A Buyer's Guide to Conic Fitting*

Andrew W. Fitzgibbon
Robert B. Fisher
Department of Artificial Intelligence, Edinburgh University
5 Forrest Hill, Edinburgh EH1 2QL

Abstract

In this paper we evaluate several methods of fitting data to conic sections. Conic fitting is a commonly required task in machine vision, but many algorithms perform badly on incomplete or noisy data. We evaluate several algorithms under various noise and degeneracy conditions, identify the key parameters which affect sensitivity, and present the results of comparative experiments which emphasize the algorithms' behaviours under common examples of degenerate data. In addition, complexity analyses in terms of flop counts are provided in order to further inform the choice of algorithm for a specific application.

1 Introduction

Vision applications often require the extraction of conic sections from image data. Common examples are the calculation of geometric invariants [2] and estimation of the centers and radii of circles for industrial inspection.

Many textbooks [5, 10] provide discussions and algorithms for least-squares approximation of conics, but these often include only the simple and fast algebraic distance algorithm (algorithm LIN below). This algorithm fares poorly on many real data sets due to its inherent statistical bias, particularly when the image curves are partially occluded. A number of authors [1, 6, 8, 9, 12, 13] have proposed alternative algorithms and while these are usually compared by the authors with LIN, there have been, to our knowledge, no comparative studies of the relative accuracy and efficiency of these alternatives.

This paper makes two important contributions to this area of computer vision research:

- Identification of the main conditions under which the algorithms fail. It is common for comparative evaluations to concentrate on noise sensitivity, but in the case of conic fitting the important parameter is the amount of occlusion.
- Presentation of the algorithm complexities in terms of *flop counts* allows evaluation of the tradeoff between accuracy and speed of execution without reference to the specifics of an implementation and environment.

*This work was funded by UK EPSRC Grant GR/H/86905.

The four methods compared differ primarily in terms of the error measure that they minimize and then in terms of the techniques that are used to minimize this measure. In particular, the error measure determines the eventual accuracy of the methods and generally dictates the choice of optimization algorithm and hence their execution speed.

2 Problem Statement

The problem which the algorithms presented in this paper solve may be stated as follows. Given:

- A set of 2D *data points* $P = \{\mathbf{x}_i\}_{i=1}^n$, where $\mathbf{x}_i = (x_i, y_i)$;
- A *family of curves* $C(\mathbf{a})$ parameterized by the vector $\mathbf{a}$;
- A *distance metric* $\delta(C(\mathbf{a}), \mathbf{x})$, which measures the distance from a point $\mathbf{x}$ to the curve $C(\mathbf{a})$;

find the value $\mathbf{a}_{\min}$ for which the error function $\epsilon^2(\mathbf{a}) = \sum_{i=1}^n \delta(C(\mathbf{a}), \mathbf{x}_i)$ attains its global minimum. The curve $C(\mathbf{a})$ is then the curve that *best fits* the data.

In this paper, the curve families considered are represented in the *implicit form* $C(\mathbf{a}) = \{\mathbf{x} \mid F(\mathbf{a}; \mathbf{x}) = 0\}$. The two families that we examine are general conic sections, with $F(\mathbf{a}; \mathbf{x}) = A_{xx}x^2 + A_{xy}xy + A_{yy}y^2 + A_x x + A_y y + A_0$; and circles, for which $F_C(\mathbf{a}; \mathbf{x}) = A_r(x^2 + y^2) + A_x x + A_y y + A_0$.

Finally we note that these forms may be written in a way that separates the parameters A_α from the terms in $\mathbf{x}$ using the dot product

$$\begin{aligned} F(\mathbf{a}; \mathbf{x}_i) &= [x_i^2 \; x_i y_i \; y_i^2 \; x_i \; y_i \; 1] \cdot [A_{xx} \; A_{xy} \; A_{yy} \; A_x \; A_y \; A_0] \\ &= \chi_i \cdot \mathbf{a} \end{aligned}$$

3 The Algorithms

3.1 Algorithm LIN: Algebraic Distance

The algebraic distance algorithm minimizes the objective function

$$\epsilon^2(\mathbf{a}) = \sum_{i=1}^{n} F(\mathbf{a}, \mathbf{x}_i)^2 = ||D\mathbf{a}||^2$$

subject to the constraint that $||\mathbf{a}||^2 = 1$. The *design matrix* D is the $n \times 6$ matrix with rows χ_i. The constrained objective function $E = ||D\mathbf{a}||^2 - \lambda(||\mathbf{a}|| - 1) = \mathbf{a}^T D^T D\mathbf{a} - \lambda(\mathbf{a}^T\mathbf{a} - 1)$ is minimized analytically to form an eigenvector problem [4]:

$$\nabla_{\mathbf{a}} E = 0 \quad \Longleftrightarrow \quad 2D^T D\mathbf{a} - 2\lambda\mathbf{a} = 0$$

where λ is a Lagrange multiplier. The minimizer $\mathbf{a}_{\min}$ is then the eigenvector of $D^T D$ corresponding to the smallest eigenvalue.

The algorithm requires $12n$ multiplications and $14n$ adds (or approximately $26n$ *flops*[1] for the construction of the 15 unique elements of $D^T D$. Evaluation of the eigensystem by Hessenberg reduction and QR generally took about 20 iterations (1700 flops) in our experiments, giving a total complexity of about $26n + 1700$ flops.

3.2 Algorithm BOOK: Algebraic distance with quadratic constraint

Bookstein's algorithm [1] attempts to reduce the bias of LIN by imposing a different constraint on the parameter vector $\mathbf{a}$. He derives the constraint $2A_{xx}^2 + A_{xy}^2 + 2A_{yy}^2 = 1$ and shows how this leads to the system

$$D^T D\mathbf{a} = \lambda \; \Delta\mathbf{a}$$

where $\Delta = \mathrm{diag}(2, 1, 2, 0, 0, 0)$. This is a rank-deficient generalized eigensystem, which Bookstein solves by block-decomposition.

BOOK requires the $26n$ flops of LIN to form the $D^T D$ matrix. The matrix inversion and eigensystem solution's mean flop count was 1165, yielding a total complexity of $26n + 1165$ flops.

3.3 Algorithm AMS: "Approximate Mean Square" Distance

The "approximate mean square distance" metric, introduced by Taubin[13], minimizes the unusual objective function

$$\epsilon^2(\mathbf{a}) = \frac{\sum_{i=1}^n F(\mathbf{a}, \mathbf{x}_i)^2}{\sum_{i=1}^n ||\nabla_{\mathbf{x}} F(\mathbf{a}, \mathbf{x}_i)||^2} = \frac{||D\mathbf{a}||^2}{||D_x\mathbf{a}||^2 + ||D_y\mathbf{a}||^2}$$

where the matrices D_x and D_y are the partial derivatives of D with respect to x and y. Restating the problem as the minimization of $||D\mathbf{a}||^2$ subject to $||D_x a||^2 + ||D_y a||^2 = 1$, the minimizer $\mathbf{a}_{\min}$ is then the eigenvector of the generalized eigensystem [4]

$$D^T D a = \lambda(D_x^T D_x + D_y^T D_y)a$$

corresponding to the largest eigenvalue.

AMS requires the $26n$ flops of LIN to form the $D^T D$ matrix, but negligible additional time to form $D_x^T D_x + D_y^T D_y$ from the elements of D. The generalized eigensystem routine's mean flop count was 9700, yielding a total complexity of $26n + 9700$ flops.

[1] MATLAB [7] defines addition and multiplication to each contribute one flop, while Golub and van Loan [4] consider one flop to consist of a multiply-accumulate. The MATLAB definition corresponds more closely to the computer used to perform the experiments, on which both multiplication and addition require one clock cycle.

3.4 Algorithm GEOM: Geometric Distance

The geometric distance metric measures the orthogonal distance from the point $\mathbf{x}$ to the conic section. This metric, proposed by Nakagawa and Rosenfeld [8], is approximately unbiased if the errors in the data points are distributed normally to the curve. The distance is evaluated at a point $\mathbf{x}$ by solving the simultaneous equations

$$\begin{aligned} \mathbf{p} + \lambda \nabla F(\mathbf{a}; \mathbf{p}) &= \mathbf{x} \\ F(\mathbf{a}; \mathbf{p}) &= 0 \end{aligned}$$

for $\mathbf{p}$ and defining $\delta(C(\mathbf{a}), \mathbf{x}) = ||\mathbf{x} - \mathbf{p}||^2$. These equations involve the solution of a quartic equation, and while closed-form solutions exist, numerical instability can result from the application of the analytic formula [10]. In our implementation we extract roots from the eigensystem of the companion matrix [4]. This in turn means that analytic derivatives of δ, and consequently $\nabla_{\mathbf{a}} \epsilon^2(\mathbf{a})$, are difficult to calculate.

Each evaluation of $\epsilon^2(\mathbf{a})$ involves the solution of n such quartics, averaging $1300n$ flops per iteration for the eigenvalue calculation. The number of iterations depends on the minimization algorithm chosen, but it is clear that even with only 10 iterations, GEOM is 3 to 4 orders of magnitude slower than the previous two algorithms, and therefore was not extensively tested in our experiments.

3.5 Algorithm BIAS: "Statistical" Distance

Another approach, proposed by Kanatani [6] and Porrill [9], improves the LIN algorithm by explicitly calculating its inherent statistical bias and subtracting the bias from the result of minimization. Because the calculation of the bias depends on knowing both the true minimum and the noise level, the process is iterated until the predicted bias results in a noise-level correction of zero. Kanatani calls this metric the "statistical distance", and argues that its bias sensitivity is in fact superior to the geometric distance in the case where the errors on the data points are spherically distributed. Due to pressure of space this paper discusses only Kanatani's bias correction algorithm. A description of algorithm itself would be too long to include here, due to its dependence on tensor arithmetic. Note however that the published noise-level update formula [6, eq 21] should be replaced by

$$c \quad \leftarrow \quad c + \frac{\lambda_m}{\mathrm{tr}(Q)\mathrm{tr}(MQ) + 2\mathrm{tr}(MQ^2)}$$

Complexity of the algorithm is of the order of $50n + 1000$ flops per iteration, with our test runs taking an average of 10 iterations.

3.6 Algorithm B2AC: Algebraic distance with quadratic constraint

The B2AC algorithm is ellipse-specific, imposing the constraint that $A_{xy}^2 - 4A_{xx}A_{yy} = 1$. This cleverly converts the inequality $A_{xy}^2 - 4A_{xx}A_{yy} > 0$ into an equality by

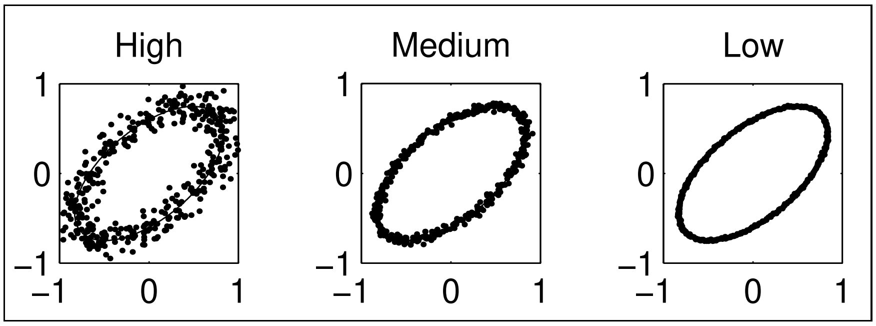

Figure 1: Visual depiction of the three noise levels used in the later experiments. The High, Medium and Low levels correspond to standard deviations of 4, 1 and $\frac{1}{4}$ pixels respectively.

incorporating the normalisation factor. The corresponding generalized eigensystem is not nonnegative definite, and is solved by the scheme of Gander [3]. B2AC requires the $26n$ flops of LIN to form the $D^T D$ matrix. The matrix inversion and eigensystem solution's mean flop count was 5182.

4 Experiments

All experiments were conducted using the MATLAB system [7]. Eigensystems are solved using the underlying EISPACK routines, while the derivative-free minimization needed for the GEOM algorithm used the Nelder-Mead simplex algorithm. Also, as the execution-speed characteristics of interpreted MATLAB programs are qualitatively different to those of equivalent programs in a compiled language, we give no timing statistics other than the flop counts in the previous section.

Noise model

The noise model applied in all experiments is isotropic zero mean normally distributed. Other models may also be considered, particularly a nonisotropic model where the variance is perpendicular to the curve. However, as the experiments illustrate, degree of occlusion of the curve rather than noise is the parameter to which the algorithms are more sensitive.

Additionally, the noise model does not include any outlier component. This is because none of the described algorithms are statistically robust. The algorithms may be made robust in the usual ways [14], in which case an outlier component would be added. Qualitatively, the most important change that this might make to the results presented here would be that timing considerations could prove less unfavourable to the iterative algorithms than in the outlier-free case.

Experiments other than the first are performed with three 'typical' noise levels, depicted visually in Figure 1. The high, medium and low noise levels correspond roughly to standard deviations of 2, 1 and $\frac{1}{4}$ pixels on an ellipse with a major diameter of 20 pixels.

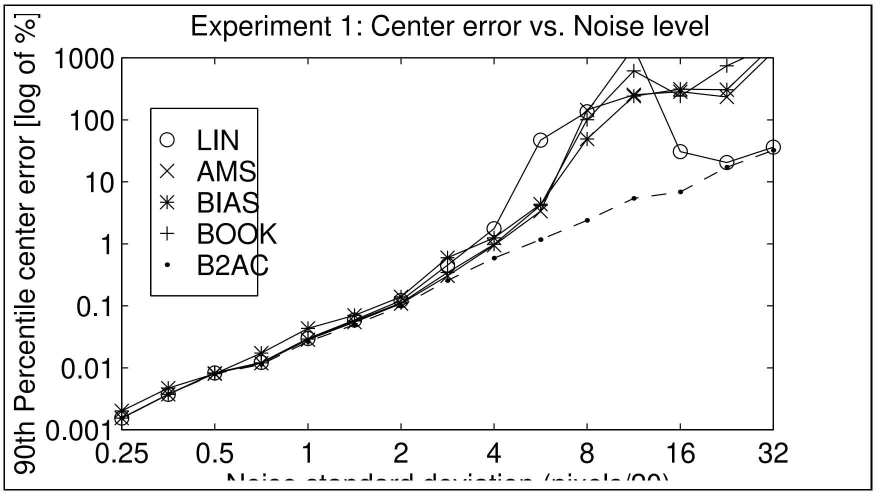

Figure 2: Results of experiment 1. Only B2AC degrades smoothly wrt noise on unoccluded data.

Conic representation

In the following for convenience in defining the experiments, central conic parameters are expressed as 5-vectors $[c_x, c_y, R_x, R_y, \theta]$, where (c_x, c_y) is the conic center, R_x and R_y are the X and Y radii respectively, and θ is the counterclockwise angle in degrees between the ellipse and the positive X axis.

All of the experiments use center position as their primary error measure. We have performed the same experiments using radius error as the measure and found no qualitative difference in the results.

4.1 Experiment 1: Noise

In this experiment, we are interested in characterizing the behaviour of the algorithms with complete data and with respect to noise. Experimental procedure was as follows:

1. The ellipse $[0, 0, 1, \frac{1}{2}, 40]$ was sampled at 100 points uniformly distributed around the circumference.

2. Noise sigma was logarithmically varied between 2^{-3} and 2^3 pixels.

3. The sampled ellipse was corrupted with noise as described above for 100 runs and the distance between the true ellipse center and the center of the conic returned by the fitting algorithm was recorded. Returned hyperbolae were included.

Figure 2 shows the 90^{th} percentile error in the centers as a function of noise level. At low noise levels ($\sigma < 0.5$), all algorithms can be seen to perform similarly, while at high levels only the B2AC algorithm degrades gracefully.

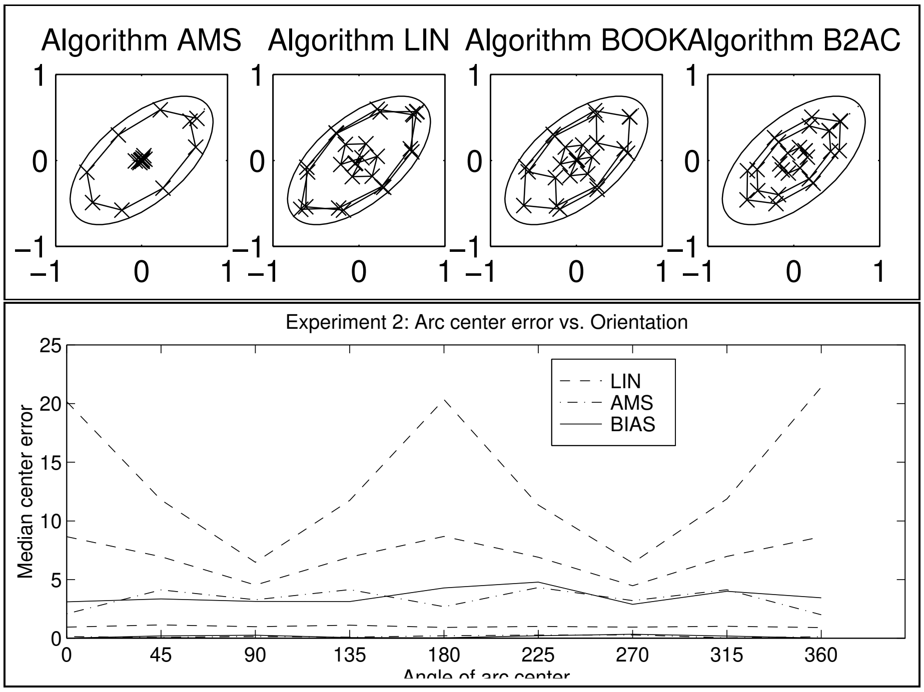

Figure 3: Results of experiment 2. Highest errors in center position occur at curvature maxima for LIN, between maxima and minima for AMS and BIAS.

4.2 Experiment 2: Orientation

In this experiment, we investigate how the errors in determining the center of a 120° elliptical arc vary as the portion of the ellipse from which the arc is sampled rotates about the ellipse. This is so that in Experiment 3 we may ensure that the subtended angle measurements are taken at the most pessimistic location about the ellipse. Experimental procedure was as follows:

1. The counterclockwise orientation of the center of the arc was varied from 0 to 360° in steps of 45°.
2. The ellipse $[0, 0, 1, \frac{1}{2}, 40]$ was sampled at 100 points uniformly distributed along the 120° arc.
3. The sampled arc was corrupted with the three 'standard' noise levels as described above.
4. The distance between the true ellipse center and the center returned by the fitting algorithm was recorded.

Figure 3 shows the results in two ways. The top three figures illustrate visually the located circle positions for algorithms AMS, LIN, BOOK and B2AC, while the bottom graph shows the error in median center position as a function of the arc

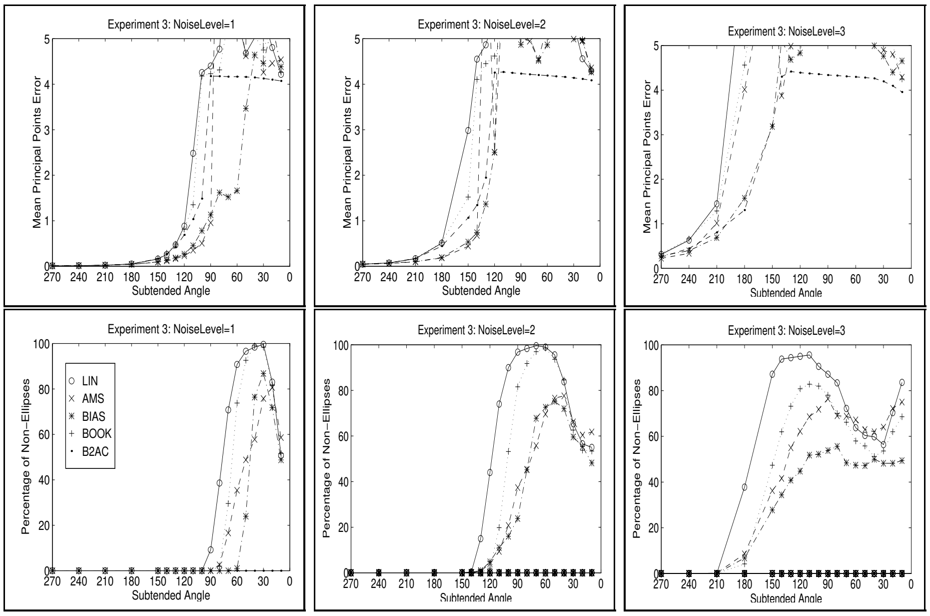

Figure 4: Results of experiment 3. Results are for low, medium and high noise from left to right. The upper curves show the average sum of the errors in the principal points, while the lower ones show the proportion of non-ellipses produced by the conic algorithms.

orientation. BIAS and AMS show their greatest errors at the 45° points, while maximum errors for LIN occur when the arc is sampled from the high curvature sections at 0° and 180°.

4.3 Experiment 3: Occlusion

The third experiment is designed to locate the breakdown point of each of the algorithms when the ellipse is progressively occluded. We measure the errors in center position and radius estimates for several arcs of decreasing subtended angle. Experimental procedure was as follows:

1. The angle subtended by the elliptical arc was varied from 360° down to 0 in steps of 10°.
2. The ellipse $[0, 0, 1, \frac{1}{2}, 40°]$ was sampled at 100 points uniformly distributed along the arc.
3. The sampled arc was corrupted with the three 'standard' noise levels as described above.
4. Over 500 runs, the noisy arcs were submitted to each fitting algorithm. The distances of the fitted principal points to their correspondents on the true ellipse were calculated. The mean sum distance where the algorithms returned

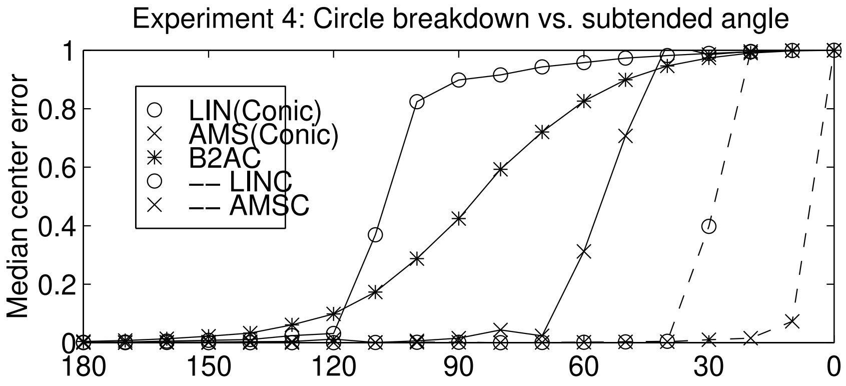

Figure 5: Results of experiment 4. The specialized circle fitters LINC and AMSC have better breakdown characteristics than the corresponding general conic algorithms.

ellipses was calculated, as was the percentage of runs in which non-ellipse conics were returned.

Figure 4 shows the plots of principal point error and percentage of returned non-ellipses as a function of decreasing subtended angle.

4.4 Experiment 4: Circles

In the final experiment, we consider the breakdown performance when the general conic fitting algorithms are made specific to a particular task; in this case, circle fitting. Procedure is similar to Experiment 3, but we add two new algorithms, LINC and AMSC, which are specializations of LIN and AMS respectively. Figure 5 shows the breakdown curves for the low-noise case. As expected, the specialized fitters LINC and AMSC break down considerably later than the general conic algorithms.

5 Discussion

This paper has discussed the problem of fitting conic sections to ellipse data. The experiments illustrate that the key parameter affecting the algorithms' accuracy is the amount of occlusion present and the qualitative noise level. With complete data, all algorithms exhibit a similar degradation in the presence of increasing noise.

As the data become progressively incomplete, a breakdown point is reached beyond which the algorithms fail catastrophically. This breakdown point is superior with the B2AC and BIAS algorithms, and, in the special case of a circle, with the circle-specific algorithms. Under high noise, BIAS has superior accuracy but produces non-ellipses up to 60% of the time on highly occluded ellipses.

Algorithm complexities are, in increasing order: BOOK, LIN, B2AC, AMS, BIAS, GEOM.

Current and future work involves implementation and testing of the ellipse-specific algorithms of [11, 12], along with Porrill's alternative BIAS algorithm, and examining some alternative error metrics such as the conic invariants.

References

[1] F. Bookstein. Fitting conic sections to scattered data. *Computer Graphics and Image Processing*, 9:56–71, 1979.

[2] D. Forsyth, J. L. Mundy, A. Zisserman, et al. Invariant descriptors for 3D object recognition and pose. *IEEE T-PAMI*, 13(10):971–991, 1991.

[3] W. Gander. Least squares with a quadratic constraint. *Numerische Mathematik*, 36:291–307, 1981.

[4] G. H. Golub and C. F. van Loan. *Matrix Computations*. Johns Hopkins, 2$^{\text{nd}}$ edition, 1989.

[5] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*, volume 1. Addison-Wesley, 1993.

[6] K. Kanatani. Statistical bias of conic fitting and renormalization. *IEEE T-PAMI*, 16(3):320–326, 1994.

[7] The MathWorks, Inc., Natick MA, USA. MATLAB *Reference Guide*, 1992.

[8] Y. Nakagawa and A. Rosenfeld. A note on polygonal and elliptical approximation of mechanical parts. *Pattern Recognition*, 11:133–142, 1979.

[9] J. Porrill. Fitting ellipses and predicting confidence envelopes using a bias corrected kalman filter. *Image and Vision Computing*, 8(1):37–41, February 1990.

[10] W. H. Press et al. *Numerical Recipes*. Cambridge University Press, 2nd edition, 1992.

[11] P. L. Rosin. Ellipse fitting by accumulating five-point fits. *Pattern Recognition Letters*, 14:661–669, 1993.

[12] P. D. Sampson. Fitting conic sections to "very scattered" data. *Computer Graphics and Image Processing*, 18:97–108, 1992.

[13] G. Taubin. Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE T-PAMI*, 13(11):1115–1138, November 1991.

[14] P. Torr. Robust vision. In *Proceedings, BMVC*, 1993.