

## A HARDWARE ARCHITECTURE FOR A PARALLEL GENETIC ALGORITHM FOR IMAGE REGISTRATION

B.C.H. Turton, T. Arslan, D.H. Horrocks

**Introduction** Parallel Genetic Algorithms (PGAs) have frequently been cited as an important area of research as they provide a means of rapidly developing a solution to a wide range of problems. Real-time image analysis is one of the areas of research which would particularly benefit from PGAs, however such algorithms are generally simulated on conventional computers or are designed for expensive hardware systems. For practical image processing on a large scale, cheap, fast and efficient PGA processors are required as part of a vision system.

Vision systems require processing techniques which are robust, fast and capable of dealing with large quantities of data. Genetic algorithms have been used because of the first of these criteria, as is exemplified in the works of Fitzpatrick 1984, Mandava 1989, McAulay 1989. However by using hardware parallel genetic algorithms the second and third criteria could also be fulfilled. Fitzpatrick suggests that a parallel implementation would be beneficial and various forms of Parallel Genetic Algorithm (PGA) have been suggested by other authors. Many of the different techniques are covered in the five international conference proceedings on genetic algorithms and the first Workshop on Parallel Problem Solving from Nature. In this paper the nature of Parallel Genetic Algorithms is described followed by the application of genetic algorithms to vision systems. A description of a hardware architecture for vision systems is detailed along with various modifications to improve the implementation. A simulation is used to produce results that verify the effectiveness of the hardware architecture and finally conclusions and future work are discussed.

**Parallel Genetic Algorithms** Parallel genetic algorithms can be categorised into three types, 'standard', 'coarse grained' and 'fine grained'. Standard GAs can be implemented on a parallel architecture by distributing the evaluation process over a number of processors (Fogarty 1990). Coarse grained genetic algorithms run several populations of genes in parallel. After a number of generations (G) the separate populations export a set of individuals (n) to other neighbouring populations. G generations is termed an 'epoch' or migration period. Normally a single processor will monitor each population and the processor swaps individuals every epoch thus parallelising the problem over the number of populations (processors) within the multiprocessor system (Figure 1). A variety of topologies have been used to define 'neighbours' typically a simple grid or hypercube (Figure 2) is used with each node corresponding to a processor (Tanese 1989, Cohoon 1990, Muhlenbein 1991, Xu 1992). Theoretical studies of a coarse grain GA have been done by Petty & Leuze 1989.

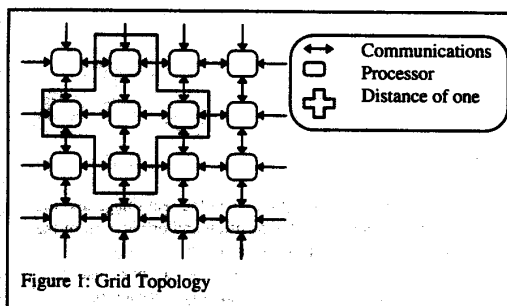


Figure 1: Grid Topology

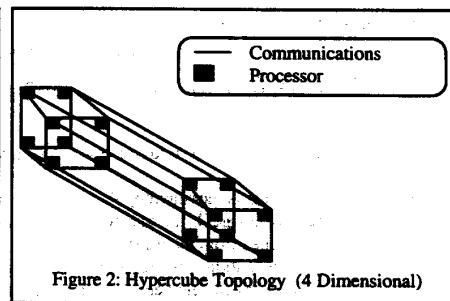


Figure 2: Hypercube Topology (4 Dimensional)

Fine-grained parallel genetic algorithms act on each member of the population in parallel. Consequently each member of the population performs crossover with its immediate neighbours, where the neighbourhood is defined by the topology and some distance parameter (Georges-Schleuter 1990, Manderick 1989, Tomassini 1993, Figure 1). Manderick's work suggests that a distance parameter of two on a simple grid topology is reasonable. There is no artificial separation of the chromosomes into distinct populations with fine-grained PGAs however the neighbourhood restriction permits separation by distance. Tomassini uses a neighbourhood of four neighbours, but also permits 'migration' to greater distances. Spiessens (1991) also reports that small neighbourhoods are preferable, when using tournament selection. Schwehm (1993) describes a mesh connected array processor which can only successfully use elitism for short distances, random selection of chromosomes is preferred for large distances.

Coarse grain genetic algorithms are more complex than fine-grain genetic algorithms because each processor must control a complete population in the coarse-grain case, whereas for the fine-grain case only one chromosome is processed. However the fine-grain case will typically need far more processors. Goldberg's work (1989) suggests that for

© 1994 The Institution of Electrical Engineers.  
Printed and published by the IEE, Savoy Place, London WC2R 0BL, UK.

School of Engineering, University of Wales College of Cardiff, Cardiff U.K.

parallel genetic algorithms the larger the population the faster the GA converges to the optimum result, this is not true of sequential genetic algorithms. Consequently for a single chip VLSI implementation, a large number of simple processors is far more attractive than a few complex processors. If a few complex processors are required then standard devices such as transputers linked together are likely to be highly efficient (Fogarty). This paper will therefore concentrate on the fine-grain GA with massive parallelism on a single chip.

**Vision Systems** Genetic algorithms have been applied in a number of different ways to vision problems. One method is by using a classifier system to identify images. McAulay (1989) used such a classifier system to identify letters of the alphabet. Similarly Ankenbrandt (1990) identified oceanic features via a system based on semantic networks. An alternative approach is to use GAs to solve clustering problems, typically required for automatic inspection. Cucchiara (1993) used GAs to identify suitable clusters. Gaborski et al used GAs to identify a suitable subset of features from a handprint in order to build an efficient classifier system. They successfully reduced the set of binary features required from 1,500 to 300 whilst maintaining the same recognition rate. One of the most unusual applications has been that of identifying criminals by using a genetic algorithm to produce 'faces' which are then given a fitness value by the 'victim'. In this case the image processing is done by the 'victim' and the genetic algorithm is used as an effective method of searching through 'face' space (Cadwell 1991).

All the above techniques are difficult to implement in hardware, the simplest to implement are those based on finding a transformation between a target image and a reference image. Mandava et al (1989) searches for a transform which will warp a 'mask' image ready for subtraction from a 'contrast' image, thus enabling the effective use of digital subtraction angiography for reducing motion artefacts in medical images. One of the problems mentioned by Mandava is that of the time taken to process the image. Starkweather et al (1990) uses a 3D to 2D transformation problem as a particularly difficult test problem for his distributed Genetic Algorithm. This type of problem will be tackled by our proposed PGA.

**Parallel Genetic Algorithm for a vision system** For this paper we will consider the problem of using a GA to identify a transform that maps a two-dimensional external image to a reference image. Once the transform has been identified the position and orientation of the target can be determined. An application may be that of using a vision system to control a robot arm which is attempting to pick up and reorient an object.

The transform used will be:

$$I = R.T \tag{1}$$

Where I is the real image, T is the transform and R is the reference image

The Transform is a 3x2 matrix

$$I(x', y') = R(x, y, I) \begin{pmatrix} S \cdot \cos \phi & -\sin \phi \\ \sin \phi & S \cdot \cos \phi \\ -x_0 & -y_0 \end{pmatrix} \tag{2}$$

Here S-refers to a scaling factor,  $\phi$  to rotation and  $x_0$  &  $y_0$  are position offsets.

The chromosome fitness is measured by the number of matches between corresponding pixels in the real image and the transformed reference image. If the transformed and real image match perfectly then the position and orientation of the object can be determined from the transform. The transform is encoded within a chromosome as six, six bit genes.

This implementation is based on the Fine Grain Parallel Genetic Algorithm model. In order to maximise the number of on-chip processing elements a simple version of the algorithm is preferred. Consequently a variant of Tomassini's 'PCGA0' has been considered with local tournament selection on a grid. All neighbours will be considered in the tournament and the best selected. This selection method is supported by the conclusions of Spiessens et al, provided a neighbour is assumed to be one unit away (figure 1). One variation to the original method is required, namely an optimisation step which tries incrementing and decrementing each parameter by one and keeping the best in order to move each gene towards its local optimum by hillclimbing. Technically this makes the process a hybrid genetic algorithm, with the hillclimbing improving the speed at which local optima are found and the genetic algorithm identifying the global optimum.

The basic algorithm (Tomassini 1993) with the extra optimisation step can be described as follows:

```

Initialise system
while not done do
    evaluate  $f(x_i)$ 
    get  $f^N, f^S, f^E, f^W$ 
    get  $x^N, x^S, x^E, x^W$ 
     $f_i^m = \min\{f^N, f^S, f^E, f^W\}$ 
     $(x_i^m, x_i^m) = x_i \otimes x_i^m$ 
    evaluate  $f(x_i^m)$  and  $f(x_i)$ 
     $f_i^m = \min\{f(x_i), f(x_i^m), f(x_i^m)\}$ 
     $x_i = x_i^m$ 
    mutate  $x_i$  with probability  $p_m$ 
    one step optimise each parameter
end while

```

(Additional step used in this paper, not used by Tomassini)  
 $\otimes$  refers to ordinary two point crossover

**Hardware** For ease of description, the proposed hardware architecture is divided into three blocks (A, B, and C) as shown in figure 3. The first block, block A, is responsible of executing the following tasks :

1. The selection of the best of the four neighbours, with associated chromosomes (Cs, Cn, Cw, Ce) and fitnesses (fs, fn, fw, fe). The best chromosome and its associated fitness will be located in registers REG2 and FREG2 respectively.
2. Crossover and mutation of the above chromosome with the original chromosome used for initialising the processor (CHROM0)

The second block, block B, is responsible for fitness calculation. A given chromosome is separated into the individual transformation parameters. Two identical units are used in order to proceed through the x and y directions in the frame buffer (FB). This architecture avoids the use of multipliers which represent an overhead in terms of area and throughput.

The final block, block C, compares each pixel extracted from the frame buffer (after applying the transform) with the corresponding pixel in the reference image, and with the aid of a counter C3, updates the fitness in register Rft. At this stage the chromosome could either be optimised (each parameter being selected using MX13 and incremented/decremented accordingly) or the fitness will be update in one of the fitness registers in block A depending on the current cycle (optimisation/post crossover fitness calculation).

A detailed description of figure 3 will be the subject of a more comprehensive article.

The performance of the above circuit is considerably fast even with the 1.5 CMOS technology. The circuit processes an individual chromosome in 2-3 milliseconds and would operate at about 8MHz .

**Results** A simulation was used to test the algorithm for three separate images. The algorithm successfully identified the position and orientation of the images in two cases. In one of the cases the match was not perfect, the results obtained were those of a local minimum. Figure 4 shows the improvement of the best and average members of the population for each of the three images over fifty generations. A perfect fit is found when the transformation matches all 4096 pixels of the image. All three images converged to a result within thirty five generations which corresponds to approximately a tenth of a second to recognise an image.

**Conclusions & Future Work** This work clearly shows the potential power of genetic algorithms when applied to image processing. For the simple images chosen and two point cross-over using a fine grain Parallel Genetic Algorithm an excellent match was found in only thirty generations (figure 4). Further work can be done on improving the fitness function, exploring other topologies and crossover operators. Hardware considerations will however limit the types of PGA that can be explored. Using the GA hardware proposed here a position, orientation and scale are estimated to be found in a tenth of a second which makes the GA practical for real-time operation. Further tests need to be done to discover how robust a genetic algorithm is when using images from a variety of real applications. Hence a GA 'Engine' may provide a cheap and powerful means of using this technology for a new generation of cheap vision chips that are suitable for a range of applications.

## References

1. Ankenbrandt C.A, Buckles B.P and Petry F.E (1990) 'Scene Recognition using Genetic Algorithms' *Pattern Recognition Letters* 11 pp 285-293
2. Cadwell C and Johnston V.S (1991) 'Tracking a Criminal through "Face-Space" with a Genetic Algorithm' in *Proceedings of the Fourth International Conference on Genetic Algorithms* Booker B.L, Belew R.K (Eds) Morgan Kaufmann:California pp 416-421
3. Cohoon J.P, Martin W.N and Richards D.S (1990) 'Genetic Algorithms and Punctuated Equilibria in VLST' in *Parallel Problem Solving in Nature* Scwefel H.P & Manner R (Eds) Springer Verlag pp 134-141
4. Cucchiaria R (1993) 'Analysis and Comparison of different Genetic Models for the Clustering problem in Image Analysis' in *Artificial Neural Nets and Genetic Algorithms*, Albrecht R.F, Reeves C.R & Steele N.C (Eds) Springer-Verlag:New York pp 423-7
5. FitzPatrick J.M, Grefenstette J.J and Van Gucht D (1984) 'Image Registration by Genetic Search' *IEEE SouthEastcon* pp 460-64
6. Fogarty T.C and Huang R (1990) 'Implementing the Genetic Algorithm on Transputer Based Parallel Processing Systems' in *Parallel Problem Solving in Nature* Scwefel H.P & Manner R (Eds) Springer Verlag:NY pp 145-49
7. Gaborski R.S, Anderson P.G and Tilley D.G (1993) 'Genetic Algorithm Selection of Features for Hand-printed Character Identification' in *Artificial Neural Nets and Genetic Algorithms*, Albrecht R.F, Reeves C.R & Steele N.C (Eds) Springer-Verlag:New York pp 417-422
8. George-Schleuter M (1990) 'Explicit Parallelism of Genetic Algorithms through Population Structures' in *Parallel Problem Solving in Nature* Scwefel H.P & Manner R (Eds) Springer Verlag:NY pp 150-59
9. Goldberg D.E, (1989) 'Sizing Populations for Serial and Parallel Genetic Algorithms' in *Proceedings of the third International Conference on Genetic Algorithms* Schaffer J.D (Ed) Morgan Kaufmann Publishers pp 70-9
10. Mandava V.R, Fitzpatrick J.M and Pickens D.R (1989) 'Adaptive Search Space Scaling in Digital Image Registration' *IEEE Transactions on Medical Imaging* MI-8 No 3 pp 251-62
11. Manderick B and Spiessens P (1989) 'Fine-Grained Parallel Genetic Algorithms' in *Proceedings of the third International Conference on Genetic Algorithms* Schaffer J.D (Ed) Morgan Kaufmann Publishers pp 428-433
12. McAulay A.D and Oh J.C ' (1989) *Image Learning Classifier System Using Genetic Algorithms* *IEEE* pp 705-10
13. Muhlenbein H (1989) 'Parallel Genetic Algorithms, Population Genetics and Combinatorial Optimization' in *Proceedings of the third International Conference on Genetic Algorithms* Schaffer J.D (Ed) Morgan Kaufmann Publishers pp 416-21
14. Muhlenbein H, Schomisch M and Born J (1991) 'The Parallel Genetic Algorithm as a Function Optimizer' in *Proceedings of the Fourth International Conference on Genetic Algorithms* Booker B.L, Belew R.K (Eds) Morgan Kaufmann:California pp 271-78
15. Petty C.C and Leuze M.R (1989) 'A Theoretical Investigation of a Parallel Genetic Algorithm' in *Proceedings of the third International Conference on Genetic Algorithms* Schaffer J.D (Ed) Morgan Kaufmann Publishers pp 398-405
16. Schwehm M (1993) 'A Massively Parallel Genetic Algorithm on the MasPar MP-1' in *Artificial Neural Nets and Genetic Algorithms*, Albrecht R.F, Reeves C.R & Steele N.C (Eds) Springer-Verlag:New York pp 503-7
17. Spiessens P and Manderick B, 'A Massively Parallel Genetic Algorithm. Implementation & First Analysis' in *Proceedings of the Fourth International Conference on Genetic Algorithms* Booker B.L, Belew R.K (Eds) Morgan Kaufmann:California pp 279-86
18. Starkweather T, Whitley D and Mathias K (1990) 'Optimization using Distributed Genetic Algorithms' in *Parallel Problem Solving in Nature* Scwefel H.P & Manner R (Eds) Springer Verlag pp 176-85
19. Tanese R (1989) 'Distributed Genetic Algorithms' in *Proceedings of the third International Conference on Genetic Algorithms* Schaffer J.D (Ed) Morgan Kaufmann Publishers pp 434-39
20. Tomassini M (1993) 'The Parallel Genetic Cellular Automata: Application to Global Function Optimization' in *Artificial Neural Nets and Genetic Algorithms*, Albrecht R.F, Reeves C.R & Steele N.C (Eds) Springer-Verlag:New York pp 385-391
21. Xu D.J and Daley M.L (1992) 'Design of a Finite Word Length FIR Digital Filter using a Parallel Genetic Algorithm' *Proceedings of the IEEE SouthEastCon 1992* vol 2 pp 834-837



