4

Estimation – 2D Projective Transformations

In this chapter, we consider the problem of estimation. In the present context this will be taken to mean the computation of some transformation or other mathematical quantity, based on measurements of some nature. This definition is somewhat vague, so to make it more concrete, here are a number of estimation problems of the type that we would like to consider.

(i) **2D homography.** Given a set of points $\mathbf{x}_i$ in $\mathbb{P}^2$ and a corresponding set of points $\mathbf{x}'_i$ likewise in $\mathbb{P}^2$, compute the projective transformation that takes each $\mathbf{x}_i$ to $\mathbf{x}'_i$. In a practical situation, the points $\mathbf{x}_i$ and $\mathbf{x}'_i$ are points in two images (or the same image), each image being considered as a projective plane $\mathbb{P}^2$.

(ii) **3D to 2D camera projection.** Given a set of points $\mathbf{X}_i$ in 3D space, and a set of corresponding points $\mathbf{x}_i$ in an image, find the 3D to 2D projective mapping that maps $\mathbf{X}_i$ to $\mathbf{x}_i$. Such a 3D to 2D projection is the mapping carried out by a projective camera, as discussed in chapter 6.

(iii) **Fundamental matrix computation.** Given a set of points $\mathbf{x}_i$ in one image, and corresponding points $\mathbf{x}'_i$ in another image, compute the fundamental matrix $\mathtt{F}$ consistent with these correspondences. The fundamental matrix, discussed in chapter 9, is a singular 3×3 matrix $\mathtt{F}$ satisfying $\mathbf{x}_i'^{\mathsf{T}}\mathtt{F}\mathbf{x}_i = 0$ for all i.

(iv) **Trifocal tensor computation.** Given a set of point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i \leftrightarrow \mathbf{x}''_i$ across three images, compute the trifocal tensor. The trifocal tensor, discussed in chapter 15, is a tensor $\mathcal{T}_i^{jk}$ relating points or lines in three views.

These problems have many features in common, and the considerations that relate to one of the problems are also relevant to each of the others. Therefore, in this chapter, the first of these problems will be considered in detail. What we learn about ways of solving this problem will teach us how to proceed in solving each of the other problems as well.

Apart from being important for illustrative purposes, the problem of estimating 2D projective transformations is of importance in its own right. We consider a set of point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ between two images. Our problem is to compute a 3×3 matrix $\mathtt{H}$ such that $\mathtt{H}\mathbf{x}_i = \mathbf{x}'_i$ for each i.

Number of measurements required. The first question to consider is how many corresponding points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are required to compute the projective transformation $\mathtt{H}$. A lower bound is available by a consideration of the number of degrees of freedom and number of constraints. On the one hand, the matrix $\mathtt{H}$ contains 9 entries, but is defined only up to scale. Thus, the total number of degrees of freedom in a 2D projective transformation is 8. On the other hand, each point-to-point correspondence accounts for two constraints, since for each point $\mathbf{x}_i$ in the first image the two degrees of freedom of the point in the second image must correspond to the mapped point $\mathtt{H}\mathbf{x}_i$. A 2D point has two degrees of freedom corresponding to the x and y components, each of which may be specified separately. Alternatively, the point is specified as a homogeneous 3-vector, which also has two degrees of freedom since scale is arbitrary. As a consequence, it is necessary to specify four point correspondences in order to constrain $\mathtt{H}$ fully.

Approximate solutions. It will be seen that if exactly four correspondences are given, then an exact solution for the matrix $\mathtt{H}$ is possible. This is the *minimal* solution. Such solutions are important as they define the size of the subsets required in robust estimation algorithms, such as RANSAC, described in section 4.7. However, since points are measured inexactly ("noise"), if more than four such correspondences are given, then these correspondences may not be fully compatible with any projective transformation, and one will be faced with the task of determining the "best" transformation given the data. This will generally be done by finding the transformation $\mathtt{H}$ that minimizes some cost function. Different cost functions will be discussed during this chapter, together with methods for minimizing them. There are two main categories of cost function: those based on minimizing an algebraic error; and those based on minimizing a geometric or statistical image distance. These two categories are described in section 4.2.

The Gold Standard algorithm. There will usually be one cost function which is optimal in the sense that the $\mathtt{H}$ that minimizes it gives the best possible estimate of the transformation under certain assumptions. The computational algorithm that enables this cost function to be minimized is called the "Gold Standard" algorithm. The results of other algorithms are assessed by how well they compare to this Gold Standard. In the case of estimating a homography between two views the cost function is (4.8), the assumptions for optimality are given in section 4.3, and the Gold Standard is algorithm 4.3(*p*114).

4.1 The Direct Linear Transformation (DLT) algorithm

We begin with a simple linear algorithm for determining $\mathtt{H}$ given a set of four 2D to 2D point correspondences, $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. The transformation is given by the equation $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$. Note that this is an equation involving homogeneous vectors; thus the 3-vectors $\mathbf{x}'_i$ and $\mathtt{H}\mathbf{x}_i$ are not equal, they have the same direction but may differ in magnitude by a non-zero scale factor. The equation may be expressed in terms of the vector cross product as $\mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i = \mathbf{0}$. This form will enable a simple linear solution for $\mathtt{H}$ to be derived.

If the j-th row of the matrix $\mathtt{H}$ is denoted by $\mathbf{h}^{j\mathsf{T}}$, then we may write

$$\mathtt{H}\mathbf{x}_i = \begin{pmatrix} \mathbf{h}^{1\mathsf{T}}\mathbf{x}_i \\ \mathbf{h}^{2\mathsf{T}}\mathbf{x}_i \\ \mathbf{h}^{3\mathsf{T}}\mathbf{x}_i \end{pmatrix}.$$

Writing $\mathbf{x}'_i = (x'_i, y'_i, w'_i)^\mathsf{T}$, the cross product may then be given explicitly as

$$\mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i = \begin{pmatrix} y'_i\mathbf{h}^{3\mathsf{T}}\mathbf{x}_i - w'_i\mathbf{h}^{2\mathsf{T}}\mathbf{x}_i \\ w'_i\mathbf{h}^{1\mathsf{T}}\mathbf{x}_i - x'_i\mathbf{h}^{3\mathsf{T}}\mathbf{x}_i \\ x'_i\mathbf{h}^{2\mathsf{T}}\mathbf{x}_i - y'_i\mathbf{h}^{1\mathsf{T}}\mathbf{x}_i \end{pmatrix}.$$

Since $\mathbf{h}^{j\mathsf{T}}\mathbf{x}_i = \mathbf{x}_i^\mathsf{T}\mathbf{h}^j$ for $j = 1, \ldots, 3$, this gives a set of three equations in the entries of $\mathtt{H}$, which may be written in the form

$$\begin{bmatrix} \mathbf{0}^\mathsf{T} & -w'_i\mathbf{x}_i^\mathsf{T} & y'_i\mathbf{x}_i^\mathsf{T} \\ w'_i\mathbf{x}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} & -x'_i\mathbf{x}_i^\mathsf{T} \\ -y'_i\mathbf{x}_i^\mathsf{T} & x'_i\mathbf{x}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \tag{4.1}$$

These equations have the form $\mathtt{A}_i\mathbf{h} = \mathbf{0}$, where $\mathtt{A}_i$ is a 3×9 matrix, and $\mathbf{h}$ is a 9-vector made up of the entries of the matrix $\mathtt{H}$,

$$\mathbf{h} = \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix}, \qquad \mathtt{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \tag{4.2}$$

with h_i the $i-$th element of $\mathbf{h}$. Three remarks regarding these equations are in order here.

(i) The equation $\mathtt{A}_i\mathbf{h} = \mathbf{0}$ is an equation *linear* in the unknown $\mathbf{h}$. The matrix elements of $\mathtt{A}_i$ are quadratic in the known coordinates of the points.

(ii) Although there are three equations in (4.1), only two of them are linearly independent (since the third row is obtained, up to scale, from the sum of x'_i times the first row and y'_i times the second). Thus each point correspondence gives two equations in the entries of $\mathtt{H}$. It is usual to omit the third equation in solving for $\mathtt{H}$ ([Sutherland-63]). Then (for future reference) the set of equations becomes

$$\begin{bmatrix} \mathbf{0}^\mathsf{T} & -w'_i\mathbf{x}_i^\mathsf{T} & y'_i\mathbf{x}_i^\mathsf{T} \\ w'_i\mathbf{x}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} & -x'_i\mathbf{x}_i^\mathsf{T} \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \tag{4.3}$$

This will be written

$$\mathtt{A}_i\mathbf{h} = \mathbf{0}$$

where $\mathtt{A}_i$ is now the 2×9 matrix of (4.3).

(iii) The equations hold for any homogeneous coordinate representation $(x'_i, y'_i, w'_i)^\mathsf{T}$ of the point $\mathbf{x}'_i$. One may choose $w'_i = 1$, which means that (x'_i, y'_i) are the coordinates measured in the image. Other choices are possible, however, as will be seen later.

Solving for $\mathtt{H}$

Each point correspondence gives rise to two independent equations in the entries of $\mathtt{H}$. Given a set of four such point correspondences, we obtain a set of equations $\mathtt{A}\mathbf{h} = \mathbf{0}$, where $\mathtt{A}$ is the matrix of equation coefficients built from the matrix rows $\mathtt{A}_i$ contributed from each correspondence, and $\mathbf{h}$ is the vector of unknown entries of $\mathtt{H}$. We seek a non-zero solution $\mathbf{h}$, since the obvious solution $\mathbf{h} = \mathbf{0}$ is of no interest to us. If (4.1) is used then $\mathtt{A}$ has dimension 12×9, and if (4.3) the dimension is 8×9. In either case $\mathtt{A}$ has rank 8, and thus has a 1-dimensional null-space which provides a solution for $\mathbf{h}$. Such a solution $\mathbf{h}$ can only be determined up to a non-zero scale factor. However, $\mathtt{H}$ is in general only determined up to scale, so the solution $\mathbf{h}$ gives the required $\mathtt{H}$. A scale may be arbitrarily chosen for $\mathbf{h}$ by a requirement on its norm such as $\|\mathbf{h}\| = 1$.

4.1.1 Over-determined solution

If more than four point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are given, then the set of equations $\mathtt{A}\mathbf{h} = \mathbf{0}$ derived from (4.3) is over-determined. If the position of the points is exact then the matrix $\mathtt{A}$ will still have rank 8, a one dimensional null-space, and there is an exact solution for $\mathbf{h}$. This will not be the case if the measurement of image coordinates is inexact (generally termed *noise*) – there will not be an exact solution to the over-determined system $\mathtt{A}\mathbf{h} = \mathbf{0}$ apart from the zero solution. Instead of demanding an exact solution, one attempts to find an approximate solution, namely a vector $\mathbf{h}$ that minimizes a suitable cost function. The question that naturally arises then is: what should be minimized? Clearly, to avoid the solution $\mathbf{h} = \mathbf{0}$ an additional constraint is required. Generally, a condition on the norm is used, such as $\|\mathbf{h}\| = 1$. The value of the norm is unimportant since $\mathtt{H}$ is only defined up to scale. Given that there is no exact solution to $\mathtt{A}\mathbf{h} = \mathbf{0}$, it seems natural to attempt to minimize the norm $\|\mathtt{A}\mathbf{h}\|$ instead, subject to the usual constraint, $\|\mathbf{h}\| = 1$. This is identical to the problem of finding the minimum of the quotient $\|\mathtt{A}\mathbf{h}\|/\|\mathbf{h}\|$. As shown in section A5.3(*p*592) the solution is the (unit) eigenvector of $\mathtt{A}^\mathsf{T}\mathtt{A}$ with least eigenvalue. Equivalently, the solution is the unit singular vector corresponding to the smallest singular value of $\mathtt{A}$. The resulting algorithm, known as the basic DLT algorithm, is summarized in algorithm 4.1.

4.1.2 Inhomogeneous solution

An alternative to solving for $\mathbf{h}$ directly as a homogeneous vector is to turn the set of equations (4.3) into a inhomogeneous set of linear equations by imposing a condition $h_j = 1$ for some entry of the vector $\mathbf{h}$. Imposing the condition $h_j = 1$ is justified by the observation that the solution is determined only up to scale, and this scale can be chosen such that $h_j = 1$. For example, if the last element of $\mathbf{h}$, which corresponds to $\mathtt{H}_{33}$, is chosen as unity then the resulting equations derived from (4.3) are

$$\begin{bmatrix} 0 & 0 & 0 & -x_i w'_i & -y_i w'_i & -w_i w'_i & x_i y'_i & y_i y'_i \\ x_i w'_i & y_i w'_i & w_i w'_i & 0 & 0 & 0 & -x_i x'_i & -y_i x'_i \end{bmatrix} \tilde{\mathbf{h}} = \begin{pmatrix} -w_i y'_i \\ w_i x'_i \end{pmatrix}$$

where $\tilde{\mathbf{h}}$ is an 8-vector consisting of the first 8 components of $\mathbf{h}$. Concatenating the equations from four correspondences then generates a matrix equation of the form

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the 2D homography matrix $\mathtt{H}$ such that $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$.

Algorithm

(i) For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ compute the matrix $\mathtt{A}_i$ from (4.1). Only the first two rows need be used in general.

(ii) Assemble the n 2×9 matrices $\mathtt{A}_i$ into a single $2n \times 9$ matrix $\mathtt{A}$.

(iii) Obtain the SVD of $\mathtt{A}$ (section A4.4(*p*585)). The unit singular vector corresponding to the smallest singular value is the solution $\mathbf{h}$. Specifically, if $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^\mathsf{T}$ with $\mathtt{D}$ diagonal with positive diagonal entries, arranged in descending order down the diagonal, then $\mathbf{h}$ is the last column of $\mathtt{V}$.

(iv) The matrix $\mathtt{H}$ is determined from $\mathbf{h}$ as in (4.2).

Algorithm 4.1. *The basic DLT for* $\mathtt{H}$ *(but see algorithm 4.2(*p*109) which includes normalization).*

$\mathtt{M}\tilde{\mathbf{h}} = \mathbf{b}$, where $\mathtt{M}$ has 8 columns and $\mathbf{b}$ is an 8-vector. Such an equation may be solved for $\tilde{\mathbf{h}}$ using standard techniques for solving linear equations (such as Gaussian elimination) in the case where $\mathtt{M}$ contains just 8 rows (the minimum case), or by least-squares techniques (section A5.1(*p*588)) in the case of an over-determined set of equations.

However, if in fact $h_j = 0$ is the true solution, then no multiplicative scale k can exist such that $kh_j = 1$. This means that the true solution cannot be reached. For this reason, this method can be expected to lead to unstable results in the case where the chosen h_j is close to zero. Consequently, this method is *not* recommended in general.

Example 4.1. It will be shown that $h_9 = \mathtt{H}_{33}$ is zero if the coordinate origin is mapped to a point at infinity by $\mathtt{H}$. Since $(0,0,1)^\mathsf{T}$ represents the coordinate origin $\mathbf{x}_0$, and also $(0,0,1)^\mathsf{T}$ represents the line at infinity $\mathbf{l}$, this condition may be written as $\mathbf{l}^\mathsf{T}\mathtt{H}\mathbf{x}_0 = (0,0,1)\mathtt{H}(0,0,1)^\mathsf{T} = 0$, thus $\mathtt{H}_{33} = 0$. In a perspective image of a scene plane the line at infinity is imaged as the vanishing line of the plane (see chapter 8), for example the horizon is the vanishing line of the ground plane. It is not uncommon for the horizon to pass through the image centre, and for the coordinate origin to coincide with the image centre. In this case the mapping that takes the image to the world plane maps the origin to the line at infinity, so that the true solution has $\mathtt{H}_{33} = h_9 = 0$. Consequently, an $h_9 = 1$ normalization can be a serious failing in practical situations. $\triangle$

4.1.3 Degenerate configurations

Consider a minimal solution in which a homography is computed using four point correspondences, and suppose that three of the points $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ are collinear. The question is whether this is significant. If the corresponding points $\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3$ are also collinear then one might suspect that the homography is not sufficiently constrained, and there will exist a family of homographies mapping $\mathbf{x}_i$ to $\mathbf{x}'_i$. On the other hand, if the corresponding points $\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3$ are not collinear then clearly there can be no transformation $\mathtt{H}$ taking $\mathbf{x}_i$ to $\mathbf{x}'_i$, since a projective transformation must preserve collinearity. Never-

theless the set of eight homogeneous equations derived from (4.3) must have a non-zero solution, giving rise to a matrix H. How is this apparent contradiction to be resolved?

The equations (4.3) express the condition that $\mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i = \mathbf{0}$ for $i = 1, \ldots, 4$, and so the matrix H found by solving the system of 8 equations will satisfy this condition. Suppose that $\mathbf{x}_1, \ldots, \mathbf{x}_3$ are collinear and let $\mathbf{l}$ be the line that they lie on, so that $\mathbf{l}^\mathsf{T}\mathbf{x}_i = 0$ for $i = 1, \ldots, 3$. Now define $\mathtt{H}^* = \mathbf{x}'_4\mathbf{l}^\mathsf{T}$, which is a 3×3 matrix of rank 1. In this case, one verifies that $\mathtt{H}^*\mathbf{x}_i = \mathbf{x}'_4(\mathbf{l}^\mathsf{T}\mathbf{x}_i) = \mathbf{0}$ for $i = 1, \ldots, 3$, since $\mathbf{l}^\mathsf{T}\mathbf{x}_i = 0$. On the other hand, $\mathtt{H}^*\mathbf{x}_4 = \mathbf{x}'_4(\mathbf{l}^\mathsf{T}\mathbf{x}_4) = k\mathbf{x}'_4$. Therefore the condition $\mathbf{x}'_i \times \mathtt{H}^*\mathbf{x}_i = \mathbf{0}$ is satisfied for all i. Note that the vector $\mathbf{h}^*$ corresponding to $\mathtt{H}^*$ is given by $\mathbf{h}^{*\mathsf{T}} = (x_4\mathbf{l}^\mathsf{T}, y_4\mathbf{l}^\mathsf{T}, w_4\mathbf{l}^\mathsf{T})$, and one easily verifies that this vector satisfies (4.3) for all i. The problem with this solution for $\mathtt{H}^*$ is that $\mathtt{H}^*$ is a rank 1 matrix and hence does not represent a projective transformation. As a consequence the points $\mathtt{H}^*\mathbf{x}_i = \mathbf{0}$ for $i = 1, \ldots, 3$ are not well defined.

We showed that if $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ are collinear then $\mathtt{H}^* = \mathbf{x}'_4\mathbf{l}^\mathsf{T}$ is a solution to (4.1). There are two cases: either $\mathtt{H}^*$ is the unique solution (up to scale) or there is a further solution H. In the first case, since $\mathtt{H}^*$ is a singular matrix, there exists no transformation taking each $\mathbf{x}_i$ to $\mathbf{x}'_i$. This occurs when $\mathbf{x}_1, \ldots, \mathbf{x}_3$ are collinear but $\mathbf{x}'_1, \ldots, \mathbf{x}'_3$ are not. In the second case, where a further solution H exists, then any matrix of the form $\alpha\,\mathtt{H}^* + \beta\,\mathtt{H}$ is a solution. Thus a 2-dimensional family of transformations exist, and it follows that the 8 equations derived from (4.3) are not independent.

A situation where a configuration does not determine a unique solution for a particular class of transformation is termed *degenerate*. Note that the definition of degeneracy involves both the configuration and the type of transformation. The degeneracy problem is not limited to a minimal solution, however. If additional (perfect, i.e. error-free) correspondences are supplied which are also collinear (lie on $\mathbf{l}$), then the degeneracy is not resolved.

4.1.4 Solutions from lines and other entities

The development to this point, and for the rest of the chapter, is exclusively in terms of computing homographies from point correspondences. However, an identical development can be given for computing homographies from line correspondences. Starting from the line transformation $\mathbf{l}_i = \mathtt{H}^\mathsf{T}\mathbf{l}'_i$, a matrix equation of the form $\mathtt{A}\mathbf{h} = \mathbf{0}$ can be derived, with a minimal solution requiring four lines in general position. Similarly, a homography may be computed from conic correspondences and so forth.

There is the question then of how many correspondences are required to compute the homography (or any other relation). The general rule is that the number of constraints must equal or exceed the number of degrees of freedom of the transformation. For example, in 2D each corresponding point or line generates two constraints on H, in 3D each corresponding point or plane generates three constraints. Thus in 2D the correspondence of four points or four lines is sufficient to compute H, since $4 \times 2 = 8$, with 8 the number of degrees of freedom of the homography. In 3D a homography has 15 degrees of freedom, and five points or five planes are required. For a planar affine transformation (6 dof) only three corresponding points or lines are required, and so on. A conic provides five constraints on a 2D homography.

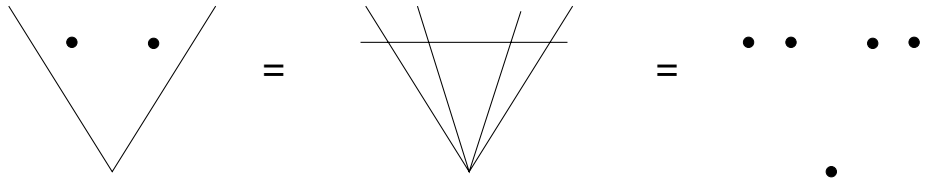


Fig. 4.1. **Geometric equivalence of point–line configurations.** *A configuration of two points and two lines is equivalent to five lines with four concurrent, or five points with four collinear.*

Care has to be taken when computing H from correspondences of mixed type. For example, a 2D homography cannot be determined uniquely from the correspondences of two points and two lines, but can from three points and one line or one point and three lines, even though in each case the configuration has 8 degrees of freedom. The case of three lines and one point is geometrically equivalent to four points, since the three lines define a triangle and the vertices of the triangle uniquely define three points. We have seen that the correspondence of four points in general position uniquely determines a homography, which means that the correspondence of three lines and one point also uniquely determines a homography. Similarly the case of three points and a line is equivalent to four lines, and again the correspondence of four lines in general position (i.e. no three concurrent) uniquely determines a homography. However, as a quick sketch shows (figure 4.1), the case of two points and two lines is equivalent to five lines with four concurrent, or five points with four collinear. As shown in the previous section, this configuration is degenerate and a one-parameter family of homographies map the two-point and two-line configuration to the corresponding configuration.

4.2 Different cost functions

We will now describe a number of cost functions which may be minimized in order to determine H for over-determined solutions. Methods of minimizing these functions are described later in the chapter.

4.2.1 Algebraic distance

The DLT algorithm minimizes the norm $\|\mathtt{A}\mathbf{h}\|$. The vector $\boldsymbol{\epsilon} = \mathtt{A}\mathbf{h}$ is called the residual vector and it is the norm of this error vector that is minimized. The components of this vector arise from the individual correspondences that generate each row of the matrix A. Each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ contributes a partial error vector $\boldsymbol{\epsilon}_i$ from (4.1) or (4.3) towards the full error vector $\boldsymbol{\epsilon}$. This vector $\boldsymbol{\epsilon}_i$ is the *algebraic error vector* associated with the point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ and the homography H. The norm of this vector is a scalar which is called the *algebraic distance*:

$$d_{\text{alg}}(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2 = \|\boldsymbol{\epsilon}_i\|^2 = \left\| \begin{bmatrix} \mathbf{0}^\mathsf{T} & -w'_i\mathbf{x}_i^\mathsf{T} & y'_i\mathbf{x}_i^\mathsf{T} \\ w'_i\mathbf{x}_i^\mathsf{T} & \mathbf{0}^\mathsf{T} & -x'_i\mathbf{x}_i^\mathsf{T} \end{bmatrix} \mathbf{h} \right\|^2 . \tag{4.4}$$

More generally, and briefly, for any two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ we may write

$$d_{\text{alg}}(\mathbf{x}_1, \mathbf{x}_2)^2 = a_1^2 + a_2^2 \text{ where } \mathbf{a} = (a_1, a_2, a_3)^\mathsf{T} = \mathbf{x}_1 \times \mathbf{x}_2.$$

The relation of this distance to a geometric distance is described in section 4.2.4.

Given a set of correspondences, the quantity $\boldsymbol{\epsilon} = \mathtt{A}\mathbf{h}$ is the algebraic error vector for the complete set, and one sees that

$$\sum_i d_{\text{alg}}(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2 = \sum_i \|\boldsymbol{\epsilon}_i\|^2 = \|\mathtt{A}\mathbf{h}\|^2 = \|\boldsymbol{\epsilon}\|^2. \tag{4.5}$$

The concept of algebraic distance originated in the conic-fitting work of Bookstein [Bookstein-79]. Its disadvantage is that the quantity that is minimized is not geometrically or statistically meaningful. As Bookstein demonstrated, the solutions that minimize algebraic distance may not be those expected intuitively. Nevertheless, with a good choice of normalization (as will be discussed in section 4.4) methods which minimize algebraic distance do give very good results. Their particular advantages are a linear (and thus a unique) solution, and computational cheapness. Often solutions based on algebraic distance are used as a starting point for a non-linear minimization of a geometric or statistical cost function. The non-linear minimization gives the solution a final "polish".

4.2.2 Geometric distance

Next we discuss alternative error functions based on the measurement of geometric distance in the image, and minimization of the difference between the measured and estimated image coordinates.

Notation. Vectors $\mathbf{x}$ represent the *measured* image coordinates; $\hat{\mathbf{x}}$ represent estimated values of the points and $\bar{\mathbf{x}}$ represent true values of the points.

Error in one image. We start by considering error only in the second image, with points in the first measured perfectly. Clearly, this will not be true in most practical situations with images. An example where the assumption is more reasonable is in estimating the projective transformation between a calibration pattern or a world plane, where points are measured to a very high accuracy, and its image. The appropriate quantity to be minimized is the *transfer* error. This is the Euclidean image distance in the second image between the measured point $\mathbf{x}'$ and the point $\mathtt{H}\bar{\mathbf{x}}$ at which the corresponding point $\bar{\mathbf{x}}$ is mapped from the first image. We use the notation $d(\mathbf{x}, \mathbf{y})$ to represent the Euclidean distance between the inhomogeneous points represented by $\mathbf{x}$ and $\mathbf{y}$. Then the transfer error for the set of correspondences is

$$\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2. \tag{4.6}$$

The estimated homography $\hat{\mathtt{H}}$ is the one for which the error (4.6) is minimized.

Symmetric transfer error. In the more realistic case where image measurement errors occur in both the images, it is preferable that errors be minimized in both images, and not solely in the one. One way of constructing a more satisfactory error function is to

consider the forward ($\mathtt{H}$) and backward ($\mathtt{H}^{-1}$) transformation, and sum the geometric errors corresponding to each of these two transformations. Thus, the error is

$$\sum_i d(\mathbf{x}_i, \mathtt{H}^{-1}\mathbf{x}'_i)^2 + d(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2. \tag{4.7}$$

The first term in this sum is the transfer error in the first image, and the second term is the transfer error in the second image. Again the estimated homography $\hat{\mathtt{H}}$ is the one for which (4.7) is minimized.

4.2.3 Reprojection error – both images

An alternative method of quantifying error in each of the two images involves estimating a "correction" for each correspondence. One asks how much it is necessary to correct the measurements in each of the two images in order to obtain a perfectly matched set of image points. One should compare this with the geometric one-image transfer error (4.6) which measures the correction that it is necessary to make to the measurements in one image (the second image) in order to get a set of perfectly matching points.

In the present case, we are seeking a homography $\hat{\mathtt{H}}$ and pairs of *perfectly* matched points $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}'_i$ that minimize the total error function

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 \quad \text{subject to } \hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i \ \ \forall i. \tag{4.8}$$

Minimizing this cost function involves determining both $\hat{\mathtt{H}}$ *and* a set of subsidiary correspondences $\{\hat{\mathbf{x}}_i\}$ and $\{\hat{\mathbf{x}}'_i\}$. This estimation models, for example, the situation that measured correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ arise from images of points on a world plane. We wish to estimate a point on the world plane $\widehat{\mathbf{X}}_i$ from $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ which is then *reprojected* to the estimated perfectly matched correspondence $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$.

This reprojection error function is compared with the symmetric error function in figure 4.2. It will be seen in section 4.3 that (4.8) is related to the Maximum Likelihood estimation of the homography and correspondences.

4.2.4 Comparison of geometric and algebraic distance

We return to the case of errors only in the second image. Let $\mathbf{x}'_i = (x'_i, y'_i, w'_i)^\mathsf{T}$ and define a vector $(\hat{x}'_i, \hat{y}'_i, \hat{w}'_i)^\mathsf{T} = \hat{\mathbf{x}}'_i = \mathtt{H}\bar{\mathbf{x}}_i$. Using this notation, the left hand side of (4.3) becomes

$$\mathtt{A}_i\mathbf{h} = \boldsymbol{\epsilon}_i = \begin{pmatrix} y'_i\hat{w}'_i - w'_i\hat{y}'_i \\ w'_i\hat{x}'_i - x'_i\hat{w}'_i \end{pmatrix}.$$

This vector is the *algebraic error vector* associated with the point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ and the camera mapping $\mathtt{H}$. Thus,

$$d_{\text{alg}}(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 = (y'_i\hat{w}'_i - w'_i\hat{y}'_i)^2 + (w'_i\hat{x}'_i - x'_i\hat{w}'_i)^2.$$

For points $\mathbf{x}'_i$ and $\hat{\mathbf{x}}'_i$ the geometric distance is

$$d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i) \quad = \quad \left((x'_i/w'_i - \hat{x}'_i/\hat{w}'_i)^2 + (y'_i/w'_i - \hat{y}'_i/\hat{w}'_i)^2\right)^{1/2}$$

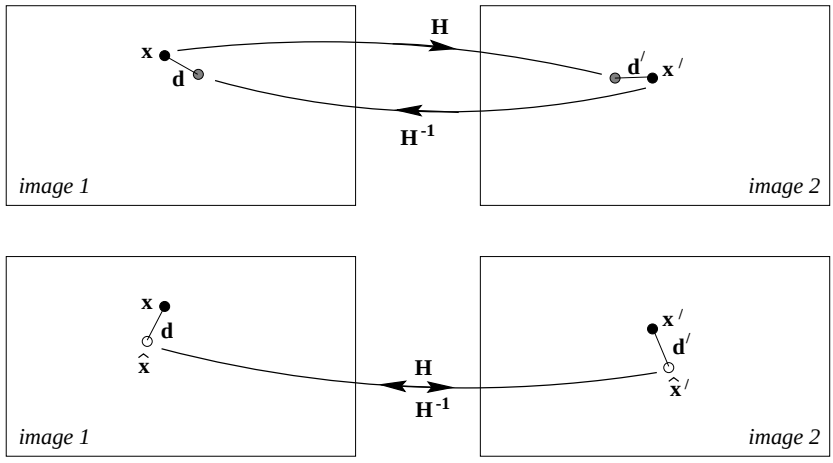


Fig. 4.2. *A comparison between symmetric transfer error (upper) and reprojection error (lower) when estimating a homography. The points* $\mathbf{x}$ *and* $\mathbf{x}'$ *are the measured (noisy) points. Under the estimated homography the points* $\mathbf{x}'$ *and* $\mathtt{H}\mathbf{x}$ *do* not *correspond perfectly (and neither do the points* $\mathbf{x}$ *and* $\mathtt{H}^{-1}\mathbf{x}'$*). However, the estimated points,* $\hat{\mathbf{x}}$ *and* $\hat{\mathbf{x}}'$*, do correspond perfectly by the homography* $\hat{\mathbf{x}}' = \mathtt{H}\hat{\mathbf{x}}$*. Using the notation* $d(\mathbf{x}, \mathbf{y})$ *for the Euclidean image distance between* $\mathbf{x}$ *and* $\mathbf{y}$*, the symmetric transfer error is* $d(\mathbf{x}, \mathtt{H}^{-1}\mathbf{x}')^2 + d(\mathbf{x}', \mathtt{H}\mathbf{x})^2$*; the reprojection error is* $d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2$.

$$= \quad d_{\text{alg}}(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)/\hat{w}'_i w'_i.$$

Thus, geometric distance is related to, but not quite the same as, algebraic distance. Note, though, that if $\hat{w}'_i = w'_i = 1$, then the two distances are identical.

One can always assume that $w_i = 1$, thus expressing the points $\mathbf{x}_i$ in the usual form $\mathbf{x}_i = (x_i, y_i, 1)^\mathsf{T}$. For one important class of 2D homographies, the values of $\hat{w}'_i$ will always be 1 as well. A 2D *affine* transformation is represented by a matrix of the form (2.10–*p*39)

$$\mathtt{H}_{\text{A}} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.9}$$

One verifies immediately from $\hat{\mathbf{x}}'_i = \mathtt{H}_{\text{A}}\bar{\mathbf{x}}_i$ that $\hat{w}'_i = 1$ if $w_i = 1$. This demonstrates that in the case of an affine transformation geometric distance and algebraic distance are identical. The DLT algorithm is easily adapted to enforce the condition that the last row of $\mathtt{H}$ has the form $(0, 0, 1)$ by setting $h_7 = h_8 = 0$. Hence, for affine transformations, geometric distance can be minimized by the linear DLT algorithm based on algebraic distance.

4.2.5 Geometric interpretation of reprojection error

The estimation of a homography between two planes can be thought of as fitting a "surface" to points in a 4D space, $\mathbb{R}^4$. Each pair of image points $\mathbf{x}$, $\mathbf{x}'$ defines a single point denoted $\mathbf{X}$ in a measurement space $\mathbb{R}^4$, formed by concatenating the inhomogeneous coordinates of $\mathbf{x}$ and $\mathbf{x}'$. For a given specific homography $\mathtt{H}$, the image correspondences $\mathbf{x} \leftrightarrow \mathbf{x}'$ that satisfy $\mathbf{x}' \times (\mathtt{H}\mathbf{x}) = \mathbf{0}$ define an algebraic variety[1] $\mathcal{V}_{\mathtt{H}}$ in $\mathbb{R}^4$ which is the

[1] A *variety* is the simultaneous zero-set of one or more multivariate polynomials defined in $\mathbb{R}^N$.

intersection of two quadric hypersurfaces. The surface is a quadric in $\mathbb{R}^4$ because each row of (4.1) is a degree 2 polynomial in x, y, x', y'. The elements of $\mathtt{H}$ determine the coefficient of each term of the polynomial, and so $\mathtt{H}$ specifies the particular quadric. The two independent equations of (4.1) define two such quadrics.

Given points $\mathbf{X}_i = (x_i, y_i, x'_i, y'_i)^\mathsf{T}$ in $\mathbb{R}^4$, the task of estimating a homography becomes the task of finding a variety $\mathcal{V}_\mathtt{H}$ that passes (or most nearly passes) through the points $\mathbf{X}_i$. In general, of course, it will not be possible to fit a variety precisely. In this case, let $\mathcal{V}_\mathtt{H}$ be some variety corresponding to a transformation $\mathtt{H}$, and for each point $\mathbf{X}_i$, let $\widehat{\mathbf{X}}_i = (\hat{x}_i, \hat{y}_i, \hat{x}'_i, \hat{y}'_i)^\mathsf{T}$ be the closest point to $\mathbf{X}_i$ lying on the variety $\mathcal{V}_\mathtt{H}$. One sees immediately that

$$\begin{aligned} \|\mathbf{X}_i - \widehat{\mathbf{X}}_i\|^2 &= (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (x'_i - \hat{x}'_i)^2 + (y'_i - \hat{y}'_i)^2 \\ &= d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2. \end{aligned}$$

Thus geometric distance in $\mathbb{R}^4$ is equivalent to the reprojection error measured in both the images, and finding the variety $\mathcal{V}_\mathtt{H}$ and points $\widehat{\mathbf{X}}_i$ on $\mathcal{V}_\mathtt{H}$ that minimize the squared sum of distances to the measured points $\mathbf{X}_i$ is equivalent to finding the homography $\hat{\mathtt{H}}$ and the estimated points $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}'_i$ that minimize the reprojection error function (4.8).

The point $\widehat{\mathbf{X}}$ on $\mathcal{V}_\mathtt{H}$ that lies closest to a measured point $\mathbf{X}$ is a point where the line between $\mathbf{X}$ and $\widehat{\mathbf{X}}$ is perpendicular to the tangent plane to $\mathcal{V}_\mathtt{H}$ at $\widehat{\mathbf{X}}$. Thus

$$d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 = d_\perp(\mathbf{X}_i, \mathcal{V}_\mathtt{H})^2$$

where $d_\perp(\mathbf{X}, \mathcal{V}_\mathtt{H})$ is the perpendicular distance of the point $\mathbf{X}$ to the variety $\mathcal{V}_\mathtt{H}$. As may be seen from the conic-fitting analogue discussed below, there may be more than one such perpendicular from $\mathbf{X}$ to $\mathcal{V}_\mathtt{H}$.

The distance $d_\perp(\mathbf{X}, \mathcal{V}_\mathtt{H})$ is invariant to rigid transformations of $\mathbb{R}^4$, and this includes as a special case rigid transformations of the coordinates $(x, y), (x', y')$ of each image individually. This point is returned to in section 4.4.3.

Conic analogue. Before proceeding further we will first sketch an analogous estimation problem that can be visualized more easily. The problem is fitting a conic to 2D points, which occupies a useful intermediate position between fitting a straight line (no curvature, too simple) and fitting a homography (four dimensions, with non-zero curvature).

Consider the problem of fitting a conic to a set of $n > 5$ points $(x_i, y_i)^\mathsf{T}$ on the plane such that an error based on geometric distance is minimized. The points may be thought of as "correspondences" $x_i \leftrightarrow y_i$. The transfer distance and reprojection (perpendicular) distance are illustrated in figure 4.3. It is clear from this figure that $d_\perp$ is less than or equal to the transfer error.

The algebraic distance of a point $\mathbf{x}$ from a conic $\mathtt{C}$ is defined as $d_{\text{alg}}(\mathbf{x}, \mathtt{C})^2 = \mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x}$. A linear solution for $\mathtt{C}$ can be obtained by minimizing $\sum_i d_{\text{alg}}(\mathbf{x}_i, \mathtt{C})^2$ with a suitable normalization on $\mathtt{C}$. There is no linear expression for the perpendicular distance of a point (x, y) to a conic $\mathtt{C}$, since through each point in $\mathbb{R}^2$ there are up to 4 lines perpendicular to $\mathtt{C}$. The solution can be obtained from the roots of a quartic. However, a function $d_\perp(\mathbf{x}, \mathtt{C})$ may be defined which returns the shortest distance between a conic

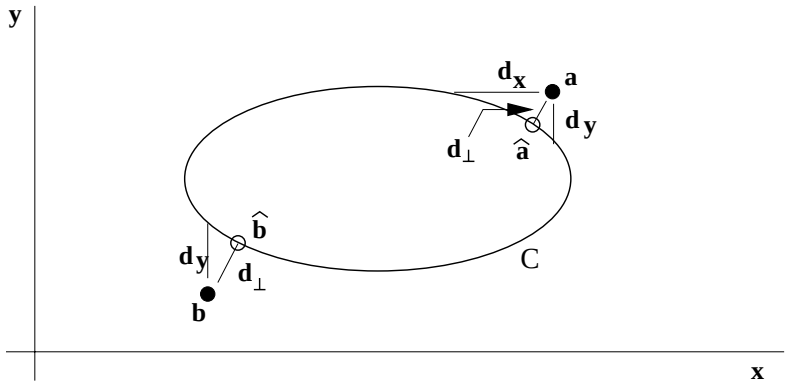


Fig. 4.3. *A conic may be estimated from a set of 2D points by minimizing "symmetric transfer error" $d_x^2 + d_y^2$ or the sum of squared perpendicular distances $d_\perp^2$. The analogue of transfer error is to consider x as perfect and measure the distance d_y to the conic in the y direction, and similarly for d_x. For point* $\mathbf{a}$ *it is clear that $d_\perp \leq d_x$ and $d_\perp \leq d_y$. Also $d_\perp$ is more stable than d_x or d_y as illustrated by point* $\mathbf{b}$ *where d_x cannot be defined.*

and a point. A conic can then be estimated by minimizing $\sum_i d_\perp(\mathbf{x}_i, \mathtt{C})^2$ over the five parameters of $\mathtt{C}$, though this cannot be achieved by a linear solution. Given a conic $\mathtt{C}$ and a measured point $\mathbf{x}$, a corrected point $\hat{\mathbf{x}}$ is obtained simply by choosing the closest point on $\mathtt{C}$.

We return now to estimating a homography. In the case of an affine transformation the variety is the intersection of two hyperplanes, i.e. it is a linear subspace of dimension 2. This follows from the form (4.9) of the affine matrix which for $\mathbf{x}' = \mathtt{H}_\mathrm{A}\mathbf{x}$ yields one linear constraint between x, x', y and another between x, y, y', each of which defines a hyperplane in $\mathbb{R}^4$. An analogue of this situation is line fitting to points on the plane. In both cases the relation (affine transformation or line) may be estimated by minimizing the perpendicular distance of points to the variety. In both cases there is a closed form solution as discussed in the following section.

4.2.6 Sampson error

The geometric error (4.8) is quite complex in nature, and minimizing it requires the simultaneous estimation of both the homography matrix and the points $\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_i'$. This non-linear estimation problem will be discussed further in section 4.5. Its complexity contrasts with the simplicity of minimizing the algebraic error (4.4). The geometric interpretation of geometric error given in section 4.2.5 leads to a further cost function that lies between the algebraic and geometric cost functions in terms of complexity, but gives a close approximation to geometric error. We will refer to this cost function as *Sampson error* since Sampson [Sampson-82] used this approximation for conic fitting.

As described in section 4.2.5, the vector $\widehat{\mathbf{X}}$ that minimizes the geometric error $\|\mathbf{X} - \widehat{\mathbf{X}}\|^2$ is the closest point on the variety $\mathcal{V}_\mathtt{H}$ to the measurement $\mathbf{X}$. This point can not be estimated directly except via iteration, because of the non-linear nature of the variety $\mathcal{V}_\mathtt{H}$. The idea of the Sampson error function is to estimate a first-order approximation to the point $\widehat{\mathbf{X}}$, assuming that the cost function is well approximated linearly in the neighbourhood of the estimated point. The discussion to follow is related directly to

the 2D homography estimation problem, but applies substantially unchanged to the other estimation problems discussed in this book.

For a given homography $\mathtt{H}$, any point $\mathbf{X} = (x, y, x', y')^\mathsf{T}$ that lies on $\mathcal{V}_\mathtt{H}$ will satisfy the equation (4.3–*p*89), or $\mathtt{A}\mathbf{h} = \mathbf{0}$. To emphasize the dependency on $\mathbf{X}$ we will write this instead as $\mathcal{C}_\mathtt{H}(\mathbf{X}) = \mathbf{0}$, where $\mathcal{C}_\mathtt{H}(\mathbf{X})$ is in this case a 2-vector. To first order, this cost function may be approximated by a Taylor expansion

$$\mathcal{C}_\mathtt{H}(\mathbf{X} + \boldsymbol{\delta}_\mathbf{X}) = \mathcal{C}_\mathtt{H}(\mathbf{X}) + \frac{\partial \mathcal{C}_\mathtt{H}}{\partial \mathbf{X}} \boldsymbol{\delta}_\mathbf{X}. \tag{4.10}$$

If we write $\boldsymbol{\delta}_\mathbf{X} = \widehat{\mathbf{X}} - \mathbf{X}$ and desire $\widehat{\mathbf{X}}$ to lie on the variety $\mathcal{V}_\mathtt{H}$ so that $\mathcal{C}_\mathtt{H}(\widehat{\mathbf{X}}) = \mathbf{0}$, then the result is $\mathcal{C}_\mathtt{H}(\mathbf{X}) + (\partial \mathcal{C}_\mathtt{H}/\partial \mathbf{X})\boldsymbol{\delta}_\mathbf{X} = \mathbf{0}$, which we will henceforth write as $\mathtt{J}\boldsymbol{\delta}_\mathbf{X} = -\boldsymbol{\epsilon}$ where $\mathtt{J}$ is the partial-derivative matrix, and $\boldsymbol{\epsilon}$ is the cost $\mathcal{C}_\mathtt{H}(\mathbf{X})$ associated with $\mathbf{X}$. The minimization problem that we now face is to find the smallest $\boldsymbol{\delta}_\mathbf{X}$ that satisfies this equation, namely:

- *Find the vector $\boldsymbol{\delta}_\mathbf{X}$ that minimizes $\|\boldsymbol{\delta}_\mathbf{X}\|$ subject to $\mathtt{J}\boldsymbol{\delta}_\mathbf{X} = -\boldsymbol{\epsilon}$.*

The standard way to solve problems of this type is to use Lagrange multipliers. A vector $\boldsymbol{\lambda}$ of Lagrange multipliers is introduced, and the problem reduces to that of finding the extrema of $\boldsymbol{\delta}_\mathbf{X}^\mathsf{T}\boldsymbol{\delta}_\mathbf{X} - 2\boldsymbol{\lambda}^\mathsf{T}(\mathtt{J}\boldsymbol{\delta}_\mathbf{X} + \boldsymbol{\epsilon})$, where the factor 2 is simply introduced for convenience. Taking derivatives with respect to $\boldsymbol{\delta}_\mathbf{X}$ and equating to zero gives

$$2\boldsymbol{\delta}_\mathbf{X}^\mathsf{T} - 2\boldsymbol{\lambda}^\mathsf{T}\mathtt{J} = \mathbf{0}^\mathsf{T}$$

from which we obtain $\boldsymbol{\delta}_\mathbf{X} = \mathtt{J}^\mathsf{T}\boldsymbol{\lambda}$. The derivative with respect to $\boldsymbol{\lambda}$ gives $\mathtt{J}\boldsymbol{\delta}_\mathbf{X} + \boldsymbol{\epsilon} = \mathbf{0}$, the original constraint. Substituting for $\boldsymbol{\delta}_\mathbf{X}$ leads to

$$\mathtt{J}\mathtt{J}^\mathsf{T}\boldsymbol{\lambda} = -\boldsymbol{\epsilon}$$

which may be solved for $\boldsymbol{\lambda}$ giving $\boldsymbol{\lambda} = -(\mathtt{J}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon}$, and so finally

$$\boldsymbol{\delta}_\mathbf{X} = -\mathtt{J}^\mathsf{T}(\mathtt{J}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon}, \tag{4.11}$$

and $\widehat{\mathbf{X}} = \mathbf{X} + \boldsymbol{\delta}_\mathbf{X}$. The norm $\|\boldsymbol{\delta}_\mathbf{X}\|^2$ is the Sampson error:

$$\|\boldsymbol{\delta}_\mathbf{X}\|^2 = \boldsymbol{\delta}_\mathbf{X}^\mathsf{T}\boldsymbol{\delta}_\mathbf{X} = \boldsymbol{\epsilon}^\mathsf{T}(\mathtt{J}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon}. \tag{4.12}$$

Example 4.2. Sampson approximation for a conic

We will compute the Sampson approximation to the geometric distance $d_\perp(\mathbf{x}, \mathtt{C})$ between a point $\mathbf{x}$ and conic $\mathtt{C}$ shown in figure 4.3. In this case the conic variety $\mathcal{V}_\mathtt{C}$ is defined by the equation $\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x} = 0$, so that $\mathbf{X} = (x, y)^\mathsf{T}$ is a 2-vector, $\boldsymbol{\epsilon} = \mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x}$ is a scalar, and $\mathtt{J}$ is the 1×2 matrix given by

$$\mathtt{J} = \left[\frac{\partial(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})}{\partial x}, \frac{\partial(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})}{\partial y}\right].$$

This means that $\mathtt{J}\mathtt{J}^\mathsf{T}$ is a scalar. The elements of $\mathtt{J}$ may be computed by the chain rule as

$$\frac{\partial(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})}{\partial x} = \frac{\partial(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial x} = 2\mathbf{x}^\mathsf{T}\mathtt{C}(1, 0, 0)^\mathsf{T} = 2(\mathtt{C}\mathbf{x})_1$$

where $(\mathtt{C}\mathbf{x})_i$ denotes the i-th component of the 3-vector $\mathtt{C}\mathbf{x}$. Then from (4.12)

$$d_\perp^2 = \|\boldsymbol{\delta}_\mathbf{X}\|^2 = \boldsymbol{\epsilon}^\mathsf{T}(\mathtt{J}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon} = \frac{\boldsymbol{\epsilon}^\mathsf{T}\boldsymbol{\epsilon}}{\mathtt{J}\mathtt{J}^\mathsf{T}} = \frac{(\mathbf{x}^\mathsf{T}\mathtt{C}\mathbf{x})^2}{4((\mathtt{C}\mathbf{x})_1^2 + (\mathtt{C}\mathbf{x})_2^2)}$$

△

A few points to note:

(i) For the 2D homography estimation problem, $\mathbf{X} = (x, y, x', y')^\mathsf{T}$ where the 2D measurements are $\mathbf{x} = (x, y, 1)^\mathsf{T}$ and $\mathbf{x}' = (x', y', 1)^\mathsf{T}$.

(ii) $\boldsymbol{\epsilon} = \mathcal{C}_\mathtt{H}(\mathbf{X})$ is the algebraic error vector $\mathtt{A}_i\mathbf{h}$ – a 2-vector – and $\mathtt{A}_i$ is defined in (4.3–*p*89).

(iii) $\mathtt{J} = \partial\mathcal{C}_\mathtt{H}(\mathbf{X})/\partial\mathbf{X}$ is a 2×4 matrix. For example

$$J_{11} = \partial(-w_i'\mathbf{x}_i^\mathsf{T}\mathbf{h}^2 + y_i'\mathbf{x}_i^\mathsf{T}\mathbf{h}^3)/\partial x = -w_i'h_{21} + y_i'h_{31}.$$

(iv) Note the similarity of (4.12) to the algebraic error $\|\boldsymbol{\epsilon}\| = \boldsymbol{\epsilon}^\mathsf{T}\boldsymbol{\epsilon}$. The Sampson error may be interpreted as being the Mahalanobis norm (see section A2.1-(*p*565)), $\|\boldsymbol{\epsilon}\|_{\mathtt{J}\mathtt{J}^\mathsf{T}}$.

(v) One could alternatively use $\mathtt{A}$ defined by (4.1–*p*89), in which case $\mathtt{J}$ has dimension 3×4 and $\boldsymbol{\epsilon}$ is a 3-vector. However, in general the Sampson error, and consequently the solution $\boldsymbol{\delta}_\mathbf{X}$, will be independent of whether (4.1–*p*89) or (4.3–*p*89) is used.

The Sampson error (4.12) is derived here for a single point pair. In applying this to the estimation of a 2D homography $\mathtt{H}$ from several point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$, the errors corresponding to all the point correspondences must be summed, giving

$$\mathcal{D}_\perp = \sum_i \boldsymbol{\epsilon}_i^\mathsf{T}(\mathtt{J}_i\mathtt{J}_i^\mathsf{T})^{-1}\boldsymbol{\epsilon}_i \tag{4.13}$$

where $\boldsymbol{\epsilon}$ and $\mathtt{J}$ both depend on $\mathtt{H}$. To estimate $\mathtt{H}$, this expression must be minimized over all values of $\mathtt{H}$. This is a simple minimization problem in which the set of variable parameters consists only of the entries (or some other parametrization) of $\mathtt{H}$.

This derivation of the Sampson error assumed that each point had isotropic (circular) error distribution, the same in each image. The appropriate formulae for more general Gaussian error distributions are given in the exercises at the end of this chapter.

Linear cost function

The algebraic error vector $\mathcal{C}_\mathtt{H}(\mathbf{X}) = \mathtt{A}(\mathbf{X})\mathbf{h}$ is typically multilinear in the entries of $\mathbf{X}$. The case where $\mathtt{A}(\mathbf{X})\mathbf{h}$ is *linear* is, however, important in its own right. The first point to note is that in this case, the first-order approximation to geometric error given by the Taylor expansion in (4.10) is exact (the higher order terms are zero), which means that *the Sampson error is identical to geometric error.*

In addition, the variety $\mathcal{V}_\mathtt{H}$ defined by the equation $\mathcal{C}_\mathtt{H}(\mathbf{X}) = \mathbf{0}$, a set of linear equations, is a hyperplane depending on $\mathtt{H}$. The problem of finding $\mathtt{H}$ now becomes a hyperplane fitting problem – find the best fit to the data $\mathbf{X}_i$ among the hyperplanes parametrized by $\mathtt{H}$.

As an example of this idea a *linear* algorithm which minimizes geometric error (4.8) for an affine transformation is developed in the exercises at the end of this chapter.

4.2.7 Another geometric interpretation

It was shown in section 4.2.5 that finding a homography that takes a set of points $\mathbf{x}_i$ to another set $\mathbf{x}'_i$ is equivalent to the problem of fitting a variety of a given type to a set of points in $\mathbb{R}^4$. We now consider a different interpretation in which the set of all measurements is represented by a single point in a measurement space $\mathbb{R}^N$.

The estimation problems we consider may all be fitted into a common framework. In abstract terms the estimation problem has two components,

- a *measurement space* $\mathbb{R}^N$ consisting of *measurement vectors* $\mathbf{X}$, and
- a *model*, which in abstract terms may be thought of simply as a subset S of points in $\mathbb{R}^N$. A measurement vector $\mathbf{X}$ that lies inside this subset is said to *satisfy the model*. Typically the subspace that satisfies the model is a submanifold, or variety in $\mathbb{R}^N$.

Now, given a measurement vector $\mathbf{X}$ in $\mathbb{R}^N$, the estimation problem is to find the vector $\widehat{\mathbf{X}}$, closest to $\mathbf{X}$, that satisfies the model.

It will now be pointed out how the 2D homography estimation problem fits into this framework.

Error in both images. Let $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$ be a set of measured matched points for $i = 1, \ldots, n$. In all, there are $4n$ measurements, namely two coordinates in each of two images for n points. Thus, the set of matched points represents a point in $\mathbb{R}^N$, where $N = 4n$. The vector made up of the coordinates of all the matched points in both images will be denoted $\mathbf{X}$.

Of course, not all sets of point pairs $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ are related via a homography $\mathtt{H}$. A set of point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$ for which there exists a projective transformation $\mathtt{H}$ satisfying $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$ for all i constitutes the subset of $\mathbb{R}^N$ satisfying the model. In general, this set of points will form a submanifold S in $\mathbb{R}^N$ (in fact a variety) of some dimension. The dimension of this submanifold is equal to the minimal number of parameters that may be used to parametrize the submanifold.

One may arbitrarily choose n points $\hat{\mathbf{x}}_i$ in the first image. In addition, a homography $\mathtt{H}$ may be chosen arbitrarily. Once these choices have been made, the points $\hat{\mathbf{x}}'_i$ in the second image are determined by $\hat{\mathbf{x}}'_i = \mathtt{H}\hat{\mathbf{x}}_i$. Thus, a feasible choice of points is determined by a set of $2n + 8$ parameters: the $2n$ coordinates of the points $\hat{\mathbf{x}}_i$, plus the 8 independent parameters (degrees of freedom) of the transformation $\mathtt{H}$. Thus, the submanifold $S \subset \mathbb{R}^N$ has dimension $2n + 8$, and hence codimension $2n - 8$.

Given a set of measured point pairs $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, corresponding to a point $\mathbf{X}$ in $\mathbb{R}^N$, and an estimated point $\widehat{\mathbf{X}} \in \mathbb{R}^N$ lying on S, one easily verifies that

$$\|\mathbf{X} - \widehat{\mathbf{X}}\|^2 = \sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2.$$

Thus, finding the point $\widehat{\mathbf{X}}$ on S lying closest to $\mathbf{X}$ in $\mathbb{R}^N$ is equivalent to minimizing the cost function given by (4.8). The estimated correct correspondences $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$ are

those corresponding to the closest surface point $\widehat{\mathbf{X}}$ in $\mathbb{R}^N$. Once $\widehat{\mathbf{X}}$ is known H may be computed.

Error in one image only. In the case of error in one image, one has a set of correspondences $\{\bar{\mathbf{x}}_i \leftrightarrow \mathbf{x}'_i\}$. The points $\bar{\mathbf{x}}_i$ are assumed perfect. The inhomogeneous coordinates of the $\mathbf{x}'_i$ constitute the measurement vector $\mathbf{X}$. Hence, in this case the measurement space has dimension $N = 2n$. The vector $\widehat{\mathbf{X}}$ consists of the inhomogeneous coordinates of the mapped perfect points $\{\mathtt{H}\bar{\mathbf{x}}_1, \mathtt{H}\bar{\mathbf{x}}_2, \ldots, \mathtt{H}\bar{\mathbf{x}}_n\}$. The set of measurement vectors satisfying the model is the set $\widehat{\mathbf{X}}$ as H varies over the set of all homography matrices. Once again this subspace is a variety. Its dimension is 8, since this is the total number of degrees of freedom of the homography matrix H. As with the previous case, the codimension is $2n - 8$. One verifies that

$$\|\mathbf{X} - \widehat{\mathbf{X}}\|^2 = \sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2.$$

Thus, finding the closest point on S to the measurement vector $\mathbf{X}$ is equivalent to minimizing the cost function (4.6).

4.3 Statistical cost functions and Maximum Likelihood estimation

In section 4.2, various cost functions were considered that were related to geometric distance between estimated and measured points in an image. The use of such cost functions is now justified and then generalized by a consideration of error statistics of the point measurements in an image.

In order to obtain a best (optimal) estimate of H it is necessary to have a model for the measurement error (the "noise"). We are assuming here that in the absence of measurement error the true points exactly satisfy a homography, i.e. $\bar{\mathbf{x}}'_i = \mathtt{H}\bar{\mathbf{x}}_i$. A common assumption is that image coordinate measurement errors obey a Gaussian (or normal) probability distribution. This assumption is surely not justified in general, and takes no account of the presence of outliers (grossly erroneous measurements) in the measured data. Methods for detecting and removing outliers will be discussed later in section 4.7. Once outliers have been removed, the assumption of a Gaussian error model, if still not strictly justified, becomes more tenable. Therefore, for the present, we assume that image measurement errors obey a zero-mean isotropic Gaussian distribution. This distribution is described in section A2.1(*p*565).

Specifically we assume that the noise is Gaussian on each image coordinate with zero mean and uniform standard deviation σ. This means that $x = \bar{x} + \Delta x$, with Δx obeying a Gaussian distribution with variance σ^2. If it is further assumed that the noise on each measurement is independent, then, if the true point is $\bar{\mathbf{x}}$, the probability density function (PDF) of each measured point $\mathbf{x}$ is

$$\Pr(\mathbf{x}) = \left(\frac{1}{2\pi\sigma^2}\right) e^{-d(\mathbf{x},\bar{\mathbf{x}})^2/(2\sigma^2)}. \tag{4.14}$$

Error in one image. First we consider the case where the errors are only in the second image. The probability of obtaining the set of correspondences $\{\bar{\mathbf{x}}_i \leftrightarrow \mathbf{x}'_i\}$ is

simply the product of their individual PDFs, since the errors on each point are assumed independent. Then the PDF of the noise-perturbed data is

$$\Pr(\{\mathbf{x}'_i\}|\mathtt{H}) = \prod_i \left(\frac{1}{2\pi\sigma^2}\right) e^{-d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2/(2\sigma^2)} \quad . \tag{4.15}$$

The symbol $\Pr(\{\mathbf{x}'_i\}|\mathtt{H})$ is to be interpreted as meaning the probability of obtaining the measurements $\{\mathbf{x}'_i\}$ given that the true homography is $\mathtt{H}$. The *log-likelihood* of the set of correspondences is

$$\log \Pr(\{\mathbf{x}'_i\}|\mathtt{H}) = -\frac{1}{2\sigma^2}\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2 + \text{ constant}.$$

The *Maximum Likelihood estimate* (MLE) of the homography, $\hat{\mathtt{H}}$, maximizes this log-likelihood, i.e. minimizes

$$\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2.$$

Thus, we note that ML estimation is equivalent to minimizing the geometric error function (4.6).

Error in both images. Following a similar development to the above, if the true correspondences are $\{\bar{\mathbf{x}}_i \leftrightarrow \mathtt{H}\bar{\mathbf{x}}_i = \bar{\mathbf{x}}'_i\}$, then the PDF of the noise-perturbed data is

$$\Pr(\{\mathbf{x}_i, \mathbf{x}'_i\}|\mathtt{H}, \{\bar{\mathbf{x}}_i\}) = \prod_i \left(\frac{1}{2\pi\sigma^2}\right) e^{-\left(d(\mathbf{x}_i, \bar{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2\right)/(2\sigma^2)}.$$

The additional complication here is that we have to seek "corrected" image measurements that play the role of the true measurements ($\mathtt{H}\bar{\mathbf{x}}$ above). Thus the ML estimate of the projective transformation $\mathtt{H}$ *and* the correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, is the homography $\hat{\mathtt{H}}$ and corrected correspondences $\{\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i\}$ that minimize

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

with $\hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i$. Note that in this case, the ML estimate is identical with minimizing the reprojection error function (4.8).

Mahalanobis distance. In the general Gaussian case, one may assume a vector of measurements $\mathbf{X}$ satisfying a Gaussian distribution function with covariance matrix Σ. The cases above are equivalent to a covariance matrix which is a multiple of the identity.

Maximizing the log-likelihood is then equivalent to minimizing the Mahalanobis distance (see section A2.1(*p*565))

$$\|\mathbf{X} - \bar{\mathbf{X}}\|^2_\Sigma = (\mathbf{X} - \bar{\mathbf{X}})^\mathsf{T}\Sigma^{-1}(\mathbf{X} - \bar{\mathbf{X}}).$$

In the case where there is error in each image, but assuming that errors in one image are independent of the error in the other image, the appropriate cost function is

$$\|\mathbf{X} - \bar{\mathbf{X}}\|^2_\Sigma + \|\mathbf{X}' - \bar{\mathbf{X}}'\|^2_{\Sigma'}$$

where Σ and Σ' are the covariance matrices of the measurements in the two images.

Finally, if we assume that the errors for all the points $\mathbf{x}_i$ and $\mathbf{x}'_i$ are independent, with individual covariance matrices Σ_i and Σ'_i respectively, then the above expression expands to

$$\sum \|\mathbf{x}_i - \bar{\mathbf{x}}_i\|^2_{\Sigma_i} + \sum \|\mathbf{x}'_i - \bar{\mathbf{x}}'_i\|^2_{\Sigma'_i} \tag{4.16}$$

This equation allows the incorporation of the type of anisotropic covariance matrices that arise for point locations computed as the intersection of two non-perpendicular lines. In the case where the points are known exactly in one of the two images, errors being confined to the other image, one of the two summation terms in (4.16) disappears.

4.4 Transformation invariance and normalization

We now start to discuss the properties and performance of the DLT algorithm of section 4.1 and how it compares with algorithms minimizing geometric error. The first topic is the invariance of the algorithm to different choices of coordinates in the image. It is clear that it would generally be undesirable for the result of an algorithm to be dependent on such arbitrary choices as the origin and scale, or even orientation, of the coordinate system in an image.

4.4.1 Invariance to image coordinate transformations

Image coordinates are sometimes given with the origin at the top-left of the image, and sometimes with the origin at the centre. The question immediately occurs whether this makes a difference to the results of computing the transformation. Similarly, if the units used to express image coordinates are changed by multiplication by some factor, then is it possible that the result of the algorithm changes also? More generally, to what extent is the result of an algorithm that minimizes a cost function to estimate a homography dependent on the choice of coordinates in the image? Suppose, for instance, that the image coordinates are changed by some similarity, affine or even projective transformation before running the algorithm. Will this materially change the result?

Formally, suppose that coordinates $\mathbf{x}$ in one image are replaced by $\tilde{\mathbf{x}} = \mathtt{T}\mathbf{x}$, and coordinates $\mathbf{x}'$ in the other image are replaced by $\tilde{\mathbf{x}}' = \mathtt{T}'\mathbf{x}'$, where $\mathtt{T}$ and $\mathtt{T}'$ are 3×3 homographies. Substituting in the equation $\mathbf{x}' = \mathtt{H}\mathbf{x}$, we derive the equation $\tilde{\mathbf{x}}' = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}\tilde{\mathbf{x}}$. This relation implies that $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$ is the transformation matrix for the point correspondences $\tilde{\mathbf{x}} \leftrightarrow \tilde{\mathbf{x}}'$. An alternative method of finding the transformation taking $\mathbf{x}_i$ to $\mathbf{x}'_i$ is therefore suggested, as follows.

(i) Transform the image coordinates according to transformations $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and $\tilde{\mathbf{x}}'_i = \mathtt{T}'\mathbf{x}'_i$.
(ii) Find the transformation $\tilde{\mathtt{H}}$ from the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$.
(iii) Set $\mathtt{H} = \mathtt{T}'^{-1}\tilde{\mathtt{H}}\mathtt{T}$.

The transformation matrix $\mathtt{H}$ found in this way applies to the original untransformed point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. What choice should be made for the transformations $\mathtt{T}$ and $\mathtt{T}'$ will be left unspecified for now. The question to be decided now is whether the

outcome of this algorithm is independent of the transformations $\mathtt{T}$ and $\mathtt{T}'$ being applied. Ideally it ought to be, at least when $\mathtt{T}$ and $\mathtt{T}'$ are similarity transformations, since the choice of a different scale, orientation or coordinate origin in the images should not materially affect the outcome of the algorithm.

In the subsequent sections it will be shown that an algorithm that minimizes geometric error is invariant to similarity transformations. On the other hand, for the DLT algorithm as described in section 4.1, the result unfortunately is not invariant to similarity transformations. The solution is to apply a normalizing transformation to the data before applying the DLT algorithm. This normalizing transformation will nullify the effect of the arbitrary selection of origin and scale in the coordinate frame of the image, and will mean that the combined algorithm is invariant to a similarity transformation of the image. Appropriate normalizing transformations will be discussed later.

4.4.2 Non-invariance of the DLT algorithm

Consider a set of correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ and a matrix $\mathtt{H}$ that is the result of the DLT algorithm applied to this set of corresponding points. Consider further a related set of correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$ where $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and $\tilde{\mathbf{x}}'_i = \mathtt{T}'\mathbf{x}'_i$, and let $\widetilde{\mathtt{H}}$ be defined by $\widetilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$. Following section 4.4.1, the question to be decided here is the following:

- *Does the DLT algorithm applied to the correspondence set $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$ yield the transformation $\widetilde{\mathtt{H}}$?*

We will use the following notation: Matrix $\mathtt{A}_i$ is the DLT equation matrix (4.3–*p*89) derived from a point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, and $\mathtt{A}$ is the $2n \times 9$ matrix formed by stacking the $\mathtt{A}_i$. Matrix $\tilde{\mathtt{A}}_i$ is similarly defined in terms of the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$, where $\tilde{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and $\tilde{\mathbf{x}}'_i = \mathtt{T}'\mathbf{x}'_i$ for some projective transformations $\mathtt{T}$ and $\mathtt{T}'$.

Result 4.3. *Let $\mathtt{T}'$ be a similarity transformation with scale factor s, and let $\mathtt{T}$ be an arbitrary projective transformation. Further, suppose $\mathtt{H}$ is any 2D homography and let $\tilde{\mathtt{H}}$ be defined by $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$. Then $\|\tilde{\mathtt{A}}\tilde{\mathbf{h}}\| = s\|\mathtt{A}\mathbf{h}\|$ where $\mathbf{h}$ and $\tilde{\mathbf{h}}$ are the vectors of entries of $\mathtt{H}$ and $\tilde{\mathtt{H}}$.*

Proof. Define the vector $\boldsymbol{\epsilon}_i = \mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i$. Note that $\mathtt{A}_i\mathbf{h}$ is the vector consisting of the first two entries of $\boldsymbol{\epsilon}_i$. Let $\tilde{\boldsymbol{\epsilon}}_i$ be similarly defined in terms of the transformed quantities as $\tilde{\boldsymbol{\epsilon}}_i = \tilde{\mathbf{x}}'_i \times \widetilde{\mathtt{H}}\tilde{\mathbf{x}}_i$. One computes:

$$\begin{aligned}\tilde{\boldsymbol{\epsilon}}_i &= \tilde{\mathbf{x}}'_i \times \widetilde{\mathtt{H}}\tilde{\mathbf{x}}_i = \mathtt{T}'\mathbf{x}'_i \times (\mathtt{T}'\mathtt{H}\mathtt{T}^{-1})\mathtt{T}\mathbf{x}_i \\ &= \mathtt{T}'\mathbf{x}'_i \times \mathtt{T}'\mathtt{H}\mathbf{x}_i = \mathtt{T}'^{*}(\mathbf{x}'_i \times \mathtt{H}\mathbf{x}_i) \\ &= \mathtt{T}'^{*}\boldsymbol{\epsilon}_i\end{aligned}$$

where $\mathtt{T}'^{*}$ represents the cofactor matrix of $\mathtt{T}'$ and the second-last equality follows from lemma A4.2(*p*581). For a general transformation $\mathtt{T}$, the error vectors $\mathtt{A}_i\mathbf{h}$ and $\tilde{\mathtt{A}}_i\tilde{\mathbf{h}}$ (namely the first two components of $\boldsymbol{\epsilon}_i$ and $\tilde{\boldsymbol{\epsilon}}_i$) are not simply related. However, in the special case where $\mathtt{T}'$ is a similarity transformation, one may write $\mathtt{T}' = \begin{bmatrix} s\mathtt{R} & \mathbf{t} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix}$ where $\mathtt{R}$ is a rotation matrix, $\mathbf{t}$ is a translation and s is a scaling factor. In this case, we

see that $\mathtt{T}'^{*} = s\begin{bmatrix} \mathtt{R} & \mathbf{0} \\ -\mathbf{t}^{\mathsf{T}}\mathtt{R} & s \end{bmatrix}$. Applying $\mathtt{T}'^{*}$ just to the first two components of $\boldsymbol{\epsilon}_i$, one sees that

$$\tilde{\mathtt{A}}_i\tilde{\mathbf{h}} = (\tilde{\epsilon}_{i1}, \tilde{\epsilon}_{i2})^{\mathsf{T}} = s\mathtt{R}(\epsilon_{i1}, \epsilon_{i2})^{\mathsf{T}} = s\mathtt{R}\mathtt{A}_i\mathbf{h}.$$

Since rotation does not affect vector norms, one sees that $\|\tilde{\mathtt{A}}\tilde{\mathbf{h}}\| = s\|\mathtt{A}\mathbf{h}\|$, as required. This result may be expressed in terms of algebraic error as

$$d_{\text{alg}}(\tilde{\mathbf{x}}'_i, \tilde{\mathtt{H}}\tilde{\mathbf{x}}_i) = sd_{\text{alg}}(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i).$$

□

Thus, there is a one-to-one correspondence between $\mathtt{H}$ and $\tilde{\mathtt{H}}$ giving rise to the same error, except for constant scale. It may appear therefore that the matrices $\mathtt{H}$ and $\tilde{\mathtt{H}}$ minimizing the algebraic error will be related by the formula $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$, and hence one may retrieve $\mathtt{H}$ as the product $\mathtt{T}'^{-1}\tilde{\mathtt{H}}\mathtt{T}$. This conclusion is **false** however. For, although $\mathtt{H}$ and $\tilde{\mathtt{H}}$ so defined give rise to the same error ϵ, the condition $\|\mathtt{H}\| = 1$, imposed as a constraint on the solution, is not equivalent to the condition $\|\tilde{\mathtt{H}}\| = 1$. Specifically, $\|\mathtt{H}\|$ and $\|\tilde{\mathtt{H}}\|$ are not related in any simple manner. Thus, there is no one-to-one correspondence between $\mathtt{H}$ and $\tilde{\mathtt{H}}$ giving rise to the same error ϵ, subject to the constraint $\|\mathtt{H}\| = \|\tilde{\mathtt{H}}\| = 1$. Specifically,

$$\begin{aligned} & \text{minimize } \sum_i d_{\text{alg}}(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2 \text{ subject to } \|\mathtt{H}\| = 1 \\ \Leftrightarrow \quad & \text{minimize } \sum_i d_{\text{alg}}(\tilde{\mathbf{x}}'_i, \tilde{\mathtt{H}}\tilde{\mathbf{x}}_i)^2 \text{ subject to } \|\mathtt{H}\| = 1 \\ \not\Leftrightarrow \quad & \text{minimize } \sum_i d_{\text{alg}}(\tilde{\mathbf{x}}'_i, \tilde{\mathtt{H}}\tilde{\mathbf{x}}_i)^2 \text{ subject to } \|\tilde{\mathtt{H}}\| = 1. \end{aligned}$$

Thus, the method of transformation leads to a different solution for the computed transformation matrix. This is a rather undesirable feature of the DLT algorithm as it stands, that the result is changed by a change of coordinates, or even simply a change of the origin of coordinates. If the constraint under which the norm $\|\mathtt{A}\mathbf{h}\|$ is minimized is invariant under the transformation, however, then one sees that the computed matrices $\mathtt{H}$ and $\tilde{\mathtt{H}}$ are related in the right way. Examples of minimization conditions for which $\mathtt{H}$ is transformation-invariant are discussed in the exercises at the end of this chapter.

4.4.3 Invariance of geometric error

It will be shown now that minimizing geometric error to find $\mathtt{H}$ is invariant under similarity (scaled Euclidean) transformations. As before, consider a point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$ and a transformation matrix $\mathtt{H}$. Also, define a related set of correspondences $\tilde{\mathbf{x}} \leftrightarrow \tilde{\mathbf{x}}'$ where $\tilde{\mathbf{x}} = \mathtt{T}\mathbf{x}$ and $\tilde{\mathbf{x}}' = \mathtt{T}'\mathbf{x}'$, and let $\tilde{\mathtt{H}}$ be defined by $\tilde{\mathtt{H}} = \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}$. Suppose that $\mathtt{T}$ and $\mathtt{T}'$ represent Euclidean transformations of $\mathbb{P}^2$. One verifies that

$$d(\tilde{\mathbf{x}}', \tilde{\mathtt{H}}\tilde{\mathbf{x}}) = d(\mathtt{T}'\mathbf{x}', \mathtt{T}'\mathtt{H}\mathtt{T}^{-1}\mathtt{T}\mathbf{x}) = d(\mathtt{T}'\mathbf{x}', \mathtt{T}'\mathtt{H}\mathbf{x}) = d(\mathbf{x}', \mathtt{H}\mathbf{x})$$

where the last equality holds because Euclidean distance is unchanged under a Euclidean transformation such as $\mathtt{T}'$. This shows that if $\mathtt{H}$ minimizes the geometric error

for a set of correspondences, then $\widetilde{\mathtt{H}}$ minimizes the geometric error for the transformed set of correspondences, and so minimizing geometric error is invariant under Euclidean transformations.

For similarity transformations, geometric error is multiplied by the scale factor of the transformation, hence the minimizing transformations correspond in the same way as in the Euclidean transformation case. Minimizing geometric error is invariant to similarity transformations.

4.4.4 Normalizing transformations

As was shown in section 4.4.2, the result of the DLT algorithm for computing 2D homographies depends on the coordinate frame in which points are expressed. In fact the result is not invariant to similarity transformations of the image. This suggests the question whether some coordinate systems are in some way better than others for computing a 2D homography. The answer to this is an emphatic *yes*. In this section a method of normalization of the data is described, consisting of translation and scaling of image coordinates. This normalization should be carried out before applying the DLT algorithm. Subsequently an appropriate correction to the result expresses the computed $\mathtt{H}$ with respect to the original coordinate system.

Apart from improved accuracy of results, data normalization provides a second desirable benefit, namely that an algorithm that incorporates an initial data normalization step will be invariant with respect to arbitrary choices of the scale and coordinate origin. This is because the normalization step undoes the effect of coordinate changes, by effectively choosing a canonical coordinate frame for the measurement data. Thus, algebraic minimization is carried out in a fixed canonical frame, and the DLT algorithm is in practice invariant to similarity transformations.

Isotropic scaling. As a first step of normalization, the coordinates in each image are translated (by a different translation for each image) so as to bring the centroid of the set of all points to the origin. The coordinates are also scaled so that on the average a point $\mathbf{x}$ is of the form $\mathbf{x} = (x, y, w)^\mathsf{T}$, with each of x, y and w having the same average magnitude. Rather than choose different scale factors for each coordinate direction, an isotropic scaling factor is chosen so that the x and y-coordinates of a point are scaled equally. To this end, we choose to scale the coordinates so that the average distance of a point $\mathbf{x}$ from the origin is equal to $\sqrt{2}$. This means that the "average" point is equal to $(1, 1, 1)^\mathsf{T}$. In summary the transformation is as follows:

(i) The points are translated so that their centroid is at the origin.
(ii) The points are then scaled so that the average distance from the origin is equal to $\sqrt{2}$.
(iii) This transformation is applied to each of the two images independently.

Why is normalization essential? The recommended version of the DLT algorithm with data normalization is given in algorithm 4.2. We will now motivate why this

version of the algorithm, incorporating data normalization, should be used in preference to the basic DLT of algorithm 4.1(*p*91). Note that normalization is also called *pre-conditioning* in the numerical literature.

The DLT method of algorithm 4.1 uses the SVD of $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^\mathsf{T}$ to obtain a solution to the overdetermined set of equations $\mathtt{A}\mathbf{h} = \mathbf{0}$. These equations do not have an exact solution (since the $2n \times 9$ matrix $\mathtt{A}$ will not have rank 8 for noisy data), but the vector $\mathbf{h}$, given by the last column of $\mathtt{V}$, provides a solution which minimizes $\|\mathtt{A}\mathbf{h}\|$ (subject to $\|\mathbf{h}\| = 1$). This is equivalent to finding the rank 8 matrix $\hat{\mathtt{A}}$ which is closest to $\mathtt{A}$ in Frobenius norm and obtaining $\mathbf{h}$ as the exact solution of $\hat{\mathtt{A}}\mathbf{h} = \mathbf{0}$. The matrix $\hat{\mathtt{A}}$ is given by $\hat{\mathtt{A}} = \mathtt{U}\hat{\mathtt{D}}\mathtt{V}^\mathsf{T}$ where $\hat{\mathtt{D}}$ is $\mathtt{D}$ with the smallest singular value set to zero. The matrix $\hat{\mathtt{A}}$ has rank 8 and minimizes the difference to $\mathtt{A}$ in Frobenius norm because

$$\|\mathtt{A} - \hat{\mathtt{A}}\|_\mathrm{F} = \|\mathtt{U}\mathtt{D}\mathtt{V}^\mathsf{T} - \mathtt{U}\hat{\mathtt{D}}\mathtt{V}^\mathsf{T}\|_\mathrm{F} = \|\mathtt{D} - \hat{\mathtt{D}}\|_\mathrm{F}.$$

where $\|.\|_\mathrm{F}$ is the Frobenius norm, i.e. the square root of the sum of squares of all entries.

Without normalization typical image points $\mathbf{x}_i, \mathbf{x}'_i$ are of the order $(x, y, w)^\mathsf{T} = (100, 100, 1)^\mathsf{T}$, i.e., x, y are much larger than w. In $\mathtt{A}$ the entries xx', xy', yx', yy' will be of order 10^4, entries xw', yw' etc. of order 10^2, and entries ww' will be unity. Replacing $\mathtt{A}$ by $\hat{\mathtt{A}}$ means that some entries are increased and others decreased such that the square sum of differences of these changes is minimal (and the resulting matrix has rank 8). However, and this is the key point, increasing the term ww' by 100 means a huge change in the image points, whereas increasing the term xx' by 100 means only a slight change. This is the reason why all entries in $\mathtt{A}$ must have similar magnitude and why normalization is essential.

The effect of normalization is related to the condition number of the set of DLT equations, or more precisely the ratio d_1/d_{n-1} of the first to the second-last singular value of the equation matrix $\mathtt{A}$. This point is investigated in more detail in [Hartley-97c]. For the present it is sufficient to say that for exact data and infinite precision arithmetic the results will be independent of the normalizing transformation. However, in the presence of noise the solution will diverge from the correct result. The effect of a large condition number is to amplify this divergence. This is true even for infinite-precision arithmetic – this is not a round-off error effect.

The effect that this data normalization has on the results of the DLT algorithm is shown graphically in figure 4.4. The conclusion to be drawn here is that data normalization gives dramatically better results. The examples shown in the figure are chosen to make the effect easily visible. However, a marked advantage remains even in cases of computation from larger numbers of point correspondences, with points more widely distributed. To emphasize this point we remark:

- *Data normalization is an* essential *step in the DLT algorithm. It must* not *be considered optional.*

Data normalization becomes even more important for less well conditioned problems, such as the DLT computation of the fundamental matrix or the trifocal tensor, which will be considered in later chapters.

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the 2D homography matrix $\mathtt{H}$ such that $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$.

Algorithm

(i) **Normalization of $\mathbf{x}$:** Compute a similarity transformation $\mathtt{T}$, consisting of a translation and scaling, that takes points $\mathbf{x}_i$ to a new set of points $\tilde{\mathbf{x}}_i$ such that the centroid of the points $\tilde{\mathbf{x}}_i$ is the coordinate origin $(0, 0)^\mathsf{T}$, and their average distance from the origin is $\sqrt{2}$.

(ii) **Normalization of $\mathbf{x}'$:** Compute a similar transformation $\mathtt{T}'$ for the points in the second image, transforming points $\mathbf{x}'_i$ to $\tilde{\mathbf{x}}'_i$.

(iii) **DLT:** Apply algorithm 4.1(*p*91) to the correspondences $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}'_i$ to obtain a homography $\widetilde{\mathtt{H}}$.

(iv) **Denormalization:** Set $\mathtt{H} = \mathtt{T}'^{-1}\widetilde{\mathtt{H}}\mathtt{T}$.

Algorithm 4.2. *The normalized DLT for 2D homographies.*

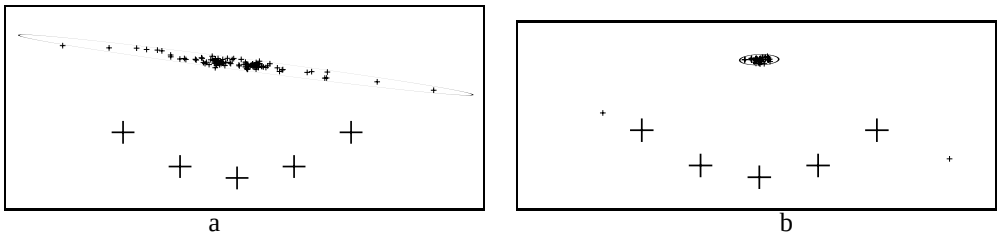


Fig. 4.4. *Results of Monte Carlo simulation (see section 5.3(*p*149) of computation of 2D homographies). A set of 5 points (denoted by large crosses) was used to compute a 2D homography. Each of the 5 points is mapped (in the noise-free case) to the point with the same coordinates, so that homography* $\mathtt{H}$ *is the identity mapping. Now, 100 trials were made with each point being subject to 0.1 pixel Gaussian noise in one image. (For reference, the large crosses are 4 pixels across.) The mapping* $\mathtt{H}$ *computed using the DLT algorithm was then applied to transfer a further point into the second image. The 100 projections of this point are shown with small crosses and the 95% ellipse computed from their scatter matrix is also shown. (a) are the results without data normalization, and (b) the results with normalization. The left- and rightmost reference points have (unnormalized) coordinates* $(130, 108)$ *and* $(170, 108)$.

Non-isotropic scaling. Other methods of scaling are also possible. In non-isotropic scaling, the centroid of the points is translated to the origin as before. After this translation the points form a cloud about the origin. Scaling is then carried out so that the two principal moments of the set of points are both equal to unity. Thus, the set of points will form an approximately symmetric circular cloud of points of radius 1 about the origin. Experimental results given in [Hartley-97c] suggest that the extra effort required for non-isotropic scaling does not lead to significantly better results than isotropic scaling.

A further variant on scaling was discussed in [Muehlich-98], based on a statistical analysis of the estimator, its bias and variance. In that paper it was observed that some columns of $\mathtt{A}$ are not affected by noise. This applies to the third and sixth columns in (4.3–*p*89), corresponding to the entry $w_i w'_i = 1$. Such error-free entries in $\mathtt{A}$ should not be varied in finding $\hat{\mathtt{A}}$, the closest rank-deficient approximation to $\mathtt{A}$. A method known

as Total Least Squares - Fixed Columns is used to find the best solution. For estimation of the fundamental matrix (see chapter 11), [Muehlich-98] reports slightly improved results compared with non-isotropic scaling.

Scaling with points near infinity. Consider the case of estimation of a homography between an infinite plane and an image. If the viewing direction is sufficiently oblique, then very distant points in the plane may be visible in the image – even points at infinity (vanishing points) if the horizon is visible. In this case it makes no sense to normalize the coordinates of points in the infinite plane by setting the centroid at the origin, since the centroid may have very large coordinates, or be undefined. An approach to normalization in this case is considered in exercise (iii) on page 128.

4.5 Iterative minimization methods

This section describes methods for minimizing the various geometric cost functions developed in section 4.2 and section 4.3. Minimizing such cost functions requires the use of iterative techniques. This is unfortunate, because iterative techniques tend to have certain disadvantages compared to linear algorithms such as the normalized DLT algorithm 4.2:

(i) They are slower.
(ii) They generally need an initial estimate at which to start the iteration.
(iii) They risk not converging, or converging to a local minimum instead of the global minimum.
(iv) Selection of a stopping criterion for iteration may be tricky.

Consequently, iterative techniques generally require more careful implementation.

The technique of iterative minimization generally consists of five steps:

(i) **Cost function.** A cost function is chosen as the basis for minimization. Different possible cost functions were discussed in section 4.2.
(ii) **Parametrization.** The transformation (or other entity) to be computed is expressed in terms of a finite number of parameters. It is not in general necessary that this be a minimum set of parameters, and there are in fact often advantages to over-parametrization. (See the discussion below.)
(iii) **Function specification.** A function must be specified that expresses the cost in terms of the set of parameters.
(iv) **Initialization.** A suitable initial parameter estimate is computed. This will generally be done using a linear algorithm such as the DLT algorithm.
(v) **Iteration.** Starting from the initial solution, the parameters are iteratively refined with the goal of minimizing the cost function.

A word about parametrization

For a given cost function, there are often several choices of parametrization. The general strategy that guides parametrization is to select a set of parameters that cover the complete space over which one is minimizing, while at the same time allowing one to

compute the cost function in a convenient manner. For example, H may be parametrized by 9 parameters – that is, it is over-parametrized, since there are really only 8 degrees of freedom, overall scale not being significant. A *minimal* parametrization (i.e. the same number of parameters as degrees of freedom) would involve only 8 parameters.

In general no bad effects are likely to occur if a minimization problem of this type is over-parametrized, as long as for all choices of parameters the corresponding object is of the desired type. In particular for homogeneous objects, such as the 3×3 projection matrix encountered here, it is usually not necessary or advisable to attempt to use a minimal parametrization by removing the scale-factor ambiguity.

The reasoning is the following: it is not *necessary* to use minimal parametrization because a well-performing non-linear minimization algorithm will "notice" that it is not necessary to move in redundant directions, such as the matrix scaling direction. The algorithm described in Gill and Murray [Gill-78], which is a modification of the Gauss–Newton method, has an effective strategy for discarding redundant combinations of the parameters. Similarly, the Levenberg-Marquardt algorithm (see section A6.2(*p*600)) handles redundant parametrizations easily. It is not *advisable* because it is found empirically that the cost function surface is more complicated when minimal parametrizations are used. There is then a greater possibility of becoming stuck in a local minimum.

One other issue that arises in choosing a parametrization is that of restricting the transformation to a particular class. For example, suppose H is known to be a homology, then as described in section A7.2(*p*629) it may be parametrized as

$$\mathtt{H} = \mathtt{I} + (\mu - 1)\frac{\mathbf{v}\mathbf{a}^\mathsf{T}}{\mathbf{v}^\mathsf{T}\mathbf{a}}$$

where μ is a scalar, and $\mathbf{v}$ and $\mathbf{a}$ 3-vectors. A homology has 5 degrees of freedom which correspond here to the scalar μ and the directions of $\mathbf{v}$ and $\mathbf{a}$. If H is parametrized by its 9 matrix entries, then the estimated H is unlikely to exactly be a homology. However, if H is parametrized by μ, $\mathbf{v}$ and $\mathbf{a}$ (a total of 7 parameters) then the estimated H is guaranteed to be a homology. This parametrization is *consistent* with a homology (it is also an over-parametrization). We will return to the issues of consistent, local, minimal and over-parametrization in later chapters. The issues are also discussed further in appendix A6.9(*p*623).

Function specification

It has been seen in section 4.2.7 that a general class of estimation problems is concerned with a measurement space $\mathbb{R}^N$ containing a model surface S. Given a measurement $\mathbf{X} \in \mathbb{R}^N$ the estimation task is to find the point $\widehat{\mathbf{X}}$ lying on S closest to $\mathbf{X}$. In the case where a non-isotropic Gaussian error distribution is imposed on $\mathbb{R}^N$, the word *closest* is to be interpreted in terms of Mahalanobis distance. Iterative minimization methods will now be described in terms of this estimation model. In iterative estimation through parameter fitting, the model surface S is locally parametrized, and the parameters are allowed to vary to minimize the distance to the measured point. More specifically,

(i) One has a *measurement vector* $\mathbf{X} \in \mathbb{R}^N$ with covariance matrix Σ.

(ii) A set of parameters are represented as a vector $\mathbf{P} \in \mathbb{R}^M$.

(iii) A mapping $f : \mathbb{R}^M \to \mathbb{R}^N$ is defined. The range of this mapping is (at least locally) the model surface S in $\mathbb{R}^N$ representing the set of allowable measurements.

(iv) The cost function to be minimized is the squared Mahalanobis distance

$$\|\mathbf{X} - f(\mathbf{P})\|^2_\Sigma = (\mathbf{X} - f(\mathbf{P}))^\mathsf{T} \Sigma^{-1} (\mathbf{X} - f(\mathbf{P})).$$

In effect, we are attempting to find a set of parameters $\mathbf{P}$ such that $f(\mathbf{P}) = \mathbf{X}$, or failing that, to bring $f(\mathbf{P})$ as close to $\mathbf{X}$ as possible, with respect to Mahalanobis distance. The Levenberg–Marquardt algorithm is a general tool for iterative minimization, when the cost function to be minimized is of this type. We will now show how the various different types of cost functions described in this chapter fit into this format.

Error in one image. Here one fixes the coordinates of points $\mathbf{x}_i$ in the first image, and varies $\mathtt{H}$ so as to minimize cost function (4.6–*p*94), namely

$$\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2.$$

The measurement vector $\mathbf{X}$ is made up of the $2n$ inhomogeneous coordinates of the points $\mathbf{x}'_i$. One may choose as parameters the vector $\mathbf{h}$ of entries of the homography matrix $\mathtt{H}$. The function f is defined by

$$f : \mathbf{h} \mapsto (\mathtt{H}\mathbf{x}_1, \mathtt{H}\mathbf{x}_2, \ldots, \mathtt{H}\mathbf{x}_n)$$

where it is understood that here, and in the functions below, $\mathtt{H}\mathbf{x}_i$ indicates the inhomogeneous coordinates. One verifies that $\|\mathbf{X} - f(\mathbf{h})\|^2$ is equal to (4.6–*p*94).

Symmetric transfer error. In the case of the symmetric cost function (4.7–*p*95)

$$\sum_i d(\mathbf{x}_i, \mathtt{H}^{-1}\mathbf{x}'_i)^2 + d(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2$$

one chooses as measurement vector $\mathbf{X}$ the $4n$-vector made up of the inhomogeneous coordinates of the points $\mathbf{x}_i$ followed by the inhomogeneous coordinates of the points $\mathbf{x}'_i$. The parameter vector as before is the vector $\mathbf{h}$ of entries of $\mathtt{H}$, and the function f is defined by

$$f : \mathbf{h} \mapsto (\mathtt{H}^{-1}\mathbf{x}'_1, \ldots, \mathtt{H}^{-1}\mathbf{x}'_n, \mathtt{H}\mathbf{x}_1, \ldots, \mathtt{H}\mathbf{x}_n).$$

As before, we find that $\|\mathbf{X} - f(\mathbf{h})\|^2$ is equal to (4.7–*p*95).

Reprojection error. Minimizing the cost function (4.8–*p*95) is more complex. The difficulty is that it requires a simultaneous minimization over all choices of points $\hat{\mathbf{x}}_i$ as well as the entries of the transformation matrix $\mathtt{H}$. If there are many point correspondences, then this becomes a very large minimization problem. Thus, the problem may be parametrized by the coordinates of the points $\hat{\mathbf{x}}_i$ and the entries of the matrix $\hat{\mathtt{H}}$ – a total of $2n + 9$ parameters. The coordinates of $\hat{\mathbf{x}}'_i$ are not required, since they are related to the other parameters by $\hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i$. The parameter vector is therefore

$\mathbf{P} = (\mathbf{h}, \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n)$. The measurement vector contains the inhomogeneous coordinates of all the points $\mathbf{x}_i$ and $\mathbf{x}'_i$. The function f is defined by

$$f : (\mathbf{h}, \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n) \mapsto (\hat{\mathbf{x}}_1, \hat{\mathbf{x}}'_1, \ldots, \hat{\mathbf{x}}_n, \hat{\mathbf{x}}'_n)$$

where $\hat{\mathbf{x}}'_i = \mathtt{H}\hat{\mathbf{x}}_i$. One verifies that $\|\mathbf{X} - f(\mathbf{P})\|^2$, with $\mathbf{X}$ a $4n$-vector, is equal to the cost function (4.8–*p*95). This cost function must be minimized over all $2n + 9$ parameters.

Sampson approximation. In contrast with $2n + 9$ parameters of reprojection error, minimizing the error in one image (4.6–*p*94) or symmetric transfer error (4.7–*p*95) requires a minimization over the 9 entries of the matrix $\mathtt{H}$ only – in general a more tractable problem. The Sampson approximation to reprojection error enables reprojection error also to be minimized with only 9 parameters.

This is an important consideration, since the iterative solution of an m-parameter non-linear minimization problem using a method such as Levenberg–Marquardt involves the solution of an $m \times m$ set of linear equations at each iteration step. This is a problem with complexity $O(m^3)$. Hence, it is appropriate to keep the size of m low.

The Sampson error avoids minimizing over the $2n+9$ parameters of reprojection error because effectively it determines the $2n$ variables $\{\hat{\mathbf{x}}_i\}$ for each particular choice of $\mathbf{h}$. Consequently the minimization then only requires the 9 parameters of $\mathbf{h}$. In practice this approximation gives excellent results provided the errors are small compared to the measurements.

Initialization

An initial estimate for the parametrization may be found by employing a linear technique. For example, the normalized DLT algorithm 4.2 directly provides $\mathtt{H}$ and thence the 9-vector $\mathbf{h}$ used to parametrize the iterative minimization. In general if there are $n \geq 4$ correspondences, then all will be used in the linear solution. However, as will be seen in section 4.7 on robust estimation, when the correspondences contain outliers it may be advisable to use a carefully selected minimal set of correspondences (i.e. four correspondences). Linear techniques or minimal solutions are the two initialization techniques recommended in this book.

An alternative method that is sometimes used (for instance see [Horn-90, Horn-91]) is to carry out a sufficiently dense sampling of parameter space, iterating from each sampled starting point and retaining the best result. This is only possible if the dimension of the parameter space is sufficiently small. Sampling of parameter space may be done either randomly, or else according to some pattern. Another initialization method is simply to do without any effective initialization at all, starting the iteration at a given fixed point in parameter space. This method is not often viable. Iteration is very likely to fall into a false minimum or not converge. Even in the best case, the number of iteration steps required will increase the further one starts from the final solution. For this reason using a good initialization method is the best plan.

Objective

Given $n > 4$ image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the Maximum Likelihood estimate $\hat{\mathtt{H}}$ of the homography mapping between the images.

The MLE involves also solving for a set of subsidiary points $\{\hat{\mathbf{x}}_i\}$, which minimize

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

where $\hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i$.

Algorithm

(i) **Initialization**: Compute an initial estimate of $\hat{\mathtt{H}}$ to provide a starting point for the geometric minimization. For example, use the linear normalized DLT algorithm 4.2, or use RANSAC (section 4.7.1) to compute $\hat{\mathtt{H}}$ from four point correspondences.

(ii) **Geometric minimization of – <u>either</u> Sampson error:**

- Minimize the Sampson approximation to the geometric error (4.12–*p*99).
- The cost is minimized using the Newton algorithm of section A6.1(*p*597) or Levenberg–Marquardt algorithm of section A6.2(*p*600) over a suitable parametrization of $\hat{\mathtt{H}}$. For example the matrix may be parametrized by its 9 entries.

<u>or</u> Gold Standard error:

- Compute an initial estimate of the subsidiary variables $\{\hat{\mathbf{x}}_i\}$ using the measured points $\{\mathbf{x}_i\}$ or (better) the Sampson correction to these points given by (4.11–*p*99).
- Minimize the cost

$$\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$$

over $\hat{\mathtt{H}}$ and $\hat{\mathbf{x}}_i, i = 1, \ldots, n$. The cost is minimized using the Levenberg–Marquardt algorithm over $2n+9$ variables: $2n$ for the n 2D points $\hat{\mathbf{x}}_i$, and 9 for the homography matrix $\hat{\mathtt{H}}$.

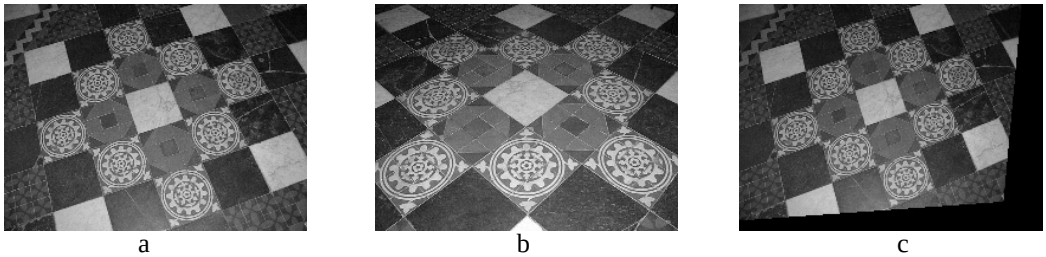
- If the number of points is large then the sparse method of minimizing this cost function given in section A6.4(*p*607) is the recommended approach.

Algorithm 4.3. *The Gold Standard algorithm and variations for estimating* H *from image correspondences. The Gold Standard algorithm is preferred to the Sampson method for 2D homography computation.*

Iteration methods

There are various iterative methods for minimizing the chosen cost function, of which the most popular are Newton iteration and the Levenberg–Marquardt method. These methods are described in appendix 6(*p*597). Other general methods for minimizing a cost function are available, such as Powell's method and the simplex method both described in [Press-88].

Summary. The ideas in this section are collected together in algorithm 4.3, which describes the Gold Standard and Sampson methods for estimating the homography mapping between point correspondences in two images.



a b c

Fig. 4.5. *Three images of a plane which are used to compare methods of computing projective transformations from corresponding points.*

Method	Pair 1 figure 4.5 a & b	Pair 2 figure 4.5 a & c
Linear normalized	0.4078	0.6602
Gold Standard	0.4078	0.6602
Linear unnormalized	0.4080	26.2056
Homogeneous scaling	0.5708	0.7421
Sampson	0.4077	0.6602
Error in 1 view	0.4077	0.6602
Affine	6.0095	2.8481
Theoretical optimal	0.5477	0.6582

Table 4.1. *Residual errors in pixels for the various algorithms.*

4.6 Experimental comparison of the algorithms

The algorithms are compared for the images shown in figure 4.5. Table 4.1 shows the results of testing several of the algorithms described in this chapter. Residual error is shown for two pairs of images. The methods used are fairly self-explanatory, with a few exceptions. The method "affine" was an attempt to fit the projective transformation with an optimal affine mapping. The "optimal" is the ML estimate assuming a noise level of one pixel.

The first pair of images are (a) and (b) of figure 4.5, with 55 point correspondences. It appears that all methods work almost equally well (except the affine method). The optimal residual is greater than the achieved results, because the noise level (unknown) is less than one pixel.

Image (c) of figure 4.5 was produced synthetically by resampling (a), and the second pair consists of (a) and (c) with 20 point correspondences. In this case, almost all methods perform almost optimally, as shown in the table 4.1. The exception is the affine method (expected to perform badly, since it is not an affine transformation) and the unnormalized linear method. The unnormalized method is expected to perform badly (though maybe not this badly). Just why it performs well in the first pair and very badly for the second pair is not understood. In any case, it is best to avoid this method and use a normalized linear or Gold Standard method.

A further evaluation is presented in figure 4.6. The transformation to be estimated is the one that maps the chessboard image shown here to a square grid aligned with the

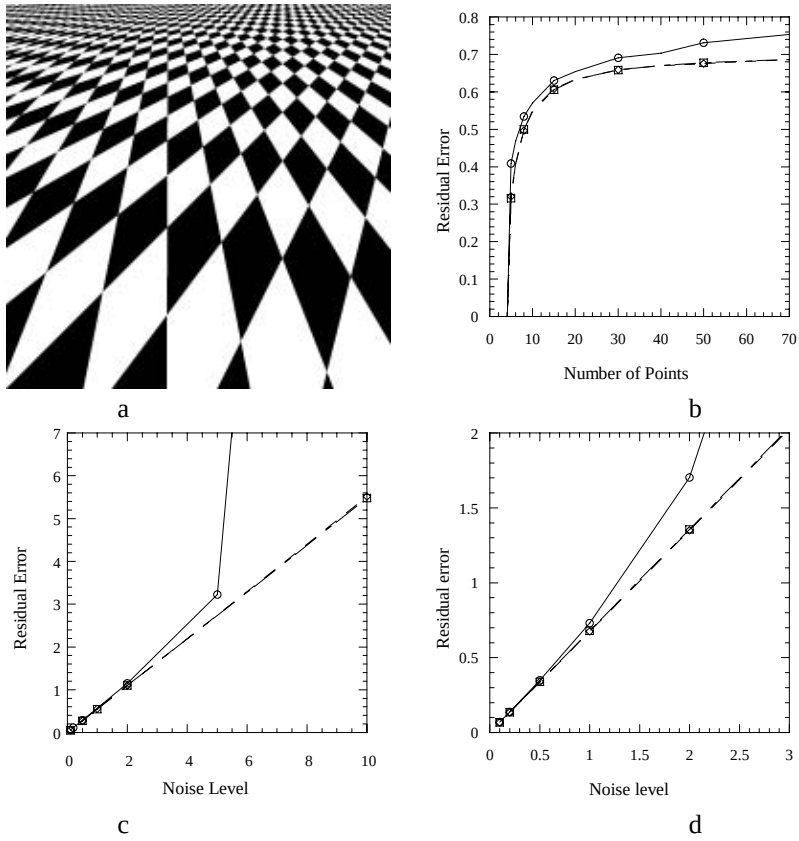


Fig. 4.6. **Comparison of the DLT and Gold Standard algorithms to the theoretically optimal residual error.** *(a) The homography is computed between a chessboard and this image. In all three graphs, the result for the Gold Standard algorithm overlap and are indistinguishable from the theoretical minimum. (b) Residual error as a function of the number of points. (c) The effect of varying noise level for 10 points, and (d) 50 points.*

axes. As may be seen, the image is substantially distorted, with respect to a square grid. For the experiments, randomly selected points in the image were chosen and matched with the corresponding point on the square grid. The (normalized) DLT algorithm and the Gold Standard algorithm are compared to the theoretical minimum or residual error (see chapter 5). Note that for noise up to 5 pixels, the DLT algorithm performs adequately. However, for a noise level of 10 pixels it fails. Note however that in a 200-pixel image, an error of 10 pixels is extremely high. For less severe homographies, closer to the identity map, the DLT performs almost as well as the Gold Standard algorithm.

4.7 Robust estimation

Up to this point it has been assumed that we have been presented with a set of correspondences, $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, where the only source of error is in the measurement of the point's position, which follows a Gaussian distribution. In many practical situations this assumption is not valid because points are mismatched. The mismatched points are *outliers* to the Gaussian error distribution. These outliers can severely disturb the

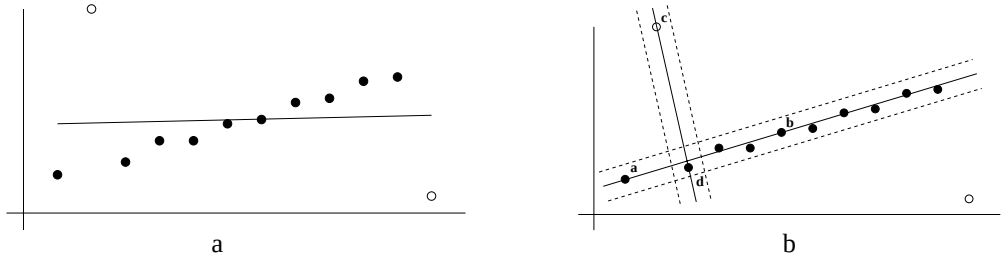


Fig. 4.7. **Robust line estimation.** *The solid points are inliers, the open points outliers. (a) A least-squares (orthogonal regression) fit to the point data is severely affected by the outliers. (b) In the RANSAC algorithm the support for lines through randomly selected point pairs is measured by the number of points within a threshold distance of the lines. The dotted lines indicate the threshold distance. For the lines shown the support is 10 for line* $\langle \mathbf{a}, \mathbf{b} \rangle$ *(where both of the points* $\mathbf{a}$ *and* $\mathbf{b}$ *are inliers); and 2 for line* $\langle \mathbf{c}, \mathbf{d} \rangle$ *where the point* $\mathbf{c}$ *is an outlier.*

estimated homography, and consequently should be identified. The goal then is to determine a set of *inliers* from the presented "correspondences" so that the homography can then be estimated in an optimal manner from these inliers using the algorithms described in the previous sections. This is *robust estimation* since the estimation is robust (tolerant) to outliers (measurements following a different, and possibly unmodelled, error distribution).

4.7.1 RANSAC

We start with a simple example that can easily be visualized – estimating a straight line fit to a set of 2-dimensional points. This can be thought of as estimating a 1-dimensional affine transformation, $x' = ax + b$, between corresponding points lying on two lines.

The problem, which is illustrated in figure 4.7a, is the following: given a set of 2D data points, find the line which minimizes the sum of squared perpendicular distances (orthogonal regression), subject to the condition that none of the valid points deviates from this line by more than t units. This is actually two problems: a line fit to the data; and a classification of the data into inliers (valid points) and outliers. The threshold t is set according to the measurement noise (for example $t = 3\sigma$), and is discussed below. There are many types of robust algorithms and which one to use depends to some extent on the proportion of outliers. For example, if it is known that there is only one outlier, then each point can be deleted in turn and the line estimated from the remainder. Here we describe in detail a general and very successful robust estimator – the RANdom SAmple Consensus (RANSAC) algorithm of Fischler and Bolles [Fischler-81]. The RANSAC algorithm is able to cope with a large proportion of outliers.

The idea is very simple: two of the points are selected randomly; these points define a line. The *support* for this line is measured by the number of points that lie within a distance threshold. This random selection is repeated a number of times and the line with most support is deemed the robust fit. The points within the threshold distance are the inliers (and constitute the eponymous *consensus* set). The intuition is that if one of the points is an outlier then the line will not gain much support, see figure 4.7b.

Furthermore, scoring a line by its support has the additional advantage of favouring better fits. For example, the line $\langle \mathbf{a}, \mathbf{b} \rangle$ in figure 4.7b has a support of 10, whereas the line $\langle \mathbf{a}, \mathbf{d} \rangle$, where the sample points are neighbours, has a support of only 4. Consequently, even though both samples contain no outliers, the line $\langle \mathbf{a}, \mathbf{b} \rangle$ will be selected.

More generally, we wish to fit a *model*, in this case a line, to data, and the random *sample* consists of a minimal subset of the data, in this case two points, sufficient to determine the model. If the model is a planar homography, and the data a set of 2D point correspondences, then the minimal subset consists of four correspondences. The application of RANSAC to the estimation of a homography is described below.

As stated by Fischler and Bolles [Fischler-81] "The RANSAC procedure is opposite to that of conventional smoothing techniques: Rather than using as much of the data as possible to obtain an initial solution and then attempting to eliminate the invalid data points, RANSAC uses as small an initial data set as feasible and enlarges this set with consistent data when possible".

The RANSAC algorithm is summarized in algorithm 4.4. Three questions immediately arise:

Objective

Robust fit of a model to a data set S which contains outliers.

Algorithm

(i) Randomly select a sample of s data points from S and instantiate the model from this subset.
(ii) Determine the set of data points S_i which are within a distance threshold t of the model. The set S_i is the consensus set of the sample and defines the inliers of S.
(iii) If the size of S_i (the number of inliers) is greater than some threshold T, re-estimate the model using all the points in S_i and terminate.
(iv) If the size of S_i is less than T, select a new subset and repeat the above.
(v) After N trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i.

Algorithm 4.4. *The RANSAC robust estimation algorithm, adapted from [Fischler-81]. A minimum of s data points are required to instantiate the free parameters of the model. The three algorithm thresholds t, T, and N are discussed in the text.*

1. What is the distance threshold? We would like to choose the distance threshold, t, such that with a probability α the point is an inlier. This calculation requires the probability distribution for the distance of an inlier from the model. In practice the distance threshold is usually chosen empirically. However, if it is assumed that the measurement error is Gaussian with zero mean and standard deviation σ, then a value for t may be computed. In this case the square of the point distance, $d_\perp^2$, is a sum of squared Gaussian variables and follows a χ_m^2 distribution with m degrees of freedom, where m equals the codimension of the model. For a line the codimension is 1 – only the perpendicular distance to the line is measured. If the model is a point the codimension is 2, and the square of the distance is the sum of squared x and y measurement errors.

The probability that the value of a χ^2_m random variable is less than k^2 is given by the cumulative chi-squared distribution, $F_m(k^2) = \int_0^{k^2} \chi^2_m(\xi)d\xi$. Both of these distributions are described in section A2.2(*p*566). From the cumulative distribution

$$\begin{cases} \text{inlier} & d_\perp^2 < t^2 \\ \text{outlier} & d_\perp^2 \geq t^2 \end{cases} \quad \text{with } t^2 = F_m^{-1}(\alpha)\sigma^2. \tag{4.17}$$

Usually α is chosen as 0.95, so that there is a 95% probability that the point is an inlier. This means that an inlier will only be incorrectly rejected 5% of the time. Values of t for $\alpha = 0.95$ and for the models of interest in this book are tabulated in table 4.2.

Codimension m	Model	t^2
1	line, fundamental matrix	3.84 σ^2
2	homography, camera matrix	5.99 σ^2
3	trifocal tensor	7.81 σ^2

Table 4.2. *The distance threshold $t^2 = F_m^{-1}(\alpha)\sigma^2$ for a probability of $\alpha = 0.95$ that the point (correspondence) is an inlier.*

2. How many samples? It is often computationally infeasible and unnecessary to try every possible sample. Instead the number of samples N is chosen sufficiently high to ensure with a probability, p, that at least one of the random samples of s points is free from outliers. Usually p is chosen at 0.99. Suppose w is the probability that any selected data point is an inlier, and thus $\epsilon = 1 - w$ is the probability that it is an outlier. Then at least N selections (each of s points) are required, where $(1 - w^s)^N = 1 - p$, so that

$$N = \log(1 - p)/\log(1 - (1 - \epsilon)^s). \tag{4.18}$$

Table 4.3 gives examples of N for $p = 0.99$ for a given s and ϵ.

Sample size	Proportion of outliers ϵ						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Table 4.3. *The number N of samples required to ensure, with a probability $p = 0.99$, that at least one sample has no outliers for a given size of sample, s, and proportion of outliers, ϵ.*

Example 4.4. For the line-fitting problem of figure 4.7 there are $n = 12$ data points, of

which two are outliers so that $\epsilon = 2/12 = 1/6$. From table 4.3 for a minimal subset of size $s = 2$, at least $N = 5$ samples are required. This should be compared with the cost of exhaustively trying every point pair, in which case $\binom{12}{2} = 66$ samples are required (the notation $\binom{n}{2}$ means the number of choices of 2 among n, specifically, $\binom{n}{2} = n(n-1)/2$). $\triangle$

Note

(i) The number of samples is linked to the proportion rather than number of outliers. This means that the number of samples required may be smaller than the number of outliers. Consequently the computational cost of the sampling can be acceptable even when the number of outliers is large.

(ii) The number of samples increases with the size of the minimal subset (for a given ϵ and p). It might be thought that it would be advantageous to use more than the minimal subset, three or more points in the case of a line, because then a better estimate of the line would be obtained, and the measured support would more accurately reflect the true support. However, this possible advantage in measuring support is generally outweighed by the severe increase in computational cost incurred by the increase in the number of samples.

3. How large is an acceptable consensus set? A rule of thumb is to terminate if the size of the consensus set is similar to the number of inliers believed to be in the data set, given the assumed proportion of outliers, i.e. for n data points $T = (1-\epsilon)n$. For the line-fitting example of figure 4.7 a conservative estimate of ϵ is $\epsilon = 0.2$, so that $T = (1.0 - 0.2)12 = 10$.

Determining the number of samples adaptively. It is often the case that ϵ, the fraction of data consisting of outliers, is unknown. In such cases the algorithm is initialized using a worst case estimate of ϵ, and this estimate can then be updated as larger consistent sets are found. For example, if the worst case guess is $\epsilon = 0.5$ and a consensus set with 80% of the data is found as inliers, then the updated estimate is $\epsilon = 0.2$.

This idea of "probing" the data via the consensus sets can be applied repeatedly in order to adaptively determine the number of samples, N. To continue the example above, the worst case estimate of $\epsilon = 0.5$ determines an initial N according to (4.18). When a consensus set containing more than 50% of the data is found, we then know that there is at least that proportion of inliers. This updated estimate of ϵ determines a reduced N from (4.18). This update is repeated at each sample, and whenever a consensus set with ϵ lower than the current estimate is found, then N is again reduced. The algorithm terminates as soon as N samples have been performed. It may occur that a sample is found for which ϵ determines an N less than the number of samples that have already been performed. In such a case sufficient samples have been performed and the algorithm terminates. In pseudo-code the adaptive computation of N is summarized in algorithm 4.5.

This adaptive approach works very well and in practice covers the questions of both

- $N = \infty$, sample_count$= 0$.
- While $N >$ sample_count Repeat
  - Choose a sample and count the number of inliers.
  - Set $\epsilon = 1 - (\text{number of inliers})/(\text{total number of points})$
  - Set N from ϵ and (4.18) with $p = 0.99$.
  - Increment the sample_count by 1.
- Terminate.

Algorithm 4.5. *Adaptive algorithm for determining the number of RANSAC samples.*

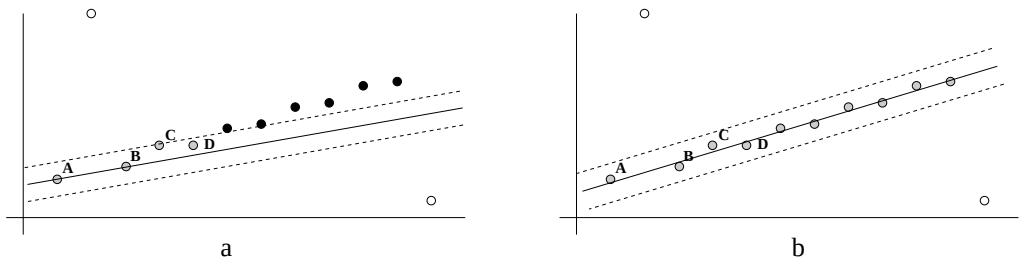


Fig. 4.8. **Robust ML estimation.** *The grey points are classified as inliers to the line. (a) A line defined by points $\langle \mathbf{A}, \mathbf{B} \rangle$ has a support of four (from points $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$). (b) The ML line fit (orthogonal least-squares) to the four points. This is a much improved fit over that defined by $\langle \mathbf{A}, \mathbf{B} \rangle$. 10 points are classified as inliers.*

the number of samples and terminating the algorithm. The initial ϵ can be chosen as 1.0, in which case the initial N will be infinite. It is wise to use a conservative probability p such as 0.99 in (4.18). Table 4.4 on page 127 gives example ϵ's and N's when computing a homography.

4.7.2 Robust Maximum Likelihood estimation

The RANSAC algorithm partitions the data set into inliers (the largest consensus set) and outliers (the rest of the data set), and also delivers an estimate of the model, M_0, computed from the minimal set with greatest support. The final step of the RANSAC algorithm is to re-estimate the model using all the inliers. This re-estimation should be optimal and will involve minimizing a ML cost function, as described in section 4.3. In the case of a line, ML estimation is equivalent to orthogonal regression, and a closed form solution is available. In general, though, the ML estimation involves iterative minimization, and the minimal set estimate, M_0, provides the starting point.

The only drawback with this procedure, which is often the one adopted, is that the inlier–outlier classification is irrevocable. After the model has been optimally fitted to the consensus set, there may well be additional points which would now be classified as inliers if the distance threshold was applied to the new model. For example, suppose the line $\langle \mathbf{A}, \mathbf{B} \rangle$ in figure 4.8 was selected by RANSAC. This line has a support of four points, all inliers. After the optimal fit to these four points, there are now 10 points which would correctly be classified as inliers. These two steps: optimal fit to inliers; re-classify inliers using (4.17); can then be iterated until the number of inliers converges.

A least-squares fit with inliers weighted by their distance to the model is often used at this stage.

Robust cost function. An alternative to minimizing $\mathcal{C} = \sum_i d_{\perp i}^2$ over the inliers is to minimize a robust version including all data. A suitable robust cost function is

$$\mathcal{D} = \sum_i \gamma\left(d_{\perp i}\right) \qquad \text{with } \gamma(e) = \begin{cases} e^2 & e^2 < t^2 \quad \text{inlier} \\ t^2 & e^2 \geq t^2 \quad \text{outlier} \end{cases} \tag{4.19}$$

Here $d_{\perp i}$ are point errors and $\gamma(e)$ is a robust cost function [Huber-81] where outliers are given a fixed cost. The χ^2 motivation for the threshold is the same as that of (4.17), where t^2 is defined. The quadratic cost for inliers arises from the Gaussian error model, as described in section 4.3. The constant cost for outliers in the robust cost function arises from the assumption that outliers follow a diffuse or uniform distribution, the log-likelihood of which is a constant. It might be thought that outliers could be excluded from the cost function by simply thresholding on $d_{\perp i}$. The problem with thresholding alone is that it would result in only outliers being included because they would incur no cost.

The cost function $\mathcal{D}$ allows the minimization to be conducted on all points whether they are outliers or inliers. At the start of the iterative minimization $\mathcal{D}$ differs from $\mathcal{C}$ only by a constant (given by 4 times the number of outliers). However, as the minimization progresses outliers can be redesignated inliers, and this typically occurs in practice. A discussion and comparison of cost functions is given in appendix A6.8-(*p*616).

4.7.3 Other robust algorithms

In RANSAC a model instantiated from a minimal set is scored by the number of data points within a threshold distance. An alternative is to score the model by the median of the distances to all points in the data. The model with least median is then selected. This is Least Median of Squares (LMS) estimation, where, as in RANSAC, minimum size subset samples are selected randomly with the number of samples obtained from (4.18). The advantage of LMS is that it requires *no* setting of thresholds or *a priori* knowledge of the variance of the error. The disadvantage of LMS is that it fails if more than half the data is outlying, for then the median distance will be to an outlier. The solution is to use the proportion of outliers to determine the selection distance. For example if there are 50% outliers then a distance below the median value (the quartile say) should be used.

Both the RANSAC and LMS algorithms are able to cope with a large proportion of outliers. If the number of outliers is small, then other robust methods may well be more efficient. These include case deletion, where each point in turn is deleted and the model fitted to the remaining data; and iterative weighted least-squares, where a data point's influence on the fit is weighted inversely by its residual. Generally these methods are **not** recommended. Both Torr [Torr-95b] and Xu and Zhang [Xu-96] describe and compare various robust estimators for estimating the fundamental matrix.

Objective

Compute the 2D homography between two images.

Algorithm

(i) **Interest points:** Compute interest points in each image.
(ii) **Putative correspondences:** Compute a set of interest point matches based on proximity and similarity of their intensity neighbourhood.
(iii) **RANSAC robust estimation:** Repeat for N samples, where N is determined adaptively as in algorithm 4.5:
  (a) Select a random sample of 4 correspondences and compute the homography H.
  (b) Calculate the distance $d_\perp$ for each putative correspondence.
  (c) Compute the number of inliers consistent with H by the number of correspondences for which $d_\perp < t = \sqrt{5.99}\,\sigma$ pixels.

  Choose the H with the largest number of inliers. In the case of ties choose the solution that has the lowest standard deviation of inliers.
(iv) **Optimal estimation:** re-estimate H from all correspondences classified as inliers, by minimizing the ML cost function (4.8–*p*95) using the Levenberg–Marquardt algorithm of section A6.2(*p*600).
(v) **Guided matching:** Further interest point correspondences are now determined using the estimated H to define a search region about the transferred point position.

The last two steps can be iterated until the number of correspondences is stable.

Algorithm 4.6. *Automatic estimation of a homography between two images using RANSAC.*

4.8 Automatic computation of a homography

This section describes an algorithm to automatically compute a homography between two images. The input to the algorithm is simply the images, with no other *a priori* information required; and the output is the estimated homography together with a set of interest points in correspondence. The algorithm might be applied, for example, to two images of a planar surface or two images acquired by rotating a camera about its centre.

The first step of the algorithm is to compute interest points in each image. We are then faced with a "chicken and egg" problem: once the correspondence between the interest points is established the homography can be computed; conversely, given the homography the correspondence between the interest points can easily be established. This problem is resolved by using robust estimation, here RANSAC, as a "search engine". The idea is first to obtain by some means a set of putative point correspondences. It is expected that a proportion of these correspondences will in fact be mismatches. RANSAC is designed to deal with exactly this situation – estimate the homography and also a set of inliers consistent with this estimate (the true correspondences), and outliers (the mismatches).

The algorithm is summarized in algorithm 4.6, with an example of its use shown in figure 4.9, and the steps described in more detail below. Algorithms with essentially the same methodology enable the automatic computation of the fundamental matrix and trifocal tensor directly from image pairs and triplets respectively. This computation is described in chapter 11 and chapter 16.

Determining putative correspondences. The aim, in the absence of any knowledge of the homography, is to provide an initial point correspondence set. A good proportion of these correspondences should be correct, but the aim is not perfect matching, since RANSAC will later be used to eliminate the mismatches. Think of these as "seed" correspondences. These putative correspondences are obtained by detecting interest points independently in each image, and then matching these interest points using a combination of proximity and similarity of intensity neighbourhoods as follows. For brevity, the interest points will be referred to as 'corners'. However, these corners need not be images of physical corners in the scene. The corners are defined by a minimum of the image auto-correlation function.

For each corner at (x, y) in image 1 the match with highest neighbourhood cross-correlation in image 2 is selected within a square search region centred on (x, y). Symmetrically, for each corner in image 2 the match is sought in image 1. Occasionally there will be a conflict where a corner in one image is "claimed" by more than one corner in the other. In such cases a "winner takes all" scheme is applied and only the match with highest cross-correlation is retained.

A variation on the similarity measure is to use Squared Sum of intensity Differences (SSD) instead of (normalized) Cross-Correlation (CC). CC is invariant to the affine mapping of the intensity values (i.e. $I \mapsto \alpha I + \beta$, scaling plus offset) which often occurs in practice between images. SSD is not invariant to this mapping. However, SSD is often preferred when there is small variation in intensity between images, because it is a more sensitive measure than CC and is computationally cheaper.

RANSAC for a homography. The RANSAC algorithm is applied to the putative correspondence set to estimate the homography and the (inlier) correspondences which are consistent with this estimate. The sample size is four, since four correspondences determine a homography. The number of samples is set adaptively as the proportion of outliers is determined from each consensus set, as described in algorithm 4.5.

There are two issues: what is the "distance" in this case? and how should the samples be selected?

(i) **Distance measure**: The simplest method of assessing the error of a correspondence from a homography $\mathtt{H}$ is to use the symmetric transfer error, i.e. $d^2_{\text{transfer}} = d(\mathbf{x}, \mathtt{H}^{-1}\mathbf{x}')^2 + d(\mathbf{x}', \mathtt{H}\mathbf{x})^2$, where $\mathbf{x} \leftrightarrow \mathbf{x}'$ is the point correspondence. A better, though more expensive, distance measure is the reprojection error, $d^2_\perp = d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2$, where $\hat{\mathbf{x}}' = \mathtt{H}\hat{\mathbf{x}}$ is the perfect correspondence. This measure is more expensive because $\hat{\mathbf{x}}$ must also be computed. A further alternative is Sampson error.

(ii) **Sample selection**: There are two issues here. First, degenerate samples should be disregarded. For example, if three of the four points are collinear then a homography cannot be computed; second, the sample should consist of points with a good spatial distribution over the image. This is because of the extrapolation problem – an estimated homography will accurately map the region straddled by the computation points, but the accuracy generally deteriorates

with distance from this region (think of four points in the very top corner of the image). Distributed spatial sampling can be implemented by tiling the image and ensuring, by a suitable weighting of the random sampler, that samples with points lying in different tiles are the more likely.

Robust ML estimation and guided matching. The aim of this final stage is twofold: first, to obtain an improved estimate of the homography by using all the inliers in the estimation (rather than only the four points of the sample); second, to obtain more inlying matches from the putative correspondence set because a more accurate homography is available. An improved estimate of the homography is then computed from the inliers by minimizing an ML cost function. This final stage can be implemented in two ways. One way is to carry out an ML estimation on the inliers, then recompute the inliers using the new estimated H, and repeat this cycle until the number of inliers converges. The ML cost function minimization is carried out using the Levenberg–Marquardt algorithm described in section A6.2(*p*600). The alternative is to estimate the homography and inliers simultaneously by minimizing a robust ML cost function of (4.19) as described in section 4.7.2. The disadvantage of the simultaneous approach is the computational effort incurred in the minimization of the cost function. For this reason the cycle approach is usually the more attractive.

4.8.1 Application domain

The algorithm requires that interest points can be recovered fairly uniformly across the image, and this in turn requires scenes and resolutions which support this requirement. Scenes should be lightly textured – images of blank walls are not ideal.

The search window proximity constraint places an upper limit on the image motion of corners (the *disparity*) between views. However, the algorithm is not defeated if this constraint is not applied, and in practice the main role of the proximity constraint is to reduce computational complexity, as a smaller search window means that fewer corner matches must be evaluated.

Ultimately the scope of the algorithm is limited by the success of the corner neighbourhood similarity measure (SSD or CC) in providing disambiguation between correspondences. Failure generally results from lack of spatial invariance: the measures are only invariant to image translation, and are severely degraded by transformations outside this class such as image rotation or significant differences in foreshortening between images. One solution is to use measures with a greater invariance to the homography mapping between images, for example measures which are rotationally invariant. An alternative solution is to use an initial estimate of the homography to map between intensity neighbourhoods. Details are beyond the scope of this discussion, but are provided in [Pritchett-98, Schmid-98]. The use of robust estimation confers moderate immunity to independent motion, changes in shadows, partial occlusions etc.

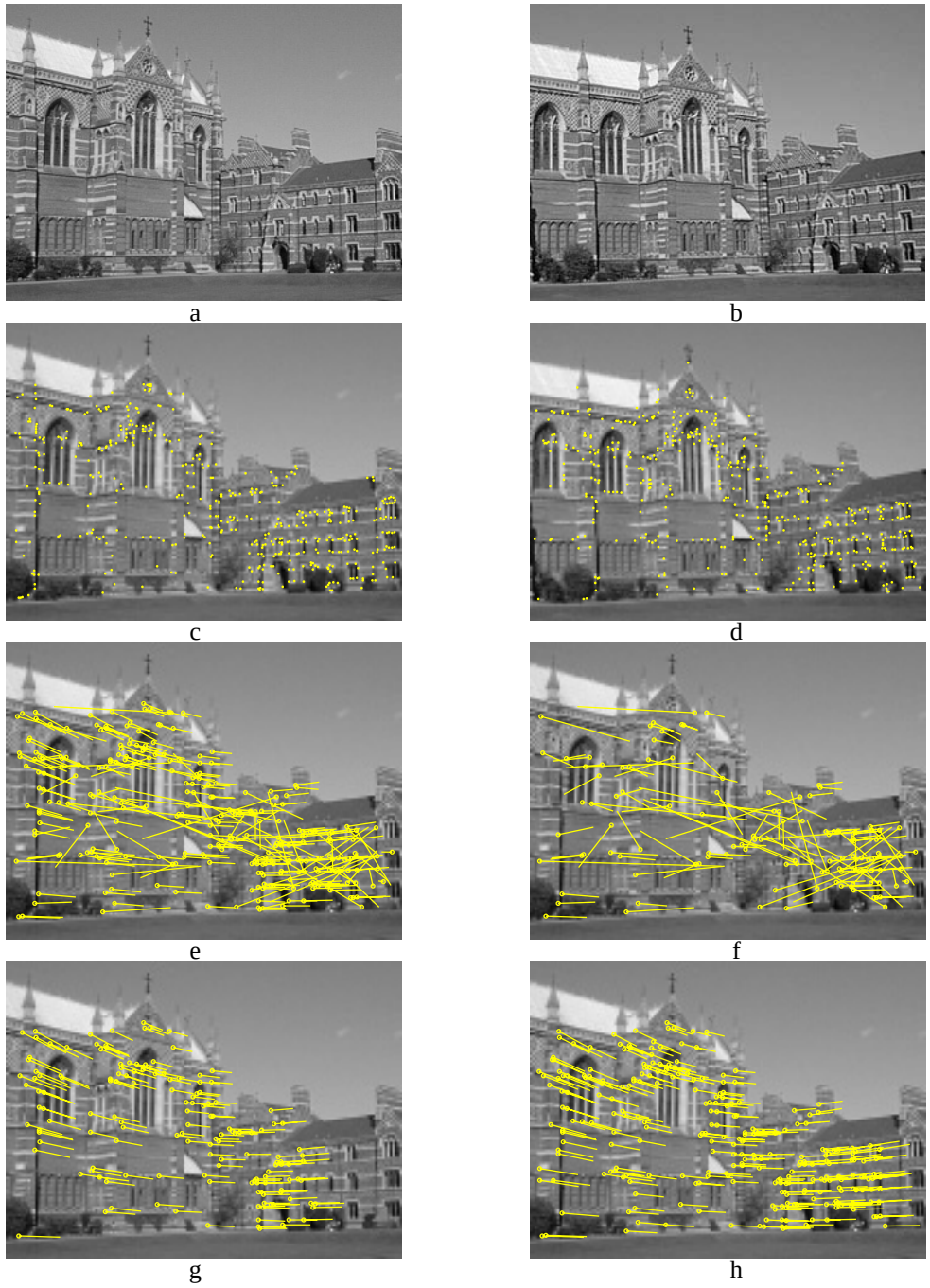


Fig. 4.9. **Automatic computation of a homography between two images using RANSAC.** *The motion between views is a rotation about the camera centre so the images are exactly related by a homography. (a) (b) left and right images of Keble College, Oxford. The images are* 640×480 *pixels. (c) (d) detected corners superimposed on the images. There are approximately 500 corners on each image. The following results are superimposed on the left image: (e) 268 putative matches shown by the line linking corners, note the clear mismatches; (f) outliers – 117 of the putative matches; (g) inliers – 151 correspondences consistent with the estimated* `H`*; (h) final set of 262 correspondences after guided matching and MLE.*

4.8.2 Implementation and run details

Interest points are obtained using the Harris [Harris-88] corner detector. This detector localizes corners to sub-pixel accuracy, and it has been found empirically that the correspondence error is usually less than a pixel [Schmid-98].

When obtaining seed correspondences, in the putative correspondence stage of the algorithm, the threshold on the neighbourhood similarity measure for match acceptance is deliberately conservative to minimize incorrect matches (the SSD threshold is 20). For the guided matching stage this threshold is relaxed (it is doubled) so that additional putative correspondences are available.

Number of inliers	$1-\epsilon$	Adaptive N
6	2%	20,028,244
10	3%	2,595,658
44	16%	6,922
58	21%	2,291
73	26%	911
151	56%	43

Table 4.4. *The results of the adaptive algorithm 4.5 used during RANSAC to compute the homography for figure 4.9. N is the total number of samples required as the algorithm runs for $p = 0.99$ probability of no outliers in the sample. The algorithm terminated after 43 samples.*

For the example of figure 4.9 the images are 640×480 pixels, and the search window ± 320 pixels, i.e. the entire image. Of course a much smaller search window could have been used given the actual point disparities in this case. Often in video sequences a search window of ± 40 pixels suffices (i.e. a square of side 80 centred on the current position). The inlier threshold was $t = 1.25$ pixels.

A total of 43 samples were required, with the sampling run as shown in table 4.4. The guided matching required two iterations of the MLE–inlier classification cycle. The RMS values for $d_\perp$ pixel error were 0.23 before the MLE and 0.19 after. The Levenberg–Marquardt algorithm required 10 iterations.

4.9 Closure

This chapter has illustrated the issues and techniques that apply to estimating the tensors representing multiple view relations. These ideas will reoccur in each of the computation chapters throughout the book. In each case there are a minimal number of correspondences required; degenerate configurations that should be avoided; algebraic and geometric errors that can be minimized when more than the minimal number of correspondences are available; parametrizations that enforce internal constraints on the tensor etc.

4.9.1 The literature

The DLT algorithm dates back at least to Sutherland [Sutherland-63]. Sampson's classic paper on conic fitting (an improvement on the equally classic Bookstein algorithm)

appeared in [Sampson-82]. Normalization was made public in the Computer Vision literature by Hartley [Hartley-97c].

Related reading on numerical methods may be found in the excellent *Numerical Recipes in C* [Press-88], and also Gill and Murray [Gill-78] for iterative minimization.

Fischler and Bolles' [Fischler-81] RANSAC was one of the earliest robust algorithms, and in fact was developed to solve a Computer Vision problem (pose from 3 points). The original paper is very clearly argued and well worth reading. Other background material on robustness may be found in Rousseeuw [Rousseeuw-87]. The primary application of robust estimation in computer vision was to estimating the fundamental matrix (chapter 11), by Torr and Murray [Torr-93] using RANSAC, and, Zhang *et al.* [Zhang-95] using LMS. The automatic ML estimation of a homography was described by Torr and Zisserman [Torr-98].

4.9.2 Notes and exercises

(i) **Computing homographies of $\mathbb{P}^n$.** The derivation of (4.1–*p*89) and (4.3–*p*89) assumed that the dimension of $\mathbf{x}'_i$ is three, so that the cross-product is defined. However, (4.3) may be derived in a way that generalizes to all dimensions. Assuming that $w'_i = 1$, we may solve for the unknown scale factor explicitly by writing $\mathtt{H}\mathbf{x}_i = k(x_i, y_i, 1)^\mathsf{T}$. From the third coordinate we obtain $k = \mathbf{h}^{3\mathsf{T}}\mathbf{x}_i$, and substituting this into the original equation gives

$$\begin{pmatrix} \mathbf{h}^{1\mathsf{T}}\mathbf{x}_i \\ \mathbf{h}^{2\mathsf{T}}\mathbf{x}_i \end{pmatrix} = \begin{pmatrix} x'_i\mathbf{h}^{3\mathsf{T}}\mathbf{x}_i \\ y'_i\mathbf{h}^{3\mathsf{T}}\mathbf{x}_i \end{pmatrix}$$

which leads directly to (4.3).

(ii) **Computing homographies for ideal points.** If one of the points $\mathbf{x}'_i$ is an ideal point, so that $w'_i = 0$, then the pair of equations (4.3) collapses to a single equation although (4.1) does contain two independent equations. To avoid such degeneracy, while including only the minimum number of equations, a good way to proceed is as follows. We may rewrite the equation $\mathbf{x}'_i = \mathtt{H}\mathbf{x}_i$ as

$$[\mathbf{x}'_i]^\perp \mathtt{H}\mathbf{x}_i = \mathbf{0}$$

where $[\mathbf{x}'_i]^\perp$ is a matrix with rows orthogonal to $\mathbf{x}'_i$ so that $[\mathbf{x}'_i]^\perp\mathbf{x}'_i = \mathbf{0}$. Each row of $[\mathbf{x}'_i]^\perp$ leads to a separate linear equation in the entries of $\mathtt{H}$. The matrix $[\mathbf{x}'_i]^\perp$ may be obtained by deleting the first row of an orthogonal matrix $\mathtt{M}$ satisfying $\mathtt{M}\mathbf{x}'_i = (1, 0, \ldots, 0)^\mathsf{T}$. A Householder matrix (see section A4.1.2(*p*580)) is an easily constructed matrix with the desired property.

(iii) **Scaling unbounded point sets.** In the case of points at or near infinity in a plane, it is neither reasonable nor feasible to normalize coordinates using the isotropic (or non-isotropic) scaling schemes presented in this chapter, since the centroid and scale are infinite or near infinite. A method that seems to give good results is to normalize the set of points $\mathbf{x}_i = (x_i, y_i, w_i)^\mathsf{T}$ such that

$$\sum_i x_i = \sum_i y_i = 0 \;\; ; \;\; \sum_i x_i^2 + y_i^2 = 2\sum_i w_i^2 \;\; ; \;\; x_i^2 + y_i^2 + w_i^2 = 1 \forall i$$

Note that the coordinates x_i and y_i appearing here are the homogeneous coordinates, and the conditions no longer imply that the centroid is at the origin. Investigate methods of achieving this normalization, and evaluate its properties.

(iv) **Transformation invariance of DLT.** We consider computation of a 2D homography by minimizing algebraic error $\|\mathtt{A}\mathbf{h}\|$ (see (4.5–*p*94)) subject to various constraints. Prove the following cases:

(a) If $\|\mathtt{A}\mathbf{h}\|$ is minimized subject to the constraint $h_9 = \mathtt{H}_{33} = 1$, then the result is invariant under change of scale but *not* translation of coordinates.

(b) If instead the constraint is $\mathtt{H}_{31}^2 + \mathtt{H}_{32}^2 = 1$ then the result is similarity invariant.

(c) **Affine case:** The same is true for the constraint $\mathtt{H}_{31} = \mathtt{H}_{32} = 0;\ \mathtt{H}_{33} = 1$.

(v) **Expressions for image coordinate derivatives.** For the map $\mathbf{x}' = (x', y', w')^\mathsf{T} = \mathtt{H}\mathbf{x}$, derive the following expressions (where $\tilde{\mathbf{x}}' = (\tilde{x}', \tilde{y}')^\mathsf{T} = (x'/w', y'/w')^\mathsf{T}$ are the inhomogeneous coordinates of the image point):

(a) Derivative wrt $\mathbf{x}$

$$\partial\tilde{\mathbf{x}}'/\partial\mathbf{x} = \frac{1}{w'}\begin{bmatrix} \mathbf{h}^{1\mathsf{T}} - \tilde{x}'\mathbf{h}^{3\mathsf{T}} \\ \mathbf{h}^{2\mathsf{T}} - \tilde{y}'\mathbf{h}^{3\mathsf{T}} \end{bmatrix} \tag{4.20}$$

where $\mathbf{h}^{j\mathsf{T}}$ is the $j-$th row of $\mathtt{H}$.

(b) Derivative wrt $\mathtt{H}$

$$\partial\tilde{\mathbf{x}}'/\partial\mathbf{h} = \frac{1}{w'}\begin{bmatrix} \mathbf{x}^\mathsf{T} & 0 & -\tilde{x}'\mathbf{x}^\mathsf{T} \\ 0 & \mathbf{x}^\mathsf{T} & -\tilde{y}'\mathbf{x}^\mathsf{T} \end{bmatrix} \tag{4.21}$$

with $\mathbf{h}$ as defined in (4.2–*p*89).

(vi) **Sampson error with non-isotropic error distributions.** The derivation of Sampson error in section 4.2.6(*p*98) assumed that points were measured with circular error distributions. In the case where the point $\mathbf{X} = (x, y, x', y')$ is measured with covariance matrix $\Sigma_\mathbf{X}$ it is appropriate instead to minimize the Mahalanobis norm $\|\boldsymbol{\delta}_\mathbf{X}\|^2_{\Sigma_\mathbf{X}} = \boldsymbol{\delta}_\mathbf{X}^\mathsf{T}\Sigma_\mathbf{X}^{-1}\boldsymbol{\delta}_\mathbf{X}$. Show that in this case the formulae corresponding to (4.11–*p*99) and (4.12–*p*99) are

$$\boldsymbol{\delta}_\mathbf{X} = -\Sigma_\mathbf{X}\mathtt{J}^\mathsf{T}(\mathtt{J}\Sigma_\mathbf{X}\mathtt{J}^\mathsf{T})^{-1}\boldsymbol{\epsilon} \tag{4.22}$$

and

$$\|\boldsymbol{\delta}_\mathbf{X}\|^2_{\Sigma_\mathbf{X}} = \boldsymbol{\epsilon}^\mathsf{T}(\mathtt{J}\Sigma_\mathbf{X}J^\mathsf{T})^{-1}\boldsymbol{\epsilon}. \tag{4.23}$$

Note that if the measurements in the two images are independent, then the covariance matrix $\Sigma_\mathbf{X}$ will be block-diagonal with two 2×2 diagonal blocks corresponding to the two images.

(vii) **Sampson error programming hint.** In the case of 2D homography estimation, and in fact every other similar problem considered in this book, the cost function $\mathcal{C}_\mathtt{H}(\mathbf{X}) = \mathtt{A}(\mathbf{X})\mathbf{h}$ of section 4.2.6(*p*98) is multilinear in the coordinates

Objective

Given $n \geq 4$ image point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, determine the affine homography $\mathtt{H}_{\mathrm{A}}$ which minimizes reprojection error in both images (4.8–*p*95).

Algorithm

(a) Express points as inhomogeneous 2-vectors. Translate the points $\mathbf{x}_i$ by a translation $\mathbf{t}$ so that their centroid is at the origin. Do the same to the points $\mathbf{x}'_i$ by a translation $\mathbf{t}'$. Henceforth work with the translated coordinates.

(b) Form the $n \times 4$ matrix $\mathtt{A}$ whose rows are the vectors

$$\mathbf{X}_i^{\mathsf{T}} = (\mathbf{x}_i^{\mathsf{T}}, \mathbf{x}_i'^{\mathsf{T}}) = (x_i, y_i, x'_i, y'_i).$$

(c) Let $\mathbf{V}_1$ and $\mathbf{V}_2$ be the right singular-vectors of $\mathtt{A}$ corresponding to the two largest (sic) singular values.

(d) Let $\mathtt{H}_{2\times 2} = \mathtt{C}\mathtt{B}^{-1}$, where $\mathtt{B}$ and $\mathtt{C}$ are the 2×2 blocks such that

$$[\mathbf{V}_1 \mathbf{V}_2] = \begin{bmatrix} \mathtt{B} \\ \mathtt{C} \end{bmatrix}.$$

(e) The required homography is

$$\mathtt{H}_{\mathrm{A}} = \begin{bmatrix} \mathtt{H}_{2\times 2} & \mathtt{H}_{2\times 2}\mathbf{t} - \mathbf{t}' \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix},$$

and the corresponding estimate of the image points is given by

$$\widehat{\mathbf{X}}_i = (\mathbf{V}_1 \mathbf{V}_1^{\mathsf{T}} + \mathbf{V}_2 \mathbf{V}_2^{\mathsf{T}}) \mathbf{X}_i$$

Algorithm 4.7. *The Gold Standard Algorithm for estimating an* affine *homography* $\mathtt{H}_{\mathrm{A}}$ *from image correspondences.*

of $\mathbf{X}$. This means that the partial derivative $\partial \mathcal{C}_{\mathtt{H}}(\mathbf{X})/\partial \mathbf{X}$ may be very simply computed. For instance, the derivative

$$\partial \mathcal{C}_{\mathtt{H}}(x, y, x', y')/\partial x = \mathcal{C}_{\mathtt{H}}(x+1, y, x', y') - \mathcal{C}_{\mathtt{H}}(x, y, x', y')$$

is *exact*, not a finite difference approximation. This means that for programming purposes, one does not need to code a special routine for taking derivatives – the routine for computing $\mathcal{C}_{\mathtt{H}}(\mathbf{X})$ will suffice. Denoting by $\mathbf{E}_i$ the vector containing 1 in the i-th position, and otherwise 0, one sees that $\partial \mathcal{C}_{\mathtt{H}}(\mathbf{X})/\partial X_i = \mathcal{C}_{\mathtt{H}}(\mathbf{X} + \mathbf{E}_i) - \mathcal{C}_{\mathtt{H}}(\mathbf{X})$, and further

$$\mathtt{J}\mathtt{J}^{\mathsf{T}} = \sum_i \left(\mathcal{C}_{\mathtt{H}}(\mathbf{X} + \mathbf{E}_i) - \mathcal{C}_{\mathtt{H}}(\mathbf{X})\right) \left(\mathcal{C}_{\mathtt{H}}(\mathbf{X} + \mathbf{E}_i) - \mathcal{C}_{\mathtt{H}}(\mathbf{X})\right)^{\mathsf{T}}.$$

Also note that computationally it is more efficient to solve $\mathtt{J}\mathtt{J}^{\mathsf{T}}\boldsymbol{\lambda} = -\boldsymbol{\epsilon}$ directly for $\boldsymbol{\lambda}$, rather than take the inverse as $\boldsymbol{\lambda} = -(\mathtt{J}\mathtt{J}^{\mathsf{T}})^{-1}\boldsymbol{\epsilon}$.

(viii) **Minimizing geometric error for affine transformations.** Given a set of correspondences $(x_i, y_i) \leftrightarrow (x'_i, y'_i)$, find an affine transformation $\mathtt{H}_{\mathrm{A}}$ that minimizes geometric error (4.8–*p*95). We will step through the derivation of a linear algorithm based on Sampson's approximation which is exact in this case. The complete method is summarized in algorithm 4.7.

(a) Show that the optimum affine transformation takes the centroid of the $\mathbf{x}_i$ to the centroid of $\mathbf{x}'_i$, so by translating the points to have their centroid at the origin, the translation part of the transformation is determined. It is only necessary then to determine the upper-left 2×2 submatrix $\mathtt{H}_{2\times 2}$ of $\mathtt{H}_{\mathrm{A}}$, which represents the linear part of the transformation.

(b) The point $\mathbf{X}_i = (\mathbf{x}_i^\mathsf{T}, \mathbf{x}_i'^\mathsf{T})^\mathsf{T}$ lies on $\mathcal{V}_\mathtt{H}$ if and only if $[\mathtt{H}_{2\times 2}| - \mathtt{I}_{2\times 2}]\mathbf{X} = \mathbf{0}$. So $\mathcal{V}_\mathtt{H}$ is a codimension-2 subspace of $\mathbb{R}^4$.

(c) Any codimension-2 subspace may be expressed as $[\mathtt{H}_{2\times 2}| - \mathtt{I}]\mathbf{X} = \mathbf{0}$ for suitable $\mathtt{H}_{2\times 2}$. Thus given measurements $\mathbf{X}_i$, the estimation task is equivalent to finding the best-fitting codimension-2 subspace.

(d) Given a matrix $\mathtt{M}$ with rows $\mathbf{X}_i^\mathsf{T}$, the best-fitting subspace to the $\mathbf{X}_i$ is spanned by the singular vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ corresponding to the two largest singular values of $\mathtt{M}$.

(e) The $\mathtt{H}_{2\times 2}$ corresponding to the subspace spanned by $\mathbf{V}_1$ and $\mathbf{V}_2$ is found by solving the equations $[\mathtt{H}_{2\times 2}| - \mathtt{I}][\mathbf{V}_1\mathbf{V}_2] = \mathtt{0}$.

(ix) **Computing homographies of $\mathbb{P}^3$ from line correspondences.** Consider computing a 4×4 homography $\mathtt{H}$ from lines correspondences alone, assuming the lines are in general position in $\mathbb{P}^3$. There are two questions: how many correspondences are required?, and how to formulate the algebraic constraints to obtain a solution for $\mathtt{H}$? It might be thought that four line correspondences would be sufficient because each line in $\mathbb{P}^3$ has four degrees of freedom, and thus four lines should provide $4 \times 4 = 16$ constraints on the 15 degrees of freedom of $\mathtt{H}$. However, a configuration of four lines is degenerate (see section 4.1.3(*p*91)) for computing the transformation, as there is a 2D isotropy subgroup. This is discussed further in [Hartley-94c]. Equations linear in $\mathtt{H}$ can be obtained in the following way:

$$\boldsymbol{\pi}_i^\mathsf{T}\mathtt{H}\mathbf{X}_j = 0\,, \quad i = 1, 2,\ j = 1, 2\,,$$

where $\mathtt{H}$ transfers a line defined by the two points $(\mathbf{X}_1, \mathbf{X}_2)$ to a line defined by the intersection of the two planes $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$. This method was derived in [Oskarsson-02], where more details are to be found.