A SURVEY OF EFFICIENT HOUGH TRANSFORM METHODS

J. Illingworth and J. Kittler

Department of Electronics and Electrical Engineering
University of Surrey, Guildford, GU2 5XH. U.K.

Abstract

The Hough Transform, HT, has long been recognised as a technique of almost unique promise for shape and motion analysis in images containing noisy, missing and extraneous data. However its widespread adoption in practical systems has been slow due to its computational complexity, the need for large storage arrays and the lack of a detailed understanding of its properties. The aims of Alvey project MMI/IP078 are

- the investigation of the properties of HTs
- the study of efficient implementations of HTs
- the production of a HT hardware device for use in real time industrial inspection systems.
- the development of general high level image interpretation strategies which use information derived from the HT.

The first three items listed above are being studied at Surrey University in collaboration with Computer Recognition Systems while the final item is under investigation at Heriot Watt University. There has recently been much activity concerned with space and time efficient implementations of HT ideas and this paper presents a survey of this area. Topics covered include the development of new algorithms, the use of optical techniques and the implementation of HTs on novel hardware architectures.

1. Introduction

Line drawings and silhouettes strongly suggest that in human vision the shape of objects provides a major part of the information needed for the interpretation of 2D images as 3D world scenes. The development of methods which will enable computers to reproduce these abilities is one of the major problems in computer vision. A good shape recognition system must be able to handle complex scenes which contain several objects that may partially overlap one another. In the case of computer vision systems the recognition algorithm must also be able to cope with difficulties caused by poor characteristics of image sensors and the limited ability of current segmentation algorithms. The Hough Transform [1] is an algorithm which has the potential to address these difficult problems. It is a likelihood based parameter extraction technique in which pieces of local evidence independently vote for many possible instances of a sought after model. In shape detection the model captures the relative position and orientation of the constituent points of the shape and distinguishes between particular instances of the shape using a set of parameters. The HT identifies specific values for these parameters and thereby allows image points of the shape to be found by comparison with predictions of the model instantiated at the identified parameter values.

Image points on a parametrically defined shape or curve can be defined by an equation of the form

$$f(\hat{\mathbf{P}}, \mathbf{X}) = 0 \tag{1.1}$$

where $\hat{\mathbf{P}} = (\hat{p}_1,...,\hat{p}_n)$ are specific values of the n parameters of the shape and $\mathbf{X} = (x_1,x_2)$ are the coordinates of any image point of the shape. Equation (1.1) defines a one to many mapping from a parameter space $\mathbf{P}$ to an image space $\mathbf{X}$. One of the basic ideas behind the HT is that the form of equation (1.1) can be used to express a one to many mapping from image space to a space of parameters i.e. there is a transform

$$g(\mathbf{P}, \hat{\mathbf{X}}) = 0 \tag{1.2}$$

which maps each specific image point, $(\hat{x}_1, \hat{x}_2)$ into a set of parameter values and the form of g is the same as that of f. The mapping g generates the parameters of all possible shape instances which could include the image point $(\hat{x}_1, \hat{x}_2)$. For example in the case of a circular shape the relationship between image and parameter space is given by

$$(x_1 - p_1)^2 + (x_2 - p_2)^2 - p_3^2 = 0 \tag{1.3}$$

where p_1 and p_2 are the parameters describing the position of the circle centre and p_3 denotes the circle radius. Mapping f is obtained by taking values for $\hat{p}_1$,$\hat{p}_2$, and $\hat{p}_3$ and (1.3) then defines the set of image points which belong to a specific circle instance. Mapping g is obtained from (1.3) when specific values are defined for $(\hat{x}_1, \hat{x}_2)$ and it then yields the set of $(\hat{p}_1,\hat{p}_2,\hat{p}_3)$ values which define circles that the image point could lie on.

The parameter surface which g generates for each image point is the surface of an upturned right circular cone with its apex at parameter values $(\hat{x}_1,\hat{x}_2,0)$. The HT involves taking each image point in turn and mapping it into multidimensional parameter space $\mathbf{P}$ using g. If several points belong to the same shape then the parameter space surfaces which they generate via g will all intersect at the parameter point which characterises this shape. The HT is a method of identifying these points of intersection.

In the standard digital computer implementation of the HT method both image and parameter space are represented by arrays of numbers which are spatially indexed to appropriately small, regularly shaped regions of space. The image array is derived from real world measurements and is a binary valued array in which 1s represent the likely presence of the shape within the corresponding region of space. Non zero elements of the image array are mapped into parameter space so that each element of the parameter array accumulates a value which is equal to the number of parameter hypersurfaces that pass through it. This discrete parameter space is usually called the accumulator array and cells where many hypersurfaces intersect have a large count and form a peak structure. In using the HT the difficult problem of identifying sets of spatially extended points is turned into the much easier task of finding compact local peaks in the accumulator array.

The above discussion applies to simple parametric shapes but the HT method can be extended to detect general shapes [2,3]. In these cases a shape is described by a list of template vectors which record the distance and orientation of each point from an arbitrary localisation point. A suitable localisation point is the centroid of the template shape. Translated instances of the shape may lie anywhere in the image plane. Each can be characterised by the position of its centroid i.e. the coordinates of the centroid are two parameters which can be used to identify translated instances of a template shape. In the generalised HT, GHT, each image point is compared to every entry of the template list and from this a candidate centroid is calculated and the corresponding cell in an accumulator array is incremented. As in other HTs, shape parameters are identified by locating peaks in an accumulator array. The method can be used to find scaled and rotated versions of a shape using two further parameters to describe these degrees of freedom.

A good way to view the HT is as an evidence gathering or voting procedure. Each image point generates or votes for all parameter instances that could have produced it. The votes are counted in a multidimensional accumulator array and the final totals indicate the relative likelihood of shapes described by parameters which are within the accumulator cell. Only sets of image points which belong to a shape will vote coherently and produce peaks in the accumulator array. Using this viewpoint it is easy to see that the HT can cope with many of the difficulties present in real world imagery. The method is robust in the presence of random extraneous data as this maps into a distributed background of votes which is unlikely to hinder detection of large compact peaks. The HT is also tolerant of missing data produced by poor segmentation or occlusion. These effects lead to a decrease in the height of parameter peaks but do not necessarily affect their detectability catastrophically. Finally the HT is inherently parallel as it combines information from each image point independently. This means that it can simultaneously accumulate evidence for several examples of a shape. Each instance of the shape produces a distinct peak in the accumulator array.

One of the principal disadvantages of the HT in its standard implementation is the large storage and computational requirements which arise from the use of a large accumulator array. For the determination of q parameters we need to represent a q dimensional space. If each parameter is to be resolved into α intervals then the accumulator array will have α^q elements. If either q or α is large then this can be a prohibitively large amount of storage . The size of the accumulator also has a direct bearing on the computational cost of the HT as most of the algorithm involves determining, for each image point, which cells of the accumulator array have to be incremented. A large accumulator means that a large number of parameter hypersurface-accumulator cell intersection tests have to be evaluated. As these considerations are of great importance to the practical adoption of the HT method we have begun to study methods which can utilise the HT ideas but overcome the inefficiencies of the most naive standard implementation. In this survey paper we report published work which relates to ideas for the efficient implementation of the HT. Section 2 discusses how efficiency problems can be tackled by consideration of task specific ideas. A very useful and widely applied idea is that instead of searching for many parameters using a single stage HT it is better to decompose large dimensional problems into multistage algorithms involving a sequence of low dimensional HTs. Section 3 considers algorithms which use novel data structures or focusing approaches to the HT. It includes our own work on a method which we have called the Adaptive Hough Transform, AHT. Section 4 describes the relationship between the HT and other transforms with a view to use of these transforms to compute the HT more easily. It can be shown that the HT is equivalent to calculating projections and these results lead to both optical and digital implementations of the HT. In Section 5 we consider how the HT has been mapped onto existing parallel architectures. Section 6 is a brief summary of the paper.

2. Task specific principles.

In the standard implementation of the HT storage increases exponentially as the number of parameters, q, increases or the parameter resolution, ($\propto \alpha$) increases. This curse of dimensionality also applies to the amount of computation required, as the parameter hypersurface which each image point generates is usually a (q-1) dimensional hypersurface. The most obvious way to improve efficiency is therefore to ensure that both q and α remain as small as practicable. An obvious strategy is to restrict the range of parameters which are to be found by using apriori knowledge of the specific problem to be solved. A smaller parameter range can be covered to a specified high resolution using fewer accumulator cells i.e. a smaller value of α. For example in many problems which involve circle finding there may be limits on the likely radius of circles or the centre of the circle may be constrained to lie within the interior of the image. In a limiting case it may be known that circles are of a single known radius and in that case the 3 parameter circle finding problem collapses to a 2 parameter problem of finding circle centres. Alternatively the shape may be known to have certain rotational symmetries and these can be exploited to reduce the number of intervals needed to determine orientation. Davies [3] has considered regular polygonal shapes and concludes that the orientation axis need only be divided into a number of intervals which is equal to the number of polygon sides.

A less obvious way of improving the computational efficiency of the HT is to use extra information to restrict the mapping of image to parameter values from a "one to many" mapping to a "one to few" mapping i.e. use extra data to constrain the dimensionality or range of the hypersurface that an image point generates. Edge direction information [4] is the most commonly used piece of extra information although curvature and gray level constraints have also been used in this way [5,6]. For example in the case of straight line detection the edge direction is directly related to the line gradient and therefore line finding can be reduced to a 1 parameter histogramming task to find the best intercept value. In the case of circle finding edge angle information restricts each image point mapping to a single line of parameter values on the surface of a cone. The idea of using edge information has been used most notably in Ballards [7] efficient HT for general shapes. This relies for its efficiency on the indexing of possible localisation point vectors by values of edge direction. Candidate shape points are then only matched to template points which possess the same value of edge direction.

A sophisticated use of prior information involves decomposition of high dimensional problems into a series of lower dimensional problems. One of the simplest examples of this type is the use of edge direction information, $\hat{\theta}$, as a constraint in circle finding. All the edge vectors on a circle must project to a common intersection at the centre of the circle. In [8] it is shown how this can be used to define a 2 stage HT. Edge vectors can be estimated using standard operators such as the Sobel operator. The first stage of the modified HT involves mapping image features described by $(\hat{x}_1,\hat{x}_2,\hat{\theta})$ into a 2 parameter space (p_1,p_2) of circle centres. Peaks identified in the 2 parameter space yield candidate $(\hat{p}_1,\hat{p}_2)$ values which are then used with image point coordinates in a second stage to calculate a histogram of possible p_3 or radius values. This radius histogram can be viewed as a trivial 1 parameter HT which maps 4 tuples of $(\hat{x}_1,\hat{x}_2,\hat{p}_1,\hat{p}_2)$ into p_3 values. The maximum dimensionality of any of the substages is only 2 and the overall efficiency of the modified HT is dominated by the demands of this stage. The penalty paid for this increased efficiency is the cost of calculating, and the reliance on, good edge information. Edge detection is a computationally intensive task but in most cases this will not outweigh the benefit gained from its use in the two stage HT. Indeed there are many situations where edge information may have to be computed for purposes other than the HT and in these cases no extra computation is incurred. However decomposing the HT into a multistage process may introduce systematic errors as inaccuracies in parameters determined at an early stage propagate into later stages. Tsukune and Goto[9] have shown that similar edge based constraints can be derived and used to decompose the 5 parameter ellipse finding problem into a 3 stage problem involving determination of at most two parameters at any single step.

Ballard and Sabbath [10] have shown that parameter decomposition can be a powerful technique in the matching of 2D views to 3D models. Descriptions of 3D objects are usually stored as object centred representations and recognition involves determining the parameters of a transformation which maps a 2D view into these 3D descriptions. The viewing transformation generally involves 7 parameters which cater for changes in scale (1 parameter), orientation (3 parameters) and translation (3 parameters). However for orthographic projection there is a natural ordering or dominance of parameters such that they can be sequentially determined in subgroups i.e. scale can be determined without knowledge of orientation or translation and orienation can be found without knowing the value of translation parameters. Ballard and Sabbah give specific algorithms for the case where objects are described by oriented edges and planar surface patches. The maximum dimensionality of any subproblem is only 3.

A further possibility for parameter reduction is to consider how any shape maps into a low dimensional space such as that used in line finding. Several authors [11,12,13,14,15] have shown that the distribution of votes in this parameter space has a characteristic shape. Finite length lines produce a butterfly shaped peak of counts rather than a symmetric peak. Leavers and Boyce [14] have designed filters to enhance this feature in order to increase the reliablilty of line detection. They have also considered the mapping of circles [15] into this parameter space. Circles produce a band of votes with a characteristic falloff at the band edges. The edges of the band can be enhanced by a filter and parameters of the circle can be estimated from their positions. Casasent and Krishnapuram [16] use the same idea of mapping any shape into a two parameter space but they suggest manipulating the resulting HT parameter space so that upon applying the inverse HT the votes from a shape map to a compact peak. They indicate a general technique for deriving the manipulations which must be performed in parameter space. The position of the peak in the inverse transform space indicates the translation parameters associated with the shape.

3. Efficient accumulation methods

In this section we consider several methods which utilise novel data structures or use techniques that employ nonuniform or multiple resolutions in order to achieve storage and computation savings. Many schemes are based on the observation that it is necessary to have high accumulator resolution only in places where a high density of votes accumulate. O'Rourke and Sloan [17,18,19] were among the first to use this principle. They developed two data structures. The first structure, Dynamically Quantised Spaces (DQS) [17,19], consists of a binary tree in which each node encloses a rectangular region of space. As data is accumulated rules are used to split and merge cells so that each cell contains, irrespective of its size, approximately the same number of counts evenly distributed throughout its volume. The total number of cells in the tree is an input parameter of the method but in the final tree there will be a concentration of small cells around peaks and this allows peaks to be located more accurately than using the same number of cells in a standard uniform quantisation schemes. The second data structure, Dynamically Quantised Pyramids (DQP) [18,19], is based on a multiresolution multidimensional quadtree in which the number and connectivity of cells is small and fixed. A disadvantage of the quadtree structure is that its boundaries, and hence its spatial resolutions, are fixed. This limitation is overcome in the DQP data structure using a dynamic technique, called hierarchical warping, to modify the boundaries of cells by tracking the mean position of data points in their regions of space. The net effect of the boundary adjustment is to produce cells containing approximately equal numbers of votes. However the warping process introduces some errors in the final accumulator totals and means that the final division of space depends on the order in which the data is accumulated i.e. recently accumulated points are most influential on the final result. O'Rourke and Sloan use both data structures in a series of experiments where a 3 parameter space is divided into about 64 cells. They demonstrate the relative abilities of each algorithm to focus on peaks, to dynamically change attention and to detect multiple peaks. Their general conclusion is that DQSs are difficult to implement but perform somewhat better than DQPs.

A useful idea suggested by several authors is the iterative focusing of the HT to identify peaks with high counts. Initially the HT can be accumulated in a coarse but uniform resolution parameter space. Regions or cells with high counts at this resolution can then be investigated at higher resolution. Only a few iterations of this procedure can lead to very high parameter resolution. Consider searching for a peak in a q dimensional space resolved into α accumulator bins. The iterative use of a much smaller accumulator of size β requires $O(\frac{log(\alpha)}{log(\beta)})$ iterations to focus down to the same resolution. The computational saving of the method is proportional to the ratio of the number of cells in the two accumulators i.e. $\left(\frac{\alpha}{\beta}\right)^q$ times the ratio of iterations i.e. $O\left(\left(\frac{\alpha}{\beta}\right)^q * \frac{log(\beta)}{log(\alpha)}\right)$. This can be very large. This strategy provides enormous efficiency benefits in terms of both amount of computation and amount of storage space required. It has been used by Adiv [20] as an efficient way to search for motion parameters in high dimensional spaces and by Silberberg [21] to identify the viewing parameters relating stored models to observed shapes.

Recently Li et al [22,23] have detailed a focusing algorithm, the Fast Hough Transform, FHT, which uses a multidimensional quadtree in conjunction with HTs which map image points into hyperplanes. This use of hyperplanes is advantageous as the approximate intersection between planes and parameter cells can be efficiently computed using incremental tests which depend on the known spatial relationship between a quadrant and its four sons. Only additions and shifts are required to implement the test and the computational cost scales only linearly with the dimensionality of the parameter space. Computational gains of $O(10^3)$ for 2D lines and $O(10^6)$ for 3D plane detection can be achieved. The algorithm is regular in structure and is therefore easily extended to higher dimensional spaces and easily implemented in VLSI hardware. However the FHT currently identifies areas for focusing by requiring that quadrants have a vote count which exceeds a fixed threshold. In complex images this threshold may be difficult to determine and if it is set too low the efficiency of the method will deteriorate as the algorithm will explore too many quadrants. Another problem is related to the incremental intersection testing method. This requires that each quadrant must store the distance of every image feature from its centre. If there are many image features this can represent a large overhead. It is also not clear that all shapes can be naturally mapped into hyperplanes. Finally the quadtree data structure is a static structure and cannot optimally adapt to situations which may require different parameter resolutions along different parameter axes. Li et al [24] have recognised the last problem and have suggested using a data structure, called the bintree, in which space is partitioned into rectangular regions.

Illingworth and Kittler [8,25] have developed an iterative coarse to fine search strategy for detecting lines and circles in 2D and 3D parameter spaces. Their implementation is called the Adaptive Hough Transform, AHT. It uses a small accumulator array (typical $\beta = 9$) which is thresholded and then analysed by a connected components algorithm. The shape and extent of connected components determine the new parameter limits which are used in the next iteration. Limits can be decreased, expanded, rotated or merely translated depending on the distribution of counts in the accumulator. Parameter limits are defined independently for each parameter dimension and therefore the method selects appropriate precision for each parameter. The method works well for single object recognition and we are currently investigating strategies for using it to detect multiple objects. The storage gain using the AHT is of the order expected for focusing methods, i.e. $\left(\frac{\alpha}{\beta}\right)^q$, while experiments with searching for circles in a 3 parameter space demonstrated that it was several hundred times times faster than a standard method.

Brown [26,27] attempted to overcome the storage problems associated with the standard HT by replacing the accumulator array by a small fixed size content addressable store i.e. a hash table in software or a cache if implemented in hardware. Counts are accumulated in the store until it is full and then a flushing or garbage collection method is invoked to release some storage for further use. The simplest flushing strategy is to remove entries with fewest votes. However it is possible to devise schemes which will favour the keeping of recent information or which will incorporate geometric locality information. Brown studied caches of size 32, 64 and 128 and compared their performance againt standard accumulators as he varied flushing strategy and the order of accumulation of the data. He concluded that finite length caches were as practical as accumulators and he was able to predict qualitatively the degradation of their performance as noise increased and cache length decreased.

4. Optical and projection based implementations.

In this section we consider the relation of the HT to other transforms. It was pointed out by Deans [28] that the HT for linear features was a special case of a well known transform much studied in the mathematical literature, the Radon Transform. The Radon transform of a function $f(x,y)$ on a two dimensional Euclidean plane is defined as

$$R(\rho,\theta) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\delta(\rho - xcos(\theta) - ysin(\theta) \quad)dxdy \tag{4.1}$$

where δ is the Dirac delta function. The delta function term forces integration of $f(x,y)$ along the line

$$\rho = xcos(\theta) + ysin(\theta) \tag{4.2}$$

The Radon Transform yields the projections of the function $f(x,y)$ across the image for various values of line parameters ρ and θ. It is a transform which is much used in computer tomography and is equivalent to the HT when considering the transform for binary images. The Radon and Hough transforms for shapes other than lines are calculated by integrating or projecting image data along paths which are identical to the curve to be detected.

An important property of the Radon transform is that it can be computed via the Fourier transform. The Fourier Slice theorem states that if $F(u,v)$ is the Fourier transform of a function $f(x,y)$, $\mathrm{R}_\theta(\rho)$ is a projection at an angle θ across $f(x,y)$ and $\mathrm{S}_\theta(\omega)$ is the Fourier transform of $\mathrm{R}_\theta(\rho)$, then

$$S_\theta(\omega) = F(\omega cos\theta, \omega sin\theta) \tag{4.3}$$

This allows efficient calculation of the Radon transform using either optical Fourier techniques [29,30,32] or the Fast Fourier Transform [31].

Several authors have used optical methods to compute the HT. Eichmann and Dong [29] suggested a setup which used coherent optics. Later Gindi and Gmitro [30] discussed a hybrid optical-digital hardware system for the calculation of the Radon transform at video rates on a 512×512 image. In their system a single projection is acquired by imaging a scene onto a special 1D sensor array whose elements span the vertical length of the image. This is achieved using an anamorphic optical system, one with differing magnification in two orthogonal directions. Data for different angles of projection is collected by rotating the image using a prism which spins at 450rpm. 512 angular slices each consisting of 512 samples could be collected at standard video rates. This projection data can be processed to yield the HT for lines. Steier and Shori [32] have built a system based on similar ideas and show results relating to the location of bridges and roadways in outdoor scenes and the detection of tool parts in an industrial inspection task.

The idea of the HT as a projection has been used by Mostafavi and Shah [33] in conjunction with hardware features of commercially available image processing systems to produce a rapid algorithm for line detection. The same idea has been used and extended in the recent work of Sanz and Dinstein [34,35]. Lines at a single orientation, θ, can be detected by convolving the binary image of the line with a template image. In the template image each pixel lying on a line a normal distance ρ from the origin has an intensity of ρ. The resultant histogram of gray levels should have large peaks only at values of intensity equal to the ρ value of the line. By generating and evaluating a series of histograms for each value of θ the full HT can be built up. Features of specialised image processing and general pipelined architectures can be used in the template image generation, the image convolution and the gray level histogram generation steps. The method can be used for shapes other than lines by using different template images.

5. Parallel architectures.

One of the main characteristics of the HT is that it consists of a series of fairly simple calculations carried out independently on every feature in an image. In this section we will review ideas and attempts to capture this inherent paralellism on specialised parallel architectures.

Several authors have investigated the implementation of the HT on currently available SIMD architectures. These usually consist of square arrays of simple processing elements, P.E.s, connected so that each can communicate with its 4 or 8 neighbours. All processors concurrently execute the same instructions on different items of data. Most often each P.E. has only a single bit arithmetic logic unit, several registers and a few kilobits of on chip RAM. The mapping of the HT onto these architectures has been considered by both Silberberg [36] and Li [37]. Silberberg considered distributing both the image and the accumulator arrays among all P.E.s i.e. each P.E. is assigned both an image feature and a cell of parameter space. However this necessitated a great deal of data transfers between P.E.s in order to pass votes to the correct P.E. for accumulation. Over 98% of the time of the algorithm was taken in shifting data around the processing array and the end result was that the use of 8000 simple P.E.s led to a speedup factor of only 7 relative to a standard implementaion on a VAX 11/785. This poor result illustrates the care required in mapping algorithms onto parallel machines.

Li [37] considered two schemes for running his Fast Hough Transform, FHT, on a SIMD architecture. In the first scheme each P.E. is assigned an image feature and the coordinates of a parameter cell are broadcast simultaneouly to every P.E. by a central controller. Each P.E. decides whether the hypersurface generated by its image feature intersects the cell and if it does it sends a vote back to the controller. The votes from each P.E. can be summed by the central controller and stored for later analyis. A second scheme is to assign each P.E. a volume of parameter space and broadcast the image features. The choice of method depends on the number of available P.E.'s, the number of image features and the number of parameter cells. For the standard HT the number of parameter cells increases exponentially with dimensionality of the problem and therefore the first alternative is likely to be the most feasible.

Simplistically one might argue that the use of n P.E.s, one for each of n image features, should speed up the HT by a factor of O(n). However votes have to be routed to their correct places in the accumulator and on a simple square 2D mesh with only local shifting operations this requires $O(\sqrt{n})$ steps. As Silberberg found out this can be a significant overhead and even the dominant time factor. However if the array is augmented by a tree based voting network this can be reduced to O(log(n)) and the expected gains in efficiency should again approach O(n).

Little et al [38] describe a possible implementation of the HT on an architecture called the Connection Machine. This is similar to the previously mentioned SIMD architectures but as well as P.E.s communicating with near neighbours there is a hardware router which implements rapid communication between any pair of processors. The architecture is based on a 12 dimensional hypercube i.e. every processor can be reached from any other by traversing at most 12 edges of the cube. Unfortunately this paper concentrates on aspects of programming and addressing and does not give any figures on the efficiency gains in using this parallel implementation.

Olsen et al [39] discuss the operation of the HT on a MIMD computer architecture. They use a commercially available system called the BBN Butterfly Parallel Processor^{TM}. This consists of 256 processing nodes connected by a switching network. Each processor is a microprocessor from the Motorola 68000 series and each node has a substantial amount of local memory i.e. 1 megabyte. Memory is shared in the sense that a switching network allows any processor to address the local memory of any other processor. Olsen et al implemented two different line finding HTs using different parameterisations for lines. The versions differed in whether the image was partitioned among the processors with the parameter cells accessed from common memory or vice versa. They consider whether the algorithm is computation or communication resource bound and their results indicate that it is possible to achieve processor utilisation factors in the region of 80% i.e. if 100 processors are used they will give a speedup of 80 times relative to using a single processor.

The HT has attracted attention from researchers interested in human vision as it is a prime example of the ideas of the connectionist school of artificial intelligence [40,41]. The unifying principle of this approach to intelligence is that low and medium level vision tasks are done by massively parallel, cooperative computations on large networks of simple neuron like units. Low level pixel based properties, like edge or gray level estimates, can be represented by nodes in a feature network which is spatially indexed to the image while parameter values associated with higher level image entities, such as straight lines, can be represented by nodes in a seperate parameter network. Each node records a measure of confidence of the occurrence of its feature or parameter value and direct connections between the two networks define possible ways in which nodes can influence these confidence values. Different connection patterns can be used to impose different image space to parameter space mappings i.e. connections can be established so that when many low level units which lie on a straight line have high confidence then the higher level unit describing the parameters of this line will acquire a large confidence value. The major characteristic of this type of implementation is the huge number of feature and parameter units needed and the very large number of connections which are required between them.

6. Summary

We have discussed some of the most important ideas which have been proposed for increasing the efficiency of the HT. These include principles which can be exploited to decrease the storage and computational demands of specific tasks as well as new, general ways of efficiently representing and accumulating the parameter space. The consideration of optical techniques has led to the development of realtime implementations of the HT while experience with programming the HT on general purpose parallel architectures has illustrated that certain global communication facilities are needed if this is to be sucessful. In future work we will investigate and develop many of these ideas as well as considering other aspects of the shape representation and recognition problem.

References

[1] Hough P.V.C., "A method and means for recognising complex patterns.", US Patent 3,069,654 (1962)

[2] Merlin P.M. and Farber D.J., "A parallel mechanism for detecting curves in pictures." IEEE Trans. on Computers. 24 (1975) 96-98.

[3] Davies E.R., "Reduced parameter spaces for polygon detection using the generalised Hough Transform." In Proc. 8th IJCPR. Paris. (1986) 495-497.

[4] Kimme C., Ballard D.H. and Sklansky J., "Finding circles by an array of accumulators." Communications of the ACM. 18 (1975) 120-122.

[5] Hakalahti H., Harwood D. and Davis L.S., "Two dimensional object recognition by matching local properties of contour points." Pattern Recognition Letters. 2 (1984) 227-234.

[6] Moring I. and Hakalahti H., "Scale independent method for object recognition." In Proc. Scandinavian Conf. on Image Processing. Trondheim. (1985) 881-889.

[7] Ballard D.H., "Generalising the Hough Transform to detect arbitrary shapes." Pattern Recognition. 13 (1981) 111-122.

[8] Illingworth J. and Kittler J., "The Adaptive Hough Transform." to appear IEEE Trans. on Pattern Analysis and Machine Intelligence. September (1987).

[9] Tsukune H. and Goto K., "Extracting elliptical figures from an edge vector field." In Proc. IEEE Conf. Computer Vision and Pattern Recognition. Washington. (1983) 138-141.

[10] Ballard D.H. and Sabbah D., "Viewer independent shape recognition." IEEE Trans. on Pattern Analysis and Machine Intelligence. 5 (1983) 653-660.

[11] Van Veen T.M. and Groen F.C.A., "Discretisation errors in the Hough transform." Pattern Recognition. 14 (1981) 137-145.

[12] Brown C.M., "Inherent bias and noise in the Hough Transform." IEEE Trans. on Pattern Analysis and Machine Intelligence. 5 (1983) 493-505.

[13] Maitre H., "Contribution to the prediction of performances of the Hough Transform." IEEE Trans. on Pattern Analysis and Machine Intelligence. 8 (1986) 669-674.

[14] Leavers V.F. and Boyce J.F., "The Radon Transform and its application to shape parameterisation in machine vision" Image and Vision Computing. 5 (1987) 161-166.

[15] Boyce J.F., Jones G.A. and Leavers V.F., "An implementation of the Hough Transform for line and circle detection." In Proc. SPIE Conf. The Hague. (1987).

[16] Casasent D. and Krishnapuram R., "Curved object location by Hough transformations and inversions." Pattern Recognition. 20 (1987) 181-188.

[17] O'Rourke J., "Dynamically quantised spaces for focusing the Hough transform." In Proc. 7th IJCAI. Vancouver. (1981) 737-739.

[18] Sloan K.R., "Dynamically quantised pyramids." In Proc. 7th IJCAI. Vancouver. (1981) 734-736.

[19] O'Rourke J. and Sloan K.R., "Dynamic quantisation: two adaptive data structures for multidimensional spaces." IEEE Trans. on Pattern Analysis and Machine Intelligence. 6 (1984) 266-279.

[20] Adiv G., "Recovering motion parameters in scenes containing multiple moving objects." In Proc. IEEE Conf. on Computer Vision and Pattern Recognition. Washington. (1983) 399-400.

[21] Silberberg T.M., Davis L. and Harwood D., "An iterative Hough procedure for 3D object recognition." Pattern Recognition. 17 (1984) 621-629.

[22] Li H., Lavin M.A. and LeMaster R.J., "Fast Hough Transform." In Proc. of 3^{rd} Workshop on Computer Vision: Representation and Control. Bellair. Michigan. (1985) 75-83.

[23] Li H., Lavin M.A. and LeMaster R.J., "Fast Hough Transform: a hierarchical approach." Computer Vision Graphics and Image Processing. 36 (1986) 139-161.

[24] Li H. and Lavin M.A., "Fast Hough Transform based on the bintree data structure." In Proc. IEEE Conf. Computer Vision and Pattern Recognition. Miami Beach. (1986) 640-642.

[25] Illingworth J. and Kittler J., "Shape detection using the Adaptive Hough Transform", In Proc. of Nato Advanced Research Workshop. Italy. (1987) to be published by Springer-Verlag.

[26] Brown C.M. and Sher D.B., "Modelling the sequential behaviour of Hough Transform schemes." In Proc. Image Understanding Workshop. ed by L.S. Baumann. Palo Alto. (1982) 115-123.

[27] Brown C.M., Curtiss M.B. and Sher D.B., "Advanced Hough Transform implementations." In Proc. 8^{th} IJCAI. Karlsruhe. (1983) 1081-1085. or Internal report TR113. Computer Science Department. University of Rochester.

[28] Deans S.R., "Hough transform from the Radon Transform." IEEE Trans. on Pattern Analysis and Machine Intelligence. 3 (1981) 185-188.

[29] Eichmann G. and Dong B.Z., "Coherent optical production of the Hough Transform." Applied Optics. 22:6 (1983) 830-834.

[30] Gindi G.R. and Gmitro A.F., "Optical feature extraction via the Radon Transform." Optical Engineering. 23:5 (1984) 499-506.

[31] Murphy L.M., "Linear feature detection and enhancement in noisy images via Radon Transform." Pattern Recognition Letters. 4 (1986) 279-284.

[32] Steier W.H. and Shori R.K., "Optical Hough Transform." Applied Optics. 25:16 (1986) 2734-2738.

[33] Mostafavi H. and Shah M., "High speed digital implementation of linear feature extraction algorithms." In Proc. SPIE Conf. 432 (1983) 182-188.

[34] Sanz J.L.C., Hinkle E.B. and Dinstein I., "Computing geometric features of digital objects in general purpose image processing pipeline architectures." In Proc. IEEE Conf. Computer Vision and Pattern Recognition. San Francisco. (1985) 265-270.

[35] Sanz J.L.C. and Dinstein I., "Projection based geometrical feature extraction for computer vision : Algorithms in pipeline architectures." IEEE Trans. on Pattern Analysis and Machine Intelligence. 9 (1987) 160-168.

[36] Silberberg T.M., "The Hough transform on the Geometric Arithmetic Parallel Processor." In Proc. IEEE Conf. on Computer Architecture for Pattern Analysis and Image Database Management. (1985) 387-393.

[37] Li H., "Fast Hough Transform for multidimensional signal processing." IBM Research report. RC 11562. Yorktown Heights.

[38] Little J.J., Blelloch G. and Cass T., "Parallel algorithms for computer vision on the connection machine." In Proc. 1^{st} IEEE International Conf. on Computer Vision. London (1987) 587-591.

[39] Olson T.J., Bukys L. and Brown C.M., "Low level image analysis on an MIMD architecture." In Proc. 1^{st} IEEE International Conf. on Computer Vision. London. (1987) 468-475.

[40] Ballard D.H., "Parameter Nets." Artificial Intelligence. 22 (1984) 235-267.

[41] Feldman J.A. and Ballard D.H., "Connectionist models and their properties", Cognitive Science. 6 (1982) 205-254.