University of Bologna

Image Formation and Acquisition (Part 3)

Luigi Di Stefano (luigi.distefano@unibo.it)

Camera Calibration (1)

- We have now at disposal a camera model thoroughly explaining the image formation and digitization process.
- Such a model includes the PPM, which, in turn, can be decomposed into 3 independent items: intrinsic parameter matrix (**A**), rotation matrix (**R**), translation vector (**T**), as well as the lens distortion parameters.
- **Camera calibration** is the process whereby all parameters defining the camera model are -as accurately as possible- estimated for a specific camera device. Depending on the application, either the PMM only or also its independent components (**A**, **R**, **T**) need to be estimated.

$\tilde{P} = [p_1\ p_2\ p_3\ p_4]$ $P = A[I\ |\ 0]G$ estimate

- Many camera calibration algorithms do exist. The basic process, though, relies always on setting up a *linear* system of equations given a set of known 3D-2D correspondences, so as to then solve for the unknown camera parameters.
- To obtain the required correspondences specific physical objects (referred to as calibration targets) providing easily detectable image features (such as, e.g., chessboard or dot patterns) are typically deployed.

$$k\tilde{m}_j^{2D} = \underbrace{A[\,R \mid T\,]}\ \tilde{M}_j^{3D}$$

$k\tilde{m}_j^{2D}$ → pixels

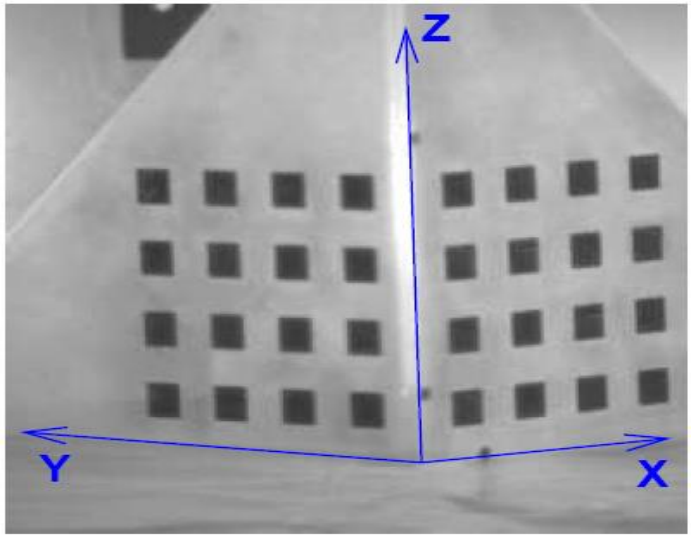
A → 6 DOF + 4 cam. param (A)

$\tilde{M}_j^{3D}$ → world ref. frame

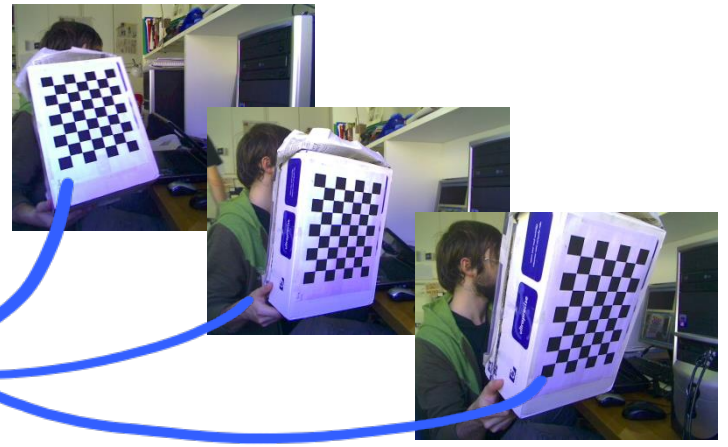
to estimate the parameter, we need to collect much $\tilde{m}_j$ of an object that we're already measured

Camera Calibration (2)

- Camera calibration approaches can be split into two main categories:
  - Those relying on a **single image** featuring **several** (at least 2) **planes** containing a known patterns**.**
  - Those relying on **several** (at least 3) different **images** of **one** given **planar pattern**.

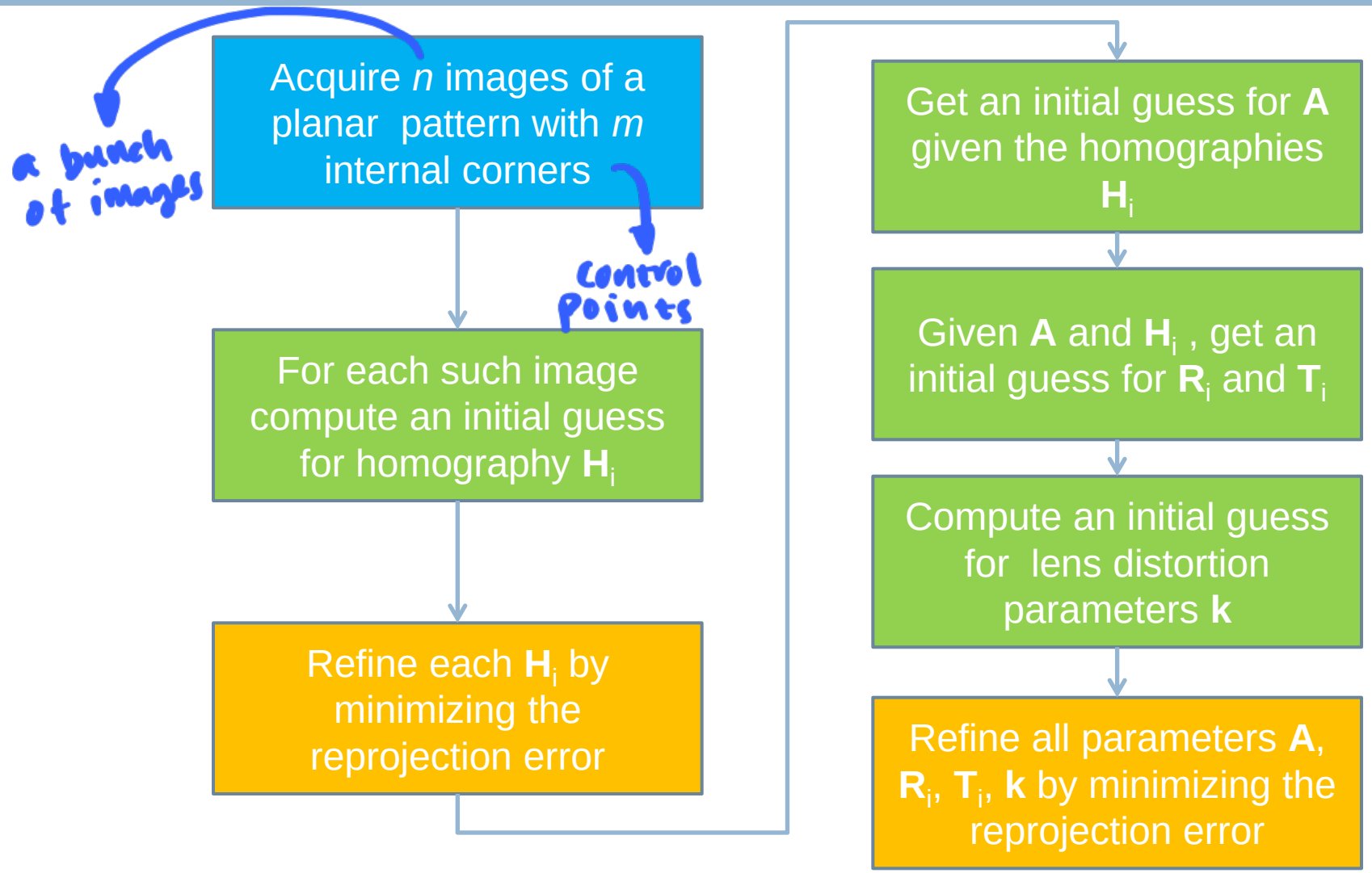


- In practise, it is difficult to build accurate targets containing multiple planes, while an accurate planar target can be attained rather easily.
- Implementing a camera calibration software requires a significant effort. However, the main Computer Vision toolboxes include specific functions (OpenCV, Matlab CC Toolbox, Halcon,...)



internal corners as control points

Zhang's Method (1)



Zhang's Method (2)

- Given a planar chessboard pattern, known are:
  - The number of internal corners of the pattern, different along the two orthogonal directions for the sake of disambiguation (i.e. rows, columns).
  - The size of the squares which form the pattern.
- Internal corners can be detected easily by standard algorithms (e.g. the Harris corner detector, possibly with sub-pixel refinement for improved accuracy).
- In each image, the 3D WRF is taken at the top-left corner of the pattern, with plane z=0 given by the pattern itself and the x,y axis aligned to the two orthogonal directions, in particular so as to keep always the same association between axis and directions (e.g. x=rows, y = columns). Thus, as for 3D points:
  - The third coordinate is always 0.
  - x and y are determined by the known size of the squares forming the chessboard.

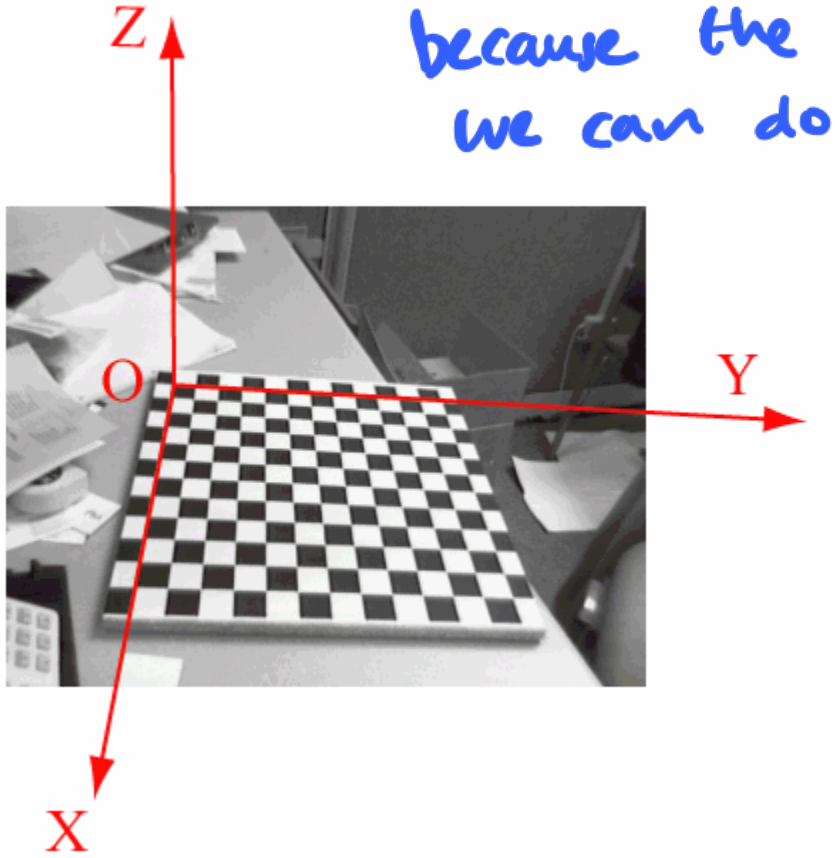
Estrinsic Parameters

- It is worth pointing out that each image requires its own estimate of the extrinsic parameters, as they are different from one to the other.
  - A global WRF can be taken to coincide with that associated with one of the calibration images, e.g. the first.

Needed :

- 3D coordinates of control points represented
- pixel coordinates in the image

because the target is planar, we can do the homography



Planar Calibration Target → Homographies

- Due to the calibration pattern being planar and the choice of the WRF associated with the calibration images, for each of them we can consider only 3D control points (i.e. corners) with z=0. Therefore, the mapping between the 3D coordinates of the control points and their projections in the calibration images is a **homography:**

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

corner coord. of corner

vector of coordinates of the corner

pixel coord. of corner

homography (estimate)

$k\tilde{m} = H\tilde{w}'$

- Given a pattern with *m* corner, we can write *m* systems of 3 linear equations as above, wherein both 3D (with z=0) as well as 2D coordinates are known due to corners having been detected in the calibration image and the unknowns being, therefore, the elements in **H**.

Estimating H (DLT Algorithm)

- Starting from the previous homography equation we can write:

$$k\tilde{\mathbf{m}} = \mathbf{H}\tilde{\mathbf{w}}' \Rightarrow \tilde{\mathbf{m}} \times \mathbf{H}\tilde{\mathbf{w}}' = \mathbf{0} \Rightarrow \begin{bmatrix} v\mathbf{h}_3^{\mathrm{T}}\tilde{\mathbf{w}}' - \mathbf{h}_2^{\mathrm{T}}\tilde{\mathbf{w}}' \\ \mathbf{h}_1^{\mathrm{T}}\tilde{\mathbf{w}}' - u\mathbf{h}_3^{\mathrm{T}}\tilde{\mathbf{w}}' \\ u\mathbf{h}_2^{\mathrm{T}}\tilde{\mathbf{w}}' - v\mathbf{h}_1^{\mathrm{T}}\tilde{\mathbf{w}}' \end{bmatrix} = \mathbf{0},\ \mathbf{H} = \begin{bmatrix} \mathbf{h}_1^{\mathrm{T}} \\ \mathbf{h}_2^{\mathrm{T}} \\ \mathbf{h}_3^{\mathrm{T}} \end{bmatrix}$$

Estimating homography in one calibr. image

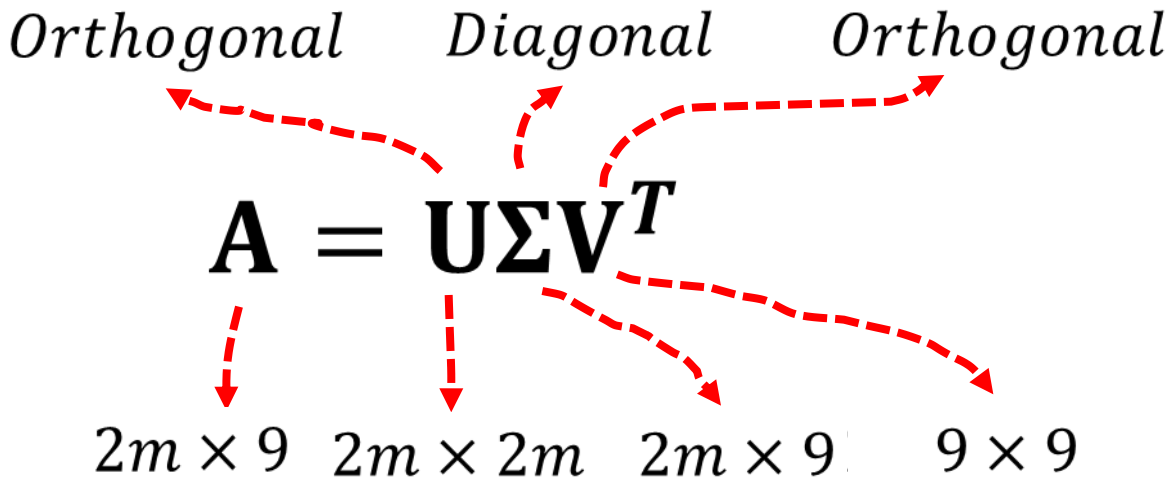
i.e.

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

2mx9 · 9x1

- Between the previous 3 equations in 9 unknowns, only 2 are linearly independent, so that typically only the first 2 are kept. By deploying all correspondences, for each image we build a linear system of *2m* equations in 9 unknowns. This allows for estimating **H** by minimizing the "algebraic error" represented by the norm of vector $\mathbf{Ah}$: $\mathbf{h}^* = \mathrm{argmin}_{\mathbf{h} \in R^9} \|\mathbf{Ah}\|$, subject to $\|\mathbf{h}\| = 1$ to avoid the trivial solution ($\mathbf{h}=\mathbf{0}$).
- It is known from *linear algebra* that the solution of the above over-constrained system of equations can be obtained in closed form by the Singular Value decomposition (SVD) of matrix **A.**

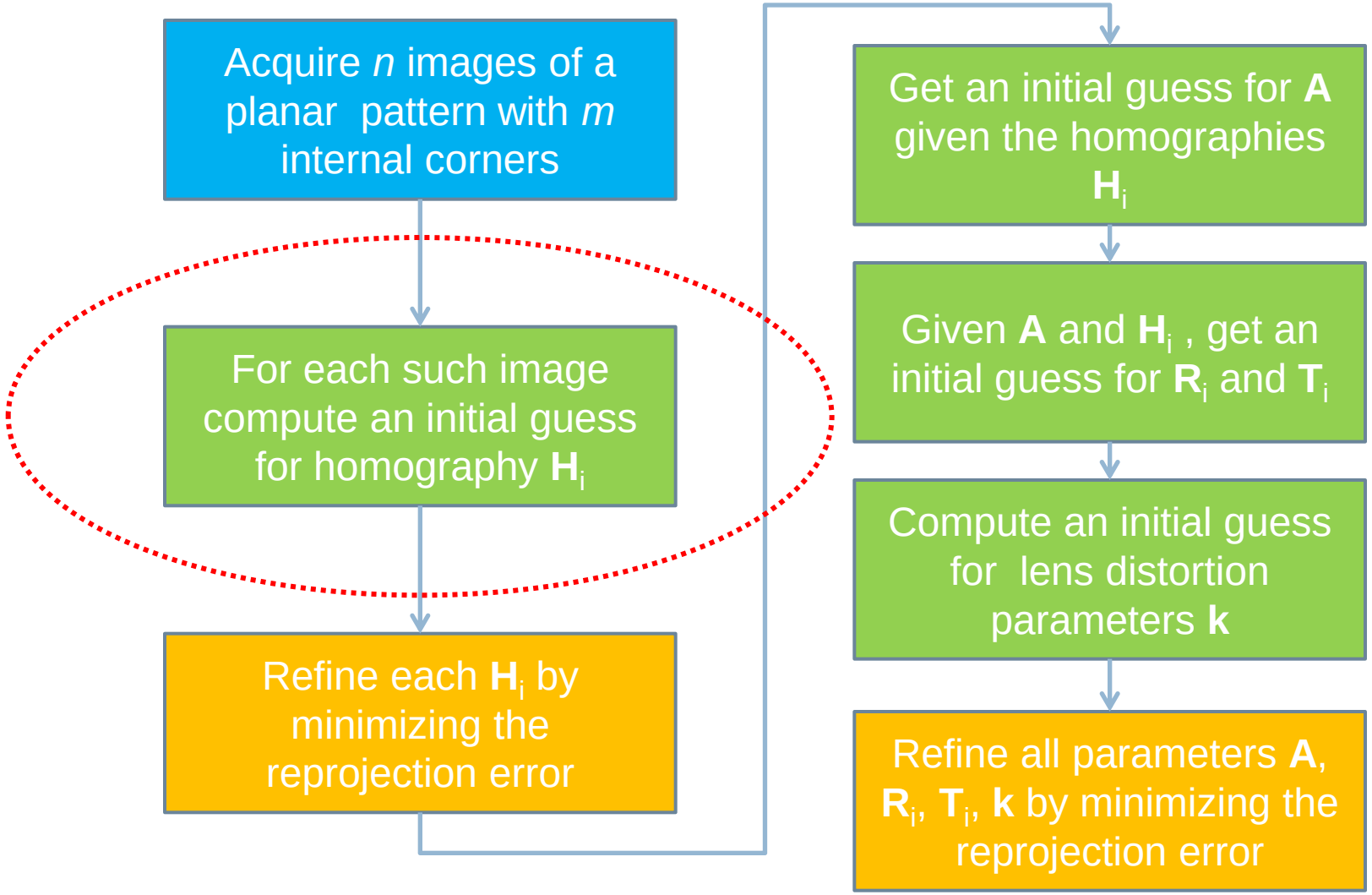
Estimating H by the SVD



$$\mathbf{V} = [\mathbf{v_1} \ldots . \mathbf{v_9}] \rightarrow \boldsymbol{h}^* = \mathbf{v_9}$$

See Appendix 1 for the solution of the homography estimation problems in case only 4 correspondences are available (i.e. the minimum number of required correspondences).

Zhang's Method



Non-linear Refinement

- Given the previous initial estimation, $\mathbf{H}$ is later refined by the least-squares solution of the non-linear minimization problem:

$$\mathbf{H} = \text{argmin}_{\mathbf{H} \in R^9} \sum_{j=1}^{m} \left\| \widetilde{m}_j - \mathbf{H}\widetilde{w}'_j \right\|^2$$

minimizes the distance

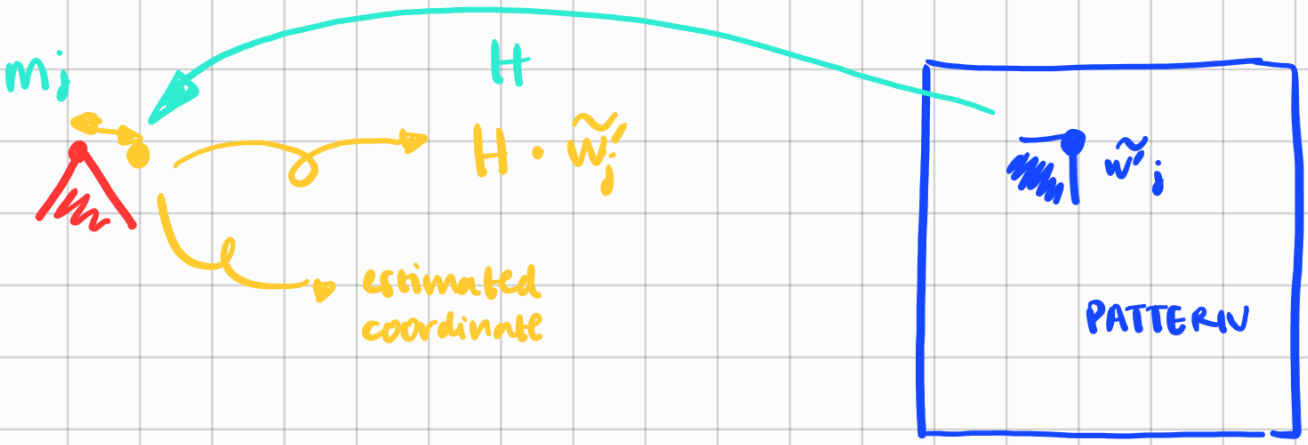
world corner

observed corner

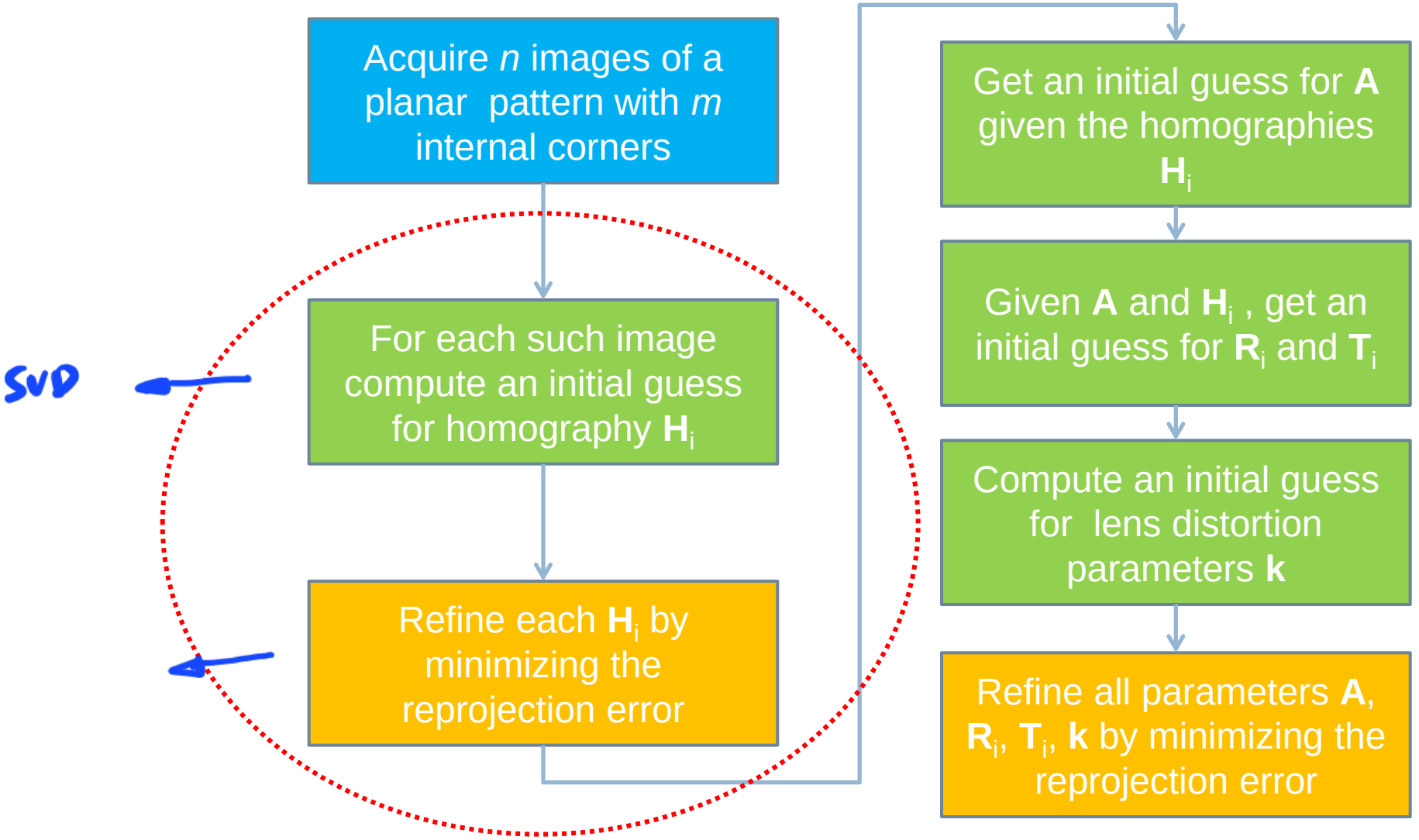
estimated / measured corner

which can be obtained in practice using by the Levenberg-Marquardt (LM) algorithm.

- This additional optimization steps corresponds to the minimization of the reprojection error measured for each of the 3D corners (with z=0) of the pattern by comparing the pixel coordinates predicted by the estimated homograpy to the pixel coordinates of the corresponding corner extracted in the image. The rationale is that the “best” homography would predict with the best accuracy the positions of the corner features actually found in the image. Such a reprojection error is typically referred to as “geometric error”.


m_j
H
$H \cdot \tilde{w}_j$
estimated coordinate
$\tilde{w}_j$
PATTERN

Zhang's Method



Estimation of the intrinsic parameters (1)

- As **H** is known up to a scale factor, we can establish the following relation between **H** and the PPM:

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_4 \end{bmatrix} = \lambda \mathbf{A} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{T} \end{bmatrix}$$

- **R** is an orthogonal matrix, its vectors being orthonormal. This constrains the intrinsic parameters to obey to the following relations:

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

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

where the unknowns are the entries of $\mathbf{B}=\mathbf{A}^{-T}\mathbf{A}^{-1}$. As **A** is upper triangular, **B** turns out to be symmetric, so that the unknowns are just 6.

- By stacking the above two equations provided by each calibration image, we obtain a 2*n* x 6 linear system, which can be solved in case at least 3 images are available (more details on next slide)

Estimation of the extrinsic parameters (2)

- By posing

$$\mathbf{B} = \mathbf{A}^{-\mathbf{T}}\mathbf{A}^{-1} = \begin{pmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{pmatrix}, \quad b = \left[B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}\right]^T$$

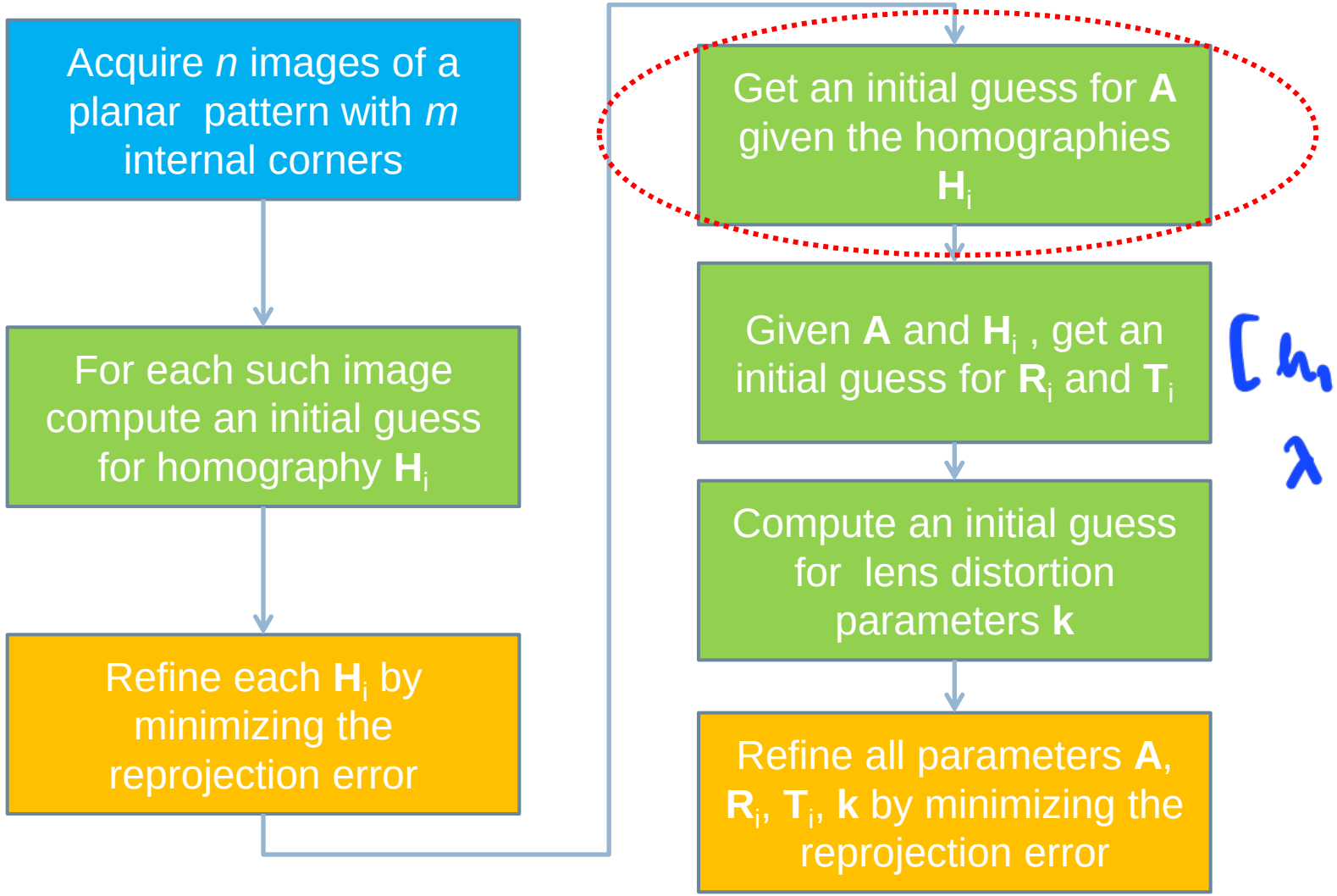
$$\mathbf{h}_i^{\,T} = \left[h_{i1}, h_{i2}, h_{i3}\right], \qquad \mathbf{v}_{ij}^{\,T} = \left[h_{i1}h_{j1},\ h_{i1}h_{j2} + h_{i2}h_{j1},\ h_{i2}h_{j2},\ h_{i3}h_{j1} + h_{i1}h_{j3},\ h_{i3}h_{j2} + h_{i2}h_{j3},\ h_{i3}h_{j3}\right]$$

it can be noticed that

$$\mathbf{h}_i^{\mathbf{T}}\mathbf{B}\,\mathbf{h}_j = \mathbf{v}_{ij}^{\,\mathbf{T}}\mathbf{b} \longrightarrow \begin{array}{l} \mathbf{h}_1^{\mathbf{T}}\mathbf{B}\mathbf{h}_2 = 0 \ \Rightarrow\ \mathbf{v}_{12}^{\,\mathbf{T}}\mathbf{b} = 0 \\ \mathbf{h}_1^{\mathbf{T}}\mathbf{B}\mathbf{h}_1 = \mathbf{h}_2^{\mathbf{T}}\mathbf{B}\mathbf{h}_2 \ \Rightarrow\ \mathbf{v}_{11}^{\,\mathbf{T}}\mathbf{b} = \mathbf{v}_{22}^{\,\mathbf{T}}\mathbf{b} \ \Rightarrow \left(\mathbf{v}_{11} - \mathbf{v}_{22}\right)^{\mathrm{T}}\ \mathbf{b} = 0 \end{array}$$

- Therefore, each image provides 2 equations in the 6 independent unknowns in $\mathbf{B}$, so that with *n* calibration images we get a homogeneous linear system of equations in the form $\mathbf{V}\mathbf{b} = 0$, which can be solved by the SVD.
- Once $\mathbf{b}$ has been calculated, the intrinsic parameters (i.e. $\mathbf{A}$) can be obtained in closed form.

Zhang's Method



Estimation of the extrinsic parameters

- Once $\mathbf{A}$ has been estimated, for each image it is possible to compute $\mathbf{R}$ and then $\mathbf{T}$ given the previously computed homography $\mathbf{H}$:

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \end{bmatrix} = \lambda \mathbf{A} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{T} \end{bmatrix}$$

how to estimate this?

$$\mathbf{h}_1 = \lambda \mathbf{A} \mathbf{r}_1 \quad \rightarrow \quad \mathbf{r}_1 = \frac{1}{\lambda} \mathbf{A}^{-1} \mathbf{h}_1$$

λ needs to be decided first. So, λ must be the norm of $\|\mathbf{A}^{-1}\mathbf{h}_1\|$

- As $\mathbf{r}_1$ is a unit vector:

$$\lambda = \left\| \mathbf{A}^{-1} \mathbf{h}_1 \right\|, \quad \mathbf{r}_2 = \frac{1}{\lambda} \mathbf{A}^{-1} \mathbf{h}_2$$

λ must be only one

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

$$\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$$

r_3 has to be perpendicular to r_1 and r_2

- Finally, we can get $\mathbf{T}$

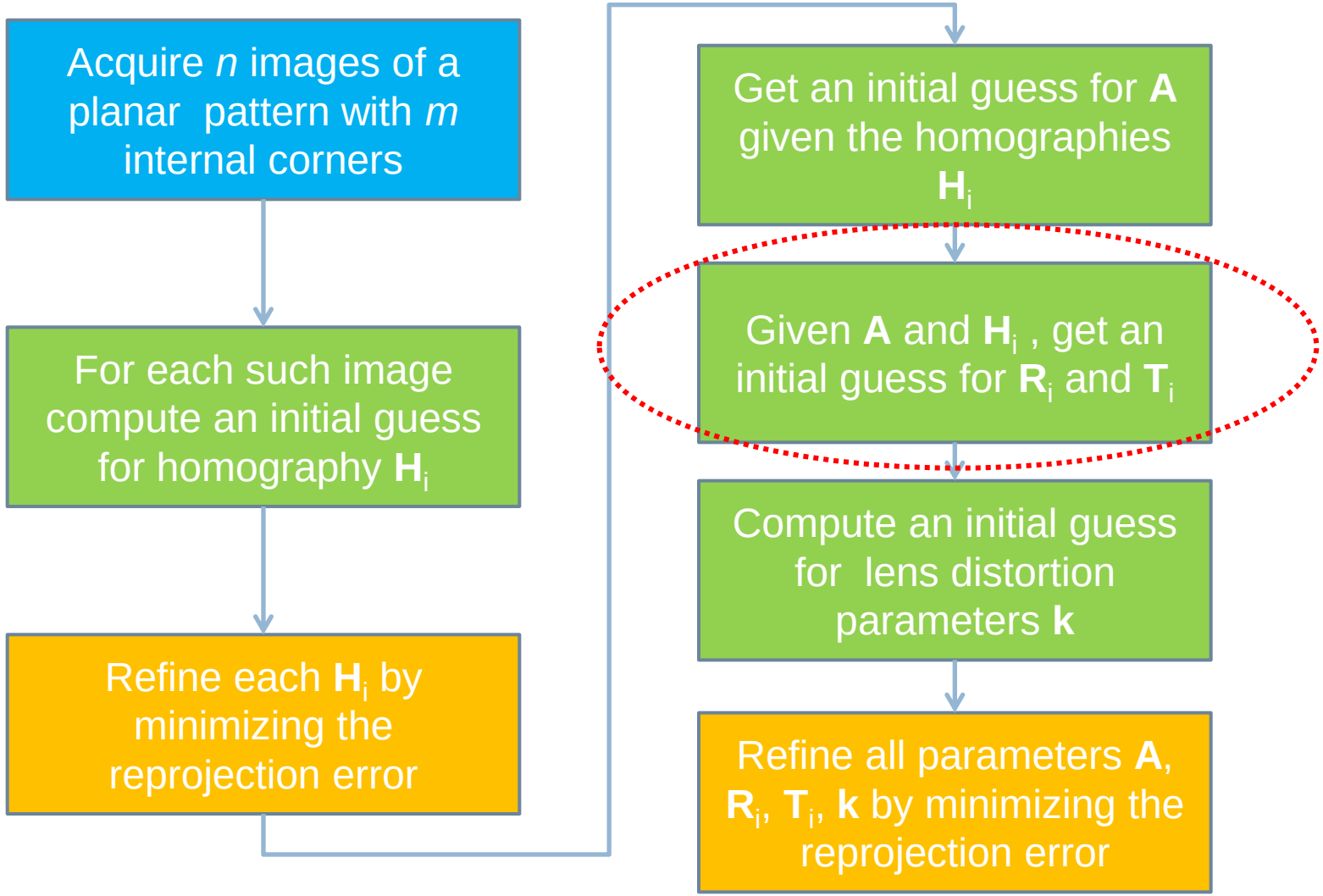
$$\mathbf{T} = \frac{1}{\lambda} \mathbf{A}^{-1} \mathbf{h}_3$$

$$R = [r_1 \; r_2 \; r_3]$$

$R^* = V \tilde{I} V^T$ using SVD. where:

$$\Sigma = \begin{bmatrix} r_1 & & 0 \\ & r_2 & \\ 0 & & r_3 \end{bmatrix}$$

Zhang's Method



Lens distortion parameters (1)

- Given the homographies, we have both the real (distorted) coordinates of the corner features found in the images as well as the ideal (undistorted) coordinates predicted by the homographies. Zhang's method deploys such information to estimate parameters k_1,k_2 of the radial distortion function.

- Given the already known intrinsic parameter matrix, **A**, and the lens distortion model reported below:

distorted coordinate — undistorted — however, we don't know undistorted c. — the unknown

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = L(r)\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \left(1 + k_1 r^2 + k_2 r^4\right)\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix}$$

we need first to establish the relationship between distorted (u', v') and ideal $(\tilde{u}, \tilde{v})$ **pixel** coordinates:

there's no way to obtain the real undistorted coordinate

focal length in horizontal — focal length in vertical

$$\begin{cases} u = \alpha_u \dfrac{x}{z} + u_0 \\ v = \alpha_v \dfrac{y}{z} + v_0 \end{cases} \rightarrow \begin{cases} \tilde{u} = \alpha_u \tilde{x} + u_0 \\ \tilde{v} = \alpha_v \tilde{y} + v_0 \end{cases} \rightarrow \begin{cases} \tilde{x} = \dfrac{\tilde{u} - u_0}{\alpha_u} \\ \tilde{y} = \dfrac{\tilde{v} - v_0}{\alpha_v} \end{cases} \quad \begin{cases} x' = \dfrac{u' - u_0}{\alpha_u} \\ y' = \dfrac{v' - v_0}{\alpha_v} \end{cases}$$

Lens distortion parameters (2)

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \left(1 + k_1 r^2 + k_2 r^4\right) \begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} \rightarrow \begin{cases} \dfrac{u' - u_0}{\alpha_u} = \left(1 + k_1 r^2 + k_2 r^4\right)\left(\dfrac{\tilde{u} - u_0}{\alpha_u}\right) \\ \dfrac{v' - v_0}{\alpha_v} = \left(1 + k_1 r^2 + k_2 r^4\right)\left(\dfrac{\tilde{v} - v_0}{\alpha_v}\right) \end{cases}$$

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

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

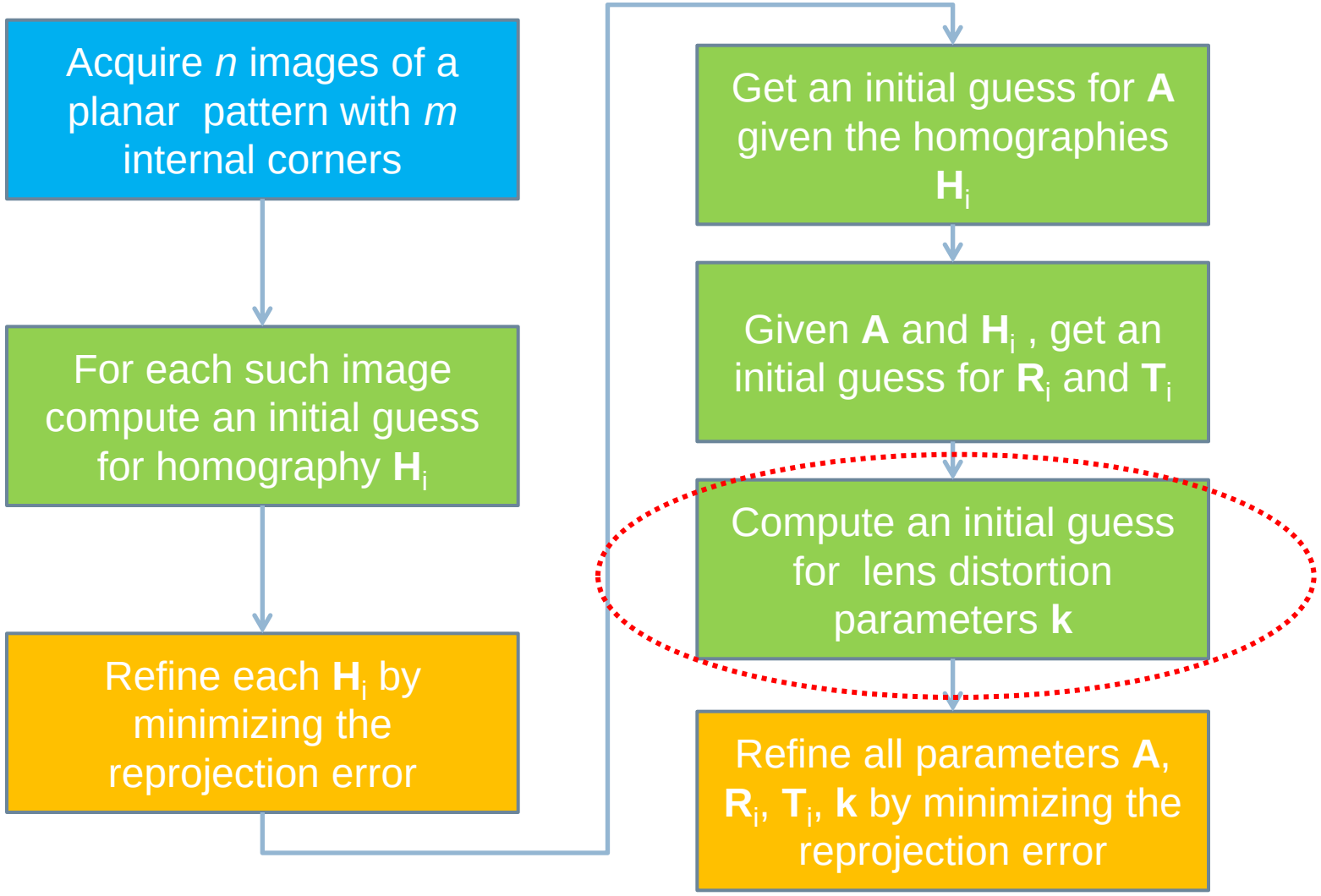
$$\begin{bmatrix} (\tilde{u} - u_0) r^2 & (\tilde{u} - u_0) r^4 \\ (\tilde{v} - v_0) r^2 & (\tilde{v} - v_0) r^4 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} u' - \tilde{u} \\ v' - \tilde{v} \end{bmatrix}$$

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

- With *m* corner features in *n* images, we get a linear system with *2mn* equations in *2* unknowns, which admits a least-squares solution:

$$\mathbf{D}\mathbf{k} = \mathbf{d} \rightarrow \mathbf{k} = \mathbf{D}^{+}\mathbf{d} = \left(\mathbf{D}^{\mathrm{T}}\mathbf{D}\right)^{-1}\mathbf{D}^{\mathrm{T}}\mathbf{d} \qquad \mathbf{D}^{+} = \mathbf{V}\boldsymbol{\Sigma}^{+}\mathbf{U}^{T}$$

Zhang's Method



Refinement by non-linear optimization

- Again, the procedure highlighted so far seeks to minimize an algebraic error rather than the actual reprojection error.
- A more accurate solution can instead be found by a so called Maximum Likelihood Estimate (MLE) aimed at minimization of the reprojection (i.e. geometric) error.
- Under the hypothesis of *i.i.d*. (independent identically distributed) noise, the MLE for our models is obtained by minimization of the error

all corners of an image

all images

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

it allows slight error

with respect to all the unknown parameters.

- The solution of the above non-linear optimization problem is again provided by the Levenberg-Marquardt algorithm.

Some important additional considerations regarding PPMs are reported in Appendix 2.

Calibration of a stereo camera (1)

to determine the distance of the obj. from the camera

- Given a stereo system (aka *rig* or *head*), we are able to calibrate separately the two cameras. This would provide us with:
  - Intrinsic parameters and lens distortion coefficients
  - Extrinsic parameters, i.e. the rigid motion between the CRF and a given WRF (e.g. associated with the calibration target).
- However, calibrating a stereo system requires determining also the rigid motion, $\mathbf{R}$ and $\mathbf{T}$, between the two cameras (i.e. between CRF_L and CRF_R)

we need to know the correlation between two cams

- Knowing the extrinsic parameters wrt the same WRF would allow to compute the rigid motion between the two cameras by simply chaining the transformations: $G_i(G_j)^{-1}$, with either (i=L, j=R) or (i=R, j=L). With Zhang's method this can be achieved, e.g., by taking the same static pattern position as the first image in both calibration sequences.
- However, this is not a robust solution, especially wrt noise and can easily yield to quite inaccurate results.

Calibration of a stereo camera (2)

- A more accurate solution is achieved by starting from an initial guess for $\mathbf{R}$ and $\mathbf{T}$ and then refining the estimation by non-linear minimization of the reprojection error.
- To obtain a robust initial guess, the same position of the planar calibration target is seen in each image of both sequences. This allows to estimate the rigid motion wrt to each such pattern positions as $G_i(G_j)^{-1}$.
- The initial guess is then taken as the median over the values $\mathbf{R}_i$ and $\mathbf{T}_i$ computed as $G_i(G_j)^{-1}$ from each pair of corresponding calibration images. Purposely, each $\mathbf{R}_i$ is converted into a 3 elements vector (rotation angles).
- Finally, the estimation is refined by non-linear minimization of the reprojection error in **both images of each pair** by Levenberg-Marquardt's:

$$\sum_{k=L,R}\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|\mathbf{m}_{i,j}^{k}-\tilde{\mathbf{m}}\left(\mathbf{A}_L,\mathbf{A}_R,\mathbf{K}_L,\mathbf{K}_R,\mathbf{R},\mathbf{T},\mathbf{w}_j\right)\right\|^2$$

Stereo Reference Frame (SRF)

Sensing the depth (or distance in the world) becomes possible

- Once the stereo rig has been calibrated, it is often useful to define a RF attached to the system, which can be thought of as the "WRF" of the Stereo Camera and will be denoted hereinafter as **SRF** (Stereo Reference Frame).
- The standard choice is just to pick-up the CRF of one of the two cameras, e.g. the **left camera**. Accordingly, the extrinsic parameters of the right camera in the SRF are given by the previously computed **R** and **T**, with the corresponding PPMs given by:

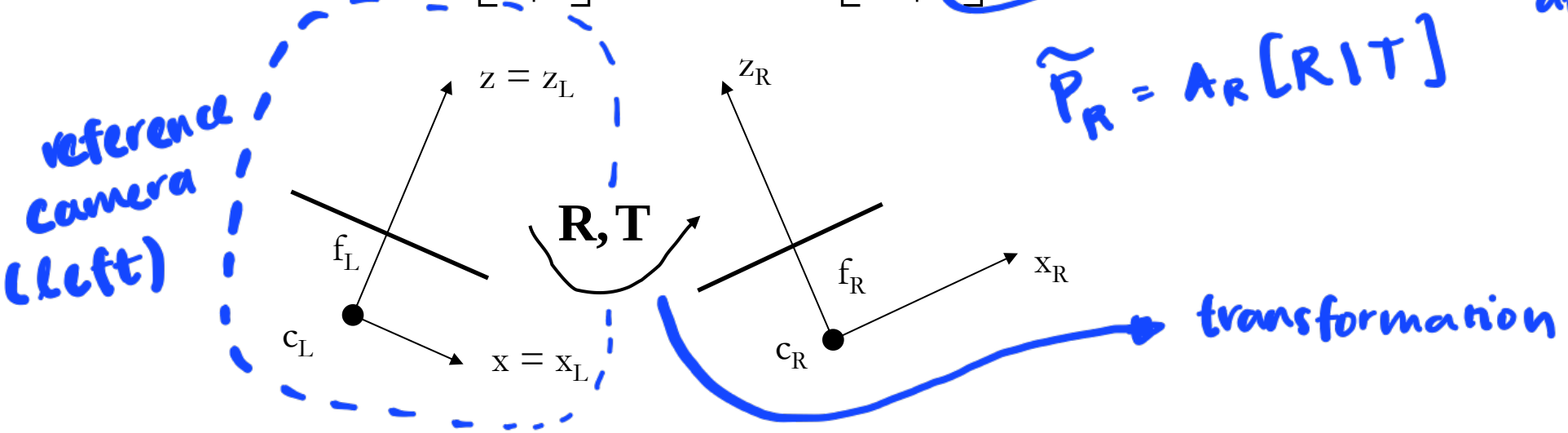
the master

considered as the world

exactly the transformation

$$\tilde{\mathbf{P}}_L = \mathbf{A}_L \left[\mathbf{I}|\mathbf{0}\right] \qquad \tilde{\mathbf{P}}_R = \mathbf{A}_R \left[\mathbf{R}|\mathbf{T}\right]$$

$\tilde{P}_R = A_R [R|T]$



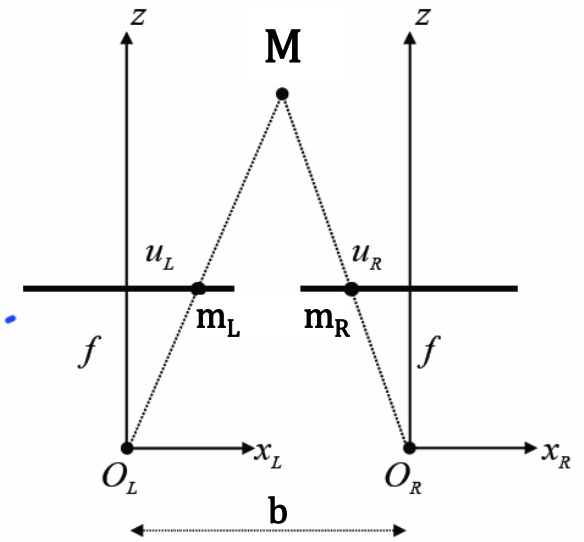
Rectification

- It is well known that a standard (i.e. lateral) stereo geometry is the most convenient choice to search for corresponding pixels:

$$(1):\ M_L = M_R + \begin{bmatrix} b \\ 0 \\ 0 \end{bmatrix}, (2):\ f_L = f_R = f \rightarrow A_L = A_R = A$$

must be identical

because there's pixelization

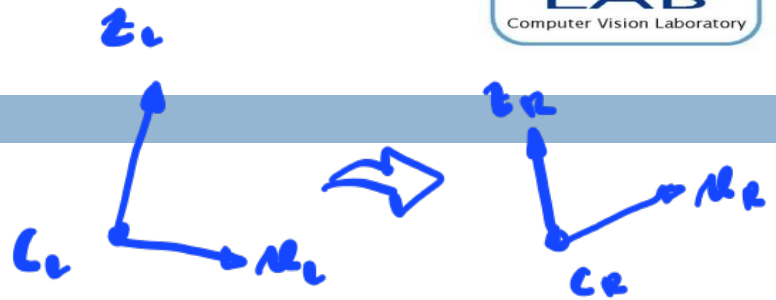


- Unfortunately, it is impossible to achieve such geometry by mechanical alignment.

We could virtually rotate along the camera axis

- However, once calibration has been performed, one can define two new cameras (i.e. two new PPMs) in standard geometry by virtually rotating the original ones about their optical centres (1) and constraining the intrinsic parameters of the new cameras to be the same (2).
- Then, the images acquired by each of the two original cameras can be **rectified**, i.e. transformed into those that would have been acquired by the new ones, by a **homography**.
- Rectification needs to happen after compensation of lens distortion (i.e. on undistorted images). An initial transformation to remove lens distortion is thus always applied before transforming again the images according to the **rectification homographies**.

The new PPMs



- Given $\mathbf{A}_L$, $\mathbf{A}_R$, $\mathbf{R}$, $\mathbf{T}$ the new PPMs can be achieved as follows:

– A new intrinsic parameter matrix, $\mathbf{A}_{\mathbf{new}}$, is arbitrarily chosen (e.g. the mean between $\mathbf{A}_L$, $\mathbf{A}_R$).

– The new rotation matrix for both cameras, $\mathbf{R}_{\mathbf{new}}$, is then determined. As the row vectors in $\mathbf{R}_{\mathbf{new}}$ represent the X, Y, e Z axis of the new CRFs into the WRF, taking as WRF the previously introduced SRF, a lateral stereo system is achieved as follows.

– The new X axis is taken parallel to the baseline vector $\mathbf{B}=\mathbf{C}_R-\mathbf{C}_L$. As we wish to express $\mathbf{B}$ into the SRF: $\mathbf{B}=-\mathbf{R}^T\mathbf{T}-\mathbf{0}=-\mathbf{R}^T\mathbf{T}=[B_x\ B_y\ B_z]^T$. Then, the first row of $\mathbf{R}_{\mathbf{new}}$, is given by the unit vector parallel to the baseline vector: $\mathbf{r_1}=\frac{\boldsymbol{B}}{\|\boldsymbol{B}\|}$

– The new Y axis is orthogonal to X and to arbitrary unit vector $\mathbf{k}$, e.g. chosen to be parallel to the old Z axis of the left camera ($\mathbf{k}=[\mathbf{0}\ \mathbf{0}\ \mathbf{1}]^T$):

$$\mathbf{r_2}=k\wedge\mathbf{r_1}=\frac{[-B_y\ B_x\ 0]^T}{\sqrt{B_x^2+B_y^2}}$$

– The new Z axis is, of course, orthogonal to X and Y: $\mathbf{r_3}=\mathbf{r_1}\wedge\mathbf{r_2}$

– Thus, the new PPMs are: $\widetilde{\mathrm{P}}'_L=\mathrm{A_{new}}[\mathrm{R_{new}}|\ 0]$, $\widetilde{\mathrm{P}}'_R=\mathrm{A_{new}}[\mathrm{R_{new}}|-\mathrm{R_{new}C_R}]$

Rectification Homographies

- Both cameras undergo a rotation about the optical center and a change of intrinsics. We have already studied that, in such a case, the images (i.e. the original and rectified ones) are related by a homography and thus we know how to compute these transformations.
- Considering the left camera first:

$$\widetilde{m}_L = \mathbf{A}_L[\mathbf{I}\ \mathbf{0}]\widetilde{\mathbf{M}}$$
$$\widetilde{\mathbf{m}}'_L = \mathbf{A}_{new}[\mathbf{R}_{new}\ \mathbf{0}]\widetilde{\mathbf{M}}$$

$$\widetilde{m}_L = \mathbf{A}_L\mathbf{R}_{new}^{-1}\,\mathbf{A}_{new}^{-1}\,\widetilde{\mathbf{m}}'_L \rightarrow \mathbf{H}_L$$

Rectification Homography for the left image

- To compute the other rectification homography, we may conveniently think of moving the origin of the WRF into the optical centre of the right camera:

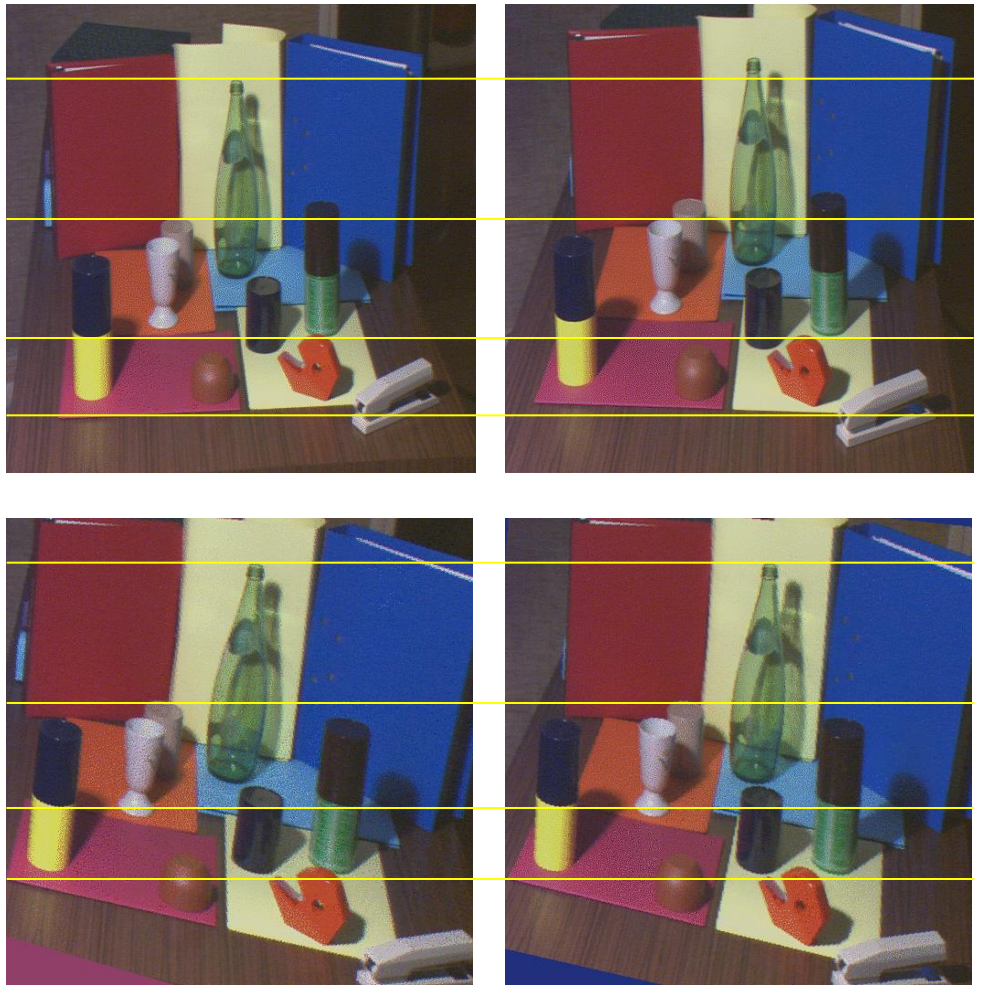
$$\widetilde{m}_R = \mathbf{A}_R[\mathbf{R}\ \mathbf{0}]\widetilde{\mathbf{M}}_R$$
$$\widetilde{\mathbf{m}}'_R = \mathbf{A}_{new}[\mathbf{R}_{new}\ \mathbf{0}]\widetilde{\mathbf{M}}_R$$

$$\widetilde{m}_R = \mathbf{A}_R\mathbf{R}\mathbf{R}_{new}^{-1}\,\mathbf{A}_{new}^{-1}\,\widetilde{\mathbf{m}}'_L \rightarrow \mathbf{H}_R$$

Rectification Homography for the right image

A different derivation of the rectification homographies can be found in Appendix 3.

An example



From pixels to 3D coordinates

emits light so the reflection is measured in time of flight

obtained from depth images

If the camera is calibrated, i.e. matrix **A** is known, and depth is also known, as it is the case, e.g., for the reference image of a stereo camera or a TOF or Structured Light (e.g. Kinect) sensor, we can estimate the **3D coordinates of pixels in the CRF** (Camera Reference Frame) in order to obtain a so called ***point cloud*** , a representation widely used in **3D Computer Vision**.

$$\mathbf{p}^* = \mathbf{A}\,\mathbf{P} \qquad \mathbf{P} = \mathbf{A}^{-1}\,\mathbf{p}^* \qquad \mathbf{P} = z\,\mathbf{A}^{-1}\,\frac{\mathbf{p}^*}{z}$$

$$\mathbf{P} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \mathbf{p}^* \qquad \mathbf{A}$$

$$\frac{\mathbf{p}^*}{z} = \begin{bmatrix} \alpha_u \frac{x}{z} + u_0 \\ \alpha_v \frac{y}{z} + v_0 \\ 1 \end{bmatrix} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{p}$$

3D coordinates in the CRF

depth

inverse of A

pixel coordinates

Thus: $\mathbf{P} = z\,\mathbf{A}^{-1}\,\mathbf{p}$

with:

$$A^{-1} = \begin{bmatrix} \frac{1}{\alpha_u} & 0 & -\frac{u_0}{\alpha_u} \\ 0 & \frac{1}{\alpha_v} & -\frac{v_0}{\alpha_v} \\ 0 & 0 & 1 \end{bmatrix}$$

From pixels to 3D coordinates

If the camera is calibrated, i.e. matrix **A** is known, and depth is also known, as it is the case, e.g., for the reference image of a stereo camera or a TOF or Structured Light (e.g. Kinect) sensor, we can estimate the **3D coordinates of pixels in the CRF** (Camera Reference Frame) in order to obtain a so called ***point cloud*** , a representation widely used in **3D Computer Vision**.

$$\begin{bmatrix} \alpha_u x + u_0 z \\ \alpha_v y + v_0 z \\ z \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$\mathbf{p}^*$ $\quad$ $\mathbf{A}$ $\quad$ $\mathbf{P}$

$$\mathbf{p}^* = \mathbf{A}\,\mathbf{P}$$

$$\mathbf{P} = \mathbf{A}^{-1}\,\mathbf{p}^*$$

$$\mathbf{P} = z\,\mathbf{A}^{-1}\,\frac{\mathbf{p}^*}{z}$$

$$\frac{\mathbf{p}^*}{z} = \begin{bmatrix} \alpha_u \frac{x}{z} + u_0 \\ \alpha_v \frac{y}{z} + v_0 \\ 1 \end{bmatrix} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{p}$$

3D coordinates in the CRF

depth

inverse of A

pixel coordinates

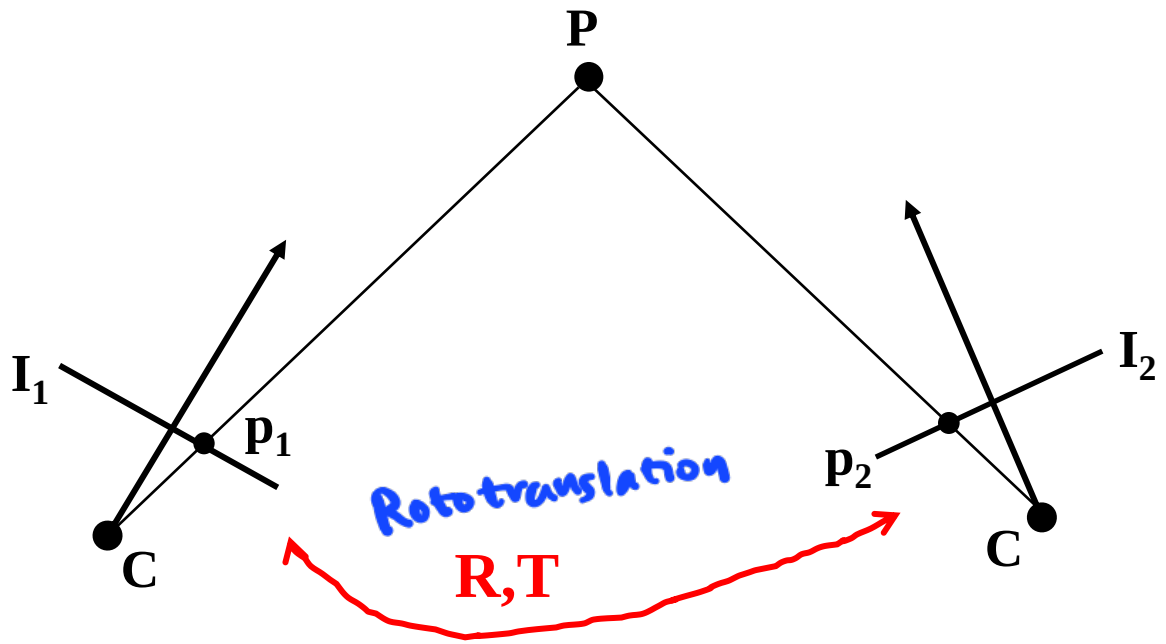
Thus: $\mathbf{P} = z\,\mathbf{A}^{-1}\,\mathbf{p}$

with:

$$A^{-1} = \begin{bmatrix} \frac{1}{\alpha_u} & 0 & -\frac{u_0}{\alpha_u} \\ 0 & \frac{1}{\alpha_v} & -\frac{v_0}{\alpha_v} \\ 0 & 0 & 1 \end{bmatrix}$$

..and back to pixels

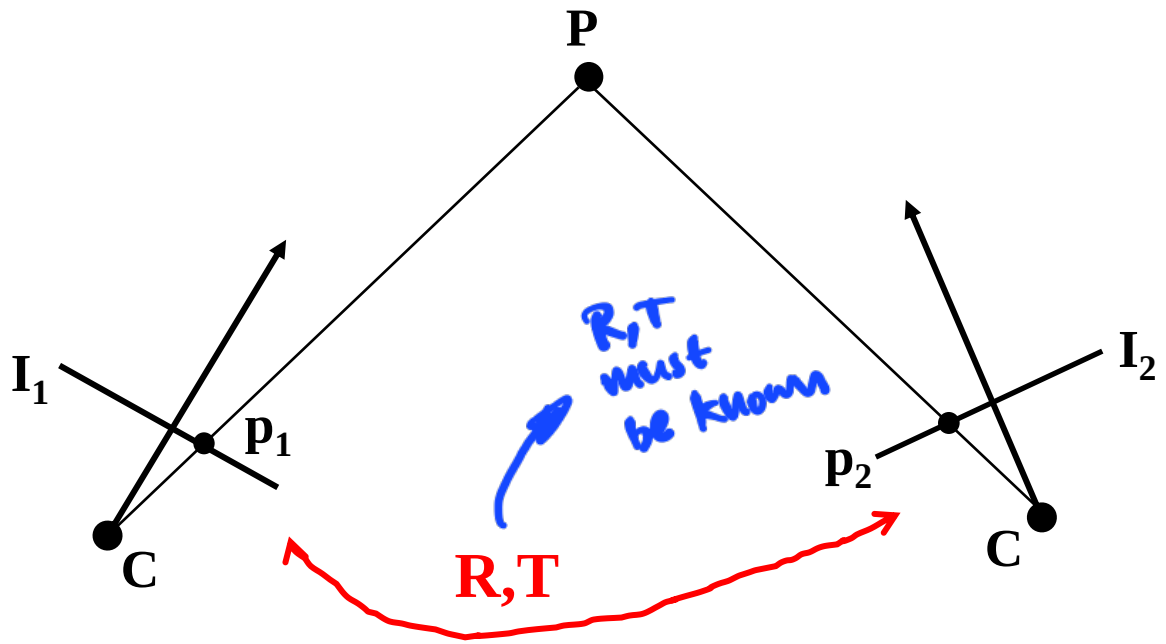
We may wish to find where a certain pixels does project into another image taken by the same camera, typically a moving camera used to scan a static scene. Purposely, alongside depth and camera intrinsics, we need to know the rigid motion (roto-translation) between the two views (i.e. CRFs).



$zA^{-1}p_1$ with: $T_{1\to2}(P_1)=RP_1+T$

..and back to pixels

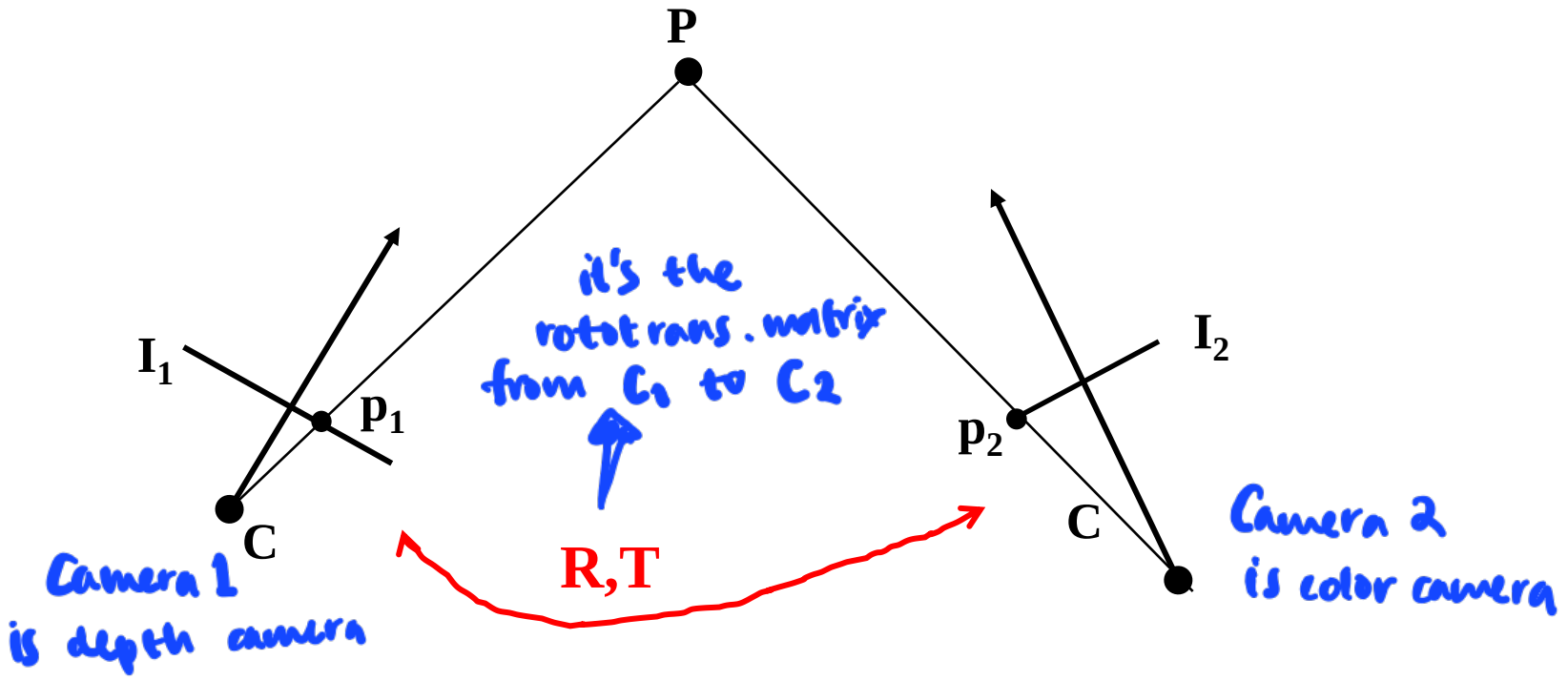
We may wish to find where a certain pixels does project into another image taken by the same camera, typically a moving camera used to scan a static scene. Purposely, alongside depth and camera intrinsics, we need to know the rigid motion (roto-translation) between the two views (i.e. CRFs).



$p_2 = \mathrm{A}\, \mathrm{T}_{1\to 2}(zA^{-1}p_1)$ with: $\mathrm{T}_{1\to 2}(\mathrm{P}_1)=\mathrm{RP}_1+\mathrm{T}$

Cameras may be different

Similarly, the other image may be taken by a different camera. We would then need to know the intrinsic parameters of the other camera too.



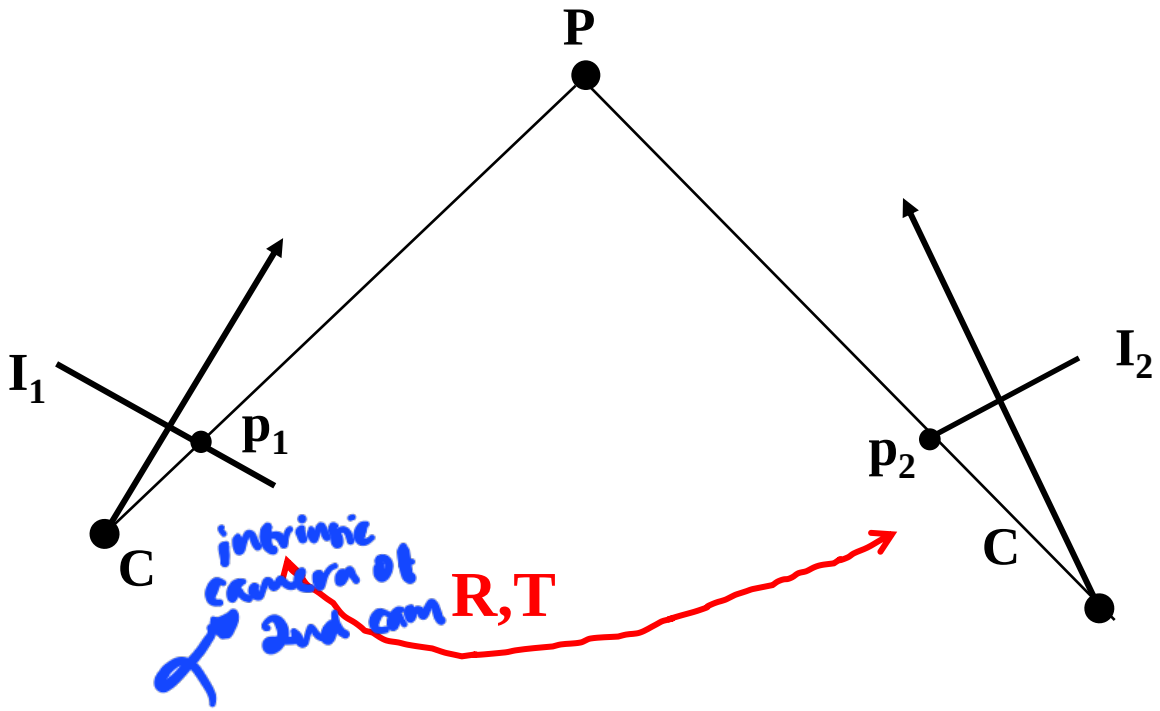
$T_{1\to2}(zA_1^{-1}p_1)$ with: $T_{1\to2}(P_1)=RP_1+T$

Cameras may be different

Similarly, the other image may be taken by a different camera. We would then need to know the intrinsic parameters of the other camera too.

Basler – Blaze ToF + RGB

Microsoft Azure Kinect



$$p_2 = A_2\, T_{1\to 2}(zA_1^{-1}p_1) \quad \text{with:} \quad T_{1\to 2}(P_1)=RP_1+T$$

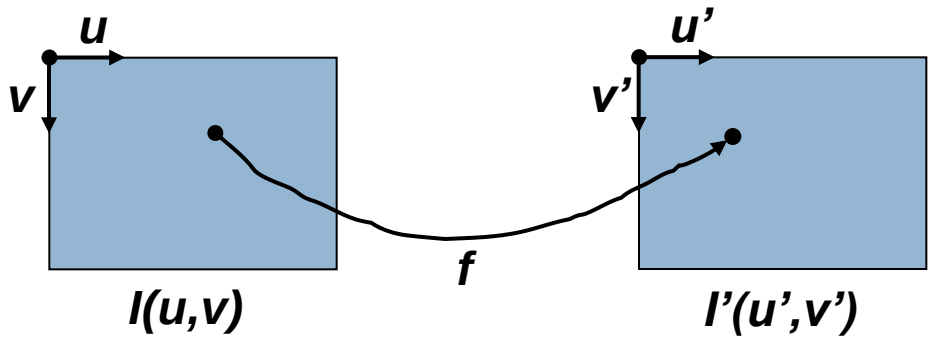
Luxonis – OAK-D-PoE

Lucid – Helios2 & Triton

This is how an RGB-D image is obtained in many *depth cameras* (e.g. MS Kinect).

Image Warping

transformation



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

$$I'\left(f_u(u,v), f_v(u,v)\right) = I(u,v)$$

copy the pixel coordinate and put it on a new coord. system

Examples of image warping:

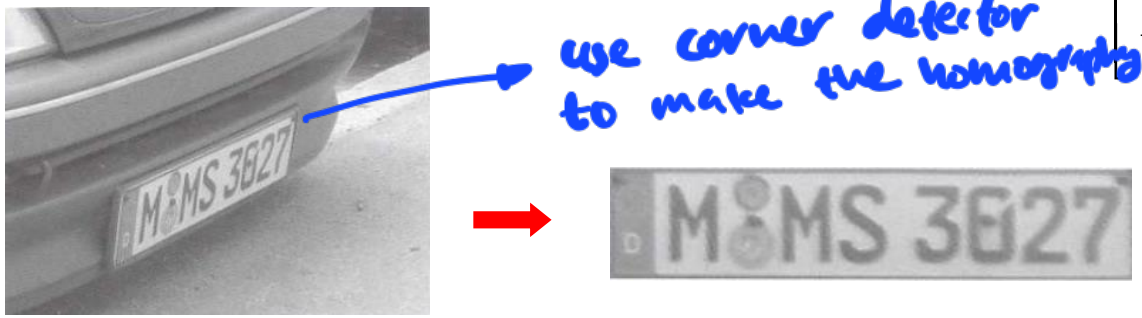
Rotation

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

Homography applies when two cameras see the same plane

Removal of perspective deformation

$$s\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{32} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



use corner detector to make the homography

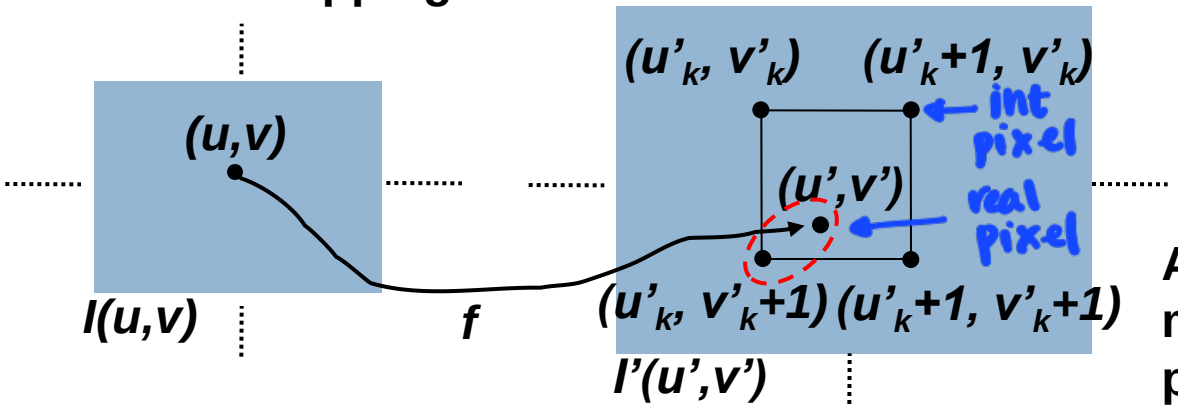
need correspondence to map pixels between 2 images

Other examples:

- ***Remove lens distortion***
- ***Stereo rectification***
- .....

Forward/Backword Mapping

- **Forward Mapping** (sub-optimal)



$$(u'_k, v'_k) = \left(\lfloor u' \rfloor, \lfloor v' \rfloor\right)$$

A better choice consists in mapping to the closest point into the destination image

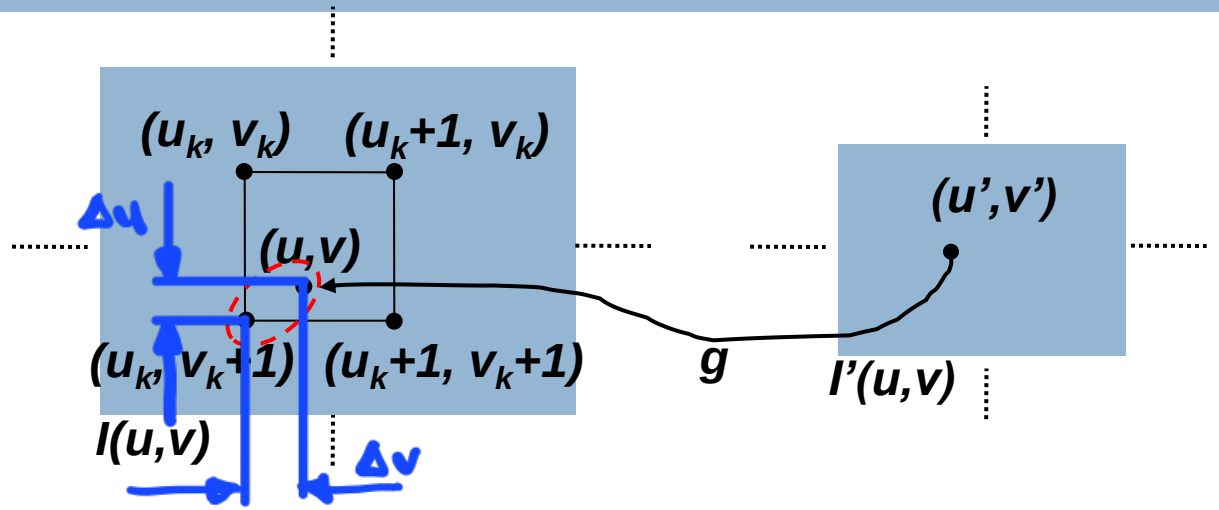
***Holes* and *folds* in I'(u',v') !** → because assigned pixels are conflicting.

- **Backward Mapping**

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

No *holes* and *folds* in I'(u',v'), which mapping strategy ?

Mapping Strategies



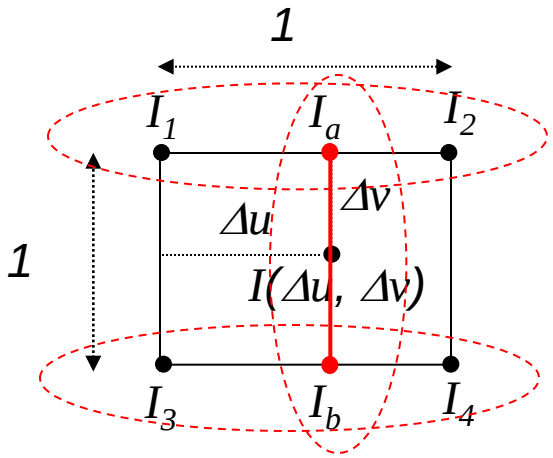
$$(u_k, v_k) = (\lfloor u \rfloor, \lfloor v \rfloor)$$

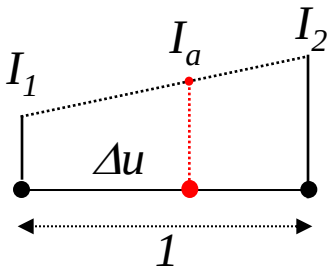
- **Mapping from the closest point (Nearest Neighbour Mapping)**
- **Interpolation between the 4 closest points (*bilinear*, bicubic,…)**

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

Bilinear Interpolation (1)

we interpolate two times : horizontally & vertically





$$\frac{I_a - I_1}{\Delta u} = I_2 - I_1$$

$$I_a = (I_2 - I_1)\Delta u + I_1$$

1

$$I_b = (I_4 - I_3)\Delta u + I_3$$

$$I(\Delta u, \Delta v) = (I_b - I_a)\Delta v + I_a$$

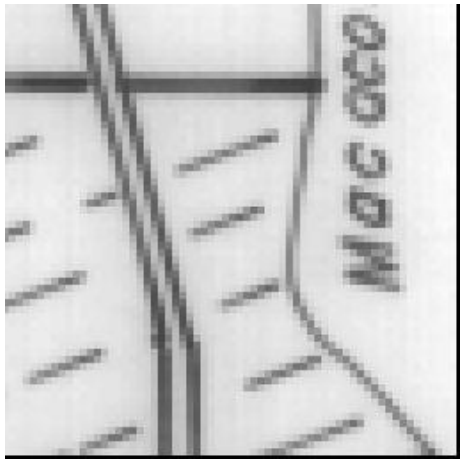
$$I(\Delta u, \Delta v) = (((I_4 - I_3)\Delta u + I_3 - ((I_2 - I_1)\Delta u + I_1))\Delta v + (I_2 - I_1)\Delta u + I_1$$

<< 0

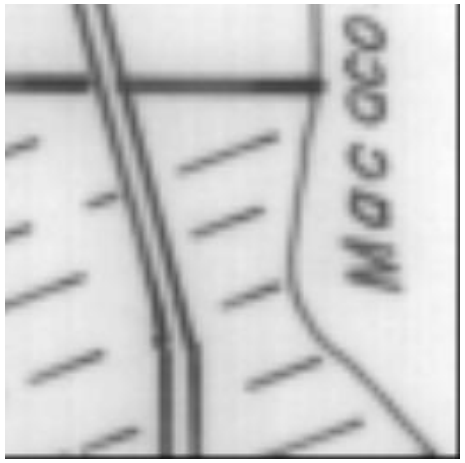
$$I'(u', v') = (1 - \Delta u)(1 - \Delta v)I_1 + \Delta u(1 - \Delta v)I_2 + (1 - \Delta u)\Delta v I_3 + \Delta u \Delta v I_4$$

Bilinear Interpolation (2)

Input

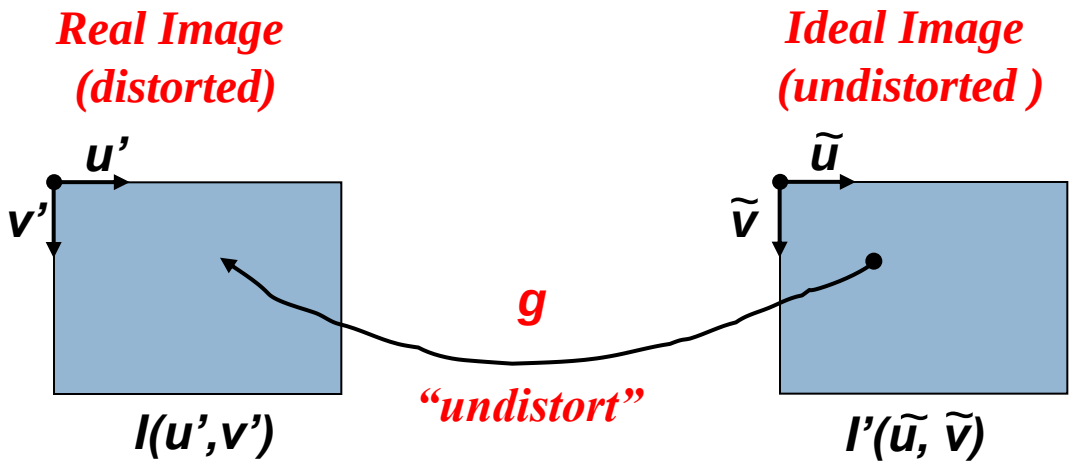


Warp (Zoom) by NNM

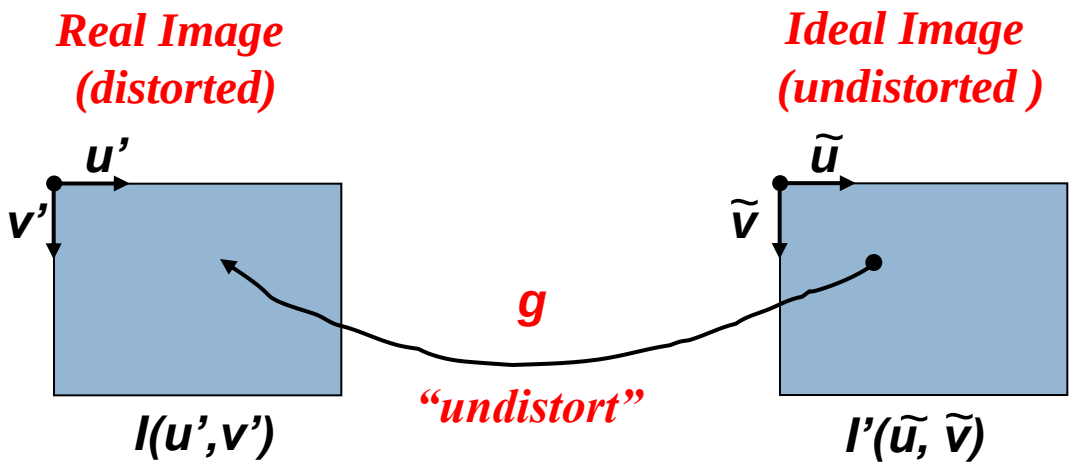


Warp (Zoom) by Bilinear Interpolation

Warping to compensate lens distortion (1)



Warping to compensate lens distortion (2)



it's a mapping from distorted to undistorted image

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:

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

For example, using Zhang's calibration method

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

Appendix 1 - DLT Algorithm: 4 points case (1)

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

$$\mathbf{Ah} = 0, \quad \mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \ldots \mathbf{A}_9 \end{bmatrix}, \quad \mathbf{h} = \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_9 \end{bmatrix}$$

where **A** is a 8x9 matrix, $\mathbf{A_1 \ldots A_9}$ are 8x1 vectors, **h** is a 9x1 vector, $\mathbf{h_1, h_2, h_3}$ are 3x1 vectors representing the rows of the 3x3 matrix **H** defining the homography, $h_1 \ldots h_9$ are the 9 elements of such matrix.

As **H** is defined up to a scale factor, we can set $h_9 = 1$, so as to obtain a non homogeneous linear system with 8 equations and 8 unknowns

$$\tilde{\mathbf{A}}\tilde{\mathbf{h}} = \mathbf{b}, \quad \tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \ldots \mathbf{A}_8 \end{bmatrix}, \quad \tilde{\mathbf{h}} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_8 \end{bmatrix}, \quad \mathbf{b} = -\mathbf{A}_9$$

which can be solved easily by standard methods (Cramer's rule, Inversion of the coefficient matrix, *Gaussian Elimination*)

Appendix 1 - DLT Algorithm: 4 points case (2)

- Alternatively, it is possible to constrain the nom of **h**, e.g. so as to render it equal to 1. Purposely, we can assume again h_9 as fixed, but no longer equal to one :

$$\tilde{\mathbf{A}}\tilde{\mathbf{h}} = \mathbf{b}, \quad \tilde{\mathbf{A}} = \left[\mathbf{A}_1 \ \mathbf{A}_2 ... \mathbf{A}_8\right], \ \tilde{\mathbf{h}} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_8 \end{bmatrix}, \ \mathbf{b} = -h_9\mathbf{A}_9$$

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

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

- According to this method, the coefficient matrix of the linear system needs to be inverted and then, using the formulas shown above it is possible to compute h_9 first and then **h**

Appendix 2 - Decomposition of the PPM

- Zhang's method provides separately $\mathbf{A}$, $\mathbf{R}$ e $\mathbf{T}$ **.** Other methods, though, compute directly the PPM (see, e.g. Hartley&Zisserman, Chapter 7).
- It is however always possible to decompose any given PPM into its elementary components. Purposely, we can deploy the so called **QR factorization,** so as to factorize a real square matrix into the product between an ***orthogonal*** and ***upper triangular*** matrices.
- Let's then $\mathbf{P}^{-1} = \mathbf{UL}$ be the factorization of the inverse of the square matrix $\mathbf{P}$, with $\mathbf{U}$ orthogonal and $\mathbf{L}$ upper triangular.
- Hence, to obtain the rotation matrix and the intrinsic parameter matrix:

$$\tilde{\mathbf{P}} = \left[\mathbf{P}\middle|\mathbf{p}_4\right] = \left[\mathbf{AR}\middle|\mathbf{AT}\right] \Rightarrow \mathbf{P}^{-1} = \mathbf{R}^{-1}\mathbf{A}^{-1} \Rightarrow$$

$$\mathbf{R} = \mathbf{U}^{-1}, \mathbf{A} = \mathbf{L}^{-1}$$

- Finally, the translation vector **T** is given by:

$$\mathbf{T} = \mathbf{A}^{-1}\mathbf{p}_4 = \mathbf{L}\mathbf{p}_4$$

Appendix 2 - Optical centre from the PPM

- It can be shown that a PPM is always a full rank (i.e. rank 3) matrix, and, alike, that every full-rank 3x4 matrix defines a perspective projection.
  - Intuitively: full rank is mandatory, as otherwise the projection of a 3D point will not be an image point but a higher dimensional subspace (such as a line or plane)
  - Recalling that (0,0,0) does not represent any valid point in homogeneous coordinates, we can observe that a point in space belonging to the matrix null space (or kernel, i.e. the solutions of the associated homogenous system) must be a point whose projection is undefined.

- Such a point is the optical centre, which is indeed the only point in space whose perspective projection onto the image plane is undefined (one cannot define a projection ray from the optical centre and itself). We can therefore write :

$$\tilde{\mathbf{P}}\begin{bmatrix}\mathbf{c}\\1\end{bmatrix}=\mathbf{0} \text{ and thus } \mathbf{c}=-\mathbf{P}^{-1}\mathbf{p}_4 \text{ with } \tilde{\mathbf{P}}=\left[\mathbf{P}\middle|\mathbf{p}_4\right]$$

3D coordinates of the optical center in the WRF

Appendix 2 - Optical ray from the PPM

- The optical ray of an image point, $\widetilde{\mathbf{m}}$, is the 3D line between $\widetilde{\mathbf{m}}$ and the optical centre, , $\tilde{\mathbf{C}} = \begin{bmatrix} \mathbf{C} \\ \mathbf{1} \end{bmatrix}$.

  Indeed, the optical ray is the locus of points, $\widetilde{\mathbf{M}}$, satisfying the equation $k\widetilde{\mathbf{m}} = \widetilde{\mathbf{P}}\widetilde{\mathbf{M}}$ for a given $\widetilde{\mathbf{m}}$.

  int is the optical centre, which is indeed the only point in space whose perspective projection onto the image plane is undefined (one cannot define a projection ray from the optical centre and itself)

- The equation of the optical ray in Euclidean coordinates can be determined as follows:

$$\mathbf{M} = \mathbf{C} + \lambda \mathbf{V}$$

optical centre

any vector parallel to the optical ray

$$\widetilde{M_\infty} = \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{V} \\ 0 \end{bmatrix}, \quad \tilde{\mathbf{P}} \begin{bmatrix} \mathbf{P}^{-1}\tilde{\mathbf{m}} \\ 0 \end{bmatrix} = [\mathbf{P} \ \boldsymbol{p_4}] \begin{bmatrix} \mathbf{P}^{-1}\tilde{\mathbf{m}} \\ 0 \end{bmatrix} = \tilde{\mathbf{m}}$$

is a point at infinity AND belongs to the optical ray

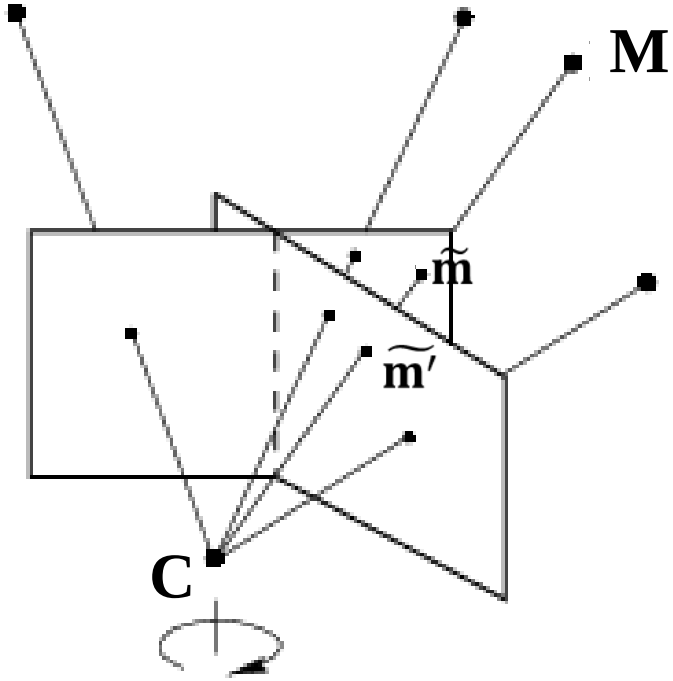
$$\mathbf{V} = \mathbf{P}^{-1}\tilde{\mathbf{m}}$$

$$\mathbf{M} = \mathbf{C} + \lambda \mathbf{P}^{-1}\tilde{\mathbf{m}}$$

$$\widetilde{\mathbf{M}} = \tilde{\mathbf{C}} + \lambda \begin{bmatrix} \mathbf{P}^{-1}\tilde{\mathbf{m}} \\ 0 \end{bmatrix}$$ (in projective coordinates)

Appendix 3 - Rectification Homographies by optical rays (Appendix 2)

- We must compute the transformations that map the original images, i.e. those acquired by the cameras with PPMs, $\widetilde{\mathbf{P}_L}$ and $\widetilde{\mathbf{P}_R}$, into a rectified pair, i.e. the two images that would have been acquired by the cameras with PPMs $\widetilde{\mathbf{P}'_L}$ and $\widetilde{\mathbf{P}'_R}$.
- Given a 3D point in the WRF (i.e. the SRF), for each of the two cameras we can write the equation of the optical ray through the point based on both $\widetilde{\boldsymbol{P}}$ and $\widetilde{\boldsymbol{P}'}$ (subscripts *L,R* are omitted for the sake of simplicity).



$$\widetilde{\mathbf{P}} = [\boldsymbol{P}\ \boldsymbol{p}_4] \longrightarrow \mathbf{M} = \mathbf{C} + \lambda \mathbf{P}^{-1}\widetilde{\mathbf{m}}$$

$$\widetilde{\mathbf{P}'} = [\mathbf{P}'\boldsymbol{p}'_4] \longrightarrow \mathbf{M} = \mathbf{C} + \lambda \mathbf{P}'^{-1}\widetilde{\mathbf{m}'}$$

$$\mathbf{P}^{-1}\widetilde{\mathbf{m}} = \lambda \mathbf{P}'^{-1}\widetilde{\mathbf{m}'}$$

$$\widetilde{\mathbf{m}} = \lambda \mathbf{P}\mathbf{P}'^{-1}\widetilde{\mathbf{m}'}$$

$$\boldsymbol{H}_L = \mathbf{P}_L\mathbf{P}_L'^{-1}, \quad \boldsymbol{H}_R = \mathbf{P}_R\mathbf{P}_R'^{-1}$$

Homography

Main References

1) R. Hartley and A. Zisserman, “Multiple View Geometry in Computer Vision”, 2° Edition, Cambridge University Press, 2003.

2) Z. Zhang, “A flexible new technique for camera calibration”, IEEE Trans. On Pattern Analysis and Machine Intelligence, 22(11):1330-1334, November 2000.

3) A. Fusiello, “Visione Computazionale – Tecniche di ricostruzione tridimensionale”, Collana Informatica Franco Angeli, 2013.

4) Bradski, Adrian Kaehler, “Learning OpenCV - Computer Vision with the OpenCV Library”, O'Reilly Media, 2008.