Camera Calibration

Extrinsic camera calibration

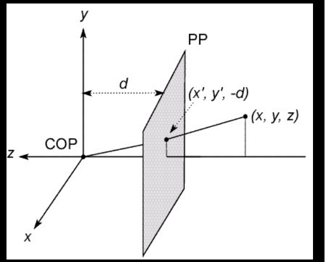
Recall: Modeling projection

Projection equations

- Compute intersection with Perspective Projection of ray from (x,y,z) to COP
- Derived using similar triangles

$$(X, Y, Z) \rightarrow (-d\frac{X}{Z}, -d\frac{Y}{Z}, -d)$$

(assumes normal Z negative – we'll change later)



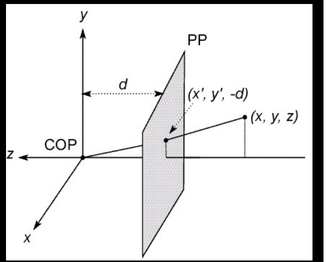
Recall: Modeling projection

Projection equations

$$(X, Y, Z) \rightarrow (-d\frac{X}{Z}, -d\frac{Y}{Z}, -d)$$

We get the projection by throwing out the last coordinate:

$$(x', y') = (-d\frac{X}{Z}, -d\frac{Y}{Z})$$



Recall: Homogeneous coordinates

Is this a linear transformation?

No – division by the (not constant) Z is non-linear

Trick: add one more coordinate:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image (2D) coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous scene (3D) coordinates

Recall: Homogeneous coordinates

Converting *from* homogeneous coordinates:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w) \qquad \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

(this makes homogenous coordinates invariant under scale)

Recall: Perspective Projection

- Projection is a matrix multiply using homogeneous coordinates (and |z|):

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \begin{bmatrix} \xi. \\ \psi. \\ |z| \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ |z/f| \end{bmatrix} \Rightarrow \left(f\frac{x}{|z|}, f\frac{y}{|z|} \right)$$

$$\Rightarrow (u, v)$$

S. Seitz

But…

- In all this discussion we have the notion of a camera's coordinate system – an origin and an orientation.
- We put the center of projection at this origin and the optic axis down the z axis.
- So to do geometric reasoning about the world we need to relate the coordinate system of the world to that of the camera and the image.
- Today we'll do from the world to the camera, and next time from the camera to the image.

Geometric Camera calibration

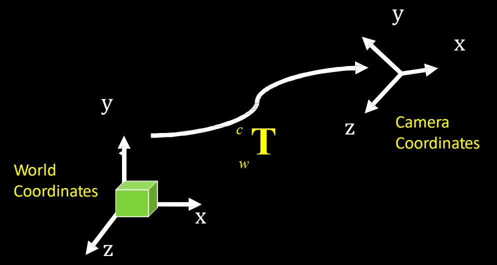
- We want to use the camera to tell us things about the world.
  - So we need the relationship between coordinates in the world and coordinates in the image: ***geometric camera calibration***
- For reference see Forsyth and Ponce, sections 1.2 and 1.3.

Geometric Camera calibration

Composed of 2 transformations:

- From some (arbitrary) world coordinate system to the camera's 3D coordinate system. *Extrinisic parameters (or camera pose)*
- From the 3D coordinates in the camera frame to the 2D image plane via projection. *Intrinisic parameters*

Camera Pose



Quiz

How many degrees of freedom are there in specifying the extrinsic parameters?

a) 5
b) 6
c) 3
d) 9

Rigid Body Transformations

Need a way to specify the six degrees-of-freedom of a rigid body. Why 6?

A rigid body is a collection of points whose positions relative to each other can't change

Fix one point, 3 DOF

3

Fix second point, 2 more DOF (must maintain distance constraint)

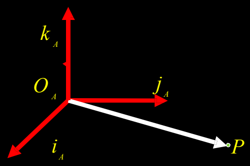
+2

Third point adds 1 more DOF, for rotation around line

+1

Notation (from F&P)

- Superscript references coordinate frame
- ${}^{A}P$ is coordinates of P in frame A
- ${}^{B}P$ is coordinates of P in frame B



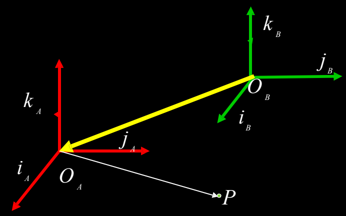
$${}^{A}P = \begin{pmatrix} {}^{A}x \\ {}^{A}y \\ {}^{A}z \end{pmatrix} \Leftrightarrow \overline{OP} = \left({}^{A}x \cdot \overline{i_A}\right) + \left({}^{A}y \cdot \overline{j_A}\right) + \left({}^{A}z \cdot \overline{k_A}\right)$$

Translation Only

$${}^{B}P = {}^{A}P + {}^{B}(O_A)$$

or

$${}^{B}P = {}^{B}(O_A) + {}^{A}P$$



Translation

- Using homogeneous coordinates, translation can be expressed as a matrix multiplication.

$${}^{B}P = {}^{A}P + {}^{B}O_A$$

3x3 identity

$$\begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} = \begin{bmatrix} I & {}^{B}O_A \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix}$$

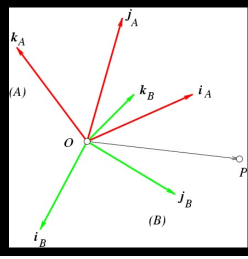
(Translation is commutative)

Rotation

$$\overline{OP} = \left(i_A \quad j_A \quad k_A\right) \begin{pmatrix} {}^{A}x \\ {}^{A}y \\ {}^{A}z \end{pmatrix} = \left(i_B \quad j_B \quad k_B\right) \begin{pmatrix} {}^{B}x \\ {}^{B}y \\ {}^{B}z \end{pmatrix}$$

$${}^{B}P = {}^{B}_{A}R \; {}^{A}P$$

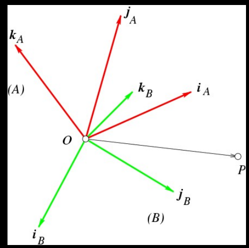
${}^{B}_{A}R$ means describing frame A in the coordinate system of frame B



Rotation

$$
{}^B_A R = \begin{bmatrix} \mathbf{i}_A \cdot \mathbf{i}_B & \mathbf{j}_A \cdot \mathbf{i}_B & \mathbf{k}_A \cdot \mathbf{i}_B \\ \mathbf{i}_A \cdot \mathbf{j}_B & \mathbf{j}_A \cdot \mathbf{j}_B & \mathbf{k}_A \cdot \mathbf{j}_B \\ \mathbf{i}_A \cdot \mathbf{k}_B & \mathbf{j}_A \cdot \mathbf{k}_B & \mathbf{k}_A \cdot \mathbf{k}_B \end{bmatrix}
$$

$$
= \begin{bmatrix} {}^B\mathbf{i}_A & {}^B\mathbf{j}_A & {}^B\mathbf{k}_A \end{bmatrix}
$$



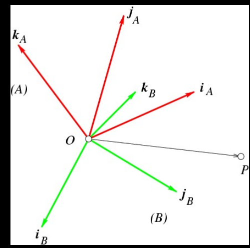
The columns of the rotation matrix are the axes of frame A expressed in frame B. Why?

Rotation

$$
{}^B_A R = \begin{bmatrix} \mathbf{i}_A \cdot \mathbf{i}_B & \mathbf{j}_A \cdot \mathbf{i}_B & \mathbf{k}_A \cdot \mathbf{i}_B \\ \mathbf{i}_A \cdot \mathbf{j}_B & \mathbf{j}_A \cdot \mathbf{j}_B & \mathbf{k}_A \cdot \mathbf{j}_B \\ \mathbf{i}_A \cdot \mathbf{k}_B & \mathbf{j}_A \cdot \mathbf{k}_B & \mathbf{k}_A \cdot \mathbf{k}_B \end{bmatrix}
$$

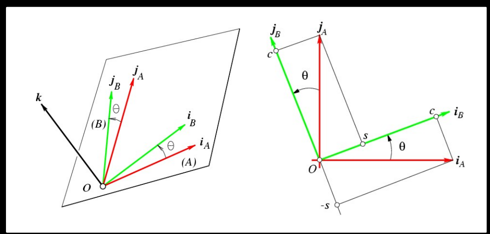
$$
= \begin{bmatrix} {}^B\mathbf{i}_A & {}^B\mathbf{j}_A & {}^B\mathbf{k}_A \end{bmatrix}
$$

$$
= \begin{bmatrix} {}^A\mathbf{i}_B{}^T \\ {}^A\mathbf{j}_B{}^T \\ {}^A\mathbf{k}_B{}^T \end{bmatrix}
$$



Orthogonal matrix! Why?

Example: Rotation about z axis



What is the rotation matrix?

$$
R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

Combine 3 to get arbitrary rotation

- Euler angles: Z, X', Z''
- Or heading, pitch roll: world Z, new X, new Y …
- Or roll, pitch and yaw …
- Or Azimuth, elevation, roll…
- Three basic matrices: order matters, but we'll not focus on that

Combine 3 to get arbitrary rotation

$$R_x(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi) & \cos(\varphi) \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_y(\kappa) = \begin{bmatrix} \cos(\kappa) & 0 & -\sin(\kappa) \\ 0 & 1 & 0 \\ \sin(\kappa) & 0 & \cos(\kappa) \end{bmatrix}$$

Rotation in homogeneous coordinates

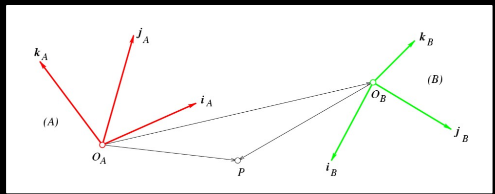
- Using homogeneous coordinates, rotation can be expressed as a matrix multiplication.

$$^{B}P = {}^{B}_{A}R \, {}^{A}P$$

$$\begin{bmatrix} ^{B}P \\ 1 \end{bmatrix} = \begin{bmatrix} ^{B}_{A}R & 0 \\ \mathbf{0}^{T} & 1 \end{bmatrix} \begin{bmatrix} ^{A}P \\ 1 \end{bmatrix}$$

- Rotation is ***not*** commutative

Rigid transformations



$$^{B}P = {}^{B}_{A}R \, {}^{A}P + {}^{B}O_{A}$$

Rigid transformations (con't)

Unified treatment using homogeneous coordinates:

$$\begin{bmatrix} ^{B}P \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & ^{B}O_{A} \\ \mathbf{0}^{T} & 1 \end{bmatrix} \begin{bmatrix} ^{B}_{A}R & 0 \\ \mathbf{0}^{T} & 1 \end{bmatrix} \begin{bmatrix} ^{A}P \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} ^{B}_{A}R & ^{B}O_{A} \\ \mathbf{0}^{T} & 1 \end{bmatrix} \begin{bmatrix} ^{A}P \\ 1 \end{bmatrix}$$

Rigid transformations (con't)

And even better:

$$\begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{B}_{A}R & {}^{B}O_{A} \\ \mathbf{0}^{T} & 1 \end{bmatrix} \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} = {}^{B}_{A}T \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix}$$

Invertible!

so

$$\begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} = {}^{A}_{B}T \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} = \left({}^{B}_{A}T\right)^{-1} \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix}$$

Translation and rotation

From frame A to B:
Non-homogeneous ("regular) coordinates

$${}^{B}\vec{p} = {}^{B}_{A}R \; {}^{A}\vec{p} + {}^{B}_{A}\vec{t}$$

3x1 translation vector

3x3 rotation matrix

Translation and rotation

Homogeneous coordinates:

$${}^{B}\vec{p} = {}^{B}_{A}T \; {}^{A}\vec{p}$$

$${}^{B}\vec{p} = \begin{pmatrix} \begin{pmatrix} & & \\ & {}^{B}_{A}R & \\ & & \end{pmatrix} & {}^{B}_{A}\vec{t} \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Homogenous coordinates allows us to write coordinate transforms as a single matrix!

From World to Camera

Rotation from world to camera frame

Translation from world to camera frame

$${}^{C}\vec{p} = {}^{C}_{W}R \; {}^{W}\vec{p} + {}^{C}_{W}\vec{t}$$

Point now in camera frame

Point in world frame

Non-homogeneous coordinates

From World to Camera

$$\begin{pmatrix} {}^{C}\vec{p} \\ 1 \end{pmatrix} = \begin{pmatrix} {}^{C}_{W}R & {}^{C}_{W}\vec{t} \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix} \begin{pmatrix} {}^{W}\vec{p} \\ 1 \end{pmatrix}$$

Homogeneous coordinates

From world to camera is the ***extrinsic*** *parameter matrix (4x4)*

(sometimes 3x4 if using for next step in projection – not worrying about inversion)

Quiz

How many degrees of freedom are there in the 3x4 extrinsic parameter matrix?

a) 12

b) 6

c) 9

d) 3

From World to Camera

$$\begin{pmatrix} {}^{C}\vec{p} \\ 1 \end{pmatrix} = \begin{pmatrix} {}^{C}_{W}R & {}^{C}_{W}\vec{t} \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix} \begin{pmatrix} {}^{W}\vec{p} \\ 1 \end{pmatrix}$$

Homogeneous coordinates

From world to camera is the ***extrinsic*** *parameter matrix (4x4)*

(sometimes 3x4 if using for next step in projection – not worrying about inversion)

Computer Vision

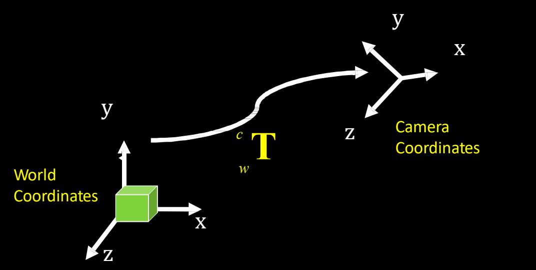
Intrinsic camera calibration

Geometric Camera calibration

Composed of 2 transformations:

- From some (arbitrary) world coordinate system to the camera's 3D coordinate system. *Extrinisic parameters (or camera pose)*

Camera Pose



From World to Camera

$$\begin{pmatrix} \\ {}^{c}\vec{p} \\ \\ \end{pmatrix} = \begin{pmatrix} {}^{c}_{w}R & {}^{c}_{w}\vec{t} \\ 0 \quad 0 \quad 0 & 1 \end{pmatrix} \begin{pmatrix} \\ {}^{w}\vec{p} \\ \\ \end{pmatrix}$$

Homogeneous coordinates

From world to camera is the ***extrinsic*** *parameter matrix (4x4)*

(sometimes 3x4 if using for next step in projection – not worrying about inversion)

Geometric Camera calibration

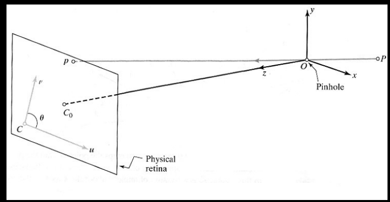
Composed of 2 transformations:

- From some (arbitrary) world coordinate system to the camera's 3D coordinate system. *Extrinisic parameters (or camera pose)*
- From the 3D coordinates in the camera frame to the 2D image plane via projection. *Intrinisic parameters*

Ideal intrinsic parameters

Ideal Perspective projection:

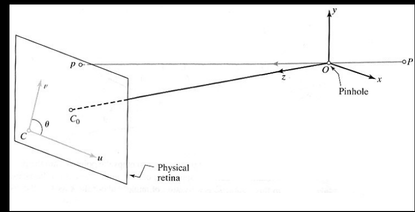
$$u = f\,\frac{x}{z}$$
$$v = f\,\frac{y}{z}$$



Real intrinsic parameters (1)

But "pixels" are in some arbitrary spatial units

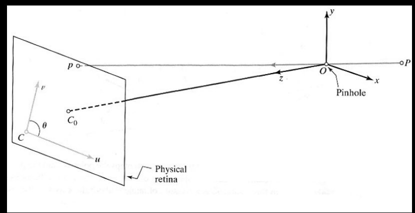
$$u = \alpha\,\frac{x}{z}$$
$$v = \alpha\,\frac{y}{z}$$



Real intrinsic parameters (2)

Maybe pixels are not square

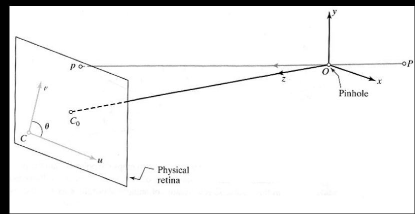
$$u = \alpha\,\frac{x}{z}$$
$$v = \beta\,\frac{y}{z}$$



Real intrinsic parameters (3)

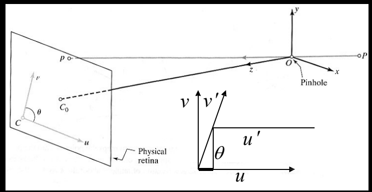
We don't know the origin of our camera pixel coordinates

$$u = \alpha\,\frac{x}{z} + u_0$$
$$v = \beta\,\frac{y}{z} + v_0$$



Really ugly intrinsic parameters (4)

May be skew between camera pixel axes



$$v'\sin(\theta) = v$$

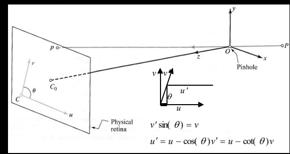
$$u' = u - \cos(\theta)v' = u - \cot(\theta)v$$

Really ugly intrinsic parameters (4)

May be skew between camera pixel axes

$$u = \alpha \frac{x}{z} - \alpha\cot(\theta)\frac{y}{z} + u_0$$

$$v = \frac{\beta}{\sin(\theta)}\frac{y}{z} + v_0$$



Intrinsic parameters, non-homogeneous coords

Notice division by z

$$u = \alpha \frac{x}{z} - \alpha\cot(\theta)\frac{y}{z} + u_0$$

$$v = \frac{\beta}{\sin(\theta)}\frac{y}{z} + v_0$$

Intrinsic parameters, *homogeneous* coords

$$\begin{pmatrix} z*u \\ z*v \\ z \end{pmatrix} = \begin{pmatrix} \alpha & -\alpha\cot(\theta) & u_0 & 0 \\ 0 & \frac{\beta}{\sin(\theta)} & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

$$\vec{p}' = K \; {}^{c}\vec{p}$$

In homogeneous pixels

Intrinsic matrix

In camera-based 3D coords

Kinder, gentler intrinsics

- Can use simpler notation for intrinsics – remove last column which is zero:

$$K = \begin{bmatrix} f & s & c_x \\ 0 & af & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

f –focal length
s – skew
a – aspect ratio
c_x, c_y - offset
(5 DOF)

Kinder, gentler intrinsics

- If square pixels, no skew, and optical center is in the center (assume origin in the middle):

$$K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In this case only one DOF, focal length ***f***

Kinder, gentler intrinsics

- Can use simpler notation for intrinsics – remove last column which is zero:

$$K = \begin{bmatrix} f & s & c_x \\ 0 & af & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

f –focal length
s – skew
a – aspect ratio
c_x, c_y - offset
(5 DOF)

Combining extrinsic and intrinsic calibration parameters

Pixels

$$\vec{p}\,' = K \; {}^{C}\vec{p}$$

Intrinsic

Camera 3D coordinates

$${}^{C}\vec{p} = \begin{pmatrix} - & - & - & | \\ - & {}^{C}_{W}R & - & {}^{C}_{W}\vec{t} \\ - & - & - & | \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \\ {}^{W}\vec{p} \\ \\ \end{pmatrix}$$

World 3D coordinates

Extrinsic

Combining extrinsic and intrinsic calibration parameters

$$\vec{p}\,' = \underbrace{K}_{\boxed{\text{K } 3\text{x}3}} \underbrace{\left({}^{C}_{W}R \quad {}^{C}_{W}\vec{t} \right)}_{\boxed{3\text{x}4}} {}^{W}\vec{p}$$

$$\vec{p}\,' = M \; {}^{W}\vec{p}$$

Other ways to write the same equation

pixel coordinates

world coordinates

Conversion back from homogeneous coordinates

$$\vec{p}\,' = M \; {}^{W}\vec{p}$$

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

$$u = \frac{m_1 \cdot \vec{P}}{m_3 \cdot \vec{P}} \qquad v = \frac{m_2 \cdot \vec{P}}{m_3 \cdot \vec{P}}$$

projectively similar

Finally: Camera parameters

- A camera (and its matrix) M (or Π) is described by several parameters
  - Translation **T** of the optical center from the origin of world coordinates
  - Rotation **R** of the camera system
  - focal length and aspect **(f, a)** [or pixel size **(s_x, s_y)**] , principle point **(x'_c, y'_c)**, and skew **(s)**
  - **blue** parameters are called "**extrinsics**," red are "intrinsics"

Finally: Camera parameters

- Projection equation – the cumulative effect of all parameters:

$$\mathbf{x} \cong \begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{M}\,\mathbf{X}$$

Finally: Camera parameters

- Projection equation – the cumulative effect of all parameters:

$$\underset{(3\text{x}4)}{\mathbf{M}} = \begin{bmatrix} f & s & x'_c \\ 0 & af & y'_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{0}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{T}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}$$

intrinsics projection rotation translation

$\boxed{DoFs:\ 5+0+3+3 = 11}$

Calibrating cameras

Finally (last time): Camera parameters

- Projection equation – the cumulative effect of all parameters:

$$\underset{(3\text{x}4)}{\mathbf{M}} = \begin{bmatrix} f & s & x'_c \\ 0 & af & y'_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{0}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{T}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}$$

intrinsics projection rotation translation

Finally (last time): Camera parameters

- Projection equation – the cumulative effect of all parameters:

$$\mathbf{x} = \begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{M}\,\mathbf{X}$$

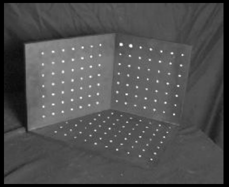
Calibration

- How to determine M?

Calibration using known points

Place a known object in the scene

- identify correspondence between image and scene
- compute mapping from scene to image



Resectioning

Estimating the camera matrix from known 3D points

Projective Camera Matrix:

$$p = K\begin{bmatrix} R & t \end{bmatrix} P = MP$$

$$\begin{bmatrix} w*u \\ w*v \\ w \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Direct linear calibration - homogeneous

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} w*u_i \\ w*v_i \\ w \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

One pair of equations for each point

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

Direct linear calibration - homogeneous

One pair of equations for each point

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}$$

$$v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}$$

Direct linear calibration - homogeneous

One pair of equations for each point

$$u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}$$

$$v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}$$

$$\begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -u_iX_i & -u_iY_i & -u_iZ_i & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -v_iX_i & -v_iY_i & -v_iZ_i & -v_i \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Direct linear calibration - homogeneous

- This is a homogenous set of equations.
- When over constrained, defines a least squares problem – minimize $\|\mathbf{A}\,\mathbf{m}\|$
- Since m is only defined up to scale, solve for unit vector m*
  - Solution: m* = eigenvector of A^TA with smallest eigenvalue
  - Works with 6 or more points

Direct linear calibration - homogeneous

$$\underbrace{\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 & -u_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 & -v_1 \\ & & & & & & \vdots & & & & & \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -u_nX_n & -u_nY_n & -u_nZ_n & -u_n \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -v_nX_n & -v_nY_n & -v_nZ_n & -v_n \end{bmatrix}}_{\substack{\mathbf{A} \\ 2n \times 12}} \underbrace{\begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \\ m_{23} \end{bmatrix}}_{\substack{\mathbf{m} \\ 12}} = \underbrace{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}}_{\substack{\mathbf{0} \\ 2n}}$$

The SVD (singular value decomposition) trick…

- Find the $\mathbf{m}$ that minimizes $||A\mathbf{m}||$subject to $||\mathbf{m}||$=1.
- Let $A = UDV^T$ (singular value decomposition, D diagonal, U and V orthogonal)
- Therefor minimizing $||UDV^T\mathbf{m}||$
- But, $||UDV^T\mathbf{m}||$ = $||DV^T\mathbf{m}||$ and $||\mathbf{m}||$=$||V^T\mathbf{m}||$
- Thus minimize $||DV^T\mathbf{m}||$ subject to $||V^T\mathbf{m}|| = \mathbf{1}$

The SVD (singular value decomposition) trick…

- Thus minimize $||DV^T\mathbf{m}||$ subject to $||V^T\mathbf{m}|| = \mathbf{1}$
- Let $\mathbf{y} = V^T\mathbf{m}$ Now minimize $||D\mathbf{y}||$ subject to$||\mathbf{y}|| = 1$.
- But D is diagonal, with decreasing values. So $||D\mathbf{y}||$ minimum is when $\mathbf{y} = (0,0,0 \ldots, 0,1)^T$
- Since $\mathbf{y} = V^T\mathbf{m}$, $\mathbf{m} = V\mathbf{y}$ since V orthogonal
- Thus $\mathbf{m} = V\mathbf{y}$ is the last column in V.

The SVD (singular value decomposition) trick…

- Thus $\mathbf{m} = \mathrm{V}\mathbf{y}$ is the last column in V.
- And, the singular values of A are square roots of the eigenvalues of A^TA and the columns of V are the eigenvectors. (Show this? Nah…)
- Recap: Given Am=0, find the eigenvector of A^TA with smallest eigenvalue, that's m.

Direct linear calibration (transformation)

Advantages:

- Very simple to formulate and solve. Can be done, say, on a problem set
- These methods are referred to as "algebraic error" minimization.

Direct linear calibration (transformation)

Disadvantages:

- Doesn't directly tell you the camera parameters (more in a bit)
- Approximate: e.g. doesn't model radial distortion
- Hard to impose constraints (e.g., known focal length)
- *Mostly: Doesn't minimize the right error function*

Direct linear calibration (transformation)

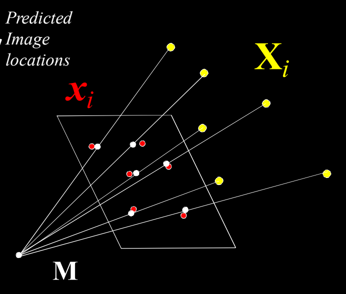
For these reasons, prefer nonlinear methods:

- Define error function *E* between projected 3D points and image positions:
  E is nonlinear function of *intrinsics, extrinsics, and radial distortion*
- Minimize E using nonlinear optimization techniques e.g., variants of Newton's method (e.g., Levenberg Marquart)

Geometric Error

$$\text{minimize } E = \sum_i d(x'_i, x'_i)$$

$$\min_{\mathbf{M}} \sum_i d(x'_i, \mathbf{M}\mathbf{X}_i)$$



"Gold Standard" algorithm (Hartley and Zisserman)

Objective

Given n≥6 3D to 2D point correspondences $\{X_i \leftrightarrow x_i'\}$, determine the "Maximum Likelihood Estimation" of **M**

“Gold Standard” algorithm (Hartley and Zisserman)

Algorithm

(i) Linear solution:

(a) (Optional) Normalization: $\tilde{\mathbf{X}}_i = \mathbf{U}\mathbf{X}_i$ $\tilde{\mathbf{x}}_i = \mathbf{T}\mathbf{x}_i$

(b) Direct Linear Transformation minimization

(ii) Minimize geometric error: using the linear estimate as a starting point minimize the geometric error:

$$\min_{\mathbf{M}} \sum_i d(\tilde{\mathbf{x}}_i, \tilde{\mathbf{M}}\tilde{\mathbf{X}}_i)$$

“Gold Standard” algorithm (Hartley and Zisserman)

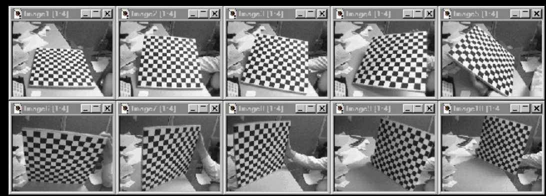
(iii) Denormalization: $\mathbf{M} = \mathbf{T}^{-1}\tilde{\mathbf{M}}\mathbf{U}$

Finding the 3D Camera Center from M

- Now the easy way. A formula! If M =[Q|b] then:

$$\mathbf{C} = \begin{pmatrix} -\mathbf{Q}^{-1}\mathbf{b} \\ 1 \end{pmatrix}$$

Alternative: multi-plane calibration



Images courtesy Jean-Yves Bouguet, Intel Corp.

Alternative: multi-plane calibration

Advantages

- Only requires a plane
- Don't have to know positions/orientations
- Good code available online!
  - OpenCV library
  - Matlab version by Jean-Yves Bouget: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html
  - Zhengyou Zhang's web site: http://research.microsoft.com/~zhang/Calib/