Did the Earth Move for You?
A Novel Metric for Matching Iconic Images

Mike Robinson

MSc in Artificial Intelligence
Department of Artificial Intelligence
University of Edinburgh
1998

Abstract

The Iconic Vision System uses a combination of image-like and symbolic evidence to match images with models previously learned using a simple perceptron. Currently, matching employs a metric analogous to calculating the pixel-wise scalar product. Proposed is the use of a novel constrained version of the '*Earth Movers Metric (EMD)*'.

The linear programming problem representing the EMD is optimised using the *Simplex* method and should give a measure with more flexibility. In the iconic system's $R\Theta$ representation, the allowed 'shifts' between matchable elements represent local variations in scale and orientation.

The constrained version of the *EMD* was developed to reduce the computational requirements and prevent unwanted 'shifts' corresponding to large scale/orientation variations.

The current tests indicate that while the approach has promise, use of the metric would slow the system down unacceptably. Further work is required to speed up the computation, possibly by implementing a streamlined *Simplex* method and applying 'early failure' conditions.

Another new metric uses Gaussian blurring of the $R\Theta$ image to implement similar flexibility and seems to give comparable results with far less intensive computation.

A secondary attempt to improve performance was retraining the perceptron used to weight contributions to the overall match from the range of 42 feature match scores. The results from the retraining were disappointing, and imply that the features currently used may be weak and possibly do not give linearly separable result sets.

Acknowledgements

First of all, I would like to thank my supervisor, Dr R. B. Fisher, who suggested any clever bits that lie herein, drew detailed pictures of how to do them and exhibited great forbearance when I repeatedly misunderstood them.

Secondly, *huge* thanks are due to Dr. Anthony Ashbrook for drumming the basics of C++ plumbing into my head, a lot of hand-holding and patience in the face of regular bewilderment and for imparting invaluable advice as to good programming and experimental practices.

Prof. Saul Teukolsky at Cornell University deserves mention for assistance provided with respect to implementation of the *Simplex* method routine, as does Yossi Rubner of Stanford University who contributed some valuable observations on future developments.

Finally, I must thank the EPSRC who provided vital financial support during my year of study via MSc Quota Grant 97401101.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The iconic recognition project has grown over several years to constitute an interesting alternative to traditional geometric/symbolic matching approaches. Although iconic representations (i.e. data in raw image format) are less compact and flexible than symbolic alternatives, they can be far simpler to extract. The system as it stands today incorporates geometric matching techniques in its use of subcomponent evidence, combining useful properties of both approaches.

The iconic recognition system was suggested by Jennens [Jennens 94] and developed by Grove [Grove 95], [Grove & Fisher 96]. The parallelisation of the system, which allows many matching processes to occur simultaneously, was initiated by Marques [Marques 96] and extensively redeveloped by MacKirdy [MacKirdy 97], [Fisher & MacKirdy 98].

The objective of the project was to incorporate alternative methodologies into the iconic object recognition system, with a view to assessing their usefulness and improving performance. The main enhancements desired were:

1. Use of the 'Earth Movers Metric' [Rubner *et al.* 98] in correlating features during the matching and model making processes.

2. Use of a Gaussian Blurred/Scalar Product metric for matching.

contributions to the match score for each model instance.

4. Investigation of the factors affecting the deterministic properties of the system and the efficiency of its basic operation.

It was also desired to restate existing results with greater clarity, including all initial conditions, to investigate the reproducibility of results and ascertain some approximate timing measures.

1.1 Structure of Dissertation

- Chapter Two covers the theoretical background and details of the overall system. This section is larger than most such reviews for two main reasons:

    1. The last such exhaustive review, at the outset of the project, was before extensive changes in both the principles and implementation of several areas of the recognition process. Since then only general overviews and examinations of revisions in isolation have been undertaken, which is not helpful in understanding the system in its current form.

    2. There have been areas of semantic ambiguity throughout previous descriptions of the detail of processes involved. These are primarily in distinguishing clearly between the definition of a pixel in two representations and between a feature in two contexts. Although simple in themselves, lack of clarity in their usage was endemic to previous discussions and presented a considerable barrier in appreciating the nature of the functioning of the system.

the system to increase efficiency in computational power usage and thus reduction in processing times.

- Chapter Four discusses the retraining of the perceptron employed in ascertaining the important features for recognising a particular model.
- Chapter Five considers the implementation of the new model/image matching approaches in model construction and image recognition.

Chapter 2

Background to the Iconic Recognition System

Deriving initially from hypotheses of the dual channel nature of human visual perception as being constituted by both symbolic and iconic components [Farah 90], the iconic recognition project has been developed to the level where its performance allows reasonable experimentation in its parallelised form.

The developmental path has diverged somewhat from attempts at strict biological plausibility in the respect that symbolic structures to enable use of subcomponent evidence have been incorporated and representations now include linked lists and hash tables – all unlikely to be found implemented in the primate visual system.

The initial motivation behind the development of the system remains however: consideration that the advantages of the ease of accessibility of iconic images when compared with the difficulties encountered in more traditional geometric methods may outweigh problems relating to manipulating iconic data.

These difficulties arise because the base data itself is bulky and centre-

creation (as noted above) have however substantially reduced the extent to which the raw data is used. In essence, useful aspects of both approaches have been fused into a dual system, still broadly iconic but drawing on desirable elements of the geometric approach.

Further, continual increases in processor speed coupled with the increases in performance derived from efficient parallelisation reduce the relative importance of such considerations.

2.1 System Outline

The overall structure of the system is as shown in Figure 2.1.

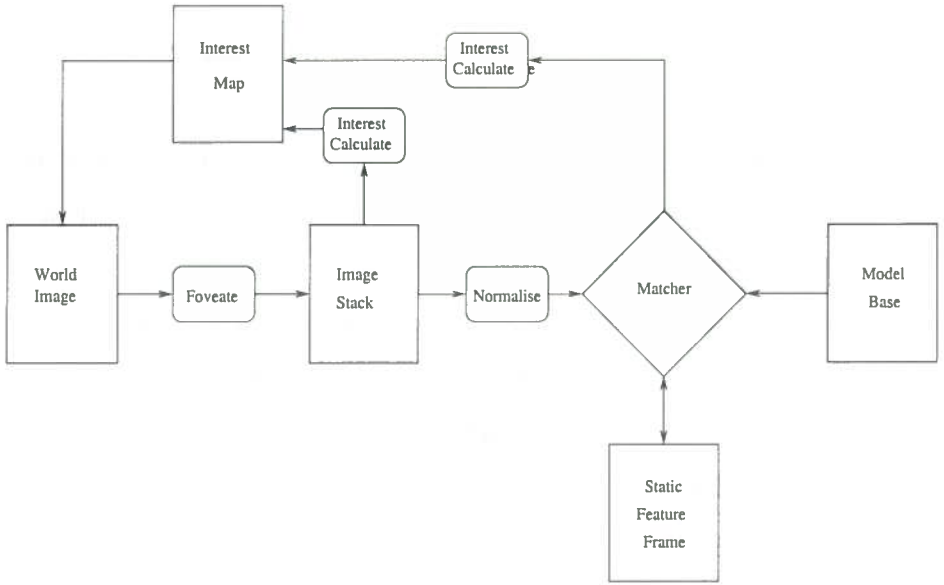


Figure 2.1: System outline

In summary:

- Conversion of a section of a Cartesian (i,j) image into a polar $R\Theta$ image with resolution variant over distance from the (foveation).

- Matching these primitives with similar sets representative of a range of models, returning a positive match for the highest score above a threshold value.

- Deciding on a new point to examine based upon an estimate of the likely position of further positive matches.

2.1.1 Input Image

The basic input image is a static RGB image, currently of size up to 512^2 pixels in *(i,j)* coordinates. This is referred to as the *'World'* henceforth.

2.1.2 Foveation

The first processing carried out on the world image is *foveation*. The name for the process derives from the sensor architecture found in the primate retina, which is such that there is high resolution at the centre (the **fovea**) and an exponential decrease in resolution as one moves outward from there.

Simulation of this pattern is one of the core elements of the system: the arrangement is one of 20 rings, each split into 48 sectors. A circular region as illustrated in the section shown in Figure 2.2 constitutes a 'pixel' in an $R\Theta$ representation. To distinguish these from the world image's Cartesian *(i,j)* pixels, I shall henceforth refer to those in the $R\Theta$ representation as *'retxels'*.

As in the primate retina, the area of the receptive field of each retxel increases exponentially (here by a factor of 1.2) with distance from the central foveation point. Not only does this achieve focusing of processor power in the central area (which must thus be engineered to be an area of high 'interest' in some way) but also the consequent 'blurring' of non-central images implements a degree of figure-ground separation automatically.

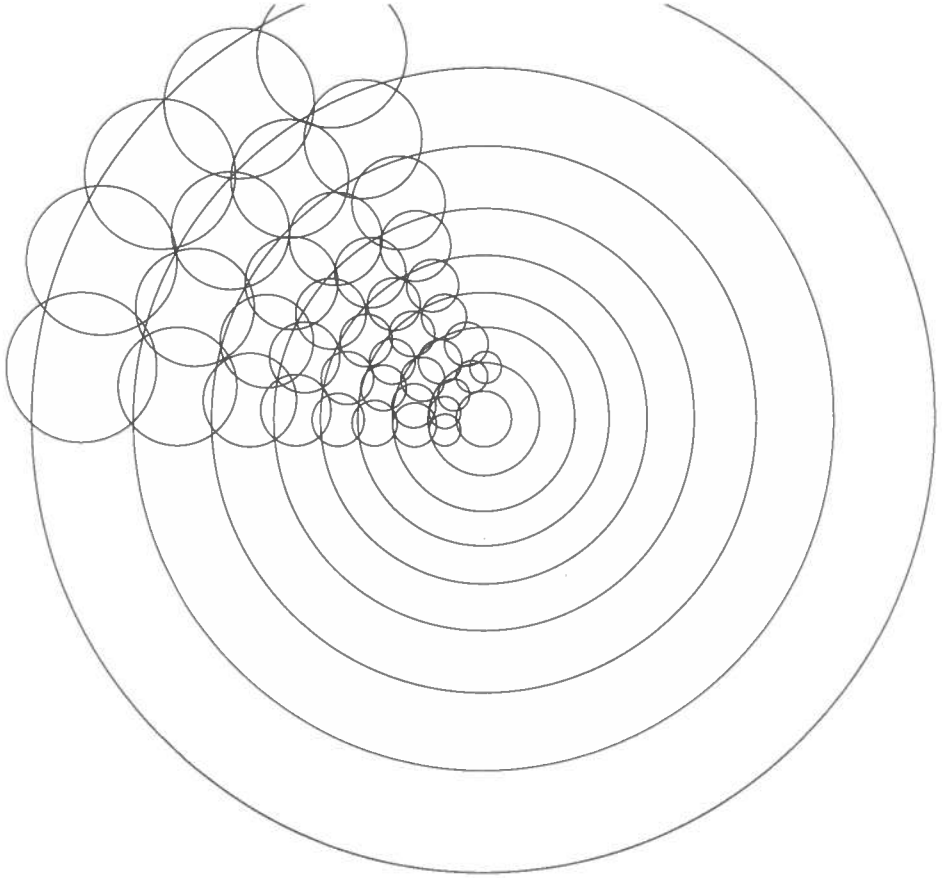


Figure 2.2: Retxels in the $R\Theta$ representation

The circular receptive fields are organised to have a 33% overlap, thus avoiding gaps in the overall receptive field. Some analogy exists between this foveation approach and the use of Gabor wavelets to partition an image.A crucial distinction is in the saccading procedure as an alternative to applying static functions at varied scales.

The area 'seen' at each foveation point is just a small subsection of the world, currently a roughly circular region of diameter 128^2 pixels. As will be noted later, the foveation centre changes on each iteration, dependent on the nature of the world image.

For each instance of foveation, a temporary image stack is constructed. The image stack records the strength of 'features' in a region around the foveation point. The 'features' extracted correspond to the primitives postulated by Marr [Marr 80] as being important in primate vision.

In previous papers these have been referred to simply as 'features', but this term has been used interchangeably, and without notice, to also refer to *world features* such as *noses, eyes* etc. This has often made the discussions of the processes involved quite opaque. Henceforth, I shall refer to these 42 basic items as *primitives*, reserving the term 'features' for the composite items as mentioned (currently facial features). Each model is of a specific feature and is characterised by a weighted sum of inputs over the 42 channels corresponding to the primitives at a point.

In summary, the following primitives are extracted:

- The three colour planes giving raw intensity images for each retxels
- Edges at four orientations ('vertical', 'horizontal' & 2 'diagonals')
- Corners (non-directional)
- Blobs: On- and Off- centre (i.e. a dark region surrounded by light pixels or a light region surrounded by dark pixels.
- Bars: On- and Off- centre, at two orientations ('vertical', 'horizontal')

Each of these is captured using images at scales of 50%, 100% and 150%, which gives the same overall effect as spatial frequency filtering of the image. The image planes are, as their name suggests, stored as grey level images, with the intensity at a point corresponding to the strength of the primitive extracted at the corresponding pixel. It is these 42 images planes which

subcomponent consideration).

Although, by their nature, such representations use up significant memory, each is only retained for the current saccade and so, in effect, the same memory may be reused in subsequent saccades.

It should be noted that 'horizontal' lines in an $R\Theta$ representation are concentric circle sections in the corresponding Cartesian world image. Similarly 'vertical' lines in an $R\Theta$ representation are, in Cartesian terms, radii sections.

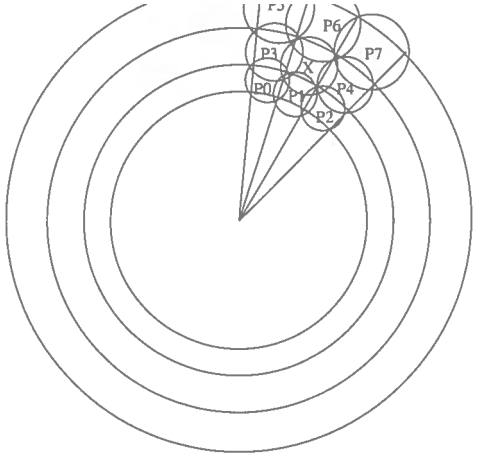
On the first iteration of the system, the image stack is the sole input to the 'image end' of the matching algorithm and is by default extracted from the centre point of the world image. This point can vary between images and should be noted explicitly in for each as a range of up to four may be chosen, which variation is sufficient to cause the foveation point to follow divergent saccade paths).

2.1.4 Primitive Extraction

Primitive extraction is achieved by a simple local neighbour process using a 3x3 operator passed across the $R\Theta$ image, registering the absolute value of the convolution score at its central retxel. The operator is illustrated to define labelling in Figure 2.3 along with a diagrammatic representation of its relationship to the overall foveated region (not to scale).

Primitives are specified by operations as typified in table 1, with their graphical representations approximated in Figure 2.4.

At the end of the primitive extraction process, the system holds data on 42 image planes which represent the strength of the relevant primitives in the currently foveated region. It is solely upon these that initial matching is performed.



P0	P1	P2
P3	X	P4
P5	P6	P7

Figure 2.3: Two depictions of the convolution region

Table 2.1: Primitive specification formulae

1. 'Vertical' Edge	$\|(P0+P3+P5)-(P2+P4+P7)\|$
2. Negative corner	$min([P1-X],[P3-X],[P4-X])-\|X-P6\|/2$
3. Neg. blob	$min([P0-X],[P1-X],[P2-X],[P3-X],[P4-X],[P5-X],[P6-X],[P7-X])$
4. Pos. blob	$min([X-P0],[X-P1],[X-P2],[X-P3],[X-P4],[X-P5],[X-P6],[X-P7])$
5. 'Vertical' Bar	$min([P1-P0],[P1-P2],[X-P3],[X-P4],[P6-P5],[P6-P7])-max([X-P1],[X-P6])$

2.1.5 Static Feature Frame

Bypassing consideration of the actual matching process until Section 2.1.8, the product of a successful match is entered as a new item in the Static Feature Frame *(SFF)*. This is in the form of a 'hash table' and constitutes the visual memory of the system. Unlike the image stack it has continuity of existence over all saccades until the end of the run.

When a positive match between an image and a model's feature planes (or primitive planes by our terminology) is achieved, the system adds a new *SFF*

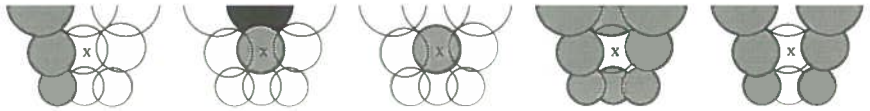


Figure 2.4: Illustrative depictions of the primitives 1-5 in table 1

item to the linked-list *'Matchlist'* which corresponds to the type of feature just matched.

Each SFF item is of the form *(Model Index, Point, Match Score)*. The *Model Index* gives the position of an entry in the SFF hash table which corresponds to a pointer to a linked list of the relevant type of feature matches. *Point* specifies the Cartesian location at which the match occurred. *Match Score* is a measure of the strength of the match.

Thus the memory of the system for recognised sub-components is implemented symbolically and geometrically.

2.1.6 Model Base

The construction of the model base is outlined in detail in Section 2.3. It is sufficient for now to note that the model base is comprised in part of one average model for each of the feature types being searched for. These components are each comprised of 42 primitive feature planes identical in nature to those of the foveated world images, i.e. $R\Theta$ images, and are constructed independently in advance from exemplars of the feature in question.

In the matching process, individual primitive feature planes are compared. The sum of the 42 match scores is denoted the *feature score* and is weighted to be the most significant contributing factor in the overall match score.

The other component of each model is of an identical format to the SFF items mentioned above. This is the model's label set which indicates all models which are associated with the current model. These *Associated Models*

in a specified spatial relationship to it.

The hash table which constitutes this structural portion of the model base has as entries pointers to linked lists of items with the form: *(Title, Point, Strength)*. Here, *Title* refers to the name of the feature of which the associated model is an exemplar, *Point* is its predicted relative position, and *Strength* is a measure of the assigned importance of the type of feature.

2.1.7 Image Scaling and Rotation

A major benefit of the $R\Theta$ representation employed is that rotation about the optical axis (i.e. in the world image plane) and scaling of the images are given by translation along, respectively, the 'Θ' and 'R' axes of the foveated representation. Also, translation is managed as part of the saccading process detailed below. Rotation out of the world image plane is outside the scope of the system, however, which would need to be trained on a range of separate representative models to handle such different object viewpoints.

Thus, by applying a range of $R\Theta$ translations corresponding to rotations and scaling (which are specified by the user as part of the matching process), the system exhibits quasi-invariance in that respect.

2.1.8 Matching

As the system stands, the matching procedure has two strands: *image matching* and *label matching*.

Image Matching

The initial source of confidence for the matching of an image with a model is purely *intrinsic evidence* i.e. derived wholly from the degree to which the foveated area matches with the models stored in the model base. When the

for matching and so is critically important.

What is required overall is the matching of 42 channels (each relating to a primitive) of image data with 42 channels of model data at each foveation point. In theory, this requires the simultaneous matching of the 42 channels i.e. using a *multi-variate correlation function.* This is computationally expensive and a good approximation has been found to be [Fisher & Oliver 97] the average of the 42 single channel correlations.

$$\rho = f(\sum_{k=1}^{42} \rho_k w_k + w_0) \tag{2.1}$$

Here, each of the 42 individual correlations is weighted by a coefficient w_k which reflects its relevance in generating an identification. These weightings are learned a single layer, linear perceptron neural net during the model base construction (as detailed in Chapter 4). w_0 is the *a priori* probability that the data is in the relevant class (defaults to 0). *g(.)* is the sigmoid function:

$$g(x) = \frac{1}{(1 + e^{-x})} \tag{2.2}$$

The single-channel correlation itself is computed as the scalar product between corresponding image and model pixels. The models vary in size and so the system extracts a similar sized area around the foveation point for the matching.

Label Matching

In addition to the basic image/model correlation, the system is capable of using *extrinsic evidence* to aid in finding good matches. This is evidence which relies upon data not directly available from the foveated image and image-like model. Knowledge about the world, specifically here knowledge

structure are used to provide additional match confidence.

Label matching takes place between the elements of the two hash tables which comprise the Static Feature Frame and the structural portion of the Model Base respectively. It is this matching which comprises the heart of the use of Full Subcomponent Evidence (FSE).

The label matching function is designed to give the best measure of correlation where the number of associated subcomponents located in line with expectations is maximised.

$$\sum_{\forall t \in T} \sum_{\forall \widehat{p} \in p_t} max_{\forall \widehat{d} \in D_t}(g(\widehat{p}\widehat{d})) \tag{2.3}$$

The intuition behind this function is that we want to sum over all model (or subcomponent) types and then for each consider the predicted point at which the label indicates there should be an instance of the model. The fit score of these points is then summed. So:

t is the type of an associated model

T is the set of all model types

p is the prediction set of labels (scale(σ),rotation(R),translation($\widehat{f}$))

(so $\widehat{p}$ is the predicted position of model type t in p_t)

D is the set of current SFF matches

p_t is p$\cap$t; D_t is D$\cap$t

$g(\widehat{p}\widehat{d})$ is the label correlation function, which is given by :

$$g(\widehat{p}\widehat{d}) = w\rho e^{-\beta\delta(g(\widehat{p}\widehat{d}\widehat{f}))} \tag{2.4}$$

where:

w is the weighting based on the model's importance

ρ is the matching score in the SFF

and δ, the goodness of fit of predicted to actual label positions is given by :

$$\delta(g(\widehat{p}\widehat{d}\widehat{f})) = \frac{\| \widehat{p} - \widehat{d} \|}{\| \frac{1}{2}(\widehat{p} + \widehat{d}) - \widehat{f} \|} \tag{2.5}$$

Integrating Intrinsic and Extrinsic Evidence

As was noted above, initially the system must rely solely upon intrinsic evidence to begin building its SFF and in effect to hypothesise about the nature and structure of the world. Thus, where a threshold is set for indicating a match from the summed correlation scores from images and labels, it must be set at such a level that the intrinsic evidence can breach it by itself.

It is, however, considered to be undesirable for this to be possible for the extrinsic component, in consideration of the potential deceptiveness of subcomponent evidence, the allied risk of false positive matching and consequent efficiency losses due to misidentifications. The arbitrary level for the relative weighting, based originally upon intuitive sense is a weighting in favour of intrinsic evidence given by $\alpha = 0.7$. In view of this setting, the overall threshold for recognition is set at $T = 0.68$.

The highest scoring match above this threshold is considered to be 'recognised'. Recognised features are added to the SFF and can be used to assist the matching procedure on the next iteration.

Where:

$$Score = \alpha(intrinsic\_score) + (1 - \alpha)(extrinsic\_score) \tag{2.6}$$

The search can be pruned whenever

$$(intrinsic\_score) < \frac{1}{\alpha}(T + \alpha - 1) \tag{2.7}$$

extrinsic score.

Where two models give the same score, that which is matched first will be recognised, which has important implications with respect to reproducibility of results (see Section 3.3).

2.1.9 Interest Map

So far, the process detailed has related to just one iteration of the system, i.e. foveation on one point, extraction of data for matching and the update of the system's memory in respect of successful matches.

The other key component of the system is a mechanism to drive the foveated point to significant areas, to direct the attention of the system in such a way that processing power is focused on the areas we wish to be explored.

The system addresses this goal by constructing an *interest map* of the world in Cartesian coordinates, such that the foveation point in the next iteration is chosen to be the point with the highest interest assigned to it. As in the case of matching evidence, there are two components which generate the level of interest assigned to a point, complicated here by the need for a suppression factor. The overall interest of a point, $A_{ij}^{(s+1)}$ at saccade $s+1$ is given by a weighted combination of $(A_{ij}^{(s)}, E_{ij}^{(s)}, I_{ij}^{(s)}, -S_{ij}^{(s)})$ as defined below.

Intrinsic Interest($I_{ij}^{(s)}$)

Once again, as with the matching considerations, at the beginning of the system's run the interest map will have no data derived from the world with which to develop any meaningful values of interest. There is no data which can be derived before a 'start point' is chosen, and so this must be arbitrary.

The system is initialised by modelling the interest as a Gaussian centred on the middle point of the world image. Thus, in the absence of any world-

from the centre in search of something to look at.

The intrinsic interest function derives its values purely from the foveated image region. The primitive responses recorded by the outer half of the retxels is defoveated into the Cartesian interest map, simulating the effect of 'seeing something out of the corner of your eye'. Thus, if the 'wandering' foveation point comes within range of something which appears interesting, the point will then be driven there.

$$I_{ij}^{(t)} = \frac{1}{d_{ij}} < defoveate > \sum_{prim'\_at\_largeR} f_{R\Theta} \tag{2.8}$$

where: d_{ij} measures the distance from the foveation point
prim' refers to an augmented subset of primitives (a heuristic combination of vertical corners, +/- blobs, negative bars and a kind of opponent colour R-G and G-R blobs at 3 scales)

The intrinsic evidence is given the greatest weighting in the overall interest calculation.

Extrinsic Interest, $(E_{ij}^{(s)})$

As its name suggests, this factor uses *a priori* knowledge to estimate the interest ascribed to various world points. Some is supplied by the model's label set of associated models, some by the system's memory of previously matched points.

Each time a new feature is matched, it suggests the existence of its group of associated models at relative points in the world. Thus, the interest assigned to the corresponding points should be increased so long as there has not already been a feature identification there.

To implement some degree of search control and also in recognition of the realities of prediction uncertainty increasing away from the foveation point,

$$I_{map}(\widehat{x}) = \alpha e^{-r\delta(\widehat{x})} \tag{2.9}$$

$$\delta(\widehat{x}) = \frac{\| \widehat{t_{pi}} - \widehat{x} \|^2}{\| \widehat{t_{pi}} - \widehat{f} \|^2} \tag{2.10}$$

$$\tag{2.11}$$

Where I_{map} specifies the points to be attended

$\widehat{x}$ specifies points in the region of the predicted point $\widehat{t_{pi}}$

α is a scaling coefficient intutatively set to 250

r is a scaling coefficient intutatively set to 100

$\widehat{f}$ is the current foveation point

The relationship specified to scale, rotate and translate the associated models is used and the resultant predicted model points are compared with the current contents of the SFF. The previously defined label evaluation function is used to ascertain whether there is an extant match within a specified range of the predicted point. Where a match has been made already, no increment in the interest of the area is made.

Where there has not been a match previously the interest of points in a specified region about the predicted point is increased. The further one moves away from the current foveation point, the less accurate will the predicted point be, and this is modelled using a Gaussian distribution which causes an area of 20% of the distance between the current foveation point and the predicted point to be specified for interest enhancement. So, if we define the radius of interest about the predicted point as δ:

$$\delta = 0.1(\| \widehat{p_i} - \widehat{f} \|) \tag{2.12}$$

$$g = \sqrt{(p_{i_x} + j - \delta)^2 + (p_{i_y} + k - \delta)^2} \tag{2.13}$$

$$\mathbf{I} = \gamma e^{\frac{-\beta g^2}{\delta^2}} \tag{2.14}$$

where (p_{i_x}, p_{i_y}) are the x and y coordinates respectively of predicted position p_i of label $\mathbf{i}$

g is the relative extrinsic interest value of points in the region
γ is the maximum interest to be added in the region.
I is the pointwise interest addition amount

γ is chosen so as not to dominate the interest calculation, yet to give a significant contribution. Again, the value was chosen intuitively: based on observations indicating that the maximum value was of the order of 1000, with a mean value of 500, γ was set at 250.

Suppression ($S_{ij}^{(s)}$)

If the overall interest calculation was to be left as based upon a two-part function reliant on summing of Intrinsic and Extrinsic evidence, the system would tend to become fixated upon a single identified point, to which foveation would thus repeatedly return. This is not desirable behaviour, as we wish all points of above a certain interest in the world to be attended at some point. Returning to the same point is not efficient unless it is to implement some form of 'double-check' on the match made at that point, which would not be the case here.

The suppression element applies a decrement to the interest when a match is made and it is currently set at the size of the located model. This prevents return of interest to the area in the short term, although interest updating still occurs and so the point can be visited again later in the run. Where the feature located is large, there is a danger that relatively small and proximate features may be neglected due to incidental suppression and furthermore, if a feature is incorrectly matched, the suppression may delay or prevent the system's recovery.

The interest calculation provides an estimate of the point of greatest interest from the respect of the current foveation point. When this is located, the system redirects its foveation focus to that next high-interest point and begins a new iteration. A new image stack replaces the former one and is used for new extraction/matching/memorising and interest calculation procedures.

To compensate in some part for the inherent spatial uncertainty of the prediction process and to implement an observed biological process, *micro-saccading* about the foveation point takes place. In essence, matching is attempted at pixels:

[(-3,-3),(-3,0),(-3,+3),(0,-3),(0,0),(0,+3),(+3,-3),(+3,0),(+3,+3)]

relative to the current foveation point. Thus the system attempts to obtain information on the best match score in the neighbourhood.

The SFF is built up based upon the best match evidence over the nine points, with the model calculated based on the evidence from the the micro-saccade point with the best score for the matched model. Further, the saccade point is allowed to 'creep' in that direction i.e. that from which the best matches are generated.

2.2 Parallel Implementation

The micro-saccade is currently used as the unit of parallelisation in the system i.e. the combined process of extraction and correlation of primitives at the nine points in the neighbourhood. The primary reason for this is one of efficiency: each micro-saccade requires its own foveation process, thus if each processor carries out one micro-saccade and 'n' correlation steps, each requires but one foveation. If each processor carries out one correlation and 'n' micro-saccades, each requires 'n' foveations.

cessors, while interest calculation and saccading proceeds in serial, carried out only by the 'main' processor (Figure 2.5). Despite this, the implement-

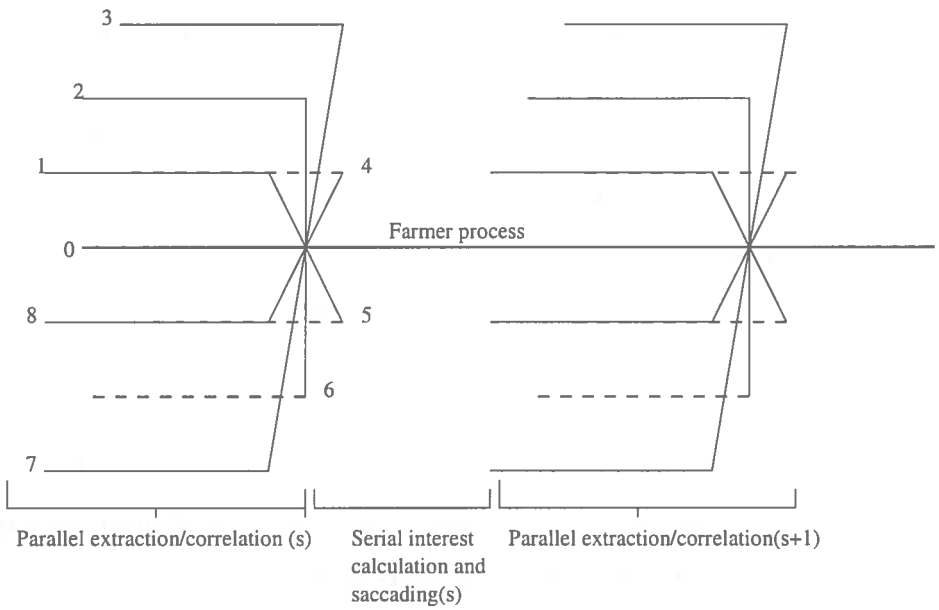


Figure 2.5: iconic system's parallel and serial processes

ation of parallelisation utilised allows for great flexibility in the work distribution process and the number of processors which may usefully be applied to the running of the system.

The model used is the 'Farmer/Worker model' which is a very flexible implementation of *task farming* whereby the granularity of the task is quite malleable, dependent on factors such as the number of processors available, changes to the nature of sub-problems created etc.

2.2.1 Farmer

The system has one processor whose number identifies it as the 'main' processor, the functions of which are:

saccade actions at a point) from the overall matching problem (the *schedule*).

- Receive and combine parallel sub-solutions (the score for the current micro-saccade action) into an overall solution (the overall score).

- Execute the run's sequential processes.

The sub-problems are specified by forming a schedule which constitutes a set of actions (here in terms of scaling, rotation and model choice) into which the overall task is divided. The farmer then 'steps through' the task allotting each new action to the first worker to finish its previous action.

The best match generated is used to update the SFF (the Farmer holds the master copy, with updated versions sent to the workers along with their instructions) and the saccading calculation is then made in serial before commencing the next iteration.

As the workers are not synchronised in terms of their communication with the farmer, there is some danger of a 'bottleneck' forming where action sizes are similar. This arises especially where the processors have different speeds: as the main saccading process is still serial the farmer waits to begin this until every processor has finished it's last action. Thus, the critical value is that of the slowest processor: as this may have either just started or nearly finished it's final task as the others begin to wait, it is not an invariant quantity between runs.

2.2.2 Worker

The system has many (currently 9) processors whose numbers identify them to be 'subsidiary' processors, the functions of which are:

- Receive parallel sub-problems from the Farmer.

- Return parallel sub-solutions to the Farmer.

The worker units are 'work hungry', which condition drives the overall balance of task resolution: as soon as a worker finishes its current action, it returns a message to the farmer requesting more work. This means that workers given small tasks are not inhibited by having to wait for those given larger tasks to finish, but can get on with handling further tasks (where they exist) in the meantime.

Each worker has a copy of the SFF so it can use subcomponent evidence in computing its relevant match scores The copy is updated by the farmer at the end of each saccade to give current match information.

2.3 Model Construction

The models used in the matching process are generated in a substantially automated process, in advance of the matching procedure. The user provides training images (worlds) and input files which specify:

- Model type and size.
- Convergence limits for the neural net (which learns to characterise general features.
- The total number of training points desired.
- The images to use for training.

The primary source of manual input required by the system is the location of positive training points in the training images. These correspond to areas of high interest which would draw the system's attention **and** which correspond to actual features. The training points are not central with respect to

command the maximum interest (e.g. for an eye, the eyelid; for a nose, a nostril edge). These are the points to which the system would saccade if run without a model base.

When the required input files have been provided, the model making utility first loads the required worlds and from each extracts an average model.

A positive training point in a training image corresponds to an instance of a particular feature. A set of 42 primitives ($\mathbf{P}$) are extracted at such a training point and if there is only one instance of a feature in the training image (for for faces, this would be *nose, mouth, face*), then this is returned as the world's average model.

Where the are multiple examples of a feature in the training image, (for for faces, this would be *eye*) there are an equal number of positive training points. The feature scores are found at each and the channels' scores are accumulated individually. The average model is returned by division of each channel's score by the number of positive training points for the training image.

Each channel of these average models is summed and division by the number of training images to give an overall average model. By using multiple training images in the model making process it is intended that the model will be representative of the general feature format, rather than 'over-fit' to the specific qualities of any one training image's feature example.

The negative training points (areas of high interest which do not correspond to an actual feature) are automatically generated by the system. They correspond to areas of maximum interest on successive saccades with no model present. A threshold around identified model features is employed to prevent positive and negative training points being in close proximity.

correlation routine to match it against all positive and negative training points in the training images, generating the *feature score* for each of the points.

The neural net takes as input these feature scores $\mathbf{F}$ and has as desired output for each its ideal score $\mathbf{I}$, which specifies whether the point is a positive or negative training point. Thus the net aims to learn the appropriate weights which will reduce the error in terms of Mean Square Difference (MSD) of *predicted* compared with *actual* values for the ideal score at a point. The ideal score was originally '1' for a positive training point, '0' for a negative training point.

This process is considered in greater detail in Section 4.1.

Thus, when a set of 42 primitive scores is returned by the correlation routine during a run of the main recognition program, the net is used to weight them appropriately. Ideally this should relate the primitive scores to a corresponding ideal score and thus identify positive instances of a particular feature's model.

Due to convergence problems, the learning criterion has been set in terms of number of iterations performed by the net. 10^6 iterations is chosen, as this corresponds to a level where the convergence gradient (i.e. MSD improvement per iteration) has levelled out at an MSD of ≈ 0.005.

Again the correlation method used is similar to the dot product of two unit vectors, i.e.

$$\rho = 2\frac{\sum_{i,j} M_{i,j}E_{i,j}}{(\sum_{i,j} M_{i,j}^2)(\sum_{i,j} E_{i,j}^2)^{\frac{1}{2}}} - 1 \tag{2.15}$$

where $M_{i,j}$,$E_{i,j}$ are respectively model and image $(R\theta)$ primitives, $\mathbf{i,j}$ are $(R\theta)$ coordinates.

Chapter 3

Minor Enhancements and Tests

3.1 Memory Access and Leakages

MacKirdy's program operated considerably faster than the original parallelised version due to many coding improvements implemented to enhance efficient operation.

Upon analysing the running of the serial version using the 'Purify' tool, however, memory access problems and memory leaks were uncovered which could reduce the efficiency of memory usage and speed of operation of the program. In the original, approximately 23 M-bytes of free memory were being allocated, 92.3% of which was leaking.

The code was extensively overhauled to change the methodology and as a result of this change, memory allocation reduced to approximately 1.2 M-bytes, with 45.6% leaking. Correction of a variety of individual, non-fatal programming errors dropped allocation to 225 K-bytes, a reduction to approximately 1% of the previous total. At this point the time cost of further investigation looked likely to outweigh the potential efficiency benefits.

locality of identification of a feature. For images where the first feature identified is large (and thus the suppression zone also), the effect on subsequent feature identification was felt to be unhelpful. Accordingly, prior to the major changes reported herein, the suppression zone's size was reduced, so all results must be considered in the light of this.

The control results prior to implementation of the current changes can be compared with those of MacKirdy to ascertain the effects of this change.

3.3 Reproducibility of Results

Prior to implementation of any enhancements, investigation was conducted into anecdotal reports that the saccading path, (and so model recognition performance of the system) was not deterministic. Initially, multiple runs of the system were executed on the same image. In this first test, the system was run 15 times for 20 saccades on image *bebie1.ppm*, under which testing no evidence to support the contention of non-determinism was produced: the saccading path and hence models identified, their recognition score and the saccade when discovered were all invariant over the runs.

Hypothesising that this behaviour could nevertheless vary depending upon the image being explored, the test was repeated on all of MacKirdy's test images (Appendix B). Taken in total this is over one hundred runs, during which significant non-deterministic behaviour would become apparent.

For five of the seven images the behaviour of the system was, as described, invariant over all runs.

In two cases however (*c5.ppm and fce5.ppm*), there were found to be two alternate saccade paths. For both images, the saccading path began as the same for each variant but diverged after three (*c5*) or four (*fce5*) saccades. The initial differences were small, but the path divergence increased with each saccade.

with an apparent weighting in the probability of one being followed over the other (for *c5* the ratio was 12:3, for *fce5* 11:2).

To further investigate this behaviour, the test was repeated on each image for 100 runs. The origin of non-determinism was not clear from examination of the behaviour at the divergence point. More detailed investigation, going back over earlier model matching behaviour, however, revealed the cause of the two paths.

To take a specific example: two saccades prior to divergence, there was a match with the *'NOSE'* in both instances at a certain point with identical scale and match scores. In one instance, however, the rotation of the match was 0 in the other, it was 1.

This is possible by virtue of two primary factors:

1. The models in the model base are, for most features currently used, and certainly for the nose (a nostril, to be precise), largely symmetric (here the key being rotational symmetry at 180 degrees).

2. When two models are located with the same model score, the first located is that reported as a model match. There is no intuitive reason to support preferring a match of any rotation, scale or indeed at any of the micro-saccade points over any other.

   It is in the nature of the parallelisation that it is dependent on external variables (at least for non-dedicated worker processors) which processor will be fastest, which tasks it will run and so which results will be reported first. This constitutes an inherent non-deterministic feature of the system in its current configuration.

Further, it was noted during this investigation that, rather than the registration of the same score being found for different model instances etc. being rare, it occurred several times in runs of 20 saccades over each of the eight

component evidence from the opposing orientations. This gives a different interest map in each case, which, after two saccades caused the saccade paths to drift apart.

3.4 Parallel Timing Results

Previous results on timing for the system have addressed individual micro-components such as message passing between parallelised units, but no data has been reported as to the overall speed for a run.

This has primarily been because of unavoidable aspects of running a parallel system on processors within a working department: the individual machines are open to access at all times by other users (present and remote) and variable machine availability restricts precise controls over the specifications of machines employed.

It is nevertheless possible to mitigate these factors sufficiently to give a meaningful, if restricted, measure of the system's relative performance speed. The system was allowed to execute 20 runs of 20 saccades each on image *bebie1.ppm*. This was performed between the hours of midnight and 4 a.m. on a Saturday night to minimise the effects of external factors on running speed, with repeated manual checks on the allocation of processor time to the system. The machines chosen were all *'Sun SPARCstation 4'* units, five being dedicated units in the vision lab, five units in the general terminal room. The processor clock rates did vary between these machines, said variation giving the non-determinism discussed in Section 3.3.

The machines had, in fact, no other users for the course of the experiment and so the results (Appendix A, Table A.4), although still machine dependent are good for comparison with a similar ring of units dedicated to such a system. The overall running time oscillated, quite regularly, alternating between 248 and 252 seconds for 20 saccades, during which five features were

each saccade in compared runs varied about a distinct mean.

This is to be expected, as:

- The amount of work done in each saccade is dependent upon the environment of the saccade point and so should change from one saccade to another,
- The runs were over identical saccade paths times should follow the same pattern.

The experiment highlighted two incidental points. Firstly, detailed examination of the saccade path showed that the foveation point saccaded to the same point twice during the run, a point where no features were found. This is not a problem in itself as the change in subcomponent evidence between the two saccades could have changed sufficiently in the meantime to push a potential feature's score above threshold.

In one case, however, the foveation point remained at the same coordinate over two consecutive saccades, despite finding no features there. This is inefficient as there could be no changes in subcomponent evidence to cause a match on the second matching attempt.

It would be possible to avert this by simply implementing a penalty for such behaviour, but this would simply mask the cause. More appropriate would be retuning of the attention mechanism.

Secondly, the system cannot be left to reliably execute multiple runs unattended in its parallelised version: roughly every 10 runs, the parallelising routine causes premature discontinuation e.g. when encountering an accessing problem with one or more of the machines. Although external to the image program itself, this is a significant practical experimental point.

where matching results for two models are equal, the first acquired is reported as recognised. Due to inherent 'random' timing effects where pool processors are used, it is possible models to be recognised at in different contexts from run to run. The effect of this can be magnified by the predictive effects of the sub-component evidence, giving divergence in results overall thereafter.

While the non-determinism does complicate evaluation of the system's performance, as it is caused by subcomponent contributions due to relatively rotated matches, there is likely to only be two possibilities, at least where on one such rotated match occurs.

The problem only arises in the system's parallelised incarnation, as do the problems noted with the inability to reliably leave the program running unattended.

The parallelisation is important to bring the system's operation speed up to acceptable levels, but the problems caused are inconvenient. Delays were caused by problems with the parallelisation module's library files. incompatability with 'Purify' and repeated loss of connection with remote units.

Where speed is important, the parallel version is the first choice. Where reliability, it terms of deterministic behaviour, robustness of operation is required, (and indeed if 'Purify' is to be run) the serial version remains more reliable.

3.6 Further Work

Further investigation of the system's basics would be useful, with details of undocumented traits being accumulated. The attention mechanism is allowing consecutive visits to the same point and investigation of changes to avoid this would be worthwhile. These include a plot of the 'region of attraction' constituted by the variation in match score around a known feature point for each new metric/ model base implemented. A restricted version of this test,

and C++, serial and parallel implementations.

In short, the code is rather a mixture of styles and conventions. Much work has been done during this project and before to tidy the structure, but more is needed. The code is usefully commented, but the comments include worrying statements concerning compiler hacks, obsolete functions, classes to be defined and other outstanding work. It is possible that inefficiencies in operation are disguised by these factors. Dispensing with the remainder of these would take a competent programmer no inconsiderable time.

Chapter 4

Perceptron Convergence Criterion

4.1 Motivation for Changes

As noted above, the role played by the perceptron is to take as input actual primitive plane matching scores and to output 'ideal scores' corresponding to identification as a positive match or to relegation to a negative result. This is achieved by the perceptron's learning from the training set of multiple instances of positive and negative feature instances. Significance is attributed to the score from a particular primitive's match for a model feature by varying the weighting assigned to its input channel.

4.1.1 Perceptron Training: Background

The perceptron used in the iconic system is very simple and really constitutes no more than a mathematical operator intended to classify its between two alternatives. For a single layer 'network', which this is, a perfect classification is theoretically possible only where the inputs are linearly separable.

To amplify, we can view the problem as one of separating points in *n-dimensional space* using an *(n-1) dimensional hyper-plane* If we consider the $n = 2$ case, we can consider the problem graphically as one of separating

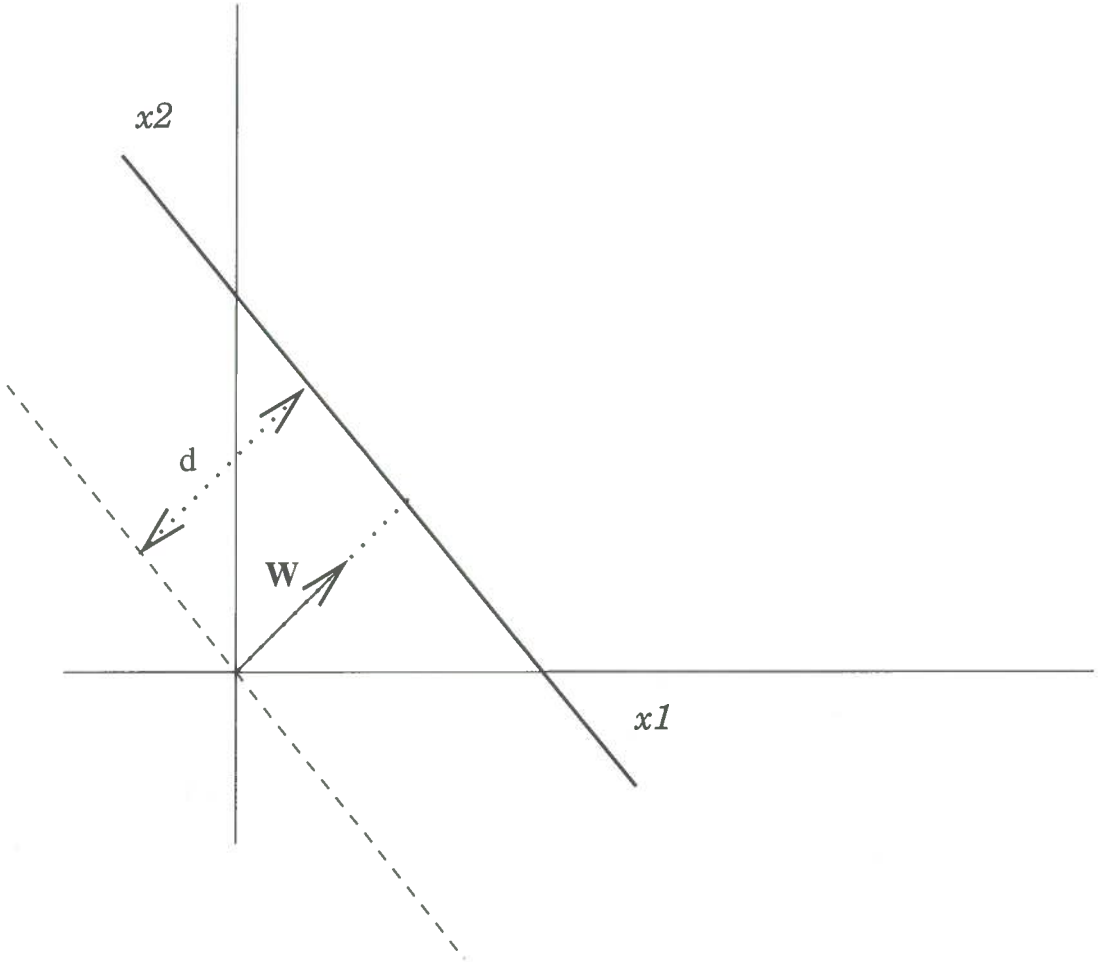


Figure 4.1: 2D linear decision boundary

we can write the equation of the line as

$$f(x) = wx + w_0 \tag{4.1}$$

Here, w_0 is called the bias and specifies the normal distance from the origin. w defines the orientation of the boundary with respect to the origin, and is shown on the diagram as the line's normal vector through the origin, referred to as the *weight vector*.

The classification problem corresponds to moving the decision boundary

shifting of a linear function can split them perfectly.

For the case of *neural networks*, of which the single layer perceptron is the simplest case, the bias and weight vector have quite specific interpretations. Figure 4.2 illustrates this: the weight vectors (here in the more general n-dimensional case) are the weights on the input channels (each item to be classified having it's own input channel). The bias corresponds to the threshold of the perceptron, and is not attached to an input.

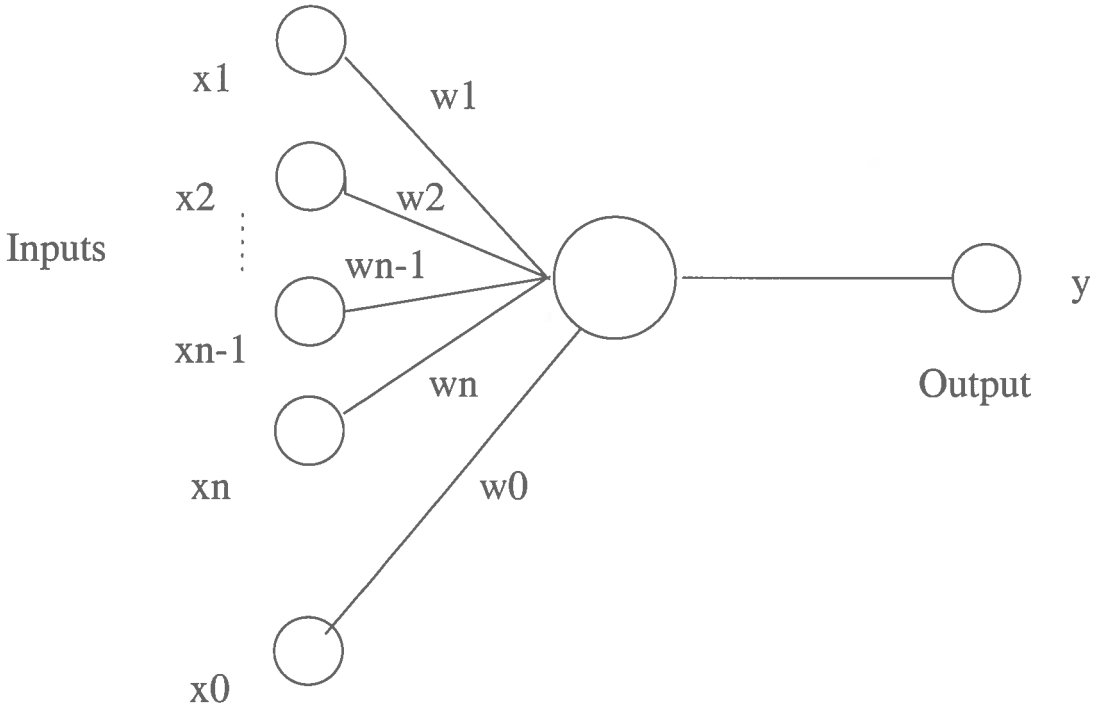


Figure 4.2: Neural net diagram

Thus, altering the weights of the perceptron can be viewed as equivalent to moving the decision boundary in response to each new input. In the supervised learning case, an input is supplied with *a priori* knowledge of which class it belongs to and so what the desired output is. The perceptron incrementally *learns* how to correctly separate the inputs by shifting the

it's desired output, and the weights are altered so as to give this. If the inputs *are* linearly separable, this is not a problem in theory. The process is iterative and the weights are changed incrementally on each iteration in an attempt to give the desired result for all inputs.

A key principle is the *Perceptron Convergence Theorem* which states that, where a solution exists, it will be achieved exactly in a finite number of steps. The output of the perceptron *converges* on the desired output.

An important choice in the training of any neural net is that of the convergence criterion. To state what is a complex mathematical process succinctly, the perceptron is considered to have converged on a certain input-output relationship when this pre-specified boundary condition is satisfied.

Ideally, it is desired that convergence occurs at a point such that the difference between desired and actual output drops below the specified minimum. In reality, however it is not uncommon that the net converges prematurely at a higher error level, and this was indeed true for several models in the current system.

An alternative criterion for learning is specifying a maximum number of iterations and this is the criterion which has been applied for the current system, set at 10^6 iterations (this being the level at which empirical observation suggested no further reduction of error was observed) as the desired error criterion of 0.0001 could not be achieved consistently for all models.

Apart from very simple situations, where the inputs can be directly used and give linear separability, we need to convert an arbitrary range of input values to a simple 'yes or no' classification. The function which achieves this is termed the *activation function.* A sigmoid function is used in the iconic system, which maps the range $[-\infty, \infty]$ to [0,1] and further is monotonic, which preserves the linear nature of the decision boundary.

scores relating to individual primitive feature planes. Conceptually, this can be seen as specifying how significant the result from a particular feature channel is in classifying something as a positive instance of the model in question.

Following on from consideration of the activation function above, it is straightforward to see why the *ideal scores* (i.e. the targets for the output of the perceptron) were originally set as 0 for negative training points and 1 for positive training points.

However, it is apparent that as the sigmoid function maps $[-\infty, \infty]$ to [0,1], the ideal scores and thus zero error would only be produced in the asymptotic case.

It has been suggested [Bishop 95] that more appropriate target levels for such perceptron training are e.g. (0.05, 0.95). This reflects the fact that it is, by the nature of the sigmoidal activation functions, exceptionally difficult to obtain values very close to the idealised 0 or +1 and excessive amounts of work can end up being focussed on achieving them.

The implication is that by such a modification the training of the perceptron may improve sufficiently so that the desired minimum error is achievable. Thus the performance of the system both in correctly identifying features and avoiding false identifications could improve, at least marginally.

This is of special interest as it was observed in previous training sessions that, when trained over a set of six images some models' primitive feature planes' were learned to the desired tolerance (in terms of mean square error) relatively quickly. Others, however did indeed suffer from premature convergence and thus the secondary termination criterion of 10^6 iterations had been imposed.

The danger of such an alternative is that the perceptron would signal recognition over a correspondingly higher range of values and so feature de-

On examining the code in detail, the error-based convergence criterion had been entirely eliminated in favour of across-the-board use of the fixed iteration based version. The criterion to be implemented will be a logical *inclusive OR* of the two, i.e. which ever is achieved first will be accepted as satisfying convergence.

In considering the system's operation, the values chosen to represent the ideal scores must be kept distinct from the matching scores returned during the actual recognition process. Changing the value of the ideal score will not directly affect the performance of the system: it is a parameter of the convergence criterion of the perceptron. As such, the anticipated effects of altering the ideal score values will be

- The perceptron's converging upon a smaller error value.
- Greater generalisation in the model produced. Training to a value very close to 0/1 is likely to give over-fitting to the training models. In this case, the area of n-space defining the possible solutions is confined too closely to that which describes the model set, not allowing 'reasonable' deviations from it. The underlying trend, say the *'eyeness'* of an eye model, would be lost in favour of a function adept at recognising (for example) 'Mike's eye'.

4.2 Implementation of Retraining

The model-making mechanism has been efficiently semi-automated in previous work. The existing training images (Appendix B) were still available as were the other relevant input files. These were:

- A description file with the convergence limits for the perceptron, information on the model details and on the images (*worlds*) to be used

- A transcription file for each world specifying the positive training points for a particular model.

Using these input images, the model maker outputs a model of the specific feature, containing details of the weights to be assigned to the primitive feature planes. The total set of models generated are used as input to the *basemaker*, which collates them into a model base, used for matching by the recognition program.

The first step of the testing, after acquiring the relevant files, was to retrain on the original images, using pre-existing training files and an unchanged perceptron training specification.

The model base thus produced was used as a control to assess the unaltered performance of the system as a comparison for the subsequent retrained versions(Table A.2).

A second training run with the perceptron ideal scores changed to (0.05, 0.95) was executed simply by changing the goal scores in the model maker code, recompiling and rerunning with the same input data.

As the model base specification is one of the inputs to the recognition routines *config* program, the unaltered base was saved as *convic.base*, the retrained as *convic2.base* so that comparative tests could be easily run.

4.3 Assessment of Performance

The initial assessment of performance was comparison of the *control* against the *retrained* model base for recognition of features over both the training set and two untrained images.

1. Number of features identified correctly (positive matches: arbitrary maximum offset of 30 pixels Euclidean distance as per previous evaluations retained).

2. Number of features identified incorrectly (negative matches: offset greater than 30 pixels Euclidean distance).

3. Saccade for initial positive matches.

4. Offset of initial positive matches.

5. Score of initial positive matches.

4.4 Results

The first fact noted was that the change of goal did not significantly affect the mean square error obtained for the model features: in the same cases as before the change, the 'fall back' criterion of 10^6 iterations as the termination condition was required.

Secondly, the change in primitive feature plane weights for the retrained perceptron was quite dramatic. Although the same underlying pattern was discernible, some weights changed by in excess of 100%.

When the model base after retraining was employed to attempt matching of seen and unseen features, the results were disappointing (AppendixA.4, Table A.5). The rate of positive matching was marginally improved for some test images/models, but catastrophically reduced in quality for others. Similarly, the rate of false matches increased more or less across the board in the retrained case.

convergence of the perceptron for certain models at the base making stage. The lack of improvement in mean square error results at this stage suggest that the problem mat not lie in the mechanics of training the perceptron. A possibility which must be considered is that the inputs to the perceptron do not constitute a linearly separable set.

As noted in the perceptron background, if the input set does not constitute such a linearly separable set, no single-layer network of this nature is able to give precise classification between (in this case) a positive model match and a negative result.

The fact that the observed weight changes were so high is further indication that this may be the case. Where such a relatively small change in error criterion gives dramatic effects it is likely that the inputs are of low magnitude, requiring large weightings to give any non-trivial classification results.

These concerns are underlined by the volatility in the matching performance when comparing the system's behaviour when using the new model base with that using the control. This extreme and somewhat inconsistent (although most models were match less well, some did have improved results) behaviour again suggests the equivalent volatility of the features themselves.

If it is the case that the primitives do not in fact give an adequately linearly separable set for positive and negative interest points, this has serious implications for the performance of the system in its current form. One of the great motivations for interest in the area is the supposed relative ease of extracting iconic features from an image as compared with geometric features. To achieve consistent performance over a range of instances, such features need to be strong enough to constitute the basis of a linearly separable input set.

perceptron with alternative matching metrics, which is investigated in the Section 5.7.

A second possibility, addressing the concerns over the features themselves is development of an alternative primitive feature set. One route which is currently being investigated is using a neural net on natural images to learn more efficient feature detection methods [Gomes *et al.* 98].

If it is possible to develop a primitive set which gives stronger, linearly separable results, both the quality of the model base and the efficiency of image/model matching overall would be enhanced.

If such new features are developed, the perceptron retraining between the revised ideals ([0.05,0.95]) should be reconsidered and tested once again alongside the original.

Either independent from or alongside this work, standard approaches to improving perceptron training exist which could be implemented here. Where sufficient data is available, increasing the size of the training set can significantly improve the training of a net and enhance it's generalisation (i.e. the ability to pick out the general characteristics of, say, 'the eye' rather than learning those of a specific instance of an eye).

One further possibility suggests itself from the erratic levels of performance in respect of different test images. If a network is left to learn from a finite test group indefinitely, *over-fitting* may be observed. Here the performance of the net as a discriminator in the general sense as noted above degrades as the net learns too much about the specifics of the test set. The error in recognising re-presented test images would reduce, while that in respect of unseen images would increase.

An easy experiment to run would be to reduce the maximum iteration condition for the perceptron, retrain the model base at this level at repeat the matching experiments.

Chapter 5

Enhancement to Feature Matching Methodology

5.1 Current Matching and Motivation for Change

In its current format, the system evaluates the match between 2 feature planes (image and model) by calculating a score based upon the sum of the element by element correlations over a region around the foveation point which corresponds to the size of the model. In essence, this is the normalised scalar product of the intensities of respective retxels in the two planes.

This measure is somewhat unsubtle in that it cannot evaluate the intuitive similarity that we would wish to be recognised between, say, an image where neighbouring retxels (one each in image and model) were of similar intensities as compared to one where similar intensities had a large spatial separation (Figure 5.1. Ideally, we would want a higher score to be assigned in the 'adjacent' instance relative to the instance where distance is greater.

It must be recalled that, in our $R\Theta$ representation, a match to an adjacent retxel corresponds to an increment in, respectively, scale or rotation. The basic matching process handles scaling and rotation of the model overall but is unsubtle as described above. A more principled matching algorithm could take account of effects which do not apply to the whole model, but are spatially restricted locally in the $R\Theta$ depiction.

Figure 5.1: Intuitively similar images which return a zero score using pixel-wise comparison.

If the image and model are seen as two distributions with their elements being retxel intensities, we would ideally like a score based upon some measure of the distance between the two distributions as a whole rather than simply that between the corresponding elements. There are two alternative metrics proposed to give this sort of measurement: one based upon the *Earth Movers Distance (EMD)* and one based upon a Gaussian blurring/scalar product combination.

5.2 Earth Movers Distance (EMD)

The *Earth Movers Distance (EMD)* [Rubner *et al.* 98] takes into account similarities in structure between two distributions and can indeed constitute a metric between them overall. The concept of the EMD derives from a special case of linear programming called the *transportation problem* which can be solved using the *simplex method.*

5.2.1 Linear Programming

Linear programming is a widely used approach to optimising a linear function (termed the *objective function*) in N independent variables subject to M linear constraints.

Without developing in full the underlying theory, which is well documented elsewhere [Hiller & Lieberman 74] [Dantzig 51], the basic concepts can be considered by viewing the function and constraints to be mapped

ematical proof, it is intuitively clear that, in the case where there is a finite solution (i.e.the constraints enclose a finite region which, by virtue of the linearity of the problem will be a convex polygon), this solution will occur at a hyper-plane corresponding the constraint limits. This follows on from consideration that for a linear function, the gradient is never zero and so no maxima or minima can exist within the polygon's boundary.

Thus, in the 2D case, each of the *feasible solutions* (or *feasible vectors* can be found on one of the edges of the polygon formed by the constraints.

Considering the diagram, Fig. 5.2, where we are maximising the result, we need to be moving through the space in a direction perpendicular to the objective function contours, and away from the origin.

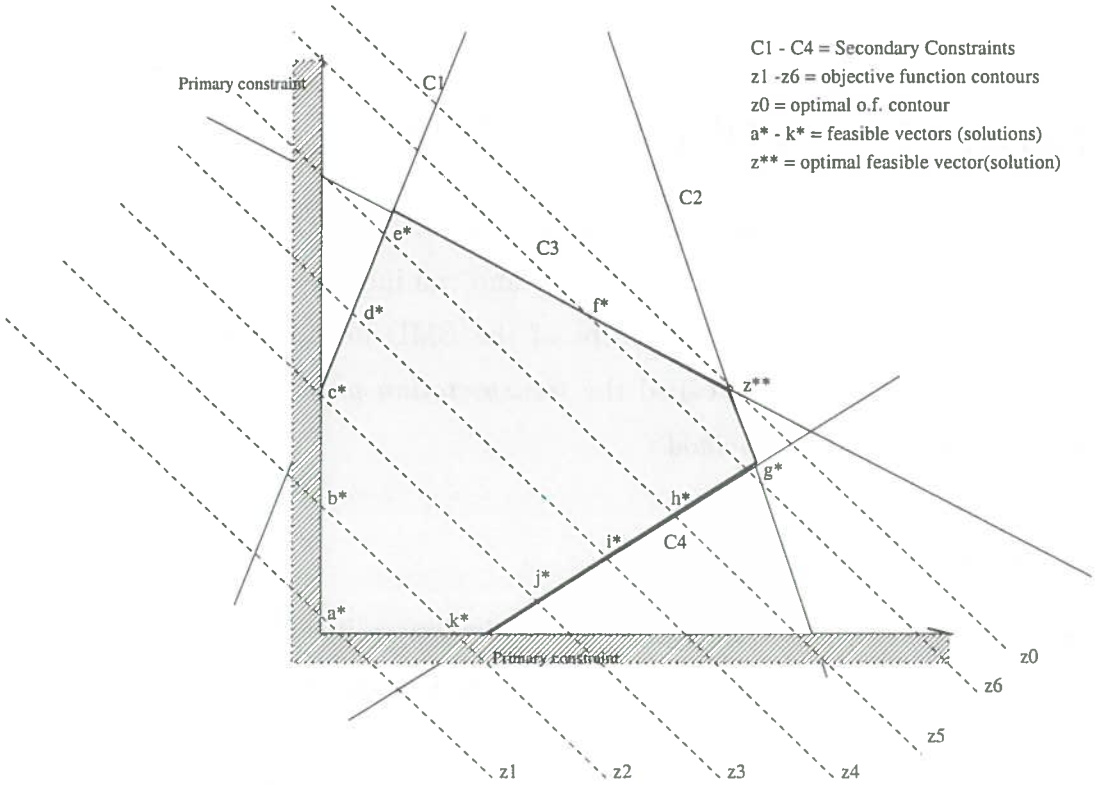


Figure 5.2: Graphic depiction of linear programming problem in 2D.

a combinatorial problem: that of discerning the vertex in question, which is equivalent to ascertaining which of the constraints should be satisfied in the optimal case. The *Simplex Method* is the standard approach taken to solve this reduced problem.

5.2.2 The Simplex Method

In brief outline, the Simplex method can be summarised as follows:

- Convert all equalities in the constraints to equalities by adding a *slack variable* to each of the inequality constraints.
- Add an *artificial variable* to each constraint equation to put the problem into what is termed *restricted normal form* i.e. where each constraint is an equality with at least one variable which has a positive coefficient appearing uniquely in that constraint alone. The motivation for this conversion is that a simple solution exists for this special form of problem.
- Solve the new problem thus created (*the auxiliary objective function*) to give the initial feasible vector for the original problem.
- The original problem is now solved from this initial feasible solution by an iterative process during each step of which the current solution is incrementally improved.

The objective function and constraints are written in the form of a *tableau* with columns corresponding to the coefficients of the respective variables and rows corresponding to the constraint equations (row one is the objective function itself). By an iterative process of matrix manipulation, a series of combinations of the constraints to be satisfied is tried.

The simplex method has the key characteristic that the objective function

The format of the solution methodology is well-suited to computer implementation.

5.2.3 The *Pure* Transportation Problem

The original formulation of the transportation problem was the optimisation of the flow of resources between suppliers and consumers within a network. The cost of transportation is expressed as a function of the amount of resource to be transported and the distance it must travel (i.e. a definition of the work associated with the transferral). The desire is to minimise the cost in terms of amount moved, how far and a cost per unit mass per unit distance.

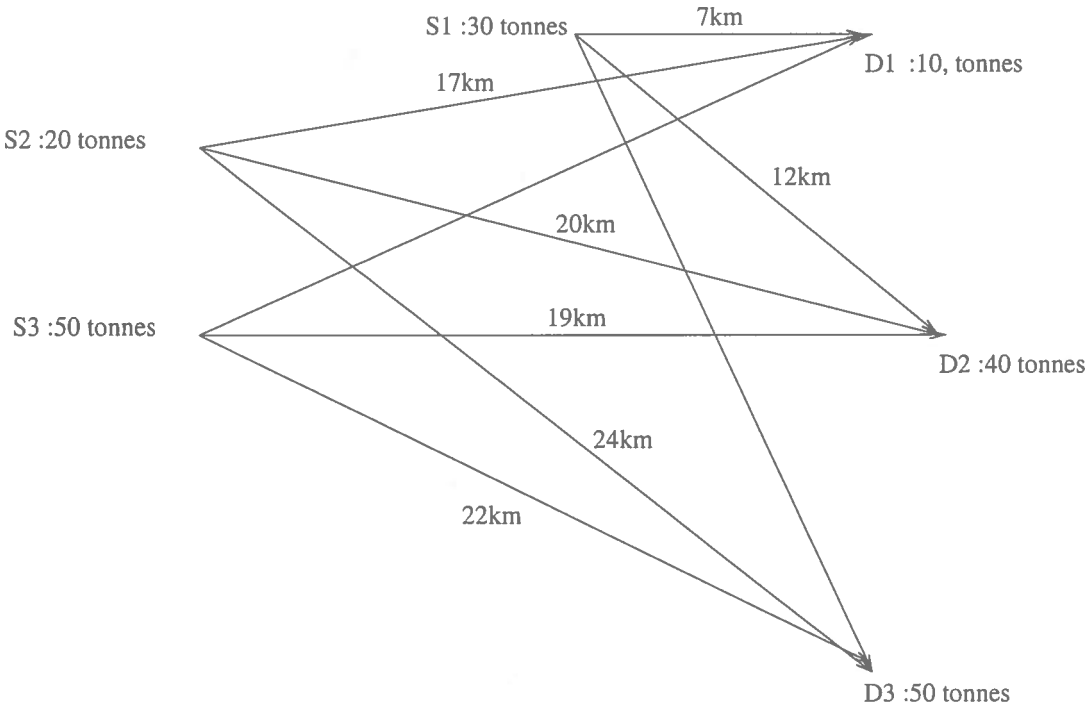


Figure 5.3: Example of a standard transportation problem

tuting the supply side, holes to be filled constituting the consumers. In this case, the cost to be minimised is simply the work done in moving earth.

To apply this to the problem at hand, i.e the distance between two feature planes we consider the retxels in one plane to be the suppliers (piles of earth with a magnitude equal to the intensity of feature response) and the retxels in the other plane to be the consumers (holes of capacity equal to the intensity of feature response). Thus, the work done will be the 'amount of intensity' transferred multiplied by the distance over which it is moved (Figure 5.4).

If we find the minimum work done for 'filling the holes', i.e. optimise the

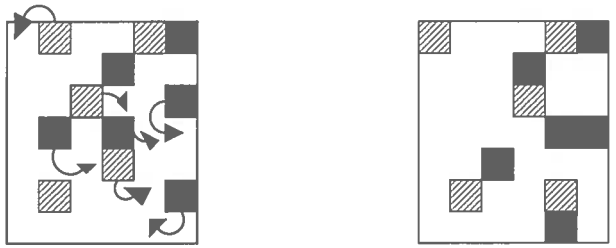


Figure 5.4: Illustration of a matching problem in terms of the transportation problem. The differently shaded squares represent retxels of different intensities. The arrows indicate possible optimum 'movements' for matching with intensity transfer minimised.

cost we have specified as a measure of the similarity of the two planes as a distribution of retxels, rather than a simple summing of the retxel-wise differences. This is the EMD between the two feature planes.

As this is a special case of linear programming, the *simplex* method may be used to optimise the cost. The transportation problem gives a matrix of constraint coefficients which have a distinctive patterned structure, which can be taken advantage of to achieve computational savings.

The shaded regions on the diagram (Fig. 5.5) will be units in the matrix identifying allowed transitions. All other entries are zero, indicating a non-

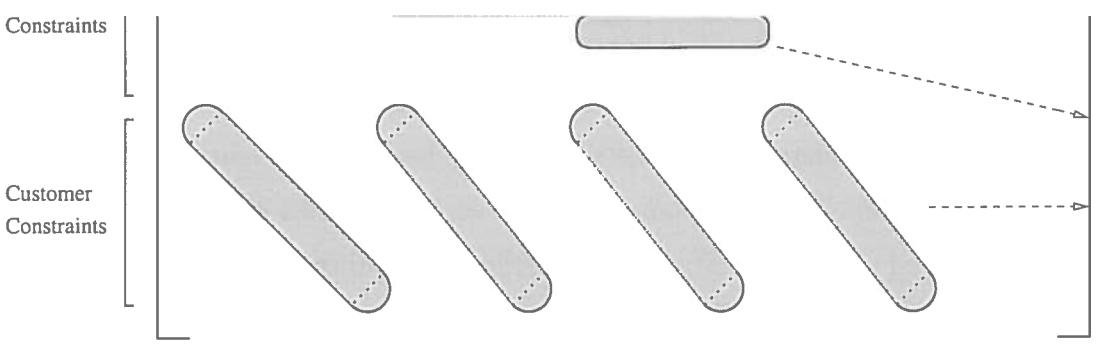


Figure 5.5: Section of a tableau specifying a transportation problem.

for solution by making the first column indicate, in order, the amounts to go from sources, then those to arrive at destinations.

The first row is equivalent to the objective function and each element will give the cost per unit of the transition indicated by the two units in the corresponding column.

Setting up the table in this way is equivalent to specifying a formal linear programming problem the optimisation of which will ensure we have found the best solution for the transportation problem so defined.

The version of the TP and thus the linear program to be solved differs in detail in our application, but the pure TP problem is as follows [Rubner *et al.* 97]:
Let $\mathbf{p}$ and $\mathbf{q}$ be the vector representations of the two feature planes to be compared, such that:

$$\mathbf{p} = (p_1, w_{p_1}), (p_2, w_{p_2}), ..., (p_m, w_{p_m}) \tag{5.1}$$

$$\mathbf{q} = (q_1, w_{q_1}), (q_2, w_{q_2}), ..., (q_m, w_{q_m}) \tag{5.2}$$

where $(p_i), (q_j)$ are points in the image, w_{p_j}, w_{q_j} are their corresponding in-

where C_{ij} is the element of the $m*n$ constraint coefficient matrix $\mathbf{C}$ which gives the amount of 'intensity transfer' between p_i and q_j.
$\| p_i - q_j \|$ is the element-wise distance between p_i and q_j.

The constraints imposed in the basic transportation problem are:

$$C_{ij} \geq 0, (1 \leq i \leq m), (1 \leq j \leq n) \tag{5.4}$$

(Supplies may only be made in one direction)

$$\sum_{i=1}^{m} C_{ij} = w_{q_j}, (1 \leq j \leq n) \tag{5.5}$$

(All consumers must be satisfied in full)

$$\sum_{j=1}^{n} C_{ij} \leq w_{p_i}, (1 \leq i \leq m) \tag{5.6}$$

(Suppliers may not deliver more than their total holdings)

$$\sum_{j=1}^{n} w_{q_j} \leq \sum_{i=1}^{m} w_{p_i} \tag{5.7}$$

(Feasibility condition: total demand does not exceed supply)

$$\sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} = \mathbf{min}(w_{\mathbf{p}}, w_{\mathbf{q}}) \tag{5.8}$$

(Optimisation condition) where

$$\mathbf{w_p} = \sum_{i=1}^{m} w_{p_i} \mathbf{w_q} = \sum_{j=1}^{n} w_{q_j} \tag{5.9}$$

Here the EMD is defined as:

$$EMD(\mathbf{p}, \mathbf{q}) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} \| p_i - q_j \|}{\sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij}} \tag{5.10}$$

$$\sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} \| p_i - q_j \|$$

Where we need to ensure that the feasibility condition is satisfied one option would be to perform an initial pass to ascertain which of the feature planes has the largest sum of intensity over all points. During this pass, an initial comparison could be made to assign the identity of supplier and customer. If this pass indicated that the overall difference was above a pre-specified tolerance, the two distributions could not constitute a good match and the loop could be terminated.

Further, the lower bound on the EMD is the distance between the centroids of the two feature planes. Thus, calculating this to be above the proscribed maximum would again reduce the search required.

If we desire to give a measure which is *strictly* a metric between two distributions, we must change the constraints to demand *equality* between overall supply and demand, which in reality will not usually be the case. One solution is to include a *dummy destination* where the excess production from the supplier is to be delivered. The cost of sending to this dummy would be set at a high value to make it the option of last resort for transitions. This is a standard approach in traditional TP applications where, for example, production is unlikely to exactly match demand and the critical factor is that for all destinations demand **must** be met.

This approach could be applied here, but if we desire to retain the special structure of the TP tableau (to allow us to use the computational savings this gives) it would be necessary to include dummies for each source and destination, increasing the number of columns in the tableau drastically.

Thus, our first proposed approach was to conduct a normalisation after the initial pass so that the overall sum of retxel intensities was equal. This would ensure that the *simplex* program need not be modified further and would avoid the need consider the best assignment of significance to the amount of overall 'slack' in intensities (varied by setting the cost of trans-

Unlike in the original application of the EMD to image recognition, where comparison was in terms of similarity of colour distributions between two images, for our purposes the distance by which an individual intensity element is translated in calculating a score is far from a negligible factor.

In the pure EMD, it would be quite permissible to have, say the last intensity amount translated from one corner of the range to the opposite corner, albeit at a potentially huge cost (dependent on the amount moved as well as the distance). For our purposes, this would not have a great significance to the feature matching process and so should be proscribed.

However, any restriction on the allowed range would again result in losing the transportation problem format which we desire to keep. Thus, the initial approach was to allow such transitions but to apply a restrictive cost on all destinations outside the specified permissible range in a way similar to that proposed for dummy variables. Transfers out of range correspond to a 'no-match' and a transfer to/from a source/sink.

Applying this restriction means that the score will not correspond to a true metric between the two distributions, but for our purposes, this is not of primary significance. With some thought, it can be seen that appropriate choice of the cost of a transfer with source/sink to reflect the weight we wish to associate with an unmatched element of intensity could in fact enhance our measure further.

Setting of a maximum distance that it is reasonable to allow transportation is initially somewhat arbitrary and must be considered as a variable potentially open to reassessment, as is the cost of a 'no-match'.

This is a novel development of the EMD approach, constituting as it does a constrained optimisation method. As is developed in Section 5.4.2, the detailed design and implementation process revealed unanticipated corollaries of the new methods which eventually required substantial reevaluation and

In contrast to the rather complex and mathematical basis to the EMD approach, the blurring route is in conception quite simple. All that is required is that one of the model or the image is probabilistically blurred over a restricted range of neighbouring retxels and then the original scalar product, element-by-element correlation carried out on the result.

What we desire is a radially symmetric function which will achieve a weighted averaging of the local intensities centred on the current retxel position. This would modify the output of the system by smoothing out the structure of the matching response - what we hope to see is that the 'basin of attraction' around a feature which specifies the area which will lead to a positive match obtains a more regular structure, conducive to more efficient saccading behaviour.

The initial choice of function is a simple Gaussian of the form:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{x^2+y^2}{2\sigma^2}} \tag{5.12}$$

which although only approaching zero asymptotically, can be approximated by a finite 2D operator. We use convolution of the retxel array with one of three kernels as illustrated in Figure 5.6 The process of applying this to

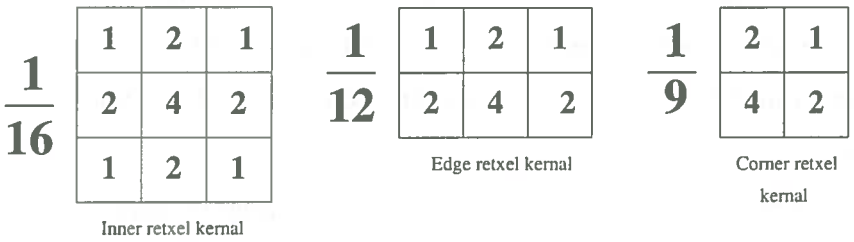


Figure 5.6: Kernels for Gaussian blurring

an image/model is extremely simple both in conception and implementation. *Convolution* the mathematical operation equivalent to multiplying two arrays together to give a result array is carried out. It is performed by passing the

For each position, a retxel value is output which corresponds to multiplying together the values of aligned cells and summing these together. Note that the kernels are all scaled to an overall total of '1', so that the total intensity of the input image is not altered by the operation.

Applying the kernels specified is equivalent to blurring using a Gaussian with mean zero and standard deviation (σ) of one retxel. The significance of the size of σ here is again the trade off between successful recognition and maintaining discriminatory abilities. The area of blurring should roughly correspond to that of translation allowed in the EMD to assist comparison of the methodologies themselves.

As in the allowing of transitions in the EMD approach, the blurring provides flexibility in the range of scale and rotation allowed for matching at a local level.

As noted, blurring could be applied to model or image or indeed both. For a simple application of blurring such as this, we are not attaching any significance to the two as distributions at the detailed level, and so a choice of a single blurring, reducing required operations as it does, is appropriate.

As the model creation is handled just once for each model base and takes place prior to the matching procedure, this is the best candidate for applying the blurring. This choice is purely based on efficiency considerations and the blurring could just as well take place on each new image, while using the original unblurred model base.

In the structure of the program, the blurring will be a member function of the Image $R\Theta$ class in either case, the only difference will be at which point the function is to be called.

is a two-stage process. First, a routine to implement the *simplex* method is needed to solve the linear programming problem outlined above. Second, a routine to pass the retxel intensity/distance details from the main program to the simplex routine.

5.4.1 Basic EMD Formulation

The simplex method has a standard algorithm which is based upon the implementation of Kueni, Tschach and Zehnder. The initial intention was to use the proprietary C recipe in [Press *et al.* 89], modified to work with the C++ code of the main program. The process of debugging the code to a runnable form was greatly aided by the assistance of Professor S.A. Teukolsky of Cornell university, one of the original authors who provided an updated version of the basic code in response to some conceptual queries

The first stage of testing was carried out upon the basic simplex routine, as a stand-alone program on basic linear programming problems and the results were checked to independently derived answers (Appendix A.5).

After the routine was ascertained to be acting correctly for such problems, a range of simple transportation problem tests were carried out. These were designed specifically to address the performance of the system from the most basic like/like problem to schemes of arbitrary complexity.

The final stage of coding required the construction of an array corresponding to the tableau elements required as input to the simplex routine, derived from the image and models for the current saccade of the recognition program.

The simplex routine takes as its primary input a pointer to memory allocated to a 2-D array of *float* values. The array constitutes the required elements of the simplex tableau and as such encodes the objective function and the constraints which apply.

Two one dimensional arrays were also required for the purpose of indexing

As noted above, the pure EMD/transportation problem allows for transitions between all sources and destinations, this being one of the factors which contributes to the measure being a true metric of the distance between two distributions. In the previous papers on its application to image processing, this was appropriate as the problem was of matching colour distributions: images were sorted according to colour composition.

The spatial distribution of elements within an image was not considered. Further, quantisation of the colour distributions by a novel method ensured that the size of the transportation problem tableau to be solved was reasonable.

When considering the size of tableau required to translate iconic object matching into a Transportation Problem, it became clear that the size of tableau required would be simply infeasible in practical terms. Each retxel in the image and the model sections being matched would require a number of allowed transitions specified in the tableau equal to the total number of retxels in each.

Thus, if we have image/model each of 'n' retxels, the size of the tableau would be of the order of $(n^2 + 1)(2n + 1)$: the columns would each correspond to a translation between two specific retxels, the rows would specify the split of intensity value in each retxel (i.e. for sources, it's destination, for destinations, their sources). The allocated memory must allow references in the range:

$\mathbf{a[i][k]}$, $i = 1 \ldots n^2 + 2, k = 1 \ldots (2n + 1)$

to allow for internal referencing needs.

With n of the order of 1000, this is clearly an unwieldy approach. Also, from the point of view of our problem, it is simply not meaningful to allow transitions to be to every point, as they are to correspond to local matching

matched with one at the bottom left of the model would not be useful.

It would be possible to impose a punitive cost on transitions as they became inappropriate, leaving the full range of transitions available, albeit the further ones being very costly. Here,the Transportation Problem tableau's unique format would be preserved but this contributes nothing to the solution while exacting a great computational cost as a full size tableau is required.

The solution proposed is to implement a novel *constrained EMD* where a region of nearest neighbours for a source/destination is specified, outside which transitions are disallowed. This is in some ways similar to the approach of blurring the model or image before matching, allowing as it does a wider range of matching than a pixel-by-pixel correlation measure, but no longer attempting to give a distance between the two distributions themselves.

The most basic approach to implementing a restriction on allowed transitions would be to simply reduce the number of **'1'** entries, corresponding to permissible moves, in the body of the tableau. This would result in a sparse matrix of the same size and would thus not benefit from the possible computational savings.

The intuitive extension of this line of thinking is to consider a regular region around a source retxel giving sources to which translation is allowed. It is only necessary to allow enough column entries for a source to specify this region in full. A partial TP tableau format for the sources would be preserved, albeit for fewer columns for each. *Within* the regions thus delineated both specification of the objective function in terms of regularly varying distances and that of allowed destinations in the 'customer rows' would be preserved. The key difference in the format of the table would be that discontinuities would be apparent in the structure of allowed destinations *between* these regions, corresponding to all retxels too far distant to be considered suitable candidates for matching. Implementing the problem in this manner would require the tableau size to be of the order:

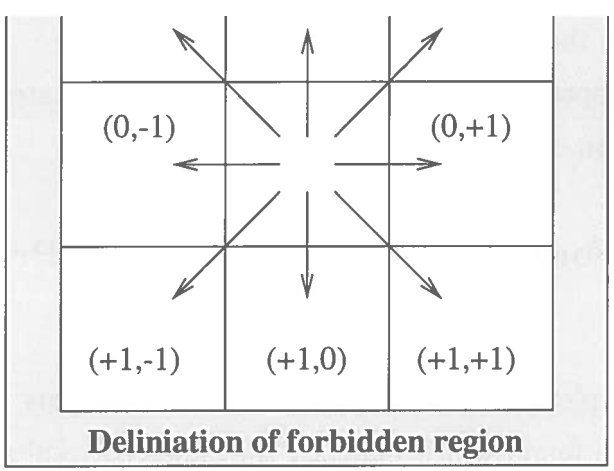


Figure 5.7: Allowed region of model retxels for matching an image retxel

a[i][k], $i = 1 \ldots (n((r\sqrt{n}) + 1)) + 2, k = 1 \ldots (2n + 1)$

where r is the number of retxels over which translation is permitted.

It is possible however to restate the problem in a more intelligent way which constitutes a novel constrained linear optimisation approach, at the cost of sacrificing the unique TP structure and thus possible computation advantages of a streamlined *simplex* approach.

In this approach, the source retxel is specified in the tableau in a similar manner, with r^2 possible translations allowed for as columns of the tableau. Thus, the tableau size can be specified simply, and the source rows as in the modified form suggested above.

When the specific destinations that each column corresponds to are considered, however, a more considered approach is appropriate. Each source is considered column by column in the light of the pre-specified allowed range. With careful specification, it is easy to consider *only* the allowed destinations, with a **1** corresponding to the transition entered in the appropriate destination row. Similarly and as a part of the same process, the objective

which will drastically reduce the degree of computation required.

This more intelligent adaptation of the EMD to a 2-D image matching problem is a novel application of the approach.

5.4.3 Broken TP Formats and Normalisation Problems

The above approach was implemented and tested on simple problems with good results - solutions were found which matched the expected results for hand-crafted problems and for small image matching problems.

As testing progressed, however, it became clear that the modifications to give a more intelligent action had given major undesirable side-effects too. In the original application of the EMD metric to this problem, the unique format of the Transportation Problem had been preserved.

One of the characteristics of this structure was that if, *globally,* the total amount to be shifted from the sources was equal to the capacity of the destinations, a solution was guaranteed. This condition was easily achieved by normalising the total of image intensities to that of model intensities. Some further development was to be applied to give a measure of the amount of normalisation carried out, as this should have been taken into account in evaluating the match score.

With the change to a spatially constrained version of the problem, however, this condition could no longer guarantee a finite solution existing. In essence, to guarantee a solution in the same manner, a normalisation would have to be carried at the level of the allowed transition range of destinations. *Prima facie* this would, at a minimum, involve a great additional cost in terms of computations, and when examined in more detail may present inherent conceptual difficulties.

Consider the basic unit of normalisation in this approach: each 'source'

This would need to be ascertained for each image retxel and *then* a similar series of normalisations applied for each model retxel with respect to the allowed range in the image.

Just this normalisation process itself would require a pass through the whole tableau, with successive normalisation constraints being applied throughout. The actual form of the normalisation, and the interpretation of the results *post facto* would present further significant problems.

In short, the normalisation approach in the constrained case is considered to be too costly in terms of computation and would necessitate modifications to the evaluation of dubious applicability.

5.4.4 Return to Simplex: Dummy Sources, Destinations and their Capacities

To solve the significant problems besetting the approach at this stage, four options were considered:

- Abandon the constrained approach entirely and revert to the full Transportation Problem format. Experiments with the routine at this stage for the maximum encountered image/model size of 48*15 retxels suggested this would not be appropriate. For the constrained problem a run on one image plane of this size took the order of three hours (in the serial implementation, on the fastest machine available for a single point) to complete. Any change likely to increase this by several orders of magnitude is not a serious option.

- Adopt the range-wise normalisation approach discussed above. For the reasons outlined, this approach did not seem promising.

- Modify the problem itself by returning to the idea of dummy sources

a suitable capacity to the constraints specifying the amount to come/go from/to the dummy source/destination. This is simple in conception, requiring just the separate summing of deficits or surpluses and their accumulation as the required dummy intensities.

To take account of the possibility of surpluses or deficits for a retxel, both source and destination dummies would require a positive and a negative version. Where there is a surplus for a source, the positive dummy destination would be incremented, where there is a deficit, the negative dummy destination would be incremented.

As it is a condition of the *simplex* formulation that the intensity be a positive amount, the distinction between positive and negative dummies would be the sign of the transition indicated, being the negative of the normal element where a negative dummy is appropriate (See Figure 5.8. Although this approach would satisfy the condition for a feasible

	X1	X2
S1	1	1
D1	1	
D2		-1

Figure 5.8: Illustration of a tableau entries for a positive (D1) and a negative destination(D2), given that normal entries are '1'

solution, it again requires an initial pass and an additional calculation for each retxel during this. The array would only require four additional

formulation.

- Moving away still further from the precise Transportation Problem format is all that was required for a simpler and more elegant solution to the problem. It is useful to take a step back from the EMD when considering the constrained problem at hand. If considered once more purely as a general linear programming problem the possibility arises of using methods already well established by their use in the *simplex* method. This approach is developed in detail below.

5.4.5 The Solution: A Constrained Pseudo-EMD Metric with Slack Dummy Variables

Essentially, we shift away from specifying in advance that each source must be have its supply capacity exactly equalled by the demand of its allowed destinations. This follows from not demanding normalisation at the level of compared 'allowed transition' windows. In terms of LP/*simplex* problems this is equivalent to saying that we no longer have constraints in the form of equalities to be satisfied, rather we have *inequality* constraints once again.

The solution adopted in respect of this in the Simplex approach is to introduce *slack variables*, one for each inequality constraint. By allowing such a variable *to be not subject to any constraints*, in essence we convert the constraint back to an equality. If we allot one such a variable for each row in our tableau, we return to having a finite solution.

The interpretation of these slack variables in our problem is once again as dummy sources/destinations. Here we have one for each real source and destination. The key difference between this and the former proposed solution is the realisation that these dummies need not be subject to any explicit constraints as to the amount they supply or receive. The dummies act as

ascertain the capacity demanded of our dummies.

The key to using such variables as used in the *simplex* method is the *Big M Method*. This refers to the practice of assigning a large value to the cost of transitions with respect to these dummy variables relative to transitions to fellow real sources/destinations. In the *simplex* method however, the precise value used for M is not significant, so long as it is large enough to make such transitions the 'last resort'.

In our application of the method, there is a definite significance to the value chosen for the dummies. To return to the overall context of the problem, our desire is to construct a useful metric for the evaluation of similarity between two images, treated here as defining the sources and destinations for shifting elements of intensity. Thus transitions between real sources and destinations are to be costed at values equivalent to a measure of the distance between the two elements in question.

The cost of matching an element of intensity with a dummy source/destination must then be defined in terms of the cost of having such an element not match with *any* corresponding fellow in its opposite. The choice of a precise value to ascribe to this must reflect our assessment of the reduction in matching score we wish to be associated with such an instance.

5.4.6 Counting the Cost and Interpreting the Results

After developing the mechanics of a new metric such as this, the vexed question of setting values to certain subjective evaluation measures arises. The iconic system itself has several such set at 'arbitrary' values initially intuitive and validated by empirical testing.

Firstly, initial test runs on single primitive planes showed that the Simplex calculation, even with the reductions in computation following from introducing constraints takes substantially longer than the previous scalar product

individual ranges either. Thus we can set a value of global dissimilarity above which a match score of zero may be safely returned. In the first instance this value has been set so that a no-match is reported without further calculation whenever one intensity sum is double more that of its comparator.

To match with the evaluation settings extant in the Iconic system itself, the scoring of results generated by our new metric should occur within the range [0 - 1]. The results of the Simplex routine used however have a range starting at a minimum cost of zero. This equates with no intensity elements having to be moved at all and so is the perfect match which is to be restated at a score of 1.

The other extreme is a maximum cost (i.e. a poor match/no match) equivalent to the cost of moving every intensity element to the maximum of its allowed range. This will be further complicated by the, as yet unspecified, cost of an intensity element not having a match within its allowed range at all.

In the current variant of the approach, the allowed range is to be restricted to a distance of one pixel from the original point. The distance measure is set as just the number of pixels and so the maximum cost of moving each intensity element will be '1'. So, the maximum total cost of matching two intensity distributions with identical global sums will be just the sum of either distribution. In this case, it would be appropriate to assign this to represent a no-match, i.e. a zero match score.

Where the two distributions do not have identical global sums, the attribution of a no-match to a particular score is more difficult. We know from our prior decision that the bigger intensity sum will not exceed double that of its opponent. Two options suggest themselves for consideration:

- Set zero match as the cost of moving all elements in the larger distribution to the farthest extent of their allowed range. The question here

- Set zero match at the cost of moving all elements in the smaller distribution to the farthest extent of their allowed range. One potential problem with this approach is that the match score could be over-sensitive to small variations if the two distributions are relatively sparse.

  Consider the case where one distribution has five intensity elements, the other but three. Here, a zero match would be implied if the total cost exceeded two then, (by the comparative sum criterion noted above) as two pixels have no match, a zero score would be recorded. To take the situation where all that changes is that the second distribution has four pixels, the cost would now be one and the metric would record a score of 0.5.

Despite its recognised problems, the method adopted is the latter of the two. The motivation for this is to give a good idea of the ability of the metric compared to the scalar product variant which employed a similar scoring, with just these limitations. A more subtle scoring is a suitable direction for further work.

Once a cost has been identified with a zero match, the decision as to the cost of making a transition to a dummy source/destination is greatly simplified. We need to specify how many intensity elements at the level of local matching (i.e. within an elements allowed range) we are willing to see go unmatched before recording a zero match score.

As our allowed transitions are currently set to each evoke a cost of '1', once we have decided on the fraction unmatched to equate with overall match failure, we need but set the cost of a transition to a dummy to be the reciprocal of this fraction. For consistency with the rejection criterion for nil matching at the global level, setting the cost of a transfer to a dummy to '2' ensures that as soon as half of the intensity elements go unmatched, a zero

as a stand-alone version initially, for basic testing. The kernels were specified as noted in 5.3 and applied to a2D array from an input file. An array of the same size was constructed to receive the results from the convolution process, which was in the first instance printed to standard output.

After testing on simple arrays, a sample array was dumped to file from the iconic program and blurring applied to check there were no problems at this level.

It was then a matter of altering the routine to receive an *Image RΘ* as input (essentially just an array of integers) and to return the blurred version as output to the calling function. Care had to be taken as we need to specify the kernels as floating point **double** values whereas all retxels are specified as **short integers**. A cast from one to the other is needed, but if carried out inappropriately will cause disastrous loss of accuracy. The integer values are cast to be floating point numbers before convolution, the results passed back are then recast to integers.

The functions for obtaining and returning pixel values already exist in the Class *Image RΘ*. Making the modified **Blur()** routine a member function of this class meant both that these functions were available and that **Blur()** could be simply called on any *Image RΘ* object.

The decision between blurring image or model will now be that between calling **Blur()** during either the model making (on the average model over all training images) or during matching on the image itself. The original plan was to blur models as this would simply reduce the number of computations required during matching itself. At the last minute, however there was a problem with the functioning of the model making mechanism, and so to avoid delays, the blurring is currently implemented on the images. The only anticipated change here is an increase in running time during matching.

The actual matching mechanism is left as in the original design, i.e. a version of element-wise scalar product. Prior to matching, however, the

5.6.1 Implementation Testing

Simplex Routine

To investigate correct functioning of the new matching methodologies, several standard linear programming problems with known answers were run through the simplex routine. The results generated were simply checked with the expected answers to ensure the basic operation was correct. Selected examples of the testing results are given in Appendix A.5.

Constrained EMD: Tableau Construction

This routine is the vital linkage between the simplex routine and the bulk of the iconic system. It takes the $R\Theta$ representation images of object and model as the source and destination amounts respectively and inserts them as the first elements in a simplex tableau.

From the number of retxels in these which, as noted above depends on the model being matched, the constrained TP tableau is constructed with the Objective Function corresponding to distances and the elements corresponding to allowed transitions.

Initial testing was by a variant of the routine which output the tableau to file. For small problems, specifically here a 3x3 array, it is feasible to manually calculate the tableau structure required for the constrained problem. Such tableau were compared with the automatically generated versions to ascertain the correct overall functioning of this aspect.

The array constructed was output to file and visually compared with the desired structure to give confidence as to the routine's correct functioning. An example of the output is noted in Appendix A.6.

with the expected structure of output and it's successful generation of the anticipated results from this.

The format which the iconic system will provide is integer arrays of sizes up to 48x20 retxels. The current matching routine uses functions of the iconic *Image Rθ* class to obtain the size of the image and model arrays, then to read the retxel intensities of each. Thus testing is directed to input files consisting of this information: specifically two integers defining the size of an integer array followed by the values of the two arrays as an integer string.

Tests were run on the following:

1. Identical model and image each consisting of a 2x2 array with a single non-zero element.

2. Model and image each consisting of a 2x2 array with a single non-zero element relatively displaced by one cell.

3. Model and image each consisting of a 2x2 array, one with a single non-zero element, the other with two half intensity non-zero elements, both relatively displaced by one cell.

4. Repeat of the above tests with intensities no longer equal.

5. Repeat of the above tests with many non-zero elements in each array.

6. Repeat of the above tests with altered array sizes.

The simple hand-crafted arrays were fed in from file, with the optimised score as output. The problems were all solved manually in advance. Where results differed, rechecking showed the manual calculation to be in error, giving good confidence as to the correct functioning of the routines.

Constrained EMD with Dummy Variables

The results of tests when using the initial version of the metric on some arrays

constructor to assign a dummy variable for each retxel (i.e. for each row) This required modifying the constructor routine to add 'm' extra columns, where 'm' is the number of retxels in image and model combined, i.e the number of rows in the tableau.

The construction of these rows is quite simple, in effect constituting adding a square 'identity' matrix to the existing tableau, i.e. with '1's in the leading diagonal, zeros elsewhere. The first row is the exception to this, as the objective function for each column in this zone is set to '2' (see above). The constructor was tested by once again running the system set to output the constructed tableau and visually inspecting the expected regular structure.

The matching tests were then repeated for the revised structure and solutions successfully obtained in all cases.

The Big Picture: Large Image Testing

Once it was confirmed that the new metric was performing as intended on simple image-style arrays, a test was needed to check the behaviour given on presentation of simulated image arrays on the scale of those generated by the iconic system.

The first step was to test two fabricated arrays of equal global intensity sum. A small program was constructed which firstly produced an array of random numbers, with the terminal column set as all zeros. A second array was then constructed by moving all of the numbers one step to the right and replacing the first column with zero values.

The expected score of matching the first array with the second is the result of taking the difference between the arrays, pixel by pixel (so removing consideration of elements which match to themselves with no cost). As we restrict pixel movement to a region with cost per element moved of '1', the

should be shifted to dummies at a cost equal to (with transition cost set at '2' as specified above) double the total noise intensity.

It was during this testing, immediately prior to planned integration with the iconic system that a practical implementation problem crystallised. The operation of the *simplex* routine on the constrained problem for array sizes of orders up to 10x10 had been investigated in the initial testing and had given results with no perceptible time-lag.

It was foreseen that the pure EMD approach would be too computationally intensive to be appropriate for real-time use our application. Indeed one of the primary motivations behind adopting the constrained format in the first place was to reduce the time cost of the calculation.

The random variable arrays to be tested were of size 24x24. The run time on a single processor was in excess of one hour. This immediately caused concern as the primitive image planes of the iconic system can currently reach scales of up to 48x10 elements.

A test run on one such plane dumped from the iconic program indeed took of the order of three hours to reach a solution. As each micro-saccade involves the matching of 42 image planes, 14 of which are of this size, it was very clear that the current form of the metric was not suitable for real time use. Indeed, with the time constraints on producing results at the point of this development, it became clear that it would not be feasible to generate a full set of results with the EMD integrated with the iconic system.

5.6.2 Assessment of Performance- Single Primitive Feature Plane Testing

To achieve some measure of the comparative performance of the three metrics, a new approach was devised with dramatically reduced computational requirements.

instance of an eye (point [135,236] in image *c1.ppm*) for each of the three metrics.

This is useful firstly in simply comparing the strength of match between the metrics. The test can be expanded, however to give useful information for each individual metric by mapping out the spatial variation in match score over the area surrounding the known positive instance.

It is desirable that a metric should achieve a distinct and regular 'basin of attraction' in the region of a positive match: if the good match is too spatially restricted, it will be difficult to detect. We would like the system to 'fall into' each positive match with a high probability over a wide area.

Results were taken using each metric over a 5x5 square region surrounding the positive match point. The detailed methodology was:

- The iconic system was modified to output to file both image and model feature plane for each instance of an attempted match, along with the model name and array size.
- The system was run with no micro-saccading, rotation or scaling at each point in the 5x5 region in turn.
- The same chosen plane arrays were extracted from the file and set aside as input for the next stage.
- Each metric routine was modified as needed to accept this as input and to return the match score for a point.
- The match scores were normalised manually to assist comparison of the results.

5.6.3 Assessment of Performance - Blurred Metric

Due to the discoveries noted, the assessment of performance when integrated

1. Number of features identified correctly (positive matches: arbitrary maximum offset of 30 pixels Euclidean distance as per previous evaluations retained).

2. Number of features identified incorrectly (negative matches: offset greater than 30 pixels Euclidean distance).

3. Saccade for initial positive matches.

4. Offset of initial positive matches.

5. Score of initial positive matches.

To comprehensively investigate all functional variants of the system, these tests were run using both the original and the the *retrained* model base for recognition of features over both the training set and two untrained images.

5.7 Results

For the single primitive feature plane testing the results are noted in table form in Appendix A.7, in addition the model plane matched at each point is given for reference. The results are best viewed in graphical format, as shown in Figures 5.9 5.10 and 5.11.

As the results are for a single, out-of-context feature plane, it is not too worrying that the basins of attraction are not centred precisely on the positive feature instance: relative comparisons will still be valid. It is nevertheless immediately obvious that the match results around the area give significantly divergent surfaces. The Simple Scalar Product and EMD Metrics diverge somewhat at the perimeter but the centre of attraction (of greater significance to matching) are coincident.

The Blurred/Scalar Product Metric has a surface whose peak is displace somewhat from that of the other two. This is understandable when consid-

retxels).

We can see that the EMD metric does appear to give a marginally more regular and distinct zone of good matching as compared with the original scalar product version. The blurred metric, as is reasonable, gives a noticeably smoother and more regular region that either of the other two. In fact, this metric gave a result in excess of threshold for all points mapped, a good sign when judging in terms of finding a complete model set. The question which arises in terms of a blurring approach is the behaviour in terms of false recognition.

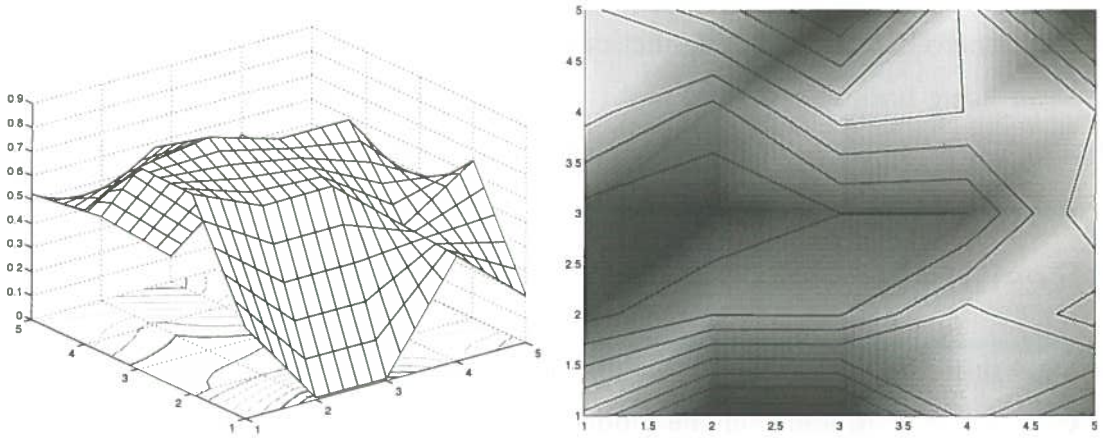


Figure 5.9: Surface plot and contour map of match results for original scalar product metric .

The results for the blurred metric integrated fully into the iconic system are given in Appendix A.8, Tables A.9 and A.10.

Again, the results are as expected. For the runs using the original model base, the blurred metric achieves better performance in terms of the total positive matches achieved over 20 saccades. The down-side is the large increase in incorrect matches, for-shadowed by the metric's performance for a single feature plane.

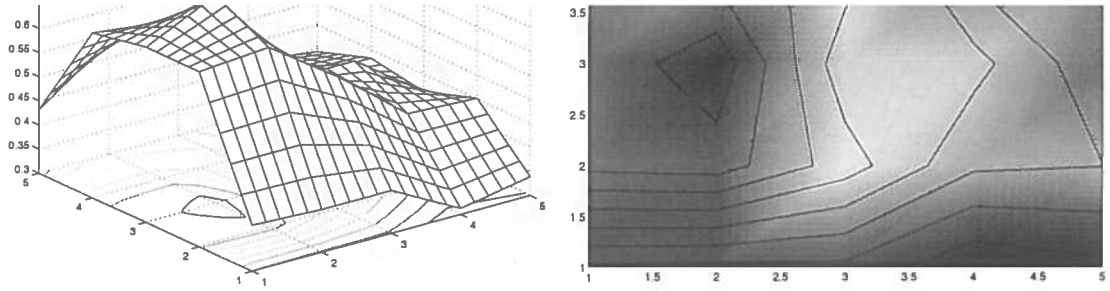


Figure 5.10: Surface plot and contour map of match results for EMD metric.

Further examination, however revealed that the metric was consistently obtaining raw feature scores (i.e. before being scaled by 0.7 and sub-component evidence being added) of 1, and so the solution was not that simple.

Again, use of the retrained model-base gave disappointing results, with an all round reduction in scores for all images.

5.8 Conclusions

The conclusions for this section overall form a large part of the main conclusion chapter (Chapter 6).

5.8.1 The EMD Metric

In short, the EMD metric shows some promise in it's applicability to solving image matching problems of this kind. Full evaluation of it's performance has not been possible due to the major current drawback in its use, that of the processor time required for it's operation.

The problem lies in the basic operation of the Simplex routine when it cannot take advantage of streamlining opportunities of the TP problem format. For the larger images, the number variables are:

$(48x20)((2r+1)^2+1+1)$

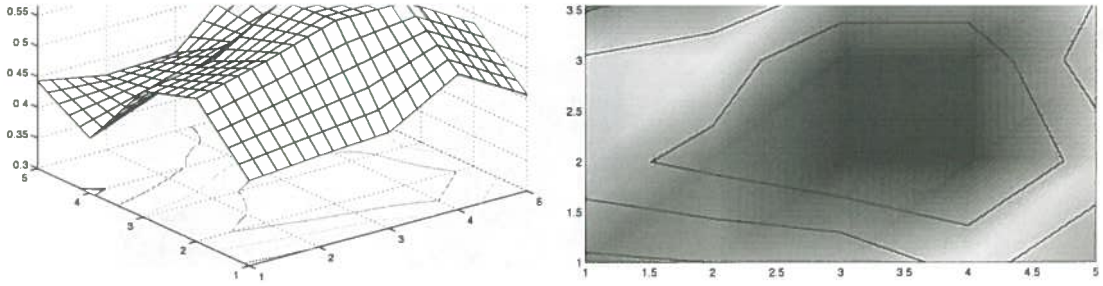


Figure 5.11: Surface plot and contour map of match results for blurred scalar product metric.

where r is the range allowed for transitions and the two additional multipliers correspond to a general unconstrained source and sink. The number of constraints will be 2(48x20) so we will be looking at an array with, for an allowed range of just one retxel, in excess of $2x10^7$ elements. For the smaller (24x5) array tested, the Simplex routine performed 730 pivot operations and took 22 seconds. For a standard run we require matching of 42 planes (14 each at scales of (48x20), (24x5) and (12x2) for each of 5 models at 9 micro-saccade points for 20 saccades.

To have a useful application we need improvement in running time of at least two orders of magnitude.

5.8.2 The Blurred/Scalar Product Metric

This metric was only investigated in the twilight hours of the project, when the computational restrictions of the EMD metric came to light. given the substantially less development time invested therein, the results are most promising. The positive recognition rate is improved and the 'basin of attraction' mapped for a single feature plane is or the form desired.

The rate of false recognitions is too high, an expected effect given the nature of the metric. There are several avenues which can be considered to

5.9 Further Work

Additional general work in mapping out the nature of the match result surface for the system using various metrics would be a valuable source of information for considering the systems overall behaviour. Similar tests could be carried out at a variety of feature points, using a variety of single feature planes or, where possible full weighted match scores.

5.9.1 The EMD Metric

Efficiency modifications to the program by Dr Fisher have already improved performance speed to some degree.

Another suggestion is to force an initial solution by building in an initial state that satisfies the linear equalities. This can be a very poor solution: we are just taking advantage of our prior knowledge of the problem to locate us on a constraint hyper-plane from the beginning. Analysis of the Simplex routine's function suggests that this should increase running speed by up to a factor of 7 by avoiding it's more general computations to achieve this. A possible solution to build in is:

$$\text{If} \quad In_i \geq Out_i \tag{5.13}$$

$$m_{ii} = Out_i \tag{5.14}$$

$$m_{i0} = In_i - Out_i \tag{5.15}$$

$$m_{0i} = 0 \tag{5.16}$$

$$m_{ij} = m_{ji} = 0; (j \neq) \tag{5.17}$$

$$\tag{5.18}$$

and

$$\text{If} \quad Out_i > In_i \tag{5.19}$$

(5.24)

Features already planned for inclusion are the early failure mechanisms: if the overall pixel intensities of image and model differ sufficiently, we register a no-match and move on. Also as noted, the distance between the centroids of the two intensity arrays is a lower bound on the pure EMD and again the EMD would only be attempted if the centroid is low enough.

This concept **may** be extended to our constrained case but it is questionable how far the similarities can be taken - the problem is essentially quite different due to our modifications.

The crucial point is probably in considering the implementation of the Simplex problem used. We no longer have the special structure of the transportation problem in it's original form, but we do have a significantly similar structure. Rubner implemented the transportation-simplex algorithm with a good initial solution ("Russell's approximation") and did achieve a speed in the region of the desired 2 orders of magnitude faster than the standard simplex implementation.In theory, it should be possible to add the neighbourhood constraints and dramatically increase the speed of the metric.

The code for this has kindly been made available by Yossi Rubner [Rubner] and so the field is open to move the metric onto a more practically useful implementation.

Where, as here, we are restricting the range of individual transfers to a certain distance an algorithm exists which computes a reasonable approximation for the EMD the time taken for which is linear in terms of the number of pixels (again, reported by Yossi Rubner in a personal communication and based on an idea from a technical report published by Karel Zikan)

discrimination ability. The magnitude of blurring is directly related to the discrimination attributes of the metric and has initially been set at an arbitrary level intended to give flexibility on a scale similar to that afforded by the EMD approach. Thus varying the amount of blurring is an obvious next step.

For pragmatic reasons the blurring is currently implemented on the image which is clearly not as efficient time-wise as applying to the model. In the former case, blurring is carried out repeatedly during the matching process, in the latter just at the model making stage. As the **blur** operation is a member function of the $R\Theta$ Class, it will be simple to call the function on the average model during training instead of on the image during matching. It should be remembered that the perceptron will need to be retrained and a new model base constructed.

Chapter 6

Conclusions and Further Work

Conclusions and further work directions specific to each section have been stated at the end of the relevant chapter and so a specifically global overview will be presented here.

6.1 Conclusions

During the course of this project, the focus of the work evolved from a general scheme to extend and enhance the iconic vision system to a more detailed development and evaluation of what became a novel metric. Latterly, investigation was extended to the Blurred/Scalar Product Metric due to drawbacks in the current time-scale of operation novel metric.

The *EMD* metric and more generally the *Transportation Problem*, *Simplex Method* and indeed *Linear Programming* itself have had applications in numerous areas in science, mathematics, industry and commerce.

Recent work [Rubner *et al.* 98] has seen the application of the approach to the realm of vision problems, there using the pure *EMD* metric to sort images by similarity of colour composition. Using the method as it stands for vision problems where spatial distribution signifies for anything other than a very small image presents considerable computational challenges.

The development herein of the *Constrained EMD* metric reduces the computational requirements substantially while not restricting the matching pos-

In a wider context, the metric has potential application to image matching problems where an element of local shifting is to be tolerated. The greater the allowed shift between the opponents in the match, the greater the computational cost.

The sacrifice of the exact Transportation Problem format to the metric, while saving substantially on computation has meant that the streamlined Simplex approach advocated by Rubner [Rubner *et al.* 98] has, for now, been set aside, and a standard Simplex formulation for solving linear programming problems is used.

At the current stage of development, the metric is too slow to be useful for real-time applications, even in the parallelised format of the iconic program. Performance would be enhanced by the incorporation of an initial solution and early failure provisions at the global level, before any detailed calculations are attempted. Although the existing streamlined Simplex routine is not compatible with the new variant, it is possible that some modification of this would be feasible given that many aspects of the TP unique format are retained.

The Simplex routine itself used herein is a proprietary version designed with optimisation problems in mind. This means there is no provision for early termination during the course of the calculation. The routine could be developed to account for the divergent requirements of vision problems: if the routine encompassed early termination, costly calculations could be reduced to a minimum. The code made available by Yossi Rubner is likely to offer significant advantages for future development.

The Blurred/Scalar Product Metric was only investigated during the last week of the project. Given the minimal development time directed into it thus far, it's results are impressive. The areas in which it could be modified are straightforward and as such likely to return good benefits for the amount

terministic, although the possible paths are finite and likely to be limited to two by their nature.

Previous evaluations of the matching abilities of the system have been accurate but have largely omitted consideration of the performance of the system without *a priori* knowledge. The recognition of features cited has been quite impressive, but unfortunately these occur in the context of a similarly 'impressive' number of false matches.

There is no clear distinction between the scores obtained for positive and negative feature matches (scores tend to increase fairly constantly over a run as subcomponent evidence accumulates), nor is there a pronounced tendency for positive matches to occur early.

Similarly previous results as to the number of saccades required for accruing a complete set of matches for features in an object were impressive, but neglect that the user may have no knowledge of the expected set composition. Waiting until no further matches seem forthcoming extends the real value of time required and leaves the user with an indistinguishable group of positive and negative feature matches.

An attempt to improve performance by retraining the perceptron gave interesting indicative evidence. The evidence issues from the large changes in weights on retraining, the persistent failure of the preceptron to reach mean square error convergence criteria and the poor performance of the retrained model base.

Consideration of improved metrics, perceptron retraining and other enhancements to the systems performance are is still important in improving the repertoire and performance of the system, as are 'engineering problems' such as parallelisation issues.

The issue underlying the system's behaviour at the most basic level though, is that of extracting a strong, linearly separable primitive feature set from an image. If, as is suggested circumstantially by many of the results

used in matching attempts may be insufficiently strong and may not constitute a linearly separable set. Work is currently being carried out to develop new feature detectors using a neural net applied to natural images which has promise for the systems further development [Gomes *et al.* 98].

If the integration of such a stronger 'backbone' becomes feasible, the various additional development options tried herein would take on a fresh aspect and should be repeated re-evaluated by future developers in that context.

Bibliography

[Bishop 95] C.M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press-Oxford, 1995.

[Dantzig 51] G.B. Dantzig. Application of the simplex method to a transportation problem. In *Activity Analysis of Production and Allocation*, pages 359–373. John Wiley and Sons, 1951.

[Farah 90] M.J. Farah. *Visual Agnosia: Disorders of Object Recognition and what they tell us About Normal Vision.* MIT Press, 1990.

[Fisher & MacKirdy 98] R.B. Fisher and A. MacKirdy. Integrating iconic and structured matching. *Proc. 5th Eur. Conf. on Computer Vision, Freiburg*, June 1998.

[Fisher & Oliver 97] R.B. Fisher and P. Oliver. Multi-variate cross correlation and image matching. *Proceedings of the British Machine Vision Conference,pp 623-632, Birmingham*, September 1997.

[Gomes *et al.* 98] H.M. Gomes, R.B. Fisher, and J.C. Hallam. A retina-like image representation of primal sketch features using a neural network approach. *Proc. NOBLESSE workshop on Non-linear Model based image analysis NMBIA98, Glasgow*, July 1998.

[Grove 95] T.D. Grove. Attention directed iconic object matching. Unpublished M.Sc. thesis, Dept. of Artificial Intelligence, University of Edinburgh, 1995.

[Hiller & Lieberman 74] F.S. Hiller and G.J. Lieberman. *Operations Research.* Holden-Day, Inc., 1974.

[Jennens 94] J. Jennens. Quasi-invariant iconic object recognition. Unpublished M.Sc. thesis, Dept. of Artificial Intelligence, University of Edinburgh, 1994.

[MacKirdy 97] A. MacKirdy. Full subcomponent evidence and further parallelism in an iconic object recognition system, 1997.

[Marques 96] P.J.P. Marques. A parallel connectionist viewpoint quasi-invariant iconic object recognition system. Summer report, Dept Artificial Intelligence, University of Edinburgh, August 1996.

[Marr 80] D. Marr. *Vision.* Freeman and Company, 1980.

[Press *et al.* 89] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1989.

[Rubner] Y. Rubner. *Code for the EMD.* URL:http://vision.stanford.edu/rubner/emd/.

[Rubner *et al.* 97] Y. Rubner, L. Guibas, and C. Tomasi. The earth mover's distance, multi-dimensional scaling and color-based image retrieval. *Proceedings of the*

Conference on Computer Vision, Bombay, India, pp59-66, January 1998.

Appendix A

Results

The Experimental work was performed on a range of Sun machines in the A.I. department's network, ranging from Sparc1+ to Sun Ultras. The set 0f 19 Ultra Sparcs used by MacKirdy were still available and these were used in parallel runs of the system.

Serial runs were on a Sun Ultra 1 Creator 3D(host name *"Sapphire"*, except where code was being purified when an older machine (host name *"Oculus"* was used.

The format of table used throughout the appendices for noting matching performance are referenced by the keys in Table A.1. For the *'EYE'* model, there is no differentiation made between left and right eye. Thus reports of false instances for each refer to matches which do not fall into the 30-retxel range of either eye. The false instances are note in both eye columns.

In previous work, saccading and thus false result accrual has been terminated when all features have been correctly located. Thus the emphasis has been on the correct matches, neglecting somewhat the fact that with no *a priori* knowledge, these are not discernible.The system is always run for 20 saccades to give a fair and consistent picture of performance.

Sac. first found	*Match score*
Offset error	*False instances*

Total Features extant	*Total positive matches*
Average offset	*Total negative matches*

the experiments to allow result comparisons. The standard settings were:

```
Configuration of iconic
=======================

World picture file [c1.ppm]: *changed to reference
                             test image as required*
Model base filename [convic.base]: *changed where
                                   retrained base tested*
Number of models per node [4]:
Number of saccades [20]:
micro-saccades per node [9]:
scale loop lower limit [0]:
scale loop upper limit [1]:
scales per node [1]:
rotation loop lower limit [0]:
rotation loop upper limit [1]:
rotations per node [1]:
Recognition threshold [0.680000]:
Initial foveation X [155]:
Initial foveation Y [191]:
```

A.1 Control Runs

To give a reliable baseline for making comparisons of performance, the system was first run with the existing matching metric and model base. The Results are shown in A.2

	face		eye 1		eye 2		nose		mouth		Summary	
c1.ppm	5	0.76	4	0.72	6	0.85	8	0.87	7	0.88	5	5
	6	1	15	0	14	0	3	0	9	0	9	1
c3.ppm	1	0.70	14	0.77	17	0.81	-	-	18	0.83	5	4
	20	1	19	3*	9	3*	-	2	18	1	16	10
c4.ppm	1	0.70	3	0.78	4	0.84	6	0.90	-	-	5	4
	8	0	20	0	9	0	4	0	-	0	14	0
c5.ppm	-	-	16	0.99	13	0.93	9	0.74	10	0.78	5	4
	-	0	13	0	11	0	6	1	14	0	11	1
c6.ppm	1	0.70	3	0.83	4	0.84	5	0.85	2	0.78	5	5
	4	0	7	0	10	0	13	1	14	1	9	2
fce5.ppm	15	0.92	13	0.83	5	0.70	14	0.81	11	0.75	5	5
	1	0	4	1*	7	1*	4	2	6	0	5	4
bebie1.ppm	9	0.86	-	-	6	0.8	-	-	-	-	2	5
	12	1	-	0	7	0	-	1	-	1	9	3

Table A.2: Results of control run

c5.ppm	13	0.93	12	0.78	11	0.84	9	0.70	10	0.74	5	5
(path 2)	7	0	4	0	13	0	2	1	2	0	5	1
fce5.ppm	15	0.92	13	0.83	5	0.70	14	0.81	11	0.75	5	5
(path 1)	1	0	4	1*	7	1*	4	2	6	0	5	4
fce5.ppm	-	-	-	-	12	0.72	-	-	-	-	2	5
(path 2)	-	0	-	4*	6	4*	0	2	3	0	9	11

Table A.3: Examples of divergent control paths

A.2 Reproducibility of Results

Table A.3 shows the alternate saccade paths found in investigations of the system's determinism.

A.3 Parallel Timing Results

Tables A.4a and A.4b show the maximum and minimum times recorded for the system to run for each saccade over 15 20-saccade runs and a summary of the total saccade times for each.

A.4 Perceptron Retraining

Table A.5 shows the performance achieved by using model base *convic2*, produced with no change in the training images supplied, but with the ideal scores for the perceptron learning changed to **[0.05, 0.95]**

saccade	$T_{\min}$(s)	$T_{\max}$(s)
1	13.0	13.4
2	13.3	13.7
3	13.3	13.7
4	12.9	13.3
5	13.3	13.3
6	12.9	13.3
7	12.1	12.4
8	12.0	12.4
9	10.8	11.0
10	11.5	11.7
11	12.8	13.0
12	11.5	11.8
13	13.1	13.5
14	11.9	12.3
15	11.2	11.5
16	10.2	10.3
17	11.5	11.7
18	11.9	12.3
19	10.7	10.9
20	14.3	14.9

run	$T_{20\ \text{saccades}}$(s)
1	252
2	247
3	252
4	247
5	248
6	252
7	247
8	247
9	252
10	247
11	252
12	247
13	247
14	248
15	252

Table A.4: Variation over 20 runs of 20 saccades of time for completing one Saccade and over 15 runs of 20 saccades of time for completing 20 Saccades.

	face		eye 1		eye 2		nose		mouth		Summary	
c1.ppm	4	0.76	3	0.70	7	0.93	6	0.88	5	0.82	5	5
	14	1	11	0	14	0	3	0	4	1	9	2
c3.ppm	-	-	-	-	-	-	-	-	-	-	5	0
	-	2	-	5*	-	5*	-	2	-	2	-	16
c4.ppm	15	0.77	-	-	16	0.87	5	0.70	-	-	5	3
	8	2	-	3*	10	3*	3	1	-	3	7	12
c5.ppm	17	0.83	13	0.75	16	0.80	19	0.92	9	0.72	5	4
	6	0	13	1*	10	1*	5	4	11	0	11	5
c6.ppm	1	0.70	4	0.89	2	0.78	5	0.91	3	0.82	5	5
	2	1	4	0	1	0	11	0	8	1	5	2
fce5.ppm	-	-	-	-	-	-	-	-	-	-	5	0
	-	2	-	1*	-	1*	-	0	-	4	-	8
bebie1.ppm	3	0.70	5	0.75	-	-	-	-	16	0.82	5	3
	15	1	4	0	-	0	-	0	5	3	8	4

Table A.5: Results of using retrained model base.

procedures applied. In each case :

- m1 is the number of 'less than' constraints
- m2 is the number of 'greater than' constraints
- m3 is the number of 'equality' constraints
- m is the total number of constraints ($m = m1 + m2 + m3$)
- n is the number of variables subject to the constraints
- aTemp constitutes the tableau input to the problem, specifying the constraints to be applied
- the results shown, indexed by row have been independently checked for accuracy

```
****************************************************************
TEST PROBLEM 1:
m=2; n=2; m1=0; m2=0; m3=2;

float aTemp[4][3]= {{0, 2,-4},
                    {2,-6,1},
                    {8 ,3,-4}} ;

finite solution found

[1,1]0.666667; [2,1]0.333333; [3,1]9
```

```
m=4; n=4; m1=2; m2=1; m3=1;

float aTemp[6][5]= { {0,   1, 1, 3,-0.5},
                     {740,-1, 0,-2, 0  },
                     {0,   0,-2, 0, 7  },
                     {0.5, 0,-1, 1,-2  },
                     {9,  -1,-1,-1,-1  }};

finite solution found

[1,1]17.025; [2,1]730.55; [3,1]3.325; [4,1]0.95; [5,1]4.725
[5,2]-0.55; [5,3]0.05; [5,4]0.55; [5,5]-0.45
**************************************************************
TEST PROBLEM 3:

m=3; n=2; m1=3; m2=0; m3=0;

float aTemp[5][6]={{0,  3, 5},
                   {4, -1, 0},
                   {12, 0,-2},
                   {18,-3,-2}};

finite solution found

[1,1]36; [2,1]2; [3,1]; 6[4,1]2
```

```
m=3; n=2; m1=3; m2=0; m3=0;

float aTemp[5][3]= {{0, -3,-5},
                    {4, -1, 0},
                    {12, 0,-2},
                    {18,-3,-2}};

finite solution found


[1,1]-36; [2,1]-2; [3,1]6; [4,1]2

*************************************************************************
TEST PROBLEM 5:
This was supposed to give no feasible solution

m=3; n=2; m1=3; m2=0; m3=0;

float aTemp[5][3]= {{ 0, 3, 5},
                    { 1,-1, 0},
                    {12, 0,-2},
                    {18,-3,-2}};

objective function has no solution
```

```
m=7; n=12; m1=0; m2=0; m3=7;

float aTemp[9][13]=
      {{0,464,513,654,867,352,416,690,791,995,682,388,685},
       {75,1,1,1,1,0,0,0,0,0,0,0,0},
       {125,0,0,0,0,1,1,1,1,0,0,0,0},
       {100,0,0,0,0,0,0,0,0,1,1,1,1},
       {80,1,0,0,0,1,0,0,0,1,0,0,0},
       {65,0,1,0,0,0,1,0,0,0,1,0,0},
       {70,0,0,1,0,0,0,1,0,0,0,1,0},
       {85,0,0,0,1,0,0,0,1,0,0,0,1}};
```

DID NOT GIVE SOLUTION, LOOKING MORE CLOSELY, THE SIGN CONVENTION IN THE ROUTINE IS REVERSED: TO MINIMISE AS REQUIRED, SHOULD SHOW THUS:

```
float aTemp[9][13]=
      {{0,-464,-513,-654,-867,-352,-416,-690,-791,-995,-682,-388,-685},
       {75,-1,-1,-1,-1, 0, 0, 0, 0, 0, 0, 0, 0},
       {125,0, 0, 0, 0,-1,-1,-1,-1, 0, 0, 0, 0},
       {100,0, 0, 0, 0, 0, 0, 0, 0,-1,-1,-1,-1},
       {80,-1, 0, 0, 0,-1, 0, 0, 0,-1, 0, 0, 0},
       {65, 0,-1, 0, 0, 0,-1, 0, 0, 0,-1, 0, 0},
       {70, 0, 0,-1, 0, 0, 0,-1, 0, 0, 0,-1, 0},
       {85, 0, 0, 0,-1 ,0, 0, 0,-1, 0, 0, 0,-1}};

[1,1]-152535; [2,1]70; [3,1]45; [4,1]30; [5,1]80; [6,1]55; [7,1]20; [8,1]0

*******************************************************************************
```

Following is a representative transcripts of the output from the tableau construction routine. The test problem shown is the smallest performed, to limit the space required for the output i.e. a 3x3 array pair.

The first part of output is the objective function which indicates the range of allowed transitions and their cost (note that by the convention of the original formulation, all tableau coefficients are negative)

```
**************************************************************************
Objective function:

0
-1 -1 -1 -1 0 -1 -1 -1 -1 (top left pixel)
-1 -1 -1 -1 0 -1 -1 -1 -1 (top right pixel)
-1 -1 -1 -1 0 -1 -1 -1 -1 (bottom left pixel)
-1 -1 -1 -1 0 -1 -1 -1 -1 (bottom right pixel)

 **************************************************************************
```

This represents the cost for transitions to any of the 8-neighbours of a pixel as 1, remaining stationary has cost zero. All transitions within the allowed 1 retxel range are considered here, the fact that the number of transitions is further limited by array size is taken into account later.

The problem is passed in as a file containing the array size (y,y) followed by its elements

```
1 0
0 0
1 0
0 0
```

****

Here, we define a 2x2 array and show that we are performing the most basic test, one value one element compared with an identical array. the routine now constructs a modified TP-format array indicating allowed transitions

****

```
-1-1-1-1-1-1-1-1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0-1-1-1-1-1-1-1-1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0-1-1-1-1-1-1-1-1 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0-1-1-1-1-1-1-1-1
 0 0 0 0-1 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0 0
 0 0 0 0 0 0-1 0 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0 0-1 0 0 0 0 0
 0 0 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0-1 0 0 0 0 0 0 0 0-1 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

****

is to be source/sink for.

```
**********************************************************************

-2-2-2-2-2-2-2-2

-1 0 0 0 0 0 0 0

0-1 0 0 0 0 0 0

0 0-1 0 0 0 0 0

0 0 0-1 0 0 0 0

0 0 0 0-1 0 0 0

0 0 0 0 0-1 0 0

0 0 0 0 0 0-1 0

0 0 0 0 0 0 0 0-1

0 0 0 0 0 0 0 0
**********************************************************************
```

To compare the performance of the metrics outside the context of the iconic matching program and irrespective of time considerations, a single [24x5] feature plane of the *EYE* model dumped from the iconic program:

```
4 2 1 3 1
5 2 2 1 1
7 4 3 3 1
8 7 4 1 1
7 8 5 0 0
8 5 3 1 0
9 3 2 0 0
8 3 2 3 1
7 3 1 5 2
5 1 3 3 2
3 2 2 2 1
3 2 3 1 0
3 2 4 4 2
3 2 4 5 2
3 3 4 5 5
5 2 6 6 4
5 2 5 10 5
6 4 6 10 7
7 5 10 10 7
6 4 6 12 6
4 2 4 11 4
4 3 5 4 4
3 5 6 7 3
3 3 5 5 4
```

0.642	0.625	0.578	0.431	0.431
0.431	0.533	0.520	0.413	0.393

Table A.6: EMD score around point [135,236] in image *c1.ppm*

0.378	0.010	0.010	0.441	0.193
0.833	0.705	0.708	0.465	0.657
0.842	0.886	0.800	0.801	0.414
0.559	0.740	0.455	0.502	0.502
0.523	0.350	0.107	0.534	0.078

Table A.7: Scalar product score around point [135,236] in image *c1.ppm*

This primative plane was matched in a [5x5] area around a known instance of an eye (point [135,236] in image *c1.ppm*) and the feature score recorded at each point was recorded. The EMD score was manually normalised to assist comparison with the simple and blurred scalar product matches.

A.8 Performance of Blurred Image Matching

The iconic system was run on the trained and unseen images for 20 saccades each using alternately the original model base and that retrained using modified ideal scores.

0.441	0.451	0.458	0.522	0.457
0.535	0.564	0.599	0.599	0.533
0.506	0.524	0.593	0.593	0.469
0.390	0.434	0.476	0.477	0.521
0.443	0.422	0.427	0.528	0.540

Table A.8: Blurred scalar product score around point [135,236] in image *c1.ppm*

	face		eye 1		eye 2		nose		mouth		Summary	
c1.ppm	9	0.76	18	0.93	7	0.87	4	0.84	12	0.84	5	5
	17	2	25	2*	15	2*	9	0	26	0	17	4
c3.ppm	1	0.70	5	0.94	2	0.78	3	0.81	4	0.89	5	5
	16	4	15	3*	14	3*	16	0	14	0	15	7
c4.ppm	11	0.93	4	0.78	2	0.72	7	0.78	-	-	5	4
	0	4	10	2*	12	2*	10	0	-	2	8	8
c5.ppm	6	0.82	1	0.7	3	0.73	7	0.82	9	0.88	5	5
	10	1	21	1*	7	1*	27	5	28	1	21	8
c6.ppm	-	-	13	0.81	-	-	-	-	-	-	1	5
	-	7	13	4*	-	4*	-	2	-	0	13	13
fce5.ppm	-	-	-	-	-	-	-	-	-	-	-	5
	-	1	-	11*	-	11*	-	2	-	0	-	14
bebie1.ppm	4	0.74	2	0.74	6	0.79	-	-	8	0.81	4	5
	20	6	10	1*	8	1*	-	2	24	1	15	11

Table A.9: Results of matching with Gaussian blurring - original model base

	face		eye 1		eye 2		nose		mouth		Summary	
c1.ppm	6	0.77	15	0.98	7	0.87	5	0.76	8	0.9	5	5
	6	1	21	2*	9	2*	28	0	20	0	17	3
c3.ppm	1	0.70	2	0.78	3	0.85	-	-	-	-	5	3
	16	2	16	7*	16	7*	-	4	-	-	16	14
c4.ppm	-	-	8	0.84	2	0.71	15	0.81	-	-	5	4
	-	8	12	0	15	0	9	2	-	2	12	12
c5.ppm	4	0.70	8	0.82	14	0.85	-	-	-	-	5	3
	13	9	19	2*	12	2*	-	-	3	1	15	15
c6.ppm	17	0.74	7	0.89	4	0.8	2	0.78	8	0.93	5	5
	30	7	3	1*	17	1*	4	1	7	0	12	9
fce5.ppm	-	-	-	-	-	-	-	-	-	-	-	5
	-	6	-	7*	-	7*	-	3	-	1	-	17
bebie1.ppm	-	-	-	-	-	-	-	-	-	-	-	5
	-	4	-	8*	-	8*	-	3	-	1	-	16

Table A.10: Results of matching with Gaussian blurring - retrained model base

Appendix B

Images and Descriptions

B.0.1 Untrained Images

```
fce5.trn
6
FACE  158 150
EYE   113 133
EYE   199 137
NOSE  137 197
MOUTH 161 221
FOV   163 190
```

```
bebie1.trn
5
FACE  197 163
EYE   164 152
EYE   237 153
NOSE  187 199
MOUTH 196 227
```

```
c1.trn
6
FACE  183 244
EYE   135 236
EYE   220 231
NOSE  161 296
MOUTH 180 332
FOV   183 278

c3.trn
6
FACE  168 179
EYE   118 157
EYE   203 163
NOSE  149 222
MOUTH 163 253
FOV   166 199

c4.trn
6
FACE  152 180
EYE   110 162
EYE   201 168
NOSE  135 228
MOUTH 149 264
FOV   152 210
```

```
EYE   133 208
EYE   207 220
NOSE  152 267
MOUTH 173 298
FOV   169 244

c6.trn
6
FACE  150 191
EYE   105 176
EYE   202 176
NOSE  136 238
MOUTH 157 272
FOV   153 218
```

c1.ppm

c3.ppm

c4.ppm

c5.ppm

c6.ppm

Figure B.1: Face images used to construct *convic.base* and *convic2.base*

Figure B.2: Untrained images used for matching