ELSEVIER

Image and Vision Computing 15 (1997) 59–76

Image and Vision Computing

Parameter estimation techniques: a tutorial with application to conic fitting

Zhengyou Zhang*

INRIA, 2004 route des Lucioles, BP 93, F-06902 Sophia-Antipolis Cedex, France

Received 20 November 1995; revised 18 April 1996; accepted 22 April 1996

Abstract

Almost all problems in computer vision are related in one form or another to the problem of estimating parameters from noisy data. In this tutorial, we present what is probably the most commonly used techniques for parameter estimation. These include linear least-squares (pseudo-inverse and eigen analysis); orthogonal least-squares; gradient-weighted least-squares; bias-corrected renormalization; Kalman filtering; and robust techniques (clustering, regression diagnostics, M-estimators, least median of squares). Particular attention has been devoted to discussions about the choice of appropriate minimization criteria and the robustness of the different techniques. Their application to conic fitting is described.

Keywords: Parameter estimation; Least-squares; Bias correction; Kalman filtering; Robust regression

1. Introduction

Almost all problems in computer vision are related in one form or another to the problem of estimating parameters from noisy data. A few examples are line fitting, camera calibration, image matching, surface reconstruction, pose determination, and motion analysis [1,2]. A parameter estimation problem is usually formulated as an optimization one. Because of different optimization criteria and because of several possible parameterizations, a given problem can be solved in many ways. The purpose of this paper is to show the importance of choosing an appropriate criterion. This will influence the accuracy of the estimated parameters, the efficiency of computation, the robustness to predictable or unpredictable errors. Conic fitting is used to illustrate these aspects because:

- it is one of the simplest problems in computer vision on one hand;
- it is, on the other hand, a relatively difficult problem because of its nonlinear nature.

Needless to say, the importance of conics in our daily life and in industry.

2. A glance over parameter estimation in general

Parameter estimation is a discipline that provides tools for the efficient use of data for aiding in mathematically modeling of phenomena and the estimation of constants appearing in these models [3]. It can thus be visualized as a study of inverse problems. Much of parameter estimation can be related to four optimization problems:

- criterion: the choice of the best function to optimize (minimize or maximize);
- estimation: the choice of the best method to optimize the chosen function;
- design: optimal implementation of the chosen method to obtain the best parameter estimates;
- modeling: the determination of the mathematical model which best describes the system from which data are measured, including a model of the error processes.

In this paper we are mainly concerned with the first three problems, and we assume the model is known (a conic in the examples).

Let $\mathbf{p}$ be the (state/parameter) vector containing the parameters to be estimated. The dimension of $\mathbf{p}$, say m, is the number of parameters to be estimated. Let $\mathbf{z}$ be the (measurement) vector which is the output of the system to be modeled. The system in noise-free case is described by a vector function $\mathbf{f}$ which relates $\mathbf{z}$ to $\mathbf{p}$ such that

$$\mathbf{f}(\mathbf{p}, \mathbf{z}) = 0.$$

* Email: zzhang@sophia.inria.fr

0262-8856/97/$15.00 © 1997 Elsevier Science B.V. All rights reserved
PII S0262-8856(96)01112-2

In practice, observed measurements $\mathbf{y}$ are only available for the system output $\mathbf{z}$ corrupted with noise ϵ, i.e.

$$\mathbf{y} = \mathbf{z} + \epsilon.$$

We usually make a number of measurements for the system, say $\{\mathbf{y}_i\}$ $(i = 1, \ldots, n)$, and we want to estimate $\mathbf{p}$ using $\{\mathbf{y}_i\}$. As the data are noisy, the equation $\mathbf{f}(\mathbf{p}, \mathbf{y}_i) = 0$ is not valid any more. In this case, we write down a function

$$\mathscr{F}(\mathbf{p}, \mathbf{y}_1, \ldots, \mathbf{y}_n)$$

which is to be optimized (without loss of generality, we will minimize the function). This function is usually called the *cost function* or the *objective function*.

If there are no constraints on $\mathbf{p}$ and the function $\mathscr{F}$ has first and second partial derivatives everywhere, necessary conditions for a minimum are

$$\frac{\partial \mathscr{F}}{\partial \mathbf{p}} = 0$$

and

$$\frac{\partial^2 \mathscr{F}}{\partial \mathbf{p}^2} > 0.$$

By the last, we mean that the $m \times m$-matrix is positive definite.

3. Conic fitting problem

The problem is to fit a conic section to a set of n points $\{\mathbf{x}_i\} = \{(x_i, y_i)\}$ $(i = 1, \ldots, n)$. A conic can be described by the following equation:

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

where A, B and C are not simultaneously zero. In practice, we encounter ellipses, where we must impose the constraint $B^2 - AC < 0$. However, this constraint is usually ignored during the fitting because

- the constraint is usually satisfied if data are not all situated in a *flat* section, even if the constraint is not imposed during fitting;
- the computation will be very expensive if this constraint is considered.

As the data are noisy, it is unlikely to find a set of parameters (A, B, C, D, E, F) (except for the trivial solution $A = B = C = D = E = F = 0$) such that $Q(x_i, y_i) = 0$, $\forall i$. Instead, we will try to estimate them by minimizing some objective function.

4. Least-squares fitting based on algebraic distances

As said before, for noisy data, the system equation, $Q(x, y) = 0$ in the case of conic fitting, can hardly hold true. A common practice is to directly minimize the *algebraic distance* $Q(x_i, y_i)$, i.e. to minimize the following function:

$$\mathscr{F} = \sum_{i=1}^{n} Q^2(x_i, y_i).$$

Note that there is no justification for using algebraic distances apart from ease of implementation.

Clearly, there exists a trivial solution $A = B = C = D = E = F = 0$. To avoid it, we should normalize $Q(x, y)$. There are many different normalizations proposed in the literature. Here we describe three of them. Please note that the comparison of different normalizations is not the purpose of this section. Our purpose is to present different techniques to solve linear least-squares problems.

4.1. Normalization with $A + C = 1$

Since the trace $A + C$ can never be zero for an ellipse, the arbitrary scale factor in the coefficients of the conic equation can be removed by the normalization $A + C = 1$. This normalization has been used by many researchers [4,5]. All ellipse can then be described by a 5-vector

$$\mathbf{p} = [A, B, D, E, F]^{\mathrm{T}},$$

and the system equation $Q(x_i, y_i) = 0$ becomes:

$$\mathbf{f}_i \equiv \mathbf{a}_i^{\mathrm{T}}\mathbf{p} - b_i = 0, \quad (2)$$

where

$$\mathbf{a}_i = [x_i^2 - y_i^2, 2x_i y_i, 2x_i, 2y_i, 1]^{\mathrm{T}}$$

$$b_i = -y_i^2.$$

Given n points, we have the following vector equation:

$$\mathbf{A}\mathbf{p} = \mathbf{b},$$

where

$$\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_n, \ldots, \mathbf{a}_n]^{\mathrm{T}}$$

$$\mathbf{b} = [b_1, b_2, \ldots, b_n]^{\mathrm{T}}.$$

The function to minimize becomes

$$\mathscr{F}(\mathbf{p}) = (\mathbf{A}\mathbf{p} - \mathbf{b})^{\mathrm{T}}(\mathbf{A}\mathbf{p} - \mathbf{b}).$$

Obtaining its partial derivative with respect to $\mathbf{p}$ and setting it to zero yield:

$$2\mathbf{A}^{\mathrm{T}}(\mathbf{A}\mathbf{p} - \mathbf{b}) = \mathbf{0}.$$

The solution is readily given by

$$\mathbf{p} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}. \quad (3)$$

This method is known as the *pseudo inverse technique*.

4.2. Normalization with $A^2 + B^2 + C^2 + D^2 + E^2 + F^2 = 1$

Let $\mathbf{p} = [A, B, C, D, E, F]^T$. As $\|\mathbf{p}\|^2$, i.e. the sum of squared coefficients, can never be zero for a conic, we can set $\|\mathbf{p}\| = 1$ to remove the arbitrary scale factor in the conic equation. The system equation $Q(x_i, y_i) = 0$ becomes

$$\mathbf{a}_i^T \mathbf{p} = 0 \quad \text{with } \|\mathbf{p}\| = 1,$$

where $\mathbf{a}_i = [x_i^2, 2x_i y_i, y_i^2, 2x_i, 2y_i, 1]^T$.

Given n points, we have the following vector equation:

$$\mathbf{A}\mathbf{p} = \mathbf{0} \quad \text{with } \|\mathbf{p}\| = 1.$$

where $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]^T$. The function to minimize becomes:

$$\mathscr{F}(\mathbf{p}) = (\mathbf{A}\mathbf{p})^T(\mathbf{A}\mathbf{p}) \equiv \mathbf{p}^T \mathbf{B}\mathbf{p} \text{ subject to } \|\mathbf{p}\| = 1, \tag{4}$$

where $\mathbf{B} = \mathbf{A}^T\mathbf{A}$ is a symmetric matrix. The solution is the eigenvector of $\mathbf{B}$ corresponding to the smallest eigenvalue (see below).

Indeed, any $m \times m$ symmetric matrix $\mathbf{B}$ ($m = 6$ in our case) can be decomposed as

$$\mathbf{B} = \mathbf{U}\mathbf{E}\mathbf{U}^T, \tag{5}$$

with

$$\mathbf{E} = \text{diag}(v_1, v_2, \ldots, v_m) \quad \text{and} \quad \mathbf{U} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m],$$

where v_i is the i-th eigenvalue, and $\mathbf{e}_i$ is the corresponding eigenvector. Without loss of generality, we assume $v_1 \le v_2 \le \cdots \le v_m$. The original problem (4) can now be related as:

Find $q_1, q_2, \ldots, q_m$ such that $\mathbf{p}^T\mathbf{B}\mathbf{p}$ is minimized with $\mathbf{p} = q_1\mathbf{e}_1 + q_2\mathbf{e}_2 + \cdots + q_m\mathbf{e}_m$ subject to $q_1^2 + q_2^2 + \cdots + q_m^2 = 1$.

After some simple algebra, we have

$$\mathbf{p}^T\mathbf{B}\mathbf{p} = q_1^2 v_1 + q_2^2 v_2 + \cdots + q_m^2 v_m.$$

The problem now becomes to minimize the following unconstrained function:

$$\mathscr{J} = q_1^2 v_1 + q_2^2 v_2 + \cdots + q_m^2 v_m + \lambda(q_1^2 + q_2^2 + \cdots + q_m^2 - 1),$$

where λ is the Lagrange multiplier. Setting the derivatives of $\mathscr{J}$ with respect to q_1 through q_m and λ yields:

$$2q_1 v_1 + 2q_1\lambda = 0$$

$$2q_2 v_2 + 2q_2\lambda = 0$$

$$\ldots\ldots$$

$$2q_m v_m + 2q_m\lambda = 0$$

$$q_1^2 + q_2^2 + \cdots + q_m^2 - 1 = 0.$$

There exists m solutions. The i-th solution is given by

$$q_i = 1, \quad q_j = 0 \text{ for } j = 1, \ldots, m, \text{ except } i, \quad \lambda = -v_i.$$

The value of $\mathscr{J}$ corresponding to the i-th solution is

$$\mathscr{J}_i = v_i.$$

Since $v_1 \le v_2 \le \cdots \le v_m$, the first solution is the one we need (the least-squares solution), i.e.

$$q_1 = 1, \quad q_j = 0 \text{ for } j = 2, \ldots, m.$$

Thus the solution to the original problem (4) is the eigenvector of $\mathbf{B}$ corresponding to the smallest eigenvalue.

4.3. Normalization with $F = 1$

Another commonly used normalization is to set $F = 1$. If we use the same notation as in the last subsection, the problem becomes to minimize the following function:

$$\mathscr{F}(\mathbf{p}) = (\mathbf{A}\mathbf{p})^T(\mathbf{A}\mathbf{p}) = \mathbf{p}^T\mathbf{B}\mathbf{p} \text{ subject to } p_6 = 1, \tag{6}$$

where p_6 is the sixth element of vector $\mathbf{p}$, i.e. $p_6 = F$.

Indeed, we seek for a least-squares solution to $\mathbf{A}\mathbf{p} = \mathbf{0}$ under the constraint $p_6 = 1$. The equation can be rewritten as

$$\mathbf{A}'\mathbf{p}' = -\mathbf{a}_n,$$

where $\mathbf{A}'$ is the matrix formed by the first $(n-1)$ columns of $\mathbf{A}$, $\mathbf{a}_n$ is the last column of $\mathbf{A}$ and $\mathbf{p}'$ is the vector $[A, B, C, D, E]^T$. The problem can now be solved by the technique described in Section 4.1.

In the following, we present another technique for solving this kind of problems, i.e.

$$\mathbf{A}\mathbf{p} = 0 \quad \text{subject to } p_m = 1$$

based on eigen analysis, where we consider a general formulation, that is $\mathbf{A}$ is a $n \times m$ matrix, $\mathbf{p}$ is an m-vector, and p_m is the last element of vector $\mathbf{p}$. The function to minimize is

$$\mathscr{F}(\mathbf{p}) = (\mathbf{A}\mathbf{p})^T(\mathbf{A}\mathbf{p}) \equiv \mathbf{p}^T\mathbf{B}\mathbf{p} \quad \text{subject to } p_m = 1. \tag{7}$$

As in the last subsection, the symmetric matrix $\mathbf{B}$ can be decomposed as in Eq. (5), i.e. $\mathbf{B} = \mathbf{U}\mathbf{E}\mathbf{U}^T$. Now if we normalize each eigenvalue and eigenvector by the last element of the eigenvector, i.e.

$$v_i' = v_i e_{im}^2 \quad \mathbf{e}_i' = \frac{1}{e_{im}}\mathbf{e}_i,$$

where e_{im} is the last element of the eigenvector $\mathbf{e}_i$, then the last element of the new eigenvector $\mathbf{e}_i'$ is equal to one. We now have

$$\mathbf{B} = \mathbf{U}'\mathbf{E}'\mathbf{U}'^T,$$

where $\mathbf{E}' = \text{diag}(v_1', v_2', \ldots, v_m')$ and $\mathbf{U}' = [\mathbf{e}_1', \mathbf{e}_2', \ldots, \mathbf{e}_m']$. The original problem (7) becomes:

Find $q_1, q_2, \ldots, q_m$ such that $\mathbf{p}^T\mathbf{B}\mathbf{p}$ is minimized with $\mathbf{p} = q_1\mathbf{e}_1' + q_2\mathbf{e}_2' + \ldots + q_m\mathbf{e}_m'$ subject to $q_1 + q_2 + \cdots + q_m = 1$.

After some simple algebra, we have

$$\mathbf{p}^T\mathbf{B}\mathbf{p} = q_1^2 v_1' + q_2^2 v_2' + \cdots + q_m^2 v_m'.$$

The problem now becomes to minimize the following unconstrained function:

$$\mathscr{J} = q_1^2 v_1' + q_2^2 v_2' + \cdots + q_m^2 v_m' + \lambda(q_1 + q_2 + \cdots + q_m - 1),$$

where λ is the Lagrange multiplier. Setting the derivatives of $\mathscr{J}$ with respect to q_1 through q_m and λ to zero yields:

$$2q_1 v_1' + \lambda = 0$$

$$2q_2 v_2' + \lambda = 0$$

$$\cdots\cdots$$

$$2q_m v_m' + \lambda = 0$$

$$q_1 + q_2 + \cdots + q_m - 1 = 0.$$

The unique solution to the above equations is given by

$$q_i = \frac{1}{v_i'} S \quad \text{for } i = 1, \ldots, m,$$

where $S = 1/\sum_{j=1}^{m} 1/v_j'$. The solution to the problem (7) is given by

$$\mathbf{p} = \sum_{i=1}^{m} q_i \mathbf{e}_i' = \left(\sum_{i=1}^{m} \frac{1}{v_j'} \mathbf{e}_i'\right) \Big/ \left(\sum_{j=1}^{m} \frac{1}{v_j'}\right).$$

Note that this normalization ($F = 1$) has singularities for all conics going through the origin. That is, this method cannot fit conics because they require to set $F = 0$. This might suggest that the other normalizations are superior to the $F = 1$ normalization with respect to singularities. However, as shown in [6], the singularity problem can be overcome by shifting the data so that they are centered on the origin, and better results by setting $F = 1$ has been obtained than by setting $A + C = 1$.

5. Least-squares fitting based on Euclidean distances

In the above section, we have described three general techniques for solving linear least-squares problems, either unconstrained or constrained, based on algebraic distances. In this section, we describe why such techniques usually do not provide satisfactory results, and then propose to fit conics using directly Euclidean distances.

5.1. Why are algebraic distances usually not satisfactory?

The big advantage of use of algebraic distances is the gain in computational efficiency, because closed-form solutions can usually be obtained. In general, however, the results are not satisfactory. There are at least two major reasons:

- The function to minimize is usually not invariant under Euclidean transformations. For example, the function with normalization $F = 1$ is not invariant with respect to translations. This is a feature we dislike, because we usually do not know in practice where is the best coordinate system to represent the data.
- A point may contribute differently to the parameter estimation depending on its position on the conic. If a priori all points are corrupted by the same amount of noise, it is desirable for them to contribute the same way. (The problem with data points corrupted by different noise will be addressed in Section 8.)

To understand the second point, consider a conic in the normalized system (see Fig. 1):

$$Q(x, y) = Ax^2 + Cy^2 + F = 0.$$

The algebraic distance of a point (x_i, y_i) to the conic Q is given by [7]:

$$Q(x_i, y_i) = Ax_i^2 + Cy_i^2 + F = -F(d_i^2/c_i^2 - 1),$$

where d_i is the distance from the point (x_i, y_i) to the center O of the conic, and c_i is the distance from the conic to its center along the ray from the center to the point (x_i, y_i). It is thus clear that a point at the high curvature sections contributes less to the conic fitting than a point having the same amount of noise but at the low curvature sections. This is because a point at the high curvature sections has a large c_i and its $|Q(x_i, y_i)|^2$ is small, while a point at the low curvature sections has a small c_i and its $|Q(x_i, y_i)|^2$ is higher with respect to the same amount of noise in the data points. Concretely, methods based on algebraic distances tend to fit better a conic to the points at low curvature sections than to those at high curvature sections. This problem is usually termed *high curvature bias*.

5.2. Orthogonal distance fitting

To overcome the problems with algebraic distances, it is natural to replace them by the orthogonal distances, which are invariant to transformations in

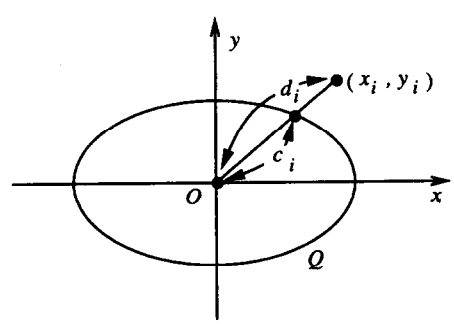

Fig. 1. Normalized conic.

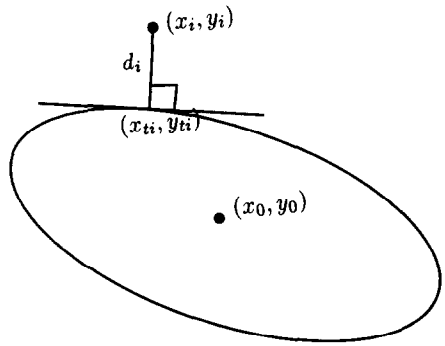

Fig. 2. Orthogonal distance of a point (x_i, y_i) to a conic. Point (x_{ti}, y_{ti}) is the point on the conic which is closest to point (x_i, y_i).

Euclidean space and which do not exhibit the high curvature bias.

The orthogonal distance d_i between a point $\mathbf{x}_i = (x_i, y_i)$ and a conic $Q(x, y)$ is the smallest Euclidean distance among all distances between $\mathbf{x}_i$ and points in the conic. The tangent at the corresponding point in the conic (denoted by $\mathbf{x}_t = (x_{ti}, y_{ti})$) is orthogonal to the line joining $\mathbf{x}_i$ and $\mathbf{x}_t$ (see Fig. 2). Given n points $\mathbf{x}_i$ $(i = 1, \ldots, n)$, the orthogonal distance fitting is to estimate the conic Q by minimizing the following function:

$$\mathscr{F}(\mathbf{p}) = \sum_{i=1}^{n} d_i^2. \tag{8}$$

However, as the expression of d_i is very complicated (see below), an iterative optimization procedure must be carried out. Many techniques are readily available, including the Gauss–Newton algorithm, Steepest Gradient Descent, the Levenberg–Marquardt procedure, and the simplex method. The software ODRPACK (written in Fortran) for weighted orthogonal distance regression is in the public domain, and is available from NETLIB (netlib@ornl.gov). An initial guess of the conic parameters must be supplied, which can be obtained using the techniques described in the last section.

Let us now proceed to compute the orthogonal distance d_i. The subscript i will be omitted for clarity. Refer again to Fig. 2. The conic is assumed to be described by

$$Q(x, y) = A(x - x_0)^3 + 2B(x - x_0)(y - y_0) + C(y - y_0)^2 - 1 = 0.$$

Point $\mathbf{x}_t = (x_t, y_t)$ must satisfy the following equations:

$$A(x_t - x_0)^2 + 2B(x_t - x_0)(y_t - y_0) + C(y_t - y_0)^2 - 1 = 0 \tag{9}$$

$$(y - y_t)\frac{\partial}{\partial x}Q(x_t, y_t) = (x - x_t)\frac{\partial}{\partial y}Q(x_t, y_t). \tag{10}$$

Eq. (9) merely says the point $\mathbf{x}_t$ is on the conic, while Eq. (10) says that the tangent at $\mathbf{x}_t$ is orthogonal to the vector $\mathbf{x} - \mathbf{x}_t$.

Let $\Delta_x = x_t - x_0$ and $\Delta_y = y_t - y_0$. From Eq. (9),

$$\Delta_y = \frac{-B\Delta_x \pm \xi}{C}, \tag{11}$$

where $\xi^2 = B^2\Delta_x^2 - C(A\Delta_x^2 - 1) = (B^2 - AC)\Delta_x^2 + C$. From Eq. (10),

$$(A\Delta_x + B\Delta_y)(y - y_0 - \Delta_y) = (C\Delta_y + B\Delta_x) \times (x - x_0 - \Delta_x).$$

Substituting the value of Δ_y (11) in the above equation leads to the following equation:

$$e_1\Delta_x^2 + e_2\Delta_x + e_3 = (e_4\Delta_x + e_5)\xi, \tag{12}$$

where

$$e_0 = B^2 - AC$$

$$e_1 = 2Be_0$$

$$e_2 = Ce_0(y - y_0)$$

$$e_3 = BC$$

$$e_4 = B^2 + C^2 + e_0$$

$$e_5 = C[B(y - y_0) - C(x - x_0)].$$

Squaring the above equation, we have

$$(e_1\Delta_x^2 + e_2\Delta_x + e_3)^2 = (e_4\Delta_x + e_5)^2(e_0\Delta_x^2 + 4C).$$

Rearranging the terms, we obtain an equation of degree four in Δ_x:

$$f_4\Delta_x^4 + f_3\Delta_x^3 + f_2\Delta_x^2 + f_1\Delta_x + f_0 = 0, \tag{13}$$

where

$$f_4 = e_1^2 - e_0e_4^2$$

$$f_3 = 2e_1e_2 - 2e_0e_4e_5$$

$$f_2 = e_2^2 + 2e_1e_3 - e_0e_5^2 - 4e_4^2C$$

$$f_1 = 2e_2e_3 - 8e_4e_5C$$

$$f_0 = e_3^2 - 4e_5^2C.$$

The two or four real roots of Eq. (13) can be found in closed form. For one solution Δ_x, we can obtain ξ from Eq. (12), i.e.

$$\xi = (e_1\Delta_x^2 + e_2\Delta_x + e_3)/(e_4\Delta_x + e_5).$$

Thus, Δ_y is computed from Eq. (11). Eventually comes the orthogonal distance d, which is given by

$$d = \sqrt{(x - x_0 - \Delta_x)^2 + (y - y_0 - \Delta_y)^2}.$$

Note that we possibly have four solutions. Only the one which gives the smallest distance is the one we are seeking.

6. Gradient weighted least-squares fitting

The least-squares method described in the last sections

is usually called *ordinary least-squares estimator* (OLS). Formally, we are given n equations:

$$f_i \equiv A_i^{\mathrm{T}}\mathbf{p} - b_i = \epsilon_i,$$

where ϵ_i is the additive error in the i-th equation with mean zero: $E(\epsilon_i) = 0$, and variance $\Lambda_{\epsilon_i} = \sigma_i^2$. Writing down in matrix form yields

$$\mathbf{Ap} - \mathbf{b} = \mathbf{e},$$

where

$$\mathbf{A} = \begin{bmatrix} A_1^{\mathrm{T}} \\ \vdots \\ A_n^{\mathrm{T}} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}, \quad \text{and} \quad \mathbf{e} = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}.$$

The OLS estimator tries to estimate $\mathbf{p}$ by minimizing the following sum of squared errors:

$$\mathscr{F} = \mathbf{e}^{\mathrm{T}}\mathbf{e} = (\mathbf{Ap} - \mathbf{b})^{\mathrm{T}}(\mathbf{Ap} - \mathbf{b}),$$

which gives, as we have already seen, the solution as

$$\mathbf{p} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}.$$

It can be shown (see, e.g. [3]) that the OLS estimator produces the optimal estimate of $\mathbf{p}$, 'optimal' in terms of *minimum covariance of* $\mathbf{p}$, if the errors ϵ_i are uncorrelated (i.e. $E(\epsilon_i\epsilon_j) = \sigma_i^2\delta_{ij}$) and their variances are constant (i.e. $\Lambda_{\epsilon_i} = \sigma^2 \forall i \in [1, \ldots, n]$).

Now let us examine whether the above assumptions are valid for conic fitting. Data points are provided by some signal processing algorithm such as edge detection. It is reasonable to assume that errors are independent from one point to another, because when detecting a point we usually do not use any information from other points. It is also reasonable to assume the errors are constant for all points because we use the same algorithm for edge detection. However, we must note that we are talking about the errors in the points, *but not those in the equations* (i.e. ϵ_i). Let the error in a point $\mathbf{x}_i = [x_i, y_i]^{\mathrm{T}}$ be Gaussian with mean zero and covariance $\Lambda_{\mathbf{x}} = \sigma^2\mathbf{I}_2$, where $\mathbf{I}_2$ is the 2×2 identity matrix. That is, the error distribution is assumed to be equal in both directions ($\sigma_x^2 = \sigma_y^2 = \sigma^2$, $\sigma_{xy} = 0$). In Section 8, we will consider the case where each point may have different noise distribution. Refer to Eq. (2). We now compute the variance Λ_{ϵ_i} of function f_i from point (x_i, y_i) and its uncertainty. Let $(\hat{x}_i, \hat{y}_i)$ be the true position of the point, we have certainly

$$\hat{f}_i = (\hat{x}_i^2 - \hat{y}_i^2)A + 2\hat{x}_i\hat{y}_iB + 2\hat{x}_iD + 2\hat{y}_iE + F + \hat{y}_i^2 = 0.$$

We now expand f_i into Taylor series at $x = \hat{x}_i$ and $y = \hat{y}_i$, i.e.

$$f_i = \hat{f}_i + \frac{\partial f_i}{\partial x_i}(x_i - \hat{x}_i) + \frac{\partial f_i}{\partial y_i}(y_i - \hat{y}_i) + O((x_i - \hat{x}_i)^2) + O((y_i - \hat{y}_i)^2),$$

where the derivatives are evaluated at $(\hat{x}_i, \hat{y}_i)$. Ignoring the high order terms, we can now compute the variance of f_i, i.e.

$$\begin{aligned}\Lambda_{\epsilon_i} = E(f_i - \hat{f}_i)^2 &= \left(\frac{\partial f_i}{\partial x_i}\right)^2 E(x_i - \hat{x}_i)^2 \\ &\quad + \left(\frac{\partial f_i}{\partial y_i}\right)^2 E(y_i - \hat{y}_i)^2 \\ &= \left[\left(\frac{\partial f_i}{\partial x_i}\right)^2 + \left(\frac{\partial f_i}{\partial y_i}\right)^2\right]\sigma^2 \equiv \|\nabla f_i\|^2\sigma^2,\end{aligned}$$

where ∇f_i is just the gradient of f_i with respect to x_i and y_i, and

$$\begin{aligned}\frac{\partial f_i}{\partial x_i} &= 2Ax_i + 2By_i + 2D \\ \frac{\partial f_i}{\partial y_i} &= -2Ay_i + 2Bx_i + 2E + 2y_i.\end{aligned} \tag{14}$$

It is now clear that the variance of each equation is not the same, and thus the OLS estimator does not yield an optimal solution.

To obtain a constant variance function, it is sufficient to divide the original function by its gradient, i.e.

$$f_i' = f_i/\nabla f_i,$$

then f_i' has the constant variance σ^2. We can now try to find the parameters $\mathbf{p}$ by minimizing the following function:

$$\mathscr{F} = \sum f_i'^2 = \sum f_i^2/\|\nabla f_i\|^2 = (\mathbf{Ap} - \mathbf{b})^{\mathrm{T}}\mathbf{W}^{-1}(\mathbf{Ap} - \mathbf{b}),$$

where $\mathbf{W} = \mathrm{diag}(\|\nabla f_1\|^2, \|\nabla f_2\|^2, \ldots, \|\nabla f_n\|^2)$. This method is thus called Gradient Weighted Least-Squares, and the solution can be easily obtained by setting $\frac{\partial \mathscr{F}}{\partial \mathbf{p}} = 2\mathbf{A}^{\mathrm{T}}\mathbf{W}^{-1}(\mathbf{Ap} - \mathbf{b}) = 0$, which yields

$$\mathbf{p} = (\mathbf{A}^{\mathrm{T}}\mathbf{W}^{-1}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{W}^{-1}\mathbf{b}. \tag{15}$$

Note that the gradient-weighted LS is in general a nonlinear minimization problem, and a closed-form solution does not exist. In the above, we gave a closed-form solution because we have ignored the dependence of $\mathbf{W}$ on $\mathbf{p}$ in computing $\frac{\partial \mathscr{F}}{\partial \mathbf{p}}$. In reality, $\mathbf{W}$ does depend on $\mathbf{p}$, as can be seen in Eq. (14). Therefore, the above solution is only an approximation. In practice, we run the following iterative procedure:

Step 1. $k = 0$. Compute $\mathbf{p}^{(0)}$ using OLS, Eq. (3);
Step 2. Compute the weight matrix $\mathbf{W}^{(k)}$ at the current conic estimate and measurements;
Step 3. Compute $\mathbf{p}^{(k)}$ using Gradient Weighted LS, Eq. (15);
Step 4. If $\mathbf{p}^{(k)}$ is very close to $\mathbf{p}^{(k-1)}$, then **stop**; otherwise **go to step 2.**

In the above, the superscript (k) denotes the iteration number.

7. Bias-corrected renormalization fitting

Consider the quadratic representation of an ellipse:

$$Ax^2 + 2Bxy + Cy^2 + 2Dx + 2Ey + F = 0.$$

Given n noisy points $\mathbf{x}_i = (x_i, y_i)$ $(i = 1, \ldots, n)$, we want to estimate the coefficients of the ellipse: $\mathbf{p} = [A, B, C, D, E, F]^T$. Due to the homogeneity, we set $\|\mathbf{p}\| = 1$.

For each point $\mathbf{x}_i$, we thus have one scalar equation:

$$f_i = M_i^T \mathbf{p} = 0,$$

where

$$M_i = [x_i^2, 2x_i y_i, y_i^2, 2x_i, 2y_i, 1]^T.$$

Hence, $\mathbf{p}$ can be estimated by minimizing the following objective function (weighted least-squares optimization)

$$\mathscr{F} = \Sigma_{i=1}^n w_i (M_i^T \mathbf{p})^2, \tag{16}$$

where w_i's are positive weights. It can be rewritten as

$$\mathscr{F} = \mathbf{p}^T (\Sigma_{i=1}^n w_i M_i M_i^T) \mathbf{p},$$

which is a quadratic form in unit vector $\mathbf{p}$, if we ignore the possible dependence of w_i's on $\mathbf{p}$. Let

$$\mathbf{N} = \Sigma_{i=1}^n w_i M_i M_i^T.$$

The solution is the eigenvector of $\mathbf{N}$ associated to the smallest eigenvalue.

Assume that each point has the same error distribution with mean zero and covariance $\Lambda_{\mathbf{x}_i} = [(\sigma^2\ 0)/(0\ \sigma^2)]$. The covariance of f_i is then given by

$$\Lambda_{f_i} = \begin{bmatrix} \frac{\partial f_i}{\partial x_i} & \frac{\partial f_i}{\partial y_i} \end{bmatrix} \Lambda_{\mathbf{x}_i} \begin{bmatrix} \frac{\partial f_i}{\partial x_i} & \frac{\partial f_i}{\partial y_i} \end{bmatrix}^T,$$

where

$$\frac{\partial f_i}{\partial x_i} = 2(Ax_i + By_i + D)$$

$$\frac{\partial f_i}{\partial y_i} = 2(Bx_i + Cy_i + E).$$

Thus we have

$$\Lambda_{f_i} = 4\sigma^2[(A^2 + B^2)x_i^2 + (B^2 + C^2)y_i^2 + 2B(A + C)x_i y_i + 2(AD + BE)x_i + 2(BD + CE)y_i + (D^2 + E^2)].$$

The weights can then be chosen to the inverse proportion of the variances. Since multiplication by a constant does not affect the result of the estimation, we set

$$w_i = 4\sigma^2/\Lambda_{f_i} = 1/[(A^2 + B^2)x_i^2 + (B^2 + C^2)y_i^2 + 2B(A + C)x_i y_i + 2(AD + BE)x_i + 2(BD + CE)y_i + (D^2 + E^2)].$$

We now show that the estimate obtained by the above method is *biased*. If each point $\mathbf{x}_i = (x_i, y_i)$ is perturbed by noise of $\Delta\mathbf{x}_i = (\Delta x_i, \Delta y_i)$ with

$$E[\Delta\mathbf{x}_i] = 0, \quad \text{and} \quad \Lambda_{\Delta\mathbf{x}_i} = E[\Delta\mathbf{x}_i \Delta\mathbf{x}_i^T] = \begin{bmatrix} \sigma^2 & \\ & \sigma^2 \end{bmatrix},$$

the matrix $\mathbf{N}$ is perturbed accordingly: $\mathbf{N} = \bar{\mathbf{N}} + \Delta\mathbf{N}$, where $\bar{\mathbf{N}}$ is the unperturbed matrix. If $E[\Delta\mathbf{N}] = 0$, then the estimate is *statistically unbiased*; otherwise, it is *statistically biased*, because, following the perturbation theorem [8], the bias of $\mathbf{p}$ is equal to $E[\Delta\mathbf{p}] = O(E[\Delta\mathbf{N}])$.

Let $\mathbf{N}_i = M_i M_i^T$, then $\mathbf{N} = \Sigma_{i=1}^n w_i \mathbf{N}_i$. We have

$$\mathbf{N}_i = \begin{bmatrix} x_i^4 & 2x_i^3 y_i & x_i^2 y_i^2 & 2x_i^3 & 2x_i^2 y_i & x_i^2 \\ 2x_i^3 y_i & 4x_i^2 y_i^2 & 2x_i y_i^3 & 4x_i^2 y_i & 4x_i y_i^2 & 2x_i y_i \\ x_i^2 y_i^2 & 2x_i y_i^3 & y_i^4 & 2x_i y_i^2 & 2y_i^3 & y_i^2 \\ 2x_i^3 & 4x_i^2 y_i & 2x_i y_i^2 & 4x_i^2 & 4x_i y_i & 2x_i \\ 2x_i^2 y_i & 4x_i y_i^2 & 2y_i^3 & 4x_i y_i & 4y_i^2 & 2y_i \\ x_i^2 & 2x_i y_i & y_i^2 & 2x_i & 2y_i & 1 \end{bmatrix}.$$

If we carry out the Taylor development and ignore quantities of order higher than 2, it can be shown that the expectation of $\Delta\mathbf{N}$ is given by

$$E[\Delta\mathbf{N}_i] = \sigma^2 \begin{bmatrix} 6x_i^2 & 6x_i y_i & x_i^2 + y_i^2 & 6x_i & 2y_i & 1 \\ 6x_i y_i & 4(x_i^2 + y_i^2) & 6x_i y_i & 4y_i & 4x_i & 0 \\ x_i^2 + y_i^2 & 6x_i y_i & 6y_i^2 & 2x_i & 6y_i & 1 \\ 6x_i & 4y_i & 2x_i & 4 & 0 & 0 \\ 2y_i & 4x_i & 6y_i & 0 & 4 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \equiv c\mathscr{B}_i.$$

It is clear that if we define

$$\hat{\mathbf{N}} = \Sigma_{i=1}^n w_i [\mathbf{N}_i - E[\Delta\mathbf{N}_i]] = \Sigma_{i=1}^n w_i [\mathbf{N}_i - c\mathscr{B}_i],$$

then $\hat{\mathbf{N}}$ is unbiased, i.e. $E[\hat{\mathbf{N}}] = \bar{\mathbf{N}}$, and hence the unit eigenvector $\mathbf{p}$ of $\hat{\mathbf{N}}$ associated to the smallest eigenvalue is an *unbiased* estimate of the exact solution $\bar{\mathbf{p}}$.

Ideally, the constant c should be chosen so that $E[\hat{\mathbf{N}}] = \bar{\mathbf{N}}$, but this is impossible unless image noise characteristics are known. On the other hand, if $E[\hat{\mathbf{N}}] = \bar{\mathbf{N}}$, we have

$$E[\bar{\mathbf{p}}^T \hat{\mathbf{N}} \bar{\mathbf{p}}] = \bar{\mathbf{p}}^T E[\hat{\mathbf{N}}] \bar{\mathbf{p}} = \bar{\mathbf{p}}^T \bar{\mathbf{N}} \bar{\mathbf{p}} = 0,$$

because $\mathscr{F} = \mathbf{p}^T \mathbf{N} \mathbf{p}$ takes its absolute minimum 0 for the exact solution $\bar{\mathbf{p}}$ in the absence of noise. This suggests that we require that $\mathbf{p}^T \mathbf{N} \mathbf{p} = 0$ at each iteration. If for the current c and $\mathbf{p}$, $\mathbf{p}^T \hat{\mathbf{N}} \mathbf{p} = \lambda_{\min} \neq 0$, we can update c by Δc such that

$$\mathbf{p}^T \Sigma_{i=1}^n [w_i \mathbf{N}_i - c w_i \mathscr{B}_i] \mathbf{p} - \mathbf{p}^T \Sigma_{i=1}^n \Delta c w_i \mathscr{B}_i \mathbf{p} = 0.$$

That is,

$$\Delta c = \frac{\lambda_{\min}}{\mathbf{p}^T \Sigma_{i=1}^n w_i \mathscr{B}_i \mathbf{p}}.$$

To summarize, the renormalization procedure can be described as:

1. Let $c = 0$, $w_i = 1$ for $i = 1, \ldots, n$.
2. Compute the unit eigenvector $\mathbf{p}$ of

   $\hat{\mathbf{N}} = \Sigma_{i=1}^{n} w_i[\mathbf{N}_i - c\mathscr{B}_i]$

   associated to the smallest eigenvalue, which is denoted by $\lambda_{\min}$.
3. Update c as

   $$c \leftarrow c + \frac{\lambda_{\min}}{\mathbf{p}^{\mathrm{T}} \Sigma_{i=1}^{n} w_i \mathscr{B}_i \mathbf{p}}$$

   and recompute w_i using the new $\mathbf{p}$.
4. Return $\mathbf{p}$ if the update has converged; go back to step 2 otherwise.

Remark 1. This implementation is different from that described in the paper of Kanatani [9]. This is because in his implementation, he uses the N-vectors to represent the 2D points. In the derivation of the bias, he assumes that the perturbation in each N-vector, i.e. $\Delta\mathbf{m}_\alpha$ in his notations, has the same magnitude $\tilde{\epsilon} = \sqrt{E(\|\Delta\mathbf{m}_\alpha^2\|)}$. This is an unrealistic assumption. In fact, to the first order,

$$\Delta\mathbf{m}_\alpha = \frac{1}{\sqrt{x_\alpha^2 + y_\alpha^2 + f^2}} \begin{bmatrix} \Delta x_\alpha \\ \Delta y_\alpha \\ 0 \end{bmatrix},$$

thus

$$\|\Delta\mathbf{m}_\alpha^2\|^2 = \frac{\Delta x_\alpha^2 + \Delta y_\alpha^2}{x_\alpha^2 + y_\alpha^2 + f^2}.$$

Hence,

$$E(\|\Delta\mathbf{m}_\alpha^2\|) = \frac{2\sigma^2}{x_\alpha^2 + y_\alpha^2 + f^2},$$

where we assume the perturbation in the image plane is the same for each point (with mean zero and standard deviation σ).

Remark 2. This method is optimal only in the sense of *unbiasness*. Another criterion of optimality, namely the *minimum variance of estimation*, is not addressed in this method. See [10] for a discussion.

Remark 3. This method is based on statistical analysis of data points. It is thus not useful if the size of data set is small (say, less than 30 data points).

8. Kalman filtering technique

Kalman filtering, as pointed out by Lowe [11], is likely to have applications throughout computer vision as a general method for integrating noisy measurements. The reader is referred to [12] for an introduction of the Kalman filter and its applications to 3D computer vision, and to [13–15] for more theoretical studies. The following subsection collects the Kalman filter formulas for simpler reference.

8.1. Kalman filter

If we denote the state vector by $\mathbf{s}$ and denote the measurement vector by $\mathbf{x}$, a linear dynamic system (in discrete-time form) can be described by

$$\mathbf{s}_{i+1} = \mathbf{H}_i\mathbf{s}_i + \mathbf{n}_i, \quad i = 0, 1, \ldots, \tag{17}$$

$$\mathbf{x}_i = F_i\mathbf{s}_i + \boldsymbol{\eta}_i, \quad i = 0, 1, \ldots. \tag{18}$$

In Eq. (17), $\mathbf{n}_i$ is the vector of random disturbance of the dynamic system and is usually modeled as white noise:

$$E[\mathbf{n}_i] = \mathbf{0} \quad \text{and} \quad E[\mathbf{n}_i\mathbf{n}_i^{\mathrm{T}}] = Q_i.$$

In practice, the system noise covariance Q_i is usually determined on the basis of experience and intuition (i.e. it is guessed). In Eq. (18), we assume the measurement system is disturbed by additive white noise $\boldsymbol{\eta}_i$, i.e.

$$E[\boldsymbol{\eta}_i] = \mathbf{0},$$

$$E[\boldsymbol{\eta}_i\boldsymbol{\eta}_j^{\mathrm{T}}] = \begin{cases} \Lambda_{\eta_i} & \text{for } i = j, \\ \mathbf{0} & \text{for } i \neq j. \end{cases}$$

The measurement noise covariance Λ_{η_i} is either provided by some signal processing algorithm or guessed in the same manner as the system noise. In general, these noise levels are determined independently. We assume then there is no correlation between the noise process of the system and that of the observation, that is

$$E[\boldsymbol{\eta}_i\mathbf{n}_j^{\mathrm{T}}] = \mathbf{0} \quad \text{for every } i \text{ and } j.$$

The standard Kalman filter (KF) is then described by the following steps:

- Prediction of states: $\hat{\mathbf{s}}_{i|i-1} = H_{i-1}\hat{\mathbf{s}}_{i-1}$
- Prediction of the state covariance matrix: $P_{i|i-1} = H_{i-1}P_{i-1}H_{i-1}^{\mathrm{T}} + Q_{i-1}$
- Kalman gain matrix: $K_i = P_{i|i-1}F_i^{\mathrm{T}}(F_iP_{i|i-1}F_i^{\mathrm{T}} + \Lambda_{\eta_i})^{-1}$
- Update of the state estimation: $\hat{\mathbf{s}}_i = \hat{\mathbf{s}}_{i|i-1} + K_i(\mathbf{x}_i - F_i\hat{\mathbf{s}}_{i|i-1})$
- Update of the covariance matrix of states: $P_i = (\mathbf{I} - K_iF_i)P_{i|i-1}$
- Initialization: $P_{0|0} = \Lambda_{\mathbf{s}_0}$, $\hat{\mathbf{s}}_{0|0} = E[\mathbf{s}_0]$

Fig. 3 is a block diagram for the Kalman filter. At time t_i, the system model inherently in the filter structure generates $\hat{\mathbf{s}}_{i|i-1}$, the best prediction of the state, using the previous state estimate $\hat{\mathbf{s}}_{i-1}$. The previous state covariance matrix P_{i-1} is extrapolated to the predicted state covariance matrix $P_{i|i-1}$. $P_{i|i-1}$ is then used to compute the Kalman gain matrix K_i and to update the covariance matrix P_i. The system model generates also $F_i\hat{\mathbf{s}}_{i|i-1}$,

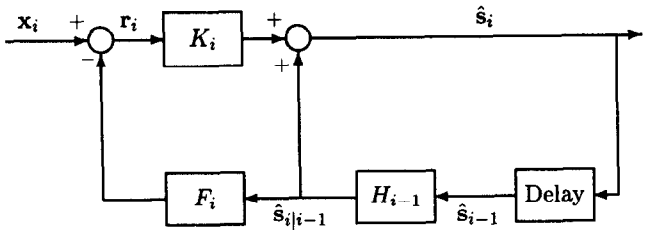

Fig. 3. Kalman filter block diagram.

which is the best prediction of what the measurement at time t_i will be. The real measurement $\mathbf{x}_i$ is then read in, and the measurement residual (also called *innovation*)

$$\mathbf{r}_i = \mathbf{x}_i - F_i \hat{\mathbf{s}}_{i|i-1}$$

is computed. Finally, the residual $\mathbf{r}_i$ is weighted by the Kalman gain matrix K_i to generate a correction term and is added to $\hat{\mathbf{s}}_{i|i-1}$ to obtain the updated state $\hat{\mathbf{s}}_i$.

If the system equation is not linear (e.g. described by $\mathbf{s}_{i+1} = \mathbf{h}_i(\mathbf{s}_i) + \mathbf{n}_i$), then the prediction of states is based on

$$\hat{\mathbf{s}}_{i|i-1} = \mathbf{h}_i(\hat{\mathbf{s}}_{i-1})$$

and the prediction of the covariance matrix of states is based on

$$P_{i|i-1} = \frac{\partial \mathbf{h}_i}{\partial \mathbf{s}_i} P_{i-1} \frac{\partial \mathbf{h}_i^{\mathrm{T}}}{\partial \mathbf{s}_i} + Q_{i-1}.$$

If the measurement equation is nonlinear, we assume that it is described by

$$\mathbf{f}_i(\mathbf{x}_i', \mathbf{s}_i) = \mathbf{0},$$

where $\mathbf{x}_i'$ is the ideal measurement. The real measurement $\mathbf{x}_i$ is assumed to be corrupted by additive noise η_i, i.e. $\mathbf{x}_i = \mathbf{x}_i' + \eta_i$. We expand $\mathbf{f}_i(\mathbf{x}_i', \mathbf{s}_i)$ into a Taylor series about $(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})$:

$$\begin{aligned}\mathbf{f}_i(\mathbf{x}_i', \mathbf{s}_i) = \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1}) &+ \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})}{\partial \mathbf{x}_i'}(\mathbf{x}_i' - \mathbf{x}_i) \\ &+ \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})}{\partial \mathbf{s}_i}(\mathbf{s}_i - \hat{\mathbf{s}}_{i|i-1}) + O((\mathbf{x}_i' - \mathbf{x}_i)^2) \\ &+ O((\mathbf{s}_i - \hat{\mathbf{s}}_{i|i-1})^2). \end{aligned} \quad (19)$$

By ignoring the second order terms, we get a linearized measurement equation:

$$\mathbf{y}_i = M_i \mathbf{s}_i + \boldsymbol{\xi}_i, \quad (20)$$

where $\mathbf{y}_i$ is the new measurement vector, $\boldsymbol{\xi}_i$ is the noise vector of the new measurement, and M_i is the linearized transformation matrix. They are given by

$$M_i = \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})}{\partial \mathbf{s}_i},$$

$$\mathbf{y}_i = -\mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1}) + \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})}{\partial \mathbf{s}_i} \hat{\mathbf{s}}_{i|i-1},$$

$$\boldsymbol{\xi}_i = \frac{\partial \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})}{\partial \mathbf{x}_i'}(\mathbf{x}_i' - \mathbf{x}_i) = -\frac{\partial \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})}{\partial \mathbf{x}_i'} \eta_i.$$

Clearly, we have $E[\boldsymbol{\xi}_i] = \mathbf{0}$, and $E[\boldsymbol{\xi}_i \boldsymbol{\xi}_i^{\mathrm{T}}] = \partial \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})/\partial \mathbf{x}_i' \, \Lambda_{\eta_i} \, \partial \mathbf{f}_i(\mathbf{x}_i, \hat{\mathbf{s}}_{i|i-1})^{\mathrm{T}}/\partial \mathbf{x}_i' \equiv \Lambda_{\xi_i}$. The KF can then be applied to the above linearized equation. This is known as the Extended Kalman filter (EKF).

8.2. Application to conic fitting

Let us choose the normalization with $A + C = 1$ (see Section 4.1). The state vector can now be defined as:

$$\mathbf{s} = [A, B, D, E, F]^{\mathrm{T}}.$$

The measurement vector is: $\mathbf{x}_i = [x_i, y_i]^{\mathrm{T}}$. As the conic parameters are the same for all points, we have the following simple system equation:

$$\mathbf{s}_{i+1} = \mathbf{s}_i,$$

and the noise term $\mathbf{n}_i$ is zero. The observation function is

$$f_i(\mathbf{x}_i, \mathbf{s}) = (x_i^2 - y_i^2)A + 2x_i y_i B + 2x_i D + 2y_i E + F + y_i^2.$$

To apply the extended Kalman filter, we need to compute the derivative of $f_i(\mathbf{x}_i, \mathbf{s})$ with respect to $\mathbf{s}$ and that with respect to $\mathbf{x}_i$, which are given by

$$\frac{\partial f_i(\mathbf{x}_i, \mathbf{s})}{\partial \mathbf{s}} = [x_i^2 - y_i^2, 2x_i y_i, 2x_i, 2y_i, 1] \quad (21)$$

$$\frac{\partial f_i(\mathbf{x}_i, \mathbf{s})}{\partial \mathbf{x}_i} = 2[x_i A + y_i B + D, -y_i A + x_i B + E + y_i]. \quad (22)$$

Note that the Kalman filtering technique is usually applied to a temporal sequence. Here, it is applied to a *spatial* sequence. Due to its recursive nature, it is more suitable to problems where the measurements are available in a serial manner. Otherwise (i.e. if all measurements are available — or could be made available at no serious overhead — at the same time), it is advantageous to exploit them in a single joint evaluation, as described in the previous sections (which are sometimes known as *batch processing*). This is because the Kalman filtering technique is equivalent to the least-squares technique only if the system is linear. For nonlinear problems, the EKF will yield different results depending on the order of processing the measurements one after the other, and may run the risk of being trapped into a false minimum.

9. Robust estimation

9.1. Introduction

As has been stated before, least-squares estimators assume that the noise corrupting the data is of zero mean, which yields an *unbiased* parameter estimate. If the noise variance is known, an *minimum-variance* parameter estimate can be obtained by choosing appropriate weights on the data. Furthermore, least-squares estimators implicitly assume that the entire set of data can

be interpreted by *only one parameter vector* of a given model. Numerous studies have been conducted, which clearly show that least-squares estimators are vulnerable to the violation of these assumptions. Sometimes even when the data contains only one bad datum, least-squares estimates may be completely perturbed. During the last three decades, many robust techniques have been proposed, which are not very sensitive to departure from the assumptions on which they depend.

Hampel [16] gives some justifications to the use of robustness (quoted in [17]):

> What are the reasons for using robust procedures? There are mainly two observations which combined give an answer. Often in statistics one is using a parametric model implying a very limited set of probability distributions thought possible, such as the common model of normally distributed errors, or that of exponentially distributed observations. Classical (parametric) statistics derives results under the assumption that these models were strictly true. However, apart from some simple discrete models perhaps, such models are never exactly true. We may try to distinguish three main reasons for the derivations: (i) rounding and grouping and other 'local inaccuracies'; (ii) the occurrence of 'gross errors' such as blunders in measuring, wrong decimal points, errors in copying, inadvertent measurement of a member of a different population, or just 'something went wrong'; (iii) the model may have been conceived only as an approximation anyway, e.g. by virtue of the central limit theorem.

If we have some a priori knowledge about the parameters to be estimated, techniques, e.g. the Kalman filtering technique, based on the test of *Mahalanobis distance* can be used to yield a robust estimate [12].

In the following, we describe four major approaches to robust estimation.

9.2. Clustering or Hough transform

One of the oldest robust methods used in image analysis and computer vision is the *Hough transform* [18,19]. The idea is to map data into the parameter space, which is appropriately *quantized*, and then seek for the most likely parameter values to interpret data through *clustering*. A classical example is the detection of straight lines given a set of edge points. In the following, we take the example of estimating plane rigid motion from two sets of points.

Given p 2D points in the first set, noted $\{\mathbf{m}_i\}$, and q 2D points in the second set, noted $\{\mathbf{m}'_j\}$, we must find a rigid transformation between the two sets. The pairing between $\{\mathbf{m}_i\}$ and $\{\mathbf{m}'_j\}$ is assumed not to be known. A rigid transformation can be uniquely decomposed into a rotation around the origin and a translation, in that order. The corresponding parameter space is three-dimensional: one parameter for the rotation angle θ and two for the translation vector $\mathbf{t} = [t_x, t_y]^T$. More precisely, if $\mathbf{m}_i$ is paired to $\mathbf{m}'_j$, then

$$\mathbf{m}'_j = \mathbf{R}\mathbf{m}_i + \mathbf{t} \quad \text{with } \mathbf{R} = \begin{pmatrix} \cos\theta & -\sin q \\ \sin\theta & \cos\theta \end{pmatrix}.$$

It is clear that at least two pairings are necessary for a unique estimate of rigid transformation. The three-dimensional parameter space is quantized as many levels as necessary according to the required precision. The rotation angle θ ranges from 0 to 2π. We can fix the quantization interval for the rotation angle at, say, $\pi/6$, and we have $N_\theta = 12$ units. The translation is not bounded theoretically, but it is in practice. We can assume for example that the translation between two images cannot exceed 200 pixels in each direction (i.e. $t_x,\ t_y \in [-200, 200]$). If we choose an interval of 20 pixels, then we have $N_{t_x} = N_{t_y} = 20$ units in each direction. The quantized space can then be considered as a three-dimensional *accumulator* of $N = N_\theta N_{t_x} N_{t_y} = 4800$ cells. Each cell is initialized to zero.

Since one pairing of points does not entirely constrain the motion, it is difficult to update the accumulator because the constraint on motion is not simple. Instead, we use n point matches, where n is the smallest number such that matching n points in the first set to n points in the second set completely determines the motion (in our case, $n = 2$). Let $(\mathbf{z}_k, \mathbf{z}'_l)$ be one set of such matches, where $\mathbf{z}_k$ and $\mathbf{z}'_l$ are both vectors of dimension $2n$, each composed of n points in the first and second set, respectively. Then the number of sets to be considered is of order of $p^n q^n$ (of course, we do not need to consider all sets in our particular problem, because the distance invariance between a pair of points under rigid transformation can be used to discard the infeasible sets). For each such set, we compute the motion parameters, and the corresponding accumulator cell is increased by 1. After all sets have been considered, peaks in the accumulator indicate the best candidates for the motion parameters.

In general, if the size of data is not sufficient large compared to the number of unknowns, then the maximum peak is not much higher than other peaks, and it may not be the correct one because of data noise and because of parameter space quantization. The Hough transform is thus highly suitable for problems having enough data to support the expected solution.

Because of noise in the measurements, the peak in the accumulator corresponding to the correct solution may be very blurred and thus not easily detectable. The accuracy in the localization with the above simple implementation may be poor. There are several ways to improve it:

- Instead of selecting just the maximum peak, we can fit a quadratic hyper-surface. The position of its maximum gives a better localization in the parameter

space, and the curvature can be used as an indication of the uncertainty of the estimation.
- Statistical clustering techniques [20] can be used to discriminate different candidates of the solution.
- Instead of using an integer accumulator, the uncertainty of data can be taken into account and propagated to the parameter estimation, which would considerably increase the performance.

The Hough transform technique actually follows the principle of *maximum likelihood estimation*. Let $\mathbf{p}$ be the parameter vector ($\mathbf{p} = [\theta, t_x, t_y]^T$ in the above example). Let $\mathbf{x}_m$ be one datum ($\mathbf{x}_m = [\mathbf{z}_k^T, \mathbf{z}_l'^T]^T$ in the above example). Under the assumption that the data $\{\mathbf{x}_m\}$ represent the complete sample of the probability density function of $\mathbf{p}$, $f_\mathbf{p}(\mathbf{p})$, we have

$$L(\mathbf{p}) = f_\mathbf{p}(\mathbf{p}) = \sum_m f_\mathbf{p}(\mathbf{p}|\mathbf{x}_m)\Pr(\mathbf{x}_m)$$

by using the law of total probability. The maximum of $f_\mathbf{p}(\mathbf{p})$ is considered as the estimation of $\mathbf{p}$. The Hough transform described above can thus be considered as a discretized version of a maximum likelihood estimation process.

Because of its nature of global search, the Hough transform technique is rather robust, even when there is a high percentage of gross errors in the data. For better accuracy in the localization of solution, we can increase the number of samples in each dimension of the quantized parameter space. The size of the accumulator increases rapidly with the required accuracy and the number of unknowns. This technique is rarely applied to solve problems having more than three unknowns, and is not suitable for conic fitting.

9.3. Regression diagnostics

Another old robust method is the so-called *regression diagnostics* [21]. It tries to iteratively detect possibly wrong data and reject them through analysis of globally fitted model. The classical approach works as follows:

1. Determine an initial fit to the whole set of data through least squares.
2. Compute the residual for each datum.
3. Reject all data whose residuals exceed a predetermined threshold; if no data has been removed, then stop.
4. Determine a new fit to the remaining data, and go to step 2.

Clearly, the success of this method depends tightly upon the quality of the initial fit. If the initial fit is very poor, then the computed residuals based on it are meaningless; so is the diagnostics of them for outlier rejection. As pointed out by Barnett and Lewis [22], with least-squares techniques, *even one or two outliers in a large set can wreak havoc*! This technique thus does not guarantee a correct solution. However, experiences have shown that this technique works well for problems with a moderate percentage of outliers, and more importantly, outliers which do not deviate too much from good data.

The threshold on residuals can be chosen by experiences using for example graphical methods (plotting residuals in different scales). It is better to use a priori statistical noise model of data and a chosen confidence level. Let r_i be the residual of the ith data, and σ_i be the predicted variance of the ith residual based on the characteristics of the data nose and the fit, the standard test statistics $e_i = r_i/\sigma_i$ can be used. If e_i is not acceptable, the corresponding datum should be rejected.

One improvement to the above technique uses *influence measures* to pinpoint potential outliers. These measures assess the extent to which a particular datum influences the fit by determining the change in the solution when that datum is omitted. The refined technique work as follows:

1. Determine an initial fit to the whole set of data through least squares.
2. Conduct a statistic test whether the measure of fit f (e.g. sum of square residuals) is acceptable; if it is, then stop.
3. For each datum I, delete it from the data set and determine the new fit, each giving a measure of fit denoted by f_i. Hence, determine the change in the measure of fit, i.e. $\Delta f_i = f - f_i$, when datum i is deleted.
4. Delete datum i for which Δf_i is the largest, and go to step 2.

It can be shown [23] that the above two techniques agrees with each other at the *first order approximation*, i.e. the datum with the largest residual is also that datum inducing maximum change in the measure of fit. However, if the second (and higher) order terms are considered, the two techniques do not give the same results in general. Their difference is that whereas the former simply rejects the datum that deviates most from the current fit, the latter rejects the point whose exclusion will result in the best fit on the *next* iteration. In other words, the second technique looks ahead to the next fit to see what improvements will actually materialize, which is usually preferred.

As can be remarked, the regression diagnostics approach depends heavily on a priori knowledge in choosing the thresholds for outlier rejection.

9.4. M-estimators

One popular robust technique is the so-called *M-estimators*. Let r_i be the *residual* of the ith datum, i.e. the difference between the ith observation and its

fitted value. The standard least-squares method tries to minimize $\sum_i r_i^2$, which is unstable if there are outliers present in the data. Outlying data give an effect so strong in the minimization that the parameters thus estimated are distorted. The M-estimators try to reduce the effect of outliers by replacing the squared residuals r_i^2 by another function of the residuals, yielding

$$\min \sum_i \rho(r_i), \tag{23}$$

where ρ is a symmetric, positive-definite function with a unique minimum at zero, and is chosen to be less increasing than square. Instead of solving directly this problem, we can implement it as an iterated reweighted least-squares one. Now let us see how.

Let $\mathbf{p} = [p_1, \ldots, p_m]^T$ be the parameter vector to be estimated. The M-estimator of $\mathbf{p}$ based on the function $\rho(r_i)$ is the vector $\mathbf{p}$ which is the solution of the following m equations:

$$\sum_i \psi(r_i) \frac{\partial r_i}{\partial p_j} = 0, \quad \text{for } j = 1, \ldots, m, \tag{24}$$

where the derivative $\psi(x) = \mathrm{d}\rho(x)/\mathrm{d}x$ is called the *influence function*. If now we define a *weight function*

$$w(x) = \frac{\psi(x)}{x}, \tag{25}$$

then Eq. (24) becomes

$$\sum_i w(r_i) r_i \frac{\partial r_i}{\partial p_j} = 0, \quad \text{for } j = 1, \ldots, m. \tag{26}$$

This is exactly the system of equations that we obtain if we solve the following iterated reweighted least-squares problem

$$\min \sum_i w(r_i^{(k-1)}) r_i^2, \tag{27}$$

where the superscript (k) indicates the iteration number. The weight $w(r_i^{(k-1)})$ should be recomputed after each iteration in order to be used in the next iteration.

The influence function $\psi(x)$ measures the influence of a datum on the value of the parameter estimate. For example, for the least-squares with $\rho(x) = x^2/2$, the influence function is $\psi(x) = x$, that is, the influence of a datum on the estimate increases linearly with the size of its error, which confirms the non-robustness of the least-squares estimate. When an estimate is robust, it may be inferred that the influence of any single observation (datum) is insufficient to yield any significant offset [17]. There are several constraints that a robust M-estimator should meet:

- The first is of course to have a bounded influence function.
- The second is naturally the requirement of the robust estimator to be unique. This implies that the objective function of parameter vector $\mathbf{p}$ to be minimized should have a unique minimum. This requires that *the individual ρ-function is* convex *in variable* $\mathbf{p}$. This is necessary because only requiring a ρ-function to have a unique minimum is not sufficient. This is the case with maxima when considering mixture distribution; the sum of unimodal probability distributions is very often multimodal. The convexity constraint is equivalent to imposing that $\partial^2 \rho(\cdot)/\partial \mathbf{p}^2$ is non-negative definite.

Table 1
A few commonly used M-estimators

Type	$\rho(x)$	$\psi(x)$	$w(x)$
L_2	$x^2/2$	x	1
L_1	$\lvert x\rvert$	$\operatorname{sgn}(x)$	$\frac{1}{\lvert x\rvert}$
$L_1 - L_2$	$2(\sqrt{1 + x^2/2} - 1)$	$\frac{x}{\sqrt{1 + x^2/2}}$	$\frac{1}{\sqrt{1 + x^2/2}}$
L_p	$\frac{\lvert x\rvert^\nu}{\nu}$	$\operatorname{sgn}(x)\lvert x\rvert^{\nu-1}$	$\lvert x\rvert^{\nu-2}$
'Fair'	$c^2\left[\frac{\lvert x\rvert}{c} - \log\left(1 + \frac{\lvert x\rvert}{c}\right)\right]$	$\frac{x}{1 + \lvert x\rvert/c}$	$\frac{1}{1 + \lvert x\rvert/c}$
Huber $\begin{cases}\text{if } \lvert x\rvert \le k \\ \text{if } \lvert x\rvert \ge k\end{cases}$	$\begin{cases}x^2/2 \\ k(\lvert x\rvert - k/2)\end{cases}$	$\begin{cases}x \\ k \operatorname{sgn}(x)\end{cases}$	$\begin{cases}1 \\ k/\lvert x\rvert\end{cases}$
Cauchy	$\frac{c^2}{2}\log(1 + (x/c)^2)$	$\frac{x}{1 + (x/c)^2}$	$\frac{1}{1 + (x/c)^2}$
Geman–McClure	$\frac{x^2/2}{1 + x^2}$	$\frac{x}{(1 + x^2)^2}$	$\frac{1}{(1 + x^2)^2}$
Welsch	$\frac{c^2}{2}[1 - \exp(-(x/c)^2)]$	$x\exp(-(x/c)^2)$	$\exp(-(x/c)^2))$
Tukey $\begin{cases}\text{if } \lvert x\rvert \le c \\ \text{if } \lvert x\rvert > c\end{cases}$	$\begin{cases}\frac{c^2}{6}(1 - [1 - (x/c)^2]^3) \\ (c^2/6)\end{cases}$	$\begin{cases}x[1 - (x/c)^2]^2 \\ 0\end{cases}$	$\begin{cases}[1 - (x/c)^2]^2 \\ 0\end{cases}$

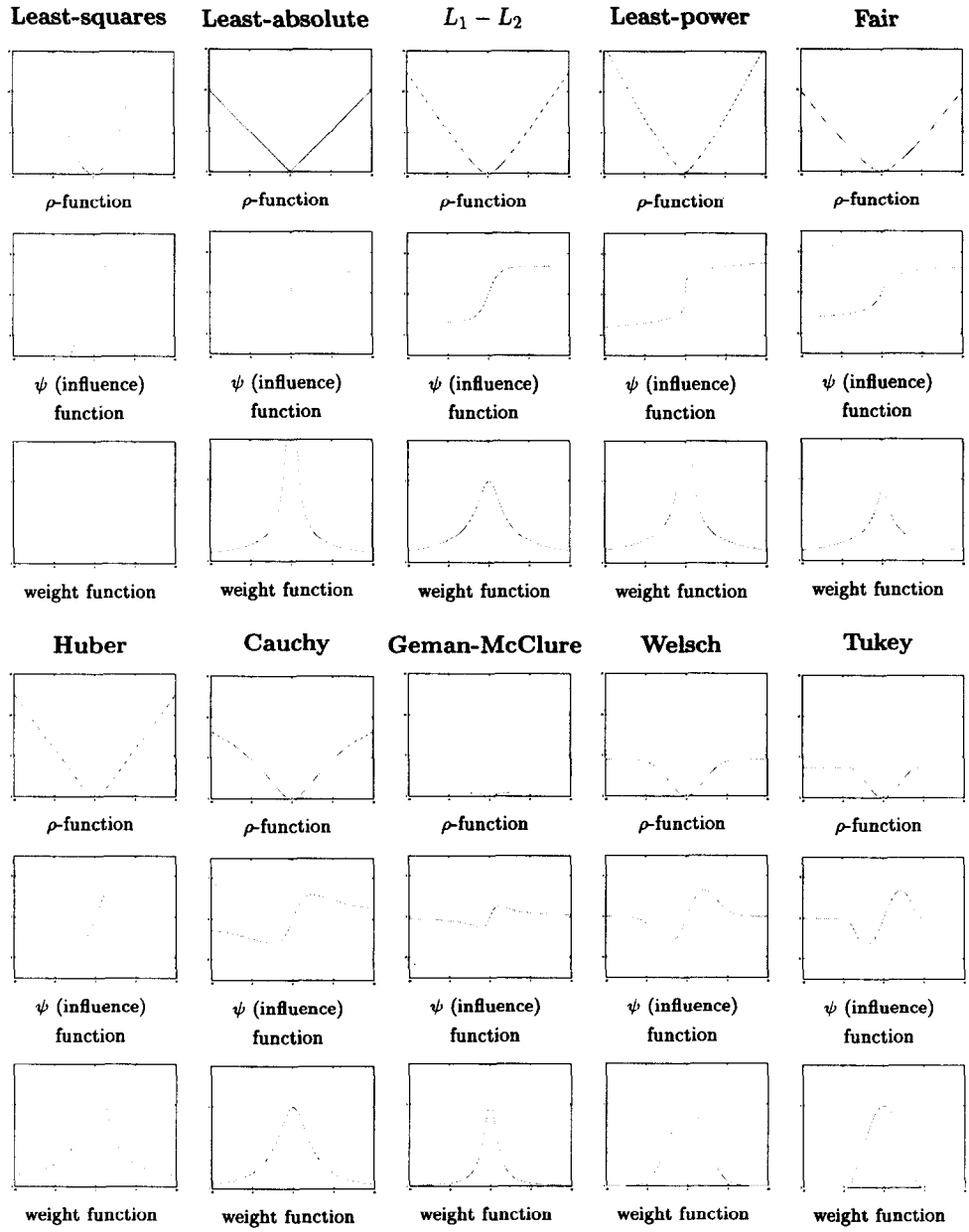

Fig. 4. Graphic representations of a few common M-estimators.

- The third one is a practical requirement. Whenever $\frac{\partial^2 \rho(\cdot)}{\partial \mathbf{p}^2}$ is singular, the objective should have a gradient, i.e. $\frac{\partial \rho(\cdot)}{\partial \mathbf{p}} \neq \mathbf{0}$. This avoids having to search through the complete parameter space.

Table 1 lists a few commonly used influence functions. They are graphically depicted in Fig. 4. Note that not all these functions satisfy the above requirements.

Before proceeding further, we define a measure called *asymptotic efficiency on the standard normal distribution*. This is the ratio of the variance-covariance matrix of a given estimator to that of the least-squares in the presence of Gaussian noise in the data. If there is no outliers and the data is only corrupted by Gaussian noise, we may expect the estimator to have an approximately normal distribution. Now, we briefly give a few indications of the above mentioned functions:

- L_2 (i.e. least-squares) estimators are not robust because their influence function is not bounded.
- L_1 (i.e. absolute value) estimators are not stable because the ρ-function $|x|$ is not strictly convex in x. Indeed, the second derivative at $x = 0$ is unbounded, and an indeterminant solution may result.
- L_1 estimators reduce the influence of large errors, but they still have an influence because the influence function has no cut off point.
- $L_1 - L_2$ estimators take both the advantage of the L_1 estimators to reduce the influence of large errors and that of L_2 estimators to be convex. They behave like L_2 for small x and like L_1 for large x, hence the name of this type of estimators.

- The L_p (*least-powers*) function represents a family of functions. It is L_2 with $\nu = 2$ and L_1 with $\nu = 1$. The smaller ν, the smaller is the incidence of large errors in the estimate $\mathbf{p}$. It appears that ν must be fairly moderate to provide a relatively robust estimator or, in other words, to provide an estimator scarcely perturbed by outlying data. The selection of an optimal ν has been investigated, and for ν around 1.2, a good estimate may be expected [17]. However, many difficulties are encountered in the computation when parameter ν is in the range of interest $1 < \nu < 2$, because zero residuals are troublesome.
- The function 'Fair' is among the possibilities offered by the Roepack package (see [17]). It has everywhere defined continuous derivatives of first three orders, and yields a unique solution. The 95% asymptotic efficiency on the standard normal distribution is obtained with the tuning constant $c = 1.3998$.
- Huber's function [24] is a parabola in the vicinity of zero, and increases linearly at a given level $|x| > k$. The 95% asymptotic efficiency on the standard normal distribution is obtained with the tuning constant $k = 1.345$. This estimator is so satisfactory that it has been recommended for almost all situations; very rarely it has been found to be inferior to some other ρ-function. However, from time to time, difficulties are encountered, which may be due to the lack of stability in the gradient values of the ρ-function because of its *discontinuous second derivative*:

$$\frac{\mathrm{d}^2\rho(x)}{\mathrm{d}x^2} = \begin{cases} 1 & \text{if } |x| \le k, \\ 0 & \text{if } |x| \ge k. \end{cases}$$

The modification proposed in [15] is the following

$$\rho(x) = \begin{cases} c^2[1 - \cos(x/c)] & \text{if } |x|/c \le \pi/2, \\ c|x| + c^2(1 - \pi/2) & \text{if } |x|/c \ge \pi/2. \end{cases}$$

The 95% asymptotic efficiency on the standard normal distribution is obtained with the tuning constant $c = 1.2107$.

- Cauchy's function, also known as the Lorentzian function, does not guarantee a unique solution. With a descending first derivative, such a function has a tendency to yield erroneous solutions in a way which cannot be observed. The 95% asymptotic efficiency on the standard normal distribution is obtained with the tuning constant $c = 2.3849$.
- The other remaining functions have the same problem as the Cauchy function. As can be seen from the influence function, the influence of large errors only decreases linearly with their size. The Geman–McClure and Welsh functions try to further reduce the effect of large errors, and the Tukey's biweight function even suppress the outliers. The 95% asymptotic efficiency on the standard normal distribution of the Tukey's biweight function is obtained with the tuning constant $c = 4.6851$; that of the Welsch function, with $c = 2.9846$.

There still exist many other ρ-functions, such as Andrew's cosine wave function. Another commonly used function is the following tri-weight one:

$$w_i = \begin{cases} 1 & |r_i| \le \sigma \\ \sigma/|r_i| & \sigma < |r_i| \le 3\sigma \\ 0 & 3\sigma < |r_i|, \end{cases}$$

where σ is some estimated standard deviation of errors.

It seems difficult to select a ρ-function for general use without being rather arbitrary. Following Rey [17], for the location (or regression) problems, the best choice is the L_p in spite of its theoretical non-robustness: they are quasi-robust. However, it suffers from its computational difficulties. The second best function is 'Fair', which can yield nicely converging computational procedures. Eventually comes the Huber's function (either original or modified form). All these functions do not eliminate completely the influence of large gross errors.

The four last functions do not guarantee unicity, but reduce considerably, or even eliminate completely, the influence of large gross errors. As proposed by Huber [24], one can start the iteration process with a convex ρ-function, iterate until convergence, and then apply a few iterations with one of those non-convex functions to eliminate the effect of large errors.

Inherent in the different M-estimators is the simultaneous estimation of σ, the standard deviation of the residual errors. If we can make a good estimate of the standard deviation of the errors of good data (inliers), then data points whose error is larger than a certain number of standard deviations can be considered as outliers. Thus, the estimation of σ itself should be robust. The results of the M-estimators will depend on the method used to compute it. The *robust standard deviation* estimate is related to the median of the absolute values of the residuals, and is given by

$$\hat{\sigma} = 1.4826[1 + 5/(n - p)] \operatorname*{median}_i |r_i|. \tag{28}$$

The constant 1.4826 is a coefficient to achieve the same efficiency as a least-squares in the presence of only Gaussian noise (actually, the median of the absolute values of random numbers sampled from the Gaussian normal distribution $N(0, 1)$ is equal to $\Phi^{-1}(\frac{3}{4}) \approx 1/1.4826$); $5/(n - p)$ (where n is the size of the data set and p is the dimension of the parameter vector) is to compensate the effect of a small set of data. The reader is referred to [25, p. 202] for the details of these magic numbers.

9.5. Least median of squares

The least-median-of-squares (LMedS) method esti-

mates the parameters by solving the nonlinear minimization problem:

$$\min_i \operatorname{med} r_i^2.$$

That is, the estimator must yield the smallest value for the median of squared residuals computed for the entire data set. It turns out that this method is very robust to gross errors as well as outliers due to bad localization. Unlike the M-estimators, however, the LMedS problem cannot be reduced to a weighted least-squares problem. It is probably impossible to write down a straightforward formula for the LMedS estimator. It must be solved by a search in the space of possible estimates generated from the data. Since this space is too large, only a randomly chosen subset of data can be analysed. The algorithm which we describe below for robustly estimating a conic follows that structured in [25, Chap. 5], as outlined below.

Given n points: $\{\mathbf{m}_i = [x_i, y_i]^T\}$:

1. A Monte Carlo type technique is used to draw m random subsamples of p different points. For the problem at hand, we select *five* (i.e. $p = 5$) points because we need at least five points to define a conic.
2. For each subsample, indexed by J, we use any of the techniques described in Section 4 to compute the conic parameters $\mathbf{p}_J$. (Which technique is used is not important because an exact solution is possible for five different points.)
3. For each $\mathbf{p}_J$, we can determine the median of the squared residuals, denoted by M_J, with respect to the whole set of points, i.e.

   $$M_J = \operatorname*{med}_{i=1,\ldots,n} r_i^2(\mathbf{p}_J, \mathbf{m}_i).$$

   Here, we have a number of choices for $r_i(\mathbf{p}_J, \mathbf{m}_i)$, the residual of the ith point with respect to the conic $\mathbf{p}_J$. Depending on the demanding precision, computation requirement, etc., one can use the algebraic distance, the Euclidean distance, or the gradient weighted distance.
4. We retain the estimate $\mathbf{p}_J$ for which M_J is minimal among all m M_J's.

The question now is: *How do we determine m*? A subsample is 'good' if it consists of p good data points. Assuming that the whole set of points may contain up to a fraction ϵ of outliers, the probability that at least one of the m subsamples is good is given by

$$P = 1 - [1 - (1 - \epsilon)^p]^m. \tag{29}$$

By requiring that P must be near 1, one can determine m for given values of p and ϵ:

$$m = \frac{\log(1 - P)}{\log[1 - (1 - \epsilon)^p]}.$$

In our implementation, we assume $\epsilon = 40\%$ and require $P = 0.99$, thus $m = 57$. Note that the algorithm can be speeded up considerably by means of parallel computing, because the processing for each subsample can be done independently.

Actually, the LMedS is philosophically very similar to RANSAC (RANdom SAmple Consensus) advocated by Bolles and Fischler [26]. The reader is referred to [27] for a discussion of their difference. A major one is that RANSAC requires a prefixed threshold to be supplied by the user to decide whether an estimated set of parameters is good enough to be accepted and end further random sampling.

As noted in [25], the LMedS *efficiency* is poor in the presence of Gaussian noise. The efficiency of a method is defined as the ratio between the lowest achievable variance for the estimated parameters and the actual variance provided by the given method. To compensate for this deficiency, we further carry out a weighted least-squares procedure. The *robust standard deviation* estimate is given by Eq. (28), that is:

$$\hat{\sigma} = 1.4826[1 + 5/(n - p)]\sqrt{M_J},$$

where M_J is the minimal median. Based on $\hat{\sigma}$, we can assign a weight for each correspondence:

$$w_i = \begin{cases} 1 & \text{if } r_i^2 \le (2.5\hat{\sigma})^2 \\ 0 & \text{otherwise,} \end{cases}$$

where r_i is the residual of the ith point with respect to the conic $\mathbf{p}$. The data points having $w_i = 0$ are outliers and should not be further taken into account. The conic $\mathbf{p}$ is finally estimated by solving the weighted least-squares problem:

$$\min_{\mathbf{p}} \sum_i w_i r_i^2$$

using one of the numerous techniques described before. We have thus robustly estimated the conic, because outliers have been detected and discarded by the LMedS method.

As said previously, computational efficiency of the LMedS method can be achieved by applying a Monte Carlo type technique. However, the five points of a subsample thus generated may be very close to each other. Such a situation should be avoided because the estimation of the conic from such points is highly instable and the result is useless. It is a waste of time to evaluate such a subsample. To achieve higher stability and efficiency, we develop a *regularly random selection method* based on bucketing techniques [28]. The idea is not to select more than one point for a given neighborhood. The procedure of generating random numbers should be modified accordingly.

We have applied this technique to matching between two uncalibrated images [28]. Given two uncalibrated images, the only available geometric constraint is the

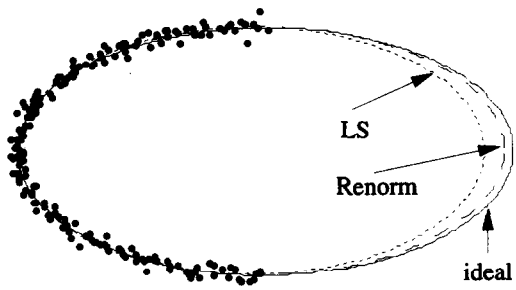

Fig. 5. Comparison between Linear Least-Squares (*LS*, in dotted lines) and Renormalization technique (*Renorm*, in dashed lines).

epipolar constraint. The idea underlying our approach is to use classical techniques (correlation and relaxation methods in our particular implementation) to find an initial set of matches, and then use the Least Median of Squares (LMedS) to discard false matches in this set. the epipolar geometry can then be accurately estimated using a meaningful image criterion. More matches are eventually found, as in stereo matching, by using the recovered epipolar geometry.

10. Two examples

This section provides two examples of conic estimation in order to give the reader an idea of the performance of each technique.

10.1. Noisy data without outliers

The ideal ellipse used in our experiment has the following parameters: the long axis is equal to 100 pixels, the short axis is equal to 50 pixels, the center is at (250, 250), and the angle between the long axis and the horizontal axis is 0°. The left half section of the ellipse is used and is sampled by 181 points. Finally, a Gaussian noise with 2 pixels of standard deviation is added to each sample points, as shown by black dots in Figs. 5 and 6, where the ideal ellipse is also shown in solid lines.

The linear least-squares technique, the renormalization technique, and the technique based on orthogonal distances between points and ellipse are applied to the noisy data. A visual comparison of the results is shown in Figs. 5 and 6. A quantitative comparison of the estimated parameters is described in Table 2. As can be observed, the linear technique introduces a significant bias (this is more prominent if a shorter section of ellipse is used). The renormalization technique corrects the bias and improves considerably the result (however, because of the underlying statistical assumption, this technique does not work as well if only a small set of data is available). The technique based on the minimization of orthogonal distances definitely gives the best result.

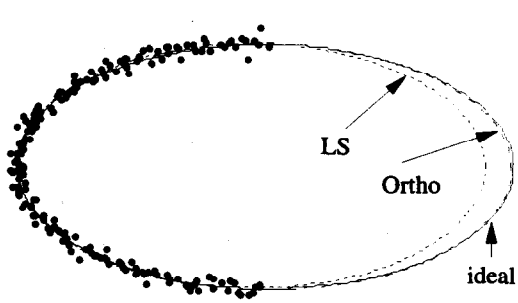

Fig. 6. Comparison between Linear Least-Squares (*LS*) and Orthogonal Regression (*Ortho*).

Table 2
Comparison of the estimated parameters by different techniques

Technique	Long axis	Short axis	Center position	Orientation
(ideal)	100.00	50.00	(250.00, 250.00)	0.00
Least-squares	94.08	49.96	(244.26, 250.25)	−0.05
Renormalization	98.15	50.10	(248.23, 250.25)	0.04
Orthogonal	99.76	50.26	(249.37, 249.92)	−0.37

Table 3 shows roughly the computation time required for each method. *Eigen least-squares* yields similar result to that of *Linear least-squares*. *Weighted least-squares* improves the result a little bit, but not as much as *Renormalization* for this set of data. *Extended Kalman filter* or *Iterated EKF* require a quite good initial estimate.

10.2. Noisy data with outliers

We now introduce outliers in the above set of noisy data points. The outliers are generated by a uniform distributed noise in an rectangle whose lower left corner is at (140, 195) and whose upper right corner is at (250, 305). The complete set of data points are shown as black dots in Fig. 7.

In Fig. 7, we show the ellipse estimated by the linear least-squares without outliers rejection (*LS*) (shown in dotted lines) which is definitely very bad, the ellipse estimated by the least-median-squares without refinement using the weighted least-squares (*LMedS*) (shown in thin dashed lines) which is quite reasonable, and the ellipse estimated by the least-median-squares continued by the weighted least-squares based on orthogonal distances (*LMedS-complete*) (shown in thick dashed lines)

Table 3
Comparison of computation time of different techniques: real time on a Sun Sparc 20 workstation (in seconds)

Technique	Time
Linear least-squares	0.006
Eigen least-squares	0.005
Weighted least-squares	0.059
Renormalization	0.029
Extended Kalman filter	0.018
Iterated EKF (5 iterations)	0.066
Orthogonal	3.009

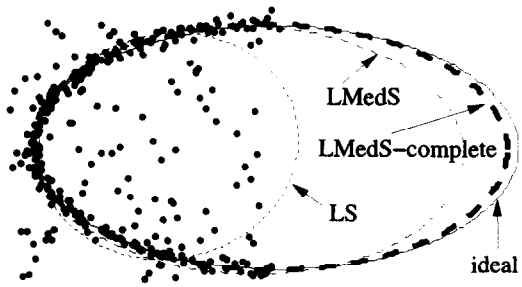

Fig. 7. Comparison of different techniques in the presence of outliers in data.

deviate significantly from the true estimate, diagnostics or M-estimators are useful; otherwise, the least-median-of-squares technique is probably the best we can recommend.

Another technique, which I consider to be very important and which is now becoming popular in computer vision, is the *Minimum Description Length* (MDL) principle. However, since I have not yet myself applied it to solve any problem, I am not in a position to present it. The reader is referred elsewhere [29–31].

Table 4
Comparison of the estimated parameters by non-robust and robust techniques

Technique	Long axis	Short axis	Center position	Orientation
(ideal)	100.00	50.00	(250.00, 250.00)	0.00
Least-squares	53.92	45.98	(205.75, 250.33)	12.65
LMedS	88.84	50.85	(238.60, 248.88)	−1.07
LMedS-complete	97.89	50.31	(247.40, 250.15)	−0.33

which is very close to the ideal ellipse. The ellipse parameters estimated by these techniques are compared in Table 4.

The computation time on a Sun Sparc 20 workstation is respectively 0.008 seconds, 2.56 seconds and 6.93 seconds for linear least-square, LMedS and LMedS-complete. Bear in mind that the time given here is only for illustration, since I did not try to do any code optimization. The most time-consuming part is the computation of the orthogonal distance between each point and the ellipse.

11. Conclusions

In this tutorial, I have presented what are probably the most commonly used techniques for parameter estimation in computer vision. Particular attention has been devoted to discussions about the choice of appropriate minimization criteria and the robustness of the different techniques. Use of algebraic distances usually leads to a closed-form solution, but there is no justification from either physical viewpoint or statistical one. Orthogonal least-squares has a much sounder basis, but is usually difficult to implement. Gradient weighted least-squares provides a nice approximation to orthogonal least-squares. If the size of data set is large, renormalization techniques can be used to correct the bias in the estimate. If data are available in a serial manner, the Kalman filtering technique can be applied, which has an advantage of explicitly taking the data uncertainty into account. If there are outliers in the data set, robust techniques must be used: if the number of parameters to be estimated is small, clustering or Hough transform can be used; if the number of outliers is small and they do not

Acknowledgements

The author thanks Théo Papadopoulo for discussions, and anonymous reviewers for careful reading and constructive suggestions.

References

[1] N. Ayache, Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception, MIT Press, Cambridge, MA, 1991.
[2] W. Förstner, Reliability analysis of parameter estimation in linear models with application to mensuration problems in computer vision, Comput. Vision, Graphics and Image Processing, 40 (1987) 273–310.
[3] J.V. Beck and K.J. Arnold, Parameter Estimation in Engineering and Science, Wiley series in probability and mathematical statistics, Wiley, New York, 1977.
[4] J. Porrill, Fitting ellipses and predicting confidence envelopes using a bias corrected kalman filter, Image and Vision Computing, 8(1) (1990) 37–41.
[5] P.L. Rosin and G.A.W. West, Segmenting curves into elliptic arcs and straight lines, in Proc. Third Int. Conf. Comput. Vision, Osaka, Japan, 1990 75–78.
[6] P.L. Rosin, A note on the least squares fitting of ellipses, Pattern Recognition Letters, 14 (1993) 799–808.
[7] F. Bookstein, Fitting conic sections to scattered data, Computer Vision, Graphics, and Image Processing, 9 (1979) 56–71.
[8] K. Kanatani, Geometric Computation for Machine Vision, Oxford University Press, Oxford, 1993.
[9] K. Kanatani, Renormalization for unbiased estimation, in Proc. Fourth Int. Conf. Comput. Vision, Berlin, 1993, pp. 599–606.
[10] Y.T. Chan and S.M. Thomas, Cramer-Rao lower bounds for estimation of a circular arc center and its radius, Graphical Models and Image Processing, 57(6) (1995) 527–532.
[11] D.G. Lowe, Review of 'TINA: The Sheffield AIVRU vision system' by J. Porrill et al., in O. Khatib, J. Craig and T. Lozano-Pérez (eds), The Robotics Review I, MIT Press, Cambridge, MA, 1989, pp. 195–198.

[12] Z. Zhang and O. Faugeras, 3D Dynamic Scene Analysis: A Stereo Based Approach, Springer, Berlin, 1992.
[13] A.M. Jazwinsky, Stochastic Processes and Filtering Theory, Academic, New York, 1970.
[14] P.S. Maybeck, Stochastic Models, Estimation and Control, vol. 1, Academic, New York, 1979.
[15] C.K. Chui and G. Chen, Kalman Filtering with Real-Time Applications, Springer Ser. Info. Sci., Vol. 17. Springer, Berlin, 1987.
[16] F.R. Hampel, Robust estimation: A condensed partial survey, Z. Wahrscheinlichkeitstheorie Verw. Gebiete, 27 (1973) 87–104.
[17] W.J.J. Rey, Introduction to Robust and Quasi-Robust Statistical Methods, Springer, Berlin, 1983.
[18] D.H. Ballard and C.M. Brown, Computer Vision, Prentice-Hall, Englewood Cliffs, NJ, 1982.
[19] J. Illingworth and J. Kittler, A survey of the Hough transform, Comput. Vision Graph. Image Process. 44 (1988) 87–116.
[20] P.A. Devijver and J. Kittler, Pattern Recognition: A Statistical Approach, Prentice Hall, Englewood-Cliffs, 1982.
[21] D.A. Belsley, E. Kuh and R.E. Welsch, Regression Diagnostics, John Wiley, New York, 1980.
[22] V. Barnett and T. Lewis, Outliers in Statistical Data, 2nd ed., John Wiley, Chichester, 1984.
[23] L.S. Shapiro, Affine analysis of image sequences, PhD thesis, Department of Engineering Science, Oxford University, 1993.
[24] P.J. Huber, Robust Statistics, John Wiley, New York, 1981.
[25] P.J. Rousseeuw and A.M. Leroy, Robust Regression and Outlier Detection, John Wiley, New York, 1987.
[26] R.C. Bolles and M.A. Fischler, A RANSAC-based approach to model fitting and its application to finding cylinders in range data, in Proc. Int. Joint Conf. Artif. Intell., Vancouver, Canada, 1981, pp. 637–643.
[27] P. Meer, D. Mintz, A. Rosenfeld and D.Y. Kim, Robust regression methods for computer vision: A review, Int. J. Computer Vision, 6(1) (1991) 59–70.
[28] Z. Zhang, R. Deriche, O. Faugeras and Q.-T. Luong, A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry, Artificial Intelligence Journal, 78 (October 1995) 87–119. (Also INRIA Research Report No. 2273, May 1994.)
[29] J. Rissanen, Minimum description length principle, Encyclopedia of Statistic Sciences, 5 (1987) 523–527.
[30] Y.G. Leclerc, Constructing simple stable description for image partitioning, Int. J. Computer Vision, 3(1) (1989) 73–102.
[31] M. Li, Minimum description length based 2D shape description, in Proc. 4th International Conferernce on Computer Vision, Berlin, Germany, May 1993, pp. 512–517.