Camera parameters (§2.2)

- Camera might not be at the origin, looking down the z-axis
  - extrinsic parameters (position and orientation of the camera)
- Units in camera coordinates are not the same as units in world coordinates
  - intrinsic parameters - focal length, principal point, aspect ratio, angle between axes.

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} \text{Transformation} \\ \text{representing} \\ \text{intrinsic parameters} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \text{Transformation} \\ \text{representing} \\ \text{extrinsic parameters} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Notes:
We set f=1, and push the actual value of f into the intrinsic parameter matrix (see §2.2.1)

Actual pixel coords are (u,v) = (U/W, V/W)

Camera parameters (§2.2)

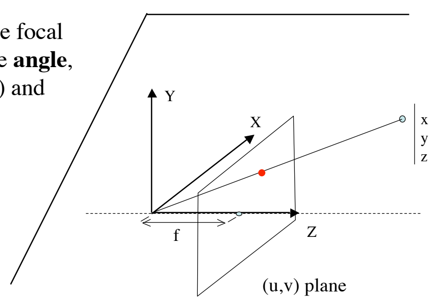
- Extrinsic parameters
  - position of the camera (3)
  - and orientation of the camera (3)
- Intrinsic parameters
  - focal length (1)
  - aspect ratio (ratio of pixel horizontal size to vertical size) (1)
  - principal point (intersection of viewing direction with camera plane) (2)
  - angle between axes of image plane---usually very close to 90 degrees (1)

Intrinsic parameters (focal length)

Recall that u = f * (x/z) and v = f * (y/z)

The natural, easy to measure, units for (u,v) are pixels.

This means that the focal length transfers the **angle**, as encoded in (x/z) and (y/z) into **pixels**.



Intrinsic parameters (focal length)

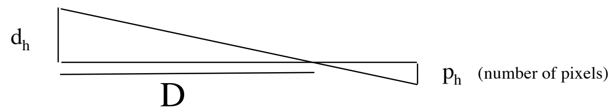
To transform a focal length in meters to pixels you would need to know the size of a pixel in meters. But you can easily measure the focal length in pixels (which is usually what you want)*.

However pixels are not always square. The ratio of width to height is called the aspect ratio.

Hence it is common to instead use two other parameters giving the horizontal and vertical focal lengths, α and β, in pixel units

* If you have the focal length both in pixels and in meters, then you can compute the size of a pixel (if you wanted it for some reason)

Measuring focal lengths



$$\alpha = D\frac{p_h}{d_h}$$

Intrinsic parameter matrix

Conversion of projected coords into pixel units is achieved by simple **scalings** based on α and β.

Translate coords to so that line through pinhole (or center of lens) perpendicular to projection plane is at origin.

Compensate for non-perpendicular pixel axis (if needed, usually this is OK) using a **shear** transformation.

Since these operations are all achievable with matrices, we see that a camera can be modeled with M composed of the three parts (intrinsic, projection, extrinsic).

Camera calibration (§3)

- Want to find out:
  - what is the camera matrix? (intrinsic+extrinsic)
  - what are intrinsic parameters of the camera?
- General strategy:
  - view calibration object
  - identify image points
  - obtain camera matrix by minimizing error
  - obtain intrinsic parameters from camera matrix
- Error minimization:
  - Linear least squares
    - easy problem numerically
    - solution can be rather bad
  - More robust methods exist, including ones that don't require a calibration object

Typical setup for calibration

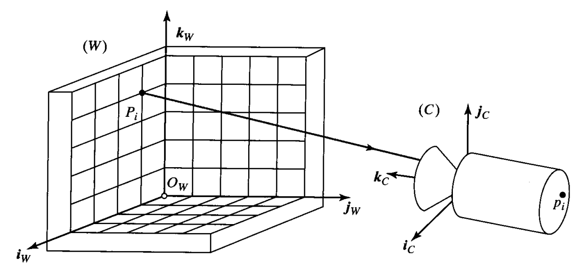


Figure 3.1 Camera calibration setup: In this example, the calibration rig is formed by three grids drawn in orthogonal planes. Other patterns could be used as well, and they may involve lines or other geometric figures.

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \underbrace{\begin{pmatrix} \text{Transformation} \\ \text{representing} \\ \text{intrinsic parameters} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \text{Transformation} \\ \text{representing} \\ \text{extrinsic parameters} \end{pmatrix}}_{\text{Camera matrix, } M} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Goal one: find M from image of calibration object

Goal two: given M, find the two matrices

Is goal one feasible?

More specifically, given (X,Y,Z) and corresponding u=(U/W) and v=(V/W), can we compute M?

First observation---if M is a solution then, because of homogeneity, k*M is an equivalent solution.

Thus, it only makes sense to recover M up to a scaling constant, and we can set the scale of M in advance for our convenience.

Is goal two feasible?

Reason by counting parameters.

We have 11 numbers, as M is 3 by 4, and we can fix the scale.

The number of parameters (degrees of freedom) are the number of intrinsic parameters *plus* the number of extrinsic parameters.

Extrinsic parameters: ?

Intrinsic parameters: ?

The number of parameters are the number of intrinsic parameters *plus* the number of extrinsic parameters.

Extrinsic parameters:

location	(3)
orientation	(3)

Intrinsic parameters:

focal length	(1)	Or α and β.
pixel aspect ratio	(1)	
principal point	(2)	
skew	(1)	
	11	

Often assume skew is zero

Finding M (goal one) (§3)

Find M from an image of calibration object. The equation relating world coordinates to image coordinates is:

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = M \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = MP$$

If we identify enough non-degenerate points whose *world coordinates are known* then we can estimate M from their *location in the image*.

Specifically we have points in space, P, and corresponding observed image coordinates, u=U/W and v=V/W

(§2.2.2, §3.2.1)

We have $\begin{pmatrix} U \\ V \\ W \end{pmatrix} = M\mathbf{P}$

Write $M = \begin{vmatrix} \mathbf{m_1} \\ \mathbf{m_2} \\ \mathbf{m_3} \end{vmatrix}$ Where $\mathbf{m}_i$ are row vectors

Thus
$$U = \mathbf{m_1} \cdot \mathbf{P}$$
$$V = \mathbf{m_2} \cdot \mathbf{P}$$
$$W = \mathbf{m_3} \cdot \mathbf{P}$$

(§2.2.2, §3.2.1)

From the previous slide

$$U = \mathbf{m_1} \cdot \mathbf{P}$$
$$V = \mathbf{m_2} \cdot \mathbf{P}$$
$$W = \mathbf{m_3} \cdot \mathbf{P}$$

So **each** point, **i**, gives two equations (§2.2.2, §3.2.1)

$$u_i = \frac{\mathbf{m}_1 \cdot \mathbf{P}_i}{\mathbf{m_3} \cdot \mathbf{P}_i} \qquad v_i = \frac{\mathbf{m}_2 \cdot \mathbf{P}_i}{\mathbf{m_3} \cdot \mathbf{P}_i}$$

Which become

$$(\mathbf{m_1} - u_i \mathbf{m_3}) \cdot \mathbf{P}_i = 0$$
$$(\mathbf{m}_2 - v_i \mathbf{m_3}) \cdot \mathbf{P}_i = 0$$

(§2.2.2, §3.2.1)

We have linear equations for the components of M

The components of the matrix M are the *variables* in linear equations

Represent M by a vector $\mathbf{m} = \begin{pmatrix} \mathbf{m}_1^T \\ \mathbf{m}_2^T \\ \mathbf{m}_3^T \end{pmatrix}$

Note that our camera matrix, M, is the unknown so we want to make it a vector in some matrix equation (where something else is going to be the matrix)---standard thing to do.

(§2.2.2, §3.2.1)

We are representing the matrix M by a vector $\mathbf{m} = \begin{pmatrix} \mathbf{m}_1^{\mathbf{T}} \\ \mathbf{m}_2^{\mathbf{T}} \\ \mathbf{m}_3^{\mathbf{T}} \end{pmatrix}$

Now rewrite $(\mathbf{m_1} - u_i \mathbf{m_3}) \cdot \mathbf{P}_i = 0$ as $\boxed{\begin{pmatrix} \mathbf{P}_i^T & 0 & -u_i \mathbf{P}_i^T \end{pmatrix}} \mathbf{m} = 0$

$(\mathbf{m}_2 - v_i \mathbf{m_3}) \cdot \mathbf{P}_i = 0$ as $\boxed{\begin{pmatrix} 0 & \mathbf{P}_i^T & -v_i \mathbf{P}_i^T \end{pmatrix}} \mathbf{m} = 0$

Thus every point leads to two rows of a matrix P.

(§2.2.2, §3.2.1)

From previous slide, each point gives two rows of a matrix P

$$\begin{pmatrix} \mathbf{P}_i^T & 0 & -u_i \mathbf{P}_i^T \end{pmatrix} \mathbf{m} = 0$$
$$\begin{pmatrix} 0 & \mathbf{P}_i^T & -v_i \mathbf{P}_i^T \end{pmatrix} \mathbf{m} = 0$$

So, in general, the 2n by 12 matrix P is:

$$\begin{pmatrix} \mathbf{P}_1^T & & -u_1 \mathbf{P}_1^T \\ & \mathbf{P}_1^T & -v_1 \mathbf{P}_1^T \\ \cdots & \cdots & \cdots \\ \mathbf{P}_i^T & & -u_i \mathbf{P}_i^T \\ & \mathbf{P}_i^T & -v_i \mathbf{P}_i^T \\ \cdots & \cdots & \cdots \\ \mathbf{P}_n^T & & -u_n \mathbf{P}_n^T \\ & \mathbf{P}_n^T & -v_n \mathbf{P}_n^T \end{pmatrix}$$

(§2.2.2, §3.2.1)

So, we want to solve P**m**=**0** for **m**, where P is 2n by 12

This problem is a bit tricky

Clearly **m**=**0** is a solution (degenerate solution)

There must be another solution (if we believe our imaging model)

If **m** is a solution, then a scalar multiple of **m** is also (homogeneity)

So, we solve P**m**=**0** under the constraint that |**m**|=1

If n>6, then this typically will not have a solution due to error (over-constrained)

To simultaneously deal with this problem, AND to use the information from multiple points, we find a "best" solution, using more than 6 points.

Math aside, #4

Homogenous linear least squares

Recall the second problem inspired by our camera calibration problem:

Solve $U\mathbf{x} = \mathbf{0}$ subject to $|\mathbf{x}| = 1$

We will sketch the solution briefly

(still §3.1.1)

Homogenous linear least squares

Because we solve $U\mathbf{x} = \mathbf{0}$ as best we can, the error vector is $U\mathbf{x}$

The squared error is then

$$(U\mathbf{x})^{\mathbf{T}}(U\mathbf{x}) = \mathbf{x}^{\mathbf{T}}\left(U^{\mathbf{T}}U\right)\mathbf{x}$$

Since $U^{\mathbf{T}}U$ is symmetric it has an eigenvalue decomposition (diagonalization) with real eigenvalues

Recall that a matrix A has an eigenvector, $\mathbf{e}$, with eigenvalue λ if $\mathrm{A}\mathbf{e} = \lambda\mathbf{e}$

Diagonalization: $U^{\mathbf{T}}U = V\Lambda V^{T}$ where V is an orthonormal basis made of the eigenvectors, $\mathbf{e}_i$, of $U^{\mathbf{T}}U$, and Λ is a diagonal matrix of the eigenvalues

Homogenous linear least squares

Critically, since $\mathrm{U}^{\mathrm{T}}\mathrm{U}$ it is positive semidefinite, the eigenvalues are **non – negative**

Recall that a matrix A is positive semidefinite if $\mathbf{x}^{\mathrm{T}}\mathrm{A}\mathbf{x}$ is never negative.

(Clearly $\mathrm{U}^{\mathrm{T}}\mathrm{U}$ it is positive semidefinite because $\mathbf{x}\mathrm{U}^{\mathrm{T}}\mathrm{U}\mathbf{x}$ is $|\mathrm{U}\mathbf{x}|^2$

Further technical comments

If the model is good, then U will **approximate** a matrix of deficient column rank because there should exist a non-zero **x** that solves U**x**=**0**.

We force the solution process to embody the assumption that the fact that $\mathrm{U}^{\mathrm{T}}\mathrm{U}$ *appears* to be of full rank is due to measurement error. This assumption helps separate the part of U that is due to errors from the part that is due to the model.

This why we say that $\mathrm{U}^{\mathrm{T}}\mathrm{U}$ is **semi-**positive definite, and *not* positive definite. We assume that there is a solution to Ux=0, that is distinctly non-zero .

(A matrix, A, is positive definite if, $\mathbf{x}^{\mathrm{T}}\mathrm{A}\mathbf{x}$ is never negative, and $\mathbf{x}^{\mathrm{T}}\mathrm{A}\mathbf{x} = \mathbf{0}$ means that $\mathbf{x}=\mathbf{0}$.)

Homogenous linear least squares

Since $\mathrm{U}^{\mathrm{T}}\mathrm{U}$ it is positive semidefinite, the eigenvalues are **non – negative**

(From two slides back)

We will write them as $\lambda_i = \sigma_i^2$.

Note: The book (at least my copy) uses λ_i^2 in the equation at the top of page 41 which is confusing. The coefficients, normally denoted, λ_i are in fact equal to the **square** of the "singular values of U", which usually are denoted by σ_i

Homogenous linear least squares

We can write $\mathbf{x}$ in terms of the eigenvector basis :

$\mathbf{x} = \sum u_{\mathrm{i}} \mathbf{e}_{\mathrm{i}}$ where $\sum u_i^2 = 1$ (why?)

Homogenous linear least squares

The error is $\mathbf{x}^{\mathbf{T}} \left(V \Lambda V^T \right) \mathbf{x} = \left(\mathbf{x}^{\mathbf{T}} V \right) \Lambda \left(V^T \mathbf{x} \right)$

What is $V^T \mathbf{x}$?

Homogenous linear least squares

The error is $\mathbf{x}^{\mathbf{T}} \left(V \Lambda V^T \right) \mathbf{x} = \left(\mathbf{x}^{\mathbf{T}} V \right) \Lambda \left(V^T \mathbf{x} \right)$

The elements of $V^T \mathbf{x}$ are u_{i}

So the error is ?

Homogenous linear least squares

From the previous slide the error to be minized is $\sum u_i^2 \sigma_i^2$

We are stuck with the values σ_i^2 and $\sum u_i^2 = 1$

To make the error small, what can we do?

Homogenous linear least squares

The best we can do is to set $u_i = 1$
for the minimum value of $\lambda_i = \sigma_i^2$ and zero for the others.

$$\mathbf{x} = \sum u_i \mathbf{e}_i$$

Set $u_{i*} = 1$, all other $u_i = 0$

So, $x^* = e_{i*}$, where $\lambda_{i^*} =$ is minimum

Thus the minimum is reached when x is set to the eigenvector corresponding to the minimum eigenvalue of $U^T U$

Homogenous linear least squares

Example 3.1 in book

(fitting a line to points, a better way for many applications)

Key initial point: The perpendicular distance from a point x_i, to a line ax+by=d is given by:

$d_i = d - ax_i - by_i$

Line Fitting (continued)

$$E = \sum d_i^2 = \sum \left(d - ax_i - by_i\right)^2$$

$$\frac{\partial E}{\partial d} = -2\sum \left(d - ax_i - by_i\right) = 0$$

So, $d = a\overline{x} + b\overline{y}$

Line Fitting (continued)

$d = a\overline{x} + b\overline{y}$ (from previous slide)

$$E = \sum \left(a\overline{x} - ax_i + b\overline{y} - by_i\right)^2$$

$$= \sum \left(\left(\overline{x} - x_i, \overline{y} - y_i\right) \bullet (a,b)\right)^2$$

$$= \left|U\mathbf{n}\right|^2, \text{ where } U = \begin{pmatrix} \overline{x} - x_1 & \overline{y} - y_1 \\ \ldots & \ldots \\ \overline{x} - x_n & \overline{y} - y_n \end{pmatrix}$$

So, we solve $U\mathbf{n}=0$ in the least squares sense, with $a^2 + b^2 = 1$

Back to cameras (§3.2.1)

Goal one: Find the matrix M linking world coordinates to image coordinates from image of calibration object.

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = M \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = MP$$

Recall, that since the above is in terms of homogeneous coordinates we have to work in terms of the observed image coordinates, u=U/W and v=V/W

Recall that we form column vectors from the rows of M and stack the columns on top of one another to get the vector of unknowns, **m**.

Recall that we derived the following equation for **m**, to be solved subject to |**m**|=1 in the least squares sense.

$$\begin{pmatrix} \mathbf{P}_1^T & & -u_1\mathbf{P}_1^T \\ & \mathbf{P}_1^T & -v_1\mathbf{P}_1^T \\ \cdots & \cdots & \cdots \\ \mathbf{P}_i^T & & -u_i\mathbf{P}_i^T \\ & \mathbf{P}_i^T & -v_i\mathbf{P}_i^T \\ \cdots & \cdots & \cdots \\ \mathbf{P}_n^T & & -u_n\mathbf{P}_n^T \\ & \mathbf{P}_n^T & -v_n\mathbf{P}_n^T \end{pmatrix} \begin{pmatrix} \mathbf{m}_1^T \\ \mathbf{m}_2^T \\ \mathbf{m}_3^T \end{pmatrix} = \mathbf{0}$$

So, now we can simply apply the eigenvalue method in the previous slides to solve for **m**.

Intrinsic/extrinsic parameters

Recall goal two: Given M, recover the intrinsic parameters.

See §3.2.2 for the development of some formulas. Grad students will use a simplified version of them in assignment three (relatively straight forward, but a bit complex)