2020 Summer Seminar

Multiple View Geometry

Chapter 4: Estimation - 2D Projective Transformations

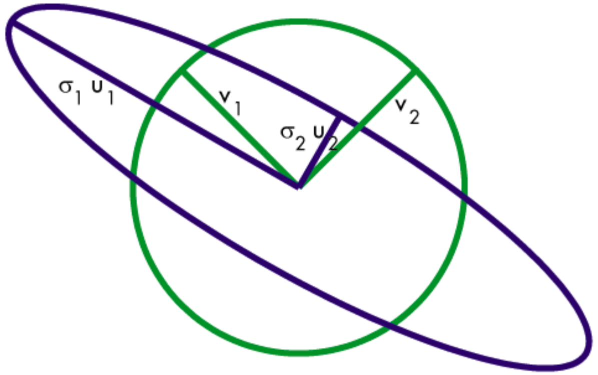
Sohee Lim

2020/8/5

Contents

4. Estimation – 2D Projective Transformations

SVD (Singular value Decomposition)

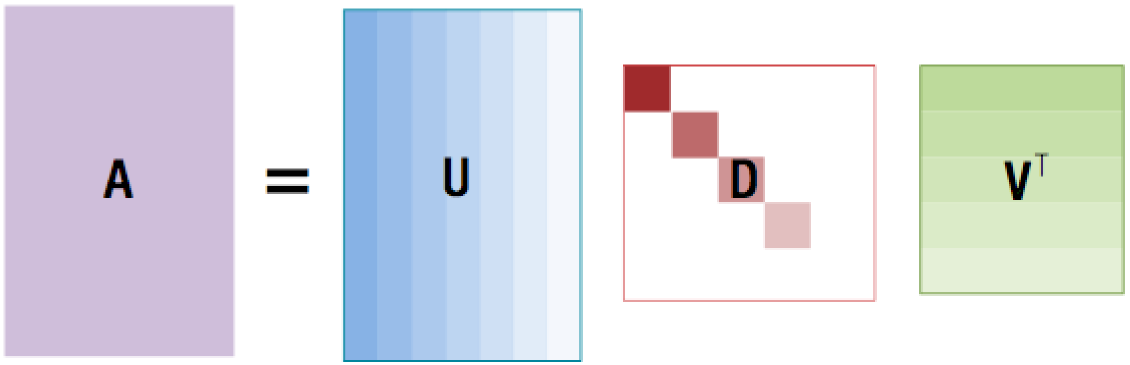


$$\mathbf{AV} = \mathbf{DU}$$

$$\mathbf{UU}^{\mathrm{T}} = \mathbf{I}, \quad \mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}$$

$\mathbf{U}$ = eigen vectors of $\mathbf{AA}^{\mathbf{T}}$
$\mathbf{V}$ = eigenvectors of $\mathbf{A}^{\mathbf{T}}\mathbf{A}$

$$\mathbf{A} = \mathbf{UDV}^{\mathrm{T}} \qquad \mathbf{D} = diag\left\{\sigma_1, \sigma_2, \cdots, \sigma_n\right\} \qquad \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$$



$$= \sigma_1 u_1 v_1^{\ T} + \sigma_2 u_2 v_2^{\ T} + \cdots + \sigma_n u_n v_n^{\ T}$$

SVD (Singular value Decomposition)

$$\mathbf{Ax} \;=\; \mathbf{UDV^T x} = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^{\mathrm{T}} \mathbf{x} + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^{\mathrm{T}} \mathbf{x} + \cdots + \sigma_n \mathbf{u}_n \mathbf{v}_n^{\mathrm{T}} \mathbf{x}$$

- Which $\mathbf{x}$ makes $\|\mathbf{Ax}\|$ to be minimum subject to $\|\mathbf{x}\| = 1$?

  if $\sigma_i \neq 0,\ i = 1, \cdots, n$
  $\Rightarrow \mathbf{x} = \mathbf{v}_n$ (the last eigen vector)
  $\Rightarrow$ the minimum is σ_n

  if $\sigma_i = 0,\ i = 1, \cdots, n$
  $\Rightarrow \mathbf{x} =$ null space of $\mathbf{A}$
  $\Rightarrow \mathbf{Ax} = 0$

Parameter Estimation

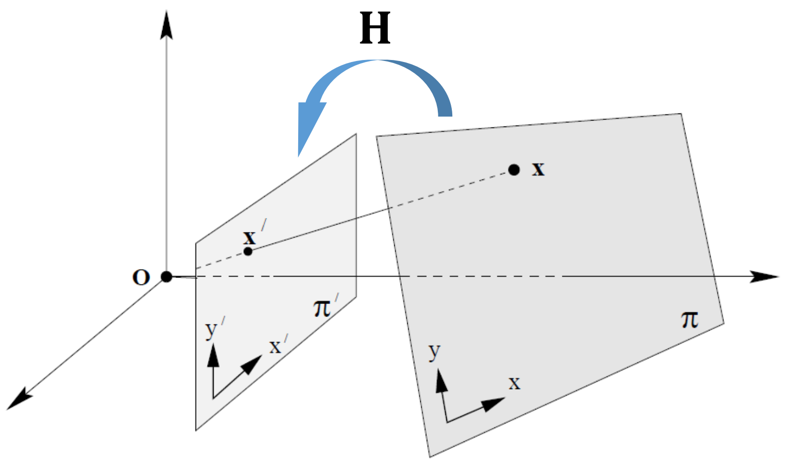
- We are interested in the following problems:
  - **2D homography**

    Given a set of $(\mathbf{x_i}, \mathbf{x_i'})$, how to determine $\mathbf{H}$ $(\mathbf{x}_i' = \mathbf{H}\mathbf{x}_i)$
  - **3d to 2D camera projection**

    Given a set of $(\mathbf{X_i}, \mathbf{x_i})$, how to determine $\mathbf{P}$ $(\mathbf{x_i} = \mathbf{P}\mathbf{X_i})$
  - **Fundamental matrix**

    Given a set of $(\mathbf{x_i}, \mathbf{x_i'})$, how to determine $\mathbf{F}$ $\left(\mathbf{x_i'}^{\mathbf{T}}\mathbf{F}\mathbf{x_i} = \mathbf{0}\right)$
  - **Trifocal tensor**

    Given a set of $(\mathbf{x_i}, \mathbf{x_i'}, \mathbf{x_i''})$, how to determine $\mathbf{T}$



<Homography>

Number of measurements required

- To estimate the parameters, we need
  # of independent equations ≥ degrees of freedom

- Example:

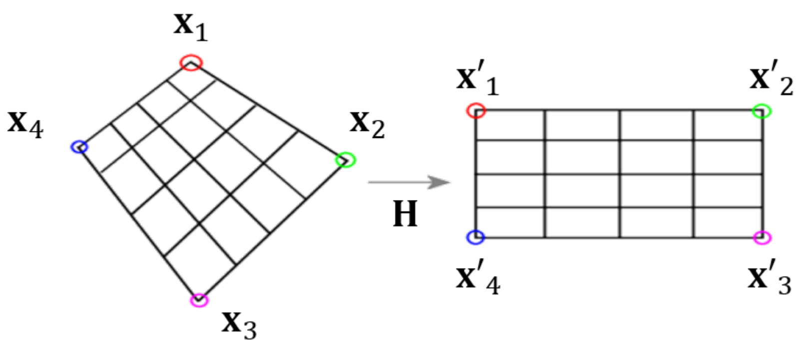
To estimate $\mathbf{H}$: $\mathbf{x}_i' = \mathbf{H}\mathbf{x}_i \quad (\mathbf{x}_i' \sim w\mathbf{x}_i' \sim \mathbf{H}\mathbf{x}_i)$

a point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$

$$\begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} \sim \begin{bmatrix} wx_i' \\ wy_i' \\ w \end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$x_i' = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}}$$

$$y_i' = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}}$$



2 independent equations per a point

✓ $\mathbf{H}$ has 8 DOF

✓ $4 \times 2 \geq 8$

✓ So, at least 4 points correspondences are required (where no 3 points are collinear)

Approximate solutions

- Minimal solution
  - ✓ 4 points yield an exact solution for **H**

- If more points…
  - ✓ No exact solution, because measurements are inexact (“noise”)
  - ✓ Find the *optimal solution* according to some cost function
  - ✓ Algebraic or geometric/statistical cost

Gold Standard algorithm

- Cost function that is optimal for some assumptions
- Computational algorithm that minimizes it is called “Gold Standard” algorithm
- Other algorithms can then be compared to it

4.1 The Direct Linear Transformation (DLT) algorithm

- $\mathbf{x}_i' = \mathbf{H}\mathbf{x}_i$ may be expressed in terms of the vector cross product as $\mathbf{x}_i' \times \mathbf{H}\mathbf{x}_i = 0$

"same directionality"

- Let $\mathbf{H}\mathbf{x}_i = \begin{bmatrix} \mathbf{h}^{1^{\mathrm{T}}}\mathbf{x}_i \\ \mathbf{h}^{2^{\mathrm{T}}}\mathbf{x}_i \\ \mathbf{h}^{3^{\mathrm{T}}}\mathbf{x}_i \end{bmatrix}$ and $\mathbf{x}_i' = (x_i', y_i', w_i')^{\mathrm{T}}$

The j-th row of the matrix $\mathbf{H}$ is denoted by $\mathbf{h}^{j\,\mathrm{T}}$

Then

$$\mathbf{x}_i' \times \mathbf{H}\mathbf{x}_i = \begin{bmatrix} y_i' \mathbf{h}^{3\mathrm{T}}\mathbf{x}_i - w' \mathbf{h}^{2\mathrm{T}}\mathbf{x}_i \\ w_i' \mathbf{h}^{1\mathrm{T}}\mathbf{x}_i - x_i \mathbf{h}^{3\mathrm{T}}\mathbf{x}_i \\ x_i \mathbf{h}^{2\mathrm{T}}\mathbf{x}_i - y_i' \mathbf{h}^{1\mathrm{T}}\mathbf{x}_i \end{bmatrix} = \mathbf{0}$$

$$\Longrightarrow \begin{bmatrix} \mathbf{0}^{\mathrm{T}} & -w_i'\mathbf{x}_i^{\mathrm{T}} & y_i'\mathbf{x}_i^{\mathrm{T}} \\ w_i'\mathbf{x}_i^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & -x_i'\mathbf{x}_i^{\mathrm{T}} \\ -y_i'\mathbf{x}_i^{\mathrm{T}} & x_i'\mathbf{x}_i^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{bmatrix} = \mathbf{0}$$

3rd row is redundant,
only 2 are linearly independent

$$\Longrightarrow \begin{bmatrix} \mathbf{0}^{\mathrm{T}} & -w_i'\mathbf{x}_i^{\mathrm{T}} & y_i'\mathbf{x}_i^{\mathrm{T}} \\ w_i'\mathbf{x}_i^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & -x_i'\mathbf{x}_i^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{bmatrix} = \mathbf{0} \Longrightarrow \mathbf{A}_i\mathbf{h} = \mathbf{0}$$

Equations are linear in $\mathbf{h}$

Holds for any homogeneous representation, $\mathbf{x}_i' = (x_i', y_i', 1)^{\mathrm{T}}$

4.1 The Direct Linear Transformation (DLT) algorithm

$$\mathbf{x}'_i = \mathbf{H}\mathbf{x}_i$$

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$x'_i = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}}$$

$$y'_i = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}}$$

$$x'_i\left(h_{31}x_i + h_{32}y_i + h_{33}\right) = h_{11}x_i + h_{12}y_i + h_{13}$$

$$y'_i\left(h_{31}x_i + h_{32}y_i + h_{33}\right) = h_{21}x_i + h_{22}y_i + h_{23}$$

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -y'_i x_i & -y'_i y_i & -y'_i \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

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

4.1 The Direct Linear Transformation (DLT) algorithm

$$\mathbf{x}_i' = \mathbf{H}\mathbf{x}_i$$

$$\begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$x_i' = \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}}$$

$$y_i' = \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}}$$

$$x_i'\left(h_{31}x_i + h_{32}y_i + h_{33}\right) = h_{11}x_i + h_{12}y_i + h_{13}$$

$$y_i'\left(h_{31}x_i + h_{32}y_i + h_{33}\right) = h_{21}x_i + h_{22}y_i + h_{23}$$

$$\begin{bmatrix} 0 & 0 & 0 & -x_i & -y_i & -1 & y_i'x_i & y_i'y_i & y_i' \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

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

4.1 The Direct Linear Transformation (DLT) algorithm

- Solving for **H**

$$\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \mathbf{A}_3 \\ \mathbf{A}_4 \end{bmatrix} \mathbf{h} = \mathbf{0} \quad \Longrightarrow \quad \mathbf{A}\mathbf{h} = \mathbf{0}$$

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1'x_1 & -x_1'y_1 & -x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y_1'x_1 & -y_1'y_1 & -y_1' \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2'x_2 & -x_2'x_2 & -x_2' \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y_2'x_2 & -y_2'y_2 & -y_2' \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_n'x_n & -x_n'x_n & -x_n' \\ 0 & 0 & 0 & x_n & y_n & 1 & -y_n'x_n & -y_n'y_n & -y_n' \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

- ✓ **A** has dimension 8×9 and rank 8
- ✓ Trivial solution is $\mathbf{h} = \mathbf{0}_9^{\mathrm{T}}$ is not interesting
- ✓ 1-D null space **h** is the nontrivial solutions
- ✓ Choose the one with $\|\mathbf{h}\| = 1$

4.1 The Direct Linear Transformation (DLT) algorithm

4.1.1 Over-determined solution

- For more than 4 points correspondences case:

$$\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_n \end{bmatrix} \mathbf{h} = \mathbf{0} \quad \Longrightarrow \quad \mathbf{Ah} = \boldsymbol{\varepsilon}$$

apply SVD: $\mathbf{A} = \mathbf{UDV^T}$

$\mathbf{h}^* = \mathbf{V_{smallest}}$

The column of **V** corresponding to the smallest singular value

The last column of **V**

Matlab

```
[U, S, V] = svd(A);
h = V(:, end);
```

- ✓ No exact solution due to the “noise”
- ✓ The system $\mathbf{Ah} = \mathbf{0}$ is overdetermined and (in general) has only the trivial solution h = 0
- ✓ Find the approximate solution

Least squares solution: $\mathbf{h}^* = \arg\min_{\mathbf{h}} \|\mathbf{Ah}\|$ subject to $\|\mathbf{h}\| = 1$

$$= \arg\min_{\mathbf{h}} \frac{\|\mathbf{Ah}\|}{\|\mathbf{h}\|} = \text{(normed) eigenvector } \mathbf{A^T A} \text{ with smallest eigenvalue}$$

4.1 The Direct Linear Transformation (DLT) algorithm

4.1.2 Inhomogeneous solution

- By setting $h_j = 1$ $(e.g.\, h_9 = 1)$, and solve for 8-vector $\tilde{\mathbf{h}}$

$$\mathbf{A}_i\,\mathbf{h} = \begin{bmatrix} \mathbf{M}_i & -\mathbf{b} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{h}} \\ 1 \end{bmatrix} = \mathbf{0}$$

$$\Longrightarrow \quad \mathbf{M}\tilde{\mathbf{h}} = \mathbf{b}$$

- ✓ **M** has 8 columns and **h** is an 8-vector.
- ✓ It can be solved by Gaussian elimination (4 points) or linear least squares (more than 4 points)
- ✓ However, if $h_9 = 0$ this approach fails, and also gives poor results if $h_9 \approx 0$
- ✓ Therefore, this approach is not recommended
- ✓ Note $h_9 = \mathbf{H}_{33} = 0$ if origin is mapped to infinity $\quad \mathbf{l}_\infty^{\mathrm{T}}\left(\mathbf{H}\mathbf{x}_0\right) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \mathbf{H} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = 0$

4.1 The Direct Linear Transformation (DLT) algorithm

4.1.2 Inhomogeneous solution

- Example 4.1

$$\mathbf{H}\mathbf{x}_0 = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} h_3 \\ h_6 \\ 0 \end{bmatrix}$$

✓ The origin $(x, y) = (0, 0)$ is mapped to a point at infinity

$$\mathbf{l}_\infty^{\mathrm{T}} \left(\mathbf{H}\mathbf{x}_0 \right) = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \mathbf{H} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = 0$$

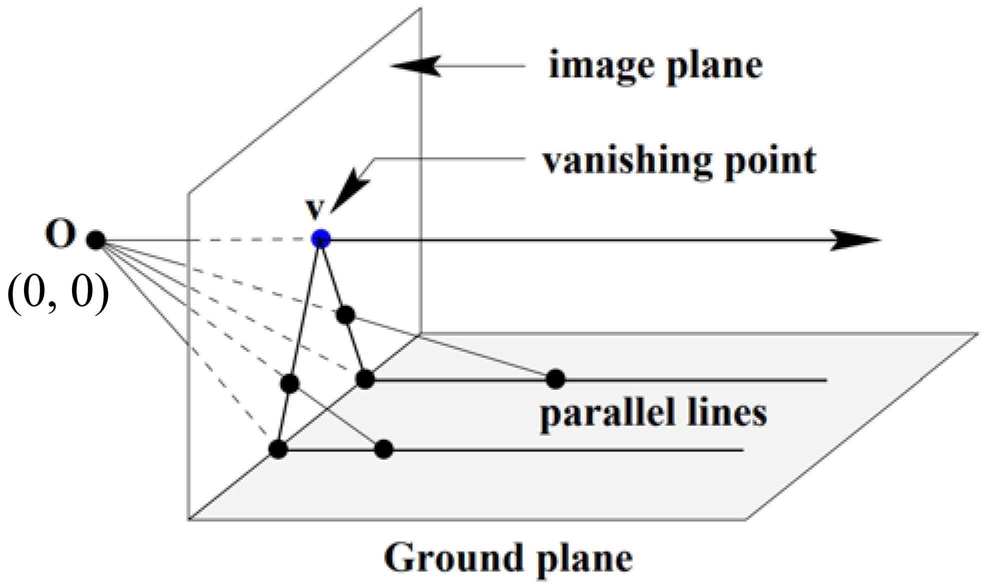
✓ In this case the mapping that takes the image to the world plane maps the origin to the line at infinity, so that the true solution has $\mathbf{H}_{33} = h_9 = 0$

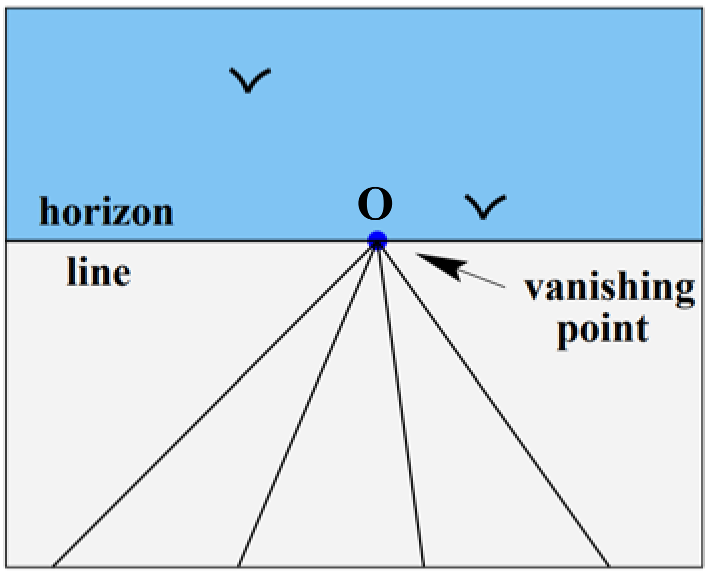
✓ Consequently, an h9 = 1 normalization can be a serious failing in practical situations

4.1 The Direct Linear Transformation (DLT) algorithm

4.1.2 Inhomogeneous solution

- Example 4.1

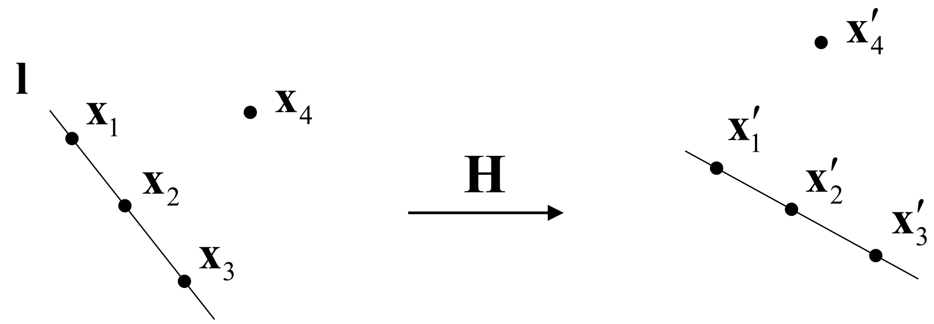




4.1 The Direct Linear Transformation (DLT) algorithm

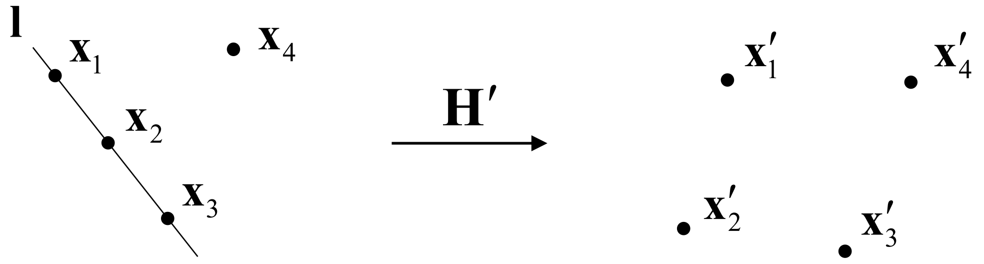
4.1.3 Degenerate configurations

- Case A



✓ The homography is not sufficiently constrained
✓ There will exist a family of homographies mapping $\mathbf{x}_i$ to $\mathbf{x}'_i$

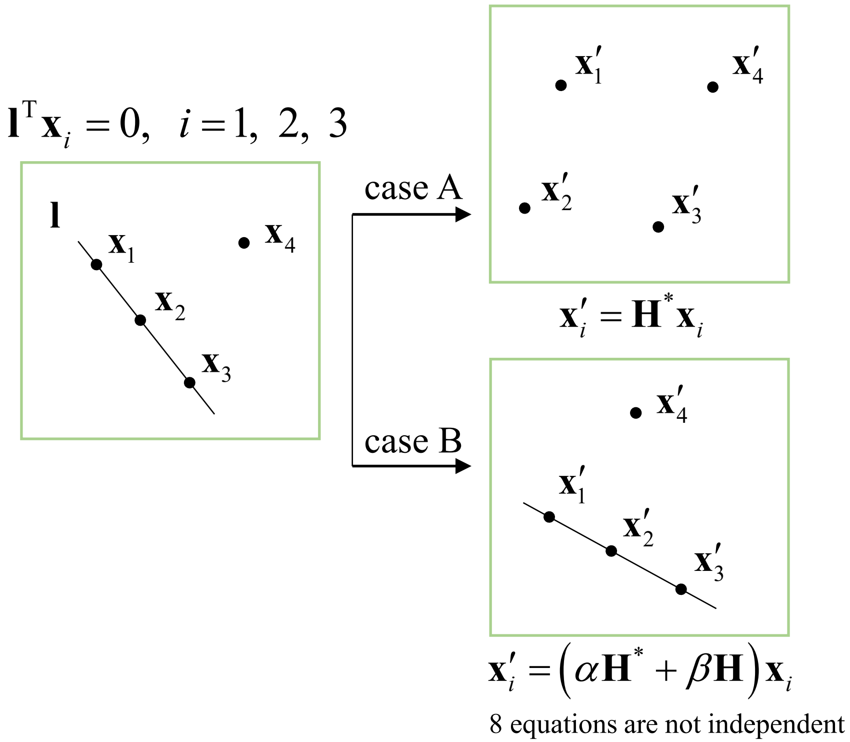
- Case B



✓ Projective transformation must preserve collinearity
✓ There can be no transformation $\mathbf{H}'$ taking $\mathbf{x}_i$ to $\mathbf{x}'_i$

4.1 The Direct Linear Transformation (DLT) algorithm

4.1.3 Degenerate configurations



- Constraints: $\mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = \mathbf{0} \qquad i = 1, 2, 3, 4$
- Define: $\mathbf{H}^* = \mathbf{x}'_4 \mathbf{l}^{\mathrm{T}} \qquad \mathbf{l} = \begin{pmatrix} a & b & c \end{pmatrix}^{\mathrm{T}}$

  Then, $\mathbf{H}^*\mathbf{x}_i = \mathbf{x}'_4\left(\mathbf{l}^{\mathrm{T}}\mathbf{x}_i\right) = \mathbf{0}, \quad i = 1,\ 2,\ 3$

  $\mathbf{H}^*\mathbf{x}_4 = \mathbf{x}'_4\left(\mathbf{l}^{\mathrm{T}}\mathbf{x}_4\right) = k\,\mathbf{x}'_4$

$$\mathbf{H}^* = \begin{bmatrix} ax'_4 & bx'_4 & cx'_4 \\ ay'_4 & by'_4 & cy'_4 \\ aw'_4 & bw'_4 & cw'_4 \end{bmatrix}$$

✓ $\mathbf{H}^*$ is 3×3 singular matrix of rank 1 and thus not a homography (rank 8)

✓ The points $\mathbf{H}^*\mathbf{x}_i = \mathbf{0}$ for $i = 1, 2, 3$ are not well defined.

✓ A situation where a configuration does not determine a unique solution for a particular class of transformation is termed degenerate.

4.1 The Direct Linear Transformation (DLT) algorithm

4.1.4 Solutions from lines and other entities

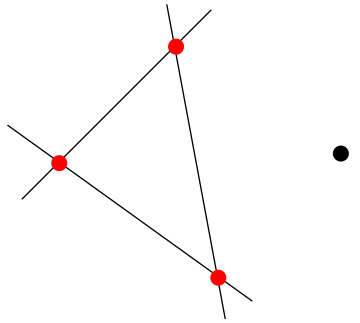
- 2D homographies (8 dof)
  - ✓ Minimum of 4 points or lines $\mathbf{l}'_i = \mathbf{H}^{\mathrm{T}}\mathbf{l}_i \rightarrow \mathbf{Ah} = \mathbf{0}$
- 3D homographies (15 dof)
  - ✓ Minimum of 5 points or 5 plane
- 2D affinities (6 dof)
  - ✓ Minimum of 3 points or lines
- How about mixed configurations?
  - ✓ 2 points and 2 lines (X)
  - ✓ 3 points and 1 line / 1 point and 3 lines (O)

- in 2D
  - ✓ Point/line → 2 constraints
  - ✓ Conic → 5 constraints
- in 3D
  - ✓ Point/plane → 3 constraints

4.1 The Direct Linear Transformation (DLT) algorithm

4.1.4 Solutions from lines and other entities

- 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.
- 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
- The case of two points and two lines is equivalent to five lines with four concurrent, or five points with four collinear $\Longrightarrow$ degenerate configuration



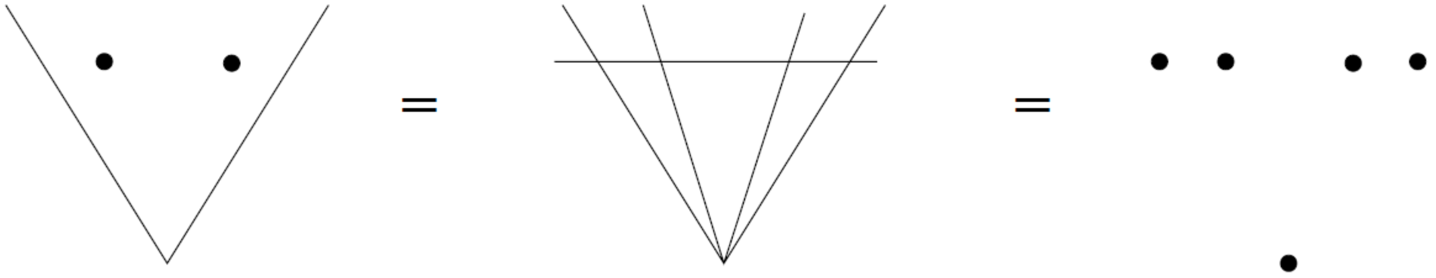


Fig. 4.1. **Geometric equivalence of point–line configurations.** *A configuration of two points and two lines is equivalent to five lines with four concurrent, or five points with four collinear.*

4.2 Different cost functions

- Cost functions which may be minimized in order to determine **H** for over-determined solutions

  - ✓ Algebraic distance
  - ✓ Geometric distance
    - Transfer error, Symmetric transfer error, Reprojection error
  - ✓ Sampson error

4.2 Different cost functions

4.2.1 Algebraic distance

- DLT minimizes $\|\mathbf{Ah}\|$
- Residual (algebraic error) vector: $\mathbf{e}_i = \mathbf{A}_i\mathbf{h}$
  - ✓ Error associated with each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$

$$d_{\text{alg}}(\mathbf{x}'_i, \mathrm{H}\mathbf{x}_i)^2 = \|\epsilon_i\|^2 = \left\| \begin{bmatrix} \mathbf{0}^\top & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i\mathbf{x}_i^\top \end{bmatrix} \mathbf{h} \right\|^2$$

  - ✓ General algebraic distance

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

- The total algebraic distance error:

$$\sum_i d_{\text{alg}}(\mathbf{x}'_i, \mathrm{H}\mathbf{x}_i)^2 = \sum_i \|\epsilon_i\|^2 = \|\mathrm{A}\mathbf{h}\|^2 = \|\epsilon\|^2$$

- linear(unique) solution
- computational cheapness

Has no geometrical and statistical meaning

However, with good normalization it works satisfactory, so it can be used for initialization

4.2 Different cost functions

4.2.2 Geometric distance

$\mathbf{x}$: measured value

$\hat{\mathbf{x}}$: estimated value

$\bar{\mathbf{x}}$: true value

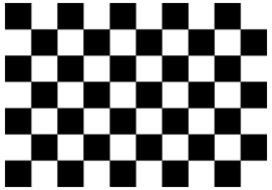
$d(\mathbf{x}, \mathbf{y})$: Euclidean distance

- Transfer error (error in one image)

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

$$\hat{\mathbf{H}} = \arg\min_{\mathbf{H}} \sum_i d\left(\mathbf{x}_i', \mathbf{H}\bar{\mathbf{x}}_i\right)^2$$

e.g. calibration pattern



- Symmetric transfer error (error in both images) $\mathbf{x}_i' \neq \mathbf{H}\,\mathbf{x}_i, \quad \mathbf{x}_i \neq \mathbf{H}^{-1}\,\mathbf{x}_i'$

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

$$\hat{\mathbf{H}} = \arg\min_{\mathbf{H}} \sum_i d\left(\mathbf{x}_i, \mathbf{H}^{-1}\mathbf{x}_i'\right)^2 + d\left(\mathbf{x}_i', \mathbf{H}\mathbf{x}_i\right)^2$$

4.2 Different cost functions

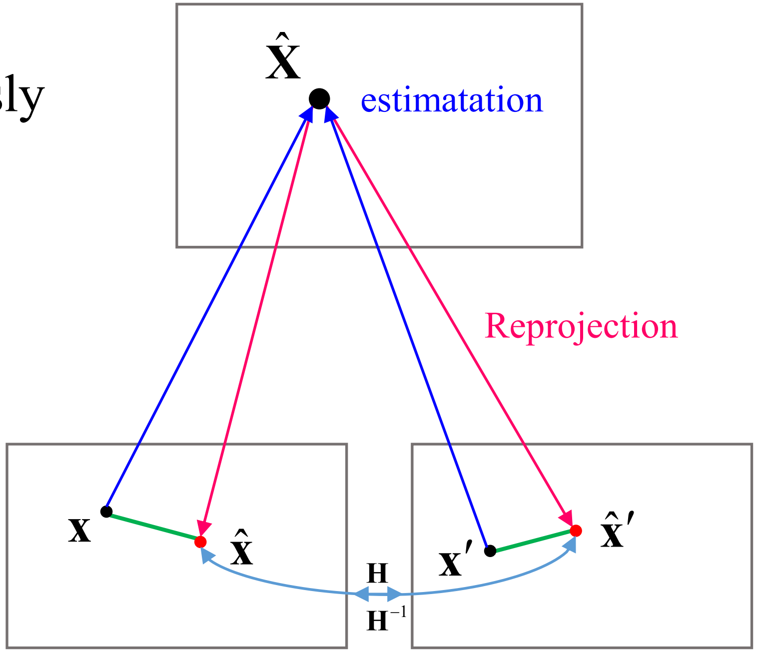
4.2.3 Reprojection error – both images

$\mathbf{x}$: measured value
$\hat{\mathbf{x}}$: estimated value
$\bar{\mathbf{x}}$: true value
$d(\mathbf{x},\mathbf{y})$: Euclidean distance

- Reprojection error
  - ✓ Find $\hat{\mathbf{H}}$ and pairs of perfectly matched points $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ simultaneously

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

$$\left(\hat{\mathbf{H}}, \hat{\mathbf{x}}_i, \hat{\mathbf{x}}'_i\right) = \underset{\mathbf{H}\ \hat{\mathbf{x}}_i,\ \hat{\mathbf{x}}'_i}{\arg\min} \sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 \quad \text{subject to} \quad \hat{\mathbf{x}}'_i = \hat{\mathbf{H}}\hat{\mathbf{x}}_i \quad \forall i$$



  - ✓ We wish to estimate a point on the world plane $\hat{\mathbf{X}}_i$ from $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ which is then *reprojected* to the estimated perfectly matched correspondence $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$

4.2 Different cost functions

4.2.3 Reprojection error – both images

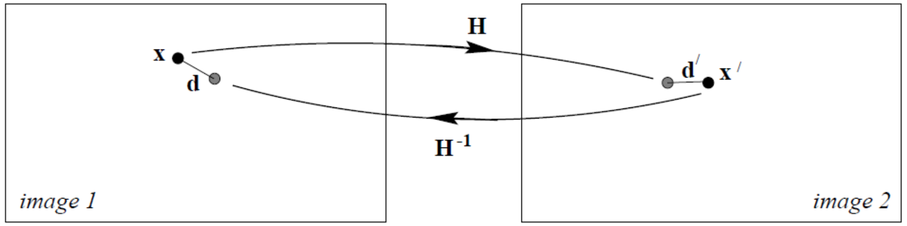
$\mathbf{x}$: measured value

$\hat{\mathbf{x}}$: estimated value

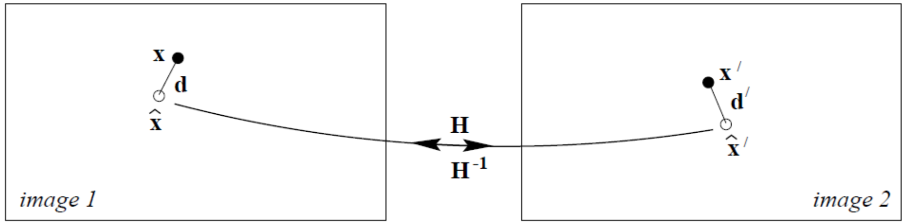
$\bar{\mathbf{x}}$: true value

$d(\mathbf{x}, \mathbf{y})$: Euclidean distance

- Comparison of symmetric transfer error and reprojection error



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



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

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 $d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2$.*

4.2 Different cost functions

4.2.4 Comparison of geometric and algebraic distance

- The algebraic error vector: $d_{alg}\left(\mathbf{x}'_i, \hat{\mathbf{x}}'_i\right)^2 = \left(y'_i\hat{w}'_i - w'_i\hat{y}'_i\right)^2 + \left(w'_i\hat{x}' - x'\hat{w}'_i\right)^2$

- The geometric distance:

$$\begin{aligned} d\left(\mathbf{x}'_i, \hat{\mathbf{x}}'_i\right)^2 &= \sqrt{\left(y'_i/w'_i - \hat{y}'_i/\hat{w}'_i\right)^2 + \left(\hat{x}'/\hat{w}'_i - x'/w'_i\right)^2} \\ &= d_{alg}\left(\mathbf{x}'_i, \hat{\mathbf{x}}'_i\right)/w'_i\hat{w}'_i \end{aligned}$$

- If $w_i = 1$, then $w'_i = \hat{w}'_i = 1$,
  - ✓ Geometric distance is equal to algebraic distance
  - ✓ Example) 2D affine transform case

$$\mathbf{H}_{\mathrm{A}} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

4.2 Different cost functions

4.2.5 Geometric interpretation of reprojection error

- Estimating $\mathbf{H}$ is to fit surface $\mathbf{V_H}$ to the measurement points, $\mathbf{X}_i = (x_i, y_i, x'_i, y'_i)^{\mathrm{T}}$ in $\mathbb{R}^4$

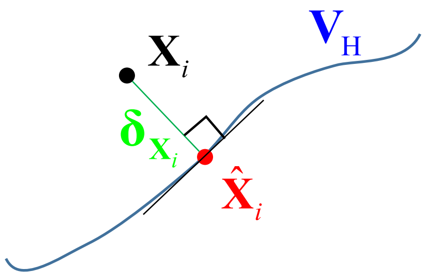
- Let $\hat{\mathbf{X}}_i = (\hat{x}_i, \hat{y}_i, \hat{x}'_i, \hat{y}'_i)^{\mathrm{T}}$ be the closest to $\mathbf{X}_i$, then

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

geometric distance in $\mathbb{R}^4$ is equivalent to reprojection error

✓ = perpendicular distance

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



4.2 Different cost functions

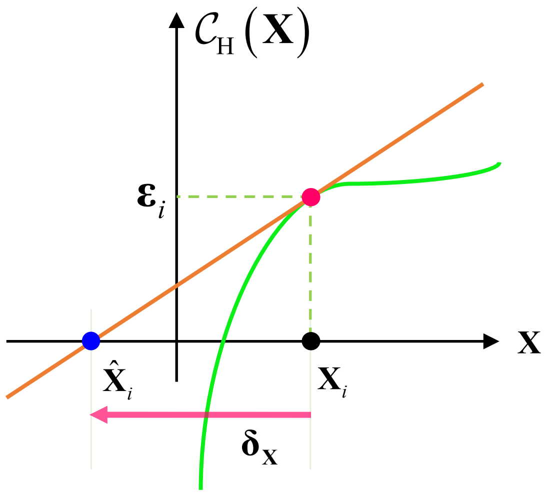
4.2.6 Sampson error

- The geometric error problem

  - ✓ Quite complex
  - ✓ Required the simultaneous estimation of both the homography matrix **H** and the point $\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_i'$
  - ✓ Non-linear estmation

4.2 Different cost functions

4.2.6 Sampson error



- Close approximation to geometric error $\mathbf{X} = (x, y, x', y')^{\mathrm{T}}$
- 1st order approximation of the point $\widehat{\mathbf{X}} = \mathbf{X} + \boldsymbol{\delta}_{\mathbf{X}}$
- $\widehat{\mathbf{X}} = \mathbf{X} + \boldsymbol{\delta}_{\mathbf{X}}$ lying on $\mathbf{V}_{\mathrm{H}}$ $\quad \mathbf{Ah} = \mathbf{0} \quad \Leftrightarrow \quad \mathcal{C}_{\mathrm{H}}(\hat{\mathbf{X}}) = \mathbf{0}$
- Linearize it by a Taylor series expansion

$$\mathcal{C}_{\mathrm{H}}(\hat{\mathbf{X}}) = \mathcal{C}_{\mathrm{H}}(\mathbf{X} + \boldsymbol{\delta}_{\mathbf{X}})$$

$$\mathbf{Ah} = \boldsymbol{\varepsilon} \quad \Leftrightarrow \quad \mathcal{C}_{\mathrm{H}}(\mathbf{X}) = \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix}$$

$$= \mathcal{C}_{\mathrm{H}}(\mathbf{X}) + \frac{\partial \mathcal{C}_{\mathrm{H}}(\mathbf{X})}{\partial \mathbf{X}} \boldsymbol{\delta}_{\mathbf{X}}$$

$$= \boldsymbol{\varepsilon} + J \boldsymbol{\delta}_{\mathbf{X}} = \mathbf{0}$$

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

4.2 Different cost functions

4.2.6 Sampson error

- Find the vector $\boldsymbol{\delta}_{\mathbf{X}}$ that minimizes $\|\boldsymbol{\delta}_{\mathbf{X}}\|$ subject to $\mathrm{J}\boldsymbol{\delta}_{\mathbf{X}} = -\boldsymbol{\epsilon}$
- Use of Lagrange method:

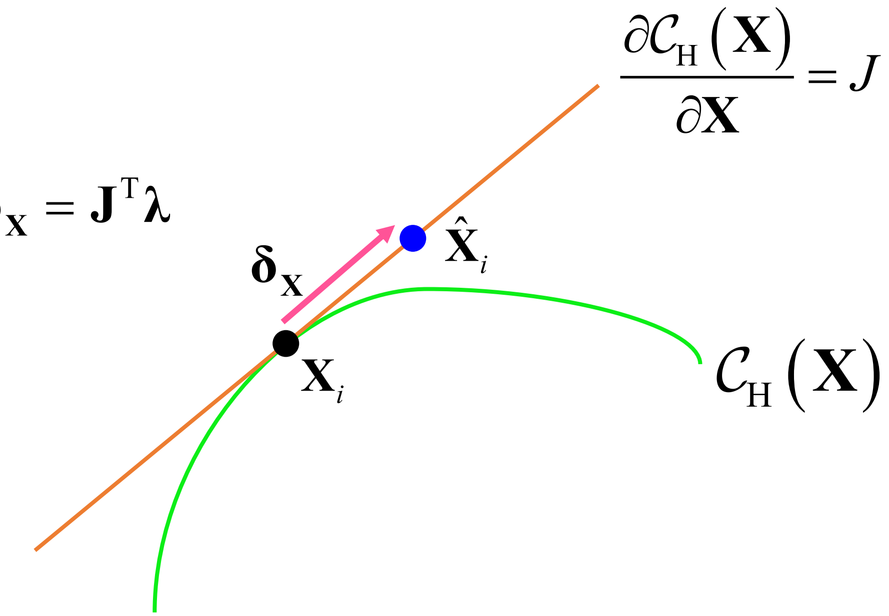
minimize $E = \boldsymbol{\delta}_{\mathbf{X}}^{\mathrm{T}} \boldsymbol{\delta}_{\mathbf{X}} - 2\boldsymbol{\lambda}^{\mathrm{T}} \left(\mathbf{J}\boldsymbol{\delta}_{\mathbf{X}} + \boldsymbol{\varepsilon} \right)$

$$\frac{\partial E}{\partial \boldsymbol{\delta}_{\mathbf{X}}} = \mathbf{0}^{\mathrm{T}} \quad \Rightarrow \quad 2\boldsymbol{\delta}_{\mathbf{X}}^{\mathrm{T}} - 2\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{J} = \mathbf{0}^{\mathrm{T}} \quad \Rightarrow \quad \boldsymbol{\delta}_{\mathbf{X}} = \mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda}$$

$$\frac{\partial E}{\partial \boldsymbol{\lambda}} = \mathbf{0} \quad \Rightarrow \quad \mathbf{J}\boldsymbol{\delta}_{\mathbf{X}} + \boldsymbol{\varepsilon} = \mathbf{0}$$

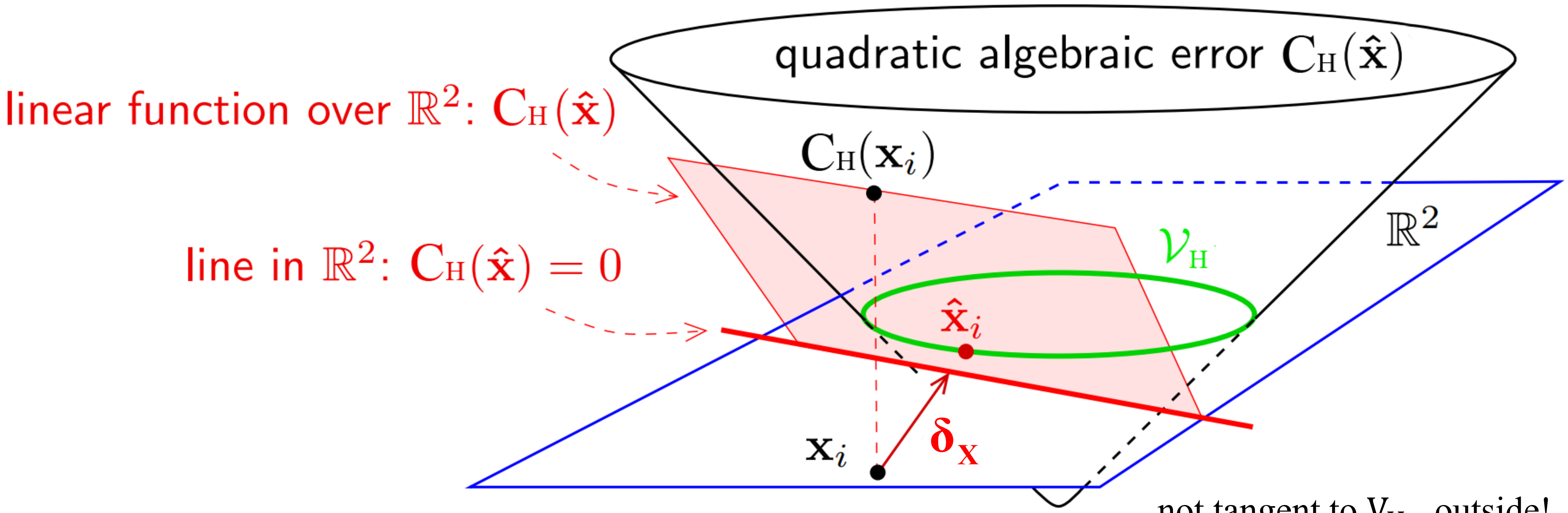
$$\mathbf{J}\mathbf{J}^{\mathrm{T}}\boldsymbol{\lambda} = -\boldsymbol{\varepsilon} \quad \Rightarrow \quad \boldsymbol{\lambda} = -\left(\mathbf{J}\mathbf{J}^{\mathrm{T}} \right)^{-1} \boldsymbol{\varepsilon}$$

$$\boldsymbol{\delta}_{\mathbf{X}} = -\mathbf{J}^{\mathrm{T}} \left(\mathbf{J}\mathbf{J}^{\mathrm{T}} \right)^{-1} \boldsymbol{\varepsilon}$$



4.2 Different cost functions

4.2.6 Sampson error



not tangent to V_H , outside!

4.2 Different cost functions

4.2.6 Sampson error

$$\hat{\mathbf{X}} = \mathbf{X} + \boldsymbol{\delta}_{\mathbf{X}}$$

$$\boldsymbol{\delta}_{\mathbf{X}} = -\mathbf{J}^{\mathsf{T}}\left(\mathbf{J}\mathbf{J}^{\mathsf{T}}\right)^{-1}\boldsymbol{\varepsilon}$$

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

- Overall error $\mathcal{D}_{\perp} = \sum_i \boldsymbol{\epsilon}_i^{\mathsf{T}}(\mathtt{J}_i\mathtt{J}_i^{\mathsf{T}})^{-1}\boldsymbol{\epsilon}_i$
  - ✓ where $\boldsymbol{\epsilon}$ and J both depend on **H**

4.2 Different cost functions

4.2.6 Sampson error

Linear cost function

- The algebraic error vector $\mathtt{C}_{\mathrm{H}}(\mathbf{X}) = \mathtt{A}(\mathbf{X})\mathbf{h}$ is typically multilinear in the entries of $\mathbf{X}$
- The case where $\mathtt{A}(\mathbf{X})\mathbf{h}$ is linear, the first-order approximation to geometric error given by the Taylor expansion is exact
- The Sampson error is identical to geometric error
- The variety $\mathbf{V}_{\mathrm{H}}$ defined by the equation $\mathtt{C}_{\mathrm{H}}(\mathbf{X}) = \mathbf{0}$, a set of linear equations, is a hyperplane depending on $\mathbf{H}$.
- The problem of finding $\mathbf{H}$ now becomes a hyperplane fitting problem

4.3 Statistical cost functions and Maximum Likelihood estimation

- Assume that image measurement errors obey a zero-mean isotropic Gaussian distribution
- The probability density function (PDF) $\Pr(\mathbf{x}) = \left(\frac{1}{2\pi\sigma^2}\right) e^{-d(\mathbf{x},\bar{\mathbf{x}})^2/(2\sigma^2)}$
- **Error in one image** $\Pr(\{\mathbf{x}'_i\}|\mathtt{H}) = \prod_i \left(\frac{1}{2\pi\sigma^2}\right) e^{-d(\mathbf{x}'_i,\mathtt{H}\bar{\mathbf{x}}_i)^2/(2\sigma^2)}$
  - Log-likelihood $\log \Pr(\{\mathbf{x}'_i\}|\mathtt{H}) = -\frac{1}{2\sigma^2}\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2 + \text{ constant}$
  - Maximum Likelihood Estimate (MLE) = maximize log-likelihood = minimize $\sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2$
- **Error in both images** $\Pr(\{\mathbf{x}_i, \mathbf{x}'_i\}|\mathtt{H}, \{\bar{\mathbf{x}}_i\}) = \prod_i \left(\frac{1}{2\pi\sigma^2}\right)^2 e^{-\left(d(\mathbf{x}_i,\bar{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i,\mathtt{H}\bar{\mathbf{x}}_i)^2\right)/(2\sigma^2)}$
  - Maximum Likelihood Estimate (MLE) = maximize log-likelihood = minimize $\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$ with $\hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i$

4.3 Statistical cost functions and Maximum Likelihood estimation

- **Mahalanobis distance**

  - ✓ Consider non-isotropic Gaussian model
  - ✓ Density normalized distance
  - ✓ Maximizing the log-likelihood is then equivalent to minimizing the Mahalanobis distance

cost function

- **Error in one image** $\|\mathbf{X} - \bar{\mathbf{X}}\|_\Sigma^2 = (\mathbf{X} - \bar{\mathbf{X}})^\mathsf{T}\Sigma^{-1}(\mathbf{X} - \bar{\mathbf{X}})$

- **Error in both images** $\|\mathbf{X} - \bar{\mathbf{X}}\|_\Sigma^2 + \|\mathbf{X}' - \bar{\mathbf{X}}'\|_{\Sigma'}^2$

- **Varying covariance** $\sum \|\mathbf{x}_i - \bar{\mathbf{x}}_i\|_{\Sigma_i}^2 + \sum \|\mathbf{x}'_i - \bar{\mathbf{x}}'_i\|_{\Sigma'_i}^2$

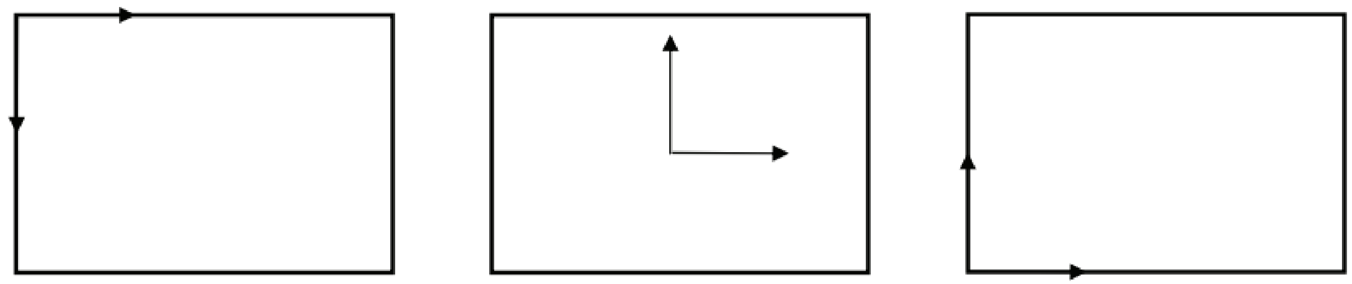
4.4 Transformation invariance and normalization

4.4.1 Invariance to image coordinate transformations

- Let $\mathbf{x}_i \underset{\mathbf{H}}{\leftrightarrow} \mathbf{x}'_i$ and $\tilde{\mathbf{x}}_i \underset{\tilde{\mathbf{H}}}{\leftrightarrow} \tilde{\mathbf{x}}'_i$, where $\tilde{\mathbf{x}}_i = \mathbf{T}\mathbf{x}_i$, $\tilde{\mathbf{x}}'_i = \mathbf{T}'\mathbf{x}'_i$ and $\tilde{\mathbf{H}} = \mathbf{T}'\mathbf{H}\mathbf{T}^{-1}$

$$\tilde{\mathbf{x}}'_i = \mathbf{T}'\mathbf{x}'_i = \mathbf{T}'\mathbf{H}\mathbf{x}_i = \mathbf{T}'\mathbf{H}\mathbf{T}^{-1}\mathbf{x}'_i = \tilde{\mathbf{H}}\tilde{\mathbf{x}}_i$$

  - e.g. Coordinate transformation



$$\tilde{\mathbf{H}} = \mathbf{T}'\mathbf{H}\mathbf{T}^{-1} \quad \Longrightarrow \quad \mathbf{H} = \mathbf{T}'^{-1}\tilde{\mathbf{H}}\mathbf{T}$$

4.4 Transformation invariance and normalization

4.4.2 Non-invariance of the DLT algorithm

- Let $\mathbf{x}_i \underset{\mathbf{H}}{\leftrightarrow} \mathbf{x}'_i$ and $\tilde{\mathbf{x}}_i \underset{\tilde{\mathbf{H}}}{\leftrightarrow} \tilde{\mathbf{x}}'_i$, where $\tilde{\mathbf{x}}_i = \mathbf{T}\mathbf{x}_i$, $\tilde{\mathbf{x}}'_i = \mathbf{T}'\mathbf{x}'_i$ and $\tilde{\mathbf{H}} = \mathbf{T}'\mathbf{H}\mathbf{T}^{-1}$
- Does the DLT algorithm applied to the correspondence set $\tilde{\mathbf{x}}_i \underset{\tilde{\mathbf{H}}}{\leftrightarrow} \tilde{\mathbf{x}}'_i$ yield the transformation $\tilde{\mathbf{H}} = \mathbf{T}'\mathbf{H}\mathbf{T}^{-1}$?

  - special case

$$\mathbf{T}' = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$$

similarity transform

$$\mathbf{T}'^{*} = s\begin{bmatrix} \mathbf{R} & \mathbf{0} \\ -\mathbf{t}^{\mathrm{T}}\mathbf{R} & s \end{bmatrix}$$

projective transform

$$\tilde{\mathbf{H}} = \mathbf{T}'\mathbf{H}\mathbf{T}^{-1}$$

- Effect of change of coordinates on algebraic error

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

where T'^{*} represents the cofactor matrix of T'

$$\left\|\tilde{\mathbf{A}}_i\tilde{\mathbf{h}}\right\| = \left\|(\tilde{e}_{i1},\tilde{e}_{i2})^{\mathsf{T}}\right\| = \left\|s\mathrm{R}(e_{i1},e_{i2})^{\mathsf{T}}\right\| = s\left\|(e_{i1},e_{i2})^{\mathsf{T}}\right\| = s\left\|\mathbf{A}_i\mathbf{h}\right\|$$

$$\Longrightarrow \quad d_{\mathrm{alg}}\left(\tilde{\mathbf{x}}'_i, \tilde{\mathbf{H}}\tilde{\mathbf{x}}_i\right) = s\,d_{\mathrm{alg}}\left(\mathbf{x}'_i, \mathbf{H}\mathbf{x}_i\right)$$

4.4 Transformation invariance and normalization

4.4.2 Non-invariance of the DLT algorithm

$$\text{minimize} \sum_i d_{\text{alg}}(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2 \text{ subject to } \|\mathtt{H}\| = 1$$

$$\Leftrightarrow \quad \text{minimize} \sum_i d_{\text{alg}}(\tilde{\mathbf{x}}'_i, \widetilde{\mathtt{H}}\tilde{\mathbf{x}}_i)^2 \text{ subject to } \|\mathtt{H}\| = 1$$

$$\not\Leftrightarrow \quad \text{minimize} \sum d_{\text{alg}}(\tilde{\mathbf{x}}'_i, \widetilde{\mathtt{H}}\tilde{\mathbf{x}}_i)^2 \text{ subject to } \|\widetilde{\mathtt{H}}\| = 1$$

4.4 Transformation invariance and normalization

4.4.3 Invariance of geometric error

- Given $\mathbf{x}_i \underset{\mathbf{H}}{\leftrightarrow} \mathbf{x}_i'$ and $\tilde{\mathbf{x}}_i \underset{\tilde{\mathbf{H}}}{\leftrightarrow} \tilde{\mathbf{x}}_i'$, where $\tilde{\mathbf{x}}_i = \mathbf{T}\mathbf{x}_i$, $\tilde{\mathbf{x}}_i' = \mathbf{T}'\mathbf{x}_i'$ and $\tilde{\mathbf{H}} = \mathbf{T}'\mathbf{H}\mathbf{T}^{-1}$

- If **T** and **T'** are Euclidean Transformations

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

- If **T** and **T'** are Similarity Transformations

$$d_{\mathrm{alg}}(\tilde{\mathbf{x}}_i', \tilde{\mathrm{H}}\tilde{\mathbf{x}}_i) = s d_{\mathrm{alg}}(\mathbf{x}_i', \mathrm{H}\mathbf{x}_i)$$

- Thus, geometric error is invariant to Euclidean and similarity transformations

4.4 Transformation invariance and normalization

4.4.4 Normalizing transformations

- How to make DLT invariant? Choice of coordinates?

- Data normalization

  - Translate so that the centroid to be the origin
  - Scale so that the average distance to the origin to be $\sqrt{2}$
  - Perform independently on both images

  ✓ Improves accuracy
  ✓ Invariant to scale and coordinate system

4.5 Iterative minimization methods

- minimizing the various geometric cost functions
- iterative techniques
  - slower
  - initial estimate
  - local minimum
  - stopping criterion

  - Consequently, iterative techniques generally require more careful implementation

- Step
  - Cost function: basis for minimization
  - Parametrization: finite number of parameters
  - Function specification: cost function expressed by parameters
  - Initialization: initial parameter estimate using DLT
  - Iteration: the parameters are iteratively refined with the goal of minimizing the cost function

4.5 Iterative minimization methods

Parametrization

- Parameters should cover complete space and allow the estimation to be efficient

- Minimal or over-parameterization e.g. 8 or 9
  - ✓ minimal parameterization gives often more complicate functions – local minima
  - ✓ good algorithms can deal with over-parameterization
  - ✓ So, it is not necessary to use minimal parameterization

4.5 Iterative minimization methods

Function specification

(i) Measurement vector $\mathbf{X} \in \mathbf{R}^N$ with covariance Σ

(ii) Set of parameters represented by vector $\mathbf{P} \in \mathbf{R}^\mathrm{M}$

(iii) Mapping f : $\mathbf{R}^\mathrm{M} \rightarrow \mathbf{R}^\mathrm{N}$. Range of mapping is surface S representing allowable measurements

(iv) Cost function: squared Mahalanobis distance

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

- The goal is to achieve $\mathbf{P}$ such that $f(\mathbf{P}) = \mathbf{X}$, or get as close as possible in terms of Mahalanobis distance

4.5 Iterative minimization methods

Let $\mathbf{X} = (x'_1, y'_1, x'_2, y'_2, \ldots, x'_n, y'_n)^T$

- **Error in one image**

$$f : \mathbf{h} \mapsto (\mathtt{H}\mathbf{x}_1, \mathtt{H}\mathbf{x}_2, \ldots, \mathtt{H}\mathbf{x}_n)$$
$$\|\mathbf{X} - f(\mathbf{h})\|^2 \text{ is equal to } \sum_i d(\mathbf{x}'_i, \mathtt{H}\bar{\mathbf{x}}_i)^2$$

- **Symmetric transfer error**

$$f : \mathbf{h} \mapsto (\mathtt{H}^{-1}\mathbf{x}'_1, \ldots, \mathtt{H}^{-1}\mathbf{x}'_n, \mathtt{H}\mathbf{x}_1, \ldots, \mathtt{H}\mathbf{x}_n)$$
$$\|\mathbf{X} - f(\mathbf{h})\|^2 \text{ is equal to } \sum_i d(\mathbf{x}_i, \mathtt{H}^{-1}\mathbf{x}'_i)^2 + d(\mathbf{x}'_i, \mathtt{H}\mathbf{x}_i)^2$$

- **Reprojection error**

$$f : (\mathbf{h}, \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n) \mapsto (\hat{\mathbf{x}}_1, \hat{\mathbf{x}}'_1, \ldots, \hat{\mathbf{x}}_n, \hat{\mathbf{x}}'_n) \qquad \mathbf{P} = (\mathbf{h}, \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n)$$
$$\|\mathbf{X} - f(\mathbf{P})\|^2 \text{ is equal to } \sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2 \text{ subject to } \hat{\mathbf{x}}'_i = \hat{\mathtt{H}}\hat{\mathbf{x}}_i \ \forall i$$

4.5 Iterative minimization methods

Initialization

- Good initialization is very important
- Use linear solution (Normalized DLT) or random sampling (Robust algorithm), etc.
- Dense sampling of parameter space
- Fixed point in parameter space → often local minima

4.5 Iterative minimization methods

Iteration methods

- For minimizing the chosen cost function
- Newton iteration
- Levenberg-Marquardt method
- Powell’ method
- Simplex method, etc.

4.6 Experimental comparison of the algorithms

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.*

4.7 Robust estimation

4.7.1 RANSAC

- How to determine *inliers* (discard *outliers* or mismatched correspondences) to estimate the homography robustly?

- RANSAC (RAndom SAmple Consensus)

- Consider a 2D line-fitting example

$x' = ax + b$ Estimation of 1D affine transform

- ✓ Select 2 points randomly – determine a line
- ✓ Check the support points that lie within a distance threshold
- ✓ The line with most support is the robust fit

4.7 Robust estimation

4.7.1 RANSAC

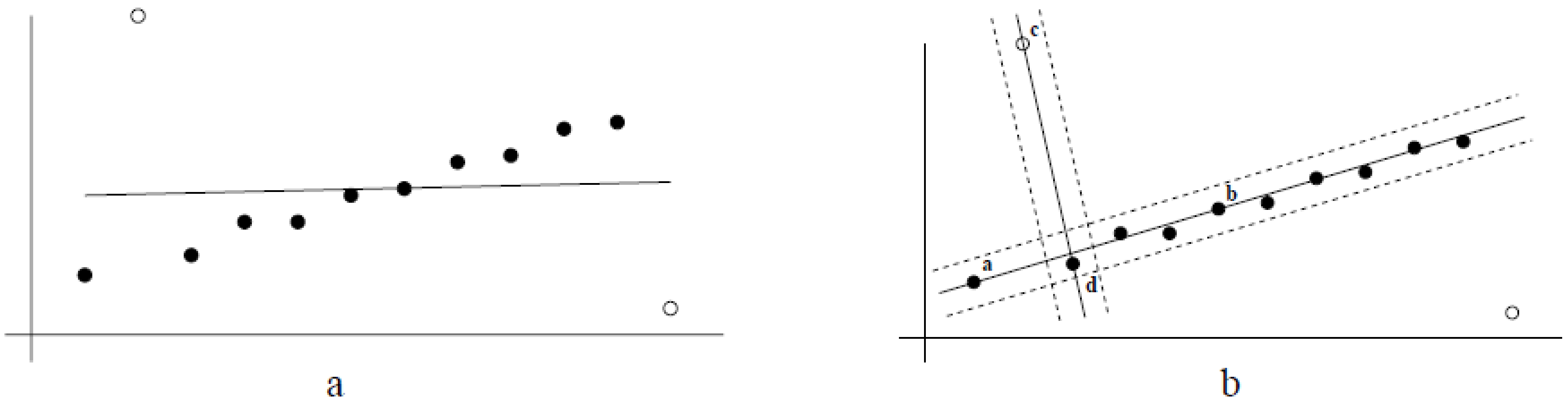


Fig. 4.7. **Robust line estimation.** *The solid points are inliers, the open points outliers. (a) A least-squares (orthogonal regression) fit to the point data is severely affected by the outliers. (b) In the RANSAC algorithm the support for lines through randomly selected point pairs is measured by 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.*

4.7 Robust estimation

4.7.1 RANSAC

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.

4.7 Robust estimation

4.7.1 RANSAC

chi-square distribution

$$\sum_{i=1}^{k}\left(\frac{X_i-\mu_i}{\sigma_i}\right)^2$$

- **What is the distance threshold?**
- Choose t so probability for inlier is α (e.g. 0.95)
- If measurement error is Gaussian with zero mean and standard deviation σ
  - ✓ The squared of the point distance $d_\perp^2$ is sum of squared Gaussain variable
  - ✓ Follows a χ_m^2 distribution with m degrees of freedom (m = codimension)

Cumulative Distribution Probability

$$\Pr(\chi_m^2 < k^2) = F_m(k^2) = \int_0^{k^2} \chi_m^2(\zeta)d\zeta$$

$$\Pr(d_\perp^2 < t^2) = \alpha \Rightarrow \Pr(\chi_m^2 < t^2) = F_m(t^2) = \alpha$$

$$\Rightarrow t^2 = F_m^{-1}(\alpha)$$

For Gaussian with $N(0,\ \sigma)$,

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

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

$t^2 = F_m^{-1}(\alpha)\sigma^2$ *for a probability of* $\alpha = 0.95$

4.7 Robust estimation

4.7.1 RANSAC

- **How may samples?**
- Let p be the probability that at least one of the random samples of s points among N selection is free from outliers, and ω be the probability that any selected point is an inlier

$$p = 1 - \Pr(\text{all N selections include outliers})$$
$$= 1 - (1 - \Pr(\text{all s points are inliers}))^N$$
$$= 1 - (1 - \omega^s)^N, \quad \omega = 1 - \varepsilon$$

$$N = \log(1-p)/\log(1-(1-\epsilon)^s)$$

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.*

- Ex 4.4) line fitting case $n = 12, \varepsilon = \frac{2}{12} = \frac{1}{6}, s = 2 \;\Rightarrow\; N = 5$

4.7 Robust estimation

4.7.1 RANSAC

- **How large is an acceptable consensus set?**
- Terminate when inlier ratio reaches expected ratio of inliers
  - ✓ For n data points, $T = (1 - \varepsilon)n$
  - ✓ For the line fitting (fig 4.7) $\varepsilon = 0.2 \rightarrow T = (1 - 0.2)12 = 9.6 \Rightarrow 10$

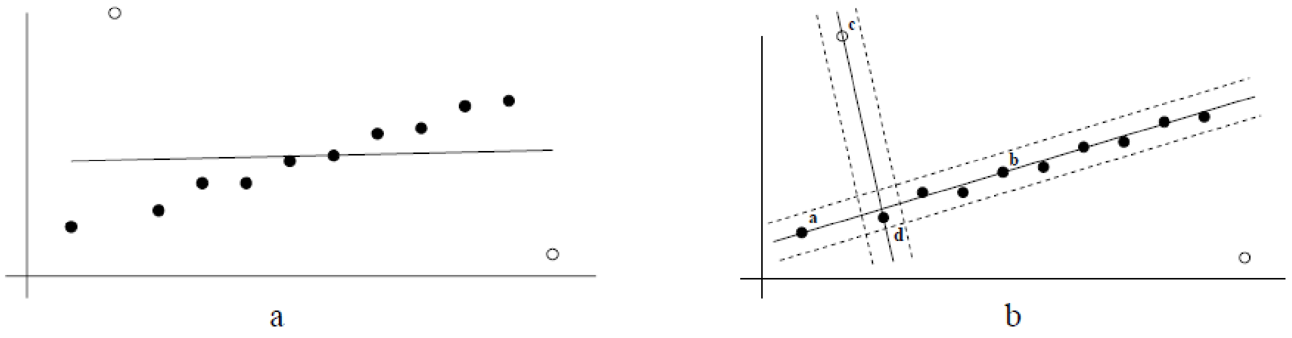


Fig. 4.7. **Robust line estimation.** *The solid points are inliers, the open points outliers. (a) A least-squares (orthogonal regression) fit to the point data is severely affected by the outliers. (b) In the RANSAC algorithm the support for lines through randomly selected point pairs is measured by 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.*

4.7 Robust estimation

4.7.1 RANSAC

- **Determining the number of samples adaptively**
- ε is often unknown, so pick the worst case, $\varepsilon = 0.5$ and a consensus set with 80% of the data is found as inliers, then the updated estimate is $\varepsilon = 0.2$

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

Adaptive algorithm for determining the number of RANSAC samples.

4.7 Robust estimation

4.7.2 Robust Maximum Likelihood estimation

- The final step of the RANSAC algorithm is to re-estimate the model using all the inliers
  - ✓ minimizing a ML cost function

    minimizing $\mathcal{C} = \sum_i d_{\perp i}^2$

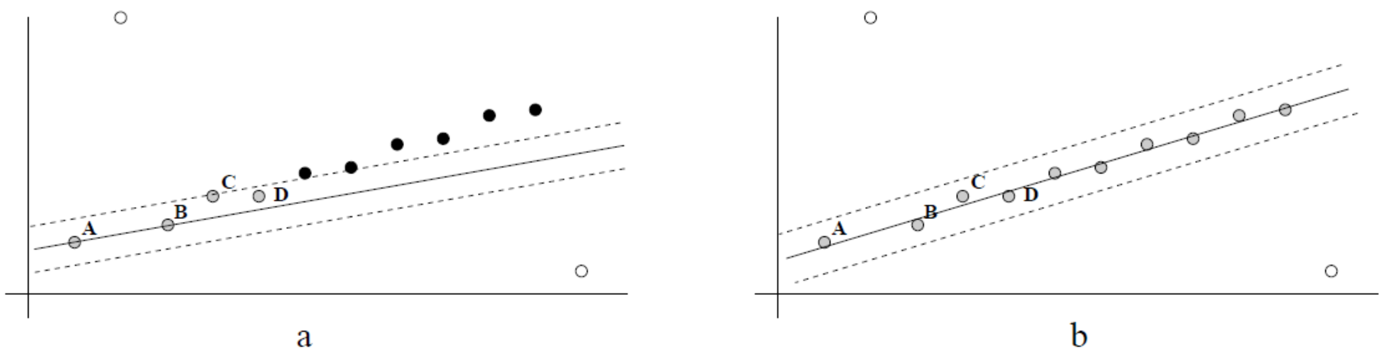


Fig. 4.8. **Robust ML estimation.** *The grey points are classified as inliers to the line. (a) A line defined by points $\langle \mathbf{A}, \mathbf{B} \rangle$ has a support of four (from points $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$). (b) The ML line fit (orthogonal least-squares) to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

- Iterative estimation
  - ✓ Optimal fit to inliers
  - ✓ Re-classify inliers
  - ✓ Iterate until no change

- Robust cost function

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

Here $d_{\perp i}$ are point errors and $\gamma(e)$ is a robust cost function

4.7 Robust estimation

4.7.3 Other robust algorithm

- RANSAC
  - ✓ Maximizes number of inliers

- LMS (Least Median Squares) estimation
  - ✓ Minimizes the median error
  - ✓ Score the model by the median of the distances to all data
  - ✓ No threshold or prior knowledge of error variance
  - ✓ However, fails if more than half of the data is outlying
  - ✓ Use the proportion outliers to determine the distance

- Not recommended
  - ✓ Case deletion
  - ✓ Iterative least-squares

4.8 Automatic computation of a homography

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.

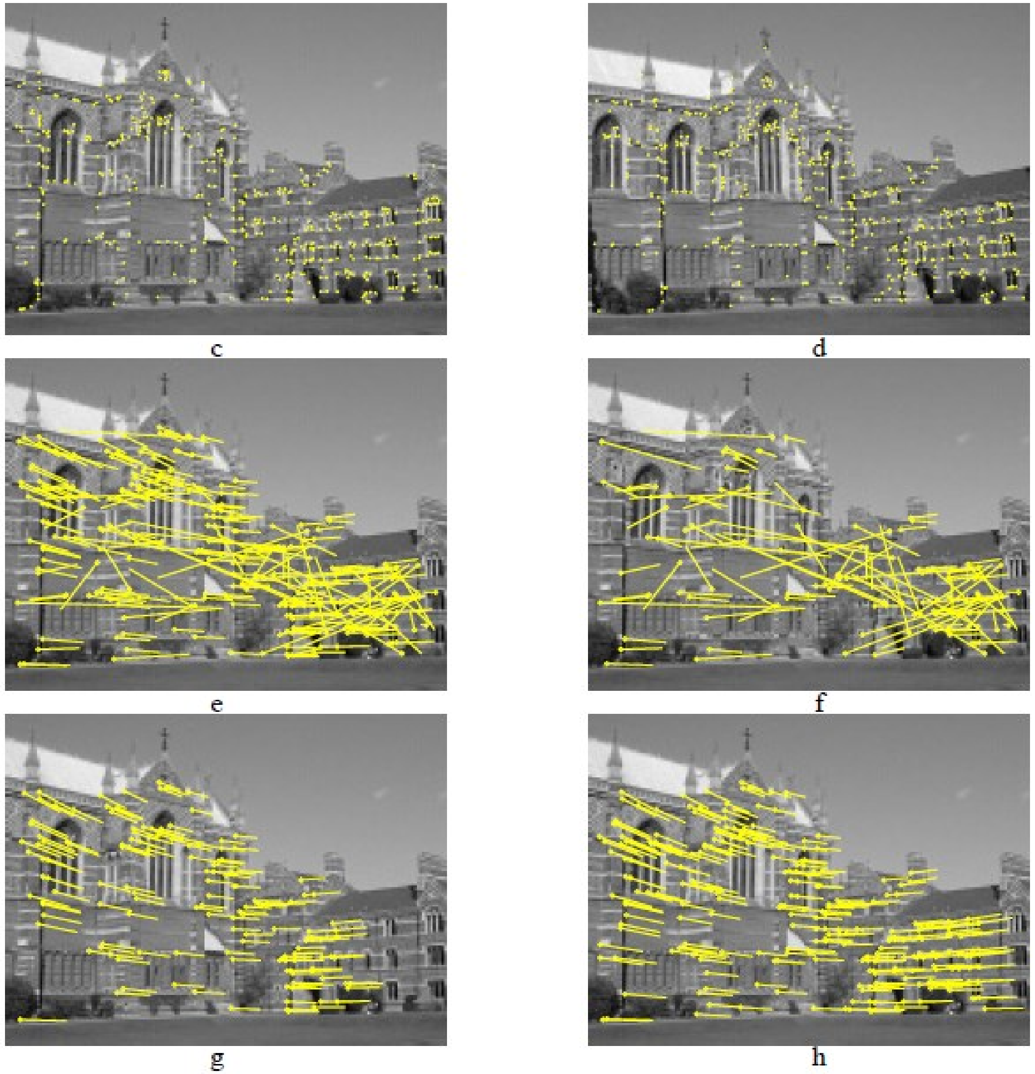
The last two steps can be iterated until the number of correspondences is stable.

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

4.8 Automatic computation of a homography

- Determining putative correspondences
  - ✓ Select corner points in each images
  - ✓ Match points using similarity measure (NCC, SSD, SAD, etc) symmetrically within a search region
- RANSAC for homography
  - ▪ Distance measure
    - ✓ Symmetric transfer error
    - ✓ Reprojection error
    - ✓ Sampson error
  - ▪ Sample selection
    - ✓ Avoid degenerate samples (collinear ones)
    - ✓ Samples with good spatial distribution
- Robust ML estimation (cycle approach)
  - ✓ Carry out ML estimation on inliers using Levenberg-Marquardt algorithm
  - ✓ Recompute the inliers using new **H**
  - ✓ Iterate cycle until converges

4.8 Automatic computation of a homography



(c, d) Interest points (500 / images)

(e) Putative correspondences (268)

(f) Outliers (117)

(g) Inliers (151)

(h) final set of correspondences (262)