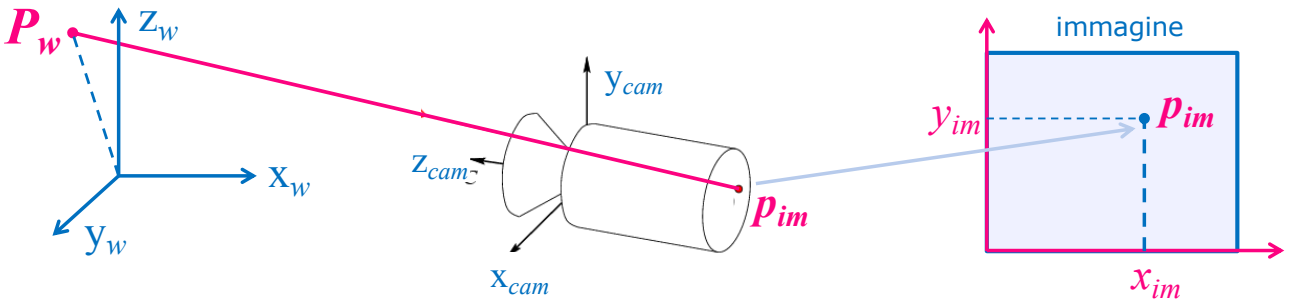
1. Generazione di un'immagine (Image Formation)

Camera Calibration
(calibrazione geometrica della camera)

(Forsyth/Ponce: Capitolo 1)

Camera Calibration:
processo attraverso il quale si determina il modello geometrico di una camera



Camera model **M**(ξ) : $\widetilde{p_{im}} = \mathbf{M} \cdot \widetilde{P_w}$

Calibrazione: determinazione di M (o dei parametri ξ:)

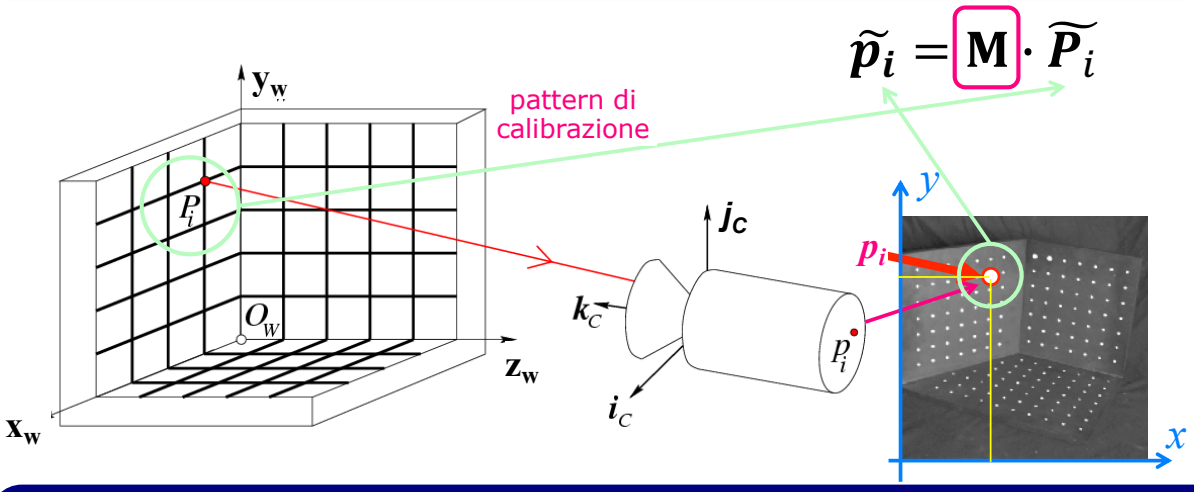
Parametri di camera ξ : $\xi = \left[\mathbf{R},\ \mathbf{T},\ f,\ \mathbf{C}\right] = \left[\varphi, \vartheta, \rho,\ t_X, t_Y, t_Z,\ f,\ x_C, y_C\right]$ (estrinseci | intrinseci)

Calibrazione di camera – **setup di calibrazione**

Setup di calibrazione:

- La camera osserva una scena con un pattern di calibrazione contenente numerosi punti fiduciali
  - ***punti fiduciali (fiducial markers):*** *oggetti localizzabili facilmente e con notevole precisione*
- Le world coordinates dei punti fiduciali sono note a priori, rispetto a una terna solidale con il pattern di calibrazione → P_i note a priori
- I punti fiduciali vengono localizzati nell'immagine → p_i determinate analizzando l'immagine

CALIBRAZIONE: sfrutto l'associazione $P_i \leftrightarrow p_i$ per determinare M

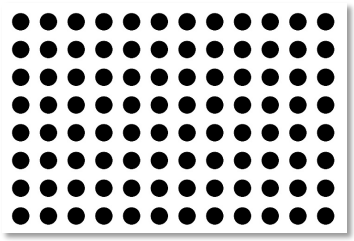


Calibrazione di camera – **pattern di calibrazione**

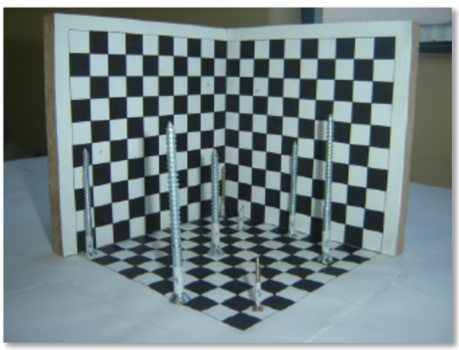
Pattern di calibrazione

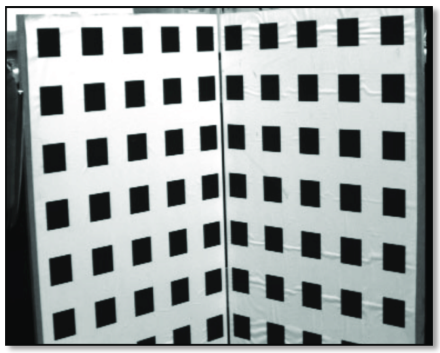
Set di punti fiduciali: semplici da localizzare, con notevole precisione

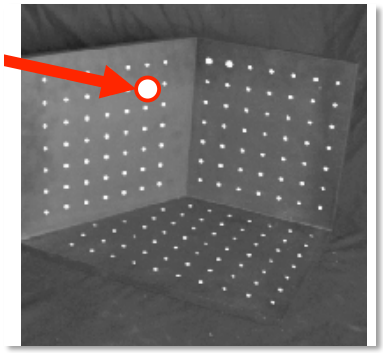
PATTERN	***Da localizzare:***
sferette	*→ centro della sfera*
cerchi piani	*→ centro del cerchio*
"chessboard"	*→ punti fiduciali: vertici dei quadrati*



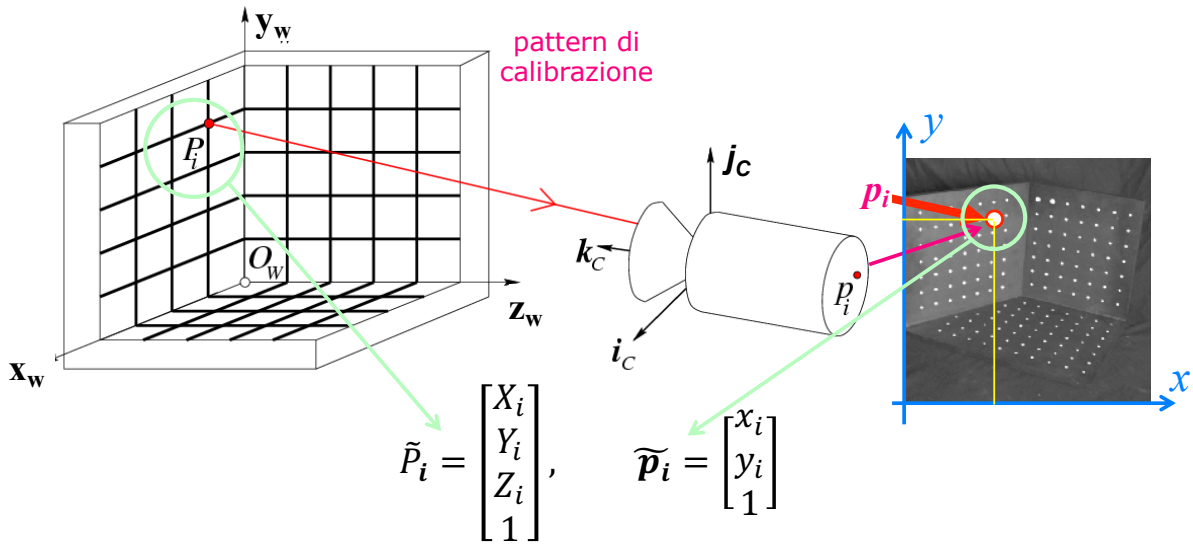
- Il set di punti fiduciali deve 'riempire' uno spazio **3D** non degenere
  - *Punti **su un piano** non bastano!*
  - *Se uso pattern planari, mi servono almeno 2 immagini, su piani differenti*







Calibrazione di camera – definizione del problema



$$\tilde{P}_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}, \qquad \tilde{p}_i = \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Definizione del problema:

- dato un set di ***N*** punti fiduciali P_i (*fiducial points*), di cui è nota la posizione 3D in un sistema di riferimento solidale con la scena…
- …e date le coordinate-immagine p_i di tali N punti, visti dalla camera,

➔ determinare il modello di camera **M** (funzione di ξ) che soddisfa:

$$\tilde{p}_i = \mathbf{M} \cdot \tilde{P}_i , \qquad i = 1..N$$

Calibrazione di camera – approccio lineare

Calibrazione – approccio lineare

Determino la matrice **M** [3 x 4] del modello lineare $\tilde{p}_i = \mathbf{M} \cdot \tilde{\mathbf{P}}_i$ conoscendo:

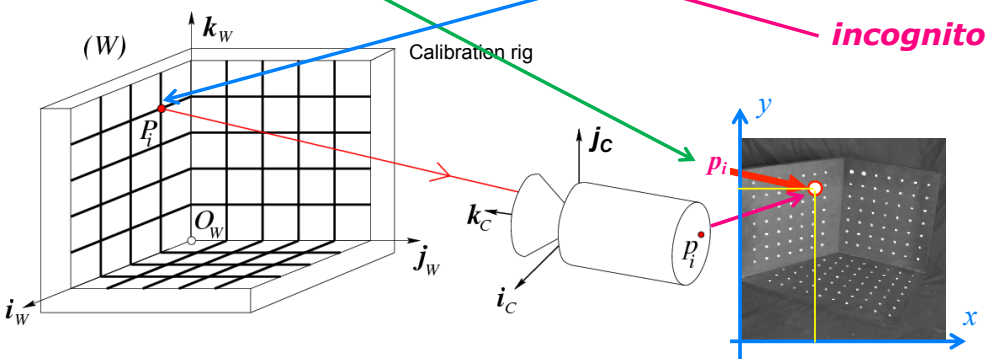
- le coordinate-mondo di *N* punti fiduciali: $\mathbf{P}_i , \; i = 1..N$
- le loro coordinate-immagine: $p_i , \; i = 1..N$

Per ogni i=1..N:

***N* equazioni:**

$$\tilde{p}_i = \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \mathbf{M} \cdot \tilde{\mathbf{P}}_i = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix} \cdot \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} , \qquad i = 1 \cdots N$$

incognito



Calibrazione di camera – approccio lineare

Per ogni punto fiduciale $\mathbf{P}_i$ $(i = 1..N)$, localizzato nell'immagine in $\mathbf{p}_i$:

$$\tilde{\mathbf{p}}_i = \begin{bmatrix} \tilde{x}_i \\ \tilde{y}_i \\ \tilde{z}_i \end{bmatrix} = \mathbf{M} \cdot \tilde{\mathbf{P}}_W = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix} \cdot \tilde{\mathbf{P}}_i = \begin{bmatrix} \mathbf{m}_1 \cdot \tilde{\mathbf{P}}_i \\ \mathbf{m}_2 \cdot \tilde{\mathbf{P}}_i \\ \mathbf{m}_3 \cdot \tilde{\mathbf{P}}_i \end{bmatrix}$$

In coordinate euclidee: $\mathbf{p}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \frac{\tilde{x}_i}{\tilde{z}_i} \\ \frac{\tilde{y}_i}{\tilde{z}_i} \end{bmatrix}$

dove:

$$\mathbf{M} = \begin{bmatrix} f & 0 & x_C & 0 \\ 0 & f & y_C & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} f\,\mathbf{r}_1 + x_C\,\mathbf{r}_3 & f\,t_X + x_C t_Z \\ f\,\mathbf{r}_2 + y_C\,\mathbf{r}_3 & f\,t_Y + y_C t_Z \\ \mathbf{r}_3 & t_Z \end{bmatrix} = \begin{bmatrix} f\,r_{11} + x_C r_{31} & f\,r_{12} + x_C r_{32} & f\,r_{13} + x_C r_{33} & f\,t_X + x_C t_Z \\ f\,r_{21} + y_C r_{31} & f\,r_{22} + y_C r_{32} & f\,r_{23} + y_C r_{33} & f\,t_Y + y_C t_Z \\ r_{31} & r_{32} & r_{33} & t_Z \end{bmatrix}$$

Quindi, considerando $\mathbf{p}_i$ (in coordinate euclidee):

$$\mathbf{p}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \tilde{x}_i / \tilde{z}_i \\ \tilde{y}_i / \tilde{z}_i \end{bmatrix} = \begin{bmatrix} \mathbf{m}_1 \tilde{\mathbf{P}}_i / \mathbf{m}_3 \tilde{\mathbf{P}}_i \\ \mathbf{m}_2 \tilde{\mathbf{P}}_i / \mathbf{m}_3 \tilde{\mathbf{P}}_i \end{bmatrix} \Rightarrow \begin{cases} \mathbf{m}_1 \tilde{\mathbf{P}}_i - x_i\, \mathbf{m}_3 \tilde{\mathbf{P}}_i = 0 \\ \mathbf{m}_2 \tilde{\mathbf{P}}_i - y_i\, \mathbf{m}_3 \tilde{\mathbf{P}}_i = 0 \end{cases}, \quad i = 1..N$$

2N equazioni, 12 (**11**) incognite

$\rightarrow$ $\mathbf{P} \cdot \mathbf{m} = \mathbf{0}$

Sistema lineare omogeneo, in **11** incognite (12, a meno di un fattore di scala) e ***2N*** equazioni
➔ risolvibile per ***N* ≥ 6** (**almeno 6 punti fiduciali**)

Calibrazione di camera – approccio lineare

Per ogni punto fiduciale $\mathbf{P}_i$ $(i = 1..N)$, localizzato nell'immagine in $\mathbf{p}_i$:

$$p_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \tilde{x}_i / \tilde{z}_i \\ \tilde{y}_i / \tilde{z}_i \end{bmatrix} = \begin{bmatrix} \mathbf{m}_1 \tilde{\mathbf{P}}_i / \mathbf{m}_3 \tilde{\mathbf{P}}_i \\ \mathbf{m}_2 \tilde{\mathbf{P}}_i / \mathbf{m}_3 \tilde{\mathbf{P}}_i \end{bmatrix} \Rightarrow \begin{cases} \mathbf{m}_1 \tilde{\mathbf{P}}_i - x_i\, \mathbf{m}_3 \tilde{\mathbf{P}}_i = 0 \\ \mathbf{m}_2 \tilde{\mathbf{P}}_i - y_i\, \mathbf{m}_3 \tilde{\mathbf{P}}_i = 0 \end{cases}, \quad i = 1..N$$

2N equazioni, **12** incognite m_{ij}

Posso scriverle le ***2N*** equazioni in forma di sistema lineare nelle **12** incognite m_{ij} (gli elementi di **M**):

$$\begin{bmatrix} P_{1x} & P_{1y} & P_{1z} & 1 & 0 & 0 & 0 & 0 & -x_1P_{1x} & -x_1P_{1y} & -x_1P_{1z} & -x_1 \\ 0 & 0 & 0 & 0 & P_{1x} & P_{1y} & P_{1z} & 1 & -y_1P_{1x} & -y_1P_{1y} & -y_1P_{1z} & -y_1 \\ \dots & \dots & \dots & \dots & \dots & \dots & & \dots & \dots & \dots & \dots & \dots \\ P_{Nx} & P_{Ny} & P_{Nz} & 1 & 0 & 0 & 0 & 0 & -x_NP_{Nx} & -x_NP_{Ny} & -x_NP_{Nz} & -x_N \\ 0 & 0 & 0 & 0 & P_{Nx} & P_{Ny} & P_{Nz} & 1 & -y_NP_{Nx} & -y_NP_{Ny} & -y_NP_{Nz} & -y_N \end{bmatrix} \cdot \begin{bmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \dots \\ 0 \end{bmatrix}$$

$$\mathbf{P}_{[2N \times 12]} \cdot \mathbf{m}_{[12 \times 1]} = \mathbf{0}_{[12 \times 1]}$$

Calibrazione di camera – approccio lineare

Sistema lineare omogeneo:

$$\mathbf{P}\cdot\mathbf{m}=\mathbf{0}$$

dove: $\mathbf{P}_{[2N\times 12]}=\begin{bmatrix}\mathbf{P}_1^T & \mathbf{0}^T & -x_1\mathbf{P}_1^T\\ \mathbf{0}^T & \mathbf{P}_1^T & -y_1\mathbf{P}_1^T\\ \cdots & \cdots & \cdots\\ \mathbf{P}_N^T & \mathbf{0}^T & -x_N\mathbf{P}_N^T\\ \mathbf{0}^T & \mathbf{P}_N^T & -y_N\mathbf{P}_N^T\end{bmatrix}$; $\mathbf{m}_{[12\times 1]}=\begin{bmatrix}\mathbf{m}_1\\ \mathbf{m}_2\\ \mathbf{m}_3\end{bmatrix}$

Per **N>6** (più di 6 punti fiduciali): **sistema lineare sovradeterminato**

- ci interessa però la soluzione **non triviale** (soluzione triviale: $\mathbf{m}=\mathbf{0}$)

➔ da risolvere ai **minimi quadrati**:

Soluzione: $\hat{\mathbf{m}} \quad t.c. \quad \|\hat{\mathbf{m}}\|=1 \quad \rightarrow \quad \hat{\mathbf{m}}=\underset{\mathbf{m}}{\operatorname{argmin}}\|\mathbf{P}\cdot\mathbf{m}\|^2$

- Soluzione in forma chiusa attraverso il calcolo degli **autovettori di $\mathbf{P}^T\mathbf{P}$** o, equivalentemente, la **decomposizione ai valori singolari (SVD) di P:**

➔ **m**: **autovettore** relativo all'**autovalore più piccolo** (non nullo) **di $\mathbf{P}^T\mathbf{P}$**

MATLAB scripts:

```
[W,d] = eig(P'*P);
m = W(:,1);
```

```
[U,S,V] = svd(P);
m = V(:,end);
```


Singular Value Decomposition (SVD)

Autovettori/autovalori

Data la matrice quadrata **A** [n x n]:

$\mathbf{x}, \lambda$ sono un autovettore/autovalore di **A** se: $\mathbf{A}\cdot\mathbf{x}=\lambda\,\mathbf{x}$

Nel nostro caso: $\mathbf{P}\cdot\mathbf{m}=\mathbf{0} \quad \rightarrow \quad \mathbf{P}^T\mathbf{P}\cdot\mathbf{m}=\mathbf{P}^T\mathbf{0}=\mathbf{0}$

Siccome: $\det(\mathbf{P}^T\mathbf{P})\neq 0$, l'autovalore/autovettore più piccolo (ma ≠0) corrisponde alla soluzione non triviale ($\mathbf{m}\neq\mathbf{0}$)

che più si avvicina a $\mathbf{P}\cdot\mathbf{m}=\mathbf{0}$

MATLAB:

```
[W,d] = eig(P'*P);
m = W(:,1);
```

MATLAB: **eig()**: autovalori/autovettori in ordine crescente

Singular Value Decomposition (SVD):

Data la matrice rettangolare **M** [P x n]:

$$\mathbf{M} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}$$

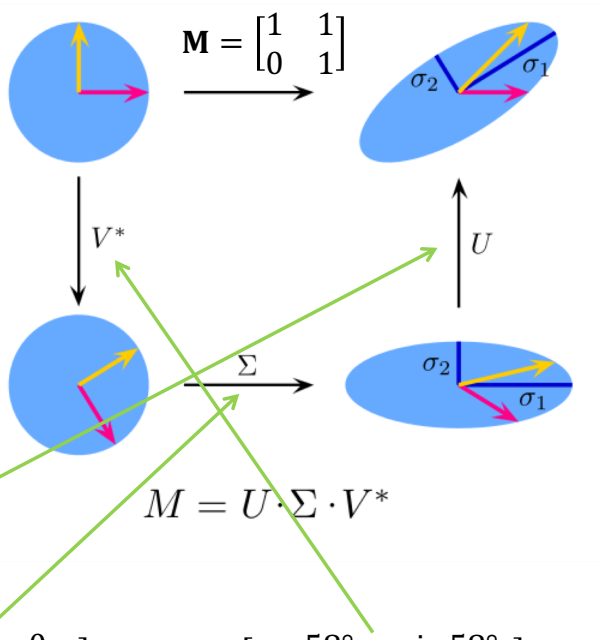
- **U** : matrice ortogonale [P x n]
- **Σ** : matrice diagonale, [n x n]
- **V** : matrice ortogonale, [n x q]

Esempio in 2D ($M_{[2x2]}$):

$$\mathbf{y} = \mathbf{M}\,\mathbf{x} \quad dove: \quad \mathbf{M} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$[\mathbf{U}, \Sigma, \mathbf{V}] = svd(\mathbf{M})$$

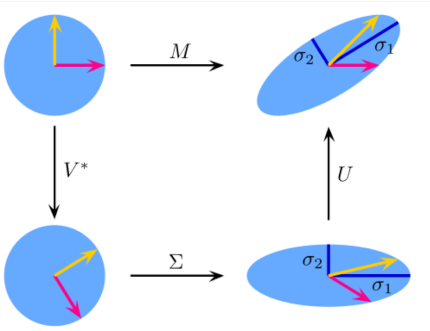
$$\Rightarrow \quad \mathbf{U} = \begin{bmatrix} \cos 32° & \sin 32° \\ \sin 32° & -\cos 32° \end{bmatrix}, \quad \mathbf{\Sigma} = \begin{bmatrix} 1.618 & 0 \\ 0 & 0.618 \end{bmatrix}, \quad \mathbf{V} = \begin{bmatrix} \cos 58° & \sin 58° \\ \sin 58° & -\cos 58° \end{bmatrix}$$



Singular Value Decomposition (SVD):

Data la matrice **P** [P x n]:

- $\mathbf{P} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}$
  - **U**: matrice ortogonale [P x q]
  - **S**: matrice diagonale, [q x q]
  - **V**: matrice ortogonale, [q x n]



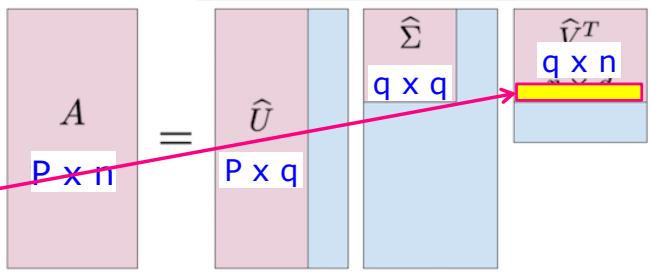
Nel nostro caso: $\mathbf{P} \cdot \mathbf{m} = \mathbf{0}$

sistema lineare omogeneo con

$$q = \mathbf{rank}(\mathbf{P}) < n$$

m noto a meno di un fattore di scala

$$\rightarrow q = 11$$



➔ soluzione:
autovettore di **V** più piccolo:
ultima colonna di V
a meno di un fattore di scala

```
MATLAB:   [U,S,V] = svd(P);
          m = V(:,end);
```

Calibrazione di camera – approccio lineare

Ottenuti i valori di $\boldsymbol{m}$ che corrispondono ai 12 coefficienti di **M** (a meno di un fattore di scala ρ), si tratta di determinare i parametri di calibrazione:

$$\xi = [\mathbf{R};\ \mathbf{T};\ f;\ x_C; y_C]$$

Estrazione dei parametri della camera ξ dalla matrice di **proiezione M**

- Abbiamo determinato **M** a meno del fattore di scala (ignoto) ρ:

$$\mathbf{M} = \rho\,\mathbf{M}_{est} = \begin{bmatrix} f\,\mathbf{r}_1 + x_C\,\mathbf{r}_3 & f\,t_X + x_C t_Z \\ f\,\mathbf{r}_2 + y_C\,\mathbf{r}_3 & f\,t_Y + y_C t_Z \\ \mathbf{r}_3 & t_Z \end{bmatrix} = \rho \left[\begin{array}{c|c} \mathbf{A} & \mathbf{b} \end{array}\right] \mapsto \rho\mathbf{A} = \rho \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{bmatrix} = \begin{bmatrix} f\,\mathbf{r}_1 + x_C\,\mathbf{r}_3 \\ f\,\mathbf{r}_2 + y_C\,\mathbf{r}_3 \\ \mathbf{r}_3 \end{bmatrix}$$

- Sapendo che **R** è ortonormale ($|\mathbf{r}_i|=1$, i=1..3), dalla terza riga si ottiene:

$$\|\rho\,\mathbf{a}_3\| = \|\mathbf{r}_3\| = 1 \quad \rightarrow \quad \rho = \frac{\pm 1}{\|\mathbf{a}_3\|} \quad \rightarrow \quad \boxed{\mathbf{r}_3 = \rho\,\mathbf{a}_3}$$

1. Estrazione dei parametri intrinseci

- dalle prime due righe si ottiene:

$$\begin{cases} \rho\,\mathbf{a}_1 = f\,\mathbf{r}_1 + x_C\,\mathbf{r}_3 \\ \rho\,\mathbf{a}_2 = f\,\mathbf{r}_2 + y_C\,\mathbf{r}_3 \end{cases}$$

Calibrazione di camera – approccio lineare

...estrazione dei parametri intrinseci (f, x_C, y_C)

- eseguendo il prodotto scalare tra le righe di **A**, grazie alla ortonormalità di **R**, si ottiene:

$$\mathbf{r}_i \cdot \mathbf{r}_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \Rightarrow \begin{cases} \rho\mathbf{a}_1 \cdot \mathbf{r}_3 = f(\mathbf{r}_1 \cdot \mathbf{r}_3) + x_C(\mathbf{r}_3 \cdot \mathbf{r}_3) = x_C \\ \rho\mathbf{a}_2 \cdot \mathbf{r}_3 = f(\mathbf{r}_2 \cdot \mathbf{r}_3) + y_C(\mathbf{r}_3 \cdot \mathbf{r}_3) = y_C \end{cases}$$

- calcolando il modulo del prodotto vettoriale tra le righe di **A**, si ottiene:

$$\mathbf{r}_i \times \mathbf{r}_j = \begin{cases} 0 & i = j \\ \pm\mathbf{k} & i \neq j \end{cases} \Rightarrow \begin{cases} \rho^2\|\mathbf{a}_1 \times \mathbf{a}_3\| = \|f(\mathbf{r}_1 \times \mathbf{r}_3) + x_C(\mathbf{r}_3 \times \mathbf{r}_3)\| = f \\ \rho^2\|\mathbf{a}_2 \times \mathbf{a}_3\| = \|f(\mathbf{r}_2 \times \mathbf{r}_3) + x_C(\mathbf{r}_3 \times \mathbf{r}_3)\| = f \end{cases}$$

versore ortogonale a i,j

prodotto scalare (3D)

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 b_3$$

prodotto vettoriale (3D)

$$\mathbf{a} \times \mathbf{b} = \det \begin{bmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix} = \begin{bmatrix} a_2 b_3 - b_2 a_3 \\ -a_1 b_3 + b_1 a_3 \\ a_1 b_2 - b_1 a_2 \end{bmatrix}$$

Calibrazione di camera – approccio lineare

2. Estrazione dei parametri estrinseci

- eseguendo il prodotto vettoriale tra le righe di **A** si ottiene:

$$\begin{cases} \rho^2(\mathbf{a}_2\times\mathbf{a}_3) = f(\mathbf{r}_2\times\mathbf{r}_3) + x_C(\mathbf{r}_3\times\mathbf{r}_3) = f\,\mathbf{r}_1 \;\rightarrow\; \boxed{\mathbf{r}_1} = \dfrac{\rho^2}{f}(\mathbf{a}_2\times\mathbf{a}_3) = \dfrac{\mathbf{a}_2\times\mathbf{a}_3}{\|\mathbf{a}_2\times\mathbf{a}_3\|} \\ \rho^2(\mathbf{a}_1\times\mathbf{a}_3) = f(\mathbf{r}_1\times\mathbf{r}_3) + y_C(\mathbf{r}_3\times\mathbf{r}_3) = -f\,\mathbf{r}_2 \;\rightarrow\; \boxed{\mathbf{r}_2} = -\dfrac{\mathbf{a}_1\times\mathbf{a}_3}{\|\mathbf{a}_1\times\mathbf{a}_3\|} = \mathbf{r}_3\times\mathbf{r}_1 \end{cases} \rightarrow \boxed{\mathbf{R} = \begin{bmatrix}\mathbf{r}_1\\ \mathbf{r}_2\\ \mathbf{r}_3\end{bmatrix}}$$

per ortonormalità

- Infine, si può ora determinare **T** considerando che:

$$\mathbf{M} = \left[\begin{array}{c:c} f\,\mathbf{r}_1 + x_C\,\mathbf{r}_3 & f\,t_X + x_C\,t_Z \\ f\,\mathbf{r}_2 + y_C\,\mathbf{r}_3 & f\,t_Y + y_C\,t_Z \\ \mathbf{r}_3 & t_Z \end{array}\right] = \rho\left[\begin{array}{c:c} \mathbf{A} & \mathbf{b} \end{array}\right] \;\rightarrow\; \rho\,\mathbf{b} = \mathbf{K}_1\cdot\mathbf{T} = \begin{bmatrix} f\,t_X + x_C\,t_Z \\ f\,t_Y + y_C\,t_Z \\ t_Z \end{bmatrix}$$

- Quindi:

$$\rho\,\mathbf{b} = \rho\begin{bmatrix}b_1\\ b_2\\ b_3\end{bmatrix} = \begin{bmatrix} f t_x - x_C t_z \\ f t_y - y_C t_z \\ t_z \end{bmatrix} \rightarrow \boxed{t_Z} = \frac{b_3}{\rho} \rightarrow \begin{cases} \boxed{t_X} = \dfrac{1}{f}(\rho b_1 - x_C t_Z) \\ \boxed{t_Y} = \dfrac{1}{f}(\rho b_2 - y_C t_Z) \end{cases}$$

Camera Calibration – lab exercise

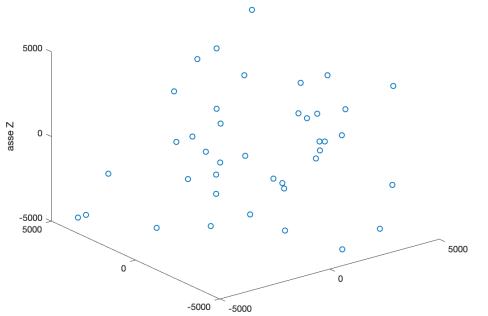
Esercitazione MATLAB®:
Calibrazione di camera con modello lineare

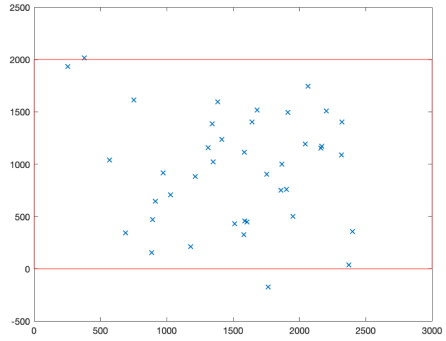
- MATLAB script: **`LinearCamCal.m`**

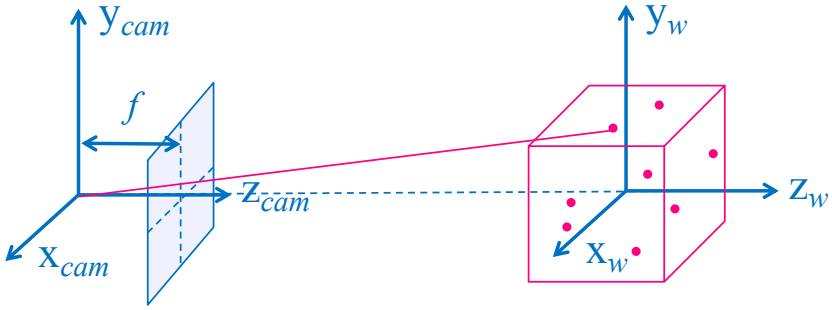
```
>> LinearCamCal
```

Esercitazione:

- *esecuzione passo-passo dello script*
- *ri-esecuzione, con parametri differenti*
- *valutazione della precisione di calcolo*







Calibrazione – approccio non lineare

- necessario, ad esempio, per calibrare in presenza di distorsione radiale

$$\tilde{\mathbf{p}}_i = \begin{bmatrix} \tilde{x}_i \\ \tilde{y}_i \\ \tilde{z}_i \end{bmatrix} = \mathbf{M}(\xi) \cdot \tilde{\mathbf{P}}_\mathbf{i} \quad \rightarrow \quad \tilde{\mathbf{p}}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = f(\xi, \mathbf{P}_\mathbf{i})$$

lineare *non lineare*

Algoritmo:

1. Risolvo dapprima con il modello lineare (trascuro la distorsione radiale)

➔ ottengo una prima stima dei parametri: ξ_0

2. Utilizzo ξ_0 come stima iniziale per una ottimizzazione non lineare che minimizza l'errore di riproiezione ***E*** → ottengo il modello finale: $\hat{\xi}$

$$E = \sum_i \left\| \mathbf{p}_\mathbf{i}^{MEAS} - \mathbf{p}_\mathbf{i}^{EST} \right\|^2 = \sum_i \left\| \mathbf{p}_\mathbf{i}^{MEAS} - f(\xi, \mathbf{P}_\mathbf{i}) \right\|^2 \quad \rightarrow \quad \hat{\xi} \quad t.c. \quad \hat{\xi} = \underset{\xi}{\operatorname{argmin}}(E)$$

Calibrazione – approccio non lineare

2. Utilizzo ξ_0 come stima iniziale per una ottimizzazione non lineare che minimizza l'errore di riproiezione ***E*** → ottengo il modello finale $\hat{\xi}$

$$E = \sum_i \left\| \mathbf{p}_\mathbf{i}^{MEAS} - \mathbf{p}_\mathbf{i}^{EST} \right\|^2 = \sum_i \left\| \mathbf{p}_\mathbf{i}^{MEAS} - f(\xi, \mathbf{P}_\mathbf{i}) \right\|^2 \quad \rightarrow \quad \hat{\xi} \quad t.c. \quad \hat{\xi} = \underset{\xi}{\operatorname{argmin}}(E)$$

Dato il modello iniziale: $\xi_0 = \{\mathbf{R}, \mathbf{t}, f, x_C, y_C, k_D\}$ *e* N *punti fiduciali* P_i *:*

$\xi = \xi_0$

$p_i^{EST} = f(\xi, P_i)$:

$$P_i \;\mapsto\; P_{cam,i} = \mathbf{R} \cdot P_i + \mathbf{t}, \qquad i = 1..N$$

$$P_{cam,i} \;\mapsto\; p_{im,i} = \begin{bmatrix} x_C \\ y_C \end{bmatrix} + \frac{f}{z_{cam,i}} \begin{bmatrix} x_{cam,i} \\ y_{cam,i} \end{bmatrix}, \qquad i = 1..N$$

$$p_{im,i} \;\mapsto\; p_i^{EST} = p_{im,i}\,(1 + k_{D1} r^2 + k_{D2} r^4), \qquad i = 1..N$$

$$Errore: \quad E(\xi) = \sum_{i=1}^{N} \left\| p_i^{MEAS} - p_i^{EST} \right\|^2 \;\rightarrow\; nuova\ stima\ di\ \xi$$

Camera Calibration: algoritmo di Tsai (1987)

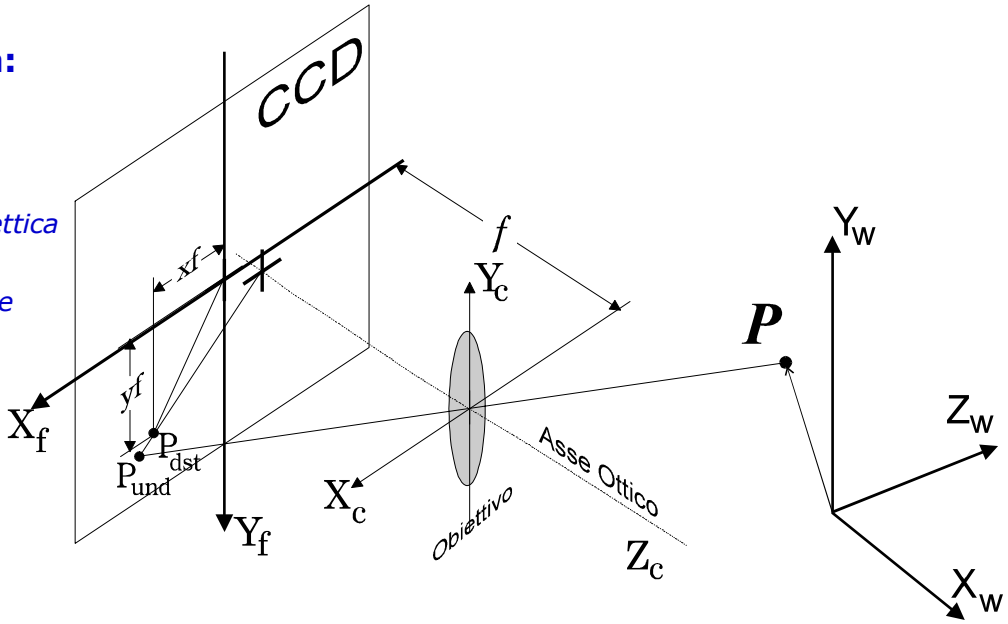
TSAI (1987):

R. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," in ***IEEE Journal on Robotics and Automation***, vol. 3, no. 4, pp. 323-344, 1987.

Tecnica di calibrazione basata su modello geometrico Euclideo di camera

Modello di camera:

1. Exterior orientation
   - *roto-traslazione*
2. Interior orientation
   - *proiezione prospettica*
3. Distortion
   - *distordione radiale*



Camera Calibration: algoritmo di Tsai (1987)

Tsai: modello di camera

Exterior Orientation (parametri estrinseci)

- Roto-traslazione: $\mathbf{P}_{CAM} = \mathbf{R} \cdot \mathbf{P}_{CAM} + \mathbf{t} \rightarrow \begin{bmatrix} X_{CAM} \\ Y_{CAM} \\ Z_{CAM} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix} + \begin{bmatrix} t_X \\ t_Y \\ t_Z \end{bmatrix}$

Interior Orientation (parametri intrinseci)

- Proiezione prospettica: $\frac{x_U - x_C}{f} = \frac{X_{CAM}}{Z_{CAM}}, \quad \frac{y_U - y_C}{f} = \frac{Y_{CAM}}{Z_{CAM}}$

Modello completo:

$$\frac{x_U - x_C}{f} = \frac{X_{CAM}}{Z_{CAM}} = \frac{r_{11}X_W + r_{12}Y_W + r_{13}Z_W + t_X}{r_{31}X_W + r_{32}Y_W + r_{33}Z_W + t_Z} \quad , \quad \frac{y_U - y_C}{f} = \frac{Y_{CAM}}{Z_{CAM}} = \frac{r_{21}X_W + r_{22}Y_W + r_{23}Z_W + t_Y}{r_{31}X_W + r_{32}Y_W + r_{33}Z_W + t_Z}$$

- Correzione delle **distorsione radiale**: $\begin{cases} x_D = x_U(1 + k_1 r^2 + k_2 r^4 + \cdots) \\ y_D = y_U(1 + k_1 r^2 + k_2 r^4 + \cdots) \end{cases}; \; r^2 = x_U^2 + y_U^2$

Camera Calibration: algoritmo di Tsai (1987)

Struttura dell'algoritmo:

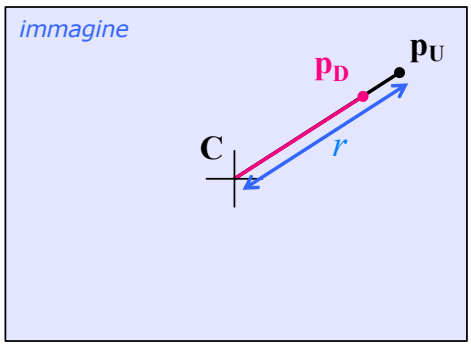
1. Stima di alcuni parametri (approssimata)
   risolvendo un sistema **lineare** sovradeterminato **non** omogeneo
   ➔ risolubile ai minimi quadrati mediante la matrice **pseudo-inversa**

2. Stima degli altri parametri (e raffinamento della stima precedente)
   risolvendo un sistema **non lineare** con tecniche di **ottimizzazione numerica**
   (es. metodo di Levemberg-Marquardt)

Radial Alignment Constraint (RAC):

- La distorsione è radiale, quindi i vettori $\mathbf{p_D}$ e $\mathbf{p_U}$ sono PARALLELI!

$$\begin{cases} x'_{U,D} = x_{U,D} - x_C \\ y'_{U,D} = y_{U,D} - y_C \end{cases} \rightarrow \frac{x'_U}{y'_U} = \frac{x'_D}{y'_D}$$



Camera Calibration: algoritmo di Tsai (1987)

Algoritmo

Modello completo:

$$\frac{x_U - x_C}{f} = \frac{x'_U}{f} = \frac{X_{CAM}}{Z_{CAM}} = \frac{r_{11}X_W + r_{12}Y_W + r_{13}Z_W + t_X}{r_{31}X_W + r_{32}Y_W + r_{33}Z_W + t_Z} \quad , \quad \frac{y_U - y_C}{f} = \frac{y'_U}{f} = \frac{Y_{CAM}}{Z_{CAM}} = \frac{r_{21}X_W + r_{22}Y_W + r_{23}Z_W + t_Y}{r_{31}X_W + r_{32}Y_W + r_{33}Z_W + t_Z}$$

Facendo il rapporto, sfrutto il *Radial Alignment Constraint*:

$$\frac{x'_D}{y'_D} = \frac{x'_U}{y'_U} = \frac{X_{CAM}}{Y_{CAM}} = \frac{r_{11}X_W + r_{12}Y_W + r_{13}Z_W + t_X}{r_{21}X_W + r_{22}Y_W + r_{23}Z_W + t_Y}$$

1 equazione/punto
8 incognite (vincolate...)

Sviluppando:

$$\left(r_{11}X_W + r_{12}Y_W + r_{13}Z_W + t_X\right)y'_D - \left(r_{21}X_W + r_{22}Y_W + r_{23}Z_W + t_Y\right)x'_D = 0$$

incognite

Ponendo arbitrariamente $t_Y=1$ (risolvo a meno di un fattore di scala), ottengo:

$$\left(r_{11}X_W + r_{12}Y_W + r_{13}Z_W + t_X\right)y'_D - \left(r_{21}X_W + r_{22}Y_W + r_{23}Z_W\right)x'_D = x'_D$$

1 equazione/punto
7 incognite

Camera Calibration: algoritmo di Tsai (1987)

Soluzione modello lineare (RAC):

- In forma matriciale:

$$\begin{bmatrix} X_W y'_D & Y_W y'_D & Z_W y'_D & y'_D & -X_W x'_D & -Y_W x'_D & -Z_W x'_D \end{bmatrix} \begin{bmatrix} r_{11} \\ r_{12} \\ r_{13} \\ t_X \\ r_{21} \\ r_{22} \\ r_{23} \end{bmatrix} = x'_D$$

- Cioè: $\mathbf{A_i} \cdot \mathbf{m} = \mathbf{b_i}, \quad \mathbf{i} = 1..N$

- Estendo a tutti i gli *N* punti fiduciali:

$$\mathbf{A} \cdot \mathbf{m} = \mathbf{b} \qquad \mathbf{A}:[N \times 7], \quad \mathbf{m}:[7 \times 1], \quad \mathbf{b}:[N \times 1]$$

A: matrice Nx7 ➔ per *N* > 7, sistema **lineare sovradeterminato, non omogeneo**

- Risolvo ai minimi quadrati mediante la **pseudoinversa:**

$$\mathbf{A}^T\mathbf{A} \cdot \mathbf{m} = \mathbf{A}^T\mathbf{b} \quad \rightarrow \quad \mathbf{m} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}$$

Camera Calibration: algoritmo di Tsai (1987)

Raffinamento dei parametri e stima globale

1. Correggo la stima di $\mathbf{r}_1$ e $\mathbf{r}_2$, imponendo la ortonormalità della matrice **R**

$$\begin{cases} r_i \cdot r_i = \|r_i\| = 1, \quad i = 1,2,3 & \leftarrow \textit{ norma = 1} \\ r_1 \cdot r_2 = r_1 \cdot r_3 = r_2 \cdot r_3 = 0 & \leftarrow \textit{ ortogonalità} \end{cases} \quad \rightarrow \mathbf{R}$$

2. Utilizzo i parametri ottenuti come stima iniziale per una ottimizzazione non lineare che minimizza l'errore di riproiezione ***E*** → ottengo il modello finale.

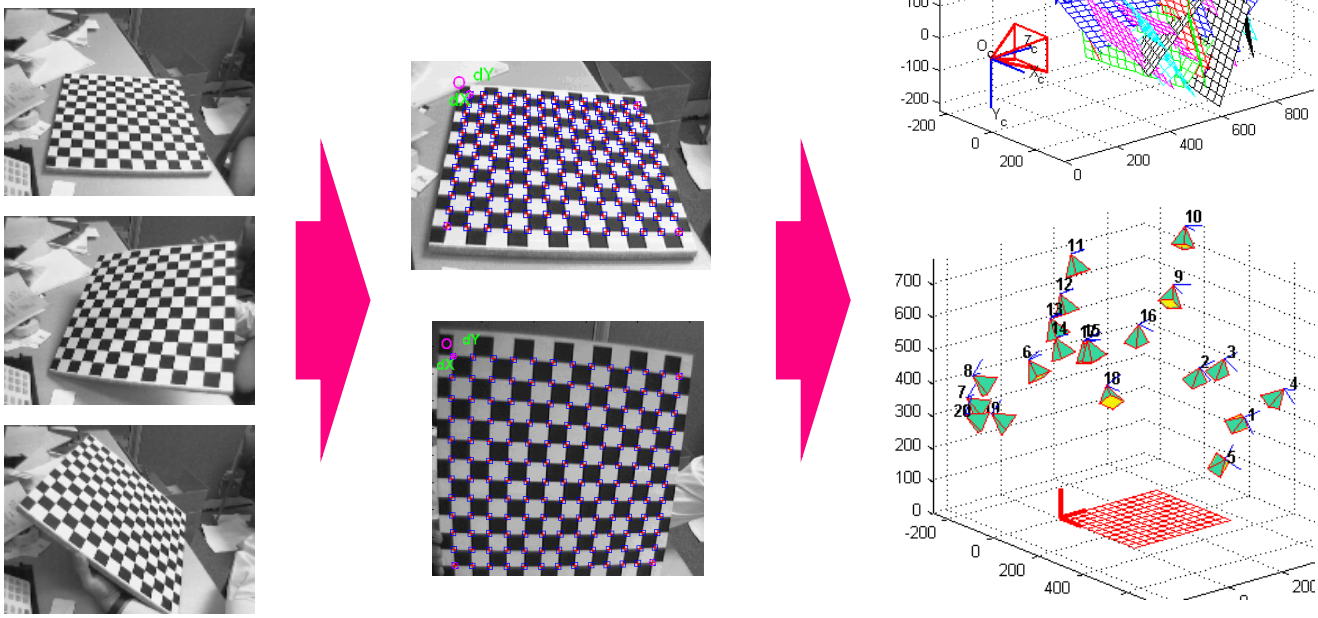
$$E = \sum_i \left\| \mathbf{p}_\mathbf{i}^{MEAS} - \mathbf{p}_\mathbf{i}^{EST} \right\|^2 = \sum_i \left\| \mathbf{p}_\mathbf{i}^{MEAS} - f\left(\xi, \mathbf{P}_\mathbf{i}\right) \right\|^2 \quad \rightarrow \quad \hat{\xi} \quad t.c. \quad \hat{\xi} = \underset{\xi}{\operatorname{argmin}}\left(E\right)$$

Algoritmo ben spiegato in:
Berthold K.P. Horn – **Tsai's Camera Calibration Revisited**, 2000
http://people.csail.mit.edu/bkph/articles/Tsai_Revisited.pdf

Camera Calibration: algoritmo di Tsai (1987)

Camera Calibration Software Tools

- Camera Calibration Toolbox for MATLAB®:
  http://www.vision.caltech.edu/bouguetj/calib_doc/



Camera Calibration – lab exercise

MATLAB®: Single Camera Calibration App

```
>> cameraCalibrator
```