New parallel Hough transform for circles

R. Chan, BSc, MPhil
W.C. Siu, PhD, DIC, CEng

Indexing terms: Image processing, Picture processing and pattern recognition

Abstract: The Hough transform is a well known medium-level image recognition technique for the detection of curves. The conventional Hough technique [1, 2] requires a three-dimensional accumulator array (AA) for the detection of circles. Because shape parameterisation and data structure of the accumulator array significantly affect the memory space and computation loading requirements of any hardware or software realisation, they should be carefully selected for any fast and efficient algorithm. A new scheme which uses only a pair of two-dimensional accumulator arrays to reduce the storage and computation time by an order of magnitude or more is proposed. This new scheme is capable of discriminating multiple (including concentric) circles in a complex real life image with a recognition rate of 95–100%. Various parallel realisations of this Hough scheme for circles on a general purpose MIMD (a reconfigurable transputer network) machine are discussed and a comparison of their performances with the conventional approach on the basis of execution time and recognition rate is presented.

1 Introduction

The Hough transform (HT) [1] is a method for detecting characteristic contours or shapes by exploiting mutual constraints between parameters and points lying on the target contour. For each target curve or shape S to be detected, the transform uniquely defines a mapping from each point in the image space I to a hypersurface H in the parameter space P. The Hough technique seeks to transform the problem of locating global features (target shape instances) in the image space to that of locating local features (peaks or maxima) in the transformed or, parameter, space. HT is often considered to be one of the most promising techniques for object detection. It has the desirable characteristics of being robust and noise resistant. Moreover, the algorithm is essentially a parallel one, and can be easily modified and ported to multiprocessor environments for parallel realisations. Ballard [2] extended the method to the generalised Hough transform (GHT) to detect nonanalytic curves by using the directional edge information and the R-table associated with each target curve. Recently, Illingworth *et al.* [3] gave a comprehensive and complete review of this transform.

Because circles are completely defined by three parameters, namely the radius and the co-ordinates of the centre (x_0, y_0), the conventional Hough transform requires a three-dimensional accumulator array to detect this family of analytic curves. The transformed hypersurface is an inverted right circular cone in the parameter space. This simple scheme is inefficient in terms of the large memory space and computation effort required. Recently, a dual plane variation of the Hough transform for circles [4] has been suggested. However, this method cannot discriminate concentric circles of different radii. Besides, the suggested transform equations rely heavily on the gradient orientations of two neighbourhood edge pixels, which are difficult to estimate to the required accuracy by computationally efficient edge operators for a noisy input image. Moreoever, floating point operations (including divisions) seem to be unavoidable in the computation of its transform equations as well as the weight of each vote. This is highly undesirable in any fast algorithm for real-time applications. The adaptive Hough transform (AHT) [5] is another Hough approach for detecting circles, but there are interpretation problems when the method is applied to complex images with multiple objects. Besides, the inherent parallel voting process has to be repeated many times for the sequential detection of each of the target shape instances in a single image because this AHT scheme can only adapt its parameter ranges to the detection of one target shape instance in each iteration.

A new and efficient approach is proposed [6], which can give approximately 30 times reduction in the memory space requirement and 50 times reduction in computation effort over the conventional approach. The new scheme replaces the three-dimensional (3-D) accumulator array in the conventional approach with a pair of two-dimensional (2-D) accumulator arrays that is capable of discriminating multiple (including concentric) circles in a complex real life image in a single voting pass. Various parallel realisations of this new scheme have been performed on a general purpose MIMD multiprocessor system which comprises eight T800 transputers and cross-bar switches for network topology reconfiguration. For the initial part of the study, a software simulation of the new algorithm is performed [6] on a single transputer system under the transputer development system (TDS2) using Occam 2. The whole recognition process (including voting, peak detection and circle validation) on a typical 256×256 complex real-life image with multiple objects can be completed in less than 3 seconds with a typical recognition rate of 95–100%.

Although a number of parallel implementations of the Hough transform have already been reported [7, 8], those implementations generally only emphasise the

Paper 8190E (C1, C2), first received 17th July 1990 and in revised form 8th April 1991

R. Chan is with the Department of Electrical Engineering, Imperial College, London SW7 2BT, United Kingdom

W.C. Siu is with the Department of Electronic Engineering, Hong Kong Polytechnic, Hung Hom, Kowloon, Hong Kong

parallel accumulation of votes through different partitionings of the parameter space. The subsequent decision and validation processes based on the accumulated votes are not usually addressed. The new parallel realisations, however, also take into consideration the processing performed before and after the transform, i.e. the initial edge detection for input to the HT and validation of candidate output from the HT, to complete the whole recognition cycle. Instead of simply thresholding the votes or detecting local maxima in the AA, special procedures are included for the validation of each candidate shape instance. This is essential for shape detection in complex images with many objects.

2 New allocation scheme of Hough space

The general equation of a circle is given by

$$(x - x_0)^2 + (y - y_0)^2 = R^2 \tag{1}$$

Instead of using each dimension of the accumulator array independently to detect the corresponding parameter, the new scheme uses two dependent groups of parameters for the detection of circles of any radii. In this approach, the two dimensions of the first accumulator (AA1) represent the x and y co-ordinates of the centres of the predicted circles [9], and those of the second accumulator (AA2) represent the x co-ordinate of the centre, and radius of the predicted circles (Fig. 1). The former is used for locating the centres of the circles and the latter for finding their radii.

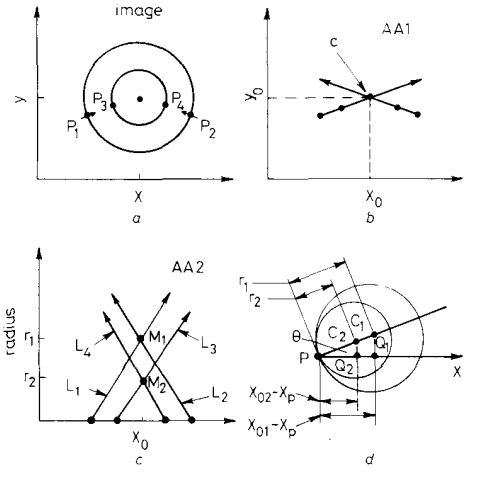

Fig. 1 *New pair of accumulator arrays and locus of cells mapped by edge pixels on concentric circles*

2.1 Derivation of transform equations

To see how these two planes can be used to detect circles, an intuitive view from the geometric standpoint is considered first. An image is assumed to be preprocessed by some edge operators to locate all the local edges that correspond to luminance discontinuity. It is also assumed that the edge operator is capable of estimating the gradient orientation of the edge at a particular point to a certain degree of accuracy (e.g. angular error $< 6^\circ$ for the Sobel operator [10]). Suppose there is an edge pixel P_1 at (x_{P1}, y_{P1}) lying on a circle and its gradient orientation is known. The gradient vector will be pointing towards the centre of the circle. In other words, the locus of the possible centre will be a straight line passing through the point P_1 with a slope equal to its edge gradient orientation as shown in Fig. 1*b*. If there are many edge pixels lying on a circle (one of which is the point P_2, say, in Fig. 1*a*), their corresponding mapped lines in the first plane (AA1) will all intersect at a single point located exactly at the centre of the circle.

The radius of the circle is found by using the second plane. Let us first consider two circles of radius r_1 and r_2 passing through a single point P as shown in Fig. 1*d*. It can be easily seen that the triangles PC_1Q_1 and PC_2Q_2 are similar and thus the following relation holds

$$(x_p - x_{02})/(x_p - x_{01}) = r_2/r_1 \tag{2}$$

Because r_1 and r_2 are arbitrarily chosen, eqn. 2 implies that the locus of the possible radius will also be a straight line on the second plane (AA2). For the four edge pixel points P_1, P_2, P_3 and P_4 in Fig. 1*a*, the mapped loci in Fig. 1*c* will be the lines L_1, L_2, L_3 and L_4, respectively. It can also be seen that edge pixels on the same circle will intersect at a single point (L_1 and L_2 intersect at (x_{01}, r_1), and L_3 and L_4 intersect at (x_{01}, r_2)). Hence, if the possible centre of a candidate circle detected using AA1 is (x_{01}, y_{01}), the radius of the circle can be found by locating peaks of the column with $x_0 = x_{01}$ in AA2. In the case of concentric circles, edge pixels on each concentric circle will contribute to a single peak in AA2. This results in multiple peaks with the same x_0 but different radii in the second accumulator (M_1 and M_2 in Fig. 1*c*). Hence, the discrimination power of concentric circles in the conventional three dimensional AA approach is retained in this new scheme but the storage requirement for the AA is much reduced.

The following is a formal derivation of the transform equations. The slope of the tangent at a point $P(x_p, y_p)$ on the circle can be found by differentiating eqn. 1 with respect to x to give

$$dy/dx = m_t = -(x_p - x_0)/(y_p - y_0) \tag{3}$$

The slope of the normal to the circle at the same point P can be found by the Sobel or super-Sobel operators [10] as

$$m_n = \tan\theta = (S_y/S_x) \tag{4}$$

where S_x and S_y are the gradient components in the x and y direction, respectively. From co-ordinate geometry, the product of the slope of the normal and tangent at a point is negative unity. Thus, for the point (x_p, y_p) with gradient components S_x and S_y, the equation describing the locus of the centre of the circle is given by

$$(y_p - y_0)/(x_p - x_0) = (S_y/S_x) \tag{5a}$$

or

$$y_0 = mx_0 + b \tag{5b}$$

where

$$m = \text{slope} = (S_y/S_x)$$

and

$$b = y_0\text{-intercept} = y_p - x_p * (S_y/S_x)$$

This is a straight line on the x_0–y_0 plane (AA1) of the parameter space. One of the endpoints of this line is the point (x_p, y_p) itself. If the radii of the circles to be

detected are constrained by an upper bound R_{Vmax}, and the gradient angle θ (eqn. 4) is quantised into N_θ values, the co-ordinates of the other endpoint will be

$$y_{0max} = y_p + B_{Vmax} \sin(\theta_k) \tag{6a}$$

$$x_{0max} = x_p + R_{Vmax} \cos(\theta_k) \tag{6b}$$

where

$$\theta_k = k(2\pi/N_\theta) \quad \text{and} \quad k = 0, 1, \ldots, (N_\theta - 1)$$

Note that the values of sines and cosines in eqn. 6 can be precomputed, scaled up (by a power of two) and stored as integer constants. Besides, the upper bound R_{Vmax} in eqn. 6 is actually a function of the edge pixel location (x_0, y_0) and the edge gradient orientation θ. This is due to inherent as well as to imposed geometric constraints on the target circles in the image. A simple, yet reasonable constraint is that all the target circles, even occluded, have to reside totally within the 2-D image space. This assumption also helps to increase the efficiency of the vertification process, and will be discussed in the next section.

The equation of the line in the second plane can be represented as

$$(x_0 - x_p)/(r - 0) = [S_x/(S_x^2 + S_y^2)^{1/2}] \tag{7}$$

The endpoints can easily be found as follows: (i) x_0 should be equal to x_p when the radius is zero; (ii) x_0 equals x_{0max} (given by eqn. 6*b*) when R equals R_{Vmax}. In summary, the loci of the two line segments on the two planes are:

$$\text{AA1}: (x_p, y_p) \rightarrow (x_{0max}, y_{0max})$$

$$\text{AA2}: (x_p, 0) \rightarrow (x_{0max}, R_{max}) \tag{8}$$

where $\rightarrow$ denotes the line joining the two endpoints.

2.2 Justification of this approach

Owing to the compression of the parameter space and the linear nature of the mapped loci, high efficiency and parallelism can be achieved in the new scheme. Detailed implementation techniques will be discussed in Section 3. The remaining part of this section will focus on the interpretation problem and the justification of this approach.

Upon completion of the voting process, each cell holds the number of votes or evidence of the hypothesised shape with the parameter set associated with that cell. If there is only one object in the image, the cell with the highest count will define the most probable parameters of the object of the target shape, if it exists, in the image. However, a cell with a high count of votes exceeding the neighbourhood cells (i.e. a local maximum) is a necessary, but not a sufficient, condition for the presence of a target shape instance. In other words, there may be some peaks in the parameter space which are contributed by noise or by edge pixels which are not on the same object as the target shape. Even though the Hough transform can be considered as a matched filter of shape recognition [11], it is not optimal for recognising one shape in the presence of others. This interpretation problem of the transformed space becomes more serious if there are many objects in the image while the target shape is very primitive, such as line segments, circular arcs, etc. Hence, in a practical realisation with a complex input image consisting of multiple objects, there may be many peaks in both x_0 columns in AA1 and AA2. Since the two planes are linked by the same parameter axis x_0, an additional procedure has to be carried out to select the correct pairs of peaks, one from each plane, so that the complete sets of three parameters of the candidate circles can be found. At first sight this seems to make the new scheme complicated, because the regular structure of the voting algorithm has to be modified. In fact, this new scheme only merges the process of verifying candidate shape instances into the voting process.

Study of the conventional Hough transform for circles using a 3-D accumulator array, has shown it to be difficult to determine the response level (i.e. the number of votes) above which the shape instance is guaranteed to be present. Conker [4] handled this problem of detection criteria by calculating a relative response which was equal to the ratio of the accumulator response to the circumference. Ye [12] used a similar strategy involving a radius-dependent threshold. In our software realisation on the conventional approach, however, the number of false alarms was unacceptably high even with some radius-dependent thresholding schemes were used. Hence, unless the image consists of only very few objects (one or two say, which is very unlikely) it is absolutely necessary to include some additional procedures in the conventional scheme to select from among the peaks in the image those corresponding to the actual occurrence of circles. Based on this argument, merging of the voting and the verifying processes can be justified.

A similar technique has already been applied by Davies [13] on a high speed circular-object location algorithm which is based on bisecting horizontal and vertical chords of a circle. Candidate circle centres are located by a 2-D AA, and candidate validation and determination of radius of each hypothesised circle are performed using a 1-D AA. However, this second stage of validation has to transform all the edge pixels once for each candidate centre, which is too demanding for real time applications with complex scenes. Milenkovic [14] treated higher dimensional parameter space as a search space and suggested two variable resolution search techniques to find sets of parameters in this search space that best fit the image data. A point-feature-distance (PFD) function was defined to guide the search so that maxima in the parameter space could be determined without histogramming. Because of the inclusion of the PFD function, the parameter set output must represent the actual target shape instance in the image space. The dynamic combinatorial Hough transform (DCHT) of Leavers *et al.* [15] and the elegant backmapping approach of Gerig *et al.* [16] also tried to provide linkage between image space and parameter space so that subsequent parameter space interpretation and hypothesised instance validation could be performed more easily. However, the computational loading of the DCHT increases drastically for an image with a complex scene, and that of the backmapping approach is at least twice that of the conventional HT using a 3-D AA although the actual votes cast are reduced. Also, the DCHT is essentially a serial algorithm and parallel realisation seems to be very difficult. The backmapping approach has to sacrifice the power of discriminating concentric circles unless no projection is performed and two 3-D AAs are used.

In the present studies, an attempt was made to find a new and extremely efficient scheme which possessed the following characteristics:

(*a*) no 3-D AA needed
(*b*) concentric circles detectable
(*c*) floating-point operations minimised
(*d*) parallel realisation easily performed.

Davies's approach [9] of locating circles of different radii on a single plane was followed and it was found that the projection of votes at the other orthogonal direction (along either x_0 or y_0 axis) could give a scheme with such characteristics.

To handle the peak pairing and Hough space interpretation problem more efficiently, it is suggested here that the input image data are partitioned and grouped systematically and efficiently during the voting process. In this scheme, an extra database in the form of numerous dynamic linked lists is built when votes are being accumulated in the two AAs. This database is just a compact and partial representation of the whole image space. The hypothesised parameters associated with each pair of peaks can define an inverse mapping back into this image space which is then searched. Some decision criteria are then defined to validate the presence of the target shape and to reject the false alarms. In this respect, the Hough transform can be considered as a method to extract a more probable and more compact set of estimated parameters from the complete parameter space for more efficient subsequent verifications.

A comparison of the conventional scheme and the new scheme is given in Table 1. In our software realisation the algorithmic parameters are defined as follows:

$$N_x = 64 \qquad N_y = 48 \qquad N_R = 36$$

This implies that resolution of the circle centre coordinates is four pixels in both x and y directions, and resolution of the radius is two pixels. Hence, the upper bounds of the theoretical speedup factor for voting and searching are 46.6 and 66.9, respectively. There is also a storage saving factor of 20.6 for the accumulator array.

Table 1: Comparison of conventional and new scheme

	Conventional approach	New approach
No. of cells in accumulator array	$N_x N_y N_R$	$N_x N_y + N_x N_R$
Computation complexity of voting in AA	$O(N_E N_R(N_R+1))$	$O(N_E(N_R+N_x))$
Computation complexity of searching in AA	$O(N_x N_y N_R(3^3-1))$	$O((N_x N_y + N_x N_R)(3^2-1))$

N_x = number of cells in x_0 dimension of AA1 and AA2
N_y = number of cells in y_0 dimension of AA1
N_R = number of cells in radius dimension of AA2
N_E = number of edge pixels in the input image
$O()$ = order of number of operations in the worst case

3 Software realisation

The verification strategy, the imposition of certain geometric constraints and some fast algorithms applicable to the new scheme in actual realisation will be discussed. The block diagram of the proposed scheme is shown in Fig. 2. There are three independent databases involved in the whole scheme. The two accumulator arrays (AA1 and AA2) have been explained clearly in the preceding section. The third database needs to be elaborated upon here.

As stated in Section 2, an extra database has to be built for subsequent pairing up of the peaks detected in the corresponding planes. If there is enough memory available in the system, this database can be declared to be a 2-D array which stores the gradient orientations of all the edge pixels in the gradient orientation array (GOA). However, all the gradient orientations of these edge pixels can be grouped to form a subspace of the complete image space. This arrangement is more reasonable and efficient, in terms of storage requirement, because the size of the subspace to be built is proportional to the complexity of the input image. In the initial

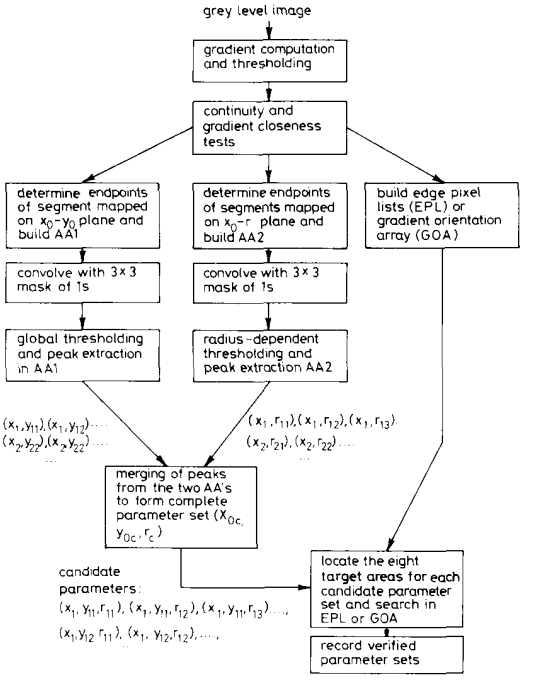

Fig. 2 *New Hough circle detection scheme*

serial software realisation on the PC-AT compatible machine, this database consists of eight dynamic linked lists (called edge pixel list or EPL), each of which stores the edge pixels with gradient orientation equal to an integral multiple of 45°, i.e.

$$L(k): \{(x_{kl}, y_{kl})$$

where

$$|S_\theta(x_{kl}, y_{kl}) - 45k| < \theta_{toler}\}$$

and

$$k = 0, \ldots, 7 \quad l = 1, \ldots, 1_{max} \tag{9}$$

Peaks extracted from the two accumulator arrays are merged to form the complete candidate parameter sets. In the merging process each peak in AA1 (say (x_1, y_{11}) marked bold in Fig. 2) is paired with every peak $\{(x_1, r_{11}), (x_1, r_{12}), (x_1, r_{13}), \ldots\}$ in the corresponding column in AA2. For each of these peak pairs found, the set of estimated parameters of the candidate circles is used to define eight target points. They are on the circumference of the candidate circle and are also oriented at an integral multiple of 45° from the candidate circle centre and are designated as $T(i)$ where $i = 0, \ldots, 7$. For each target point $T(i)$, the linked list $L((i + 4)\ Mod\ 8)$ is searched for any element whose distance from the target point is below a certain distance threshold. Circles are declared if elements are found in more than a certain number

Table 2: Algorithm of proposed scheme in pseudo-codes

```
Begin
  Repeat
    Calculate S_x, S_y and S_mag := f(S_x, S_y);
    If S_mag ≥ S_thres Then
    Begin
      S_θ := Round (tan^-1(S_y/S_x));
      If (|(S_θ Mod 45) - 45| ≤ θ_toler)
      Then
        Update EdgePixelList(k): k = 0, ..., 7;

      AA1(q) := AA1(q) + 1 where q: (x_p, y_p) - (x_0max, y_0max);
      AA2(q) := AA2(q) + 1 where q: (x_p, 0) - (x_0max, R_max)
    End
  Until EndOfImageData;

  Convolve AA1 & AA2 with 3 × 3 mask of 1s;
  i := 0;

  Repeat
    Locate q_i = (x_0i, y_0i) where [AA1(q_i) - AA1(q_i + d_q)] > 0;
    j := 0;
    Repeat
    Locate z_j = (x_0j, R_0j) where [AA1(z_j) - AA2(z_j + dz)] > 0;
      Match := 0;
      For k := 0 to 7 Do
      Begin
        x_s := x_0i + R_0j cos (k * 45);
        y_s := y_0i + R_0j sin (k * 45);
        n := 1;
        Repeat
          If EdgePixelList(k) contains (x_l, y_l) such that
            |x_kl - x_s| < Δx_toler AND |y_kl - y_s| < Δy_toler
          Then
            Match := Match + 1
          Else
            n := n + 1
        Until (EndOfList(k) OR OnePixelFound)
      End;

      If Match ≥ Match_thres
        Then DeclareCircleDetected (x_0i, y_0i, R_0j);
      j := j + 1
    Until AA2_NoMorePeak;
    i := i + 1;
  Until AA1_NoMorePeak
End.
```

(Match$_{thres}$ in Table 2) of the eight lists. Although this approach is simple, it is sufficiently accurate and efficient to eliminate most of the invalid pairs, giving the scheme a recognition rate of 95–100%. More robust and reliable tests can be taken on the whole circle to reduce the false alarm rate.

To further increase the accuracy and speed of the algorithm, several mechanisms have been included. First, local continuity and closeness of edge gradient of neighbourhood edge pixels are used to restrict and even disallow votings from these edge pixels that are not likely to lie on a circle.. Typically, about 25–40% of the edge pixels are disallowed to vote. Moreover, circles are also assumed to be completely within the image (even if they are partially occluded), so that the value of $R_{V_{max}}$ in eqns. 6*a* and 6*b* is adjusted according to the location of the edge pixel (x_p, y_p) and to the gradient orientation θ given by

$$\begin{aligned} R_{Vx} &= |x_p/\cos\theta| \quad \text{if } x_p < |R_{max}\cos\theta| \\ &= |(x_{max} - x_p)/\cos\theta| \\ &\qquad \text{if } x_p > |x_{max} - R_{max}\cos\theta| \\ &= R_{max} \qquad \text{otherwise} \end{aligned} \tag{10a}$$

$$\begin{aligned} R_{Vy} &= |y_p/\sin\theta| \quad \text{if } y_p < |R_{max}\sin\theta| \\ &= |(y_{max} - y_p)/\sin\theta| \\ &\qquad \text{if } y_p > |y_{max} - R_{max}\sin\theta| \\ &= R_{max} \qquad \text{otherwise} \end{aligned} \tag{10b}$$

$$R_{V_{max}} = MIN(R_{Vx}, R_{Vy}) \tag{10c}$$

This geometrical constraint can further reduce the computation required for the verifying process. If a peak is found at (x_c, y_c) in the first plane, the maximum radius to be searched in the image space will be R_{Smax} given by

$$R_{Sx}(x_c) = MIN(R_{max}, x_c, |x_{max} - x_c|) \tag{11a}$$

$$R_{Sy}(y_c) = MIN(R_{max}, y_c, |y_{max} - y_c|) \tag{11b}$$

$$R_{S_{max}} = MIN(R_{Sx}, R_{Sy}) \tag{11c}$$

Two very small lookup tables can be built offline to store $R_{Sx}(x_c)$ and $R_{Sy}(y_c)$ for more efficient determination of R_{Smax} during runtime.

To reduce the dispersion and to give more peaked responses for subsequent peak detection, the two accumulator arrays are filtered by convolving with a 3 × 3 mask of all 1s upon completion of the voting process. The inclusion of this process is justified as it significantly reduces the number of subsequent searches, thus reducing the overall recognition time. In addition, the execution time for this process is fixed (0.45 s in our serial realisation on the PC-AT compatible machine), and can be reduced still further by simple hardware because of the regularity of the process and simplicity of the kernel.

Because the loci of both transformed curves on the two parameter planes are straight lines, some fast techniques for computing the co-ordinates or addresses of the cells approximating these two lines can be applied. A modified version of the Bresenham algorithm [17], which was originally designed for line drawings in computer graphics, is included in our software realisation. The major advantage of the algorithm is that it involves only integer arithmetic.

Note that the arctangent function has to be computed to estimate the gradient orientation of each edge pixel. Since there is inherent estimation error from the edge operator itself [10] and the valid orientation is quantised, an output precision to the nearest 2° is sufficient. In fact, all quantities concerning the orientation or angular measures have a resolution of 2° in all our actual realisations. With this tolerance allowed, it is thus possible to implement the arctangent function by pure integer arithmetic using the 'bisect and compare' method [18]. Only a small table of 23 precomputed integer constants is needed, and a maximum of five iterations is required to complete the arctangent function.

The serial software realisation is carried out on a PC-AT compatible personal computer, and the program is completely written in Pascal. An outline of our realisation using pseudo-codes is given in Table 2.

4 Parallel realisations

Many low-level image processing algorithms (e.g. median filtering, correlation, etc.) have been implemented on transputer networks [19]. It was found that image parallelism, in general, provides better performance improvement over task parallelism on a transputer network. In this section, the potential of task parallelism of the Hough transform, a medium-level image processing task, is investigated through the parallel realisations of the new Hough scheme for circles on the transputer network.

All the parallel realisations are carried out on a multiprocessor system, called Superlink, which is a second generation transputer system that has been designed and implemented by the same group of research workers within the department. It comprises eight T800 transputers running at 17.5 MHz, with an IBM AT compatible machine acting as the host. The topology of the

network is software reconfigurable through the 32 × 32 crossbar switching network so that each link of a transputer can be connected to any other link of another transputer. Each transputer has 128 Kbytes of external memory, and link 0 of the root transputer ($T0$) is connected to the host. The host acts as a terminal for user interface, and provides secondary storage for the network. All communication with the host is carried out via DMA channel on the host computer. All the programs are written in Occam 2 and developed under the transputer development system (TDS V2.0). However, system programs running on the host are written in both 80286 assembly language and in C programming language to handle all the necessary I/O between the network and the host.

4.1 Choice of architecture

As pointed out in Section 1, a MIMD architecture is used for the parallel realisations of the new Hough scheme. The reasons for this choice are threefolded. First, the computation loading required for each edge pixel is not fixed and is a function of $R_{Vmax}(x_p, y_p, \theta)$ and $R_{Smax}(x_c, y_c)$. The number of edge pixels for different images is also indeterminate until runtime. These imply that any systolic or finegrain pipeline approaches will not be appropriate.

Secondly, the mapped loci of different edge pixels are located on the two planes at random, and are dependent on the input image data. If the co-ordinates (x_{0i}, y_{0i}) of the cells to be incremented are calculated by one group of processors, and routed to another group of processors which actually perform the voting (i.e. incrementing the vote count), there will then be excessive amounts of data (co-ordinates of cells mapped) moving around the networks. Besides, explicit tests on whether the routed co-ordinate pairs are within the local partition of the processor may be required. Moreover, the workload of the voting processors will not be balanced. Otherwise, there will not be any peak vote count in the two accumulator arrays. Hence, a SIMD architecture in which each processor (or PE) caters for one specific region of the parameter space is not suitable because data movements between processors are not structural, processor workloads will not be balanced and interprocessor communcation overhead will be intolerably high because of bus or link contention.

Thirdly, major computation involved in the voting process is on the determination of the addresses of the cells to be incremented. Although the modified Bresenham algorithm can efficiently compute the addresses of the cells approximating the two line segments mapped on the pair of planes, its inclusion has also constrained the network topology for efficient parallel realisation. This is because, for each of the mapped locus (i.e. each of the linear segments), there is a fixed overhead for the initialisation of the loop in the Bresenham algorithm. The longer the segment mapped on the two planes, the smaller the proportion of this loop overhead and the higher the efficiency of the address generation algorithm. Hence, if the two planes are partitioned into rectangular regions, each of which is managed by one processor, then each original mapped segment will be subdivided into still shorter segments, making the initialisation overhead of the addressing scheme larger and larger. Moreover, additional overheads of determining the endpoints of each subsegment at the local partition boundaries may further lower the efficiency of the whole addressing scheme.

To minimise interprocessor communication and to take full advantage of the Bresenham algorithm, the whole plane of each accumulator array will be kept in the same processor. This implies that for each of the line segments mapped, the co-ordinates of the two endpoints will be distributed to a processor that casts votes to all the cells lying on the mapped linear segment joining these two points. Different mapped segments can therefore be processed by different processors at the same time in a multiprocessor system. After the whole voting process has been completed, the votes in the 2-D arrays managed by various processors have to be summed to get the total vote count for each cell. If the distribution and the routing strategies are efficient, this task parallel scheme can potentially achieve a linear speedup with the number of voting processors. This argument of linear speedup is, of course, valid only if subsequent summation of the distributed votes to give the complete parameter space has a negligible contribution (as compared to the address generation and actual incrementation) to the total execution time. However, it should be emphasised that the computation loading of this subsequent processing is fixed for different input data and is dependent solely on the network topology.

In all of the parallel realisations, the linked lists defined in eqn. 9 are replaced by a two-dimensional array whose elements hold the gradient orientation of all the edge pixels. During the verification, however, the strategy of searching for eight target points as an initial test is still used to reject most of the false alarms. In addition, this gradient orientation array can be utilised for further verification of the candidate circles which have passed the initial test.

4.2 Network topologies

For the parallel realisations of the new Hough scheme for circles, the network has been configured in various topologies as shown in Fig. 3. They are designed so that the four different sets of data, namely the input image data, the two accumulator arrays and the gradient orientation array, are separately managed by different transputers to achieve minimum interprocessor communication. It should be noted that all three topologies are modified from the standard binary tree structure.

In the first topology with three transputers (Fig. 3*a*), $T0$ manages the input image data (or more precisely the edge pixel data) and the gradient orientation array. It determines the two pairs of endpoints of the two mapped segments (given by eqn. 8) and sends them to $T1$ and $T2$ which manage, and cast votes in, AA1 and AA2, respectively. The two transputers, $T1$ and $T2$, will perform subsequent filtering, thresholding and peak detection on the corresponding plane, and transmit all the peaks found to the root transputer. Verification of the candidate circles are then performed by $T0$ by searching the gradient orientation array, and verified circle parameters are transmitted to the host.

The second topology (Fig. 3*b*) is modified from the first one with two additional processors $T3$ and $T4$ sharing the workloads of $T1$ and $T2$, respectively. In other words, $T1$ and $T3$ will cast votes in the first plane while $T2$ and $T4$ will cast votes in the second. A software demultiplexor (in the form of a circular link buffer set at the higher priority level in a PRI PAR construct) is added in $T1$, so that segment endpoints received from $T0$ are passed to $T1$ or $T3$, whichever is idle or is ready to perform voting for the next mapped segments. Although this dynamic runtime scheduling arrangement introduces

additional overhead it helps to maximise the parallelism and equalise the workloads of the two processors because the computation loading for processing segments of different lengths are different and lengths of the mapped segments are indeterminate until runtime. Hence, compile-time scheduling will always cause the processors to be idle, waiting for the next pairs of endpoints to arrive. The same mechanism has been arranged in $T2$ so that endpoints are fed to either $T2$ or $T4$, whichever is ready to cast votes for the next mapped segments.

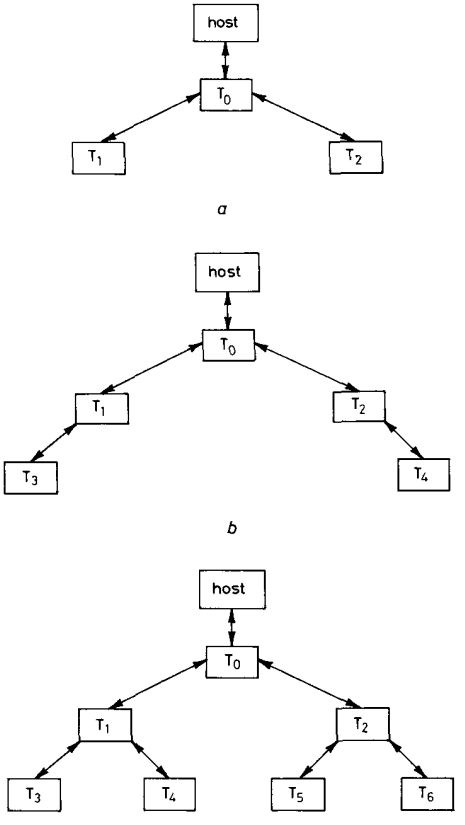

Fig. 3 *Network topologies for parallel realisations*

The third topology (Fig. 3*c*) enhance the dynamic scheduling arrangement by having three processors ($T1$, $T3$ and $T4$) for casting votes on the first plane and another three ($T2$, $T5$ and $T6$) for vote accumulation on the second. After the voting has been completed, the whole 2-D arrays processed by $T3$ and $T4$ ($T5$ and $T6$ respectively) are sent to $T1$ ($T2$ respectively) so that summation of corresponding cells can be performed to give the final transformed space.

The parallel verification process has also been realised in the network of the second topology (Fig. 3*b*). Two different schemes have been studied. The first scheme (see Fig. 4*a*) uses all five transputers for parallel verification. Upon completion of the voting process, while the whole 2-D accumulator array processed by $T3$ and $T5$, respectively, are sent to $T1$ and $T2$, respectively, to be summed to give the final transformed space, the root transputer will concurrently transmit the gradient orientation array to the other four transputers in the network. All the peaks extracted in the two planes are transmitted to transputer $T1$, so that candidate circle parameters are extracted and distributed dynamically to any one of the five processors in the network, whichever is the first to be ready to verify the next candidate set. The physical links and the data routing are arranged (Fig. 4) so that only one type of data (i.e. either candidate parameter, verified parameter, peaks in AA2 or processor-ready signal for the next candidate parameter) is transmitted in any one link throughout the whole verification process.

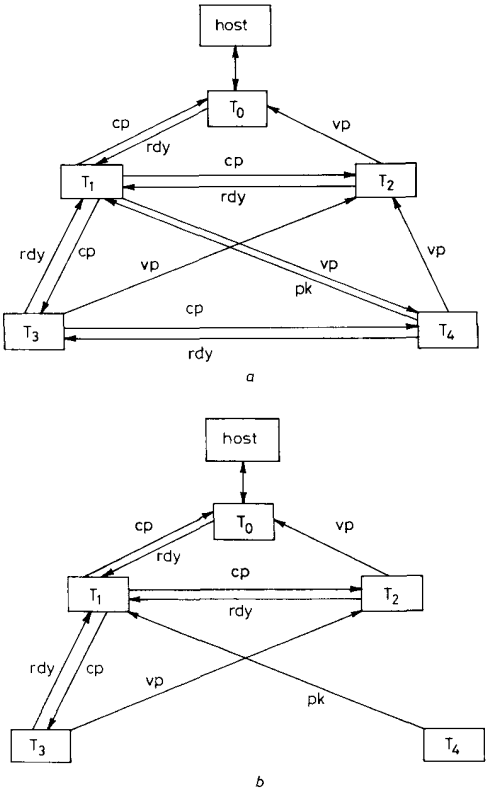

Fig. 4 *Data routing for parallel verification*

cp = candidate parameter; rdy = processor ready for next cp
vp = verified parameter; pk = peaks in AA2

The second scheme uses only three transputers ($T0$, $T2$ and $T3$) for parallel verification. The gradient orientation array will be distributed to $T2$ and $T3$ instead of all four transputers. $T4$ will be dedicated to peak detection in AA2 while $T1$ extracts peaks in AA1 and distributes the candidate circle parameters dynamically to any of $T0$, $T2$ or $T3$. The physical link connections are basically the same as the first scheme, but the data routing paths have been modified as shown in Fig. 4*b*.

5 Results

The results of four test images (see Fig. 5) are given. The first image is generated by computer and the other three are real life images acquired from a camera. The resolution of all the test images is 256 × 256 pixels with a depth of 8 bits per pixel.

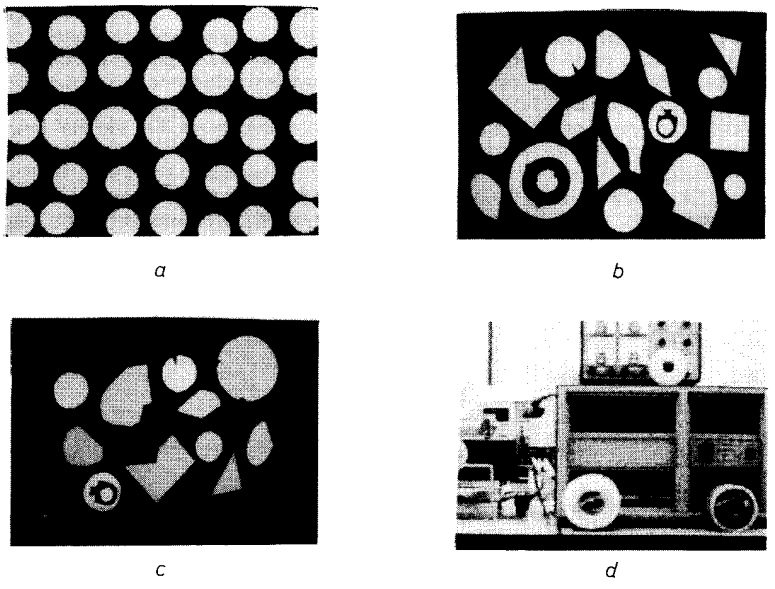

Fig. 5 *Test images 1–4*
a Image 1 (35cir) *b* Image 2 (conccir) *c* Image 3 (circles) *d* Image 4 (lab1)

As a single system which is capable of performing all levels of processing tasks (e.g. numeric and symbolic processing) in image processing and computer vision, a transputer system will be a good choice for design and simulation of parallel algorithms. Digital signal processors, which are usually more cost effective and more suitable for low-level processing tasks (e.g. convolution, edge detection, etc.) [20], may be added as the front end of the transputer network to form a real-time or near real-time system. Hence, all the timings for the voting process do not include the extraction of edge pixels (convolution with gradient masks and gradient thresholding) and the determination of the gradient orientation.

5.1 Serial realisation

A comparison of the execution time for the voting process between the conventional and the new scheme is given in Table 3. Because of the large memory requirement of the conventional scheme, this part of the realisation was performed on an INMOS B004 board with a single T414 transputer so that a fair comparison could be made. The new approach achieved about 25 times speedup over the conventional approach. The discrepancy between this experimental speedup and the theoretical speedup (46.6 and 66.9 mentioned in section 2) is due to the overhead for building the addition database (the eight edge pixel lists defined in eqn. 9, or more precisely the gradient orientation array in our realisation), and also to the use of the efficient Bresenham algorithm for circles for calculating the addresses of the cells on the mapped plane. Because of the memory limit of the computer, the maximum detectable radius is set to 72 pixels ($N_R = 36$) in the conventional approach. Although N_R of the new scheme can be set higher in our realisation ($N_R = 50$ or larger), it is reduced to this same value of 36 for the sake of performance comparison. It should be noted that the speedup factor will increase if N_R is increased, making the overhead for building the extra database (the eight linked lists or the two dimensional gradient orientation array) less significant relative to the actual voting process.

Table 4 gives a comparison of the execution time for the searching process between the conventional and new schemes. Special care must be taken in interpreting the results in this table. After employing radius dependent thresholding and strict maxima criteria (i.e. the vote count of a cell is larger than all the 26 neighbourhood cells in the 3-D AA), the number of peaks detected in the conventional approach is the same as the number of candidates circles which are not yet validated. Thus it takes 4.36 seconds to extract 935 candidate circles for test

Table 3: Execution time for voting by single T414 on B004 board

Test image	Number of edge pixels	Conventional approach	New approach	Speedup factor
35Cir*	7340	112.3 s	4.51 s	24.9
Conccir	4273	79.5 s	2.91 s	27.3
Circles	2109	40.3 s	1.49 s	27.0
Lab1	8950	130.2 s	5.36 s	24.3

35Cir*: Synthesised image with 35 (29 complete) circles of different radii;
Conccir: Image with four concentric and five other circles;
Circles: Image with two concentric and four other circles;
Lab1: Scene of laboratory with three circular objects.

Table 4: Execution time for peak detection and searching by single T414 on B004 board

Test image	Conventional approach		New approach		
	Number of peaks	Time for locating peaks	Number of search	Time for searching	Recognition rate
35Cir*	935	4.36 s	223	1.03 s	96.5%
Conccir	620	2.66 s	198	0.89 s	100%
Circles	260	1.47 s	137	0.58 s	93.3%
Lab1	1053	5.79 s	375	1.28 s	100%

image 1, where in fact there are only 29 complete circles actually present. Thus, further testing procedures (and extra processing time) are needed to process these 935 candidate circles. On the other hand, the new scheme spends 1.03 seconds to locate exactly 28 out of the 29 complete circles in only 223 search trials. Note that although there are 35 circles in this image, the searching constraint defined by eqn. 11 makes detectable only the 29 circles which reside completely in the image and the scheme totally ignores the remaining six incomplete circles located near the boundary of the image.

5.2 Parallel realisations

The absolute speedup of the parallel algorithm over the serial counterpart is defined as

$$S = t_1/t_n \tag{14}$$

where

t_1 = execution time with one processor

and

t_n = execution time with n processors.

The efficiency of the parallel algorithm with n processors can thus be defined as

$$\eta = S/n = t_1/(t_n * n) \tag{15}$$

The speedup of the parallel voting and verifying algorithms with different network topologies are studied and tabulated in Table 5 and Table 6*b* respectively. The job allocation of different processors in the parallel verifying process is given in Table 6*a*. There is a continual increase in overall speedup with the increasing number of processors. Although the efficiency decreases with the increasing number of processors dedicated to voting, its value of about 70% (in the range 60.9%–66.6%) is quite satisfactory for these parallel realisations in high level language without any code optimisation.

The fixed overhead in Table 6*b* refers to the convolution of the accumulators with a 3 × 3 mask of unity, the vote thresholding and the distribution of the gradient orientation array. It is measured by setting the gradient threshold to a very high value so that no pixel is allowed to contribute votes to the accumulators and to be involved in any subsequent processing.

The efficiency of the parallel verification is not very high. This is because the computation loading for the peak extraction is quite demanding so that the processors are very often idle, waiting for the candidate parameters to arrive. This can be observed by inspecting the use of the software buffer which distributes the candidate parameters to any processors which are ready. The fact that the second scheme has both higher efficiency and faster absolute execution time is because $T0$ and $T4$ are totally dedicated to peak detection in the corresponding accumulator and released from concurrently performing the verification process. This can be deduced from the fact that the fixed overhead has decreased by 32 ms while the absolute execution time is decreased by a still larger amount (in the range of 55–76 ms).

The efficiency is especially low if only a very small number of search trials is attempted because the fixed overhead of transmitting the gradient orientation array will become more significant. However, this array can be built by individual processors during the voting process so that distribution of the whole array after the voting process can be eliminated.

It should be stressed that the above absolute timings are for parallel programs with no optimisation. The total execution time can, of course, be further reduced by using faster transputers, and including inline transputer instructions (under the GUY construct [21]) for time-critical sections of the program.

After completing the software realisations of the algorithms described in the last section, it can be confirmed that balancing of the workload of each transputer in networks of this nature is of utmost importance for achieving maximum efficiency. The major factors affecting the efficiency are:

(*a*) *Data availability:* Since there is an indeterminate amount of data (edge pixels) which are not arriving at regular intervals, the workloads of the various transputers are fluctuating with time, with the input parameters (e.g. gradient threshold) and with the image data. These workload gaps can be balanced to a certain extent by implementing software link buffers between processors and this is elaborated in the following paragraphs.

(*b*) *Buffered communications.* The interprocessor communication between transputers via links is synchronised and unbuffered. Hence, a software buffer process may be

Table 5: Execution time for voting using different number of transputers (numbers in brackets indicate speedup with respect to one processor)

Test image	Number of edge pixels	1 T800	3 T800s	5 T800s	7 T800s
35Cir*	7340	2.96 s (1.0)	1.59 s (1.86)	0.90 s (3.28)	0.66 s (4.48)
Conccir	4273	1.90 s (1.0)	1.03 s (1.84)	0.58 s (3.28)	0.41 s (4.63)
Circles	2109	0.95 s (1.0)	0.52 s (1.83)	0.29 s (3.28)	0.21 s (4.52)
Lab1	8950	3.53 s (1.0)	1.88 s (1.87)	1.06 s (3.33)	0.79 s (4.47)

Table 6*a*: Processors job allocation for different schemes

	Scheme 0	Scheme 1	Scheme 2	Scheme 3
Total number of processors	1 T414	3 T800s	5 T800s	5 T800s
Processors for peak detection	1	2	2	2
Processors for verification	1	1	5	3

Table 6*b*: Execution time for verifying using different number of transputers

Test image	Number of search trial	Scheme 0 1 T414	Scheme 1 3 T800s	Scheme 2 5 T800s (Fig. 5.4*a*)	Scheme 3 5 T800s (Fig. 5.4*b*)
35Cir*	223	1.03 s	0.547 s	0.322 s	0.256 s
Conccir	198	0.891 s	0.418 s	0.287 s	0.211 s
Circles	137	0.582 s	0.267 s	0.203 s	0.148 s
Lab1	375	1.28 s	0.646 s	0.340 s	0.268 s
Fixed overhead	—	91 ms	49 ms	97 ms	65 ms

added to equalise the gap between the varying workloads of two communicating transputers. However, the optimum size of the buffers is usually dependent on the algorithm, the image data and the memory usage in the system. It was found that the software buffer could reduce the total execution time by about 5–15%. The actual saving varies, of course, with the workload difference among communicating processors, which is, in turn, dependent of the input data themselves.

(c) *Workspace utilisation.* The Occam language implementation allows data arrays to be forced to reside in the internal RAM of the transputer for maximum data throughput by using the statement

PLACE x IN WORKSPACE:

Since there are only 4 Kbytes of on-chip RAM on each T800, a compromised and optimum usage of this internal RAM between program code and data has to be found (by trial and error when Occam is used) for the maximum overall throughput.

6 Conclusion

A new parameter representation scheme has been proposed for fast and efficient Hough transform of circles. This approach replaced the conventional three-dimensional accumulator array with a pair of two-dimensional planes without sacrificing the power of discrimination of concentric circles. An inverse mapping from the estimated circle parameter sets to a subspace of the original image was used to verify the candidate circles, giving a recognition rate of 95–100%. Parallel realisations of this new scheme on a reconfigurable MIMD transputer network, Superlink, were also presented.

7 Acknowledgment

The authors would like to acknowledge the financial support of the Research Committee of the Hong Kong Polytechnic under the research project grant 341/031.

8 References

1 HOUGH, P.V.C.: 'Method and means for recognising complex patterns', *U.S. Patent 3069654*, 1962
2 BALLARD, D.H.: 'Generalising the Hough transform to detect arbitrary shapes', *Pattern Recognit.*, 1981, **13,** (2), pp. 111–122
3 ILLINGWORTH, J., and KITTLER, J.: 'A survey of the Hough transform', *Comput. Vis. Graph. Image Process.*, 1988, **44,** pp. 87–116
4 CONKER, R.S.: 'A dual plane variation of the Hough transform for detecting non-concentric circles of different radii', *ibid.*, 1988, **43,** pp. 115–132
5 ILLINGWORTH, J., and KITTLER, J.: 'The adaptive Hough transform', *IEEE Trans. Pattern Anal. Mach. Intell.*, 1987, **PAMI-9,** (5), pp. 690–698
6 CHAN, R., and SIU, W.C.: 'A new approach for efficient Hough transform for circles', *Proc. IEEE Pacific RIM Conf. Commun. Comput. Signal Process.*, 1989, pp. 99–102, Victoria, Canada
7 SANDLER, M.B., and EGHTESADI, S.: 'Transputer based implementations of the Hough transform for computer vision', *Microprocess. Microprogr.*, 1988, (24), pp. 403–408
8 FISHER, A.L., and HIGHNAM, P.T.: 'Computing on Hough transform on a scan line array processor', *IEEE Trans. on Pattern Anal. Mach. Intell.*, 1989, **PAMI-11,** (3), pp. 262–265
9 DAVIES, E.R.: 'A modified Hough scheme for general circular location', *Pattern Recognit. Lett.*, 1988, **7,** pp. 37–43
10 DAVIES, E.R.: 'Circularity — A new principle underlying the design of accurate edge orientation operators', *Image Vis. Comput.*, 1984, **2,** (3)
11 SKLANSKY, J.: 'On the Hough technique for curve detection', *IEEE Trans. Comput.*, 1986, **C-27,** pp. 923–926
12 YE, Q.Z.: 'A preprocessing method for Hough transform to detect circles', *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 1986, pp. 651–653
13 DAVIES, E.R.: 'A high speed algorithm for circular object location', *Pattern Recognit. Lett.*, 1987, **6,** pp. 323–333
14 MILENKOVIC, V.J.: 'Multiple resolution search technique for the Hough transform in high dimensional parameter spaces', *in* ROSENFELD, A. (Ed.): 'Techniques for 3-D machine perception', 1986, pp. 231–254
15 LEAVERS, V.F., BEN-TZVI, D., and SANDLER, M.B.: 'A dynamic combinatorial Hough transform for straight lines and circles', *Proc. Alvey Vis. Club Conf.*, Reading, 1989, pp. 163–168
16 GERIG, G., and KLEIN, F.: 'Fast contour identification through efficient Hough transform and simplified interpretation strategy', *Proc. Int. Conf. Pattern Recognit.*, Paris, 1986, **1,** pp. 498–500
17 BRESENHAM, J.E.: 'Algorithm for computer control of a digital plotter', *IBM Syst. Journal.*, 1965, **4,** (1), pp. 25–30
18 KNUTH, D.E.: 'The art of computer programming, Vol. 3, Sorting and Searching' (Addison-Wesley, Reading, Mass., 1973) pp. 406–422
19 MORROW, P.J., and PERROTT, R.H.: 'The design and implementation of low-level image processing algorithms on a transputer network', *in* PAGE, I. (Ed.): 'Parallel architectures and computer vission' (Oxford Univ. Press, 1988) pp. 243–259
20 SANDLER, M.B., HAYAT, L., COSTA, L., and NAQVI, A.: 'A comparative evaluaton of DSPs, microprocessors and the transputer for image processing', *Proc. IEEE ICASSP 89*, 1989, pp. 1532–1535
21 'Transputer development system, INMOS Ltd.' (Prentice Hall, 1988) pp. 119–120