Geometric Camera Calibration
Chapter 2

Guido Gerig

CS 6643 Spring 2017

Slides modified from Marc Pollefeys, UNC Chapel Hill, Comp256,
Other slides and illustrations from J. Ponce, addendum to course book,
and Trevor Darrell, Berkeley, C280 Computer Vision Course.

Equation: World coordinates to image pixels

pixel coordinates

world coordinates

$$\vec{p} = \frac{1}{z} M \; {}^{W}\vec{p}$$

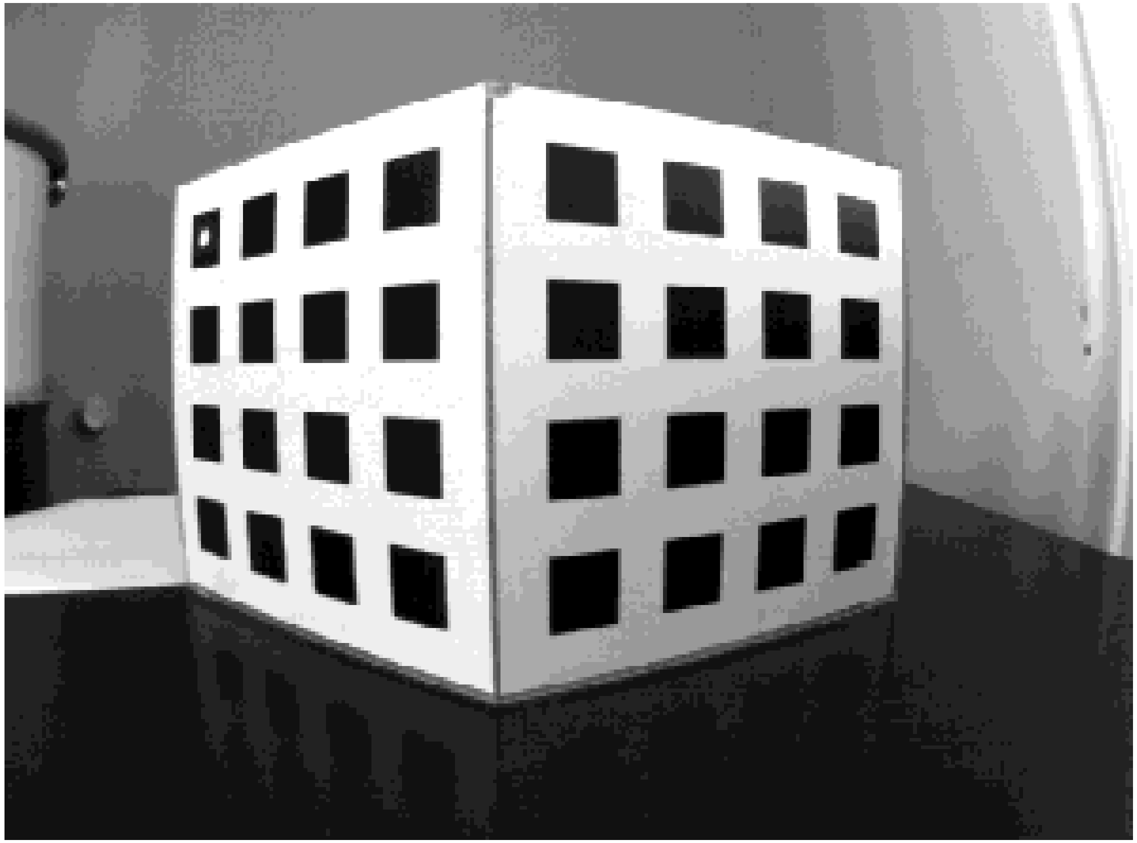
$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{1}{z}\begin{pmatrix} . & m_1^T & . & . \\ . & m_2^T & . & . \\ . & m_3^T & . & . \end{pmatrix}\begin{pmatrix} {}^{W}p_x \\ {}^{W}p_y \\ {}^{W}p_z \\ 1 \end{pmatrix}$$

$$\begin{cases} u = \dfrac{m_1 \cdot \vec{P}}{m_3 \cdot \vec{P}} \\[2ex] v = \dfrac{m_2 \cdot \vec{P}}{m_3 \cdot \vec{P}} \end{cases}$$

Conversion back from homogeneous coordinates leads to:

W. Freeman

Calibration target



The Opti-CAL Calibration Target Image

Find the position, u_i and v_i, in pixels,
of each calibration object feature point.

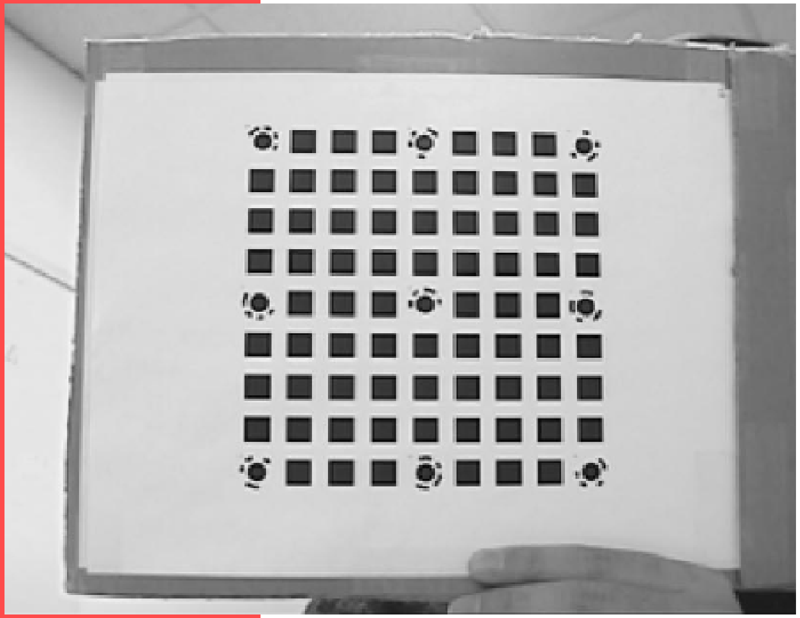
http://www.kinetic.bc.ca/CompVision/opti-CAL.html

Camera calibration

From before, we had these equations relating image positions, u,v, to points at 3-d positions P (in homogeneous coordinates):

$$u = \frac{m_1 \cdot \vec{P}}{m_3 \cdot \vec{P}}$$

$$v = \frac{m_2 \cdot \vec{P}}{m_3 \cdot \vec{P}}$$



So for each feature point, i, we have:

$$(m_1 - u_i m_3) \cdot \vec{P}_i = 0$$

$$(m_2 - v_i m_3) \cdot \vec{P}_i = 0$$

Camera calibration

Stack all these measurements of i=1…n points

$$(m_1 - u_i m_3) \cdot \vec{P}_i = 0$$

$$(m_2 - v_i m_3) \cdot \vec{P}_i = 0$$

into a big matrix:

$$\begin{pmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \\ \cdots & \cdots & \cdots \\ P_n^T & 0^T & -u_n P_n^T \\ 0^T & P_n^T & -v_n P_n^T \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

W. Freeman

Camera calibration

In vector form:

$$\begin{pmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \\ \cdots & \cdots & \cdots \\ P_n^T & 0^T & -u_n P_n^T \\ 0^T & P_n^T & -v_n P_n^T \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

Showing all the elements:

$$\begin{pmatrix} P_{1x} & P_{1y} & P_{1z} & 1 & 0 & 0 & 0 & 0 & -u_1 P_{1x} & -u_1 P_{1y} & -u_1 P_{1z} & -u_1 \\ 0 & 0 & 0 & 0 & P_{1x} & P_{1y} & P_{1z} & 1 & -v_1 P_{1x} & -v_1 P_{1y} & -v_1 P_{1z} & -v_1 \\ & & & & & & \cdots & \cdots & \cdots & & & \\ P_{nx} & P_{ny} & P_{nz} & 1 & 0 & 0 & 0 & 0 & -u_n P_{nx} & -u_n P_{ny} & -u_n P_{nz} & -u_n \\ 0 & 0 & 0 & 0 & P_{nx} & P_{ny} & P_{nz} & 1 & -v_n P_{nx} & -v_n P_{ny} & -v_n P_{nz} & -v_n \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

W. Freeman

Camera calibration

$$\begin{pmatrix} P_{1x} & P_{1y} & P_{1z} & 1 & 0 & 0 & 0 & 0 & -u_1P_{1x} & -u_1P_{1y} & -u_1P_{1z} & -u_1 \\ 0 & 0 & 0 & 0 & P_{1x} & P_{1y} & P_{1z} & 1 & -v_1P_{1x} & -v_1P_{1y} & -v_1P_{1z} & -v_1 \\ & & & & & & \cdots & \cdots & \cdots & & & \\ P_{nx} & P_{ny} & P_{nz} & 1 & 0 & 0 & 0 & 0 & -u_nP_{nx} & -u_nP_{ny} & -u_nP_{nz} & -u_n \\ 0 & 0 & 0 & 0 & P_{nx} & P_{ny} & P_{nz} & 1 & -v_nP_{nx} & -v_nP_{ny} & -v_nP_{nz} & -v_n \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \\ m_{34} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

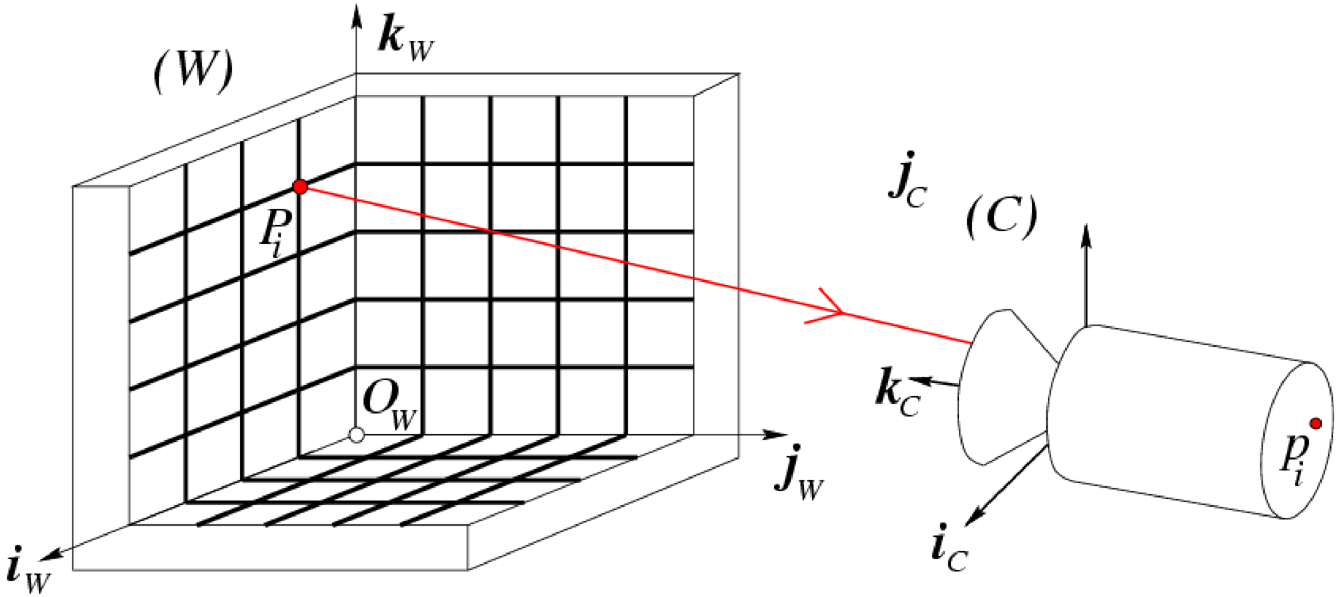
Q m = 0

We want to solve for the unit vector m (the stacked one) that minimizes $|Qm|^2$

The eigenvector assoc. to the minimum eigenvalue of the matrix Q^TQ gives us that because it is the unit vector x that minimizes $x^T Q^TQ\ x$.

W. Freeman

Calibration Problem



Given n points $P_1, \ldots, P_n$ with *known* positions and their images $p_1, \ldots, p_n$

Find $\boldsymbol{i}$ and $\boldsymbol{e}$ such that

$$\sum_{i=1}^{n}\left[\left(u_i - \frac{\boldsymbol{m}_1(\boldsymbol{i},\boldsymbol{e})\cdot\boldsymbol{P}_i}{\boldsymbol{m}_3(\boldsymbol{i},\boldsymbol{e})\cdot\boldsymbol{P}_i}\right)^2 + \left(v_i - \frac{\boldsymbol{m}_2(\boldsymbol{i},\boldsymbol{e})\cdot\boldsymbol{P}_i}{\boldsymbol{m}_3(\boldsymbol{i},\boldsymbol{e})\cdot\boldsymbol{P}_i}\right)^2\right] \quad \text{is minimized}$$

Analytical Photogrammetry

Given n points $P_1, \dots, P_n$ with *known* positions and their images $p_1, \dots, p_n$

Find $\boldsymbol{i}$ and $\boldsymbol{e}$ such that

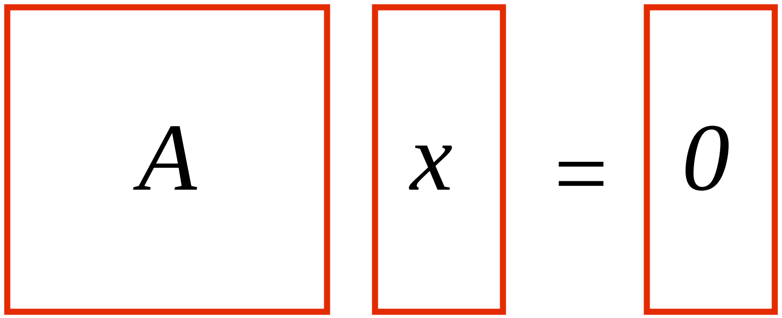
$$\sum_{i=1}^{n} \left[\left(u_i - \frac{\boldsymbol{m}_1(\boldsymbol{i},\boldsymbol{e}) \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3(\boldsymbol{i},\boldsymbol{e}) \cdot \boldsymbol{P}_i} \right)^2 + \left(v_i - \frac{\boldsymbol{m}_2(\boldsymbol{i},\boldsymbol{e}) \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3(\boldsymbol{i},\boldsymbol{e}) \cdot \boldsymbol{P}_i} \right)^2 \right] \quad \text{is minimized}$$

Non-Linear Least-Squares Methods

- Newton
- Gauss-Newton
- Levenberg-Marquardt

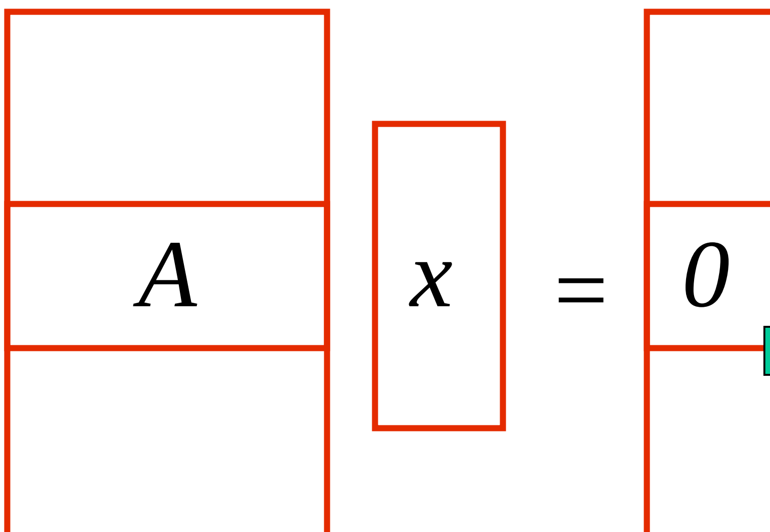
Iterative, quadratically convergent in favorable situations

Homogeneous Linear Systems



Square system:

- unique solution: 0
- unless Det(A)=0



Rectangular system ??

- 0 is always a solution

Minimize $|Ax|^2$
under the constraint $|x|^2=1$

How do you solve overconstrained homogeneous linear equations ??

$$E = |\mathcal{U}\boldsymbol{x}|^2 = \boldsymbol{x}^T(\mathcal{U}^T\mathcal{U})\boldsymbol{x}$$

- Orthonormal basis of eigenvectors: $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_q$.
- Associated eigenvalues: $0 \leq \lambda_1 \leq \ldots \leq \lambda_q$.
- Any vector can be written as

$$\boldsymbol{x} = \mu_1\boldsymbol{e}_1 + \ldots + \mu_q\boldsymbol{e}_q$$

for some μ_i $(i = 1, \ldots, q)$ such that $\mu_1^2 + \ldots + \mu_q^2 = 1$.

$$\begin{aligned} E(\boldsymbol{x}) - E(\boldsymbol{e}_1) &= \boldsymbol{x}^T(\mathcal{U}^T\mathcal{U})\boldsymbol{x} - \boldsymbol{e}_1^T(\mathcal{U}^T\mathcal{U})\boldsymbol{e}_1 \\ &= \lambda_1^2\mu_1^2 + \ldots + \lambda_q^2\mu_q^2 - \lambda_1^2 \\ &\geq \lambda_1^2(\mu_1^2 + \ldots + \mu_q^2 - 1) = 0 \end{aligned}$$

The solution is e_1 .

remember: $EIG(U^TU)=SVD(U)$, i.e. solution is V_n

Matlab Solution

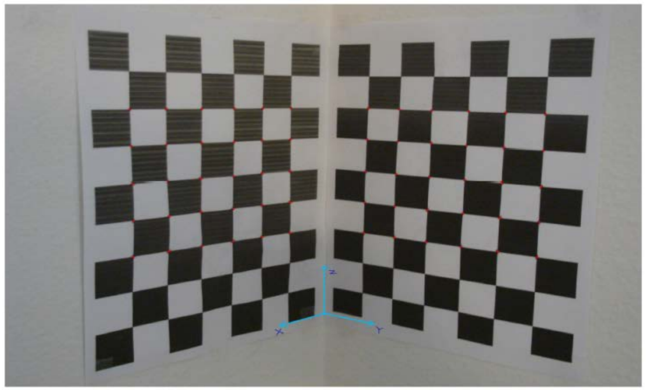


Figure 1: Checkerboard pattern on the wall corner and the world frame coordinate axes

Example:
60 point pairs

Least squares method is used to estimate the calibration matrix. There are 120 homogeneous linear equations in twelve variables, which are the coefficients of the calibration matrix $\mathcal{M}$. Lets denote this system of linear equations as

$$\mathcal{P}\mathbf{m} = 0, \qquad \mathbf{m} := [\mathbf{m}_1 \quad \mathbf{m}_2 \quad \mathbf{m}_3]^T, \tag{1}$$

where, $\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3$ are first, second and third rows of the matrix $\mathcal{M}$ respectively. $\mathbf{m}$ is a 12×1 vector, and $\mathcal{P}$ is a 120×12 matrix. The problem of least square estimation of $\mathcal{P}$ is defined as

$$\min \|\mathcal{P}\mathbf{m}\|^2, \qquad \text{subject to} \quad \|\mathbf{m}\|^2 = 1. \tag{2}$$

As it turns out, the solution of above problem is given by the eigenvector of matrix $\mathcal{P}^T\mathcal{P}$ having the least eigenvalue. The eigenvectors of matrix $\mathcal{P}^T\mathcal{P}$ can also be computed by performing the singular value decomposition (SVD) of $\mathcal{P}$. The 12 right singular vectors of $\mathcal{P}$ are also the eigenvectors of $\mathcal{P}^T\mathcal{P}$.

```
%Perform SVD of P
[U S V] = svd(P);
[min_val, min_index] = min(diag(S(1:12,1:12)));

%m is given by right singular vector of min. singular value
m = V(1:12, min_index);
```

Degenerate Point Configurations

Are there other solutions besides *M* ??

$$\mathbf{0} = \mathcal{P}\boldsymbol{l} = \begin{pmatrix} \boldsymbol{P}_1^T & \mathbf{0}^T & -u_1\boldsymbol{P}_1^T \\ \mathbf{0}^T & \boldsymbol{P}_1^T & -v_1\boldsymbol{P}_1^T \\ \dots & \dots & \dots \\ \boldsymbol{P}_n^T & \mathbf{0}^T & -u_n\boldsymbol{P}_n^T \\ \mathbf{0}^T & \boldsymbol{P}_n^T & -v_n\boldsymbol{P}_n^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\nu} \end{pmatrix} = \begin{pmatrix} \boldsymbol{P}_1^T\boldsymbol{\lambda} - u_1\boldsymbol{P}_1^T\boldsymbol{\nu} \\ \boldsymbol{P}_1^T\boldsymbol{\mu} - v_1\boldsymbol{P}_1^T\boldsymbol{\nu} \\ \dots \\ \boldsymbol{P}_n^T\boldsymbol{\lambda} - u_n\boldsymbol{P}_n^T\boldsymbol{\nu} \\ \boldsymbol{P}_n^T\boldsymbol{\mu} - v_n\boldsymbol{P}_n^T\boldsymbol{\nu} \end{pmatrix}$$

$$\begin{cases} \boldsymbol{P}_i^T\boldsymbol{\lambda} - \dfrac{\boldsymbol{m}_1^T\boldsymbol{P}_i}{\boldsymbol{m}_3^T\boldsymbol{P}_i}\boldsymbol{P}_i^T\boldsymbol{\nu} = 0 \\ \boldsymbol{P}_i^T\boldsymbol{\mu} - \dfrac{\boldsymbol{m}_2^T\boldsymbol{P}_i}{\boldsymbol{m}_3^T\boldsymbol{P}_i}\boldsymbol{P}_i^T\boldsymbol{\nu} = 0 \end{cases} \Rightarrow \begin{cases} \boldsymbol{P}_i^T(\boldsymbol{\lambda}\boldsymbol{m}_3^T - \boldsymbol{m}_1\boldsymbol{\nu}^T)\boldsymbol{P}_i = 0 \\ \boldsymbol{P}_i^T(\boldsymbol{\mu}\boldsymbol{m}_3^T - \boldsymbol{m}_2\boldsymbol{\nu}^T)\boldsymbol{P}_i = 0 \end{cases}$$

- Coplanar points: $(\lambda,\mu,\nu)=(\Pi,0,0)$ or $(0,\Pi,0)$ or $(0,0,\Pi)$
- Points lying on the intersection curve of two quadric surfaces = straight line + twisted cubic

Does not happen for 6 or more random points!

Camera calibration

Once you have the M matrix, can recover the intrinsic and extrinsic parameters.

Estimation of the Intrinsic and Extrinsic Parameters, see pdf slides S.M. Abdallah.

$$\mathcal{M} = \begin{pmatrix} \alpha \boldsymbol{r}_1^T - \alpha \cot\theta \boldsymbol{r}_2^T + u_0 \boldsymbol{r}_3^T & \alpha t_x - \alpha \cot\theta t_y + u_0 t_z \\ \dfrac{\beta}{\sin\theta}\boldsymbol{r}_2^T + v_0 \boldsymbol{r}_3^T & \dfrac{\beta}{\sin\theta} t_y + v_0 t_z \\ \boldsymbol{r}_3^T & t_z \end{pmatrix}$$

Once *M* is known, you still got to recover the intrinsic and extrinsic parameters !!!

This is a decomposition problem, not an estimation problem.

$$\boxed{\rho}\ \mathcal{M} = \begin{pmatrix} \alpha \boldsymbol{r}_1^T - \alpha \cot\theta \boldsymbol{r}_2^T + u_0 \boldsymbol{r}_3^T & \alpha t_x - \alpha \cot\theta t_y + u_0 t_z \\ \dfrac{\beta}{\sin\theta} \boldsymbol{r}_2^T + v_0 \boldsymbol{r}_3^T & \dfrac{\beta}{\sin\theta} t_y + v_0 t_z \\ \boldsymbol{r}_3^T & t_z \end{pmatrix}$$

- Intrinsic parameters
- Extrinsic parameters

3 × 3 | 3 × 1

$$\mathcal{M} = \begin{pmatrix} \alpha \boldsymbol{r}_1^T - \alpha \cot\theta \boldsymbol{r}_2^T + u_0 \boldsymbol{r}_3^T & \alpha t_x - \alpha \cot\theta t_y + u_0 t_z \\ \dfrac{\beta}{\sin\theta} \boldsymbol{r}_2^T + v_0 \boldsymbol{r}_3^T & \dfrac{\beta}{\sin\theta} t_y + v_0 t_z \\ \boldsymbol{r}_3^T & t_z \end{pmatrix}$$

A | b

we have

$$\rho \begin{pmatrix} \boldsymbol{a}_1^T \\ \boldsymbol{a}_2^T \\ \boldsymbol{a}_3^T \end{pmatrix} = \begin{pmatrix} \alpha \boldsymbol{r}_1^T - \alpha \cot\theta \boldsymbol{r}_2^T + u_0 \boldsymbol{r}_3^T \\ \dfrac{\beta}{\sin\theta} \boldsymbol{r}_2^T + v_0 \boldsymbol{r}_3^T \\ \boldsymbol{r}_3^T \end{pmatrix}.$$

In particular, using the fact that the rows of a rotation matrix have unit length and are perpendicular to each other yields immediately

$$\begin{cases} \rho = \varepsilon / |\boldsymbol{a}_3|, \\ \boldsymbol{r}_3 = \rho \boldsymbol{a}_3, \end{cases} \quad (6.3.5)$$

Orthogonality of r3 and r2

$$\rho^2(\boldsymbol{a}_2 \cdot \boldsymbol{a}_3) \rightarrow \boldsymbol{r}_3^T\left(\frac{\beta}{\sin\theta}\boldsymbol{r}_2^T + v_0\boldsymbol{r}_3^T\right)$$

ρ is known -> calculate v0 via dot product of a2 and a3

Slide Samer M Abdallah, Beirut

Estimation of the intrinsic and extrinsic parameters

Write $M = (\mathcal{A}, \boldsymbol{b})$, therefore

$$\rho(\mathcal{A} \quad \boldsymbol{b}) = \mathcal{K}(\mathcal{R} \quad \boldsymbol{t}) \Longleftrightarrow \rho \begin{pmatrix} \boldsymbol{a}_1^T \\ \boldsymbol{a}_2^T \\ \boldsymbol{a}_3^T \end{pmatrix} = \begin{pmatrix} \alpha \boldsymbol{r}_1^T - \alpha \cot\theta \boldsymbol{r}_2^T + u_0 \boldsymbol{r}_3^T \\ \dfrac{\beta}{\sin\theta} \boldsymbol{r}_2^T + v_0 \boldsymbol{r}_3^T \\ \boldsymbol{r}_3^T \end{pmatrix}$$

Using the fact that the rows of a rotation matrix have unit length and are perpendicular to each other yields

$$\begin{cases} \rho = \varepsilon / |\boldsymbol{a}_3|, \\ \boldsymbol{r}_3 = \rho \boldsymbol{a}_3, \\ u_0 = \rho^2 (\boldsymbol{a}_1 \cdot \boldsymbol{a}_3), \\ v_0 = \rho^2 (\boldsymbol{a}_2 \cdot \boldsymbol{a}_3), \end{cases} \quad \text{where} \quad \varepsilon = \mp 1.$$

Since θ is always in the neighborhood of $\pi/2$ with a positive sine, we have

$$\begin{cases} \rho^2 (\boldsymbol{a}_1 \times \boldsymbol{a}_3) = -\alpha \boldsymbol{r}_2 - \alpha \cot\theta \boldsymbol{r}_1, \\ \rho^2 (\boldsymbol{a}_2 \times \boldsymbol{a}_3) = \dfrac{\beta}{\sin\theta} \boldsymbol{r}_1, \end{cases} \quad \text{and} \quad \begin{cases} \rho^2 |\boldsymbol{a}_1 \times \boldsymbol{a}_3| = \dfrac{|\alpha|}{\sin\theta}, \\ \rho^2 |\boldsymbol{a}_2 \times \boldsymbol{a}_3| = \dfrac{|\beta|}{\sin\theta}. \end{cases}$$

Thus,

$$\begin{cases} \cos\theta = -\dfrac{(\boldsymbol{a}_1 \times \boldsymbol{a}_3) \cdot (\boldsymbol{a}_2 \times \boldsymbol{a}_3)}{|\boldsymbol{a}_1 \times \boldsymbol{a}_3| |\boldsymbol{a}_2 \times \boldsymbol{a}_3|}, \\ \alpha = \rho^2 |\boldsymbol{a}_1 \times \boldsymbol{a}_3| \sin\theta, \\ \beta = \rho^2 |\boldsymbol{a}_2 \times \boldsymbol{a}_3| \sin\theta, \end{cases} \quad \text{and} \quad \begin{cases} \boldsymbol{r}_1 = \dfrac{\rho^2 \sin\theta}{\beta} (\boldsymbol{a}_2 \times \boldsymbol{a}_3) = \dfrac{1}{|\boldsymbol{a}_2 \times \boldsymbol{a}_3|} (\boldsymbol{a}_2 \times \boldsymbol{a}_3), \\ \boldsymbol{r}_2 = \boldsymbol{r}_3 \times \boldsymbol{r}_1. \end{cases}$$

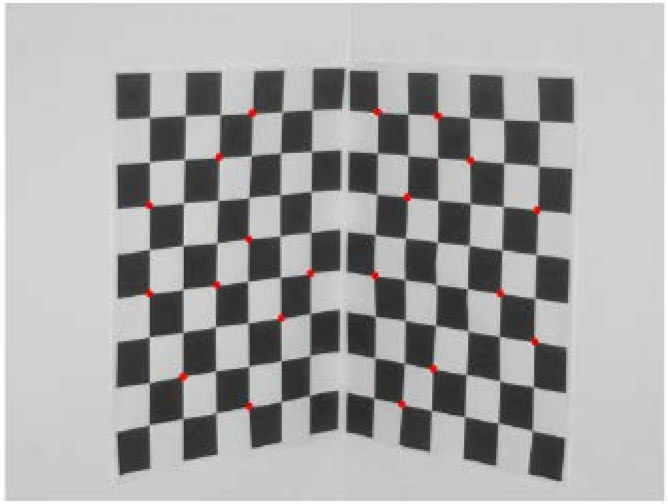
Note that there are two possible choices for the matrix $\mathcal{R}$ depending on the value of ε.

Slide Samer M Abdallah, Beirut

Estimation of the intrinsic and extrinsic parameters

The translation parameters can now be recovered by writing $\mathcal{K}\boldsymbol{t} = \rho\boldsymbol{b}$, and hence $\boldsymbol{t} = \rho\mathcal{K}^{-1}\boldsymbol{b}$. In practical situations, the sign of t_z is often known in advance (this corresponds to knowing whether the origin of the world coordinate system is in front or behind the camera), which allows the choice of a unique solution for the calibration parameters.

Typical Result



α	2816.8
β	2782.9
θ	1.5662
u_0	510.2025
v_0	318.3568

89.74°

Sensor:
1075x806,

$$\mathbf{R} = \begin{bmatrix} 0.6909 & 0.0101 & -0.7229 \\ 0.0744 & -0.9956 & 0.0572 \\ -0.7191 & -0.0933 & -0.6886 \end{bmatrix}$$

There is considerable rotation around the y axis (up) but the rotation around x and z was kept to a minimum.

$$r_x = -5.3457^\circ,\ r_y = 45.98^\circ,\ \text{and}\ r_z = 6.1478^\circ$$

$$\mathbf{t} = \begin{bmatrix} 1.2279 & 72.0368 & 135.7777 \end{bmatrix}^T$$

Camera was roughly 72 cm up from the floor, and 136 cm back from the pattern.

Degenerate Point Configurations

Are there other solutions besides *M* ??

$$\mathbf{0} = \mathcal{P}\boldsymbol{l} = \begin{pmatrix} \boldsymbol{P}_1^T & \mathbf{0}^T & -u_1\boldsymbol{P}_1^T \\ \mathbf{0}^T & \boldsymbol{P}_1^T & -v_1\boldsymbol{P}_1^T \\ \dots & \dots & \dots \\ \boldsymbol{P}_n^T & \mathbf{0}^T & -u_n\boldsymbol{P}_n^T \\ \mathbf{0}^T & \boldsymbol{P}_n^T & -v_n\boldsymbol{P}_n^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\nu} \end{pmatrix} = \begin{pmatrix} \boldsymbol{P}_1^T\boldsymbol{\lambda} - u_1\boldsymbol{P}_1^T\boldsymbol{\nu} \\ \boldsymbol{P}_1^T\boldsymbol{\mu} - v_1\boldsymbol{P}_1^T\boldsymbol{\nu} \\ \dots \\ \boldsymbol{P}_n^T\boldsymbol{\lambda} - u_n\boldsymbol{P}_n^T\boldsymbol{\nu} \\ \boldsymbol{P}_n^T\boldsymbol{\mu} - v_n\boldsymbol{P}_n^T\boldsymbol{\nu} \end{pmatrix}$$

$$\begin{cases} \boldsymbol{P}_i^T\boldsymbol{\lambda} - \dfrac{\boldsymbol{m}_1^T\boldsymbol{P}_i}{\boldsymbol{m}_3^T\boldsymbol{P}_i}\boldsymbol{P}_i^T\boldsymbol{\nu} = 0 \\ \boldsymbol{P}_i^T\boldsymbol{\mu} - \dfrac{\boldsymbol{m}_2^T\boldsymbol{P}_i}{\boldsymbol{m}_3^T\boldsymbol{P}_i}\boldsymbol{P}_i^T\boldsymbol{\nu} = 0 \end{cases} \Rightarrow \begin{cases} \boldsymbol{P}_i^T(\boldsymbol{\lambda}\boldsymbol{m}_3^T - \boldsymbol{m}_1\boldsymbol{\nu}^T)\boldsymbol{P}_i = 0 \\ \boldsymbol{P}_i^T(\boldsymbol{\mu}\boldsymbol{m}_3^T - \boldsymbol{m}_2\boldsymbol{\nu}^T)\boldsymbol{P}_i = 0 \end{cases}$$

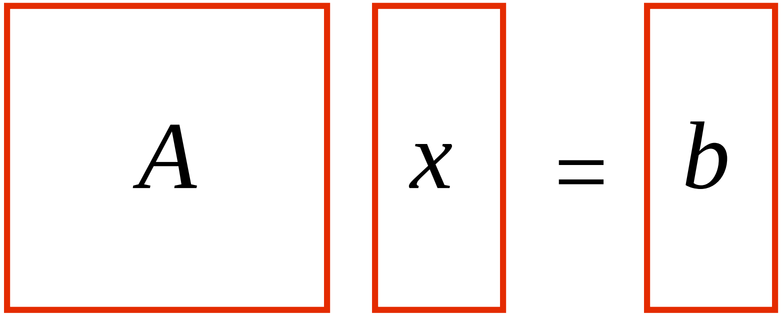
- Coplanar points: $(\lambda,\mu,\nu) = (\Pi,0,0)$ or $(0,\Pi,0)$ or $(0,0,\Pi)$
- Points lying on the intersection curve of two quadric surfaces = straight line + twisted cubic

Does not happen for 6 or more random points!

Other Slides following Forsyth&Ponce

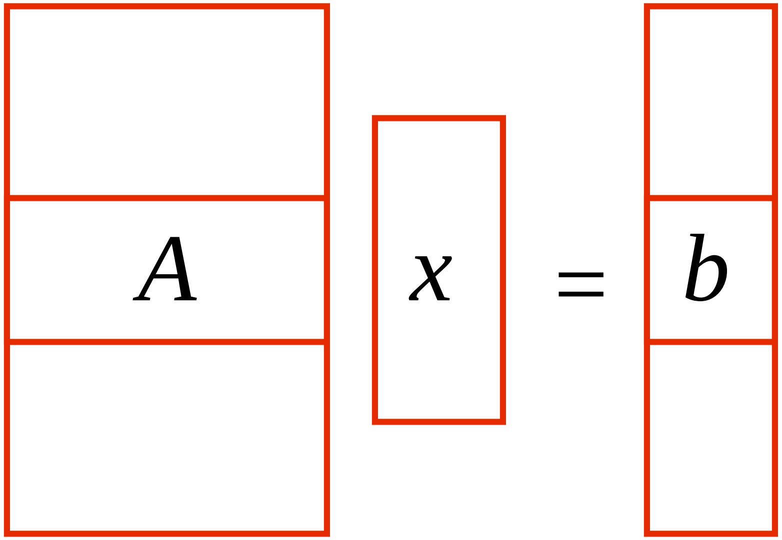
Just for additional information on previous slides

Linear Systems



Square system:

- unique solution
- Gaussian elimination



Rectangular system ??

- underconstrained:
  infinity of solutions
- overconstrained:
  no solution

Minimize $|Ax-b|^2$

How do you solve overconstrained linear equations ??

- Define $E = |\boldsymbol{e}|^2 = \boldsymbol{e} \cdot \boldsymbol{e}$ with

$$\boldsymbol{e} = A\boldsymbol{x} - \boldsymbol{b} = \left[\boldsymbol{c}_1 \middle| \boldsymbol{c}_2 \middle| \ldots \middle| \boldsymbol{c}_n \right] \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} - \boldsymbol{b}$$

$$= x_1\boldsymbol{c}_1 + x_2\boldsymbol{c}_2 + \cdots x_n\boldsymbol{c}_n - \boldsymbol{b}$$

- At a minimum,

$$\frac{\partial E}{\partial x_i} = \frac{\partial \boldsymbol{e}}{\partial x_i} \cdot \boldsymbol{e} + \boldsymbol{e} \cdot \frac{\partial \boldsymbol{e}}{\partial x_i} = 2\frac{\partial \boldsymbol{e}}{\partial x_i} \cdot \boldsymbol{e}$$

$$= 2\frac{\partial}{\partial x_i}(x_1\boldsymbol{c}_1 + \cdots + x_n\boldsymbol{c}_n - \boldsymbol{b}) \cdot \boldsymbol{e} = 2\boldsymbol{c}_i \cdot \boldsymbol{e}$$

$$= 2\boldsymbol{c}_i^T(A\boldsymbol{x} - \boldsymbol{b}) = 0$$

- or

$$0 = \begin{bmatrix} \boldsymbol{c}_i^T \\ \vdots \\ \boldsymbol{c}_n^T \end{bmatrix} (A\boldsymbol{x} - \boldsymbol{b}) = A^T(A\boldsymbol{x} - \boldsymbol{b}) \Rightarrow A^T A\boldsymbol{x} = A^T\boldsymbol{b},$$

where $\boldsymbol{x} = A^\dagger \boldsymbol{b}$ and $A^\dagger = (A^T A)^{-1} A^T$ is the *pseudoinverse* of A !

Homogeneous Linear Systems

$$A \, x = 0$$

Square system:

- unique solution: 0
- unless Det(A)=0

$$A \, x = 0$$

Rectangular system ??

- 0 is always a solution

Minimize $|Ax|^2$
under the constraint $|x|^2=1$

How do you solve overconstrained homogeneous linear equations ??

$$E = |\mathcal{U}\boldsymbol{x}|^2 = \boldsymbol{x}^T(\mathcal{U}^T\mathcal{U})\boldsymbol{x}$$

- Orthonormal basis of eigenvectors: $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_q$.
- Associated eigenvalues: $0 \leq \lambda_1 \leq \ldots \leq \lambda_q$.
- Any vector can be written as

$$\boldsymbol{x} = \mu_1\boldsymbol{e}_1 + \ldots + \mu_q\boldsymbol{e}_q$$

for some μ_i $(i = 1, \ldots, q)$ such that $\mu_1^2 + \ldots + \mu_q^2 = 1$.

$$\begin{aligned} E(\boldsymbol{x}) - E(\boldsymbol{e}_1) &= \boldsymbol{x}^T(\mathcal{U}^T\mathcal{U})\boldsymbol{x} - \boldsymbol{e}_1^T(\mathcal{U}^T\mathcal{U})\boldsymbol{e}_1 \\ &= \lambda_1^2\mu_1^2 + \ldots + \lambda_q^2\mu_q^2 - \lambda_1^2 \\ &\geq \lambda_1^2(\mu_1^2 + \ldots + \mu_q^2 - 1) = 0 \end{aligned}$$

The solution is e_1 .

remember: EIG(U^TU)=SVD(U), i.e. solution is V_n

Linear Camera Calibration

Given n points $P_1, \ldots, P_n$ with *known* positions and their images $p_1, \ldots, p_n$

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} \dfrac{\boldsymbol{m}_1 \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3 \cdot \boldsymbol{P}_i} \\ \dfrac{\boldsymbol{m}_2 \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3 \cdot \boldsymbol{P}_i} \end{pmatrix} \Longleftrightarrow \begin{pmatrix} \boldsymbol{m}_1 - u_i \boldsymbol{m}_3 \\ \boldsymbol{m}_2 - v_i \boldsymbol{m}_3 \end{pmatrix} \boldsymbol{P}_i = 0$$

$$\mathcal{P}\boldsymbol{m} = 0 \text{ with } \mathcal{P} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{P}_1^T & \mathbf{0}^T & -u_1 \boldsymbol{P}_1^T \\ \mathbf{0}^T & \boldsymbol{P}_1^T & -v_1 \boldsymbol{P}_1^T \\ \ldots & \ldots & \ldots \\ \boldsymbol{P}_n^T & \mathbf{0}^T & -u_n \boldsymbol{P}_n^T \\ \mathbf{0}^T & \boldsymbol{P}_n^T & -v_n \boldsymbol{P}_n^T \end{pmatrix} \text{ and } \boldsymbol{m} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{m}_1 \\ \boldsymbol{m}_2 \\ \boldsymbol{m}_3 \end{pmatrix} = 0$$

Useful Links

Demo calibration (some links broken):

- http://mitpress.mit.edu/e-journals/Videre/001/articles/Zhang/CalibEnv/CalibEnv.html

Bouget camera calibration SW:

- http://www.vision.caltech.edu/bouguetj/calib_doc/

CVonline: Monocular Camera calibration:

- http://homepages.inf.ed.ac.uk/cgi/rbf/CVONLINE/entries.pl?TAG250