Machine Vision and Applications (2013) 24:683–694
DOI 10.1007/s00138-012-0420-x

ORIGINAL PAPER

A convolution approach to the circle Hough transform for arbitrary radius

Christopher Hollitt

Received: 8 July 2011 / Revised: 14 January 2012 / Accepted: 21 February 2012 / Published online: 5 April 2012

Abstract The Hough transform is a well-established family of algorithms for locating and describing geometric figures in an image. However, the computational complexity of the algorithm used to calculate the transform is high when used to target complex objects. As a result, the use of the Hough transform to find objects more complex than lines is uncommon in real-time applications. We describe a convolution method for calculating the Hough transform for finding circles of arbitrary radius. The algorithm operates by performing a three-dimensional convolution of the input image with an appropriate Hough kernel. The use of the fast Fourier transform to calculate the convolution results in a Hough transform algorithm with reduced computational complexity and thus increased speed. Edge detection and other convolution-based image processing operations can be incorporated as part of the transform, which removes the need to perform them with a separate pre-processing or post-processing step. As the Discrete Fourier Transform implements circular convolution rather than linear convolution, consideration must be given to padding the input image before forming the Hough transform.

Keywords Hough transform · Circle Hough transform · Convolution · Edge detection

1 Introduction

Locating geometric figures in image data is a common requirement in machine vision and image analysis. The family of algorithms collectively known as the Hough transform has long been a standard method for solving this problem due to their ability to cope with partially occluded targets and their robustness to the presence of noise [1].

The classical Hough transform maps an image into an abstract parameter space via a voting process. Each significant pixel in the input image is examined to find the set of target objects of which it could form part. Each pixel can therefore be considered to generate a "vote" for each of the plausible target objects. Target objects that are actually present in the image will be consistent with a relatively large number of image pixels, so will accumulate more votes.

In practice, votes are accumulated in an abstract parameter space known as the Hough space, which has dimension equal to the number of parameters required to uniquely describe the figure in question. For example, a Hough transform intended to find arbitrary circles will have a three-dimensional Hough space, with two parameters corresponding to the coordinates of the centre of the circle and a third to describe the circle's radius.

The Hough space is discretized to cover the expected range of target object geometries and has resolution chosen to be appropriate for the application. The Hough space is thus broken into a large number of cells, each of which acts as an accumulator of votes. When the transform is complete this Hough space can be searched to find peaks. The location of each peak will correspond to a geometric object that is present in the image.

Some care must be taken with the resolution of the Hough space quantisation, as an excessively fine grid can result in votes being spread over adjacent cells in Hough space. Conversely, using too coarse a grid reduces the discriminatory power of the transform.

Unfortunately, the Hough transform is both slow and memory intensive. Each image pixel generates a large number of votes that must be accumulated in Hough space.

C. Hollitt (✉)
School of Engineering and Computer Science,
Victoria University of Wellington, Wellington, New Zealand
e-mail: chollitt@ieee.org

This problem is exacerbated as the complexity of the target objects increases because the dimension of the Hough space increases commensurately. Real time application of the classical Hough transform algorithm therefore tends to be used for detection of straight lines [2–5], while the detection of arbitrary circles or more complex shapes is most often seen in offline applications [6,7].

There are several modifications that can be made to the simple algorithm to reduce the time and/or storage demands of the Hough transform. Three main classes of improved algorithm have received most attention in the literature;

1. Statistical algorithms that do not completely sample the image [8–10],
2. Algorithms that are more frugal with their distribution of votes [11–13], and
3. Algorithms that reduce the dimension of the Hough space [14].

The first approach allows significant peaks in Hough space to be generated without accumulating votes from every image pixel. In particular, when the target objects are large, a full Hough transform results in many more votes than are required to generate a statistically significant peak in Hough space. Considerable speed increases may thus be realised by sparsely sampling the image. The speed increase is roughly proportional to the sparseness of the sampling. However, excessive reduction in the sampling density can render peak location more difficult. Development of an optimal sampling strategy is thus non-trivial as the optimal sampling strategy depends on the properties of the input image and of the target objects.

The second approach is to generate fewer spurious votes in Hough space by extracting more information from the input image than is available in a single pixel. Examples of this approach include using the image gradient to constrain vote location. One disadvantage of this method is that it is critically dependent on the accuracy of the gradient estimate to constrain the appropriate vote distribution. It thus fares more poorly than the standard Hough transform for noisy images [15,16].

Reduction of the dimension of the Hough space is a very powerful technique that can produce significant reductions in computational complexity of the algorithm. Perhaps the best known example of this approach is the use of complex numbers to encode the radius of circles in the circle Hough transform. In this case the dimension of Hough space is reduced from three to two, resulting in a very significant increase in transform speed. Other approaches attempt to decompose the full circle Hough transform into multiple parts, each of which is more tractable than the full Hough transform [13,17].

In this paper we extend an alternative computational approach that we have proposed earlier [18]. We do not employ any of the techniques described above, but instead seek to calculate the classical Hough transform more efficiently than previously. This technique accumulates all Hough space votes in one pass by convolving the input image with an appropriate kernel. This method has the advantage of calculating the Hough transform much faster than the conventional method in many cases.

We review the basic convolution technique and then discuss a number of additions intended to make the process more robust or efficient. In particular we discuss how additional convolution-based filters can be incorporated into the calculation of the Hough transform with little additional computational cost. These filters can be used to spread votes across adjacent Hough cells or to eliminate the need for a preliminary edge-detection step before calculating the Hough transform. Using an FFT-based circular convolution technique for can produce wrap-around artifacts in the resulting Hough transform. We discuss the use of edge-padding techniques to counter those effects for applications where that is necessary.

In the remainder of the paper we will concentrate on the circle Hough transform, though the method described herein is applicable to other shapes. This generalisation only requires the determination of the appropriate kernel for each target shape using a procedure such as taking the autoconvolution of the target shape [19]. We will address the problem of finding circles of unknown radius, though the specialization to finding circles of known size is addressed in Sect. 2.2.

2 Method

A circle in $\mathbb{R}^2$ can be described by the equation

$$(x - x_0)^2 + (y - y_0)^2 - r^2 = 0, \tag{1}$$

where x_0 and y_0 are the x and y coordinates of the centre of the circle and r is the circle's radius. The three-tuple (x_0, y_0, r) uniquely parametrises each possible circle in an image and can be therefore used to specify the circles that are present.

Each point $f(x_i, y_i)$ within an image $f(x, y)$ could potentially form part of one or more circles. As any such circle must pass through (x_i, y_i), the equation

$$(x_i - x_0)^2 + (y_i - y_0)^2 - r^2 = 0 \tag{2}$$

must be satisfied for some appropriate values of x_0, y_0 and r. During calculation of the Hough transform, each image location $f(x_i, y_i)$ generates a vote for each possible tuples (x_0, y_0, r) that is consistent with itself. That is, each pixel generates a vote for all possible tuples that satisfy (2). The set of possible values for (x_0, y_0, r) describe a cone [15,20]. Thus, each image position generates a conical kernel of votes in Hough space. We will hereafter refer to this patten as the Hough cone.

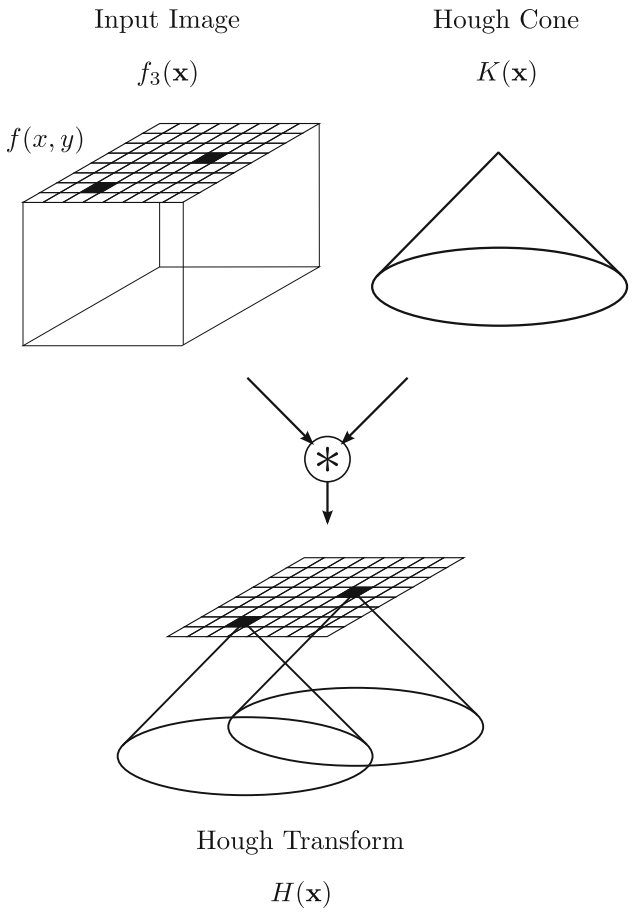

Fig. 1 Formation of the Hough transform via convolution. The input image $f(x, y)$ is extended to three dimensions and then convolved with the vote distribution $K(\mathbf{x})$ to generate the Hough transform $H(\mathbf{x})$. For clarity we have represented the Hough cone as smooth, though it is discretized on the same scale as the input image

The fact that the circle Hough transform generates a conical distribution of votes has been recognised since the earliest description of the circle transform [12,20] and the fact that the Hough transform can be described as a convolution operation has previously been discussed [12,21]. However, the observation has not previously been exploited in the manner described here. As each image pixel generates a known conical pattern we can generate the entire Hough transform in a single step by convolving the image with the conical vote distribution [18]. Figure 1 illustrates the proposed process.

We wish to formulate the Hough transform as a function that takes an input image $f(x, y) \in \mathbb{R}^{n\times n}$, and produces the three-dimensional Hough transform $H(\mathbf{x})$ with $\mathbf{x} = (x, y, r)$. That is, a peak in $H(\mathbf{x})$ at $\mathbf{x} = (x_i, y_i, r_i)$ indicates that $f(x, y)$ contains a circle centered at (x_i, y_i) that has a radius r_i. We consider each pixel of the input image to be real-valued, though a binary image may be considered as a special case.

As we begin with a two-dimensional image but need to perform the convolution operation in three dimensions, we must first prepare a suitable three-dimensional representation of the input. Let ι be an inclusion map that takes the $n \times n$ pixel image $f(x, y)$ and extends it into a three-dimensional space that is zero everywhere except for on the $r = 0$ plane, where it is equal to the input image.

$$\iota : \mathbb{R}^{n\times n} \hookrightarrow \mathbb{R}^{n\times n\times n}, \quad \iota(f(x, y)) = f(x, y)\delta(r) \tag{3}$$

We use this inclusion map to form the three-dimensional extension of the input image,

$$f_3(\mathbf{x}) = \iota(f(x, y)) \tag{4}$$
$$= f(x, y)\delta(r) \tag{5}$$

Let K be the indicator function for the set of votes generated a single image pixel. That is, K has value one for those $\mathbf{x}$ that fall on the surface of the Hough cone and zero elsewhere;

$$K(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} \in \{\mathbb{R}^2 \times \mathbb{R}_+ : x^2 + y^2 - r^2 = 0\} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

We additionally introduce a scaling factor of $\frac{1}{r}$ to the Hough cone so that different size circles produce the same peak magnitude in Hough space [22].

The Hough transform can therefore be expressed as

$$H(\mathbf{x}) = f_3(\mathbf{x}) * \frac{1}{r}K(\mathbf{x}) \tag{7}$$

where $*$ denotes convolution in three dimensions.

We can use the Fourier transform to compute the convolution efficiently. If we denote the Fourier transform of $g(\mathbf{x})$ as $(\mathcal{F}g)(\boldsymbol{\xi})$ where $\boldsymbol{\xi} = (\xi_x, \xi_y, \xi_r)$ is the vector of variables conjugate to the components of $\mathbf{x}$, then we can write the above expression as

$$(\mathcal{F}H)(\boldsymbol{\xi}) = (\mathcal{F}f_3)(\boldsymbol{\xi})(\mathcal{F}\tfrac{K}{r})(\boldsymbol{\xi}) \tag{8}$$
$$= (\mathcal{F}f)(\xi_x, \xi_y)(\mathcal{F}\delta)(\xi_r)(\mathcal{F}\tfrac{K}{r})(\boldsymbol{\xi}) \tag{9}$$
$$= (\mathcal{F}f)(\xi_x, \xi_y)(\mathcal{F}\tfrac{K}{r})(\boldsymbol{\xi}) \tag{10}$$
$$H(\mathbf{x}) = \mathcal{F}^{-1}\{(\mathcal{F}f)(\mathcal{F}\tfrac{K}{r})\} \tag{11}$$

The relationships between the various terms appearing in Eqs. 8–11 are illustrated in Fig. 2.

Notice that the three-dimensional Fourier transform of the Hough cone, $\mathcal{F}\frac{K}{r}$, can be precomputed as it is independent of the input image. Note also that $(\mathcal{F}f_3)(\boldsymbol{\xi})$ has no variation in the ξ_r direction, so only a two-dimensional Fourier transform of the image is required.

The algorithm for calculation of the Hough transform requires the following steps:

1. Precompute the three-dimensional Fourier transform of the Hough cone.

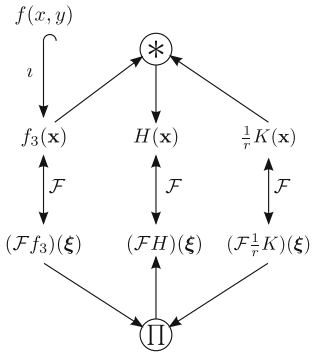

Fig. 2 Relationships between the various mathematical entities appearing in the calculation of the circle Hough transform. The $*$ indicates a convolution operation and $\prod$ represents multiplication

2. Calculate the two-dimensional Fourier transform of the image.
3. Multiply the transform of the image by the transform of the Hough cone to find the three-dimensional Fourier transform of the Hough transform.
4. Perform a three-dimensional inverse Fourier transform to recover the Hough transform.

Calculation of the last step can be expected to dominate computation time for the algorithm. Presuming that the Fourier transforms are performed with a fast Fourier transform algorithm, then the computational complexity of the Hough transform is $O(n^3 \log n)$ for a discretised Hough space with size n in each dimension. This can be compared to the conventional circle Hough transform which is $O(n^4)$, because it must accumulate votes on a two-dimensional surface for each of the order n^2 pixels in the image [12].

The time required for the computation of the Hough transform is independent of the complexity of the image. This can be contrasted with conventional Hough transform techniques, which typically accumulate votes in Hough space only for significant pixels in the input image [20]. This allows the traditional algorithm to run faster when dealing with simple images, but does lead to inconsistency in calculation times. In many cases this does not cause problems, but does make the algorithm difficult to use in real time systems, where predictability of the computation time is useful.

The above discussion has assumed that convolution can be satisfactorily calculated using the Fourier transform. However, use of the Fourier transform in this way produces the circular convolution of the input image and the Hough cone, rather than the normal convolution. As a result we can expect artifacts in the Hough transform calculated using the convolution technique. Resolution of this issue is discussed in Sect. 2.4 and demonstrated in Sect. 3.1.

2.1 Formation of the Hough cone

While the accumulation of votes in a conical distribution is suggested by the discussion above, in practice we can introduce several refinements to improve the effectiveness of the method.

It should be recognized that there is often no imperative to use the entire Hough cone, as for many applications there is no need to search for circles with the complete range of possible radii. In such cases it is sensible to truncate the cone to reduce processing time. For clarity we will use the full cone throughout this paper.

Treatment of real images always leads to some ambiguity in the position of a circle's centre. Factors such as the discrete grid of pixelated images, artifacts from edge detection, image noise and minor eccentricity in the target circles all result in votes being spread over a number of adjacent Hough space cells, rather than the single cell that an ideal model would suggest. While the amount of spreading is application dependent, it is universally undesirable as it causes a reduction in the peak magnitude of Hough space peaks and can therefore lead to difficulty in determining which of a set of adjacent cells in Hough space provides the best description of a geometric object.

This problem can be reduced by allowing each vote to count toward a number of adjacent Hough space locations. Each vote is thus spread across Hough space according to some appropriate point spread function that can be determined empirically or by a variety of theoretical considerations [23–26]. The nature of this function is again application specific, but usually takes the form of a Gaussian with a standard deviation of a few pixels. When implementing a conventional Hough transform this refinement adds to the calculation time as extra accumulation steps must be performed for each image pixel.

The desired spreading of the vote distribution can be produced by modifying the ideal Hough cone to account for the desired smearing. In effect we replace the ideal Hough cone with a weighted sum of multiple Hough cones [27]. The formation of the modified voting kernel can be regarded as a convolution of the simple conical vote distribution with the appropriate point spread function [24]. If we denote the required point spread function as $P(\mathbf{x})$, then we can write

$$H = f_3 * \frac{1}{r} K * P \tag{12}$$

$$= \mathcal{F}^{-1}\{(\mathcal{F}f_3)(\mathcal{F}\tfrac{K}{r})(\mathcal{F}P)\} \tag{13}$$

$$H = \mathcal{F}^{-1}\{(\mathcal{F}f_3)(\mathcal{F}K')\} \tag{14}$$

where $K' = \frac{K}{r} * P$ denotes a modified Hough kernel that includes the effect of the point spread function and the $\frac{1}{r}$ normalisation term. The required voting point spread function would normally be determined ahead of time, allowing

$\mathcal{F}K'$ to be precomputed rather than $\mathcal{F}K$. In this case, the addition of the point spread function imposes no time or storage penalty for the overall Hough transform calculation. For some applications it might be desirable to make the kernel function adaptive by updating $\mathcal{F}K'$. In many cases such adaptation could be relatively infrequent, so the necessary calculations could be performed with low priority and $\mathcal{F}K'$ updated when practicable.

2.2 Known circle size

When the size of the target circles being sought is known a-priori, the form of the equations from the previous section can be simplified. Under these conditions the three-dimensional Hough space collapses to two dimensions as there is no need to accumulate votes for different possibilities of r. The technique described here is then identical with that described in [22]. That is, the Hough transform with known circle size can be found by two-dimensional convolution of the image with a circle.

$$H(x, y) = f(x, y) * \tfrac{1}{r} K(x, y; r) \tag{15}$$

where K has the same form as in (6), but r is now a known parameter of the circle, rather than a variable required to allow description of a cone as previously.

2.3 Edge detection

Hough transform calculations are conventionally proceeded by an edge-detection operation. This separate step can potentially be avoided when using the convolution approach, as edge detection can be incorporated into the main calculation.

As was described in Sect. 2.1, a convolution operation may be incorporated into the Hough kernel. Thus any edge-detection algorithm that can be expressed as a convolution can be included into the Hough transform without imposing any additional computational burden.

As an example, we will describe implementation of a Sobel-like filter for edge detection, though the argument can be extended to other possibilities. A Sobel filter consists of two convolution masks G_x and G_y that are independently applied to the image to detect edges in the x and y directions, respectively [28]. We can find an image f_x that contains the edges in the x-direction and and a second image f_y that contains the edges in the y-direction by performing two convolution operations. We would typically apply each of these convolution masks to an image and then combine their results in quadrature to find an edge-detected image *before* calculating a Hough transform. However, here we seek to incorporate this additional convolution as part of the Hough transform calculation.

One approach would be to separately convolve each of the Sobel filters with the Hough cone to form two edge-sensitive Hough cones. These modified cones could be used to find Hough transforms of edges in the x and y directions separately. We could then combine the results of the two to form the final Hough transform.

A more efficient approach is to allow the Hough cone to be complex. We can then encode one of the edge-detection filters in the real part of a modified Hough cone and the other in the imaginary part. This is equivalent to forming a complex combination of the Sobel masks which is then convolving with the Hough cone. The appropriate mask to encode the vertical edges in the real part of the Hough cone and the horizontal edges in the imaginary part is given by $G \in \mathbb{C}^{3\times 3}$, where

$$G = G_y + jG_x = \begin{bmatrix} 1-j & 2 & 1+j \\ -2j & 0 & 2j \\ -1-j & -2 & -1+j \end{bmatrix} \tag{16}$$

and $j = \sqrt{-1}$.

In fact the above procedure is only an approximation to a Sobel filter, as the two edge-detected images do not remain independent when multiplied by the complex Fourier transform of the Hough cone. However, as described in Sect. 3 the performance of the practical filter implemented in this way is close to the ideal Sobel filter. As a result, we will refer to a filter implemented in this way as Sobel-like.

There is some computational penalty to using a complex convolution kernel, but the penalty is small and is only paid once during the precalculation phase. There is no additional overhead in computing the Fourier transforms needed during each Hough transform. This should be compared to the conventional approach, where an edge-detection calculation is required for each input image.

A drawback of incorporating the edge detection into the Hough transform calculation is that the generated Hough transform will be complex, with the real parts of votes corresponding to vertical edges and the imaginary parts corresponding to horizontal edges. In some cases it might prove useful to separate the two sets of votes, but in many cases it will be necessary to find the magnitude in Hough space before searching for peaks.

In some cases it may be appropriate to use a simpler edge detector, such as a discrete Laplacian. In this case the Hough cone can be modified by convolution with the discrete Laplacian kernel

$$G_{\text{lap}} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{17}$$

and the Hough cone remains real.

2.4 Image padding

The technique described in Sect. 2 describes the use of the Fourier transform to implement convolution. However, it is well known that such a procedure implements circular convolution, rather than linear convolution. Consequently, the vote distribution arising from a given pixel will deviate from the ideal cone if the cone encounters the edge of the Hough space. In such cases the cone wraps around the Hough space and votes appear at an incorrect location on the other side of the side of the image.

In many cases we would expect that this wrapping would not cause any difficulties. In general the vote distribution produced in this manner is quite diffuse, with many small votes distributed across many possible circle parameters. It is unlikely that this would significantly perturb an estimate of the location of a real peak in Hough space.

Nevertheless, to prevent the wrapping effect the input image can be padded before application of the Hough transform. In the case where the convolution-based Hough transform is used to target circles with centres lying within the input image, then a border with width half the circles' largest possible radius is sufficient to prevent the wrapped votes contributing to a Hough space peak that corresponds to a circle centered within the image.

Some applications require the detection of circles that have their centres outside the image boundaries. For such problems the Hough space must be further extended to include the additional possible centre locations. To ensure that all circles with centres outside the image can be characterised the image must be padded by an amount equal to the radius of the largest targeted circle and then padding again to avoid the wrapping effect if that is necessary.

Padding of the image is also normally required when the edge-detection technique described in Sect. 2.3 is used. The edge-detection technique employs Fourier transforms that assume the image to be periodic. As images are rarely periodic either top-to-bottom, or left-to-right, application of the edge detectors described in Sect. 2.3 finds apparent edges at the periphery of the image. When convolved with the Hough cone, these edges lead to contamination of Hough space with linear distributions of misplaced votes, potentially making peak extraction more difficult.

When padding is used to prevent the wrapping effect, then simple zero padding is sufficient. However, when edge detection is in use it is important to avoid any discontinuity where the image and padding regions intersect. This requires that each of the four sides of the image be extended without discontinuity by a distance equal to the maximum radius of targeted circles. As shown in Fig. 6 this procedure ensures that the Hough space features caused by the boundaries of the image are confined to the padding regions.

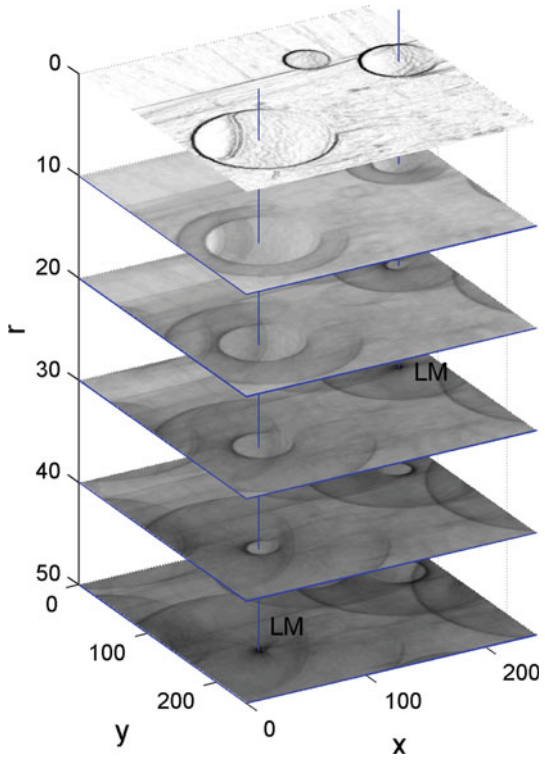

Fig. 3 Slices through the Hough transform of an image of three tennis balls sitting on a bench. The edge-detected input image is shown at $r = 0$. Local maxima of the Hough transform are indicated with the annotation 'LM' and *lines* from the maxima are projected back to the input image to show the centre of circles inferred from the Hough transform

3 Results

The Hough transform of an image of three tennis balls can be seen in Fig. 3. The image was first preprocessed by a Sobel filter to extract the edges and then the Hough transform was calculated using the convolution method as described above. No additional point spread function P (see Eq. 12) was used in this experiment or any of the subsequent experiments. Each slice in the figure shows the vote distribution for a given value of target circle radius. Local maxima in the Hough space correspond to the location and radius of circles that are in the input image. So, for example, the local maximum at $(x, y, r) = (54, 167, 50)$ in the lower part of the figure corresponds to the position and radius of the tennis ball on the lower left of the input image.

3.1 Comparison with the classical transform

Figure 4 compares the Hough transform of the tennis ball image when calculated using the conventional Hough transform and the convolution technique. The test image in this case is a grayscale image of size 243×243 pixels. A Sobel edge detector is first used on the image and then the

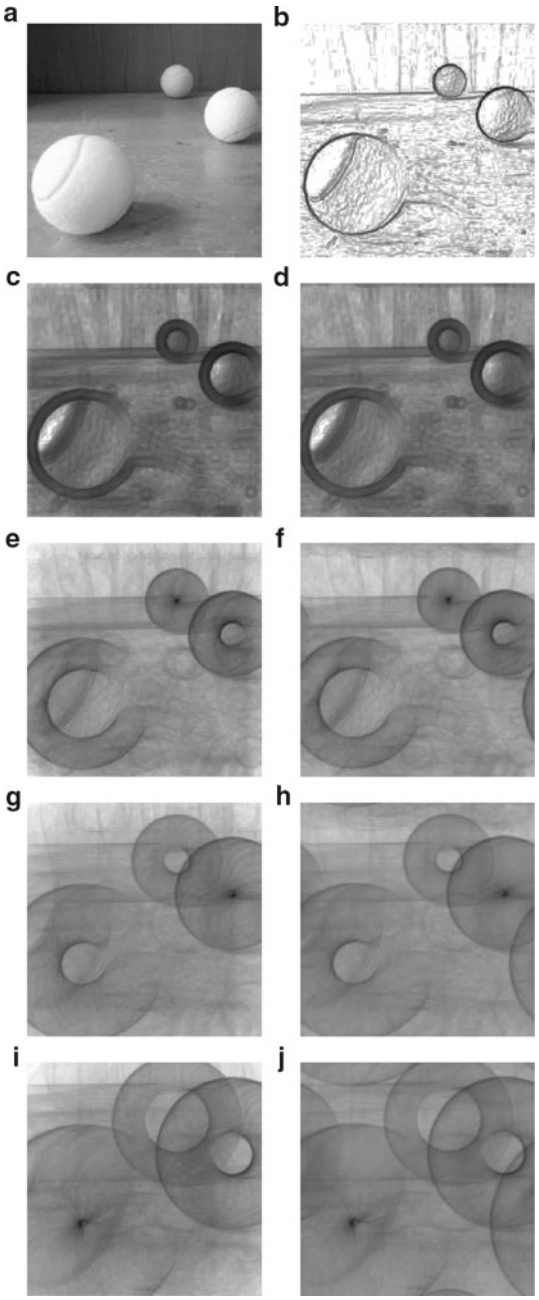

Fig. 4 Comparison of the Hough transforms calculated using the classical algorithm. The original image is shown in (**a**), with the result of edge detection using a Sobel pre-filter in (**b**). Planes of the Hough transforms calculated with the classical algorithm (on the *left*) are compared with the convolution based algorithm (on the *right*). **c**, **e**, **g**, **i** Correspond to planes in the Hough transform at $r = \{5, 16, 30, 50\}$, as do **d**, **f**, **h**, **j**

two Hough transforms are applied. The colour density of the Hough transform depictions represents the logarithm of the Hough transform, so that the features of the transforms are more apparent.

It is evident that the basic features of the convolution-based Hough transform are the same as the conventional transform. However, the effect of using circular convolution become apparent for larger values of target circle radius. As can be seen, the conical vote distributions wrap around after reaching the edge of the Hough space. This leads to anomalous arcs in the Hough transform.

Figure 5 shows the Hough transforms of an additional four input images. Again the Hough transforms calculated by the conventional and convolution methods show good agreement with the exception of wrap around effects. As such we expect the convolution technique to have similar effectiveness at finding circles as the conventional Hough transform. The figure also shows the locations of circles corresponding to peaks in the convolution-derived Hough transform, demonstrating the effectiveness of the technique.

In many cases the wrapping of the vote distribution is not of great concern, as the arcs form a diffuse distribution of anomalous votes. As the number of anomalous votes in any location of Hough space is low, it is unlikely to perturb the apparent position of a real peak in Hough space. However, as described in Sect. 2.4, when necessary the effect can be removed by padding the input image before processing. Figure 6 shows the effectiveness of this approach. This image was produced using an identical procedure to that for Fig. 4 except that the input image was first smoothly extended by 30 pixels before producing the transform on the right. As can be seen, the extra space bordering the image provides room for the vote distribution so it does not wrap.

The effectiveness of incorporating the Sobel edge-detection filter into the Hough transform is illustrated in Fig. 7. This version of the Hough transform is produced by omitting the initial edge-detection step, but modifying the Hough cone to include the Sobel filter.

Figure 7a shows the linear features in Hough space that are caused by the non-periodicity in the input image. Under some circumstances these features may produce an erroneous characterisation of a peak in Hough space. For example the figure shows that the edge artifact has placed votes near the peak of the rightmost tennis ball. This perturbation of the correct vote distribution leads to a slight error in the estimate of the centre of the ball.

Where estimation errors are to be minimised, the input image can be padded. For example the Hough transform in Fig. 7b was obtained by extending the input image smoothly by 30 pixels on all sides. This ensures that the effects of the apparent edges induced by the image boundary do not intrude into the central part of the Hough transform.

A comparison of three methods of performing the edge detection can be seen in Fig. 8. For this illustration an 80×80 pixel section of the input image was selected (a) and padded to avoid edge effects as described above. The Hough transform

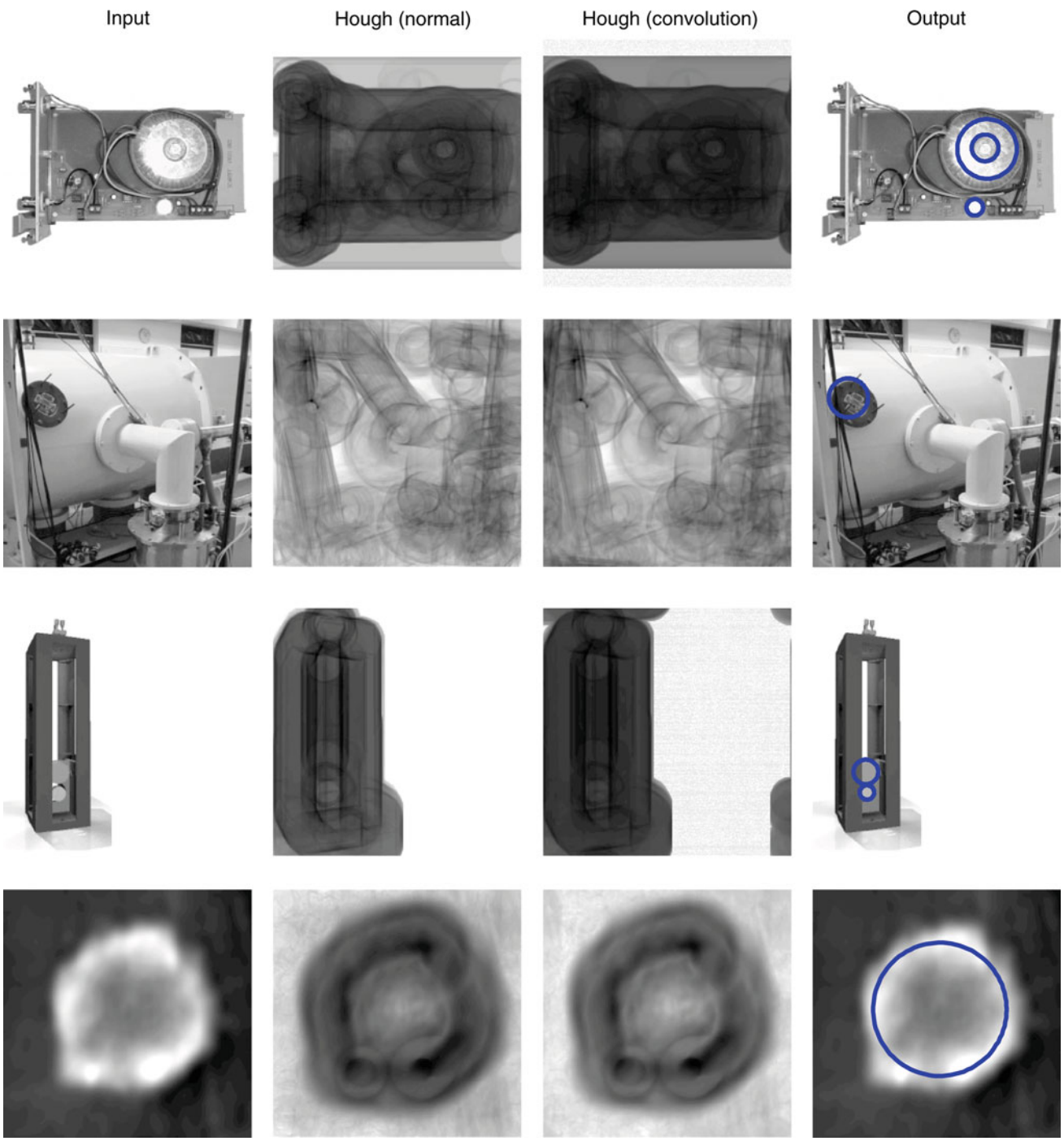

Fig. 5 Comparisons of the Hough transform calculated by the conventional- and convolution-based techniques. The *first column* shows a set of four input images. The *second* and *third columns* display the $r = 20$ layer of the Hough transform calculated by the conventional (column 2) and convolution (column 3) methods. The *fourth column* shows the input images with Hough transform derived circles superimposed upon them. The fourth input image is the supernova remnant G337.3+1.0 taken from the Molonglo Observatory Synthesis Telescope Supernova Remnant Catalogue [29]

was then calculated by pre-filtering with a Sobel-filter (c), incorporating the Sobel-like filter into a modified Hough cone (d), and incorporating a discrete Laplacian filter into the Hough cone (e). The figure shows the shape of the peaks in Hough space corresponding to the location of the tennis ball.

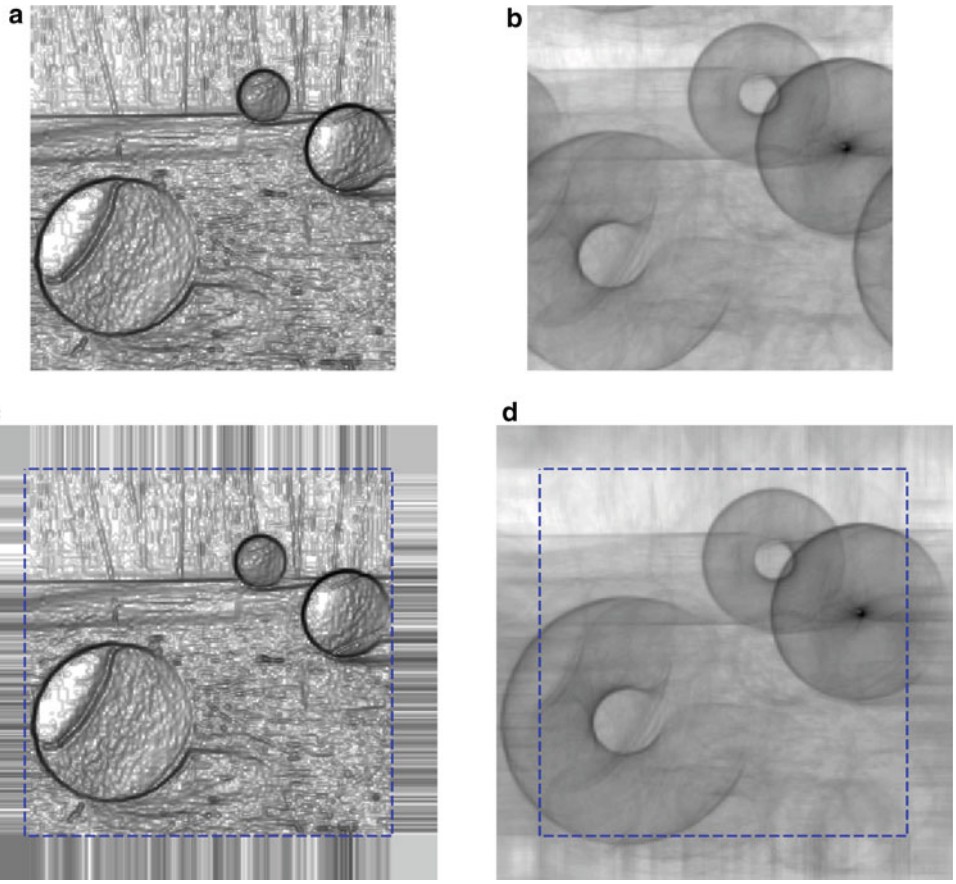

Fig. 6 Comparison of the $r = 30$ plane of the convolution-based Hough transform for the unpadded (**b**) and padded (**d**) algorithms. **a** The edge-detected input image used to produce **b**, and **c** the image padded by 30 pixels on each edge. The area corresponding to the original image is indicated by the *dashed box*

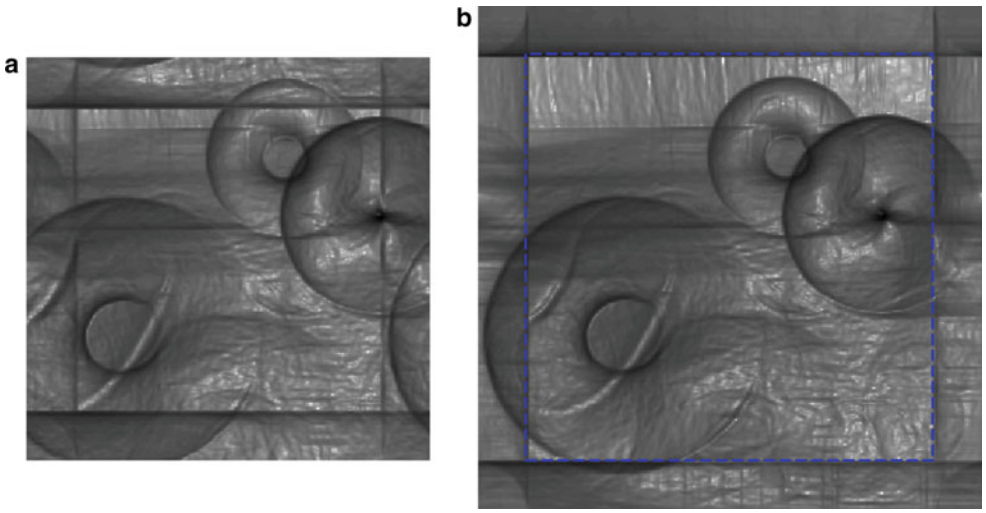

Fig. 7 The $r = 30$ plane of the Hough transform calculated using the Sobel edge-detection filter. **a** No padding and **b** the image padded by 30 pixels on each edge. The *dashed box* indicates the area of the original image

As can be seen there is a small but discernable difference between using a Sobel pre-filter and including the Sobel-like filter in the Hough transform. The Laplacian edge filter does not perform as well, but may be sufficient for some applications.

3.2 Computational complexity

The convolution method was used to calculate the Hough transform of a variety of images of different sizes. The integrated Sobel filter described in Sect. 2.3 was used throughout,

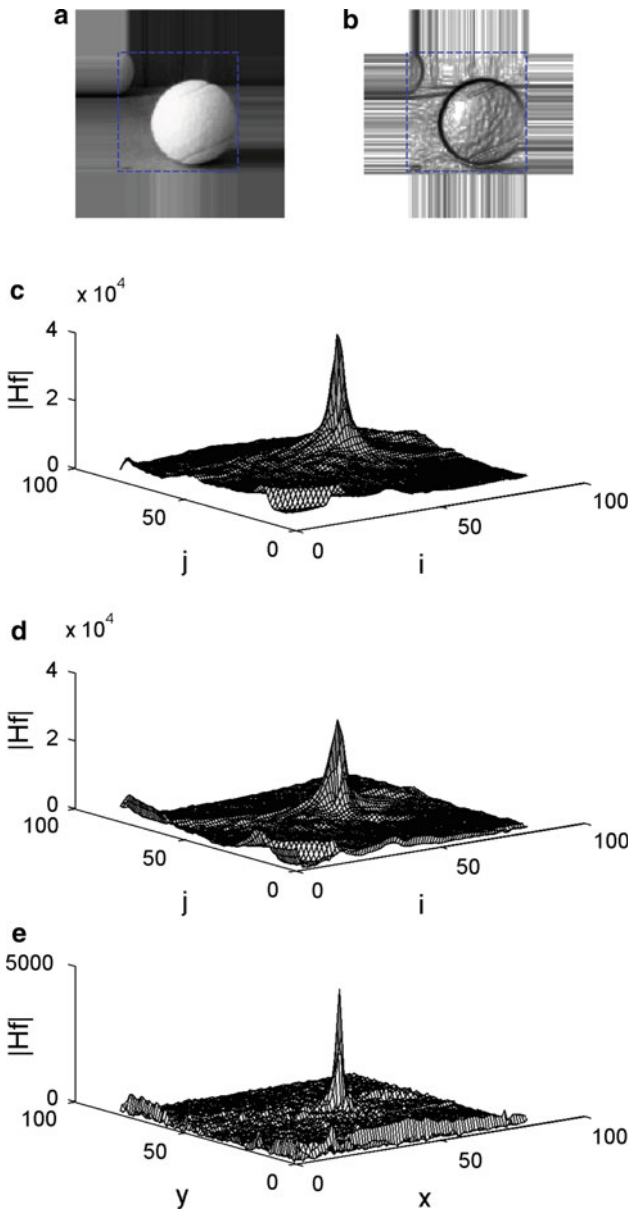

Fig. 8 Vote distributions in the $r = 30$ plane of the Hough transform calculated using three different edge-detection techniques. **a** The smoothly padded test image and **b** is the result of using a Sobel filter to extract the edges. **c** The Hough transform of the edge-detected image, **d** is the Hough transform calculated with the Sobel-like filter included in the Hough cone and **e** is the Hough transform calculated with a Laplacian filter included into the Hough cone. In each case the displayed portion of the Hough transform corresponds to the area of the original image indicated by the *dashed box* in (**a**)

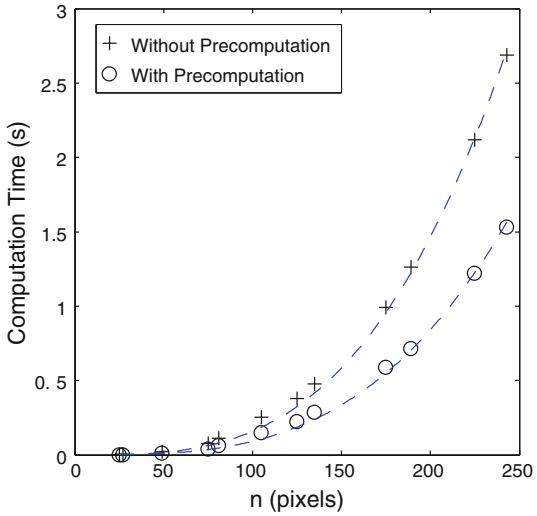

Fig. 9 Median time taken to complete a Hough transform of an $n \times n$ pixel image using the convolution technique. The *crosses* show the computation times with no precomputation of the Hough cone. The *circles* indicate the computation when $\mathcal{F}K'$ was precalculated. In each case the *dashed line* shows the best fitting curve with time increasing as $n^3 \log n$

but no additional filtering or image padding was employed. In all cases, the full range of target circle radii from one pixel to the image size was explored.

For each image size the transform was calculated ten times with no access to a precomputed transform of the Hough cone, $\mathcal{F}K'$. For comparison, each transform was also calculated ten times with $\mathcal{F}K'$ available. In each case there was no appreciable variation in computation from run to run, so the median run time is used for the following analysis. All transforms were calculated on a machine with a 3 GHz Intel Core 2 Duo processor and 3.25 GHz of available RAM. MATLAB® was used to perform the calculations.

Figure 9 shows the time taken to complete the Hough transforms when $\mathcal{F}K'$ was not precomputed. As expected, the transform becomes significantly faster when $\mathcal{F}K'$ is precomputed because only a single three-dimensional Fourier transform is then required per Hough transform, rather than the two needed when $\mathcal{F}K'$ is also calculated. Figure 9 also shows the computation time with precomputation of the $\mathcal{F}K'$ and as suggested the computation time is approximately halved.

Comparison of the experimental data with the dashed $N^3 \log N$ curves in each figure demonstrates that the scaling of the computation time is in good agreement with the predicted computational complexity.

4 Conclusions

Reformulation of the Hough transform as a convolution provides a useful alternative to existing Hough transform algorithms. The new algorithm a computational complexity of $O(n^3 \log n)$, which is better than the $O(n^4)$ complexity of the original transform algorithm. Implementation of the new approach is very simple, arguably simpler than any of the available alternatives.

Any image processing operation that can be expressed as a convolution (such as filtering) can be incorporated into the Hough transform calculation, rather than being carried out in either pre- or post-processing. As Sobel (and similar) edge-detection filters can be expressed as convolutions, the new method allows the elimination of the separate edge-detection step often required when finding the Hough transform.

Use of the discrete Fourier transform to implement convolution results in wrapping of the vote distribution around the edges of Hough space, an effect that does not occur with conventional Hough transform. However, the misplaced votes are well dispersed and therefore unlikely to cause difficulties. In cases where they are problematic then the wrapping can be avoided by padding the input image so that the vote distributions do not intrude into the undesirable portion of Hough space.

Unlike some previous implementations of the Hough transform, the calculation time of the convolution-based Hough transform is independent of the complexity of the input image. This makes the approach useful when calculation time cannot be allowed to increase in the presence of visual clutter and/or noise. This combination of deterministic calculation time and relatively high speed means that the convolution-based algorithm is well suited to real-time systems.

Acknowledgments We would like to thank the anonymous reviewers of this paper for their careful and constructive comments.

References

1. Illingworth, J., Kittler, J.: A survey of the Hough transform. Comput. Vis. Graph. Image Process. **44**(1), 87–116 (1988)
2. Borkar, A., Hayes, M., Smith, M.: Robust lane detection and tracking with ransac and Kalman filter. In: 16th IEEE International Conference on Image Processing (ICIP), pp. 3261–3264 (2009)
3. Zhao, C.J., Jiang, G.Q.: Baseline detection and matching to vision-based navigation of agricultural robot. In: International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR), pp. 44–48 (2010)
4. Zhang, M., Li, K., Liu, Y.: Head pose estimation from low-resolution image with Hough forest. In: Chinese Conference on Pattern Recognition (CCPR), pp. 1–5 (2010)
5. Satzoda, R., Sathyanarayana, S., Srikanthan, T., Sathyanarayana, S.: Hierarchical additive Hough transform for lane detection. IEEE Embed. Syst. Lett. **2**(2), 23–26 (2010)
6. Khairosfaizal, W., Nor'aini, A.: Eyes detection in facial images using circular Hough transform. In: 5th International Colloquium on Signal Processing Its Applications (CSPA 2009), pp. 238–242 (2009)
7. Liew, L.H., Lee, B.Y., Chan, M.: Cell detection for bee comb images using circular Hough transformation. In: International Conference on Science and Social Research (CSSR), pp. 191–195 (2010)
8. Xu, L., Oja, E., Kultanen, P.: A new curve detection method: randomized Hough transform (RHT). Pattern Recognit. Lett. **11**(5), 331–338 (1990)
9. McLaughlin, R.A.: Randomized Hough transform: improved ellipse detection with comparison. Pattern Recognit. Lett. **19**(3–4), 299–305 (1998)
10. Chiu, S., Lin, K., Liaw, J.: A fast randomized Hough transform for circle/circular arc recognition. Int. J. Pattern Recognit. Artif. Intell. **24**(3), 457–474 (2010)
11. Kimme, C., Ballard, D., Sklansky, J.: Finding circles by an array of accumulators. Commun. ACM **18**(2), 120–122 (1975)
12. Ballard, D.H.: Generalizing the Hough transform to detect arbitrary shapes. Pattern Recognit. **13**(2), 111–122 (1981)
13. Davies, E.R.: A modified Hough scheme for general circle location. Pattern Recognit. **7**(1), 37–43 (1988)
14. Kerbyson, D., Atherton, T.: Circle detection using Hough transform filters. In: Fifth International Conference on Image Processing and Its Applications, pp. 370–374 (1995)
15. Yuen, H., Princen, J., Illingworth, J., Kittler, J.: Comparative study of Hough transform methods for circle finding. Image Vis. Comput. **8**(1), 71–77 (1990)
16. Ioannou, D., Huda, W., Laine, A.F.: Circle recognition through a 2D Hough transform and radius histogramming. Image Vis. Comput. **17**(1), 15–26 (1999)
17. Gerig, G., Klein, F.: Fast contour identification through efficient Hough transform and simplified interpretation strategy. In: Proceedings of the 8th International Joint Conference on Pattern Recognition, pp. 495–500 (1986)
18. Hollitt, C.: Reduction of computational complexity of Hough transforms using a convolution approach. In: Proceedings of the 24th International Conference on Image and Vision Computing New Zealand IVCNZ '09, pp. 373–378 (2009)
19. Brown, C.M.: Inherent bias and noise in the Hough transform. IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-5**(5), 493–505 (1983)
20. Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. Commun. ACM **15**(1), 11–15 (1972)
21. Sklansky, J.: On the Hough technique for curve detection. IEEE Trans. Comput. **C-27**(10), 923–926 (1978)
22. Atherton, T.J., Kerbyson, D.J.: Size invariant circle detection. Image Vis. Comput. **17**(11), 795–803 (1999)
23. Shapiro, S.: Properties of transforms for the detection of curves in noisy pictures. Comput. Graph. Image Process. **8**(2), 219–236 (1978)
24. Niblack, W., Petkovic, D.: On improving the accuracy of the Hough transform. Mach. Vis. Appl. **3**(2), 87–106 (1990)
25. Strauss, O.: Reducing the precision/uncertainty duality in the Hough transform. In: Proceedings of the International Conference on Image Processing, 1996, vol. 1, 2, pp. 967–970 (1996)
26. Fernandes, L.A., Oliveira, M.M.: Real-time line detection through an improved Hough transform voting scheme. Pattern Recognit. **41**(1), 299–314 (2008)
27. Princen, J., Illingworth, J., Kittler, J.: A formal definition of the Hough transform: properties and relationships. J. Math. Imaging Vis. **1**(2), 153–168 (1992)
28. Shapiro, K.G.; Stockman, G.C.: Computer Vision, p. 146. Prentice-Hall, Englewood Cliffs (2001)
29. Whiteoak, J.B.Z., Green, A.J.: The most supernova remnant catalogue (MSC). Astron. Astrophys. Suppl. **118**, 329–380 (1996)

Author Biography

Christopher Hollitt is a lecturer at Victoria University of Wellington. His research interests are in image processing, control engineering and analogue electronics for applications in robotics, instrumentation and astronomy. He obtained BE (Elec), BSc and PhD degrees from the University of Adelaide in 1994,1996 and 2007, respectively.