Life beyond Pascal - MODULA-2 from a users point of view

J.M.Bishop,
Department of Cybernetics, University of Reading, Berkshire, UK
email : J.M.Bishop@Reading.ac.uk

ABSTRACT

Using a large number of specially configured small Random Access Memories mapped over a target picture it is possible to generate the edge filtered image on every clock cycle of the hardware.[1]

This massively parallel process is expensive to develop discretely in hardware but can be simulated on a standard sequential computer. This paper describes such a simulation and gives reasons why the language Modula-2 was selected for the software implementation of the project.

INTRODUCTION

The special use of Random Access Memories (RAMs) in image processing is well documented in the literature [1, 2, 3]. A novel use for an array of specially programmed small RAMs mapped over a target image is to edge filter the image. This is accomplished by programming the RAMs to act in a manner analogous to the Retinal Ganglion cells of the eye which can be said to function as edge processing units. An advantage of this technique over other edge filtering algorithms, such as the Sobel or Laplace transforms, is that the hardware can produce a complete edge filtered image at each clock cycle of the memories, typically around every 100nS. However, circuits with many small memories are expensive to make from discrete elements, so instead a practical system could be made from a VLSI design or a subset of the elements scanned sequentially across the entire picture or as a software emulation of the hardware.

The VLSI design has the advantage that it is completely parallel and could operate at the theoretical maximum speed of the system and also could feasibly be included on a one chip CCD camera circuit. The second, semi parallel option, of using an array of RAMs covering a subset of the source image could also be built in to the digitisation stage of the image capturing process. This could be achieved by mapping an array of RAMs over one vertical line of the image generated by a column digitiser or by sequentially scanning the RAMs across a complete image captured by a frame store.

The third inherently sequential method of simulating the system on a standard computer loses the speed advantage of the technique, but can illustrate quickly and cheaply that the method is works.

This paper outlines one such software emulation of a neural network edge filtering system.

---

[1] See also, Aleksander, I, '*Adaptive Windows for Image Processing*', IEE Proceedings, **132:5** pt E, September 1985.

Neural Networks

Historical

In pioneering work at Brunel University Professor Igor Aleksander developed the concept of the Unprogrammed Neural Network. This is a semi-distributed architecture based upon a RAM / Neuron analogy and which enabled the construction of cheap Neural Network Image Processing systems with a form of content addressability. A simple Neural Net is a group of RAM / Neurons mapped over a target image and the network classifies its input by summing the number of RAM / Neurons that fire on presentation of an arbitrary image.

Simple Neural Networks

A single RAM can be shown to classify simple binary images by mapping the image onto its address lines and storing a TRUE value at the image generated address, however more complex images force the use of 'Compound RAMs' [4], since RAM size increases exponentially with image size. A compound RAM uses multiple small RAMs to cover its image space. The number of address lines from each RAM mapped onto the image is defined as the n-tuple size of the system. The Network can be trained to recognise a specific pattern by storing TRUE at the address generated at each RAM by the target image. The systems response to an arbitrary image is defined as the number of RAM / Neurons that fire (output TRUE) on the presentation of the image. An Image Discriminator is one large compound RAM covering the target image, and a Neural Network system capable of classifying [n] distinct images is a set of [n] such Discriminators acting in parallel over the target image.

Edge Filtering by Retinal Ganglion Cells

In broad outline, the visual pathway from light passing through the eye lens to electrochemical 'perception' in the brain is well defined; light entering the eye is focussed onto Receptor cells on the retina. Output from these cells is collated by Bipolar cells which in turn connect to Retinal Ganglion cells, whose output passes to the Optic Nerve. This nerve carries signals from the eye until reaching a junction with the Optic Nerve from the other eye, called the chiasm. Here, about half of the fibres from each nerve cross into opposite hemispheres of the brain and lead to two cell clusters called the Lateral Geniculate Bodies from where signals pass on to the visual area of the Cerebral Cortex [5].

Contrast Processing occurs as information filters across Retinal Ganglion (RG) cells, of which Kuffler found two types in his investigation of the visual pathway, On-Centre and Off-Centre field cells. An On-Centre field cell receives excitatory input from a circular area at the centre of its receptive field and inhibitory input from the circular area surrounding the On-Centre (vice-versa for an Off-Centre field cell). The mechanism by which edge filtering takes place is that a spot of light covering the centre of the cell produces maximal cell firing, whereas any light falling on the surrounding Off field inhibits this firing, such that if light covers the entire receptive field, the cell fires weakly if at all. If however light falls only on the On-Centre the cell fires strongly, peaking when

the entire On Centre is covered.

Considering the behaviour of a network of such On-Centre field cells mapped over an image of a light square on a dark background, we see little activity on any of the cells whose receptive fields cover the dark background. There will also be little activity from the cells covering the light square, as the excitatory and inhibitory inputs tend to cancel out. However, at the boundary of light and dark there will be some activity as some cells will be more strongly excited by light on their On-Centre than inhibited by light falling on their Off surround. The net response of the system will thus be areas of high cell activity at the boundary of the light square and dark background - the edges of the square.

Simulating Edge Filtering by Neural Network Methods

To simulate an RG cell consider a Neural Network with two Discriminators of [n] RAMs, the first mapped over the central circular area of an image and the second covering the surrounding circular area. The RAMs in both Discriminators are programmed to fire only when all their input lines are firing, that is they are TRUE. The output of the system is defined as the MODULUS of (DISC[1] - DISC[2]) where DISC[n] is a count of the RAMs in Discriminator [n] that are firing. The behaviour of the system when presented with an arbitrary image is analogous to that of an On-Centre RG Cell.

However the method can be simplified when considering a simply binary system. Now an RG cell can be simulated crudely by one RAM of n-tuple size nine, with its inputs arranged in a 3 by 3 matrix. The centre input is defined as being on an edge if it is TRUE and any of the surrounding inputs are FALSE. Thus the RAM is programmed to output FALSE for all addresses where the centre input is FALSE and also for the address where all the surrounding inputs are TRUE. It is programmed to output TRUE for all other addresses.

If a 2D Boolean image is completely covered by a 2D array of such RAMs mapped uniformly across it, then the edge filtered output is defined as the 2D Boolean array formed from the output of the RAMs.

Software simulation

The above Neural Network system designed in hardware has the capability of producing results very quickly (at the cycle time of the RAM), however it is expensive to build discretely, thus a software simulation was designed. At the outset of the design procedure a choice of language had to be made. Neural Networks are best hand coded in assembler for optimum speed, however for research purposes it was desirable to code in a high level language to enable quicker development and good readability.

Choice of High Level Language

Program development was on the Commodore Amiga microcomputer for which a large Neural Network work station (Osiris) had already been designed. The standard Osiris system allowed experimentation with various Neural Network techniques such

as displaying the output of the RAMs as a count of the number of RAMs firing or as another smaller image. It also enabled feedback of the resulting image into the source image and it was decided to add Edge Filtering as an extra function of this system.

Osiris was programmed in Modula-2, this language being selected from those available such as BCPL, C, Pascal, Modula-2, Lisp, Basic, for the following reasons.

The functional outline of Osiris mapped neatly into separate Modula-2 type modules and modern programming methods favour modular program development where the system is divided into small modules which can be designed and tested separately. This technique speeds up program development because designing small modules individually is easier than developing them all together in one large program. It reduces compilation time as working modules need only be compiled once and it allows the use of hidden variables and data types.

Secondly, Modula-2 supports a very strongly typed environment and this can help find problems at compile time that would otherwise remain hidden until program run time and be consequently harder to trace.

As Osiris is part of an active research program it was important that the program code should be easily readable and not obscure the underlining principles of the system; it is difficult to make programs readable in Basic as it does not provide many tools for the practise of structured programming techniques and both BCPL and C have a poor language syntax with respect to program legibility. However because target pictures were fairly large, typically around 320*128 pixels, efficient object code was needed to enable analysis of the images in reasonable time. This suggested a fully compiled language.

Finally there was a historical aspect guiding choice of language to one of the Pascal family, as Osiris was an extension of a smaller system programmed in Pascal on a BBC-B micro computer. Modula-2 was preferred to Pascal for reasons given above and conversion from Pascal to Modula-2 is relatively simple as the languages are similar.

Implementation

Using the Modula-2 concept of the Module, we can define *Retina* which takes as input an image in the form of a 2D array of Boolean values, and returns an output image in the same format. To generate a point on the output image Retina needs access to a lower level module called *RAM* to which it passes the address generated by the selection of nine (the n-Tuple size) Boolean values from a corresponding area on the input image. *RAM* can then use these values to access its own pretaught *hidden* array of 2^(n-tuple) values returning as output the Boolean value stored there.

At the highest level the module *Edge-Filter* can be defined to program the system RAMs to act as edge filtering cells as described in section four. Thus to get Osiris to

edge filter an image it is necessary to initially program the system *RAMs* to act as edge filtering cells by calling *Edge-Filter*, then simply passing to *Retina* the target image and displaying on the screen the resulting output image.

The above description highlights the use of hidden data structures within program modules, and also how the edge filtering process uses two library modules, *RAM* and *Retina*.

Problems with the Modula-2 Environment

The use of Modula-2 on the Amiga was not without problems. The system used (from Modula-2 Software Ltd) did not provide a particularly sheltered environment from the raw machine and most of the program development time was taken finding out how to plot a point/line on the screen, reading/writing to the serial port, etc. It was felt that a Modula-2 environment should provide a high level interface to the target machine such as that postulated in the OSSI system [6]. This approach still allows those users who need it access to low level machine features, while allowing a high degree of abstraction from the target machine for the majority of programmers.

Results

Figure 1a - Target Image to Edge Filter

Figure 1b - Negative of Target Image

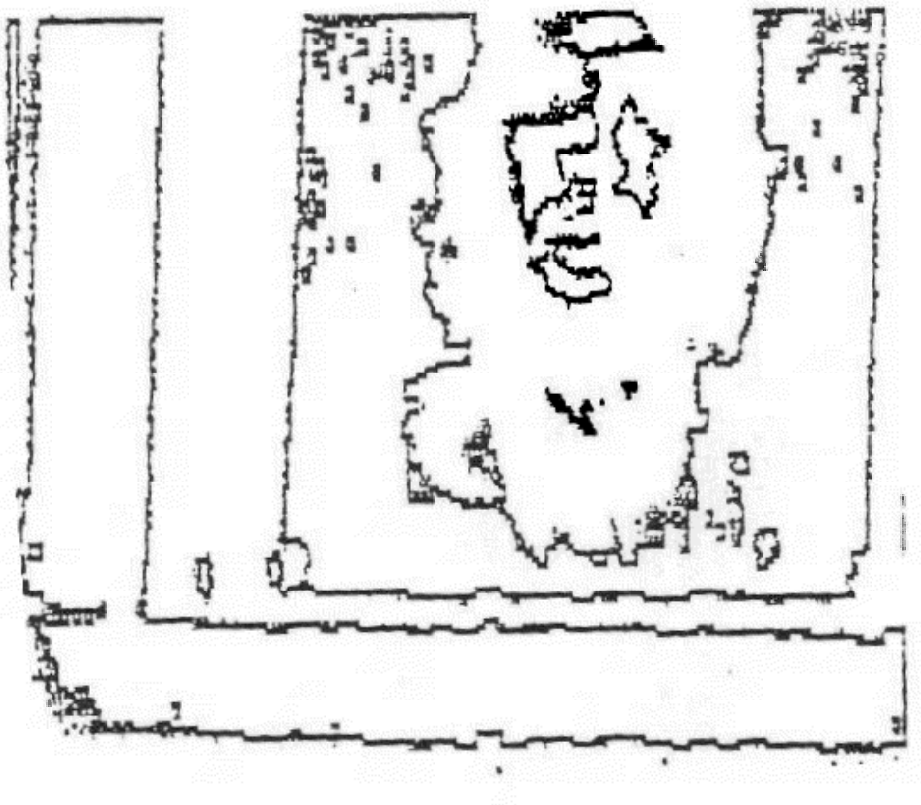

Figure 2 - Edge Filtered Image

Figures 1 and 2 show the performance of the edge filtering system when presented with a target image of 320 by 128 binary pixels. Figure 1a is the actual input image, Figure 1b is its negative from which the edge filtered result can more easily be compared.

These images were produced on a small system running on the BBC-B micro computer

and took around two hours to produce compared to around one minute on an early version of Osiris programmed in Modula-2 on the Amiga.

The time of around one minute to edge filter a Boolean image could be speeded up to less than a second by hand coding Osiris in assembler however, the sequential nature of the software emulation has lost most of the speed advantage of the neural network technique.

Conclusions

Neural Network methods have been used successfully in practical image processing systems for many years at Brunel University and commercially by CRS of Wokingham. Their speed, simple operator retraining and performance with noise, making them very useful in some industrial applications. This paper has shown how small neural networks can be trained to act as edge filtering devices and the Osiris software simulation has shown that the technique performs well, albeit slightly slowly, on a standard computer but offers the potential for extremely high performance in a completely parallel hardware system.

REFERENCES

1. Aleksander, I. & Burnett, P., (1983), *Reinventing man*, Kogan Page, London, UK.

2. Aleksander, I. & Stonham, T.J., (1979), *A guide to pattern recognition, using random access memories*, IEE J. Comput. & Digital Tech, **2:1**, pp. 29-40.

3. Aleksander, I., Thomas, W.V. & Bowden, P.A., (1984), *WISARD, a radical step forward in image recognition*, Sensor Review, **4:3**, pp. 120-124.

4. Aleksander, I. & Burnett, P., (1983), *Reinventing man*, pp. 221-224, Kogan Page, London, UK.

5. Hubel, D.H., (1963), *The Visual Cortex of the Brain*, Scientific American, **225:11**, pp. 447-448.

6. Biagioni, E., Heiser, G., Hinrichs, K. & Muller, C., (1987), *OSSI - A program environment for developing portable software*, IERE Colloquium on Modula-2 (11/2/87), London, UK.