

Computer Vision, Lecture 9

Professor Hager

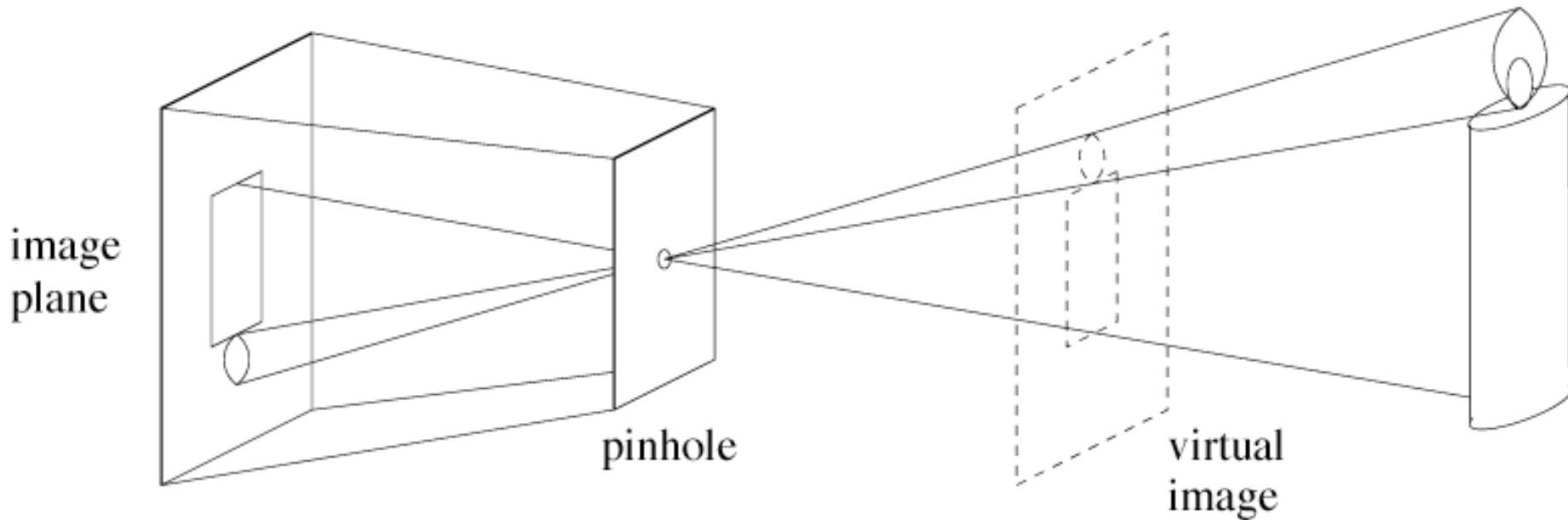
<http://www.cs.jhu.edu/~hager>

Outline for Today

- Geometric Image Formation
- Camera Models

Pinhole cameras

- Abstract camera model - box with a small hole in it
- Pinhole cameras work in practice



Real Pinhole Cameras

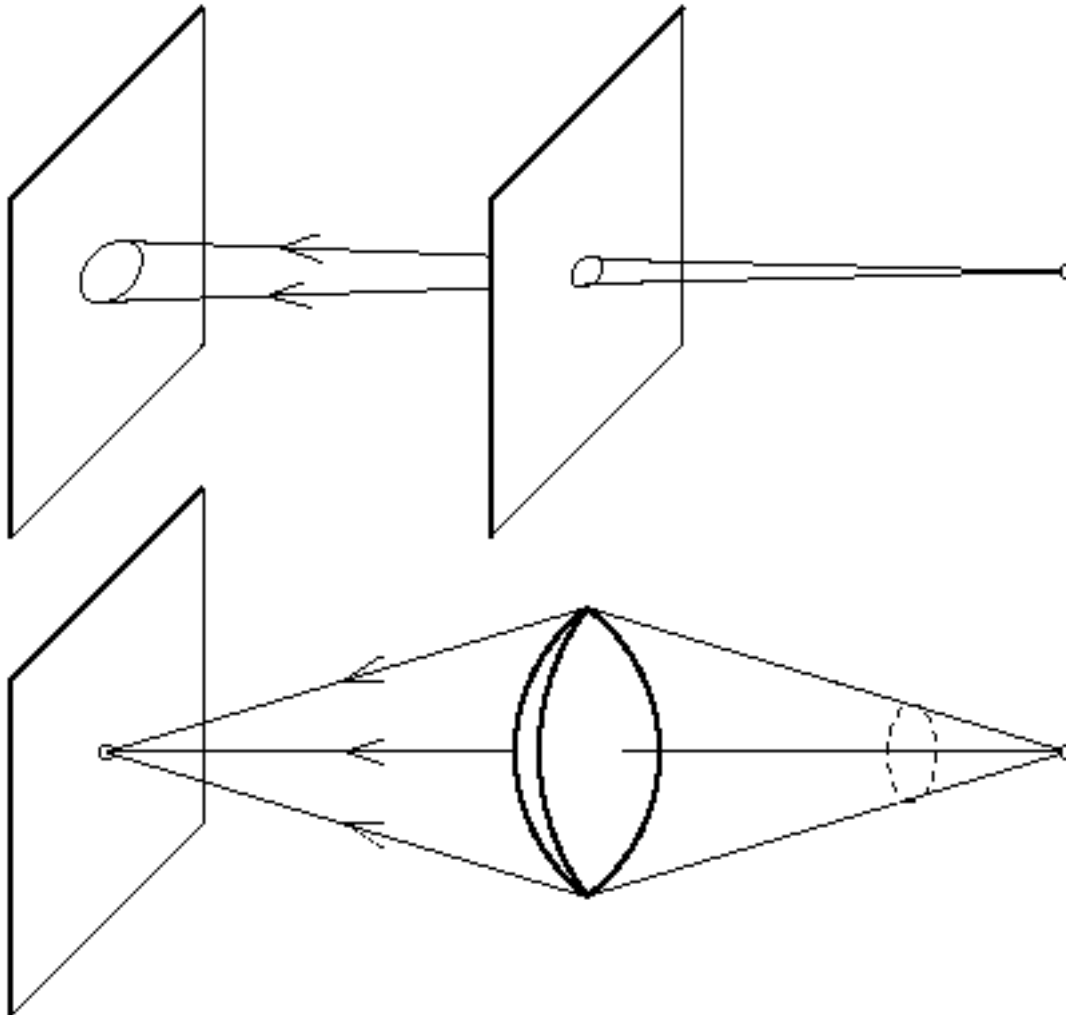
Pinhole too big -
many directions are
averaged, blurring the
image

Pinhole too small -
diffraction effects blur
the image

Generally, pinhole
cameras are *dark*, because
a very small set of rays
from a particular point
hits the screen.

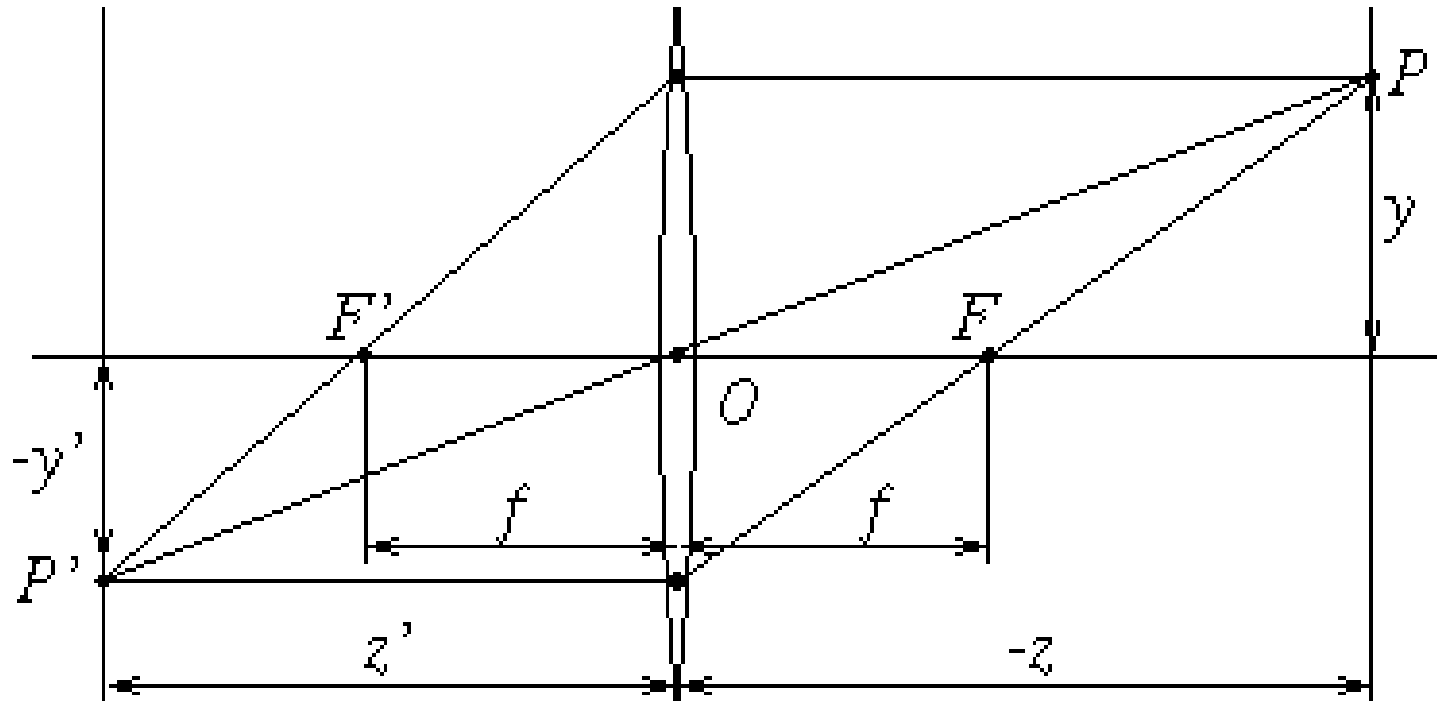


The reason for lenses



Lenses gather and focus light, allowing for brighter images.

The thin lens

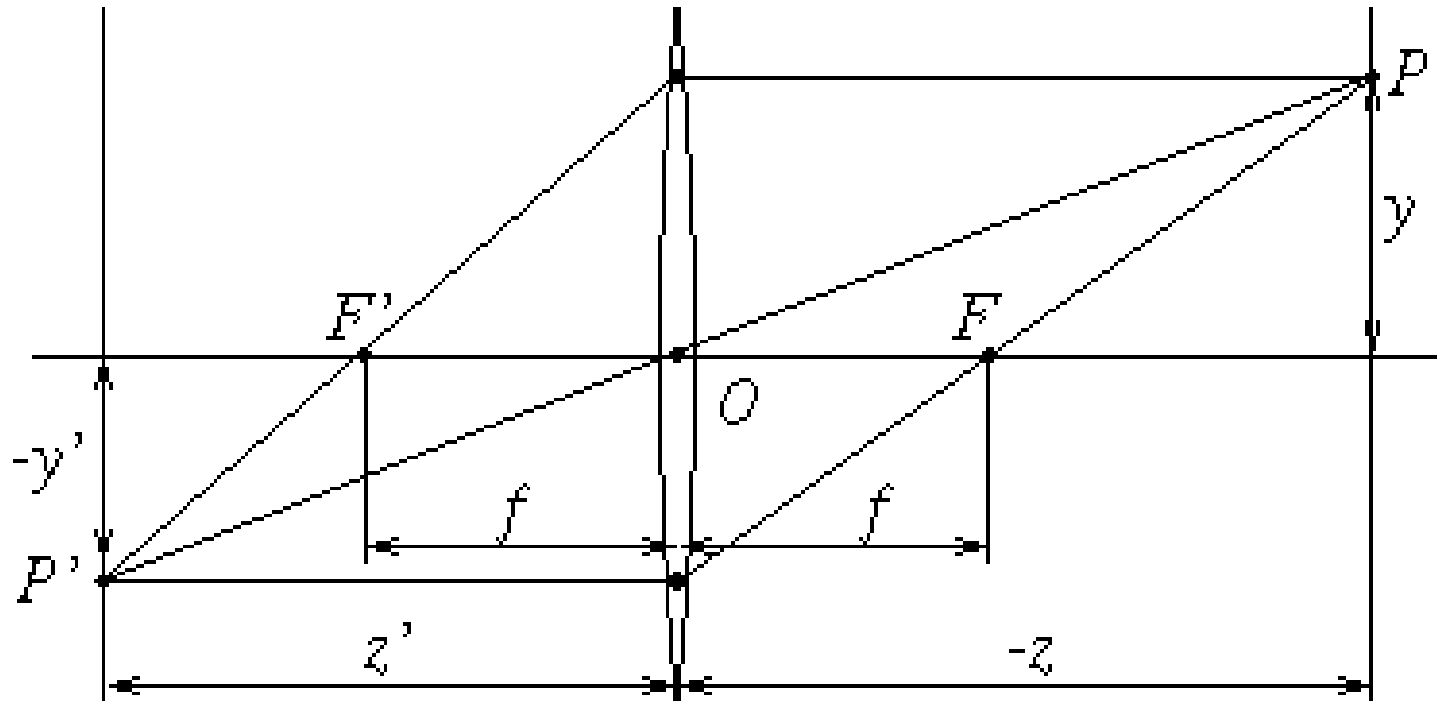


Thin Lens Properties:

1. A ray entering parallel to optical axis goes through the focal point.
2. A ray emerging from focal point is parallel to optical axis
3. A ray through the optical center is unaltered

$$\frac{1}{z'} - \frac{1}{z} = \frac{1}{f}$$

The thin lens



$$\frac{1}{z'} - \frac{1}{z} = \frac{1}{f}$$

Note that, if the image plane is very small and/or $z \gg z'$, then $z' \approx f$

Field of View

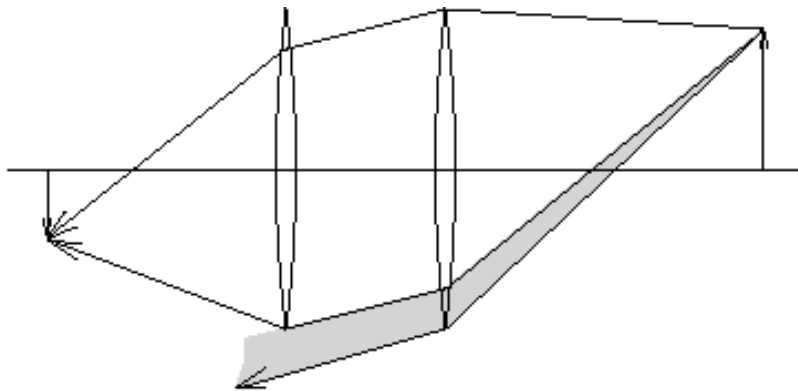
- The *effective diameter* of a lens (d) is the portion of a lens actually reachable by light rays.
- The effective diameter and the focal length determine the field of view:

$$\tan w = d / (2f)$$

- w is the half the total angular “view” of a lens system.
- Another fact is that in practice points at different distances are imaged, leading to so-called “circles of confusion” of size $d/z |z' - z|$ where z is the nominal image plane and z' is the focusing distance given by the thin lens equation.
- The “depth of field” is the range of distances that produce acceptably focused images. Depth of field varies inversely with focal length and lens diameter.

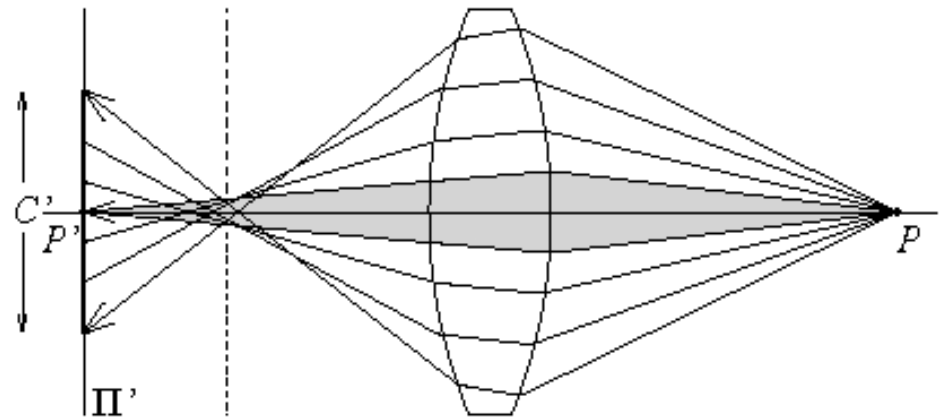
Lens Realities

Real lenses have a finite depth of field, and usually suffer from a variety of defects



vignetting

Spherical Aberration



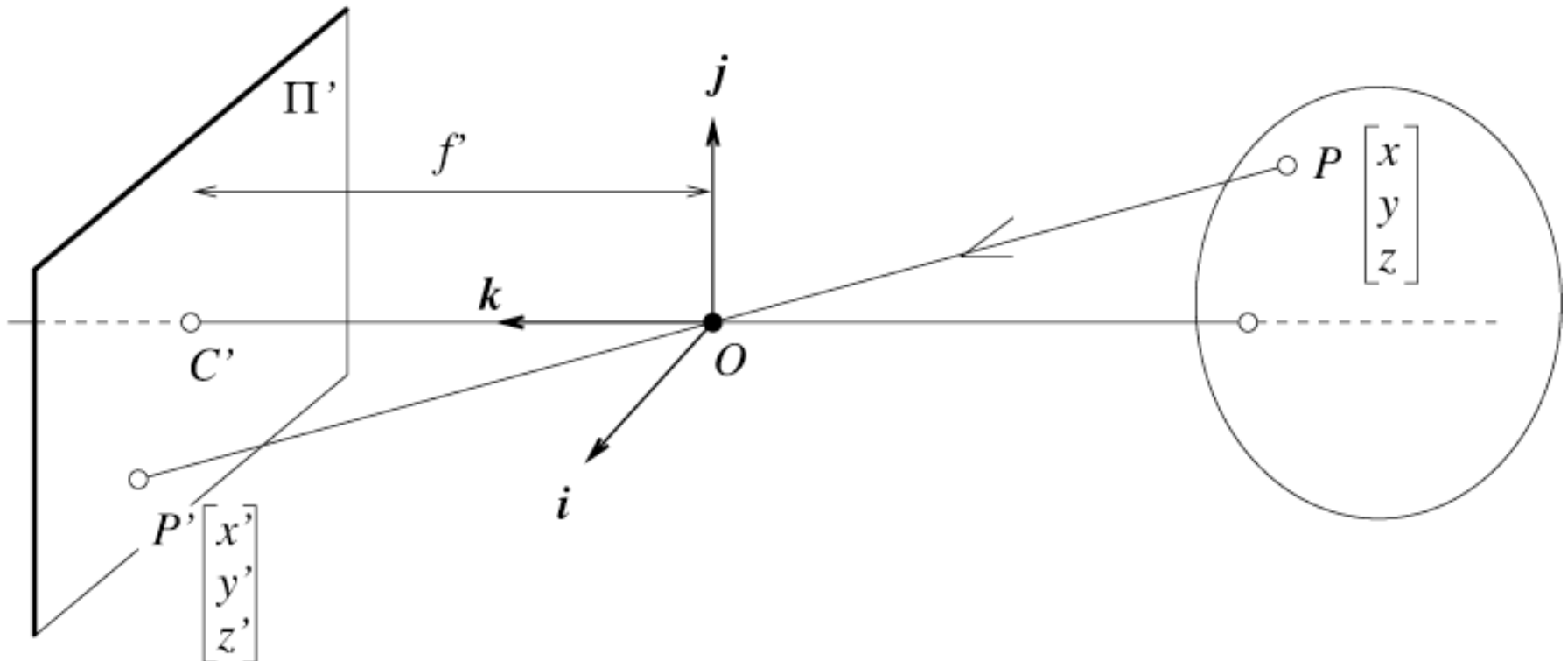
Standard Camera Coordinates

- Optical axis is z axis pointing outward
- X axis is parallel to the scanlines (rows) pointing to the right!
- By the right hand rule, the Y axis must point downward
- Note this corresponds with indexing an image from the upper left to the lower right, where the X coordinate is the column index and the Y coordinate is the row index.

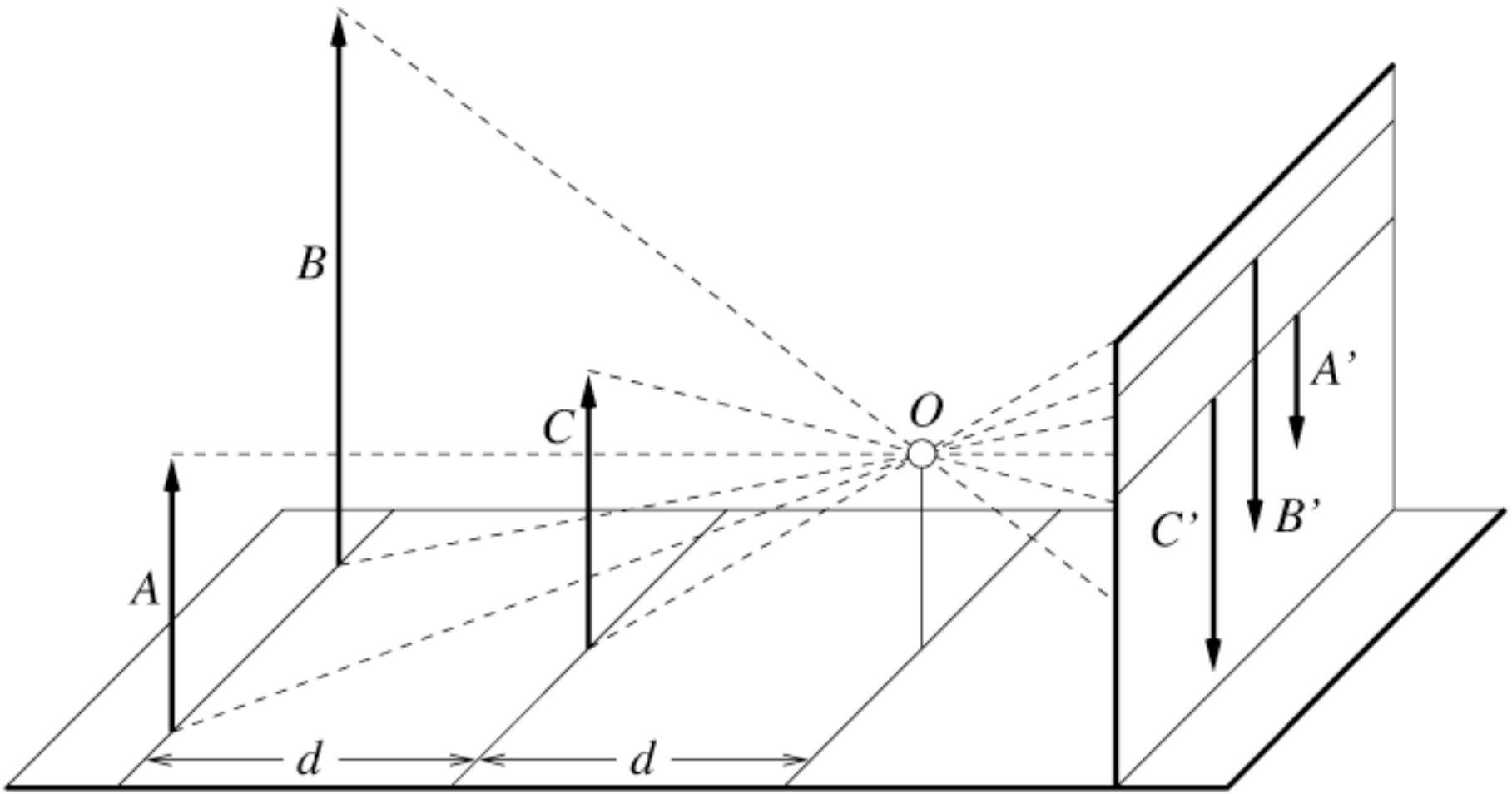
The equation of projection

- Equating z' and f
 - We have, by similar triangles, that $(x, y, z) \rightarrow (-f x/z, -f y/z, -f)$
 - Ignore the third coordinate, and flip the image around to get:

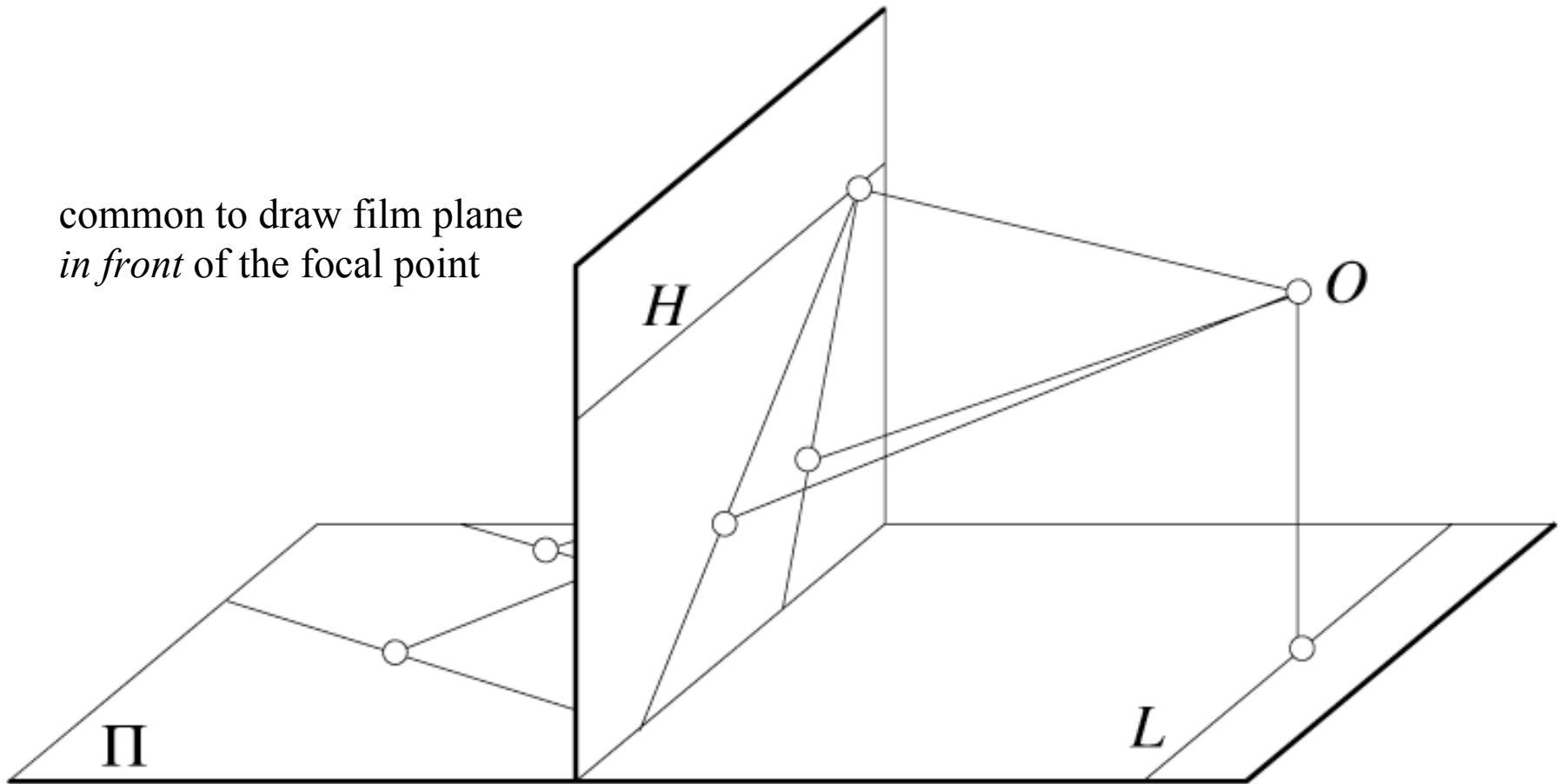
$$(x, y, z) \rightarrow \left(f \frac{x}{z}, f \frac{y}{z}\right)$$



Distant objects are smaller



Parallel lines meet



A Good Exercise: Show this is the case!

Parallel lines meet

- First, show how lines project to images.
- Second, consider lines that have the same direction (are parallel)
- Third, consider the degenerate case of lines parallel in the image
 - (by convention, the vanishing point is at infinity!)

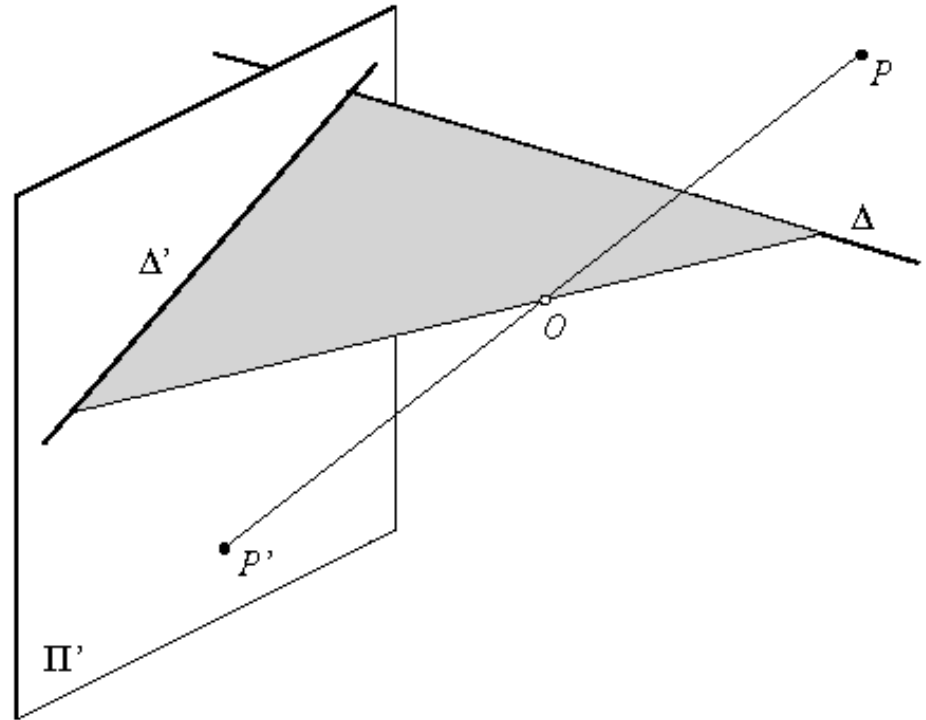
A Good Exercise: Show this is the case!

Vanishing points

- Another good exercise (really follows from the previous one): show the form of projection of *lines* into images.
- Each set of parallel lines (=direction) meets at a different point
 - The *vanishing point* for this direction
- Sets of parallel lines on the same plane lead to *collinear* vanishing points.
 - The line is called the *horizon* for that plane

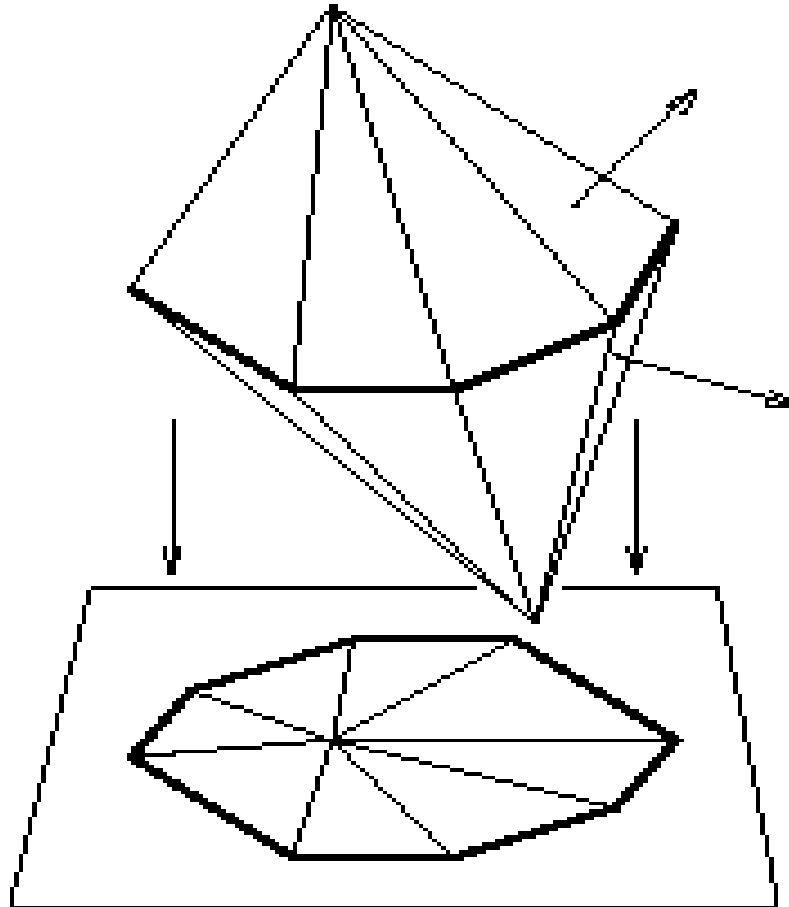
Geometric properties of projection

- Points go to points
- Lines go to lines
- Planes go to whole image
- Polygons go to polygons
- Degenerate cases
 - line through focal point to point
 - plane through focal point to line



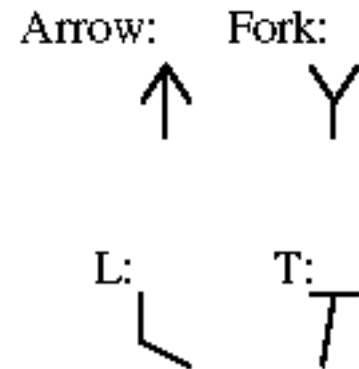
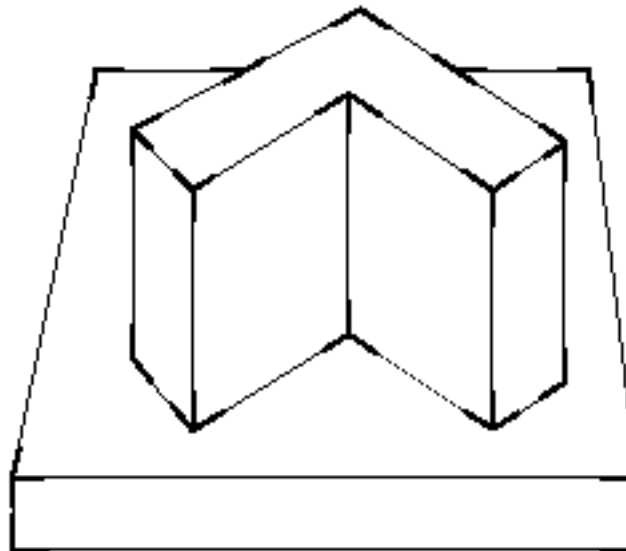
Polyhedra project to polygons

- (because lines project to lines)



Junctions are constrained

- This leads to a process called “line labelling”
 - one looks for consistent sets of labels, bounding polyhedra
 - disadv - can't get the lines and junctions to label from real images



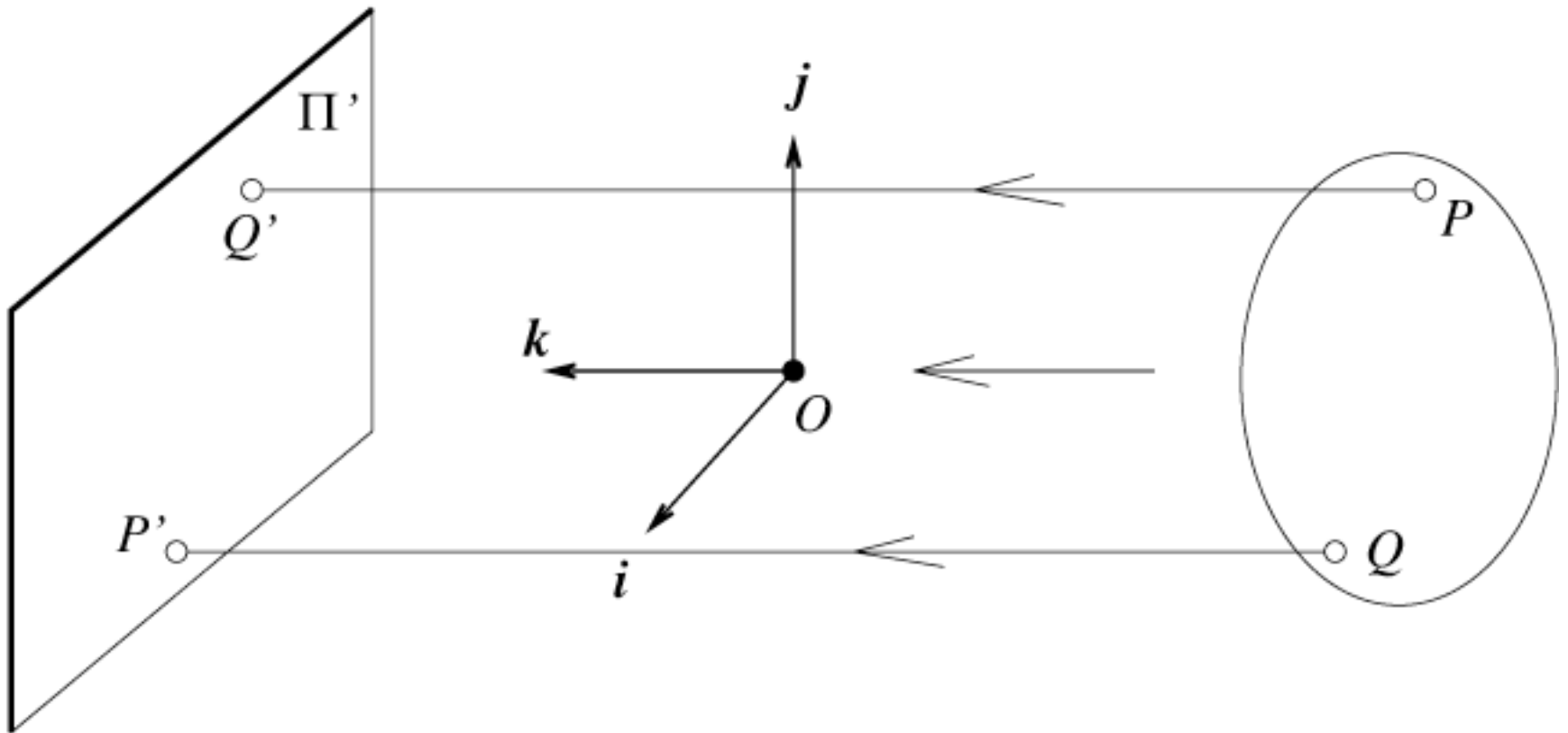
The Camera Matrix

- Homogenous coordinates for 3D
 - four coordinates for 3D point
 - equivalence relation (X,Y,Z,T) is the same as $(k X, k Y, k Z, k T)$
- Turn previous expression into HC's
 - HC's for 3D point are (X,Y,Z,T)
 - HC's for point in image are (U,V,W)

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix}$$

$$(U, V, W) \rightarrow \left(\frac{U}{W}, \frac{V}{W} \right) = (u, v)$$

Orthographic projection



Suppose I let f go to infinity; then

$$u = x$$

$$v = y$$

The model for orthographic projection

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix}$$

Weak perspective

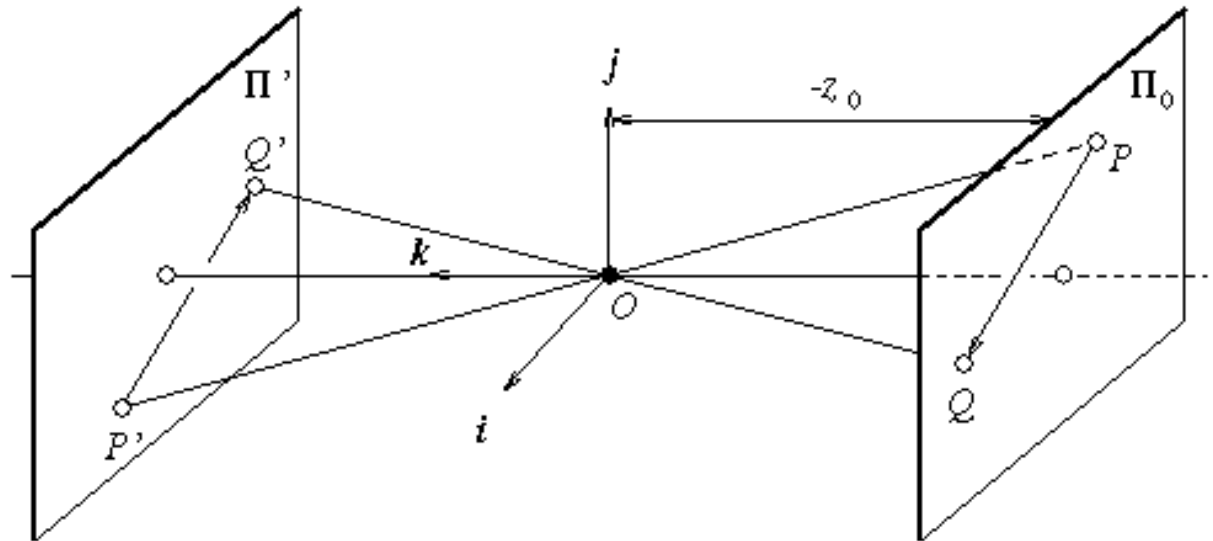
- Issue

- perspective effects, but not over the scale of individual objects
- collect points into a group at about the same depth, then divide each point by the depth of its group
- Adv: easy
- Disadv: wrong

$$u = sX$$

$$v = sy$$

$$s = f / Z^*$$



The model for weak perspective projection

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Z^* / f \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix}$$

Geometric Transforms

In general, a point in n-D space transforms by

$$P' = \text{rotate}(\text{point}) + \text{translate}(\text{point})$$

In 2-D space, this can be written as a matrix equation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} tx \\ ty \end{pmatrix}$$

In 3-D space (or n-D), this can be generalized as a matrix equation:

$$p' = R p + T \quad \text{or} \quad p = R^t (p' - T)$$

Geometric Transforms

Now, using the idea of homogeneous transforms,
we can write:

$$p' = \begin{pmatrix} R & T \\ 0 & 0 & 0 & 1 \end{pmatrix} p$$

R and T both require 3 parameters. These correspond to the 6 extrinsic parameters needed for camera calibration

Intrinsic Parameters

Intrinsic Parameters describe the conversion from unit focal length metric to pixel coordinates (and the reverse)

$$\begin{aligned}x_{\text{mm}} &= - (x_{\text{pix}} - o_x) s_x \rightarrow -1/s_x x_{\text{mm}} - o_x = -x_{\text{pix}} \\y_{\text{mm}} &= - (y_{\text{pix}} - o_y) s_y \rightarrow -1/s_y y_{\text{mm}} - o_y = -y_{\text{pix}}\end{aligned}$$

or

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix}_{\text{pix}} = \begin{pmatrix} -1/s_x & 0 & o_x \\ 0 & -1/s_y & o_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ w \end{pmatrix}_{\text{mm}} = M_{\text{int}} p$$

It is common to combine scale and focal length together as they are both scaling factors; note projection is unitless in this case!

Camera parameters

- Summary:
 - points expressed in external frame
 - points are converted to canonical camera coordinates
 - points are projected
 - points are converted to pixel units

$$\begin{array}{c}
 \left(\begin{array}{c} U \\ V \\ W \end{array} \right) = \left(\begin{array}{c} \text{Transformation} \\ \text{representing} \\ \text{intrinsic parameters} \end{array} \right) \left(\begin{array}{c} \text{Transformation} \\ \text{representing} \\ \text{projection model} \end{array} \right) \left(\begin{array}{c} \text{Transformation} \\ \text{representing} \\ \text{extrinsic parameters} \end{array} \right) \left(\begin{array}{c} X \\ Y \\ Z \\ T \end{array} \right)
 \end{array}$$

point in pixel coords.
point in metric image coords.
point in cam. coords.
point in world coords.

10/5/2001
CS 461, Copyright G.D. Hager

Lens Distortion

- In general, lens introduce minor irregularities into images, typically radial distortions:

$$x = x_d(1 + k_1r^2 + k_2r^4)$$

$$y = y_d(1 + k_1r^2 + k_2r^4)$$

$$r^2 = x_d^2 + y_d^2$$

- The values k_1 and k_2 are additional parameters that must be estimated in order to have a model for the camera system.

Other Models

- The *affine camera* is a generalization of weak perspective.
 - $u = A p + d$
 - A is 2×3 and d is 2×1
 - This can be derived from scaled orthography
- The *projective camera* is a generalization of the perspective camera.
 - $u' = M p$
 - M is 3×4 nonsingular defined up to a scale factor
 - This just a generalization (by one parameter) from “real” model
- Both have the advantage of being linear models on real and projective spaces, respectively.

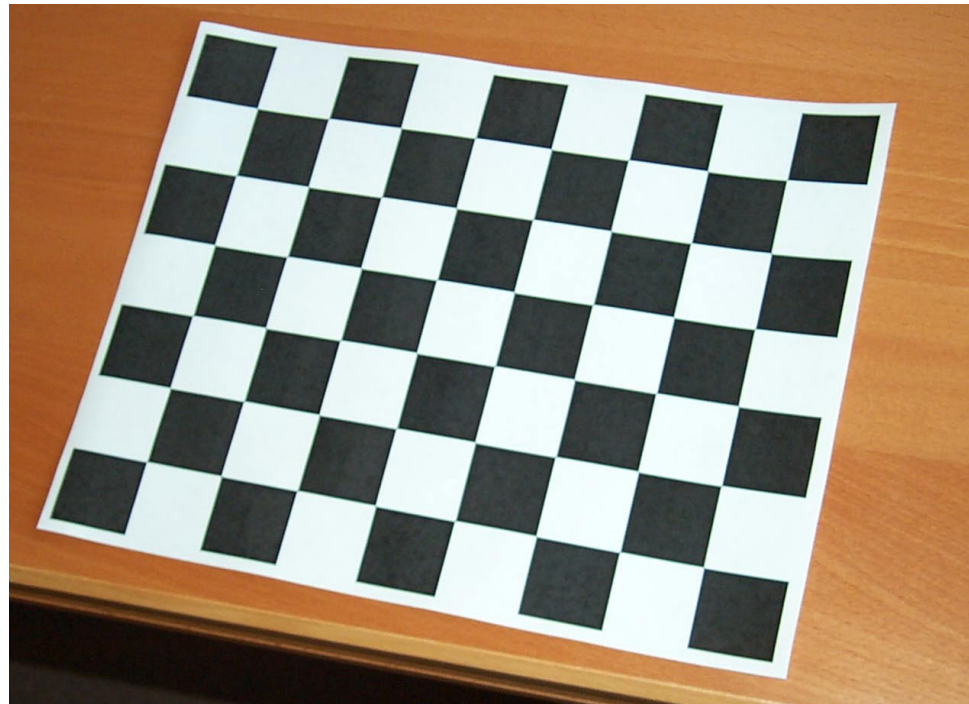
Camera calibration

- Issues:
 - what are intrinsic parameters of the camera?
 - what is the camera matrix? (intrinsic+extrinsic)
- General strategy:
 - view calibration object
 - identify image points
 - obtain camera matrix by minimizing error
 - obtain intrinsic parameters from camera matrix
- Most modern systems employ the multi-plane method
 - avoids knowing absolute coordinates of calibration points
- Error minimization:
 - Linear least squares
 - easy problem numerically
 - solution can be rather bad
 - Minimize image distance
 - more difficult numerical problem
 - solution usually rather good, but can be hard to find
 - start with linear least squares
 - Numerical scaling is an issue

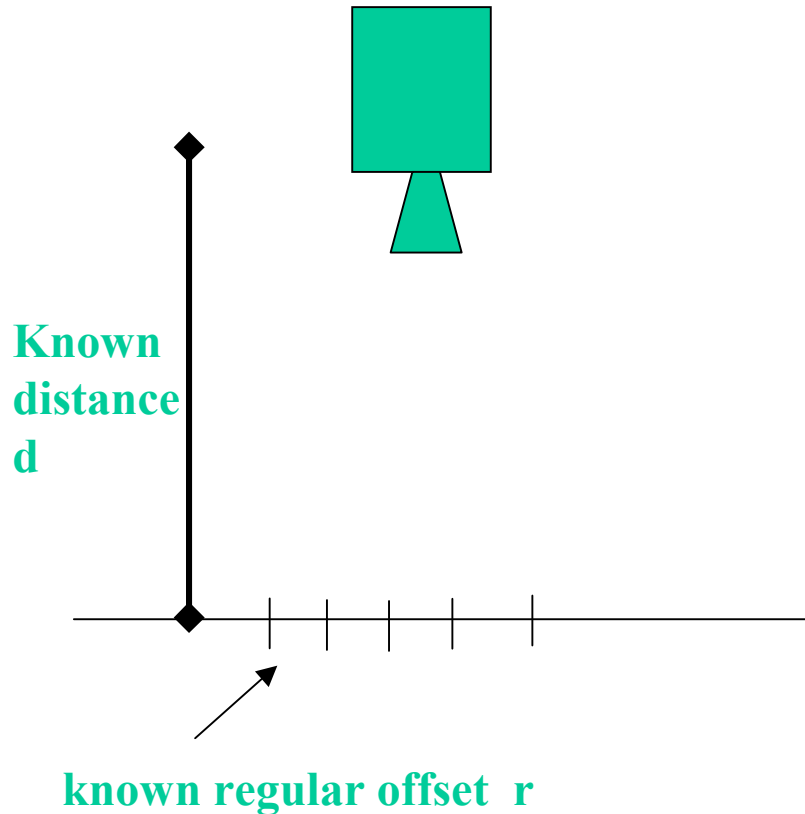
CAMERA CALIBRATION

The problem:

Compute the camera intrinsic and extrinsic parameters using only observed camera data.



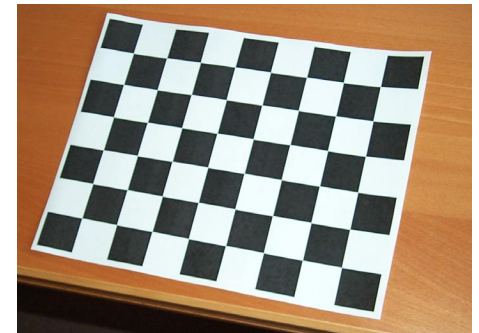
CAMERA CALIBRATION: A WARMUP



$$\frac{rk_i}{d} = (x_i - o_x)s_x$$

$$\frac{r}{d} = (x_{i+1} - x_i)s_x$$

A simple way to get scale parameters; we can compute the optical center as the numerical center and therefore have the intrinsic parameters



More Generally

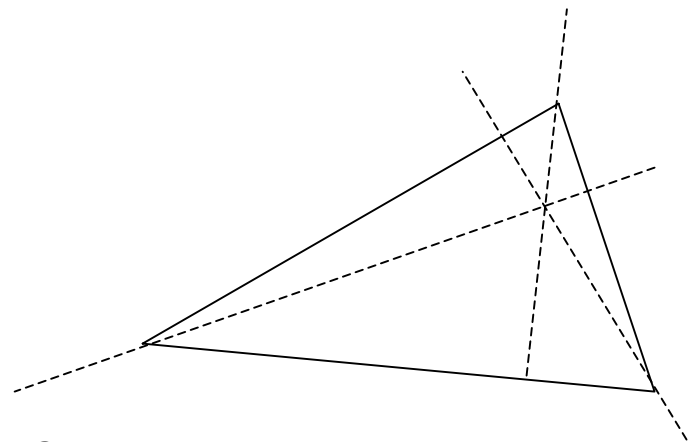
- Separate estimation is problematic and error prone
 - still have to solve for image center and camera pose
- Observation ---- perhaps it is possible to generate a calibration rig that allows direct estimation of all parameters
 - Two orthogonal planes
 - Regular grid of points

A Warmup Exercise

- Suppose we want to calibrate the affine camera and we know $u_i = A p_i + d$ for many pairs i
- m is mean of u 's and q is mean of p 's; note $m = A q + d$
- $U = [u_1 - m, u_2 - m, \dots, u_n - m]$ and $P = [p_1 - q, p_2 - q, \dots, p_n - q]$
- $U = A P \rightarrow U P' (P P')^{-1} = A$
- d is now mean of $u_i - A p_i$

The General Case

- Affine is “easy” because it is linear and unconstrained (note orthographic is harder because of constraints)
- Perspective case is also harder because it is both nonlinear and constrained
- Observation: optical center can be computed from the *orthocenter* of vanishing points of orthogonal sets of lines.



Basic Equations

$${}^c T_w = (T_x, T_y, T_z)'$$

$${}^c R_w = (R_x, R_y, R_z)'$$

$${}^c p = {}^c R_w {}^w p + {}^c T_w$$

$$u = -f \frac{R_x p + T_x}{R_z p + T_z}$$

$$v = -f \frac{R_y p + T_y}{R_z p + T_z}$$

Basic Equations

$$u_{pix} = \frac{1}{s_x} u + o_x$$

$$v_{pix} = \frac{1}{s_y} v + o_y$$

$$\bar{u} = u_{pix} - o_x = -f_x \frac{R_x p + T_x}{R_z p + T_z}$$

$$\bar{v} = v_{pix} - o_y = -f_y \frac{R_y p + T_y}{R_z p + T_z}$$

Basic Equations

$$\bar{u}_i f_y(R_y p_i + T_y) = \bar{v}_i f_x(R_x p_i + T_x)$$

$$\bar{u}_i(R_y p_i - T_y) - \bar{v}_i \alpha(R_x p_i + T_x) = 0$$

$$r = \alpha R_x \text{ and } w = \alpha T_x$$

$$t = R_y \text{ and } s = T_y$$

one of these for each point


$$A_i = (u_i p_i, u_i, -v_i p_i, -v_i) \text{ and } A[t, s, w, r]' = 0$$

Basic Equations

$A_i = (u_i p_i, u_i, -v_i p_i, -v_i)$ and

$$A[t, s, w, r]' = Am = 0$$

Note that m is defined up a scale factor!

$A = UDV'$ and choose m as column of V corresponding to the smallest singular value

Basic Equations

$$A_i = (u_i p_i, u_i, -v_i p_i, -v_i) \text{ and}$$
$$A[t, s, w, r]' = Am = 0$$

$\|t\| = |\gamma|$ gives scale factor for solution

$$\|w\| = |\gamma|\alpha$$

We now know R_x and R_y up to a sign and gamma.

$$R_z = R_x \times R_y$$

We will probably use another SVD to orthogonalize this system ($R = U D V'$; set D to I and multiply).

Last Details

- We still need to compute the correct sign.
 - note that the denominator of the original equations must be positive (points must be in front of the cameras)
 - Thus, the numerator and the projection must disagree in sign.
 - We know everything in numerator and we know the projection, hence we can determine the sign.
- We still need to compute T_z and f_x
 - we can formulate this as a least squares problem on those two values using the first equation.

$$\begin{aligned}\bar{u} &= -f_x \frac{R_x p + T_x}{R_z p + T_z} \rightarrow \\ \bar{u}(R_z p + T_z) &= -f_x(R_x p + T_x) \\ f_x(R_x p + T_x) + \bar{u}T_z &= -\bar{u}R_z p \\ A(f_x, T_z)' &= b \rightarrow (f_x, T_z)' = (A'A)^{-1} A'b\end{aligned}$$

The Algorithm

1. Compute image center from orthocenter
2. Compute the A matrix (6.8)
3. Compute solution with SVD
4. Compute gamma and alpha
5. Compute R (and normalize)
6. Compute f_x and T_z