9th Symposium on Sensing via Image Information (SSII2002), July 2002, Yokohama, Japan, pp. 355–360.

Automatic Detection of Circular Objects by Ellipse Growing

Kenichi Kanatani† Naoya Ohta‡

†Okayama University, Okayama 700-8530 Japan

‡Department of Computer Science, Gunma University, Kiryu, Gunma 376-8515 Japan

E-mail: kanatani@suri.it.okayama-u.ac.jp, ohta@cs.gunma-u.ac.jp

Abstract

We present a new method for automatically detecting circular objects in images: we detect an osculating circle to an elliptic arc using a Hough transform, iteratively deforming it into an ellipse, removing outlier pixels, and searching for a separate edge. The voting space for the Hough transform is restricted to one and two dimensions for efficiency, and special weighting schemes are introduced to enhance the accuracy. We demonstrate the effectiveness of our method using real images. Finally, we apply our method to the calibration of a turntable for 3-D object shape reconstruction.

1 Introduction

Since a circle in the scene is projected into an ellipse in the image, circular objects in the scene can be detected by finding ellipses in the image. Recently, many algorithms have been proposed for fitting an ellipse to edge pixels with high precision [4, 5, 13, 14]. Using the equation of the fitted ellipse, the 3-D position and orientation of the circular object can be computed analytically [8, 19].

However, finding correct edge pixels to fit an ellipse is still a very difficult problem. One approach is to segment digital curves into linear and curved parts, fitting lines to the linear parts and ellipses to the curved parts [6, 16]. The segmentation is based on the digital curvature, the residual of fitting, and miscellaneous heuristics, but the accuracy of the fit depends very much on segmentation. This may not be a great problem for visual applications but is crucial for precise measurement in robotics applications, with which we are concerned.

Most research in the past has been focused on directly searching for an ellipse using the Hough transform. Since an ellipse is specified by five parameters, we can find it by voting in a quantized 5-dimensional Hough space the parameters of the ellipses that could pass through each edge pixel and picking out the value that wins the maximum number of votes. However, directly computing this would require a long computation time and a memory space proportional to the fifth power of the quantization levels. Various techniques have been proposed for reducing the computation time and memory space. For example, the computation is divided into a cascade: voting is done in a 2-dimensional Hough space with the remaining three parameters fixed, and this is repeated for all quantized values of the three parameters. In this process, one can introduce various constraints for pruning unnecessary search, hierarchically change the resolution of the image and the Hough space, and do random sampling instead of exhaustive search, a variant of this being the genetic algorithm. Instead of using edge pixels alone, one can also use complex primitives and extra information such as point pairs, triplets, edge segments, and their orientations.

Although many extensions and variations have been proposed in the past [1, 3, 10, 11, 12, 15, 17, 18, 21, 22, 23, 24], the Hough transform alone is not efficient enough, while the use of digital curves alone does not warrant sufficient accuracy. In this paper, we integrate these two: we detect a circle osculating to an edge segment by the Hough transform, find edge pixels tangent to it, refit an ellipse to the detected pixels, and repeat this process. We call these iterations *ellipse growing*.

A closely related method to ours was proposed by Asayama and Shiono [1], who detected via the 5-dimensional Hough transform the two circles osculating to an ellipse from both sides, from which they computed the parameters of the ellipse. Their method works only when the entire ellipse is visible without occlusion, and a long computation time and a large memory space are required for the Hough transform. In contrast, our method is able to fit an ellipse to a partially occluded image of a circular object very efficiently with high precision.

The Hough transform we propose for detecting an osculating circle is made efficient by using, instead of the standard 3-dimensional Hough space, a 2-dimensional array for the center and a 1-dimensional array for the radius. We also introduce special weight functions for enhancing the vote peaks. We then iteratively deform the osculating circle into an ellipse. We also remove outlier edge pixels by LMedS, and search for another edge segment to combine. Fig. 1 shows the flow chart of the procedure.

Our system is not intended to have universal merits, whose pursuit is rather unrealistic. We have in mind robotics environments where

1. only a small number of circular objects exist in the field of view, and
2. the circular objects we are looking at have a

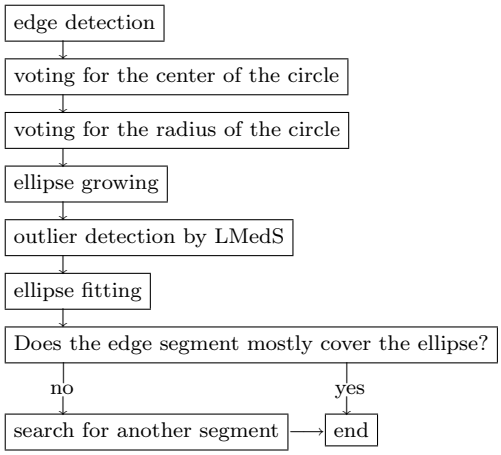


Figure 1 Flow chart of the procedure.

clearly visible size in the image.

We do not consider such complicated scenes as aggregates of circular and elliptic particles or cells. Our method is strengthened by specifically exploiting this situation.

In the following, we describe each step separately and confirm the effectiveness of our method using real images. Finally, we apply our method to the calibration of a turntable for 3-D object shape reconstruction.

2 Procedure

2.1 Edge detection

From among many edge detection techniques available, we adopt the following method for obtaining long edge segments without branches. First, we cut out from the zero-crossing edges of one-pixel width those parts with high Sobel values. Starting from an arbitrary pixel of the resulting edges, we trace the 8-connected neighbors in both directions until we arrive at an end-point or a branch point. This defines a connected edge segment without branches. The judgment for an end-point or a branch point is done by checking the number of edge pixels adjacent to it in its 8-connected neighborhood. This process is repeated until all the edge pixels are exhausted. Among the resulting edge segments, we discard short ones (less than 60 pixels in our experiment) and register the remaining ones in an edge segment list.

2.2 Voting for the center

In the resulting list of edge segments, we choose one which is nearly circular and fit a circle to it as an approximation to an ellipse. To do this, we first estimate the center of the circle that fits to the segment.

For an exactly circular arc, its center is easily estimated by voting along all lines normal to the arc [7]. However, the lines normal to an ellipse envelop a star-like curve with four cusp vertices known as the *evolute* of the ellipse (Fig. 2). An osculating circle whose center is on the evolute is generally of second degree contact to the curve, but if the center is at one of the four singularities of the evolute, the degree of contact is three [2]. Hence, if we vote along all lines

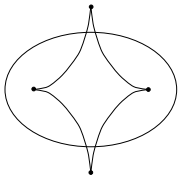


Figure 2 The evolute of an ellipse and its singularities.

normal to an elliptic arc, we obtain ridges along the evolute and peaks at its singularities. However, the peaks are less conspicuous than for a circular arc. So, we directly vote on the evolute as follows.

For each pixel P on the edge segment, we compute the center C and the radius r of the circle that passes through P and two points on the segment k pixels away from it on both sides (we exclude the k end points of the segment). Then, we vote around C with Gaussian diffusion of standard deviation γr in all directions (Fig. 3(a)). We also weight each vote by $1/\sqrt{r}$ to give preference to edge segments of higher curvature if the length is the same and to edge segments of larger radii if the central angle is the same. In our experiment, we set $k = 30$ and $\gamma = 1/10$. Since it is difficult to derive theoretically optimal values for the parameters and weights, we set them heuristically.

We do this for all the edge segments in the list[1], detect peaks of the votes, and take as candidates the four pixels that win the largest numbers of votes (a real image example is given in Fig. 5; the details are described in Sec. 3).

2.3 Voting for the radius

For each center C thus detected, we next estimate the corresponding radius by voting the distance R from C to each edge point; we adopt the value that wins the maximum number of votes. This may be obvious for a circular arc [7], but for an elliptic arc the number of false peaks increases. So, we adopt the following strategy.

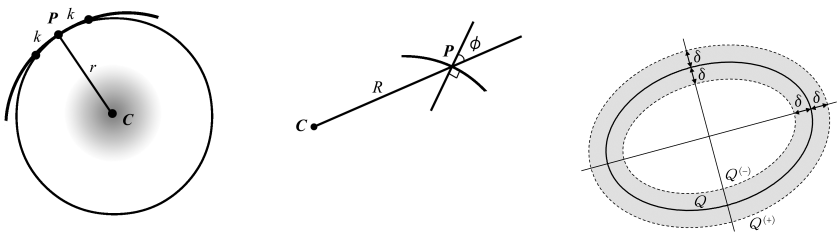
First, instead of casting one vote to the distance R of each edge pixel P from the detected center C, we cast a *positive/negative real value* $e^{-R^2/2s^2}\cos\phi$, where ϕ is the angle between the line CP and the normal direction to P (Fig. 3(b)), which is computed from the output of the Sobel operator.

Second, we vote not merely for an integer obtained by rounding R but also the two integers on both sides by ± 1 with the weight $e^{-1/2}$ (i.e., the Gaussian weight of standard deviation 1).

The value[2] $e^{-R^2/2s^2}$ effectively limits the voting to be approximately within distance s from C (in our experiment, we empirically set s to be 1/4 of the image size). Due to $\cos\phi$, those edge segments that

[1] In our experiment, we voted every three pixels for efficiency. We also introduced various approximations such as limiting the voting region to a parallelogram with one edge orthogonal to the edge segment and the other parallel to one of the coordinate axes.

[2] We used a polynomial approximation of e^{-x^2} for efficiency.



(a) (b) (c)

Figure 3 (a) Voting around the center of the circle passing through P and the two points away from it by k pixels on both sides. (b) A pixel away from the estimated center C by distance R votes the value R with weight $e^{-R^2/2s^2}\cos\phi$. (c) Fitting an ellipse to the longest edge segment inside the region within distance δ from the estimated ellipse.

make large angles from the circumference direction are given small weights. In particular, *the weight is 0 if the segment is oriented along the radius direction*, and edge segments with opposite gray-level gradients have weights of *opposite signs*, so the contributions from closed loops that do not include P inside are mutually canceled, enhancing only the contributions from the edge segment to which the circle osculates (see Fig. 6 for this effect; the details are described in Sec. 3).

After this voting, we choose the value that wins the maximum number of votes. We do this for each of the four candidates for the center and pick out the one that wins the largest number of votes for the radius.

2.4 Ellipse growing

Regarding the detected osculating circle as the initial ellipse, we make it grow into an ellipse that fits to the segment better. For the current ellipse, we define a ring region around it by expanding and contracting the ellipse by δ pixels along its major and minor axes (Fig. 3(c)). From among the registered edge segments, we choose the one that has the largest number of consecutive pixels inside this region and fit an ellipse to it. For this, we used a technique called *renormalization*[3], which is known to attain the highest possible accuracy in the first order [9]. Around the fitted ellipse, we recompute a ring region, to which we apply the same procedure and repeat this until no new pixels are added.

The ring region is defined as follows. Quadratic curves (including ellipses) have equations of the form

$$Ax^2 + 2Bxy + Cy^2 + 2f(Dx + Ey) + f^2F = 0, \quad (1)$$

where f is an arbitrarily fixed constant (e.g., the image size). The condition that this equation describes an ellipse is

$$AC - B^2 > 0. \quad (2)$$

If we define the vector $\boldsymbol{x}$ and the matrix $\boldsymbol{Q}$

$$\boldsymbol{x} = \begin{pmatrix} x/f \\ y/f \\ 1 \end{pmatrix}, \qquad \boldsymbol{Q} = \begin{pmatrix} A & B & D \\ B & C & E \\ D & E & F \end{pmatrix}, \quad (3)$$

[3]The program is publicly available from `http://www.ail.cs.gunma-u.ac.jp/Labo/programs-e.html`.

eq. (1) is rewritten in the form

$$(\boldsymbol{x}, \boldsymbol{Q}\boldsymbol{x}) = 0, \quad (4)$$

where $(\boldsymbol{a}, \boldsymbol{b})$ denotes the inner product of vectors $\boldsymbol{a}$ and $\boldsymbol{b}$. The matrices $\{\boldsymbol{Q}^{(+)}, \boldsymbol{Q}^{(-)}\}$ of the expanded and contracted ellipses are computed as follows:

1. Compute the matrix $\boldsymbol{S}$, the vector $\boldsymbol{c}$, and the scalar c as follows:
$$\boldsymbol{S} = \begin{pmatrix} A & B \\ B & C \end{pmatrix}, \qquad \boldsymbol{c} = \begin{pmatrix} D \\ E \end{pmatrix},$$
$$c = (\boldsymbol{c}, \boldsymbol{S}^{-1}\boldsymbol{c}) - F. \quad (5)$$
2. Compute the eigenvalues $\{\lambda_1, \lambda_2\}$ of $\boldsymbol{S}$ and the corresponding unit eigenvectors $\{\boldsymbol{u}_1, \boldsymbol{u}_2\}$.
3. Compute $\lambda_1^{(\pm)}$ and $\lambda_2^{(\pm)}$ as follows:
$$\lambda_1^{(\pm)} = \frac{c}{(\sqrt{c/\lambda_1} \pm f\delta)^2}, \quad \lambda_2^{(\pm)} = \frac{c}{(\sqrt{c/\lambda_2} \pm f\delta)^2}. \quad (6)$$
4. Compute the matrices $\boldsymbol{S}^{(+)}$ and $\boldsymbol{S}^{(-)}$ by
$$\boldsymbol{S}^{(\pm)} = \boldsymbol{U} \begin{pmatrix} \lambda_1^{(\pm)} & \\ & \lambda_2^{(\pm)} \end{pmatrix} \boldsymbol{U}^\top, \quad (7)$$
where $\boldsymbol{U}$ is the 2×2 matrix consisting of vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ as its columns in that order.
5. Compute the matrices $\{\boldsymbol{Q}^{(+)}, \boldsymbol{Q}^{(-)}\}$ as follows:
$$\boldsymbol{Q}^{(\pm)} = \begin{pmatrix} \boldsymbol{S}^{(\pm)} & \boldsymbol{S}^{(\pm)}\boldsymbol{S}^{-1}\boldsymbol{c} \\ (\boldsymbol{S}^{(\pm)}\boldsymbol{S}^{-1}\boldsymbol{c})^\top & F + (\boldsymbol{c}, \boldsymbol{S}^{-1}(\boldsymbol{S}^{(\pm)} - \boldsymbol{S})\boldsymbol{S}^{-1}\boldsymbol{c}) \end{pmatrix}. \quad (8)$$

Then, the ring region around the ellipse of eq. (4) is define by

$$(\boldsymbol{x}, \boldsymbol{Q}^{(+)}\boldsymbol{x})(\boldsymbol{x}, \boldsymbol{Q}^{(-)}\boldsymbol{x}) < 0. \quad (9)$$

The renormalization fits a general quadratic curve of the form of eq. (1), so it sometimes fits a hyperbola or a parabola when the segment is very short due to occlusion. In our experiment, we initially set $\delta = 4$ (pixels) and incremented it one by one as long as the condition (2) was not satisfied. If δ reached 10, we returned the initial circle and went on to the next process (see Fig. 7 for a real image example; the details are described in Sec. 3).

2.5 Outlier removal

The edge pixels to which the ellipse is thus fitted need not be part of an ellipse. So, we remove "outliers[4] " by *LMedS* [20].

Let $\{\boldsymbol{x}_\alpha\}$, $\alpha = 1, ..., N$, be the vector representations (see eq. (3)) of the pixels to fit an ellipse. Initializing the matrix $\boldsymbol{Q}_m = \boldsymbol{O}$ and the scalar $S_m = \infty$, we repeat the following computation until it converges:

1. Randomly choose five points from $\{\boldsymbol{x}_\alpha\}$.
2. Compute an ellipse $\boldsymbol{Q}$ that passes through them.

\# This is done by solving simultaneous linear equations in A, B, ..., F in the form of eq. (1) up to a constant factor.

3. Go back to Step 1 if eq. (3) is not satisfied.
4. Compute the following median:

$$S = \mathrm{med}_{\alpha=1}^{N} \frac{(\boldsymbol{x}_\alpha, \boldsymbol{Q}\boldsymbol{x}_\alpha)^2}{\|\boldsymbol{P}_k \boldsymbol{Q}\boldsymbol{x}_\alpha\|^2}. \tag{10}$$

\# We define $\boldsymbol{P}_k = \mathrm{diag}(1, 1, 0)$ (the diagonal matrix with 1, 1, and 0 as the diagonal elements in that order). The quotient on the right-hand side equals the squared distance of the αth pixel to the fitted ellipse to a first approximation [9]:

5. If $S < S_m$, update $\boldsymbol{Q}_m \leftarrow \boldsymbol{Q}$ and $S_m \leftarrow S$.

In our experiment, we stopped the iterations if $S \geq S_m$ for 10 consecutive times. Then, we selected inliers by the criterion

$$\frac{(\boldsymbol{x}_\alpha, \boldsymbol{Q}\boldsymbol{x}_\alpha)^2}{\|\boldsymbol{P}_k \boldsymbol{Q}\boldsymbol{x}_\alpha\|^2} < 10 S_m, \tag{11}$$

which can be obtained by estimating the data standard deviation σ according to the formula $\hat{\sigma} \approx 1.4826\sqrt{S_m}$ given in [20] and setting the threshold to $(2.13\hat{\sigma})^2$. Finally, an ellipse $\boldsymbol{Q}$ is refitted to the detected inliers by renormalization, and the longest segment e that covers all the inliers is cut out.

2.6 Search for another segment

In order to cope with ellipses consisting of multiple segments, we do the following search.

First, we check if the detected segment e covers more than half of the entire ellipse. This is done by checking if the end parts of e are in converging orientations (Fig. 4). If so, we stop and return the ellipse $\boldsymbol{Q}$ we have obtained. Otherwise, we expand $\boldsymbol{Q}$ by $1 + \gamma$. This is done by replacing eqs. (6) by

$$\lambda_1^{(+)} = \frac{\lambda_1}{(1+\gamma)^2}, \qquad \lambda_2^{(+)} = \frac{\lambda_2}{(1+\gamma)^2}. \tag{12}$$

(We let $\gamma = 1.2$). We randomly choose a segment e' that is within the expanded ellipse and is not too short (we ignored those of less than 20 pixels). Randomly choosing four pixels from e and one pixel from e', we fit an ellipse by LMedS as described earlier. This time, we evaluate, instead of eq. (10), the sum of the medians computed for e and e' separately. We repeat this until the sum of the medians converges (we stopped if no update occurred 100 consecutive times). We return the initial $\boldsymbol{Q}$ if the resulting sum of medians is larger than four times the initial median for e. Otherwise, we detect inliers by applying eq. (11) to e and e' separately and fit an ellipse to the detected inliers by renormalization.

[4]In the statistical literature, outliers are the data generated by unmodeled random fluctuations [20]. Here, they are contiguous pixels on non-elliptic edge segments. In other words, "outlier removal" actually means "segmentation" of elliptic parts.

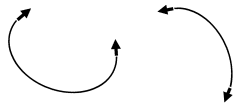


Figure 4 Judging if the edge segment covers more than half of the ellipse.

We can find more segments by repeating this procedure. However, the chance to pick out wrong segments will increase. Also, finding two segments is usually sufficient for robustly fitting an ellipse. So, we stop the search after one separate segment is found. Since it is difficult to derive theoretically the best strategy for searching and voting, we introduced the above procedure empirically.

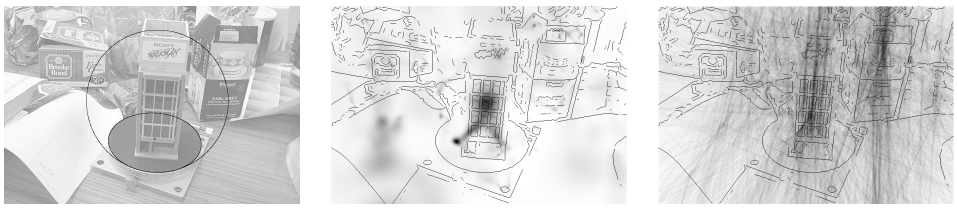
3 Real Image Examples

Fig. 5(a) shows the detected osculating circle and the final ellipse superimposed onto the original image. Fig. 5(b) is the initial edge image, onto which the votes for possible centers of osculating circles are superimposed in gray levels. For comparison, Fig. 5(c) shows the result obtained by the standard Hough transform: we vote along the normal line to each edge pixel with Gaussian diffusion of one-pixel standard deviation on both sides [7]. As we can see, our scheme concentrates more votes along the evolute of the ellipse, in particular at its singularities.

Fig. 6(a) plots the absolute value (recall that we vote positive and negative values in our scheme) of the number of votes (ordinate) for the radius R (abscissa) from the detected center detected. For comparison, Fig. 6(b) shows the plot obtained by the standard Hough transform: the distance from the estimated center to each edge pixel is voted with Gaussian diffusion of one-pixel standard deviation [7]. We adjusted the scale so that the total number of votes is equal for both. As we can see, our scheme reduces the contributions from clustered edges to almost zero because of the cancellation of the signs, enhancing only the contributions from isolated arcs.

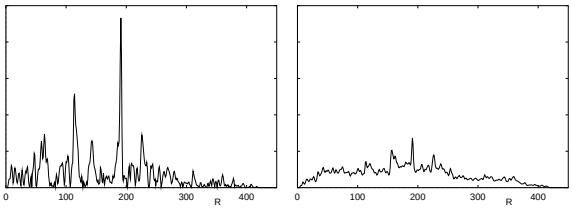
Fig. 7(a) shows an edge image of a partially occluded ellipse. For this image, a hyperbola was fitted in the ring region along the initial circle even for $\delta = 10$. So, we fitted an ellipse by LMedS and went on to search for another segment. Fig. 8(b) shows (1) the initial circle, (2) the hyperbola resulting from the ellipse growing, (3) the ellipse fitted by LMedS, and (4) the ellipse fitted after detecting another segment.

Fig. 8 shows various examples, showing the edge images (upper rows) and the initial circles and the final ellipses superimposed onto the original images



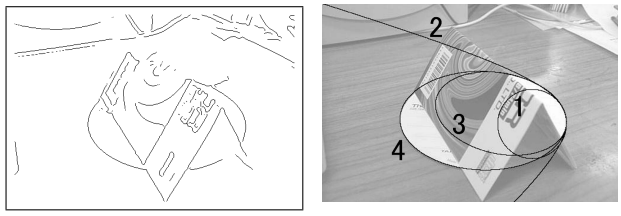
(a) (b) (c)

Figure 5 Estimation of the center of the osculating circle. (a) The detected osculating circle and the fitted ellipse. (b) Voting for the candidates of the center. (c) Standard Hough transform.



(a) (b)

Figure 6 The number of votes (ordinates) for the radius R of the initial circle (abscissa). (a) Proposed method. (b) Standard Hough transform.



(a) (b)

Figure 7 (a) The edge image used. (b) (1) The initial circle, (2) the hyperbola resulting from the ellipse growing, (3) the ellipse fitted by LMedS, and (4) the ellipse fitted after detecting another segment.

(lower rows). In all cases, one or two iterations of the ellipse growing produced almost satisfactory shapes, and the iterations terminated after four or five runs. As we see, we can obtain good fits even when the initial segments are very short due to occlusion.

We used Pentium III for the CPU and Linux for the OS. For 500×333-pixel images, the average computation time was 18 seconds, most of which was spent on the preprocess of edges. As a result, computation time is nearly proportional to the number of edge segments. However, there is much room to increase efficiency by refining the program code.

As a final example, we show an application to the calibration of a turntable. One of the widely used methods for 3-D shape reconstruction is to place an object on a rotating turntable and take its images. Since this is equivalent to rotating the camera around the object and taking its images, the object shape can be reconstructed by the standard triangulation. To do this, we need to calibrate the position of the turntable relative to the camera. For a circular turntable, this is easily done using its camera image alone if we measure its radius [8, 19].

Fig. 9 (a) is an edge image of a circular turntable, and Fig 9(b) shows the initially detected circle and the finally fitted ellipse superimposed onto the original image (768×512 pixels). The focal length was calibrated to be 700 pixels, and the radius of the turntable was measured to be 6.45cm. The angle between the camera optical axis and the turntable axis was computed to be 59.4°, and the distance between the lens center to the turntable center was computed to be 27.3cm. The accuracy of this computation depends on the accuracy of the camera parameters and the image used (evaluation of the camera imaging system is out of the scope of this paper).

4 Concluding Remarks

With the motivation of obtaining accurate 3-D measurement of circular objects, we presented a new method of automatically detecting ellipses in images: we detect an osculating circle using a Hough transform, iteratively improve the fit, remove outlier pixels by LMedS, and search for separate arcs. We limited the Hough space to be one and two dimensions for efficiency and introduced special weighting schemes for enhancing accuracy. Using real images, we demonstrated that our method can detect partially occluded circular objects within a reasonable time. We also showed an example of turntable calibration for 3-D object shape reconstruction.

Acknowledgments. The authors thank Mitsuo Okabe of Fujitsu Cadtech, Ltd., Japan, for real image experiments. This work was supported in part by the Ministry of Education, Culture, Sports, Science and Technology, Japan, under a Grant in Aid for Scientific Research C(2) (No. 13680432), the Support Center for Advanced Telecommunications Technology Research, and Kayamori Foundation of Informational Science Advancement.

References

[1] Y. Asayama and M. Shiono, A slanted ellipse detection by a circle detecting Hough transform using a par of arcs, *Proc. IAPR Workshop on Machine Vision Applications*, November 1998, Makuhari, Chiba, Japan, pp. 494–497.

[2] J. W. Bruce and P. J. Giblin, *Curves and Singularities*, Cambridge University Press, Cambridge, U.K., 1984.

[3] Y. C. Cheng and S. C. Lee, A new method for quadratic curve detection using K-RANSAC with acceleration techniques, *Patt. Recog.*, **28**-5 (1995), 663–682.

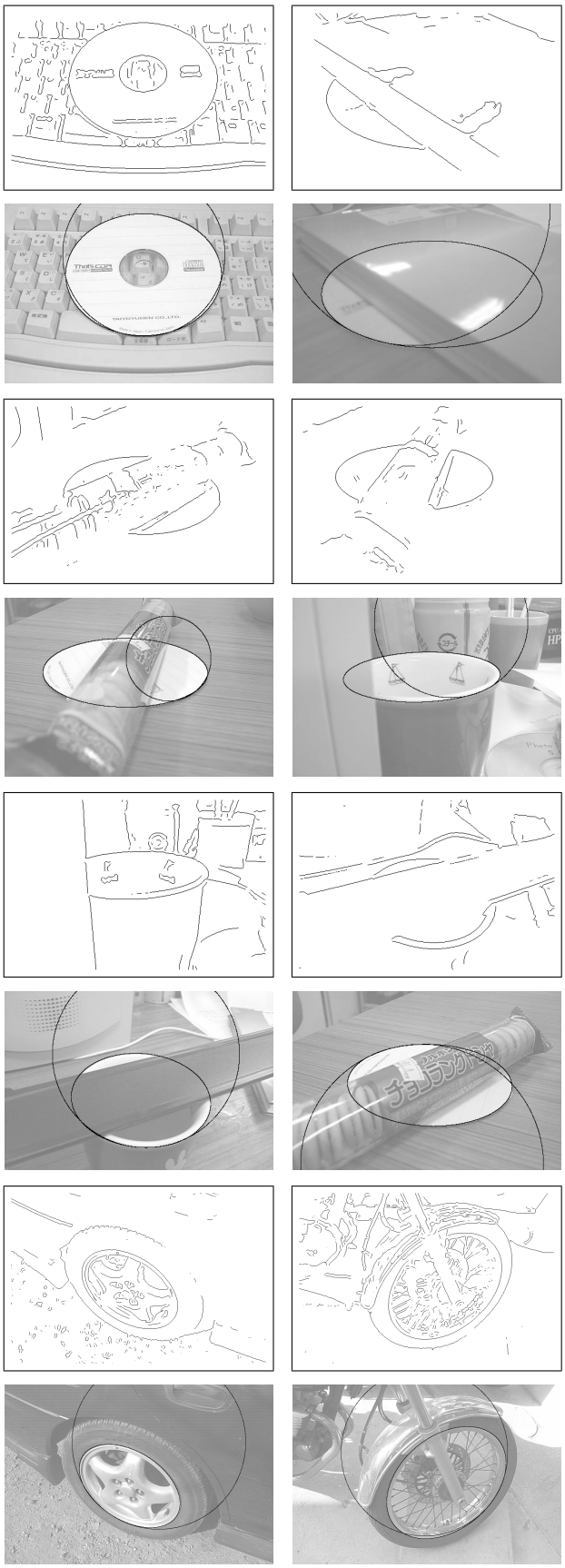
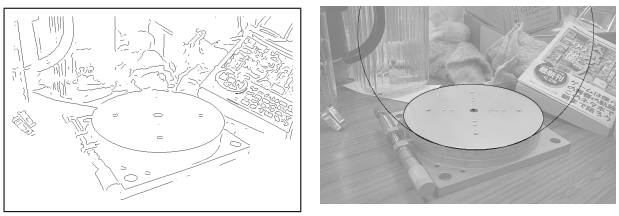


Figure 8 Examples of ellipse fitting: the edge images (upper rows); the initial circles and the fitted ellipses superimposed onto the original images (lower rows).



(a) (b)

Figure 9 Calibration of a turntable. (a) The edge image used. (b) The initial circle and the fitted ellipse superimposed onto the original image.

[4] W. Chojnacki, M. J. Brooks and A. van den Hengel, Rationalising the renormalisation method of Kanatani, *J. Math. Imaging Vision*, **14**-1 (2001), 21–38.

[5] W. Chojnacki, M. J. Brooks, A. van den Hengel and D. Gawley, On the fitting of surfaces to data with covariances, *IEEE Trans. Patt. Anal. Mach. Intell.*, **22**-11 (2000), 1294–1303.

[6] D. B. Cooper and N. Yalabik, On the computational cost of approximating and recognizing noise-perturbed straight lines and quadratic arcs in the plane, *IEEE Trans. Comp.*, **25**-10 (1976), 1020–1032.

[7] D. Ioannou, W. Huda and A. F. Laine, Circle recognition through a 2D Hough Transform and radius histogramming, *Image Vision Comput.*, **17**-1 (1999), 15–26.

[8] K. Kanatani, *Geometric Computation for Machine Vision*, Oxford University Press, Oxford, 1993.

[9] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*, Elsevier, Amsterdam, 1996.

[10] N. Guil and E. L. Zapata, Lower order circle and ellipse Hough transform, *Patt. Recog.*, **30**-10 (1997), 1729–1744.

[11] C.-T. Ho and L.-H. Chen, A fast ellipse/circle detector using geometric symmetry, *Patt. Recog.*, **28**-1 (1995), 117–124.

[12] C. L. Huang, Elliptical feature extraction via an improved Hough transform, *Patt. Recog. Lett.*, **10**-2 (1989), 93–100.

[13] Y. Leedan and P. Meer, Heteroscedastic regression in computer vision: Problems with bilinear constraint, *Int. J. Comput. Vision*, **37**-2 (2000), 127–150.

[14] B. Matei and P. Meer, Reduction of bias in maximum likelihood ellipse fitting, *Proc. 15th Int. Conf. Patt. Recog.*, September 2000, Barcelona, Spain, Vol.3, pp. 801–806.

[15] D. Pao, H. F. Li and R. Jayakumar, A decomposable parameter space for the detection of ellipses, *Patt. Recog.*, **14**-12 (1993), 951–958.

[16] P. L. Rosin and G. A. W. West, Nonparametric segmentation of curves into various representations, *IEEE Trans. Patt. Anal. Mach. Intell.*, **17**-12 (1995), 1140–1153.

[17] G. Roth and M. D. Levine, Extracting geometric primitives, *CVGIP: Image Understand.*, **58**-1 (1993), 1–22.

[18] G. Roth and M. D. Levine, Geometric primitive extraction using a genetic algorithm, *IEEE Trans. Patt. Anal. Mach. Intell.*, **16**-9 (1994), 901–905.

[19] C. A. Rothwell, A. Zisserman, C. I. Marinos, D. A. Forsyth and J. L. Mundy, Relative motion and pose from arbitrary plane curves, *Image Vision Comput.*, **10**-4 (1992), 250–262.

[20] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*, Wiley, New York, 1987.

[21] S. Tsuji and F. Matsumoto, Detection of ellipses by a modified Hough transform, *IEEE Trans. Comp.*, **27**-8 (1978), 777–781.

[22] W.-Y. Wu and M.-J. J. Wang, Elliptical object detection by using its geometric properties, *Patt. Recog.*, **26**-10 (1993), 1449–1500.

[23] H. K. Yuen, J. Illingworth and J. Kittler, Detecting partially occluded ellipses using the Hough transform, *Image Vision Comput.*, **7**-1 (1989), 31–37.

[24] J. H. Yoo and I. K. Seth, An ellipse detection method from the polar and pole definition of conics, *Patt. Recog.*, **26**-2 (1993), 307–315.