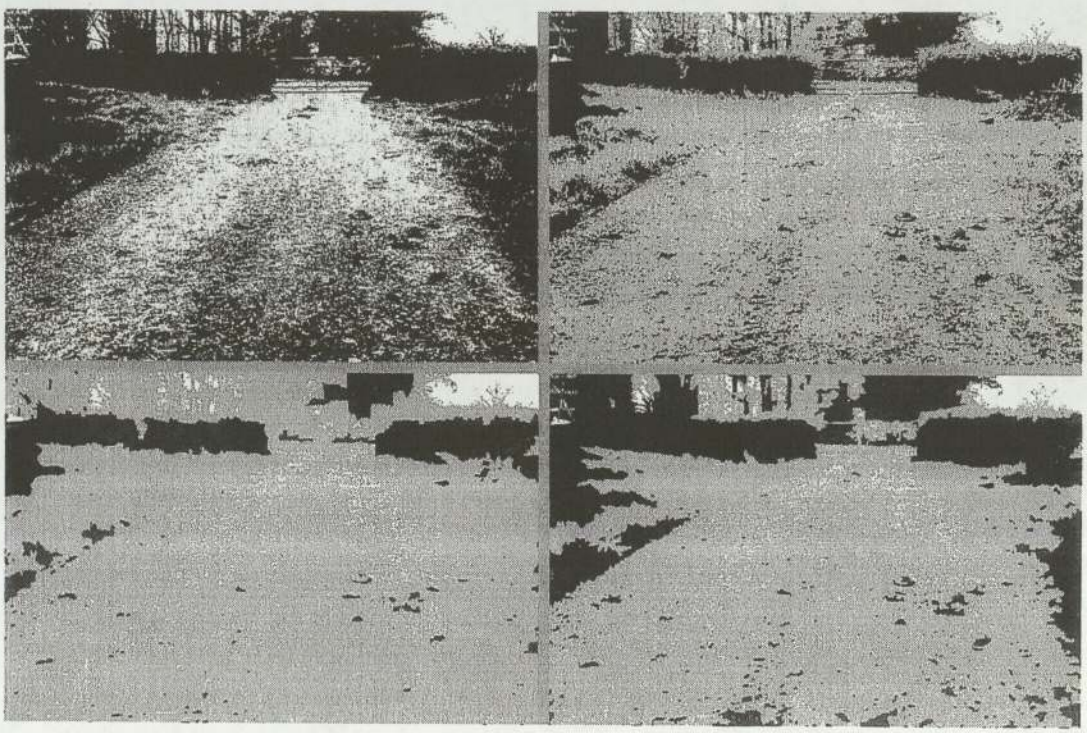


Figure 2.27. Dirt-track sampling image segmentations. Clockwise from top left: simple dynamic thresholding, the safe method, the intensity-coded method, the augmented method.

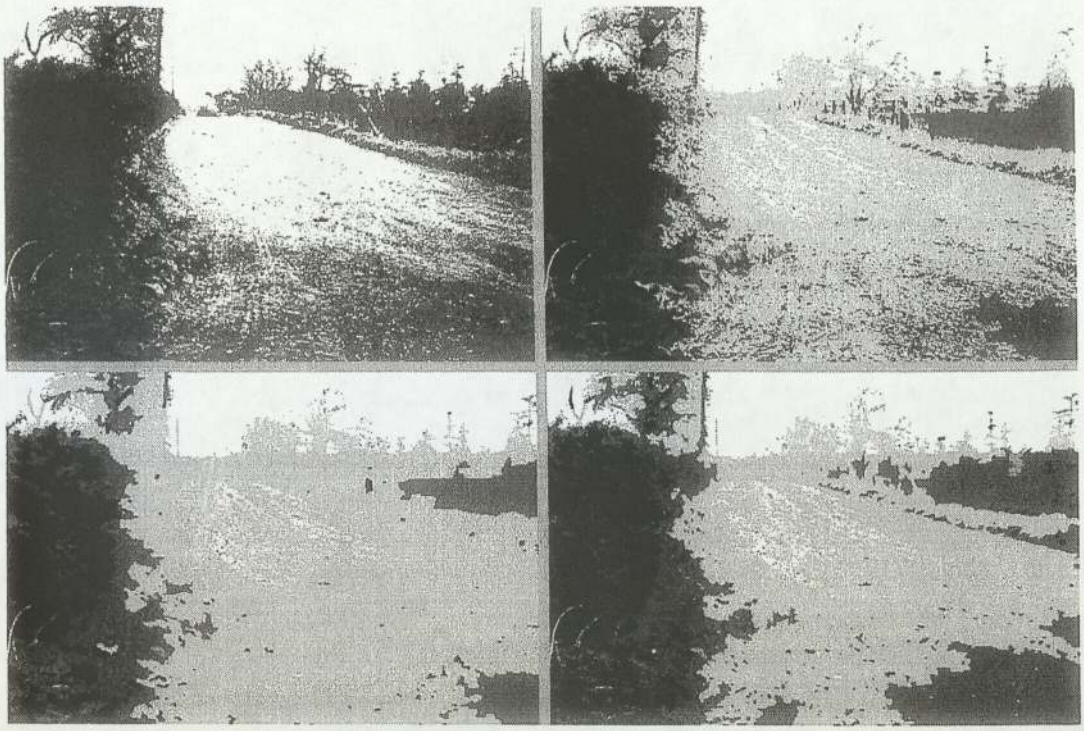


Figure 2.28. Dirt-track test image segmentations. Clockwise from top left: simple dynamic thresholding, the safe method, the intensity-coded method, the augmented method.

2.8.3.2 Quantitative Results

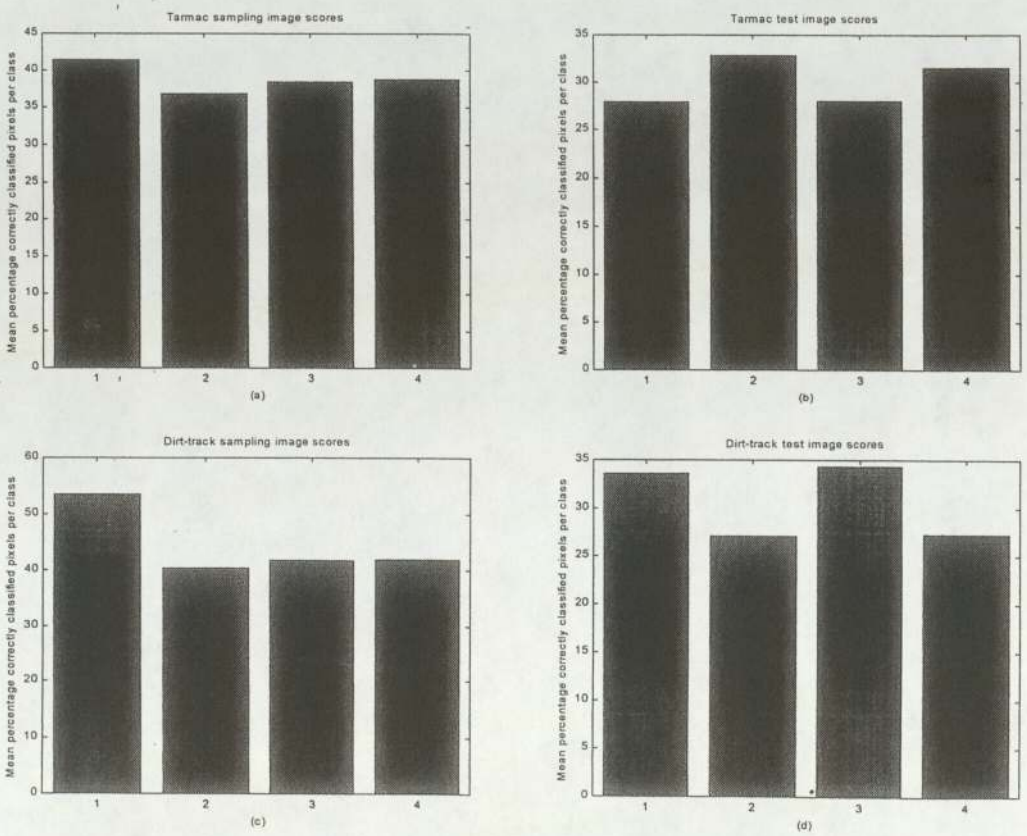


Figure 2.29. Mean correctly classified pixels per class. (a) Tarmac sampling image, (b) Tarmac test image, (c) Dirt-track sampling image, (d) Dirt-track test image.

3 Analysis and Discussion

The development of effective parameterisation methods for the pulse coupled neural network family is an important step in the continuing process of legitimising their use in image processing. Once available, such methods should remove the need for expert, task-specific intervention in the operation of these models and enable their integration into truly automated and versatile image processing systems.

Here we have pursued a number of promising avenues in this developmental process beginning with an exploration of methods designed to achieve perfect image segmentation under limited conditions before moving onto alternative methods designed to be of more general utility under a wider range of conditions.

3.1 Perfect Segmentation Methods

The perfect segmentation method proposed by Kuntimad and Ranganath helps to justify the claim that effective parameterisation methods for the PCNN are within reach and ought therefore to be an active area of research and development. The method is founded in a principled manner on measured or otherwise known characteristics of the image regions we wish to segment and can thus be seen as data-driven and task-specific, thus avoiding the overly knowledge-driven approach prevalent in PCNN research. The method is also capable of optimal performance within certain constraints and we felt that in many ways it formed a good model for effective parameterisation methods. It was therefore quite influential in shaping the methods we would subsequently develop.

However, the method was critically limited by the degree of intensity range overlap between adjacent regions it could overcome as well as by its reliance on specific border geometry conditions between the image regions to be separated from one another. The alternative perfect segmentation method we developed in section 2.1.8 shows us that if we rely on this latter condition then the former can be largely overcome. However, it also suggests that a reliance on border geometry conditions is a weakness that ought to be avoided. Even within the specified restrictions we found that we could falsify K&Rs claim that their method could guarantee perfect segmentation of an image.

Initially these findings might seem to cast into doubt the whole parameterisation development process. However, the method is still effective in its task given fairly minor restrictions on the

spatial and intensity distributions present and we felt that more general methods might still be produced. Kuntimad and Ranganath's suggested that future work should focus on the determination of optimal values for certain parameters based on the image probability density function and this was particularly influential in determining the form of the methods we developed here.

3.2 Other Parameterisation Methods

3.2.1 Simple Dynamic Thresholding

Simple dynamic thresholding is obviously not one of our PCNN parameterisation methods but it nevertheless acted as a way-marker in the developmental process. A moving, global threshold here performs temporal segmentation, and the optimisation of the threshold values is based on a simple error minimisation criterion applied to the image p.d.f. and operating entirely outside the spatial domain.

By adopting a probability-based form, the method avoided the pitfalls of a reliance on border geometry conditions but expectations were not high for the performance of the method since all local spatial context was also abandoned. Both qualitatively in Figure 2.11 and quantitatively in Figure 2.17, we found that our expectations were borne out, with performance being poor in both cases. Segmentation errors were common and occurred at an approximately linearly increasing rate with increasing Gaussian variance in the quantitative test. However, the errors occurred here for very obvious reasons and it was felt that the addition of limited spatial context in developing the PCNN parameterisation methods would enable us to address the sources of error systematically.

3.2.2 The Safe Method

The safe method was based on a relatively simple PCNN model designed to supply minimal spatial context information while still being compatible with a probability-based parameterisation method. By employing stepped linking in this model, it was found that we could effectively apply hard threshold values to the intensity range over which pulse capture could occur. This suggested a parameterisation method that could take advantage of the same decision boundary values as before but here, the segmentation of an image region would

occur by the spread of a wave of pulse-capture within a fast-linking loop and as such would be influenced by local spatial context not available to the thresholding method.

Two distinct forms of error were identified in the simple dynamic thresholding method's segmentation of regions of overlapping intensity ranges. These were dubbed low-tail and high-tail errors respectively and it was found that the safe method was capable of suppressing high-tail but not low-tail errors. From a qualitative perspective, the resultant partial improvement in performance is obvious from a brief comparison of Figures 2.11 and 2.14. The segmentation in the latter is much smoother than the former while still featuring well-preserved edges but overall it is still fairly rough. Likewise looking at the quantitative comparisons in Figure 2.17, we see a significant improvement over simple dynamic thresholding maintained over a range of variance values. Again, performance appears to decrease roughly linearly with linear increases to the variance but the rate of decrease is lower here. Testing processing time for this model in Figure 2.17(b) suggested that the computational cost of the algorithm increased approximately linearly with increasing class overlap from an initial figure close to that of the simple dynamic thresholding method.

Overall, the safe method was a promising development from the simple dynamic thresholding method. However, the improvement in segmentation performance was limited by the method's inability to address low-tail errors so that this became the focus of the next development effort.

3.2.3 The Augmented Method

Where the spatial context available to the safe method was limited to a yes-or-no signal as to whether pulsing behaviour was occurring in a given neighbourhood, the augmented model added quantitative information regarding the number of pulsing neurons within the neighbourhood as well as their position. The modification to the model employed, resulted in a soft threshold on the pulse capture intensity range with the range being affected by the quantitative information available.

By introducing a flexible pulse capture range, the augmented method was able to address low-tail as well as high-tail errors and was consequently capable of significantly better segmentation performance than the safe method. Figure 2.15 qualitatively displays this improvement in the form of a visibly much smoother segmentation than we saw previously and edges are still excellently well preserved. In Figure 2.17 we see a correspondingly large

quantitative improvement in performance with processing time close to that of the safe method and in Figure 2.18 we found that this method measured up well against the perfect segmentation methods: degrading gracefully in performance at a fairly gentle rate and being relatively cheap computationally.

The drawback of the augmented method is that by making the pulse capture range flexible we can no longer directly derive optimal linking coefficients using the minimum error decision boundary values calculated from the image p.d.f. The decision boundaries still play a central role in the parameterisation but here they become a component of a heuristic prescription for beta, which is flexible by virtue of the lambda coefficient.

3.2.4 The Intensity-Coded Method

The intensity-coded method employed a model allowing for still further spatial context information to be communicated. As well as a quantitative measure of neighbourhood pulsing activity, here the intensity values of the corresponding image pixels was also included in the context passed between neurons.

It was hoped that this additional information would enable the intensity-coded method to further reduce the error still present in the segmentations produced by the augmented method. Unfortunately this was not found to be the case and we must conclude that the transmission of intensity values are of no value to this form of model. Processing time was similar to the previous methods.

3.2.5 Multi-Channel Extensions

The method chosen here for extending the parameterisation methods into multi-channel form was very simple and yet surprisingly effective. The models employed were modified to perform a logical AND operation over the input channels to produce a single-channel output as before. This treatment was selected because there seemed to be a simple way of adjusting each of our parameterisation methods to account for this alteration.

The more traditional way of performing image fusion with the PCNN is to operate individual PCNNs on each channel but to modify the models so as to perform inter-channel as well as intra-channel linking modulation. This approach has been applied to the fusion of RGB colour

images [10] and has also been used to fuse different filtered feature maps of greyscale images [13]: the authors here claiming that performance is better than using a logical AND over the filtered images. However, an appropriate parameterisation method for this technique is currently less clear than for the logical AND scheme so the models have been overlooked here.

In experiments, it was found that with the addition of the logical AND operation, we could segment colour images that would have been impossible to segment by the single-channel methods. Comparing Figure 2.22 with Figure 2.17 we find also that segmentation performance of the multi-channel methods was comparable with that of the single channel methods on Gaussian classes of the same variance. The safe method no longer performed significantly better than simple dynamic thresholding but this was due to an improvement in the thresholding method's performance rather than a failure of the safe method.

Processing times for the multi-channel methods were approximately five times that of their single-channel counterparts and this is a major drawback of the method. However, more efficient software implementations are in development and a number of parallel hardware implementations for PCNNs have been proposed [14] – [16] so this is not necessarily such a great problem.

3.3 Application to Real Images

The three multi-channel PCNN parameterisation methods developed here were applied to a limited selection of RGB colour images from the Sowerby Image Database (SID). Once again, the simple dynamic thresholding method was used for comparison and if we look at Figures 2.25 to 2.28 we can see that the results were less than inspiring.

In general, the smoothness of the segmentation was ordered as follows:

simple dynamic thresholding < safe < intensity-coded < augmented.

The sky was consistently well separated from neighbouring regions by each method but this is a trivial problem and the result cannot be interpreted as offering support to the use of the PCNN in the role of segmenter. In the tarmac images, the road regions were relatively well segmented by the augmented method: they were separated from the surrounding landscape regions and the region itself was smooth. However, large regions of building and landscape

that should have been separated from the roads were segmented at the same time step and by the classification scheme would have been classed as road. This problem was even more pronounced in the dirt-track images, where no clear separation between road and landscape was achieved by any of the PCNN methods. Other classes such as bounding object and electrical object were even less well dealt with and in general we can say that all the methods failed in the segmentation task presented to them.

Looking at the quantitative results confirms the failure of each method employed. Figure 2.29 displays the performance of each method on each image, this performance being defined as the mean number of correctly classified pixels per class. As should be expected, each method performed better on the sampling images than the corresponding test images but a score above 50% was registered only once. On average, each PCNN method performed worse than the simple dynamic thresholding method and this can only be viewed as a failure of the PCNN.

3.3.1 Reasons for the Failure

In hindsight there are a number of good reasons as to why our models performed poorly on the segmentation task we set them here and we argue that the poor performance should not be interpreted as a general weakness of the PCNN approach to image segmentation but rather as a mismatch between problem and model form.

The images in the SID were initially hand-labelled according to high-level semantic definitions like 'electrical object' or 'landscape' rather than being labelled according to the low-level image characteristics of colour and intensity upon which the PCNN operates. As a result, most class intensity distributions were far from the Gaussian form assumed in the parameterisation methods. Indeed, the true distributions were almost certainly multi-modal in nature and the components of the distribution appearing in a given image would differ considerably. For example, the dirt-track sampling image featured landscape areas composed entirely of green regions while the test image featured a brown plough fields still classed as landscape in the labelled image.

Other problems include the apparently arbitrary labelling of road border regions and shadows where there was really no hope of separating these regions from their neighbours. For example, no road border regions are labelled in the tarmac test image when areas appear to be visible while in the dirt-track sampling image, road border regions are labelled to the right of the dirt track which are virtually impossible to discern in the input image.

We do not feel that the sampling problems from the JPEGs was a major determining factor in the failure of our methods. Outliers within classes are largely ignored by the parameterisation process because T_{vals} are set according to total class variance and accept that some of the 'lower' intensity class may lie above the threshold.

In summary, the PCNN is a low-level segmentation algorithm working on very primitive image features and as such was ill-suited to segmenting the SID images. The poor performance therefore stemmed from a misapplication of the model rather than ineffective parameterisation methods. To achieve automated labelling of the classes in the SID would require higher-level object recognition algorithms and although we feel that PCNNs could still play a valuable role as a pre-processing component for the recognition system, different parameterisation schemes would be required here, of a form as yet undetermined.

3.4 Towards a Probabilistic Analysis

Throughout the development process detailed above, we attempted to approach problems in a principled manner but were forced ultimately to adopt heuristic prescriptions for the key model parameters. While each of these could be justified logically as an incremental modification of a previous method, this led to the sort of isolated, relativistic analysis that does little to place the PCNN in a wider context.

We wanted to understand in a more formal fashion exactly why each of the models and methods performed as they did and this reasoning led us to attempt to place the PCNN within a statistical framework. If this could be achieved, it was envisaged that the following principal benefits would become available for the whole PCNN family:

1. Probabilistic analyses of model behaviour

2. Deeper comparisons with alternative segmentation methods

The problem facing us was that the dynamical nature of the PCNN makes a complete description in statistical terms complex. As a result, to the author's knowledge, no such description has yet been attained.

Consideration has been limited here to simplified, fast-linking models of the form used in the safe, augmented and intensity-coded parameterisation methods presented here. We also limit

ourselves to the single-channel case although it is hoped that any solution might subsequently be extended to a wider range of models.

Our approach was to study network behaviour within the crucial fast-linking loop since calculations external to this are not dynamically complex. It has been observed that the pattern of pulsing activity within the fast-linking loop of all the models used above, always progresses to a fixed point. We want to understand both the evolution from start point to the eventual fixed point and the fixed point itself.

3.4.1 An Energy Function Formalism

Energy functions are used in physics to describe the energy states of real physical systems. In the neural network literature, the term is used by analogy to refer to similar equations used to describe the state of the network or components thereof. The introduction of energy functions has been found to be of particular benefit in describing and analysing the behaviour of recurrent neural networks, where feedback loops exist in the connection pathway. Most famously, Hopfield [17] employed an energy-based formalism to prove the convergence of a class of recurrent network now commonly referred to as Hopfield Nets. This convergence proof relied upon the energy defined for the network always either decreasing or remaining constant and only being able to stay constant for a finite number of steps of the algorithm.

Energy functions have also been employed in the development of Bayesian approaches to image analysis [18] and it was hoped that a similar formalism for the fast-linking PCNN might not only benefit our understanding of network behaviour but also lead towards the desired statistical interpretation. We begin by running through a brief comparison of the PCNN and Hopfield models before using this to motivate the form of our energy function.

3.4.1.1 PCNN and Hopfield Models

The PCNN and Hopfield networks feature numerous similarities, some of which are listed below:

1. Both are dynamic and recurrent in nature

2. The state of either model is best expressed as the pattern of firing activity across the network. Due to differences in network topology, this is expressed as a state vector in the case of the Hopfield net and the output matrix **Y** in the case of the PCNN.

3. This pattern of firing activity displays convergent behaviour.

4. The weights on connections between neurons are symmetrical in both models.

These similarities helped to motivate the energy-based approach to PCNN analysis but the form of the energy function developed owed more to the differences between the models, some of the more important of which are as follows:

1. Hopfield nets are globally connected: each neuron is connected to every other neuron across the entire network. PCNNs are only locally connected over an area determined by the linking kernel **W**.

2. Incoming weights to each neuron in the PCNN are identical, being defined by **W**. This is not true of the Hopfield net.

3. The Hopfield net features no external inputs. It simply evolves from a starting point given its internal weight structure. In contrast, each neuron of the PCNN has an external feeding input that affects its firing behaviour.

4. The standard Hopfield net features asynchronous update where neurons are updated individually. In contrast, PCNNs employ synchronous update where the firing status of all neurons is updated simultaneously.

5. Hopfield nets feature both negatively and positively weighted connection while the PCNN usually only features positive connections. An exception to this rule would be the model with additional inhibitory linking fields introduced by Kuntimad and Ranganath and commented on above.

The energy function for the Hopfield net is defined in a per-connection fashion with the energy for the whole network being the sum of these connection energies. Specifically, for neurons i,j the inter-neuron energy is defined by (37) where y_i and y_j are the output values of the two neurons.

$$e_{ij} = -w_{ij} y_i y_j \tag{37}$$

However, because of local, identically weighted connectivity and the existence of feeding inputs, it was felt that the energy function of the PCNN would be better expressed in a per-neuron fashion. The role of the feeding inputs in particular would be difficult to express via per-connection energies, while the connectivity arrangement in the PCNN makes it more natural to express interactive influence as an energy on each of the neurons involved than it would have been in the Hopfield model.

3.4.1.2 The Energy Function Proposed

For convergence purposes, we want to produce an energy function that will monotonically decrease in value throughout the fast-linking process. We note that at time step $[n]$, once a given neuron is pulsing, $Y_{ij}[n] = 1$, the neuron will remain in the pulsing state throughout that time step. This follows from the fact that there are no negative linking values and it is because K&R's inhibitory modification violates this property that we suggested its analysis would be more complex than a more standard model. This stay-on property alone ensures convergence of the algorithm since in the extreme case the network will progress to an all-pulsing state and otherwise will halt somewhere on the path to that state.

However, we want the energy function to say something meaningful about how far along the path to the all-pulsing state the network will travel. In order to do this, we needed it to reflect calculations as to whether individual neurons will pulse at this time step or not. The energy function for individual neurons was therefore designed to satisfy the following conditions:

1. It must remain constant while the neuron is in the non-pulsing state

2. It must decrease when the neuron moves from the non-pulsing to pulsing state

3. It must either further decrease or remain constant while in the pulsing state

By looking once again at the equation for internal activation (38), we can see exactly how this was achieved. The pulsing of a neuron at time step n corresponds to the condition $U_{ij}[n] > T_{ij}[n]$. We can therefore rearrange (38) to form (39) giving the conditions for pulsing at time step n.

$$U_{ij}[n] = S_{ij}\{1 + \beta[n]L_{ij}[n]\} \tag{38}$$

$$\left(\frac{T_{ij}[n]}{S_{ij}} - 1\right) - \beta[n]L_{ij}[n] < 0 \tag{39}$$

We will base our energy function around the LHS of (39) which takes on a value less than zero when $Y_{ij}[n] = 1$, and a value greater than or equal to zero when $Y_{ij}[n] = 0$. To satisfy condition one however, we need our energy function to be constant when $Y_{ij}[n] = 0$ rather than having a range of values. This is achieved by multiplying both terms by $Y_{ij}[n]$ so that when $Y_{ij}[n] = 0$, the whole equation equals zero but when $Y_{ij}[n] = 1$, the equation takes on the same value as before. Applying this modification and disregarding time indices as we are only considering behaviour within a single time step, we arrive at the following energy function for neuron $_{ij}$:

$$e_{ij} = Y_{ij}\left(\frac{T_{ij}}{S_{ij}} - 1\right) - Y_{ij}\beta L_{ij} \tag{40}$$

This satisfies condition one by taking on a value of zero if $Y_{ij} = 0$. Condition two is also satisfied because by (39), e_{ij} will always be less than zero if $Y_{ij} = 1$. In considering condition three we note that the only variable here is the linking value L_{ij}, which will never decrease during the fast linking process. If L_{ij} remains constant then so too does e_{ij} while if L_{ij} increases and $Y_{ij} = 1$, e_{ij} will decrease and so the third condition is satisfied.

The Energy function for the entire network is simply the sum of the energies defined for each neuron.

$$E = \sum_{ij} e_{ij} \tag{41}$$

3.4.1.3 Convergence Proof

There were initial fears that the convergence proof would be complicated by the synchronous dynamics of the PCNN but given the proposed form for our per-neuron energy function, the proof becomes very straightforward. We want to show that the network will necessarily progress from its initial state to a fixed state corresponding to an energy minimum. In demonstrating this, we note that any possible change in the state of the network Y, is the sum

of possible changes to the state of individual neurons. It is therefore sufficient to look at an arbitrary neuron $_{ij,}$ and demonstrate that any change in its state Y_{ij}, would result in the network energy decreasing and that the energy is bounded below by some value.

We recall that, by positive linking, Y_{ij} can only possibly change from zero to one and by satisfying condition two above, we have shown that the energy for the neuron in question is guaranteed to fall in this case. However, a change in Y_{ij} from zero to one can also affect the energy of the neuron's neighbours by altering the value of the linking input. By (40) we know that this will only occur if the neighbour is already in a pulsing state and we know also that in this case the result will be a reduction in the value of e_{kl}. Thus e_{ij} is guaranteed to fall if Y_{ij} changes value and all neighbourhood energies are guaranteed to either remain the same or fall. This guarantees a reduction in total network energy and hence guarantees that any possible change in **Y** does likewise.

Network energy has a lower bound determined by the kernel selected, the value of the linking coefficient beta and the ratio of pixel intensity to threshold value across the whole network. We have shown above that any possible changes in network state reduce network energy but by the direct correspondence between network equations and the energy function, we know also that if a change in state that would reduce E were currently available then the network would adopt the lower energy state. Indeed, movement on the energy landscape is of the steepest-descent variety because each neuron in the network will always adopt the lowest energy state available during a given loop. Energy will continue this steepest descent process until no available change in state would result in a further reduction. Thus the network will descend to a local energy minimum but the question remains as to whether or not this is a global minimum.

We define the global minimum as the minimum possible network energy given the constants **T**, **S**, **W** and β. Since **T** and **S** determine the initial state of the network and because the behaviour subsequent to that is deterministic then there is only a single energy minimum, which is necessarily global. In the augmented scheme this makes sense since substituting for L_{ij} shows that the energy function is quadratic in **Y**.

$$e_{ij} = Y_{ij}\left(\frac{T_{ij}}{S_{ij}} - 1\right) - \beta \sum_{kl} W_{ijkl} Y_{ij} Y_{kl} \qquad (42)$$

3.4.2 A Bayesian Interpretation

For a Bayesian interpretation of our energy function, we consider the following properties:

1. The first term is data-dependent while the second is data-independent

2. For the general case where $Y_{ij} = 1$, $S_{ij} \leq T_{ij}$, the first term takes on a value greater than or equal to zero of a magnitude dependent on the ratio of S_{ij} to T_{ij} while the second term takes on a value less than or equal to zero whose magnitude depends on the weighted sum of local neighbourhood pulsing activity. Thus the two terms operate antagonistically.

The case stated in 2. is general because if $Y_{ij} = 0$, $e_{ij} = 0$ and if $S_{ij} > T_{ij}$, the first term becomes negative as does e_{ij}, and Y_{ij} is thus initially set to one. These alternatives are therefore special cases while the case listed is that which determines network behaviour within the fast linking loop. If the magnitude of the second term is larger than that of the first then pulsing will occur, otherwise it will not.

The proposal is that the first term is related to a likelihood on the network state while the second term is related to a prior. To make the proposal more concrete let us specify the following new variables:

$$E_1 = \sum_{ij} Y_{ij} \left(\frac{T_{ij}}{S_{ij}} - 1 \right) \tag{43}$$

$$E_2 = -\sum_{ij} Y_{ij} \beta L_{ij} \tag{44}$$

We then define the likelihood and prior terms as follows:

$$p(\mathbf{S} \mid \mathbf{Y}) = \exp\{-c_1 E_1\} \tag{45}$$

$$P(\mathbf{Y}) = \exp\{-c_2 E_2\} \tag{46}$$

Using Bayes' theorem (47), we can write (48) for the posterior.

$$P(\mathbf{Y} \mid \mathbf{S}) = \frac{p(\mathbf{S} \mid \mathbf{Y})P(\mathbf{Y})}{p(\mathbf{S})} \tag{47}$$

$$P(\mathbf{Y} \mid \mathbf{S}) \propto \exp\{-c_1 E_1\}\exp\{-c_2 E_2\} \tag{48}$$

To justify this interpretation, we note first that the state of the network **Y** at time step n corresponds to a classification where values of one mean the pixel has been classified as belonging to the class under consideration and values of zero mean the pixel has been classified as belonging to some other class. Thus **Y**[n] is directly equivalent to C_n. The exponential form for the prior and likelihood terms follows from information theory and the principle of maximum entropy [19]. This shows that when you look at the distribution with a given average energy that minimizes the entropy you arrive at an exponential expression for the probability terms.

4 Conclusions and Future Work

In this research project we set out to develop effective parameterisation methods for the PCNN family and in section 1.4 we made an attempt to qualify what exactly we meant by effective. The following four characteristics were suggested:

1. They should be fully automated.

2. They should be flexible and yet powerful.

3. They ought to produce parameter sets that are in some sense optimal for the application under consideration.

4. They should improve the performance and reliability of the PCNN as an image processing tool.

To conclude our work, we now judge how well each of the methods featured here measures up to these characteristics.

The perfect segmentation methods were fully automated and powerful in their potential performance but inflexible by virtue of the limited conditions for their operation. They produced optimal parameters within the constraints placed on border geometry and intensity range overlap and could certainly be said to improve the performance of the PCNN while operating within these limits, but we don't feel that they could be said to be reliable. They are simply too brittle and constrained for that.

The safe method was also fully automated provided the kernel was fixed and the threshold values were set at fixed points defined by the class-conditionals models employed. We found that this was possible and could be effective across a range of problems. The safe method is not particularly powerful as an image segmenter but against this we can weigh its great flexibility. All it needs is a reasonable set of class-conditional models for the distributions present in the image. While Gaussians were assumed here, this need not be the case in general and once the class-conditionals are present, statistical theory will provide us with the minimum error decision boundaries we require. Given the threshold values, kernel and the particular model form selected, and given also an assumption of smooth spatial distributions within the image, we believe that the safe method can be considered an optimal way of setting

the parameters. The reason this optimal solution does not look very optimal is due to the limits of the model itself. Overall, we do not feel that the safe method can be viewed as improving the performance of the PCNN as compared with alternatives but it does offer reliable results over a wide range of problems.

The augmented method is only fully automated if the kernel and thresholds are not user-adjusted and the lambda coefficient is also fixed. The method did sometimes benefit from minor adjustments to the lambda coefficient but these are far simpler to handle than alterations to the original model parameters. We class this model both as a powerful and flexible in an image segmentation role thus improving on either of the previous methods above, but while performance was excellent, we saw no sense in which the parameters could be said to be optimal here. Overall, the augmented model definitely seems to offer improved performance and reliability over more traditional methods for the treatment of the PCNN.

The intensity-coded method measures up to our effectiveness criteria in almost identical manner to the augmented method. It performed very similarly in all experiments, causing us to suggest that the modification to the model employed had no overall effect on potential performance.

The multi-channel extensions of our methods produced results that largely mirrored those of the single-channel experiments. Most of the comments above are therefore equally applicable here but there are notable exceptions. Automation was slightly reduced for each method as it seemed more difficult to produce consistently effective parameter sets. As well as this, processing time was considerably increased though we feel this might be overcome by hardware implementations.

The application of our methods to the real images of the Sowerby Image Database proved to be largely unsuccessful but we feel there were generally good reasons why the PCNN as a class of model was unsuited to the task presented to it. As a result, we cannot blame the parameterisation methods themselves.

The energy function approach to characterising PCNN behaviour was introduced to begin to place these models on a firmer foundation. Closely matching the central equation of the algorithms themselves, the energy function lays bare the antagonistic nature of the main model components within the fast-linking loop. It can be found that the any of the model forms featured will descend to a global energy minimum for that model, given the value of

certain constants but as yet this is of little use to us. What we need to be able to do is compare the minima for each model in an objective fashion and also the different minima achieved by the same model with different parameter values. Ideally we would be able to define the parameter values and model form that would give us an absolute global minimum energy and we would hope that these would correspond to the optimal model and parameters for the task at hand. A Bayesian/entropic interpretation of the energy function was tentatively proposed and we hope that this might also prove beneficial in the principled design of future parameterisation methods.

Until then though we have to make do with our current parameterisation methods and on balance, we feel that the augmented and intensity-coded methods are the most promising considered here. They exhibit a flexibility lacking from the perfect segmentation methods while still offering high levels of performance in an image segmentation role. By Occam's Razor, we would recommend the augmented method because of the simpler model used.

4.1 Future Work

Our current work has suggested a number of key areas that will be the focus of future research:

1. The further investigation and development of the energy function method for PCNN analysis along with its Bayesian interpretation - We hope that this could yield truly optimal, general parameterisation methods and would help to establish the place of the PCNN within the image processing field.

2. The detailed comparison of the PCNN with alternative low-level image processing algorithms - We believe that a statistical framework could prove to be of particular benefit in comparative studies.

3. The development of parameterisation methods for the fusion PCNN suggested in [10] and [13] - Again, we hope that a probabilistic framework could be of use here.

4. The investigation of multi-layer models and the development of parameterisation methods for them.

5. The introduction of inhibitory connections and the investigation of their influence on the behaviour of our models.

6. The introduction of alternative feeding receptive field types - Current work has focussed on direct feeding, where the receptive field is a highly local measure of pixel intensity but the linking modulation within the PCNN is a general mechanism for grouping neurons according to similarities in their feeding input and if the characteristics of the feeding receptive field change then so too does the type of data we group together in the image. Alternative receptive field types might include physiologically motivated fields of a centre-surround form for example but there is no reason to restrict our models in this area. Feeding values might be local variances or the result of any number of filtering operations. The possibilities are endless.

References

[1] R. Eckhorn, R. Bauer, W. Jordan, M. Brosch, W. Kruse, M. Munk, H. J. Reitboeck, "Coherent Oscillations: A mechanism of feature linking in the visual cortex?" *Biol. Cybern.*, vol. 60, pp 121-130, 1988.

[2] C. M. Gray, P. König, A. K. Engel, W. Singer, "Oscillatory responses in cat visual cortex exhibit intercolumnar synchronization which reflects global stimulus properties," *Nature*, vol. 338, pp 334-337, 1989.

[3] R. Eckhorn, "Neural mechanisms of from visual cortex suggest basic circuits for linking field models," *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp 464-479, 1999.

[4] R. Eckhorn, H. J. Reitboeck, M. Arndt, P. W. Dicke, "A neural network for feature linking via synchronous activity: Results from cat visual cortex and from simulations," *Models of Brain Function*, R. M. J. Cotterill Ed. Cambridge, U.K.: Cambridge Univ. Press, 1989, pp. 255-272.

[5] R. Eckhorn, H. J. Reitboeck, M. Arndt, P. W. Dicke, Feature linking via synchronization among distributed assemblies: Simulation of results from cat visual cortex," *Neural Comput.*, vol. 2, pp 293-307, 1990.

[6] J. L. Johnson, "Pulse-coupled neural nets: Translation, rotation, scale, distortion, and intensity signal invariance for images," *Appl. Opt.*, vol. 33, no. 26, pp 6239-6253, 1994.

[7] H. S. Ranganath, G. Kuntimad, J. L. Johnson, "Pulse coupled neural networks for image processing," *Proc. IEEE Southeastcon.*, Raleigh, NC, Mar. 1995, pp 37-43.

[8] H. S. Ranganath, G. Kuntimad, "Iterative segmentation using pulse coupled neural networks," *Proc. SPIE.*, vol. 2760, pp 543-554, 1996.

[9] J. M. Kinser, "A simplified pulse-coupled neural network," *Proc. SPIE.*, vol. 2760, pp 563-567, 1996.

[10] T. Lindblad, J. M. Kinser, "Image processing using pulse-coupled neural networks," Springer, 1998.

[11] J. L. Johnson, M. L. Padgett, "PCNN models and applications," *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp 480-498, 1999.

[12] G. Kuntimad, H. S. Ranganath, "Perfect image segmentation using pulse coupled neural networks," *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp 591-598, 1999.

[13] R. P. Broussard, S. K. Rogers, M. E. Oxley, G. L. Tarr, "Physiologically motivated image fusion for object detection using a pulse coupled neural network," *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp 554-563.

[14] T. Schoenauer, N. Mehrtash, H. Klar, "Architecture of a neuroprocessor chip for pulse-coded neural networks," *ICCIN*, RTP, N. Carolina, Oct. 1998, pp 17-20.

[15] J. M. Kinser, T. Lindblad, "Implementation of pulse-coupled neural networks in a CNAPS environment," *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp (585-590), 1999.

[16] G. Franke, G. Hartmann, A. Janke, M. Schäfer, "An accelerator for neural networks with pulse-coded model-neurons," *IEEE Trans. Neural Networks*, vol. 10, no. 3, pp 527-538, 1999.

[17] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational properties," *Proc National Academy of Sciences of the USA*, vol. 79, pp 2554-2588, 1982.

[18] G. Winkler, "Image analysis, random fields, and dynamic monte carlo methods: a mathematical introduction," Springer Verlag, Berlin, 1995.

[19] T. M. Cover, J. A. Thomas, "Elements of information theory," Wiley, New York, 1991.