Image Procesing and Computer Vision Notes

by Mattia Orlandi

2b. Image Formation and Acquisition - Camera Calibration

2b.1. Perspective Projection Matrix

Projective Space

- The physical space is a 3D Euclidean space ($\mathbb{R}^3$) whose points can be represented as 3D vectors in a given reference frame:
  - parallel lines intersect at infinity;
  - points at infinity cannot be represented.
- By adding a fourth coordinate to the triples, s.t. $\begin{bmatrix} kx & ky & kz \end{bmatrix}$ becomes $\begin{bmatrix} kx & ky & kz & k \end{bmatrix}$ $\forall k \neq 0$, **homogeneous** (or projective) **coordinates** are obtained, which are associated with **Projective Space** ($\mathbb{P}^3$).
- In $\mathbb{P}^3$ a point in space is represented by an *equivalence class* of *quadruples*, wherein equivalent quadruples differ just by a multiplicative factor.
- Any point $\begin{bmatrix} x & y & z & 0 \end{bmatrix} \in \mathbb{P}^3$ cannot be represented in 3D Euclidean space, since it corresponds to a point $\begin{bmatrix} x/0 & y/0 & z/0 \end{bmatrix}$ at **infinity** (not valid in $\mathbb{R}^3$).
- The point $\begin{bmatrix} 0 & 0 & 0 & k \end{bmatrix}$ $\forall k \neq 0$ is the origin of $\mathbb{R}^3$, whereas the point $\begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$ is undefined.
- All points at infinity of $\mathbb{P}^3$ lie on a plane, called *plane at infinity*.
- Extension to Euclidean spaces of any other dimension straightforward (sufficient to add an extra coordinate).

Perspective Projection in projective coordinates

- The **non-linear** transformation $u = x \cdot f/z,\ v = y \cdot f/z$ map 3D points to image points.
- The corresponding image point of the 3D point $\mathbf{M} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ is $\mathbf{m} = \begin{bmatrix} u & v \end{bmatrix}^T$.
- The representations of $\mathbf{M}$ and $\mathbf{m}$ in projective coordinates are the followings:

$$\tilde{\mathbf{M}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix},\ \tilde{\mathbf{m}} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

- The perspective projection becomes a **linear** transformation:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f\frac{x}{z} \\ f\frac{y}{z} \\ 1 \end{bmatrix} = \begin{bmatrix} fx \\ fy \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

  or, in matrix notation, $k\tilde{\mathbf{m}} = \tilde{\mathbf{P}}\tilde{\mathbf{M}}$.
- Often the transformation is expressed as $\tilde{\mathbf{m}} \approx \tilde{\mathbf{P}}\tilde{\mathbf{M}}$, where $\approx$ means "equal up to an arbitrary scale factor".

Vanishing points in projective coordinates

- Given a 3D point at infinity, its representation in projective coordinate would be $\begin{bmatrix} a & b & c & 0 \end{bmatrix}^T$.
- By applying the linear transformation, it's possible to obtain the coordinates of the vanishing point:

$$\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix} = \begin{bmatrix} fa \\ fb \\ c \end{bmatrix} = \begin{bmatrix} f\frac{a}{c} \\ f\frac{b}{c} \\ 1 \end{bmatrix} \Rightarrow u = f\frac{a}{c},\ v = f\frac{b}{c}$$

- The cosine direction of lines parallel to z axis is:

$$\begin{bmatrix} \cos(\frac{\pi}{2}) \\ \cos(\frac{\pi}{2}) \\ \cos(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

  and its projection is:

$$\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Rightarrow u = 0,\ v = 0$$

  which means that the vanishing point of lines parallel to z axis is the center of the image center.

Perspective Projection Matrix - PPM

- Matrix $\tilde{\mathbf{P}}$, known as **Perspective Projection Matrix** (PPM), represents the geometric camera model.
- Assuming distances measure in focal length units ($f = 1$), the PPM becomes:

$$\tilde{\mathbf{P}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = [\mathbf{I}|\mathbf{0}]$$

  which is called *canonical* or *standard* PPM.
- The core operation carried out by the perspective projection is to scale lateral coordinates (x, y) according to the distance from the cameras (z); the focal length f introduces an additional and

fixed (that is independent from z) scaling factor.

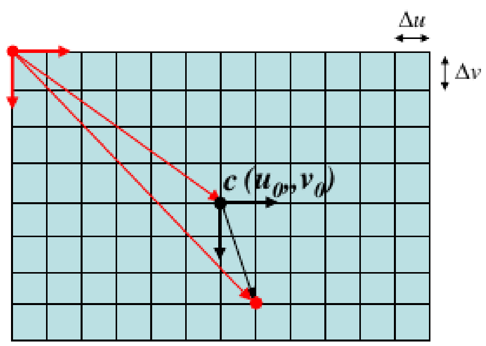
2b.2. A more comprehensive camera model

- Two additional issues must be taken into account:
  - Image digitalization.
  - The rigid motion (6 degrees of freedom, i.e. 3D rotation and translation) between the Camera Reference Frame (CRF) and the World Reference Frame (WRF).

Image Digitalization

- It can be accounted for by including into the projection equations the scaling factors along the two axes due to quantization $\Rightarrow$ the u, v coordinates are divided by the horizontal and vertical pixel size.

- It is necessary to model the translation of piercing point (intersection between optical axis and image plane) w.r.t. origin of the image coordinate system (top-left corner of the image) $\Rightarrow$ the vector obtained through the transformation is summed to the vector from 0 to c, which is $[u_0, v_0]^T$.



Δu= horizontal pixel size
Δv= vertical pixel size

$$u = \frac{f}{z} x \qquad \rightarrow u = \frac{1}{\Delta u} \frac{f}{z} x = k_u \frac{f}{z} x + u_0$$

$$v = \frac{f}{z} y \qquad \rightarrow v = \frac{1}{\Delta v} \frac{f}{z} y = k_v \frac{f}{z} y + v_0$$

- The PPM can be rewritten as:

$$\tilde{\mathbf{P}} = \begin{bmatrix} fk_u & 0 & u_0 & 0 \\ 0 & fk_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} fk_u & 0 & u_0 \\ 0 & fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \mathbf{A}\,[\mathbf{I}|\mathbf{0}]$$

  where $\mathbf{A}$ is the **Intrinsic Parameter Matrix**, and models the characteristics of the image sensing device.

- Intrinsic parameters can be reduced in number by setting $\alpha_u = fk_u,\ \alpha_v = fk_v$, which represent the focal length expressed in horizontal and vertical pixel sizes $\Rightarrow$ 4 intrinsic parameters **to calibrate**.

- A more general model would include a fifth parameter, the *skew*, to account for possible non-orthogonality between the axis of the image sensor; it would be $\mathbf{A}(1, 2)$, but in practice it is usually $\cot(\pi/2) = 0$.

Rigid motion

- 3D coordinates are not measured in the Camera Reference Frame (CRF), but in the World Reference Frame (WRF) external to the camera.

- WRF is related to CRF by:
  - rotation around optical center (expressed by 3×3 rotation matrix $\mathbf{R}$);
  - translation (expressed by 3×1 translation vector $\mathbf{T}$).
- The relation between the coordinates of a point in the two reference systems is:

$$\mathbf{W} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \mathbf{M} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \mathbf{M} = \mathbf{RW} + \mathbf{T}$$

  which can be rewritten in projective coordinates as:

$$\tilde{\mathbf{W}} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tilde{\mathbf{M}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \Rightarrow \tilde{\mathbf{M}} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0} & 1 \end{bmatrix} \cdot \tilde{\mathbf{W}} = \mathbf{G}\tilde{\mathbf{W}}$$

  where $\mathbf{G}$ is a 4×4 matrix, and it **must be calibrated**.
- The mapping between a 3D point in the CRF and an image point is the following:

$$\begin{cases} k\tilde{\mathbf{m}} = \tilde{\mathbf{P}}\tilde{\mathbf{M}} \\ \tilde{\mathbf{P}} = \mathbf{A}\left[\mathbf{I}|\mathbf{0}\right] \\ \tilde{\mathbf{M}} = \mathbf{G}\tilde{\mathbf{W}} \end{cases} \Rightarrow k\tilde{\mathbf{m}} = \mathbf{A}\left[\mathbf{I}|\mathbf{0}\right]\mathbf{G}\tilde{\mathbf{W}} = \mathbf{A}\left[\mathbf{I}|\mathbf{0}\right]\begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0} & 1 \end{bmatrix}\tilde{\mathbf{W}}$$

  so the general form of the PPM can be expressed either as $\tilde{\mathbf{P}} = \mathbf{A}\left[\mathbf{I}|\mathbf{0}\right]\mathbf{G}$ or $\tilde{\mathbf{P}} = \mathbf{A}\left[\mathbf{R}|\mathbf{T}\right]$.
- Matrix $\mathbf{G}$ encodes position and orientation of the camera w.r.t. WRF (6 *extrinsic* parameters, i.e. the rotations angles around the 3 axes, and the 3 translation degrees of freedom), and is called **Extrinsic Parameter Matrix**.
- The general form of the PPM encodes:
  - the position of the camera w.r.t. WRF into $\mathbf{G}$;
  - the perspective projection carried out by a pinhole camera into the canonical PPM $\left[\mathbf{I}|\mathbf{0}\right]$;
  - the actual characteristics of the sensing device into $\mathbf{A}$.

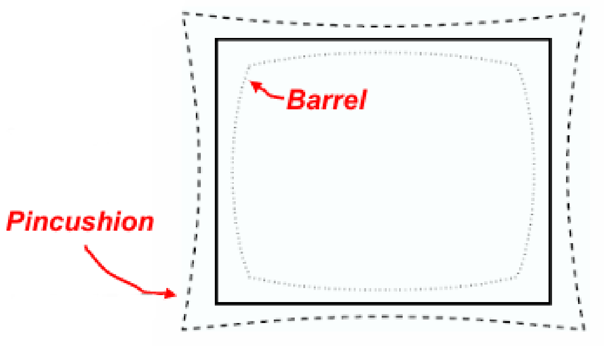
2b.3. Lens Distortion

- The PPM is based on the pinhole camera model.
- Real lenses introduces distortions:
  - **radial distortion** (lens curvature);
  - **tangential distortion** (misalignment of optical components).
- Lens distortion is modeled through a non-linear transformation which maps ideal *undistorted* image coordinates $(\tilde{x}, \tilde{y})$ into the observed *distorted* ones (x', y'):

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = L(r)\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} + \begin{pmatrix} d\tilde{x} \\ d\tilde{y} \end{pmatrix}$$

  depending on the distance r from the distortion center $(\tilde{x}_c, \tilde{y}_c)$:

$$r = \sqrt{(\tilde{x} - \tilde{x}_c)^2 + (\tilde{y} - \tilde{y}_c)^2}$$



- For lens distortion, continuous coordinates are used (undistortion is applied after perspective projection and before pixelization).

Lens distortion parameters

- The radial distortion function $L(r)$ is a non-linear function defined for positive r only, and such as $L(0) = 1$; it is tipically approximated by its Taylor series up to a certain order:

$$L(r) = 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 + \dots$$

- The tangential distortion is instead approximated as follows:

$$\begin{pmatrix} d\tilde{x} \\ d\tilde{y} \end{pmatrix} = \begin{pmatrix} 2p_1 \tilde{x}\tilde{y} + p_2 (r^2 + 2\tilde{x}^2) \\ p_1 (r^2 + 2\tilde{y}^2) + 2p_2 \tilde{x}\tilde{y} \end{pmatrix}$$

- The set of lens distortion parameters is composed of:
  - radial distortion coefficients $k_1, \dots, k_n$;
  - distortion center $(\tilde{x}_c, \tilde{y}_c)$;
  - tangential distortion coefficients p_1, p_2.
- For the sake of simplicity, distortion center is taken to coincide with image center, that is the piercing point.

Image Formation Flow

1. Transformation of 3D points from WRF to CRF, according to extrinsic parameters:

$$\mathbf{M} = \mathbf{R}\mathbf{W} + \mathbf{T}$$

2. Canonical perspective projection (scaling by the third coordinate):

$$\tilde{x} = x/z, \;\; \tilde{y} = y/z$$

3. Non-linear mapping due to lens distortion:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = L(r) \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} + \begin{pmatrix} d\tilde{x} \\ d\tilde{y} \end{pmatrix}$$

4. Mapping from image coordinates to pixels coordinates according to intrinsic parameters:

$$\mathbf{m} = \mathbf{A} \cdot \begin{pmatrix} x' \\ y' \end{pmatrix}$$

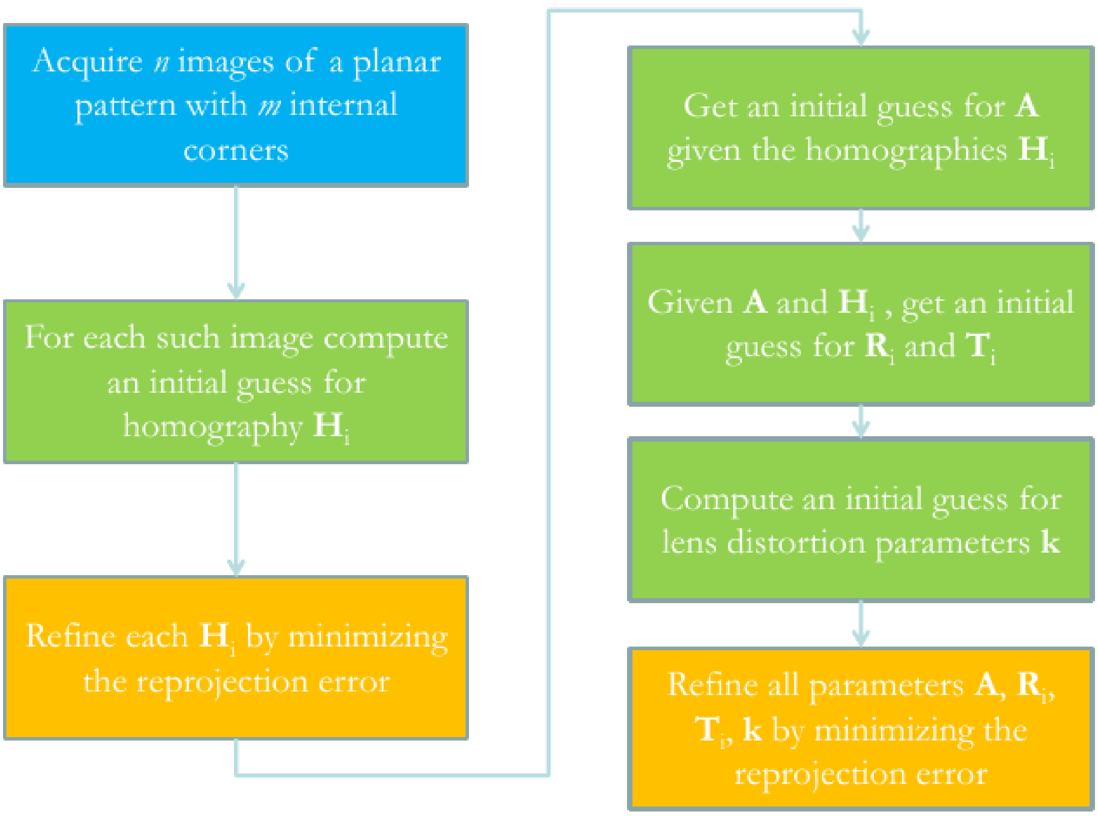
2b.4. Calibration

- Calibration is about manually finding correspondences between known 3D coordinates and pixels.
- Unknown parameters to calibrate (**at least $\mathbf{10}$**):
  - Intrinsic parameters (matrix $\mathbf{A}$):
    - α_u (focal length expressed in horizontal pixel size);
    - α_v (focal length expressed in vertical pixel size);
    - u_0, v_0 (coordinates of image center).
  - Extrinsic parameters (matrix $\mathbf{G}$):
    - $\mathbf{R}$ (rotation matrix);
    - $\mathbf{T}$ (translation matrix).
  - Lens distortion parameters.
- The basic process of calibration relies on setting up a system of *linear* equations given a set of 3D-2D correspondences, and then on solving such equations for the unknown camera parameters.

Calibration Targets

- To obtain required correspondences specific physical objects (known as **calibration targets**), having easily detectable features (e.g. chessboard), are used.
- Main approaches:
  - single image featuring several (at least 2) planes containing a known pattern (target difficult to build accurately);
  - several (at least 3) different images of one given planar pattern (target easier to build accurately).

Zhang's Method



- Given a planar chessboard pattern (others are possible), two things are known:
  - number of internal corners of the pattern (different along the two orthogonal directions for disambiguation);
  - the size of the squares which form the pattern.
- The first step is to acquire n **images** of a planar chessboard pattern with m **internal corners**.
- Internal corners can be detected easily by standard algorithms (e.g. Harris corner detector with sub-pixel refinement for improved accuracy).
- In each image, the WRF is taken at the top-left corner of the pattern, with x, y aligned to the orthogonal directions and plane $z = 0$ given by the pattern itself.
- Two main steps carried out:
  - initial guess by linear optimization (**minimization of algebraic error**);
  - refinement by non-linear minimization (**minimization of geometric error**).
- Each image requires its own estimate of extrinsic parameters, since the WRF will change between each calibration image $\Rightarrow$ transformation chaining to relate every WRF to a unique one.

$\mathbf{P}$ as a Homography

- In each calibration image, only 3D points with $z = 0$ are considered $\Rightarrow$ simpler transformation:

$$k\tilde{\mathbf{m}} = \tilde{\mathbf{P}}\tilde{\mathbf{W}} = \begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,3} & p_{1,4} \\ p_{2,1} & p_{2,2} & p_{2,3} & p_{2,4} \\ p_{3,1} & p_{3,2} & p_{3,3} & p_{3,4} \\ p_{4,1} & p_{4,2} & p_{4,3} & p_{4,4} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,4} \\ p_{2,1} & p_{2,2} & p_{2,4} \\ p_{3,1} & p_{3,2} & p_{3,4} \\ p_{4,1} & p_{4,2} & p_{4,4} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{H}\tilde{\mathbf{W}}'$$

  where $\mathbf{H}$ is an **homography** and represents a general linear transformation between planes; $\mathbf{H}$ can be thought of a semplification of $\mathbf{P}$ in case the imaged object is planar.
- Given a pattern with m **corners**, m **systems of** 3 **linear equations** can be written, wherein both 3D and 2D coordinates are known (due to corners having been detected in the i^{th} image).
- The 9 elements of $\mathbf{H}_i$ are unknown; however, since $\mathbf{H}_i$, alike $\mathbf{P}_i$, are known up to an arbitrary scale, the independent elements of $\mathbf{H}_i$ are actually 8.

Estimating $\mathbf{H}_i$

- Given the vector $\tilde{\mathbf{W}}$ with the coordinates of a corner (**control points**) w.r.t. WRF ($\tilde{\mathbf{W}} \to \tilde{\mathbf{W}}'$ by removing z coordinate since $z = 0$), and the vector $\tilde{\mathbf{m}}$ with the pixel coordinates of the same corner, the following holds:

$$k\tilde{\mathbf{m}} = \mathbf{H}\tilde{\mathbf{W}}', \; \mathbf{H} = \begin{bmatrix} \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix}$$

  thus, their cross product is zero:

$$\tilde{\mathbf{m}} \times \mathbf{H}\tilde{\mathbf{W}}' = \mathbf{0}$$

$$\Rightarrow \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix} \cdot \tilde{\mathbf{W}}' = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{h}_1^T\tilde{\mathbf{W}}' \\ \mathbf{h}_2^T\tilde{\mathbf{W}}' \\ \mathbf{h}_3^T\tilde{\mathbf{W}}' \end{bmatrix} = \begin{bmatrix} v\mathbf{h}_3^T\tilde{\mathbf{W}}' - \mathbf{h}_2^T\tilde{\mathbf{W}}' \\ \mathbf{h}_1^T\tilde{\mathbf{W}}' - u\mathbf{h}_3^T\tilde{\mathbf{W}}' \\ u\mathbf{h}_2^T\tilde{\mathbf{W}}' - v\mathbf{h}_1^T\tilde{\mathbf{W}}' \end{bmatrix} = \mathbf{0}$$

then, by factoring out $\mathbf{H}^T$, the following is obtained:

$$\begin{bmatrix} \mathbf{0}^T & -\tilde{\mathbf{W}}'^T & v\tilde{\mathbf{W}}'^T \\ \tilde{\mathbf{W}}'^T & \mathbf{0}^T & -u\tilde{\mathbf{W}}'^T \\ -v\tilde{\mathbf{W}}'^T & u\tilde{\mathbf{W}}'^T & \mathbf{0}^T \end{bmatrix} \cdot \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} = \mathbf{A}\mathbf{h} = \mathbf{0}$$

where just 2 of the 3 previous equations (in 9 unknowns) are linearly independent, so only the first 2 are kept.

- Therefore **for each image a linear system of $2m$ equations in 9 unknowns** is deployed, and the initial estimation of $\mathbf{H}_i$ is obtained by **minimizing the algebraic error represented by the norm of vector $\mathbf{A}\mathbf{h}$** and by enforcing constraint $\|\mathbf{h}\| = 1$ (Direct Linear Transform algorithm, or DLT).
- The solution of the estimation problem can be obtained by the **Singular Value Decomposition** (SVD) of matrix $\mathbf{A}$.
- Given previous initial estimation, $\mathbf{H}_i$ is later refined by applying **non-linear least squares** method to minimize the difference between the real pixel coordinates $\tilde{\mathbf{m}}_j$ and the predicted pixel coordinates $\mathbf{H}_i\tilde{\mathbf{W}}'_j$:

$$\min_{\mathbf{H}_i} \sum_j \|\tilde{\mathbf{m}}_j - \mathbf{H}_i\tilde{\mathbf{W}}'_j\|^2, \ j = 1 \ldots m$$

  which can be obtained in practice using **Levenberg-Marquardt** algorithm.
- This additional optimization step corresponds to the minimization of the reprojection error (also referred to as *geometric error*), measured for each of the 3D corners ($z = 0$) by comparing the real pixel coordinates to the ones estimated by the homography.

DLT algorithm - 4 points case

- By considering 4 point pairs, a system of 8 equations in 9 unknowns is obtained:

$$\mathbf{A}\mathbf{h} = \mathbf{0}, \ \mathbf{A} = [\,\mathbf{A}_1 \quad \ldots \quad \mathbf{A}_9\,], \ \mathbf{h} = \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_9 \end{bmatrix}$$

  where:
  - $\mathbf{A}$ is a 8×9 matrix;
  - $\mathbf{A}_1, \ldots, \mathbf{A}_9$ are the 8×1 column vectors composing $\mathbf{A}$;
  - $\mathbf{h}$ is a 9×1 vector;
  - $\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3$ are the 3×1 vectors composing $\mathbf{h}$ and representing the rows of the 3×3 matrix $\mathbf{H}$ which defines the homography;
  - $h_1, \ldots, h_9$ are the 9 elements of $\mathbf{H}$.
- Since $\mathbf{H}$ is defined up to a certain scale factor, there are two possibilities:
  - Set $h_9 = 1$, so as to obtain a non-homogeneous linear system with 8 equations in 8 unknowns:

$$\tilde{\mathbf{A}}\tilde{\mathbf{h}} = \mathbf{b}, \ \tilde{\mathbf{A}} = [\,\mathbf{A}_1 \quad \ldots \quad \mathbf{A}_8\,], \ \tilde{\mathbf{h}} = \begin{bmatrix} h_1 \\ \vdots \\ h_8 \end{bmatrix}, \ \mathbf{b} = -\mathbf{A}_9$$

which can be solved by standard methods (e.g. Cramer's rule, Gaussian Elimination, etc.).

- Constraint $\|\mathbf{h}\| = 1$ and assume h_9 fixed:

$$\tilde{\mathbf{A}}\tilde{\mathbf{h}} = \mathbf{b},\ \tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_1 & \dots & \mathbf{A}_8 \end{bmatrix},\ \tilde{\mathbf{h}} = \begin{bmatrix} h_1 \\ \vdots \\ h_8 \end{bmatrix},\ \mathbf{b} = -h_9\mathbf{A}_9$$

$$\Rightarrow \tilde{\mathbf{h}} = -h_9\tilde{\mathbf{A}}^{-1}\mathbf{A}_9 \Rightarrow \mathbf{h} = \begin{bmatrix} -h_9\tilde{\mathbf{A}}^{-1}\mathbf{A}_9 \\ h_9 \end{bmatrix} = h_9\begin{bmatrix} -\tilde{\mathbf{A}}^{-1}\mathbf{A}_9 \\ 1 \end{bmatrix}$$

$$\Rightarrow \|\mathbf{h}\| = 1 \Rightarrow h_9\sqrt{\|\tilde{\mathbf{A}}^{-1}\mathbf{A}_9\|^2 + 1} = 1 \Rightarrow h_9 = \frac{1}{\sqrt{\|\tilde{\mathbf{A}}^{-1}\mathbf{A}_9\|^2 + 1}}$$

after h_9 is found, $\mathbf{h}$ can be computed.

Estimation of intrinsic parameters

- Since $\mathbf{H}_i$ is known up to a certain scale factor, the following relation between $\mathbf{H}_i$ and the PPM can be established:

$$\begin{cases} \mathbf{H} = \begin{bmatrix} \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_4 \end{bmatrix} \\ \tilde{\mathbf{P}} = \mathbf{A}\begin{bmatrix} \mathbf{I}|\mathbf{0} \end{bmatrix}\mathbf{G} = \mathbf{A}\begin{bmatrix} \mathbf{R}|\mathbf{T} \end{bmatrix} = \mathbf{A}\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{T} \end{bmatrix} \end{cases}$$
$$\Rightarrow \mathbf{H} = \lambda\mathbf{A}\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{T} \end{bmatrix}$$

- $\mathbf{R}$ is an orthogonal matrix (its vectors are orthonormal), therefore the following constraints hold:

$$\mathbf{r}_1^T \cdot \mathbf{r}_2 = 0 \Rightarrow \mathbf{h}_1^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_2 = 0$$
$$\mathbf{r}_1^T\mathbf{r}_1 = \mathbf{r}_2^T\mathbf{r}_2 \Rightarrow \mathbf{h}_1^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_1 = \mathbf{h}_2^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{h}_2$$

where $\mathbf{A}^{-T}$ is the transpose of the inverse of $\mathbf{A}$.

- The unknowns are the entries of $\mathbf{B} = \mathbf{A}^{-T}\mathbf{A}^{-1}$; since $\mathbf{A} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$ is upper triangular, $\mathbf{B}$ turns out to be symmetric, so the unknowns are just 6.
- Given n calibration images, by stacking together the above two equations a $2n \times 6$ linear system is obtained, which can be solved in case **at least 3 calibration images** are available (if there are more, the system can be solved with a least squares approach).
- By posing:

$$\mathbf{B} = \mathbf{A}^{-T}\mathbf{A}^{-1} = \begin{bmatrix} B_{1,1} & B_{1,2} & B_{1,3} \\ B_{1,2} & B_{2,2} & B_{2,3} \\ B_{1,3} & B_{2,3} & B_{3,3} \end{bmatrix},$$
$$\mathbf{b} = \begin{bmatrix} B_{1,1} & B_{1,2} & B_{2,2} & B_{1,3} & B_{2,3} & B_{3,3} \end{bmatrix}^T,$$
$$\mathbf{h}_i^T = \begin{bmatrix} h_{i,1} & h_{i,2} & h_{i,3} \end{bmatrix},$$
$$\mathbf{v}_{i,j}^T = \begin{bmatrix} h_{i,1}h_{j,1} & h_{i,1}h_{j,2} + h_{i,2}h_{j,1} & h_{i,2}h_{j,2} & h_{i,1}h_{j,3} + h_{i,3}h_{j,1} & h_{i,2}h_{j,3} + h_{i,3}h_{j,2} & h_{i,3}h_{j,3} \end{bmatrix}$$

it can be noticed that:

$$\mathbf{h}_i^T\mathbf{B}\mathbf{h}_j = \mathbf{v}_{i,j}^T\mathbf{b} \Rightarrow \begin{cases} \mathbf{h}_1^T\mathbf{B}\mathbf{h}_2 = 0 \Rightarrow \mathbf{v}_{1,2}^T\mathbf{b} = 0 \\ \mathbf{h}_1^T\mathbf{B}\mathbf{h}_1 = \mathbf{h}_2^T\mathbf{B}\mathbf{h}_2 \Rightarrow \mathbf{v}_{1,1}^T\mathbf{b} = \mathbf{v}_{2,2}^T\mathbf{b} \Rightarrow (\mathbf{v}_{1,1} - \mathbf{v}_{2,2})^T\mathbf{b} = 0 \end{cases}$$

therefore, each image provides 2 equations in 6 independent unknowns in $\mathbf{B}$, s.t. with n calibration images a homogeneous linear system in the form $\mathbf{V}\mathbf{b} = 0$ is obtained (it can be solved with a least squares approach).

- Once $\mathbf{b}$ is computed, the intrinsic parameters $\mathbf{A}$ can be obtained in a closed form.

Estimation of extrinsic parameters

- Given $\mathbf{A}$ and having obtained $\mathbf{H}_i$ for each image, it is possible to compute $\mathbf{R}_i, \mathbf{T}_i$ for each image i:

$$\mathbf{H}_i = \begin{bmatrix} \mathbf{h}_{1,i} & \mathbf{h}_{2,i} & \mathbf{h}_{3,i} \end{bmatrix} = \lambda\mathbf{A}\begin{bmatrix} \mathbf{r}_{1,i} & \mathbf{r}_{2,i} & \mathbf{T}_i \end{bmatrix}$$
$$\mathbf{h}_{k,i} = \lambda\mathbf{A}\mathbf{r}_{k,i} \Rightarrow \lambda\mathbf{r}_{k,i} = \mathbf{A}^{-1}\mathbf{h}_{k,i},\ k = 1,2$$

- As $\mathbf{r}_{k,i}$ is a unit vector:

$$\mathbf{r}_{k,i} = \frac{1}{\lambda}\mathbf{A}^{-1}\mathbf{h}_{k,i},\ \lambda = \|\mathbf{A}^{-1}\mathbf{h}_{k,i}\|,\ k = 1,2$$

- $\mathbf{r}_3$ can be derived from $\mathbf{r}_1, \mathbf{r}_2$ by exploiting orthonormality:

$$\mathbf{r}_{3,i} = \mathbf{r}_{1,i} \times \mathbf{r}_{2,i}$$

- Finally, $\mathbf{T}_i$ is computed:

$$\mathbf{T}_i = \frac{1}{\lambda}\mathbf{A}^{-1}\mathbf{h}_{3,i}$$

- The rotation matrix found with this approach is not perfect, but provides a good approximation; to compute a proper rotation matrix, SVD can be applied.

Estimation of lens distortion parameters

- Given the homographies, both real distorted coordinates of the corners found in the images and the corresponding ideal undistorted coordinates predicted by the homographies are known; such information are used to estimate distortion coefficients k_1, k_2 of the radial distortion function.
- Given the already known intrinsic parameter matrix $\mathbf{A}$, the relationship between distorted (u', v') and ideal $(\tilde{u}, \tilde{v})$ **pixel** coordinates is:

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \mathbf{A}\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \Rightarrow \begin{cases} x' = \frac{u' - u_0}{\alpha_u} \\ y' = \frac{v' - v_0}{\alpha_v} \end{cases}$$

and by applying the lens distortion model:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = L(r)\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = (1 + k_1r^2 + k_2r^4)\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix}$$

the following relationship is found:

$$\begin{cases} \frac{u'-u_0}{\alpha_u} = (1 + k_1 r^2 + k_2 r^4)\frac{\tilde{u}-u_0}{\alpha_u} \\ \frac{v'-v_0}{\alpha_v} = (1 + k_1 r^2 + k_2 r^4)\frac{\tilde{v}-v_0}{\alpha_v} \end{cases}$$
$$\Rightarrow \begin{cases} u' = \tilde{u} + (k_1 r^2 + k_2 r^4)(\tilde{u} - u_0) \\ v' = \tilde{v} + (k_1 r^2 + k_2 r^4)(\tilde{v} - v_0) \end{cases}$$

- It is possible to set up a linear system where the unknowns are the distortion coefficients:

$$\begin{bmatrix} (\tilde{u} - u_0)r^2 & (\tilde{u} - u_0)r^4 \\ (\tilde{v} - v_0)r^2 & (\tilde{v} - v_0)r^4 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} u' - \tilde{u} \\ v' - \tilde{v} \end{bmatrix}$$

  where the squared distance r^2 from the distortion center, assumed coincident with the image center (u_0, v_0), is:

$$r^2 = x'^2 + y'^2 = \left(\frac{u' - u_0}{\alpha_u}\right)^2 + \left(\frac{v' - v_0}{\alpha_v}\right)^2$$

- Given n images with m corner features, it is possible to set up a linear system with $2nm$ equations in 2 unknowns, and apply a least squares approach:

$$\mathbf{Dk} = \mathbf{d} \Rightarrow \mathbf{k} = \mathbf{D}^{\dagger}\mathbf{d} = (\mathbf{D}^T\mathbf{D})^{-1}\mathbf{D}^T\mathbf{d}$$

Refinement by non-linear optimization

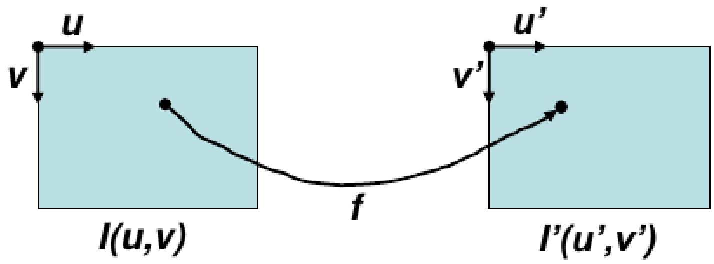
- Procedure highlighted so far seeks to minimize an algebraic error.
- A more accurate solution can be found by applying the **Maximum Likelihood Estimation** (MLE) to minimize the geometric reprojection error.
- Under the hypothesis of independent identically distributed noise, the MLE of the camera model is obtained by minimizing the error:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \|\mathbf{m}_{i,j} - \hat{\mathbf{m}}(\mathbf{A}, \mathbf{k}, \mathbf{R}_i, \mathbf{T}_i, \mathbf{w}_j)\|^2$$

  w.r.t. all unknown camera parameters.
- The solution of above non-linear optimization problem is provided by **Levenberg-Marquardt** algorithm.

2b.5. Image warping

- Image warping is about **transforming pixel coordinates** from a source image to pixel coordinates in a second image, called *target*.



- It is defined by two mapping functions:

$$\begin{cases} u' = f_u(u,v) \\ v' = f_v(u,v) \end{cases} \Rightarrow I'(f_u(u,v), f_v(u,v)) = I(u,v)$$

which, given pixel coordinates in source image, computes the corresponding horizontal and vertical coordinates in target image.

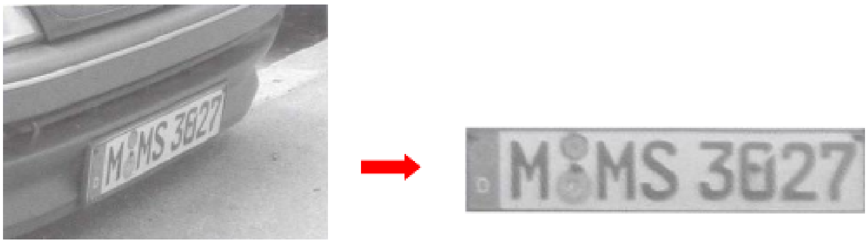
- Examples:
  - Rotation:

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

  - Removal of perspective deformation:

$$s \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} \\ h_{2,1} & h_{2,2} & h_{2,3} \\ h_{3,1} & h_{3,2} & h_{3,3} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

    in which the homography is estimated using at least 4 correspondences (if there are more, least squares estimation, more robust to noise, can be applied).



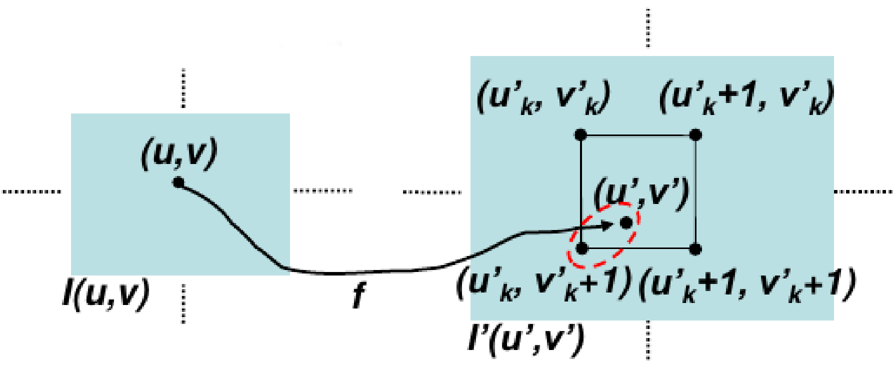
  - Lens distortion correction.
  - Stereo rectification.

Forward/Backward Mapping

The coordinates obtained from the transformation are often real numbers and not integers, and therefore they might not correspond to pixels of the target image.

- Forward Mapping:
  - the real coordinates are mapped to the closest point into the target image:

$$(u'_k, v'_k) = (\lfloor u' \rceil, \lfloor v' \rceil)$$



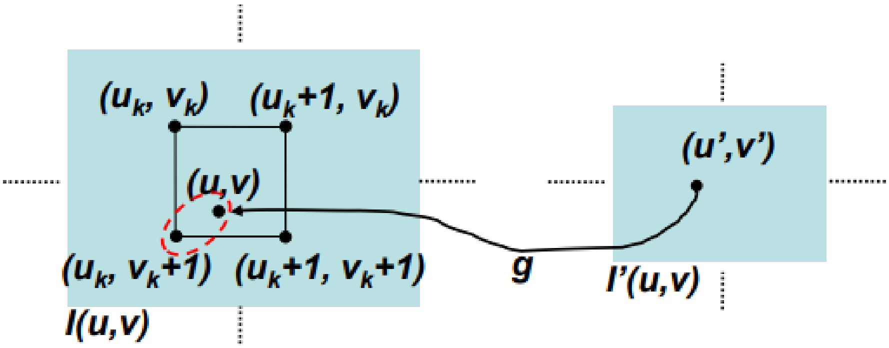
  - this results in the presence of *holes* and *folds* in $I'(u', v')$.

- Backward Mapping:
    - the mapping is inverted, thus every pixel in the target is mapped to a pixel in the source, which in general will be a real value:

$$\begin{cases} u = g_u(u', v') \\ v = g_v(u', v') \end{cases} \Rightarrow \forall (u', v') : I'(u', v') = I(g_u(u', v'), g_v(u', v'))$$

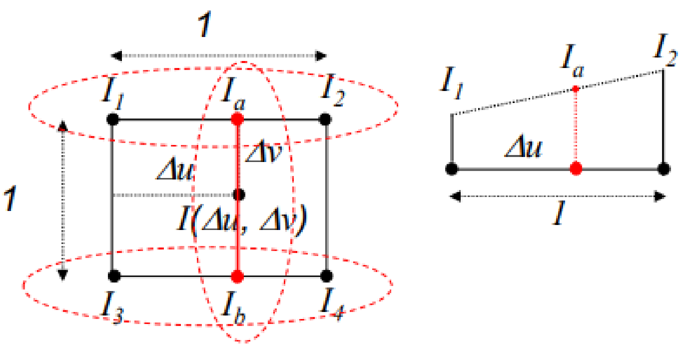
    - the real coordinates on the source image can be mapped following **two mapping strategies**:
        - **Nearest Neighbour Mapping**: the value of the closest pixel is chosen:

$$(u_k, v_k) = (\lfloor u \rceil, \lfloor v \rceil)$$

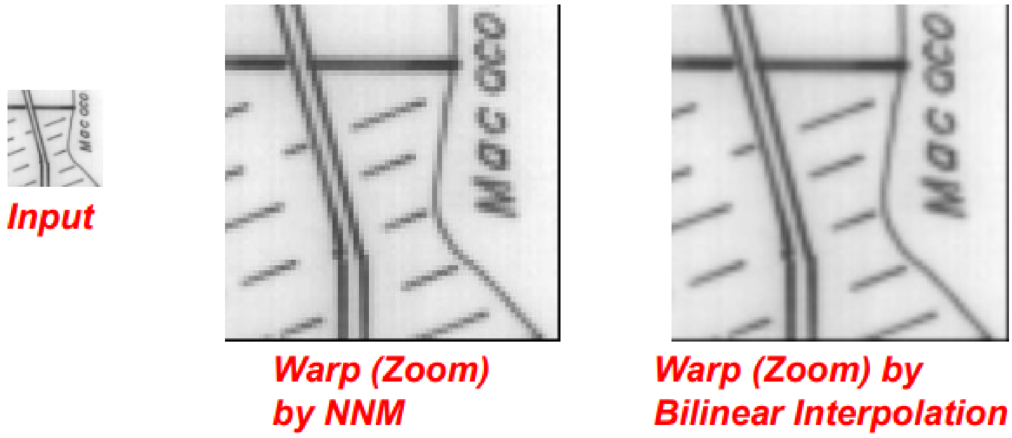


        - **Interpolation**: the value of the pixel is determined by an interpolation between the 4 closest pixels (e.g. **bilinear**):

$$\Delta u = u - u_k, \Delta v = v - v_k$$
$$I_1 = I(u_k, v_k), I_2 = I(u_k + 1, v_k), I_3 = I(u_k, v_k + 1), I_4 = (u_k + 1, v_k + 1)$$



$$\frac{I_a - I_1}{\Delta u} = I_2 - I_1, \ \frac{I_b - I_3}{\Delta u} = I_4 - I_3$$
$$\Rightarrow I_a = (I_2 - I_1)\Delta u + I_1, \ I_b = (I_4 - I_3)\Delta u + I_3$$
$$\Rightarrow I(\Delta u, \Delta v) = (I_b - I_a)\Delta v + I_a$$
$$\Rightarrow I(\Delta u, \Delta v) = ((I_4 - I_3)\Delta u + I_3 - ((I_2 - I_1)\Delta u + I_1))\Delta v + (I_2 - I_1)\Delta u + I_1$$
$$\Rightarrow I(\Delta u, \Delta v) = (I_2 - I_1)\Delta u + (I_3 - I_1)\Delta v + (I_4 - I_3 - I_2 - I_1)\Delta u \Delta v + I_1$$
$$\Rightarrow I(\Delta u, \Delta v) = a\Delta u + b\Delta v + c\Delta u \Delta v + d$$
$$\Rightarrow I'(u', v') = (1 - \Delta u)(1 - \Delta v)I_1 + \Delta u(1 - \Delta v)I_2 + (1 - \Delta u)\Delta v I_3 + \Delta u \Delta v I_4$$

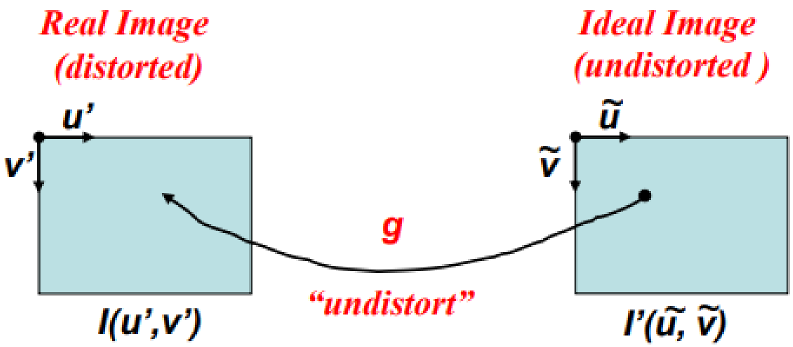


- in this way there are no *holes* nor *folds*.

Warping to compensate lens distortion

Once the lens distortion parameters have been computed by camera calibration, the image can be corrected by a backward warp from the undistorted to the distorted image, based on the adopted lens distortion model:

$$\forall(\tilde{u},\tilde{v}) : I'(\tilde{u},\tilde{v}) = I(g_u(\tilde{u},\tilde{v}), g_v(\tilde{u},\tilde{v}))$$



For example, using Zhang's calibration method:

$$\begin{cases} u' = \tilde{u} + (k_1 r^2 + k_2 r^4)(\tilde{u} - u_0) \\ v' = \tilde{v} + (k_1 r^2 + k_2 r^4)(\tilde{v} - v_0) \end{cases}$$