CAMERA GEOMETRY

Ajit Rajwade,
CS 763, Winter 2016,
IITB, CSE department

Contents

- Transformations of points/vectors in 2D
- Transformations of points/vectors in 3D
- Image Formation: geometry of projection of 3D points onto 2D image plane
- Vanishing Points
- Camera Calibration

Transformations of points and vectors in 2D

Translation and Rotation

- Let us denote the coordinates of the original point as *(x_1,y_1)* and those of the transformed point as as *(x_2,y_2)*.
- Translation:

$$x_2 = x_1 + t_x$$
$$y_2 = y_1 + t_y$$

- Rotation about point *(0,0)* anti-clockwise through angle θ

$$x_2 = x_1 \cos\theta - y_1 \sin\theta$$
$$y_2 = x_1 \sin\theta + y_1 \cos\theta$$
$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

2D Rotation matrix (orthonormal matrix and with determinant 1)

Translation and rotation

- Rotation about point *(x_c,y_c)* anti-clockwise through angle θ

$$x_2 = (x_1 - x_c)\cos\theta - (y_1 - y_c)\sin\theta + x_c$$
$$y_2 = (x_1 - x_c)\sin\theta + (y_1 - y_c)\cos\theta + y_c$$

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & x_c \\ \sin\theta & \cos\theta & y_c \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 - x_c \\ y_1 - y_c \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} \cos\theta & -\sin\theta & x_c \\ \sin\theta & \cos\theta & y_c \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

Note: we are introducing a third (fictitious) coordinate even though the points are in 2D. This enables translation operations to be expressed in the matrix multiplication framework. This new coordinate is called as the **“homogeneous coordinate”**.

-Perform translation such that (x_c,y_c) coincides with the origin.

-Rotate about the new origin.

-Translate back.

Translation and rotation

- Rotation and translation:

$$x_2 = x_1 \cos\theta - y_1 \sin\theta + t_x$$

$$y_2 = x_1 \sin\theta + y_1 \cos\theta + t_y$$

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

- Translations and rotations are rigid transformations – they preserve angles, lengths and areas.

Translations/rotation to Affine Transformations

- Affine transformation: rotation, translation, **scaling and shearing**

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & t_x \\ A_{21} & A_{22} & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

Assumption: the 2 x 2 sub-matrix **A** is NOT rank deficient, otherwise it will transform two-dimensional figures into a line or a point. The matrix **A** in general is NOT orthonormal. If it is orthonormal, this reduces to either a rotation (determinant 1) or reflection (determinant = -1)

Affine Transformations

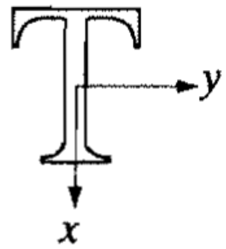
- Scaling

Identity

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$x = v$

$y = w$

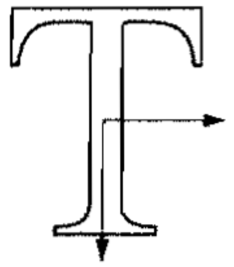


Scaling

$$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$x = c_x v$

$y = c_y w$



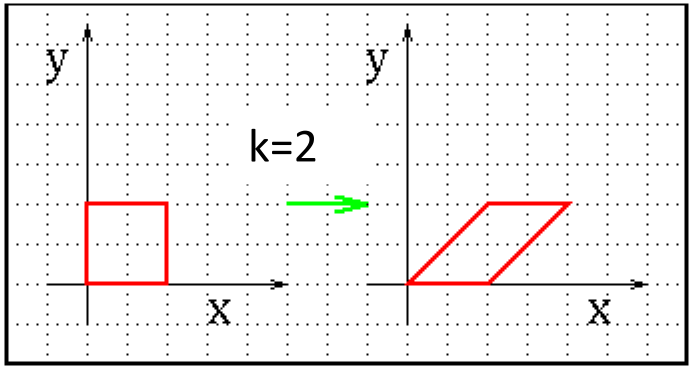
If cx = -1 and cy = 1, then the scaling operation is called reflection across the Y axis. If cx = 1 and cy = -1, then it is called reflection across X axis. Reflections can occur across an arbitrary axis.

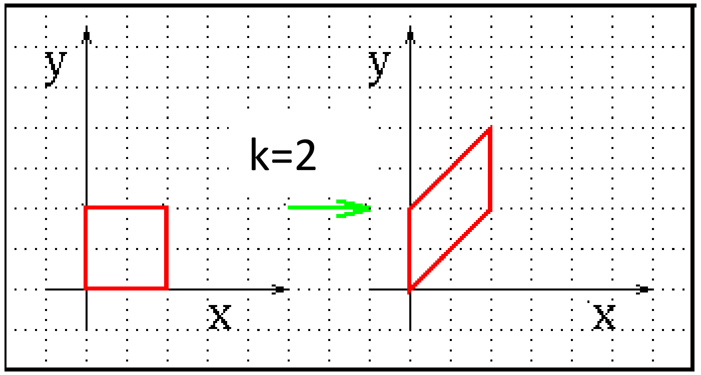
Affine transformations

- Shearing:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \Rightarrow x' = x + ky, y' = y$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ k & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \Rightarrow x' = x; y' = kx + y$$





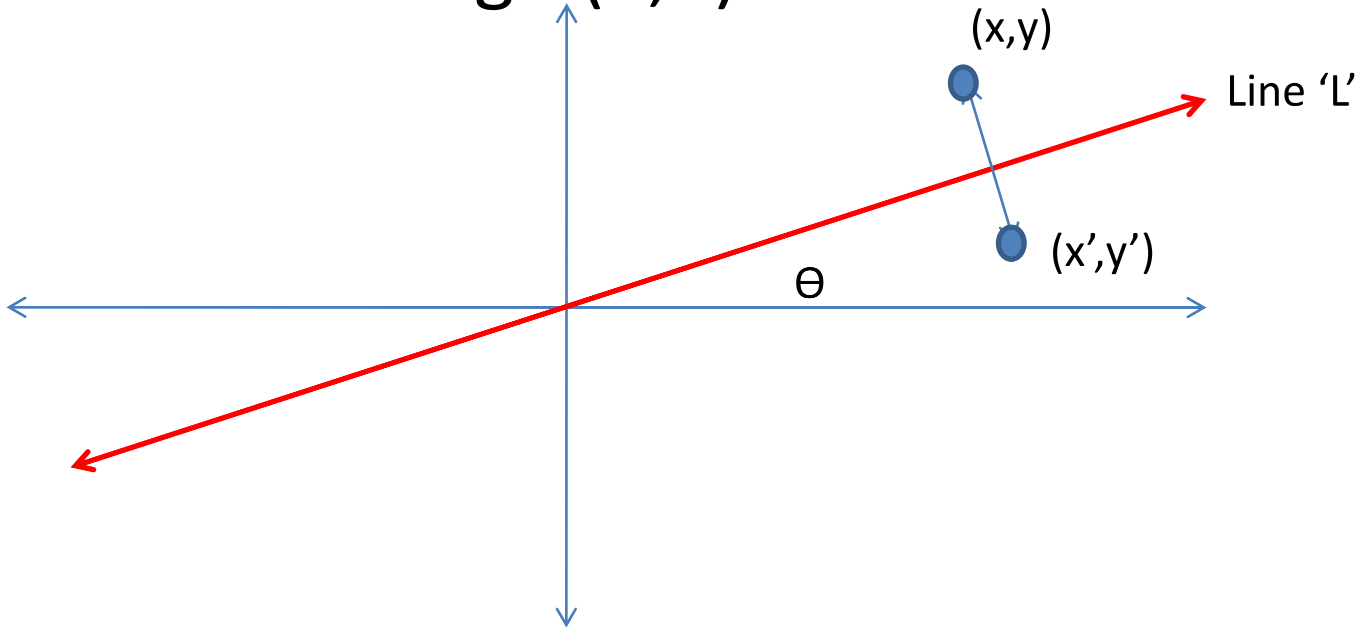
http://cs.fit.edu/~wds/classes/cse5255/thesis/shear/shear.html

Affine transformation

- The 2D affine transformation model (including translation in X and Y direction) includes **6 degrees of freedom**.
- The affine transformations generally do **not** preserve lengths, angles and areas.
- But they preserve collinearity of points, i.e. points on one line remain on a line after affine transformation.
- They preserve ratios of lengths of collinear line segments and ratios of areas.
- In fact area of transformed object = |**A**| x area of original object, where **A** is the transformation matrix.

Composition of affine transformations

- Composition of multiple types of affine transformations is given by the multiplication of their corresponding matrices.
- Example: reflection across an arbitrary direction through (0,0).



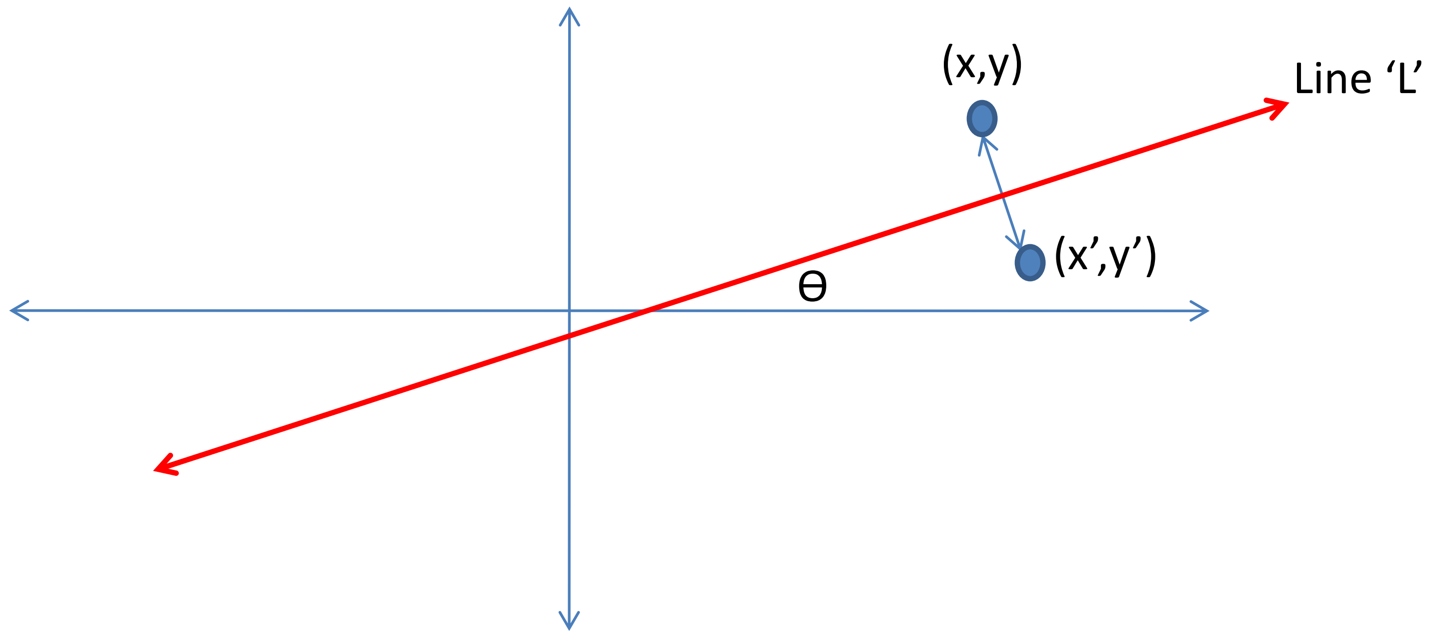
Composition of affine transformations

- Apply a rotation **R** such that line *L* coincides with either the X axis (or the Y axis).
- Apply a reflection **R'** about the X axis (or the Y axis).
- Apply a reverse rotation **R''=R**$^{-1}$ such that the original orientation of *L* is restored.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \underbrace{\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}}_{\mathbf{R''}} \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}}_{\mathbf{R'}} \underbrace{\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}}_{\mathbf{R}} \begin{pmatrix} x \\ y \end{pmatrix}$$

Composition of affine transformations

- How will you change this if the direction *L* didn't pass through the origin?



Composition of affine transformations

- In general, affine transformations are not commutative, i.e. $T_A T_B \neq T_B T_A$.
- When composing transformations, remember that successive translations are additive (and commutative).

$$x_2 = x_1 + t_x + u_x$$

$$y_2 = y_1 + t_y + u_y$$

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & u_x \\ 0 & 1 & u_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & u_x + t_x \\ 0 & 1 & u_y + t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

Composition of affine transformations

- When composing transformations, remember that successive rotations are additive (and commutative).

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta+\phi) & -\sin(\theta+\phi) & 0 \\ \sin(\theta+\phi) & \cos(\theta+\phi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

- Successive scalings are multiplicative (and commutative).

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} = \begin{pmatrix} w_x & 0 & 0 \\ 0 & w_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} = \begin{pmatrix} w_x s_x & 0 & 0 \\ 0 & w_y s_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

Transformations of points/vectors in 3D

Translation and Rotation in 3D

$$x' = x + t_x$$
$$y' = y + t_y$$
$$z' = z + t_z$$

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \mathbf{R} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

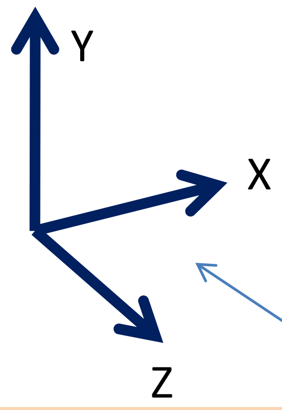
3D Rotation matrix, size 3 x 3. **R** is orthonormal ($\mathbf{R}^T = \mathbf{R}^{-1}$)and has a determinant of 1.

Rotation in 3D is associated with an **axis**. We refer to rotation in 3D as rotation **about an axis**. The following are the rotation matrices for an angle about the X,Y,Z axes respectively.

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

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

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



Right-handed coordinate system (X axis to the right, Y axis points up)

Note: Rotation about X axis leaves X coordinate unchanged, rotation about Y axis leaves Y coordinate unchanged, rotation about Z axis leaves Z coordinate unchanged.

All these 3 matrices represent *anti-clockwise* rotation of *column* vectors, assuming the axis of rotation points toward the *observer* and we have a ***right-handed*** **coordinate system**.

http://www.cs.iastate.edu/~cs577/handouts/rotation.pdf
http://paulbourke.net/geometry/rotate/

Rotation about arbitrary axis

- Let the arbitrary axis be given as a unit vector **r** = (r_1,r_2,r_3) passing through the origin. We want to rotate about this axis through angle ρ.

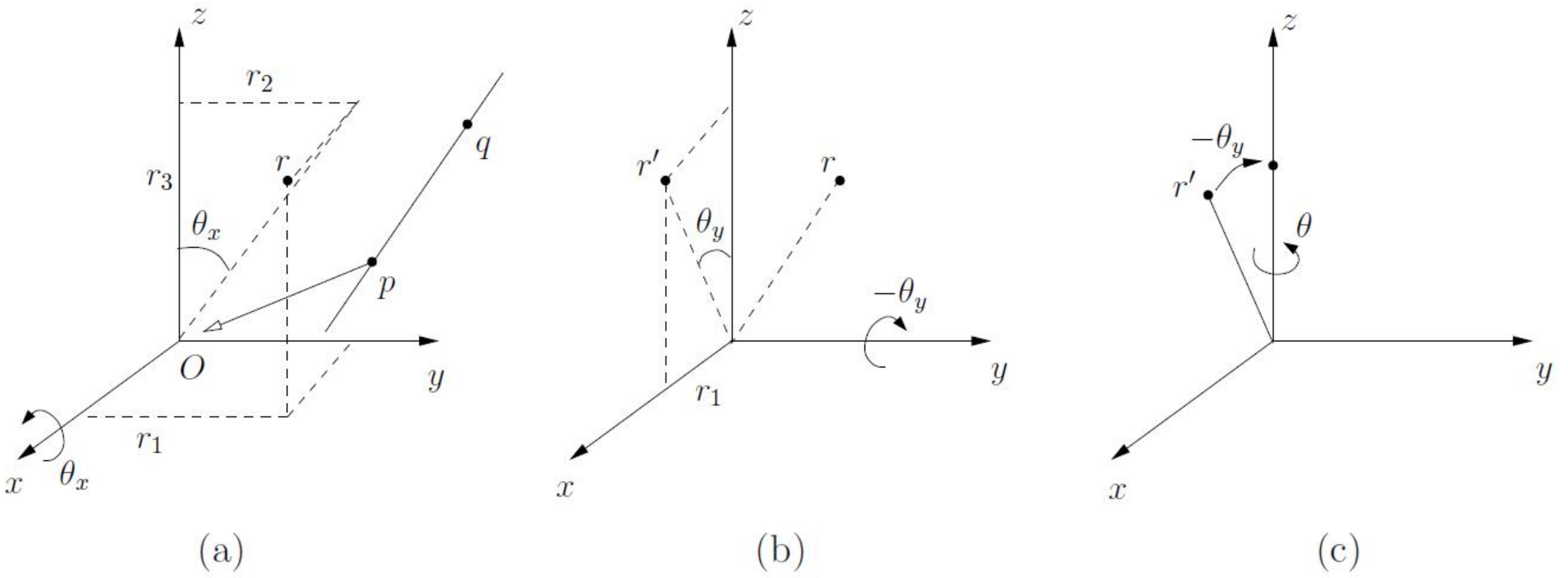


Figure 1: Rotation about an arbitrary axis performed by transforming the axis to the z-axis, applying the rotation, and transforming back to the original axis.

Rotation about an arbitrary axis

- Rotate **r** about the X axis so that it now aligns with the XZ plane (i.e. we want to nullify its Y coordinate). For this, we rotate about X axis through an angle Θ_x given by

$$\sin\theta_x = \frac{r_2}{\sqrt{r_2^2 + r_3^2}}, \cos\theta_x = \frac{r_3}{\sqrt{r_2^2 + r_3^2}}$$

transforming $\mathbf{r} = (r_1, r_2, r_3)$ to $\mathbf{r}' = (r_1, 0, \sqrt{r_2^2 + r_3^2})$

Note: As the projection of **r** onto the ZY planes aligns itself with the Z axis (that's how the angle θ_x is calculated), the vector **r** gets into the XZ plane.

Rotation about an arbitrary axis

- Now rotate **r’** about Y axis through angle (-θ_y) (to align it with the Z axis) given by:

$$\sin\theta_y = \frac{r_1}{\sqrt{r_1^2 + r_2^2 + r_3^2}}, \cos\theta_y = \frac{\sqrt{r_2^2 + r_3^2}}{\sqrt{r_1^2 + r_2^2 + r_3^2}}$$

transforming $\mathbf{r'} = (r_1, 0, \sqrt{r_2^2 + r_3^2})$ to

$$\mathbf{r''} = (0,0,1)$$

Rotation about an arbitrary axis

- **Now rotate about the Z axis through angle *ρ*.**
- Now *reverse rotate* about Y-axis through angle $\Theta_{y.}$
- Now *reverse rotate* about X-axis through angle $\Theta_{x.}$
- The final rotation matrix is given as:

$$\mathbf{R_r}(\rho) = \mathbf{R_x}(-\theta_x)\mathbf{R_y}(\theta_y)\mathbf{R_z}(\rho)\mathbf{R_y}(-\theta_y)\mathbf{R_x}(\theta_x)$$

- Note: in this derivation, we could have also applied transformations to bring **r** into the YZ plane by rotation about Y axis (instead of bringing it into the XZ plane by rotating about X axis), and then aligned it with Z axis by rotation about X axis.
- Also there is no specific reason why we chose Z axis to align vector **r** with. We could have chosen the X or Y axes as well.

Rotation about an arbitrary axis

- The final rotation matrix can be shown to have the following form (axis is unit vector **L =** *[l,m,n],* i.e. $l^2+m^2+n^2=1$, passing through the origin):

$$\begin{bmatrix} ll(1-\cos\theta)+\cos\theta & ml(1-\cos\theta)-n\sin\theta & nl(1-\cos\theta)+m\sin\theta \\ lm(1-\cos\theta)+n\sin\theta & mm(1-\cos\theta)+\cos\theta & nm(1-\cos\theta)-l\sin\theta \\ ln(1-\cos\theta)-m\sin\theta & mn(1-\cos\theta)+l\sin\theta & nn(1-\cos\theta)+\cos\theta \end{bmatrix}.$$

Rotation matrices

- Regardless of axis, any rotation matrix is always orthonormal.
- The dot product of any two (different) rows is 0, the dot product of any two (different) columns is 0.
- The magnitude of any row (or column) is 1.
- Thus $\mathbf{R}\mathbf{R}^{T} = \mathbf{R}^{T}\mathbf{R} = \mathbf{I}$, $\mathbf{R}^{\mathbf{T}}\mathbf{=}\mathbf{R}^{\mathbf{-1}}$
- A rotation matrix is orthonormal, but not every orthonormal matrix is a rotation matrix (why?)

Affine transformations in 3D

- 3D affine transformations are represented as follows:

$$\begin{pmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{13} & t_x \\ A_{21} & A_{22} & A_{23} & t_y \\ A_{31} & A_{32} & A_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix}$$

There are 12 degrees of freedom. This subsumes rotations, **scalings, shearing, reflections** and translations.

Rigid transformations (rotations and translations) preserve lengths, areas and angles.
Affine transformations preserve collinearity of points, and ratios of distances, but **not** lengths or areas or angles.

Reflections in 3D

- Across XY plane: Z coordinate changes

$$\begin{pmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix}$$

- Across YZ plane: X coordinate changes

$$\begin{pmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix}$$

Composition of affine transformations

- Composition of multiple types of affine transformations is given by the multiplication of their corresponding matrices.
- Example: Reflection across an arbitrary plane P:

1. Translate so that a known point in plane P coincides with the origin.
2. Rotate such that the normal vector of the plane coincides with the Z axis.
3. Perform reflection across XY plane (Z axis is normal to the XY plane).
4. Apply inverse transform for step 2.
5. Apply inverse transform for step 1.

Composition of affine transformations

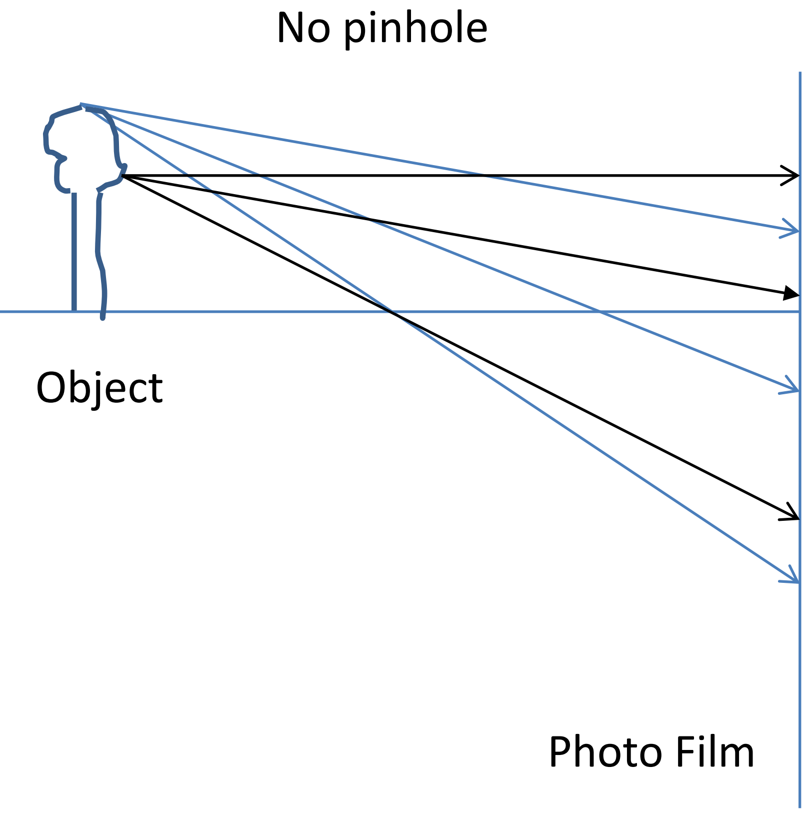
- Just as in 2D, affine transformations in 3D do not commute in general.
- Successive translations are additive, successive scaling *about the same axis* are multiplicative, successive rotations *about the same axis* are additive.

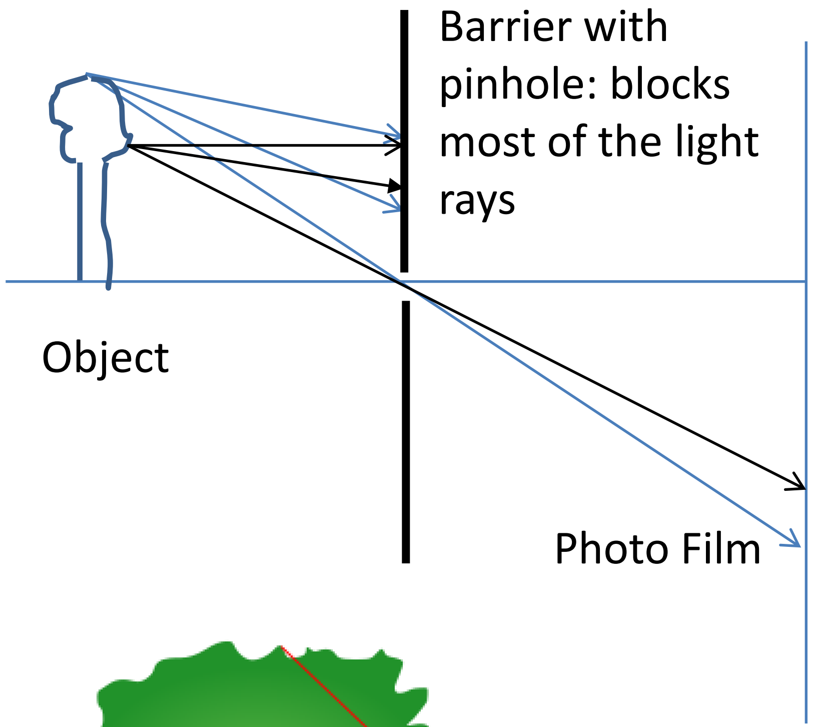
Degrees of freedom (DoF) in 3D rotation matrix

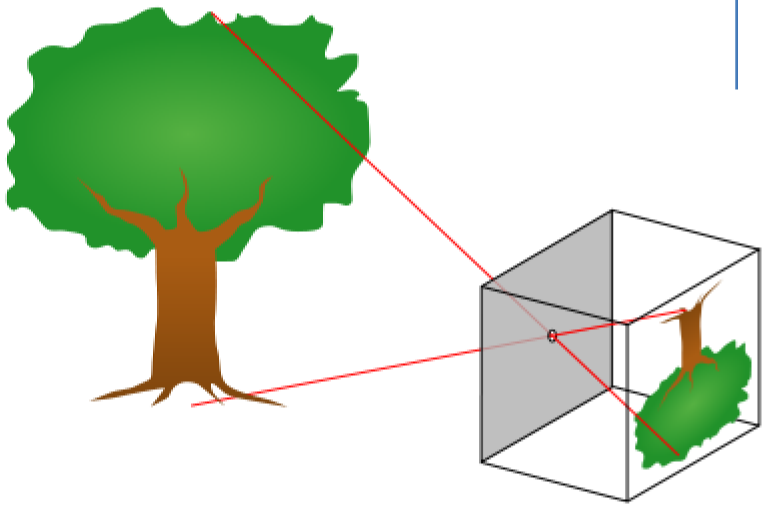
- The number of DoF refers to the number of independent parameters required to characterize the rotation matrix.
- A 3D rotation matrix has size 3 x 3, but it has only 3 DoF.
- This is because the rotation in 3D is parameterized by an axis (which is a unit vector and has 2 DoF) and the angle of rotation (which gives the 3rd DoF).
- Another way of seeing this: the first column of the orthonormal matrix accounts for 2 DoF (as it is a unit vector), the second column being perpendicular to the first one accounts for one more DoF (why just one more? *), and the third column is a cross product of the first two columns (taking care to choose the sign such that the determinant of the matrix is 1).
- * If you consider a plane perpendicular to the vector given by the first column, then the second vector is given by a rotation by angle (say) Υ inside this plane – that's why just one DoF.

Image Formation in a Camera

Pinhole Camera







Pinhole Camera with Non-Inverted Plane

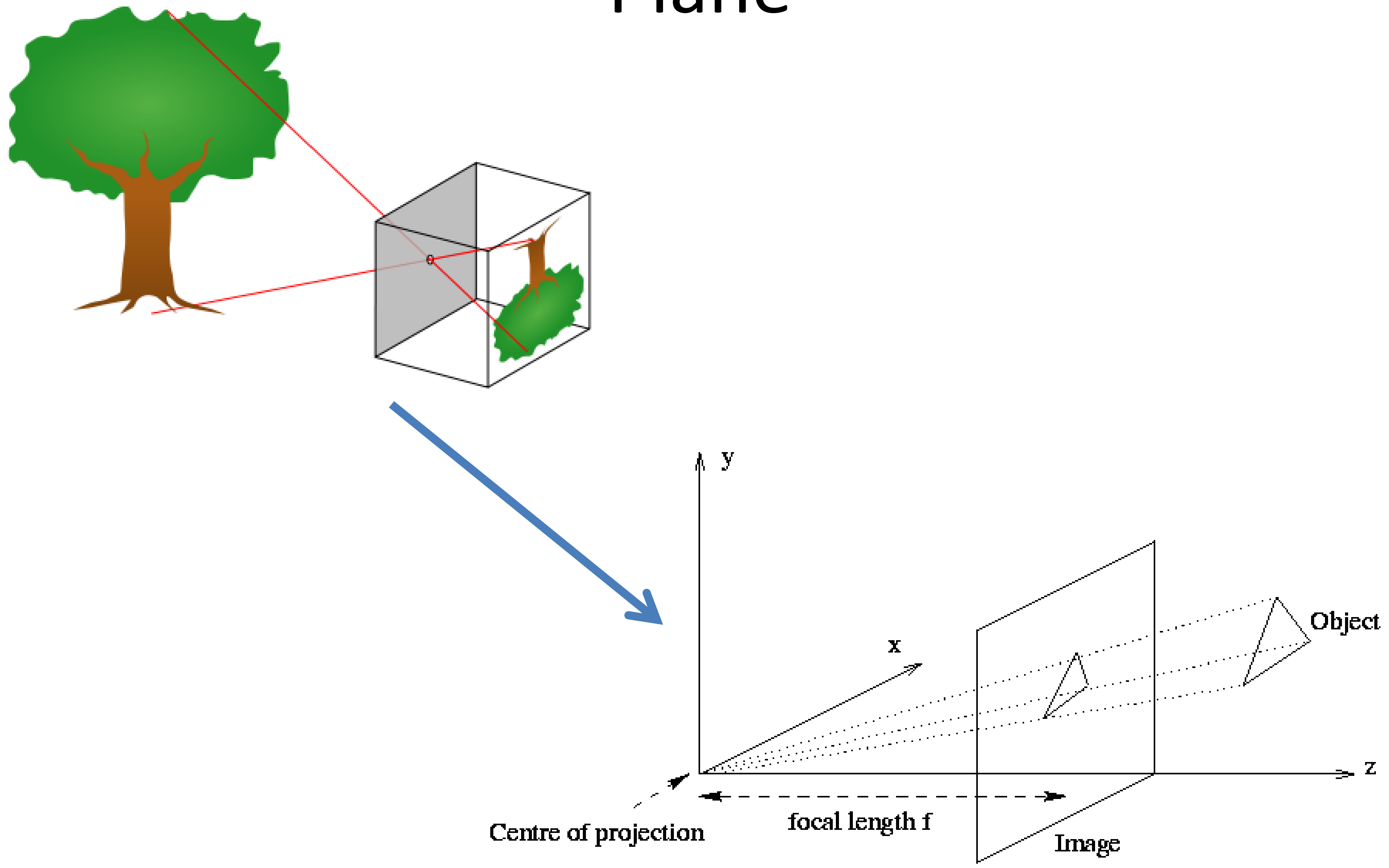
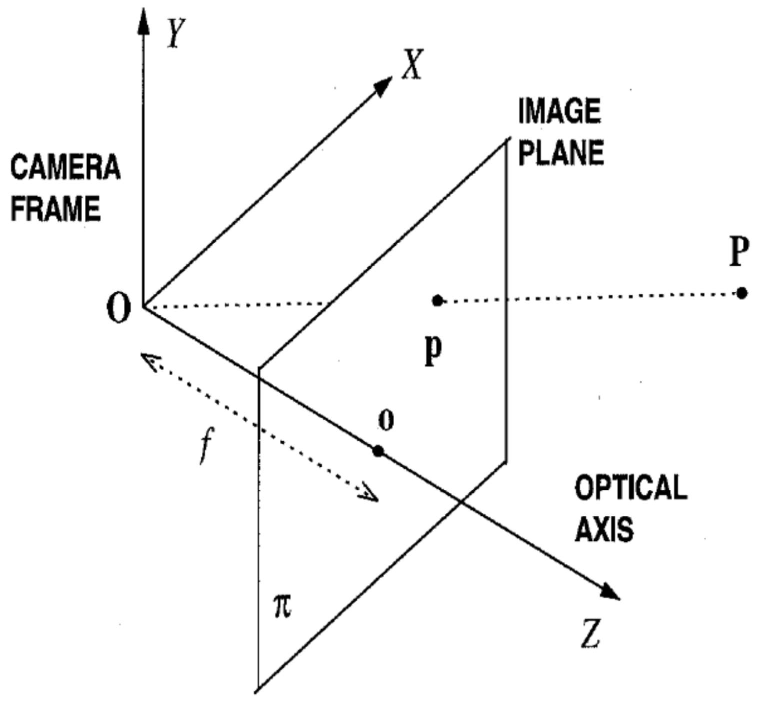
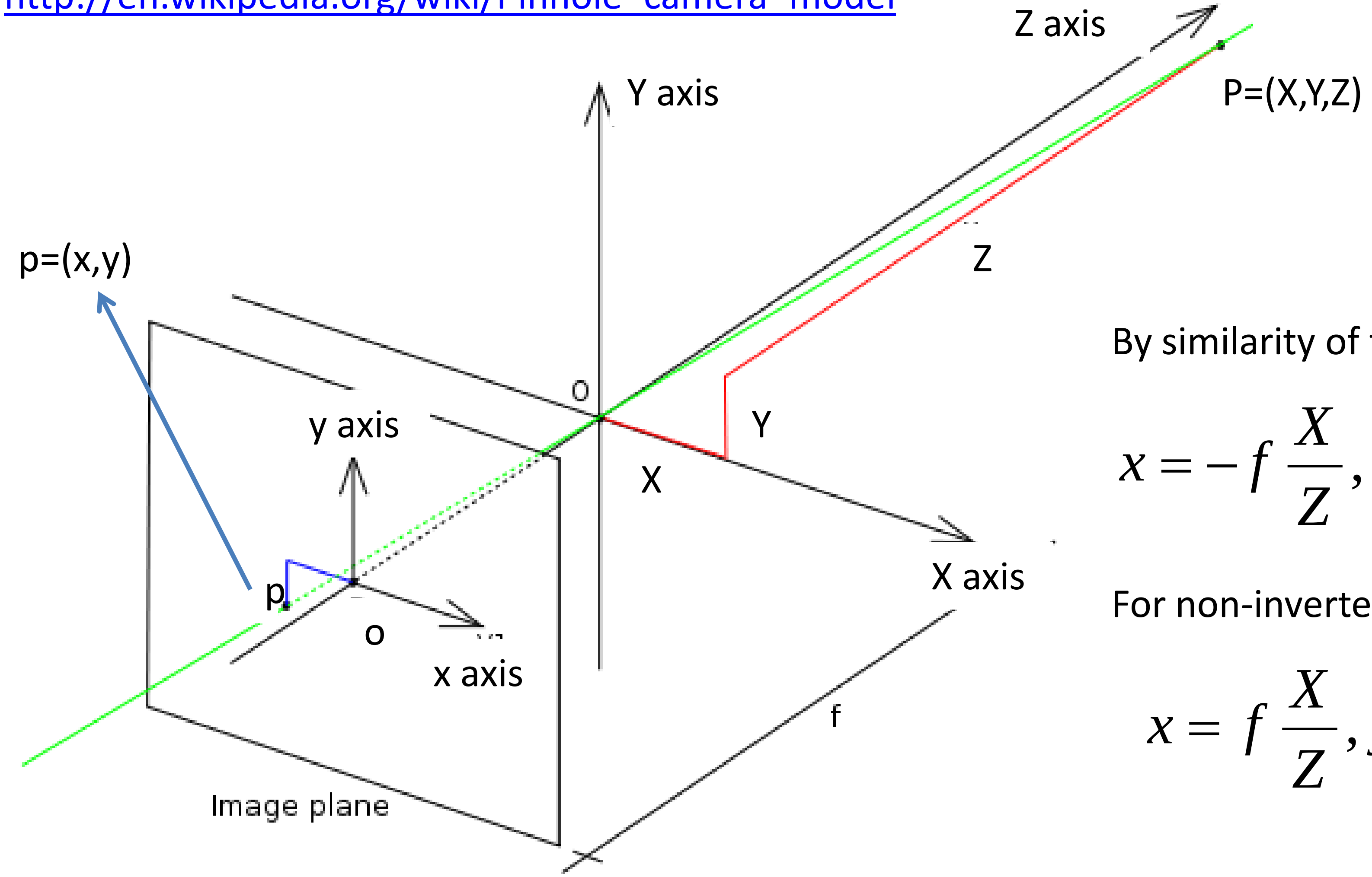


Image Formation in a Pinhole Camera



- **O** = center of projection of the camera (pinhole), also treated as origin of the coordinate system for the object ("camera coordinate system" or "camera frame")
- **P** = (X,Y,Z) = object point (3D vector)
- **p** = (x,y,z=*f*) image of **P** on the image plane π, Z axis is normal to π (i.e. π is parallel to XOY plane)
- Note: **p** is obtained by the intersection of line OP with π
- **o** = image center- given by the point where the vector normal to π and passing through **O** (this vector is the Z axis!) intersects π.
- **Line Oo** = optical axis of the camera lens (coincides with the Z axis)
- *f* = focal length = perpendicular distance from **O** onto the image plane π = length of segment Oo

http://en.wikipedia.org/wiki/Pinhole_camera_model



By similarity of triangles:

$$x = -f\frac{X}{Z}, y = -f\frac{Y}{Z}$$

For non-inverted image plane:

$$x = f\frac{X}{Z}, y = f\frac{Y}{Z}$$

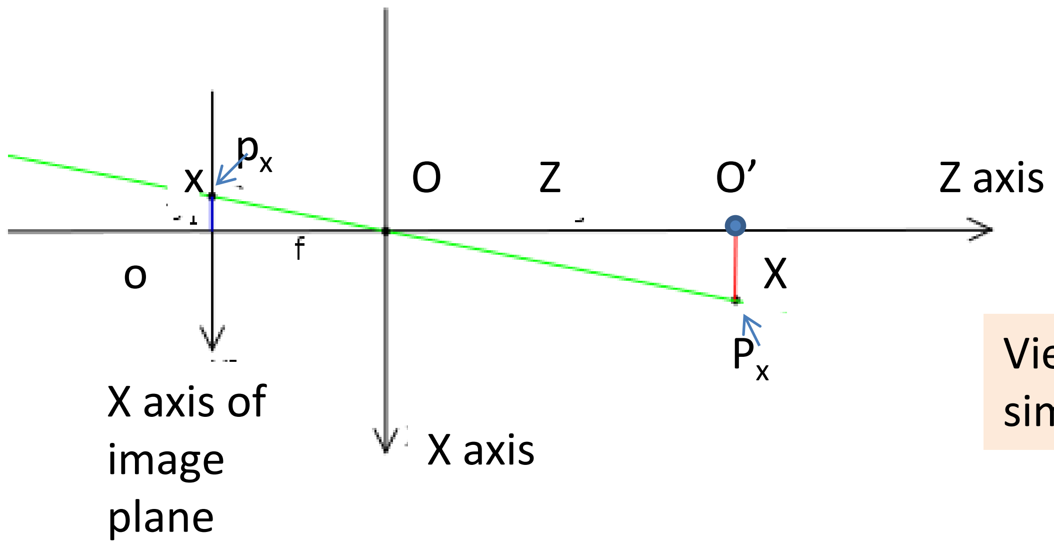
Which triangles are similar? Triangle Oop$_x$ and Triangle OO'P$_x$ (analogously Triangle Oop$_y$ and Triangle OO'P$_y$ are similar)

p$_x$ = projection of **p** onto XZ plane,

O' = Consider plane passing through **P** and parallel to image plane. O' is the intersection of the optical axis Oo with this plane.

P$_x$ = projection of **P** onto XZ plane.

http://en.wikipedia.org/wiki/Pinhole_camera_model



Viewing down the Y axis

$$x = -f\,\frac{X}{Z}$$

Viewing down the X axis will yield a similar figure and lead you to derive:

$$y = -f\,\frac{Y}{Z}$$

Which triangles are similar? Triangle Oop$_x$ and Triangle OO'P$_x$ (analogously Triangle Oop$_y$ and Triangle OO'P$_y$ are similar)
p$_x$ = projection of **p** onto XZ plane,
O' = Consider plane passing through P and parallel to image plane. O' is the intersection of the optical axis Oo with this plane.
P$_x$ = projection of **P** onto XZ plane.

Image Formation in a Pinhole Camera

- The projection **p** of point **P** is called a **perspective projection**.
- The relation between **p** = (*x*,*y*,*z*=*f*) and **P** = (*X*,*Y*,*Z*) is non-linear, as the Z coordinate will vary from point to point.
- This type of a transformation does not preserve angles, lengths or areas. It is also non-linear.
- In matrix form, we have:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f/Z & 0 & 0 \\ 0 & f/Z & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

Note that the entries in this matrix themselves depend on the Z coordinate of the object point(s). So this is a non-linear relationship!

Image Formation in a Pinhole Camera: Homogeneous Coordinates

- The relationship between (*x*,*y*) and (*X*,*Y*,*Z*) can also be expressed as follows using homogeneous coordinates:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix},$$

$$x = \frac{x'}{z'}, y = \frac{y'}{z'}$$

Pixels and Image points

- The object points project onto points in an image.
- The image represented in a camera is in discretized format, in the form of a 2D array.
- Thus the coordinates **p** = (x,y,z=*f*) are converted into discrete spatial coordinates in terms of pixels. The exact relationship depends upon the sampling rate of the camera **(pixel resolution)** and the **aspect ratio**.

$$x = -(x_{im} - o_x)s_x, \; y = -(y_{im} - o_y)s_y$$

Coordinates of optical center in terms of pixels

Pixel aspect ratio

Weak-perspective projection

- If the variation in depth (Z coordinate) of different points in the scene is negligible compared to the average depth *Z'* of the scene, then the projection is called **weak-perspective**:

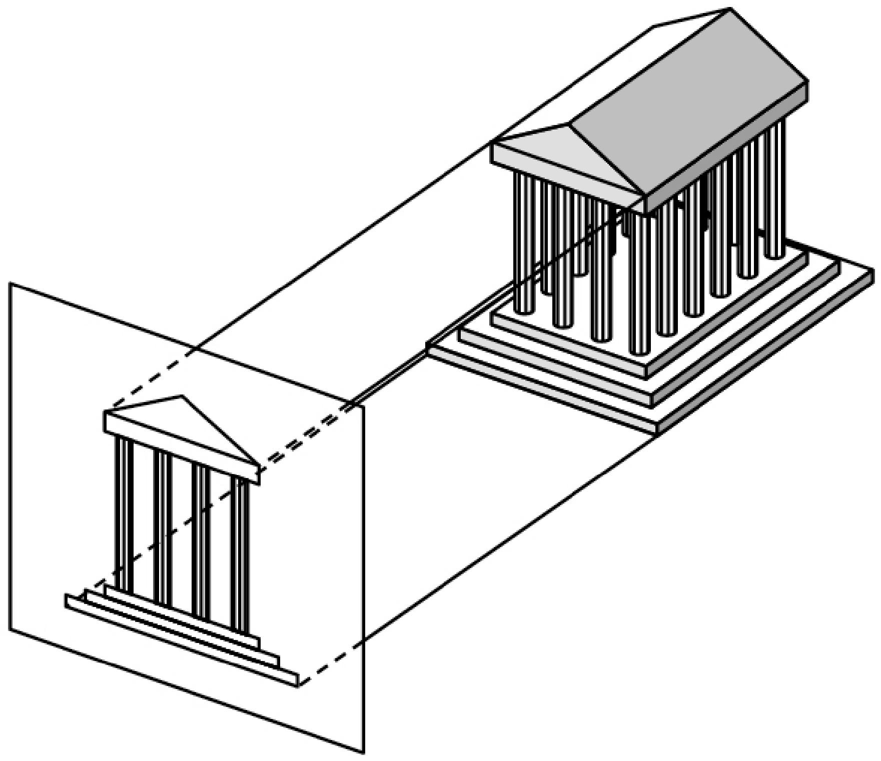
$$x = f\,\frac{X}{Z'+\delta} = f\,\frac{X}{Z'}\left(1-\delta/Z'+(\delta/Z')^2\right) \approx f\,\frac{X}{Z'},$$

$$y \approx f\,\frac{Y}{Z'}$$

Orthographic or Parallel Projection

- If the directions of projection are parallel to each other and orthogonal to the image plane, the projection of the 3D points onto 2D preserves angles, lengths and areas. Such a projection is called as **orthographic or parallel projection.**
- The weak perspective projection is a scaled form of an orthographic projection.
- Directions of projection are parallel implies infinite focal length (center of projection is very far away from the image plane).
- In such cases, $x = X$, $y = Y$.

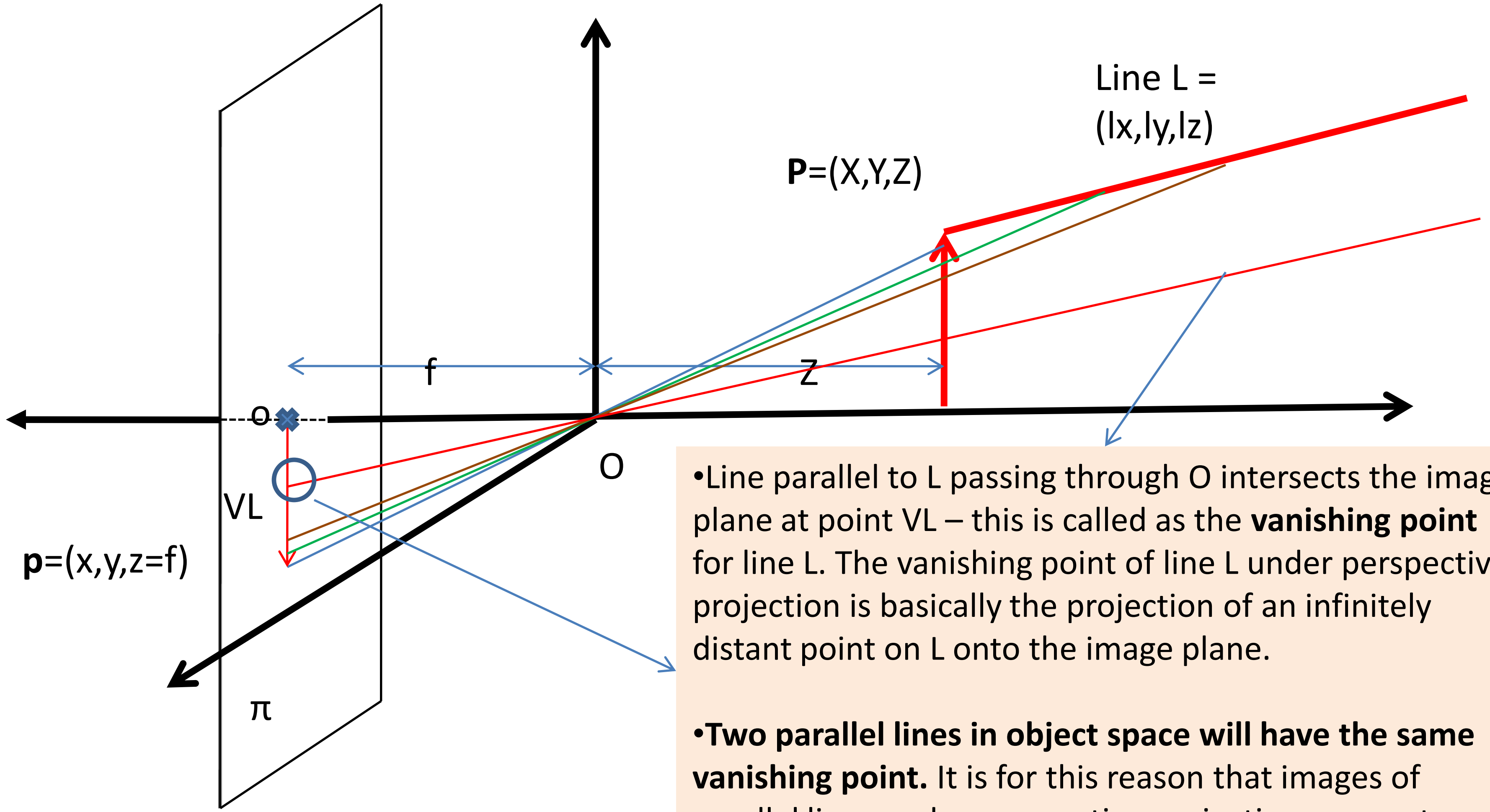
Orthographic or Parallel Projection



http://www.cs.cmu.edu/afs/cs/academic/class/15462-s09/www/lec/06/lec06.pdf

Vanishing Points

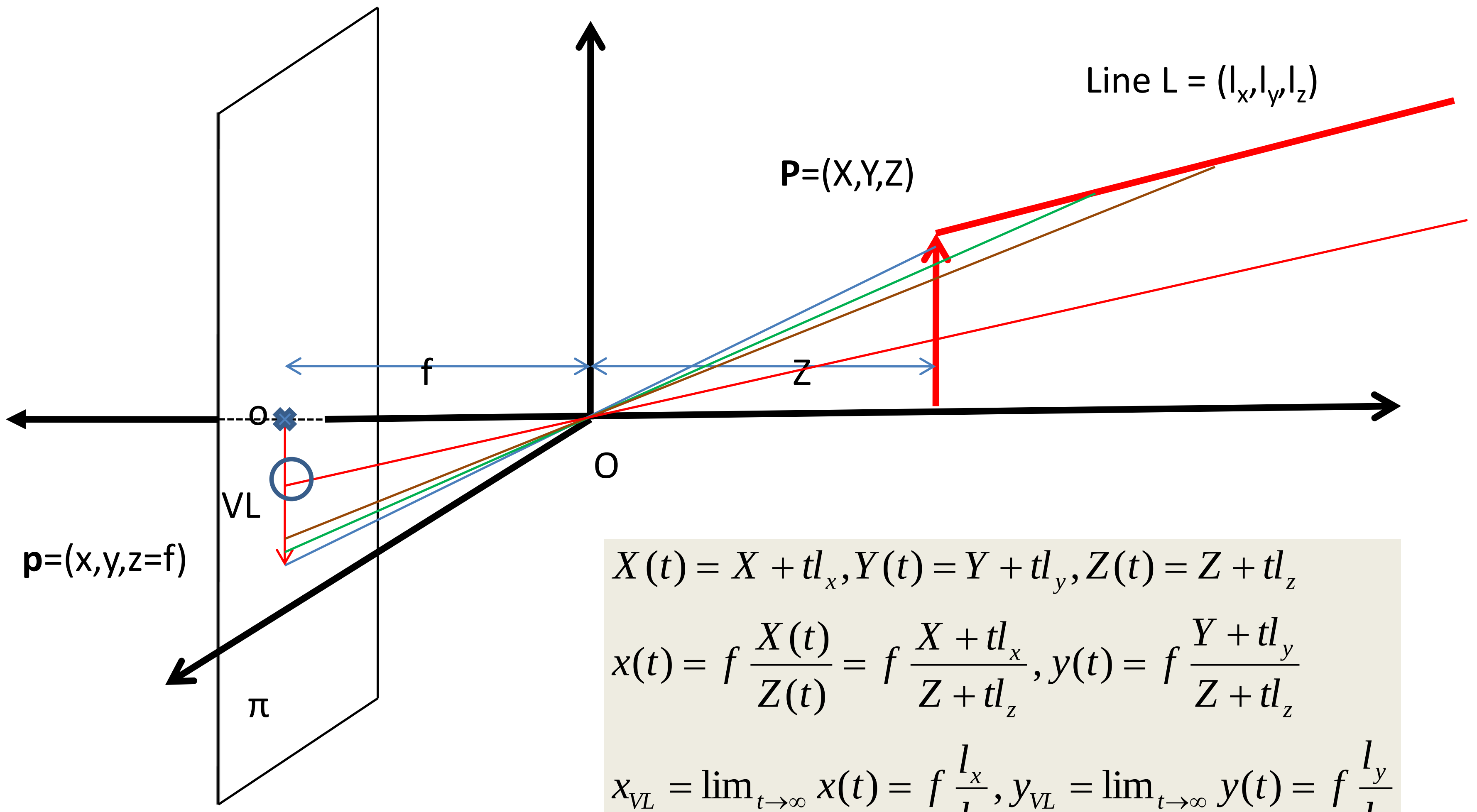
Vanishing Points



•Line parallel to L passing through O intersects the image plane at point VL – this is called as the **vanishing point** for line L. The vanishing point of line L under perspective projection is basically the projection of an infinitely distant point on L onto the image plane.

•**Two parallel lines in object space will have the same vanishing point.** It is for this reason that images of parallel lines under perspective projection appear to intersect.

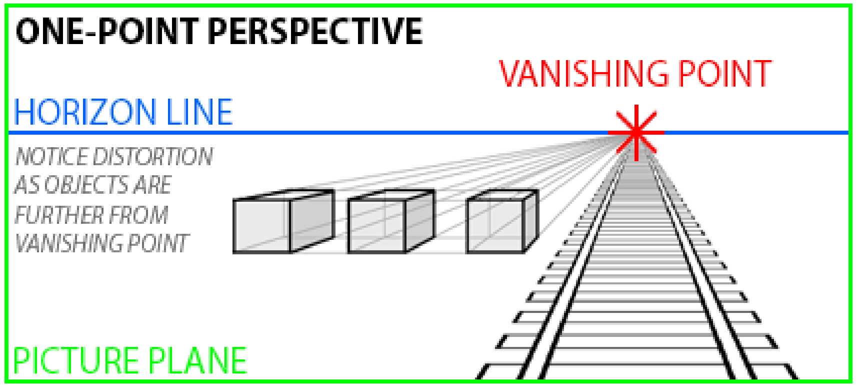
Vanishing Points



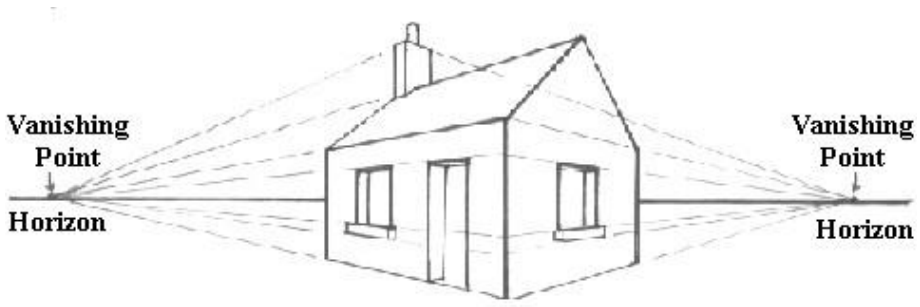
$$X(t) = X + tl_x, Y(t) = Y + tl_y, Z(t) = Z + tl_z$$

$$x(t) = f\frac{X(t)}{Z(t)} = f\frac{X + tl_x}{Z + tl_z}, y(t) = f\frac{Y + tl_y}{Z + tl_z}$$

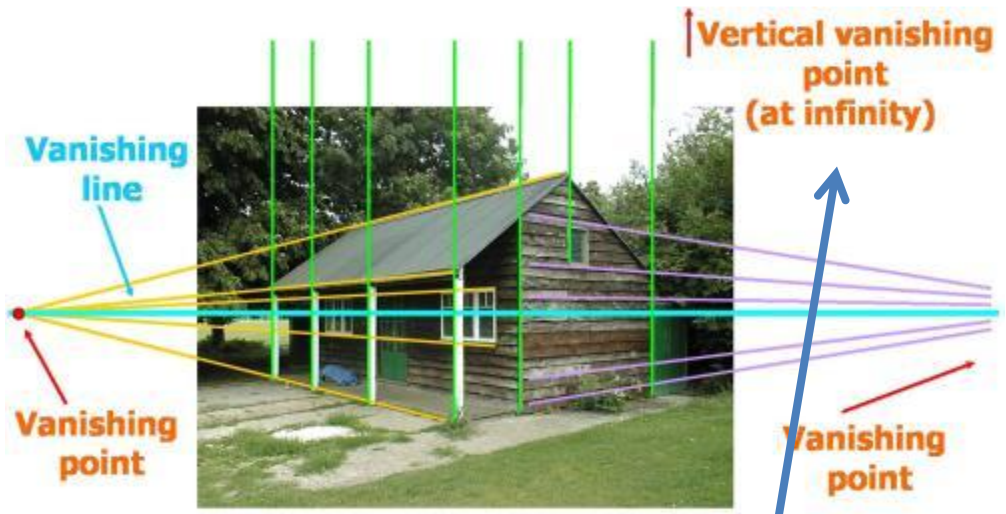
$$x_{VL} = \lim_{t\to\infty} x(t) = f\frac{l_x}{l_z}, y_{VL} = \lim_{t\to\infty} y(t) = f\frac{l_y}{l_z}$$


ONE-POINT PERSPECTIVE
VANISHING POINT
HORIZON LINE
NOTICE DISTORTION AS OBJECTS ARE FURTHER FROM VANISHING POINT
PICTURE PLANE

http://www.atpm.com/9.09/design.shtml



http://www.picturescape.co.uk/class%20pages/perspective%201.htm http://plus.maths.org/content/getting-picture



$$X(t) = X + tl_x, Y(t) = Y + tl_y, Z(t) = Z + tl_z$$

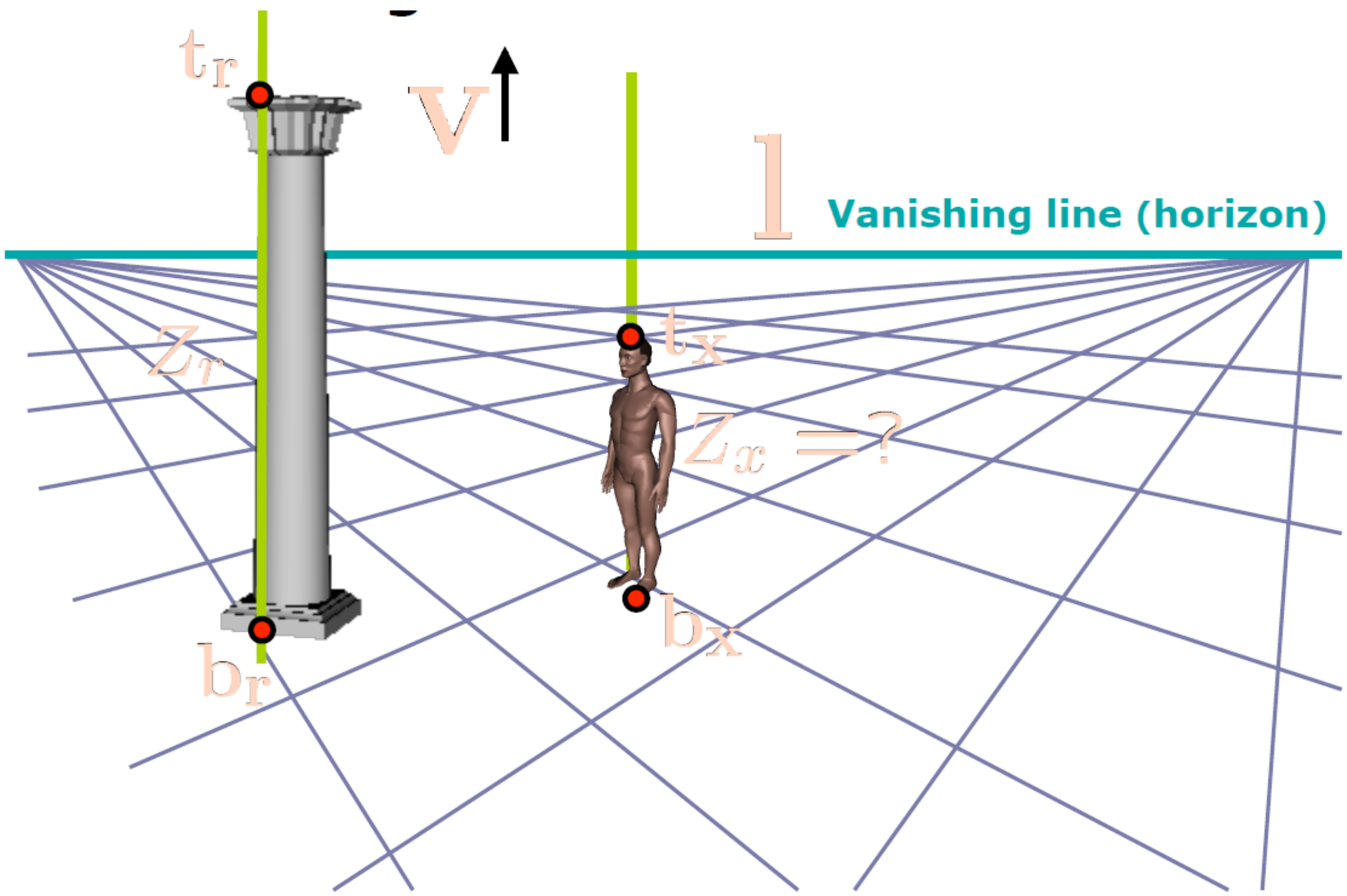
$$x(t) = f\frac{X(t)}{Z(t)} = f\frac{X + tl_x}{Z + tl_z}, y(t) = f\frac{Y + tl_y}{Z + tl_z}$$

$$x_{VL} = \lim_{t\to\infty} x(t) = f\frac{l_x}{l_z}, y_{VL} = \lim_{t\to\infty} y(t) = f\frac{l_y}{l_z}$$

If $l_z = 0$, it means the set of original parallel lines in 3D space lie in a plane parallel to the image plane (which is often the XY plane). In such cases, there is no vanishing point, or we have a vanishing point at infinity.

Vanishing Lines

- Consider two or more sets of parallel lines in 3D space lying on the same plane.
- The vanishing points of all these sets of lines are collinear (why?).
- This line containing all these vanishing points is called the **vanishing line**.
- If the plane involved is the ground plane, then the vanishing line is called the **horizon**.



From slides by Frank Dallaert

Camera Calibration

From 2D to 3D?

- Given pixel coordinates of some points in an image, we want to infer the 3D coordinates of the underlying object points.
- For this, we need to know (1) the relationship between the **camera coordinate system** and the “**world coordinate system”**, and (2) the relationship between **pixel coordinates** and the **actual coordinates of the 2D image points (in the camera coordinate system)**.

From 2D to 3D?

- What is the **world coordinate system**?
- It is a coordinate system chosen by the user (say the designer of an environment for a robot to travel, the architect of a building, etc).
- It is typically *different* from the camera coordinate system.

From 2D to 3D?

- The orientation and position of the camera coordinate system with respect to the world coordinate system is given by the **extrinsic camera parameters**.
- The relationship between the image coordinates and their representation in terms of pixels is given by the **intrinsic camera parameters**.

From 2D to 3D?

- The extrinsic and intrinsic parameters are unknown.
- They can be determined using a process called as **(geometric) camera calibration**.

Camera calibration: why?

- Given pictures from a calibrated camera (or a set of calibrated cameras), we can reconstruct parts of the underlying 3D scene.

- We can estimate various dimensions in 3D just given the 2D image(s).

- This is all apart from the fact that we get to estimate the camera resolution, focal length, etc. (in several cases, these parameters may be unknown!).

Camera calibration: how?

- To perform calibration, we take photographs of an object whose geometry is *known* and which will produce *salient feature points* in an image.

- Example: checkerboard cube. Such an object is called **calibration object**.

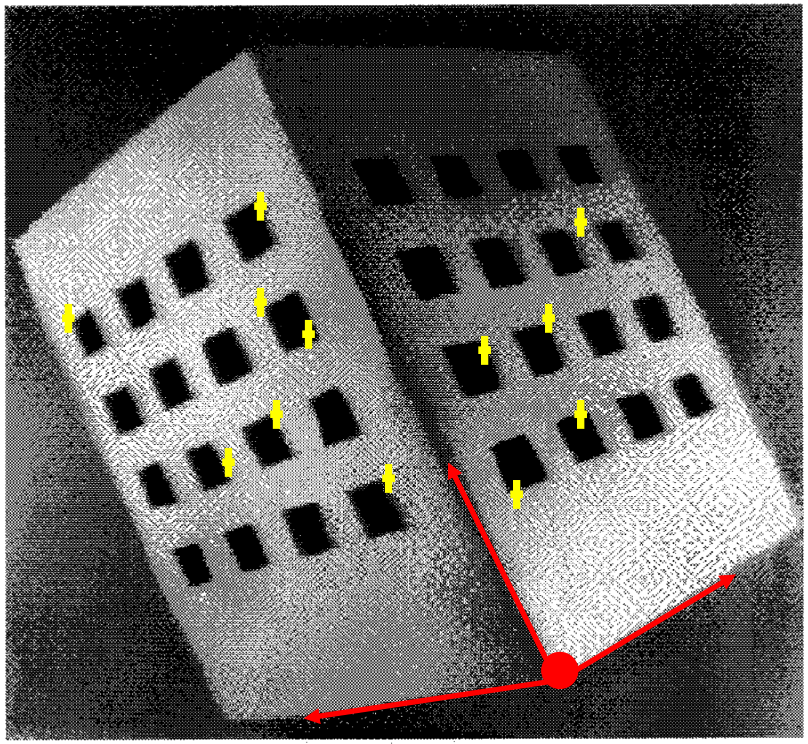


Figure 6.1 The typical calibration pattern used in this chapter.

We will measure the 3D coordinates of various **junction points** *(marked yellow)* of such a calibration object. Ideally, we would directly like to express these coordinates in terms of the origin of the **camera coordinate system** (i.e. with respect to the point **O** in the previous slides). But that is not physically possible, so we express the coordinates with respect to an origin **O'** that *we* choose. Example: it could be the bottom left corner of the checkerboard pattern (marked here as a red circle). This is the **world coordinate system**.

Secondly, we will measure the coordinates of the **pixel locations** of the various junction points from the image that we captured with the camera.

Thus for each of the *N* junction points, we have the pixel coordinates as well as the world coordinates. From this, we have to derive the extrinsic and intrinsic parameters of the camera.

Extrinsic parameters

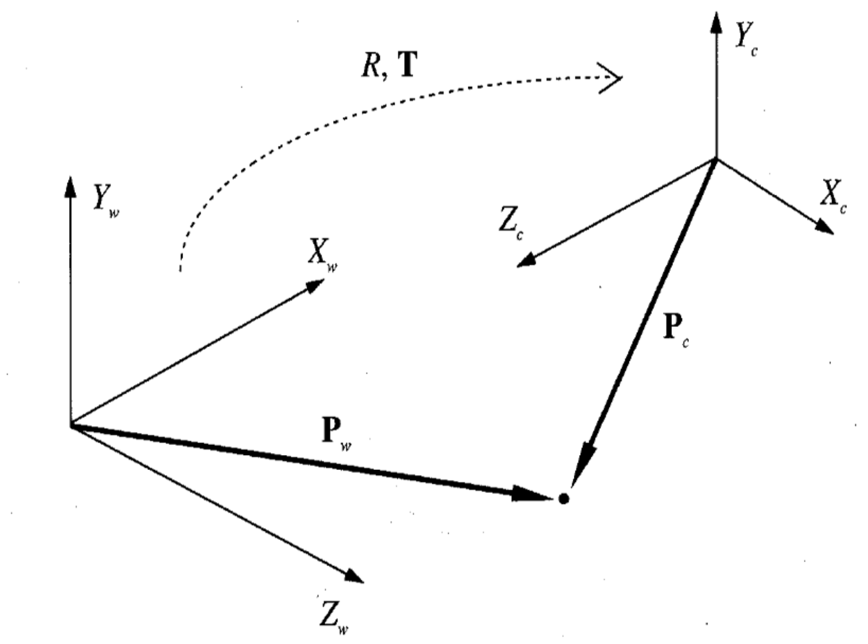


Figure 2.13 The relation between camera and world coordinate frames.

$$\mathbf{P_c} = \mathbf{R}(\mathbf{P_w} - \mathbf{t}),$$

$$\mathbf{t} = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}, \underbrace{\mathbf{P_c}}_{\text{Coordinates of physical point }\mathbf{P}\text{ in terms of camera coordinate system}} = \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix}, \underbrace{\mathbf{P_w}}_{\text{Coordinates of physical point }\mathbf{P}\text{ in terms of world coordinate system (this is measured physically, }e.g.\text{ with a measuring tape)}} = \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix}$$

The extrinsic camera parameters are typically unknown. They account for 6 degrees of freedom (i.e. 6 unknowns) – why?

Extrinsic parameters

- Note: the world coordinate system and the camera coordinate system are not aligned.
- **R** and **t** represent the rotation and translation (respectively) that takes you from the world coordinate system to the camera coordinate system.

Intrinsic parameters

- The relationship between image coordinates and pixel coordinates is given as:

$$x = -(x_{im} - o_x)s_x, \; y = -(y_{im} - o_y)s_y$$

Coordinates of optical center in terms of pixels

Pixel aspect ratio

- The other intrinsic camera parameter is the focal length *f*.
- Intrinsic parameters are also unknown! They account for 5 degrees of freedom.

Three main equations and a new one

$$x = -(x_{im} - o_x)s_x, y = -(y_{im} - o_y)s_y$$

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \mathbf{R}\begin{bmatrix} X_w - t_x \\ Y_w - t_y \\ Z_w - t_z \end{bmatrix}$$

$$x = f\frac{X_c}{Z_c}, y = f\frac{Y_c}{Z_c}$$

$$-(x_{im} - o_x)s_x = f\frac{X_c}{Z_c}$$

$$-(y_{im} - o_y)s_y = f\frac{Y_c}{Z_c}$$

$$\mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_1 \\ \mathbf{R}_2 \\ \mathbf{R}_3 \end{pmatrix}$$

Row of the matrix **R**

$$-(x_{im} - o_x)s_x = f\frac{\mathbf{R}_1(\mathbf{P}_w - \mathbf{t})}{\mathbf{R}_3(\mathbf{P}_w - \mathbf{t})}$$

$$-(y_{im} - o_y)s_y = f\frac{\mathbf{R}_2(\mathbf{P}_w - \mathbf{t})}{\mathbf{R}_3(\mathbf{P}_w - \mathbf{t})}$$

Projection equation in matrix form

$$-(x_{im}-o_x)s_x = f\frac{\mathbf{R}_1(\mathbf{P_w}-\mathbf{t})}{\mathbf{R}_3(\mathbf{P_w}-\mathbf{t})}$$

$$-(y_{im}-o_y)s_y = f\frac{\mathbf{R}_2(\mathbf{P_w}-\mathbf{t})}{\mathbf{R}_3(\mathbf{P_w}-\mathbf{t})}$$

Intrinsic camera matrix (3 x 3)

Extrinsic camera matrix (3 x 4)

M_{int} M_{ext}

$$\begin{pmatrix}\hat{x}\\ \hat{y}\\ \hat{z}\end{pmatrix} = \underbrace{\underbrace{\begin{pmatrix}-\frac{f}{s_x} & 0 & o_x\\ 0 & -\frac{f}{s_y} & o_y\\ 0 & 0 & 1\end{pmatrix}}_{M_{\text{int}}}\underbrace{\begin{pmatrix}r_{11} & r_{12} & r_{13} & -\mathbf{R}_1\mathbf{t}\\ r_{21} & r_{22} & r_{23} & -\mathbf{R}_2\mathbf{t}\\ r_{31} & r_{32} & r_{33} & -\mathbf{R}_3\mathbf{t}\end{pmatrix}}_{M_{ext}}}_{\mathbf{M}}\begin{pmatrix}X_w\\ Y_w\\ Z_w\\ 1\end{pmatrix},$$

$$x_{im} = \frac{\hat{x}}{\hat{z}}, y_{im} = \frac{\hat{y}}{\hat{z}}$$

M = 3 x4 camera matrix

Camera Calibration

$$\forall i, 1 \le i \le N, \begin{pmatrix} \hat{x}_i \\ \hat{y}_i \\ \hat{z}_i \end{pmatrix} = \begin{pmatrix} -\frac{f}{s_x} & 0 & o_x \\ 0 & -\frac{f}{s_y} & o_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & -\mathbf{R}_1\mathbf{t} \\ r_{21} & r_{22} & r_{23} & -\mathbf{R}_2\mathbf{t} \\ r_{31} & r_{32} & r_{33} & -\mathbf{R}_3\mathbf{t} \end{pmatrix} \begin{pmatrix} X_{w,i} \\ Y_{w,i} \\ Z_{w,i} \\ 1 \end{pmatrix},$$

$$x_{im,i} = \frac{\hat{x}_i}{\hat{z}_i}, y_{im,i} = \frac{\hat{y}_i}{\hat{z}_i}$$

$$\text{Given} \left\{ \begin{pmatrix} x_{im,i} \\ y_{im,i} \end{pmatrix} \right\}_{i=1}^{N} \text{ and } \left\{ \begin{pmatrix} X_{w,i} \\ Y_{w,i} \\ Z_{w,i} \end{pmatrix} \right\}_{i=1}^{N}, \text{estimate } f, s_x, s_y, o_x, o_y, \mathbf{R}, \mathbf{t}, \text{i.e., 11 unknowns.}$$

Camera Calibration

- Consider the equations, two for each of the *N* points:

$$x_i = x_{im,i} - o_x = \frac{-f}{s_x}\frac{\mathbf{R}_1\mathbf{P}_{\mathbf{w,i}} + t_x}{\mathbf{R}_3\mathbf{P}_{\mathbf{w,i}} + t_z} = -f_x \frac{r_{11}X_{w,i} + r_{12}Y_{w,i} + r_{13}Z_{w,i} + t_x}{r_{31}X_{w,i} + r_{32}Y_{w,i} + r_{33}Z_{w,i} + t_z}$$

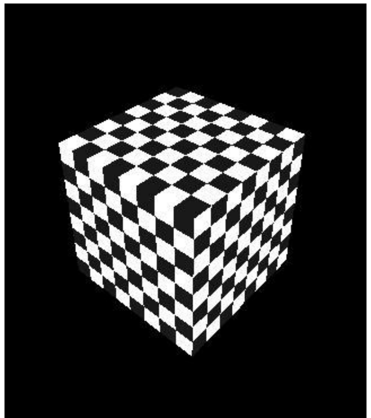
$$y_i = y_{im,i} - o_y = \frac{-f}{s_y}\frac{\mathbf{R}_2\mathbf{P}_{\mathbf{w,i}} + t_x}{\mathbf{R}_3\mathbf{P}_{\mathbf{w,i}} + t_z} = -f_y \frac{r_{21}X_{w,i} + r_{22}Y_{w,i} + r_{23}Z_{w,i} + t_y}{r_{31}X_{w,i} + r_{32}Y_{w,i} + r_{33}Z_{w,i} + t_z}$$

Note: In the above equations, we have $t_x = -\mathbf{R}_1.\mathbf{t}$, $t_y = -\mathbf{R}_2.\mathbf{t}$, $t_z = -\mathbf{R}_3.\mathbf{t}$

- Refer to lecture scans and textbook by Trucco and Verri (section 6.1 and 6.2.1) for the two different calibration procedures.

Camera calibration: Image center

- The first calibration procedure mentioned in the lectures yields all parameters except for the coordinates of the image center, given by (o_x, o_y).
- To find the image center, we compute the vanishing points of three mutually perpendicular sets of lines. The three vanishing points form a triangle, and the image center turns out to be the **orthocenter** of that triangle (i.e. point where the three **altitudes** of the triangle meet)!
- The second calibration procedure determines the image center as well.



First procedure (by Tsai, sections 6.2.1 to 6.2.3 of Trucco and Verri)

- It explicitly finds the camera parameters during calibration.
- Given $N \geq 8$ pairs of points, it sets up an equation of the form $\mathbf{Av} = \mathbf{0}$ where $\mathbf{A}$ has size N x 8 and contains functions of known coordinates, and $\mathbf{v}$ is an 8 x 1 vector of unknown values.
- The solution $\mathbf{v}$ is obtained (up to an unknown scalar and sign) as the eigenvector of $\mathbf{A}^T\mathbf{A}$ with the least (or zero) eigenvalue. (In the noise-free case, $\mathbf{A}$ has rank 7.)
- The camera parameters are derived from $\mathbf{v}$ as mentioned in the book.
- The optical center (o_x,o_y) is separately estimated using the orthocenter property of vanishing points, and this estimate must be given as input to this method.

Second procedure (by Faugeras and Toscani, section 6.3 of Trucco and Verri)

- It seeks to directly find the camera matrix **M** (which is a 3 x 4 matrix).
- The camera matrix **M** has only 11 degrees of freedom (why?).
- The camera matrix is found by setting up an equation of the form **Am** = **0** where m is a 12 x 1 vector containing the entries of **M**, and **A** is a 2*N* x 12 matrix.
- Vector **m** is computed (up to an unknwon scale and sign) as the eigenvector of $\mathbf{A}^T\mathbf{A}$ with the least (ideally zero) eigenvalue.

Second procedure (by Faugeras and Toscani, section 6.3 of Trucco and Verri)

- Vector **m** is reshaped to get a 3 x 4 matrix **M**.
- The unknown scale and sign need not bother us (why?)
- What if we wanted to find the intrinsic and extrinsic parameters?
- The book by Trucco and Verri gives one method for this.
- In the next slide, we will see another method!

Second procedure (by Faugeras and Toscani)

- Recall that the camera matrix is given as:

$$\mathbf{M} = \begin{pmatrix} -\frac{f}{s_x} & 0 & o_x \\ 0 & -\frac{f}{s_y} & o_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & -\mathbf{R}_1\mathbf{t} \\ r_{21} & r_{22} & r_{23} & -\mathbf{R}_2\mathbf{t} \\ r_{31} & r_{32} & r_{33} & -\mathbf{R}_3\mathbf{t} \end{pmatrix}$$

Upper triangular!

$$= \mathbf{M}_{int}\mathbf{M}_{ext} = \mathbf{M}_{int}(\mathbf{R} \mid -\mathbf{R}\mathbf{t}) = (\mathbf{M}_{int}\mathbf{R} \mid -\mathbf{M}_{int}\mathbf{R}\mathbf{t}) = (\mathbf{G} \mid \mathbf{g})$$

- Consider the RQ decomposition of the 3 x 3 submatrix **G**, **G** = $\mathbf{G}_u\mathbf{G}_o$ where $\mathbf{G}_u$ is *upper triangular* and $\mathbf{G}_o$ is *orthonormal*.

Second procedure (by Faugeras and Toscani)

- This RQ decomposition always exists and is unique for a full-rank matrix (which G is).
- The RQ decomposition is derived from the more popular QR decomposition – which also always exists for any matrix, and is unique for a full-rank matrix.

Code (Text):

```
ReverseRows = [0 0 1; 0 1 0 ; 1 0 0]; [Q R] = qr((ReverseRows * A)'); R =
ReverseRows * R' * ReverseRows; Q = ReverseRows * Q';
```

https://www.physicsforums.com/threads/rq-decomposition-from-qr-decomposition.261739/

Second procedure (by Faugeras and Toscani)

- The aforementioned RQ decomposition can be used to determine $\mathbf{M}_{\mathbf{int}}$ and **R**.
- The translation vector $\mathbf{t} = \mathbf{R}^T(\mathbf{M}_{\mathbf{int}})^{-1}\mathbf{g}$ (see previous slides).
- This method gives us f_x, f_y, o_x and o_y as well as **R** and **t**. Notice that (o_x, o_y) need not be separately estimated unlike method 1.

Using Calibrated Cameras

Determining depth

- Let $\mathbf{p}_1$ be the image of point **P** captured using a **calibrated** camera C_1 with 3 x 4 **known** projection matrix $\mathbf{M}_1$. Hence we have:

$$\mathbf{p}_1 = \begin{pmatrix} \hat{x}_1 \\ \hat{y}_1 \\ \hat{z}_1 \end{pmatrix} = \mathbf{M}_1 \mathbf{P}_w = \mathbf{M}_1 \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}, u_1 = \frac{\hat{x}_1}{\hat{z}_1}, v_1 = \frac{\hat{y}_1}{\hat{z}_1}$$

- Given just the image point coordinates, we can only determine the 3D direction on which the object point lies. This direction is given as (u_1, v_1, f), where:

$$u_1 = \frac{\hat{x}_1}{\hat{z}_1}, v_1 = \frac{\hat{y}_1}{\hat{z}_1}$$

Determining depth

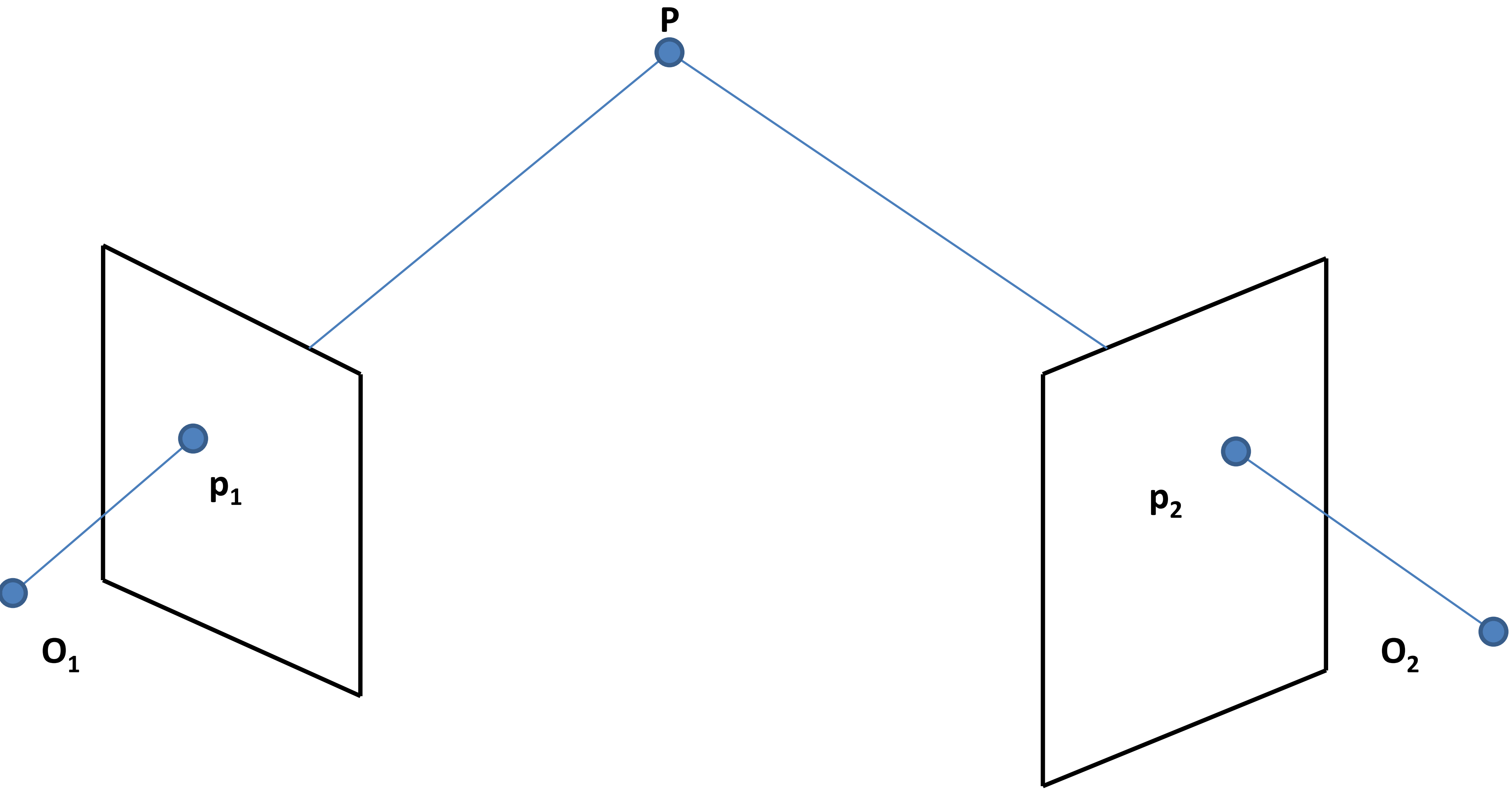
- To determine the exact location, we will need *another calibrated* camera. Let $\mathbf{p_2}$ be the image of the <u>*same*</u> point **P** captured using *calibrated* camera C_2 with 3 x 4 *known* projection matrix $\mathbf{M_2}$. Hence we have:

$$\mathbf{p_2} = \begin{pmatrix} \hat{x}_2 \\ \hat{y}_2 \\ \hat{z}_2 \end{pmatrix} = \mathbf{M_2 P_w} = \mathbf{M_2} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}, u_2 = \frac{\hat{x}_2}{\hat{z}_2}, v_2 = \frac{\hat{y}_2}{\hat{z}_2}$$

- We again determine the 3D direction on which the object point lies. This direction is given as *(u_2, v_2, f)*, where:

$$u_2 = \frac{\hat{x}_2}{\hat{z}_2}, v_2 = \frac{\hat{y}_2}{\hat{z}_2}$$

- **The intersection of the two directions yields the 3D coordinate of the point.**

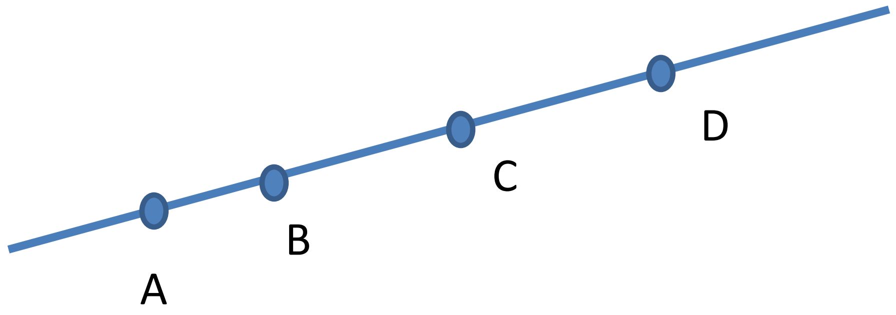

P
p_1
p_2
O_1
O_2

Cross-Ratios: Projective Invariants

Measuring Height using Cross-Ratios

- Consider four collinear points A, B, C, D (in that order). The following quantity is called the cross-ratio of A, B, C, D:

$$cross-ratio(A, B, C, D) = \frac{AC \times BD}{AD \times BC} = \frac{AC / BC}{AD / BD}$$

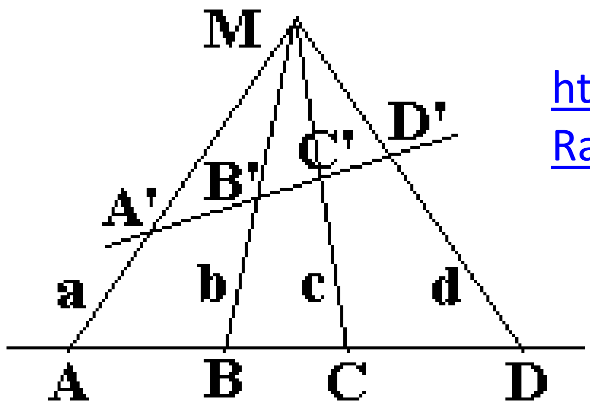


Cross-Ratio: Projective invariant

- Cross-ratios are invariant under perspective projection, i.e. the cross ratio of four collinear in 3D = cross ratio of their projections (i.e., images) onto a 2D image plane!
- We will prove this algebraically in class! A geometric proof is on the next slide.
- Recall – lengths, areas, ratios of lengths, ratios of areas are not preserved under perspective projection (ratios of lengths and areas are preserved under affine transformation).

Cross ratio: projective invariant

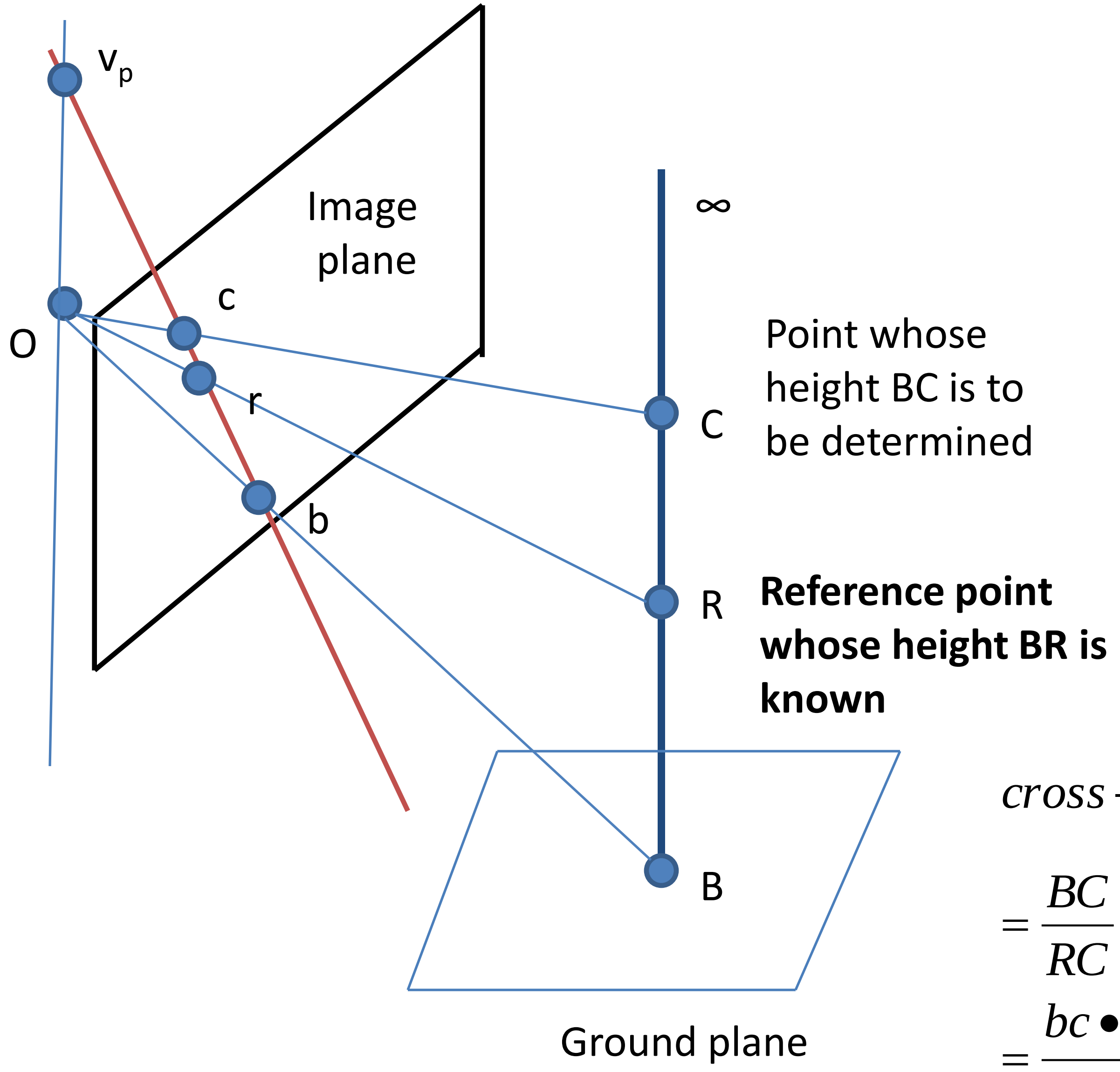
- There exists a beautiful theorem in geometry which says that the cross ratio of the points (A,B,C,D) and that of the points (A’,B’,C’,D’) in the figure below are equal.



http://www.cut-the-knot.org/pythagoras/Cross-Ratio.shtml

Cross ratio: projective invariant

- Now consider 4 collinear points A’,B’,C’,D’ in 3D space whose images on the image plane as A,B,C,D. Then clearly lines AA’, BB’, CC’, DD’ intersect at point M which is the center of projection.
- Invoking the theorem proves our result!


v_p
Image plane
c
O
r
b
∞
C
Point whose height BC is to be determined
R
Reference point whose height BR is known
B
Ground plane

$$cross-ratio(B,R,C,\infty) = \frac{BC \times R\infty}{B\infty \times RC}$$

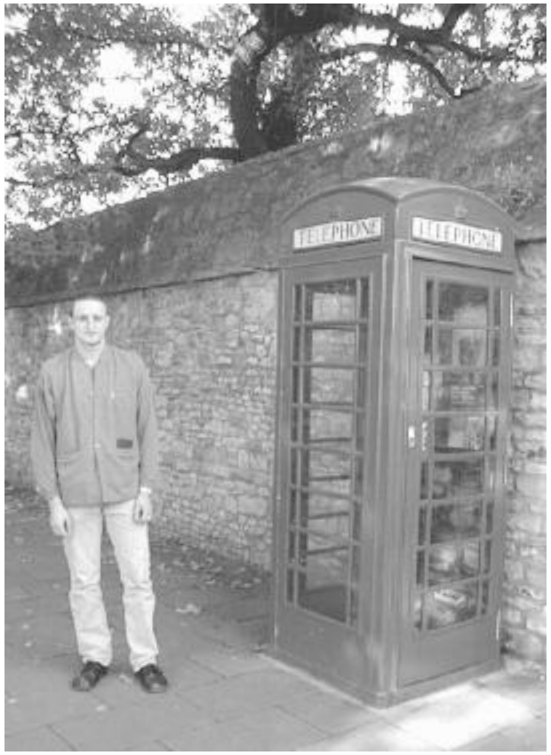
$$= \frac{BC}{RC} = \frac{BR + RC}{RC}$$

$$= \frac{bc \bullet rv_p}{bv_p \bullet rc}$$

RC is unknown – rest are known

Measuring Height using Cross-Ratios

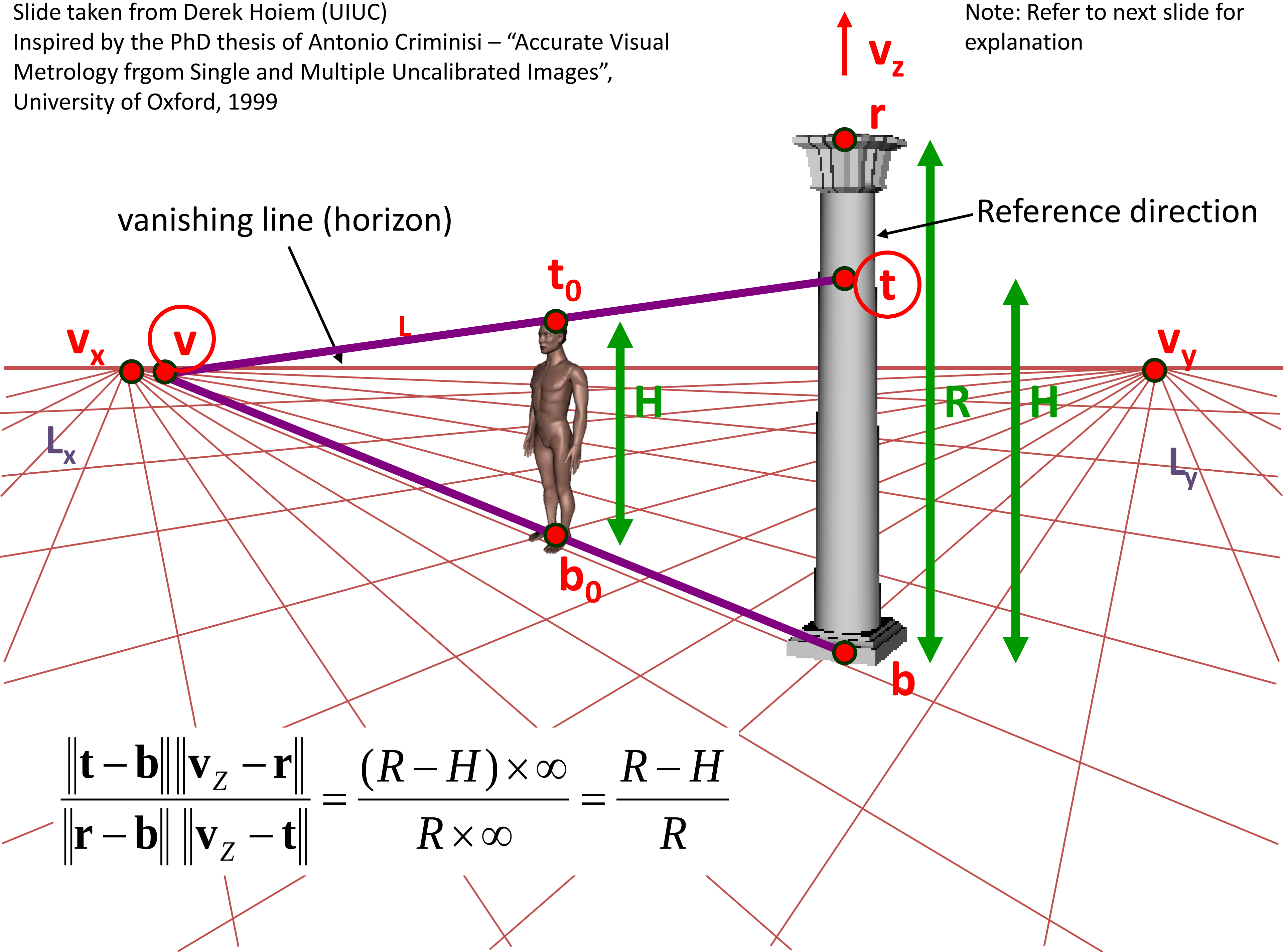
Slide taken from Derek Hoiem (UIUC)
Inspired by the PhD thesis of Antonio Criminisi – "Accurate Visual Metrology frgom Single and Multiple Uncalibrated Images", University of Oxford, 1999



How tall is this person ?

Slide taken from Derek Hoiem (UIUC)
Inspired by the PhD thesis of Antonio Criminisi – “Accurate Visual Metrology frgom Single and Multiple Uncalibrated Images”, University of Oxford, 1999

Note: Refer to next slide for explanation



$$\frac{\|\mathbf{t}-\mathbf{b}\|\|\mathbf{v}_Z-\mathbf{r}\|}{\|\mathbf{r}-\mathbf{b}\|\|\mathbf{v}_Z-\mathbf{t}\|}=\frac{(R-H)\times\infty}{R\times\infty}=\frac{R-H}{R}$$

- **Input**: an image of a person standing next to a building or a structure whose height *R* is known.
- **To determine**: height of the person (*H*) from the image.
- **Assumption 1**: the building is in a direction (called *reference direction*) not parallel to the image plane – so that its vanishing point exists (called $\mathbf{v}_z$)
- **Assumption 2**: We are able to find two sets of parallel lines (called $\mathbf{G}_1$ and $\mathbf{G}_2$) on the ground plane.
- Their corresponding vanishing points form the *horizon*.
- Points $\mathbf{b}$ and $\mathbf{b}_0$ are both on the ground plane. Consider a line (say $\mathbf{L}$) passing through $\mathbf{t}_0$ and parallel to line $\mathbf{bb}_0$.
- The images of $\mathbf{L}$ and $\mathbf{bb}_0$ will intersect at a point on the horizon (why? Because $\mathbf{bb}_0$ is coplanar with $\mathbf{G}_1$ and $\mathbf{G}_2$, and vanishing points of coplanar lines are collinear). Let's call this point $\mathbf{v}$.

- Now line $\mathbf{vt}_0$ (i.e. **L**) is parallel to $\mathbf{bb}_0$ in 3D space. Also line $\mathbf{vt}_0$ intersects the reference direction at point **t**.
- Hence in 3D space, $\mathbf{tb} = \mathbf{t}_0\mathbf{b}_0$ = unknown height *H* (on the image plane, of course their lengths are different).
- Now use the cross-ratio invariance property to find *H*:

$$\frac{\|\mathbf{t}-\mathbf{b}\|\|\mathbf{v}_Z-\mathbf{r}\|}{\|\mathbf{r}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{t}\|}=\frac{R-H}{R}$$

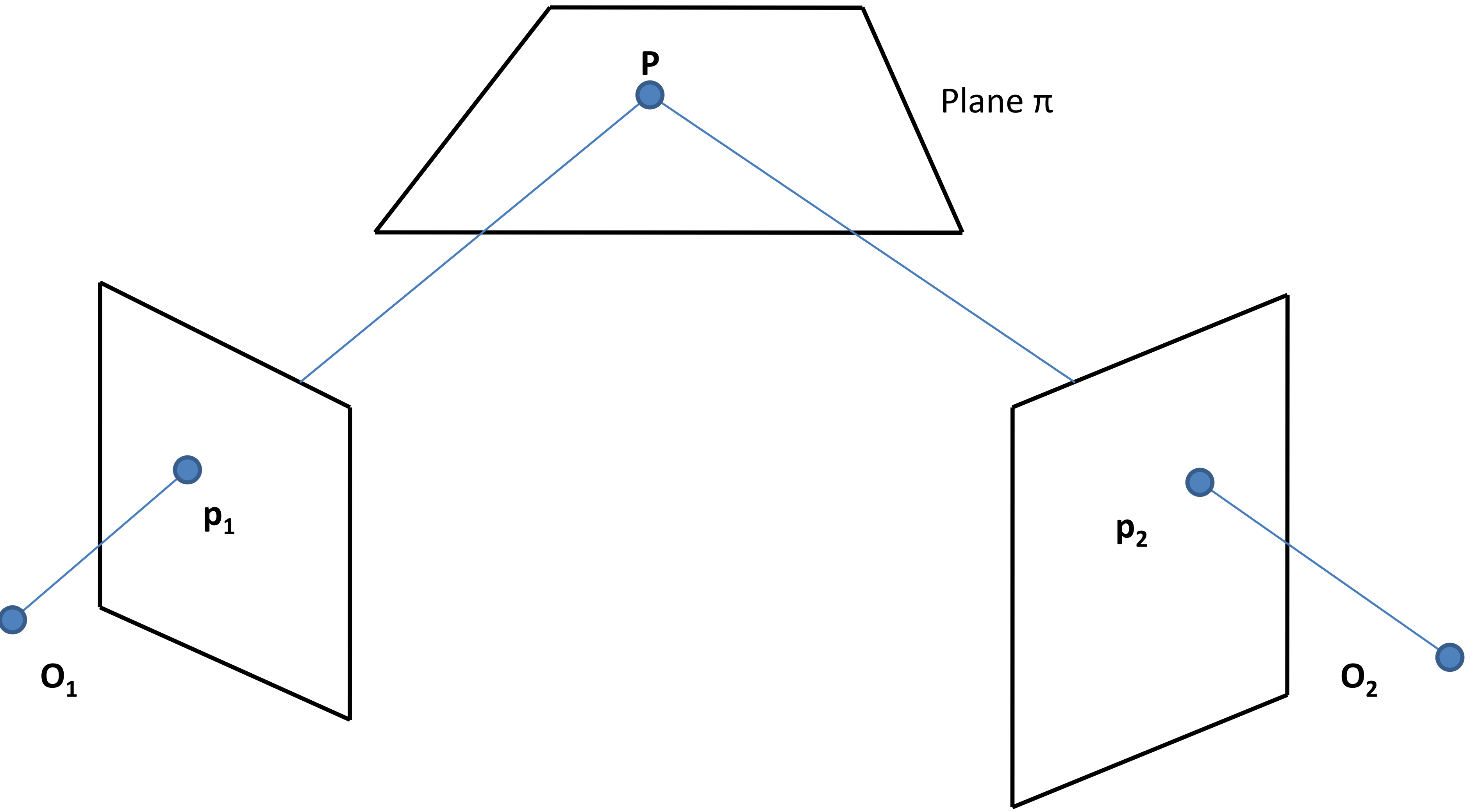
Algorithm

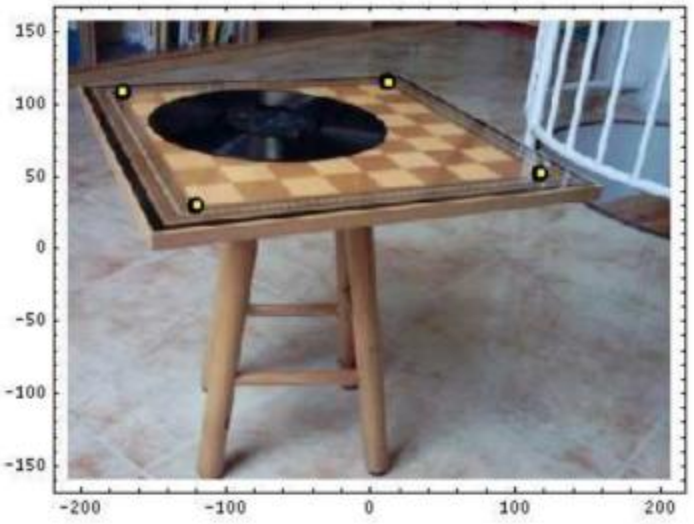
- Find the vanishing point in the reference direction $\mathbf{v}_z$.
- Find the horizon line by joining the vanishing points of two pairs of parallel lines on the ground plane.
- Find the point of intersection (**v**) of the horizon with the line joining the base of the building (**b**) and the point where the person is standing ($\mathbf{b}_0$).
- Find the point of intersection (**t**) of the line $\mathbf{v}\mathbf{t}_0$ ($\mathbf{t}_0$ = person's head) with reference direction.
- Use the cross-ratio formula to determine the height of the person.

Planar Homography

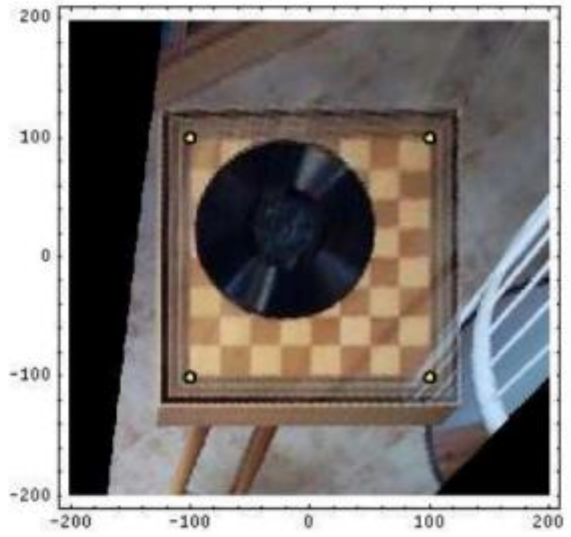
Planar Homography

- Consider a planar object in 3D space, imaged by two cameras.
- The coordinates of corresponding points in the two images are said to be related by a **planar homography**. This relationship can be determined **without** calibrating either camera.


P
Plane π
p_1
p_2
O_1
O_2



150
100
50
0
-50
-100
-150
-200
-100
0
100
200



200
100
0
-100
-200
-200
-100
0
100
200

Whole Grain Guaranteed
Cheerios
send Cheer to military families
thank you!
Toasted Whole Grain Oat Cereal
NET WT 14 OZ (396g)

Cheerios
send Cheer to military families
Toasted Whole Grain Oat Cereal
NET WT 14 OZ (396g)
Cheerios

Planar Homography (with homogeneous coordinates)

- Let equation of plane π be *aX + bY +cZ + 1 = 0*, i.e. [**N** 1]**P** = 0 where **N** = (a,b,c) is the vector (in 3D) normal to the plane and **P** = $(X,Y,Z,1)^t$ in the coordinate system of the *first camera*.
- The coordinates of its image in the first camera (in coordinate system of *first camera*) are given by:

$$\mathbf{p}_1 = \begin{pmatrix} u_1 \\ v_1 \\ fw_1 \end{pmatrix} = \mathbf{M}_1 \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \mathbf{M}_1\mathbf{P}$$

$\mathbf{M}_1$ is the extrinsic (3x4) matrix for the first camera. Note that the image coordinates corresponding to **P** are $(u_1/w_1, v_1/w_1, f)$ which in a directional sense is equivalent to $(u_1/(fw_1), v_1/(fw_1), 1)$.

Planar Homography (with homogeneous coordinates)

- Clearly **P** lies on a ray from the pinhole (origin) to the point $\mathbf{p}_1$ in 3D space. Hence we can say that

$$\mathbf{P} = \begin{pmatrix} u_1 \\ v_1 \\ fw_1 \\ k \end{pmatrix} \Rightarrow au_1 + bv_1 + cfw_1 + k = 0 \Rightarrow \mathbf{N}\mathbf{p}_1 = -k \Rightarrow \mathbf{P} = \boxed{\begin{pmatrix} \mathbf{I} \\ -\mathbf{N} \end{pmatrix}} \mathbf{p}_1$$

I has size 3 x 3, **N** has size 1 x 3, so it is a 4 x 3 matrix

The (X,Y,Z) coordinates of P can be expressed as (u_1/k,v_1/k,fw_1/k). The homogeneous coordinate '*k*' can be interpreted as a parameter which tells you **exactly where** on the line from the pinhole to $\mathbf{p}_1$, our point **P** lies. In general, this *k* would be unknown, but here we have more information – that P lies on a plane whose equation is known. This would yield $au_1/k + bv_1/k + cfw_1/k + 1 = 0$, i.e. $au_1 + bv_1 + cfw_1 + k = 0$.

Planar Homography (with homogeneous coordinates)

- For the second camera, we have:

$$\mathbf{p}_2 = \begin{pmatrix} u_2 \\ v_2 \\ fw_2 \end{pmatrix} = \mathbf{M}_2 \begin{pmatrix} X' \\ Y' \\ Z' \\ 1 \end{pmatrix} = \mathbf{M}_2 \mathbf{M}_{1,2} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$= \tilde{\mathbf{M}}_2 \mathbf{P} = \tilde{\mathbf{M}}_2 \begin{pmatrix} \mathbf{I} \\ -\mathbf{N} \end{pmatrix} \mathbf{p}_1$$

$$\mathbf{p}_2 = \tilde{\mathbf{M}}_2 \begin{pmatrix} \mathbf{I} \\ -\mathbf{N} \end{pmatrix} \mathbf{p}_1 = \mathbf{H}\,\mathbf{p}_1$$

Note: Let (X’,Y’,Z’,1) be the coordinates of point **P** in the coordinate frame of the **second** camera. **p**$_2$ is the coordinate of its image again in the system of the *second camera*. Matrix **M**$_2$ is the extrinsic (3x4) matrix for the second camera and 4 x4 matrix **M**$_{1,2}$ contains suitable extrinsic parameters to align the coordinate systems of the first camera to that of the second camera.

H is a 3 x 3 matrix, as **M~**$_2$ is a 3 x 4 matrix, and (**I**;-**N**) is a 4 x 3 matrix.

Planar Homography (with homogeneous coordinates)

- The previous relationship was between the coordinates of the images of **P**: $\mathbf{p}_1$ in the first camera's coordinate system, and $\mathbf{p}_2$ in the second camera's coordinate system.

- What about a similar relationship in the respective image coordinate systems?

$$
\begin{aligned}
&\mathbf{p}_2 = \mathbf{H}\mathbf{p}_1 \\
&\therefore \mathbf{M}^{-1}_{2,\mathrm{int}}\mathbf{p}_{2,\mathrm{im}} = \mathbf{H}\mathbf{M}^{-1}_{1,\mathrm{int}}\mathbf{p}_{1,\mathrm{im}} \\
&\therefore \mathbf{p}_{2,\mathrm{im}} = \mathbf{M}_{2,\mathrm{int}}\mathbf{H}\mathbf{M}^{-1}_{1,\mathrm{int}}\mathbf{p}_{1,\mathrm{im}} = \hat{\mathbf{H}}\mathbf{p}_{1,\mathrm{im}}
\end{aligned}
$$

$\mathbf{M}_{1,\mathrm{int}}$ and $\mathbf{M}_{2,\mathrm{int}}$ are intrinsic (3x3) camera matrices giving the relation between the coordinates of a point in camera coordinate system and image coordinate system.

This is also a 3 x 3 matrix which relates the image coordinates in the two images (in their respective image coordinate systems)

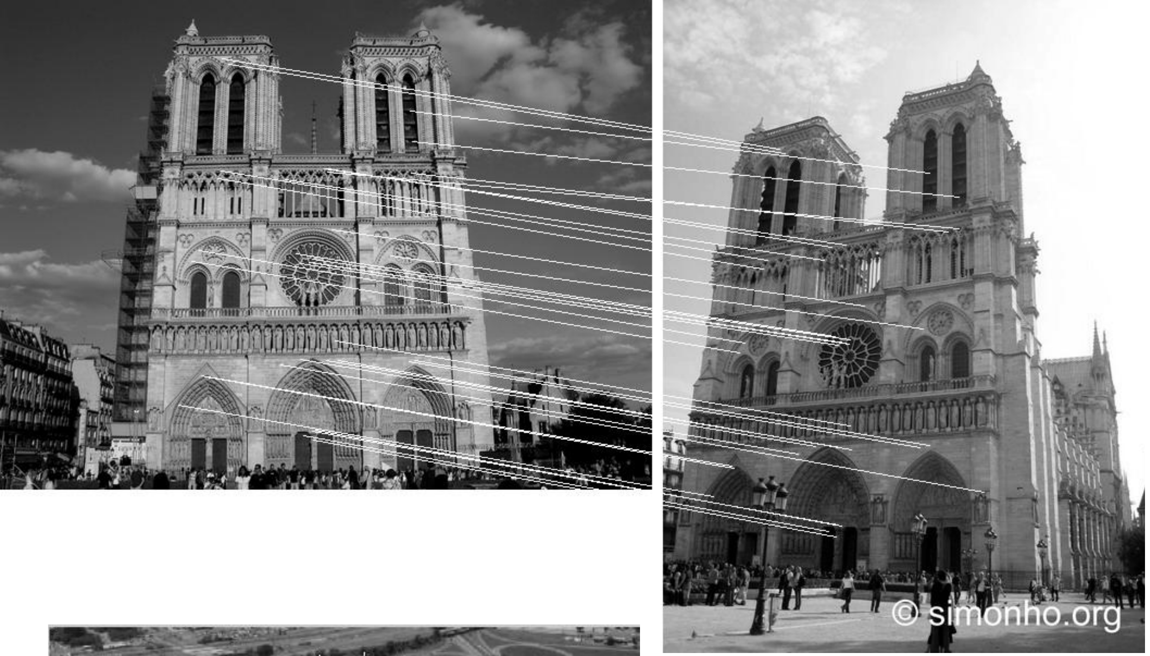
Planar Homography

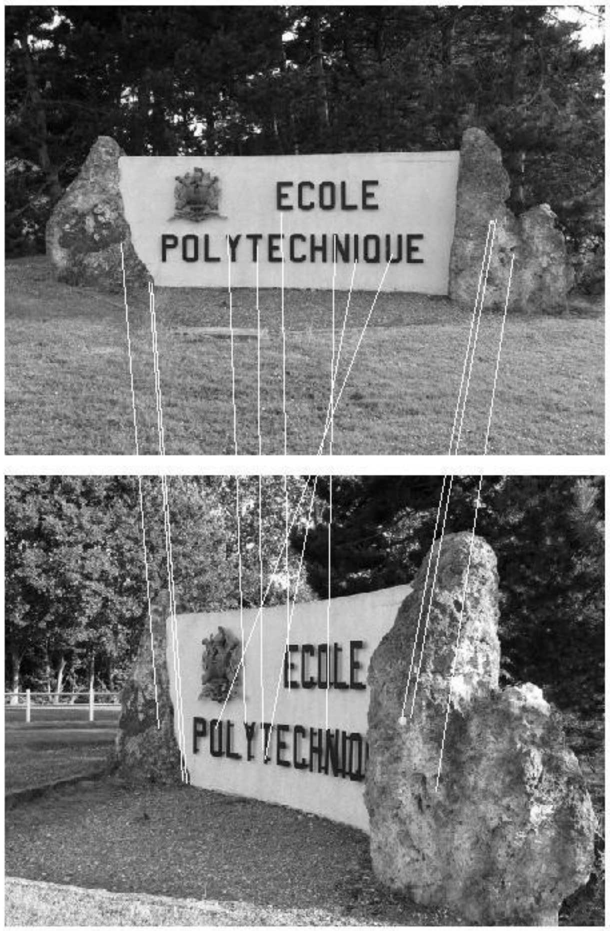
- Given two images of a coplanar scene taken from two different (uncalibrated) cameras, how will you determine the planar homography matrix **H** (rather **H^**)?
- How many point correspondences will you require? The answer is 4 so that we have at least 8 equations for the 8 degrees of freedom.

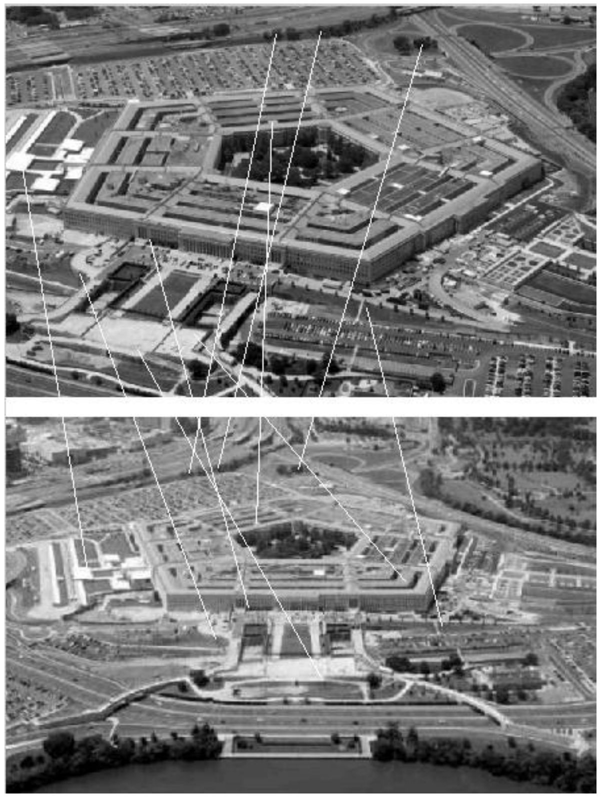
$$\mathbf{p}_{2,im} = \begin{pmatrix} u_2 \\ v_2 \\ w_2 \end{pmatrix} = \hat{\mathbf{H}}\mathbf{p}_{1,im} = \begin{pmatrix} \hat{H}_{11} & \hat{H}_{12} & \hat{H}_{13} \\ \hat{H}_{21} & \hat{H}_{22} & \hat{H}_{23} \\ \hat{H}_{31} & \hat{H}_{32} & \hat{H}_{33} \end{pmatrix} \begin{pmatrix} u_1 \\ v_1 \\ w_1 \end{pmatrix}$$

$$x_{2,im} = \frac{\hat{H}_{11}u_1 + \hat{H}_{12}v_1 + \hat{H}_{13}w_1}{\hat{H}_{31}u_1 + \hat{H}_{32}v_1 + \hat{H}_{33}w_1} = \frac{\hat{H}_{11}x_1 + \hat{H}_{12}y_1 + \hat{H}_{13}}{\hat{H}_{31}x_1 + \hat{H}_{32}y_1 + \hat{H}_{33}}, x_{1,im} = \frac{u_1}{w_1}, y_{1,im} = \frac{v_1}{w_1}$$

$$y_{2,im} = \frac{\hat{H}_{21}u_1 + \hat{H}_{22}v_1 + \hat{H}_{23}w_1}{\hat{H}_{31}u_1 + \hat{H}_{32}v_1 + \hat{H}_{33}w_1} = \frac{\hat{H}_{21}x_1 + \hat{H}_{22}y_1 + \hat{H}_{23}}{\hat{H}_{31}x_1 + \hat{H}_{32}y_1 + \hat{H}_{33}}$$







http://www.cmap.polytechnique.fr/~yu/research/ASIFT/demo.html

$$\boxed{\begin{aligned} &x_{2i}x_{1i}\hat{H}_{31} + x_{2i}y_{1i}\hat{H}_{32} + x_{2i}\hat{H}_{33} - x_{1i}\hat{H}_{11} - y_{1i}\hat{H}_{12} - \hat{H}_{13} = 0 \\ &y_{2i}x_{1i}\hat{H}_{31} + y_{2i}y_{1i}\hat{H}_{32} + y_{2i}\hat{H}_{33} - x_{1i}\hat{H}_{21} - y_{1i}\hat{H}_{22} - \hat{H}_{23} = 0 \end{aligned}}$$

$$\begin{pmatrix} -x_{1i} & -y_{1i} & -1 & 0 & 0 & 0 & x_{2i}x_{1i} & x_{2i}y_{1i} & x_{2i} \\ 0 & 0 & 0 & -x_{1i} & -y_{1i} & -1 & y_{2i}x_{1i} & y_{2i}y_{1i} & y_{2i} \\ & & & & & & & & \\ & & & & & & & & \end{pmatrix} \begin{pmatrix} \hat{H}_{11} \\ \hat{H}_{12} \\ \hat{H}_{13} \\ \hat{H}_{21} \\ \hat{H}_{22} \\ \hat{H}_{23} \\ \hat{H}_{31} \\ \hat{H}_{32} \\ \hat{H}_{33} \end{pmatrix} = \mathbf{0}$$

$$\mathbf{Ah} = \mathbf{0}, \mathbf{A} \text{ has size } 2N \times 9, \mathbf{h} \text{ has size } 9 \times 1$$

There will be *N* such pairs of equations (i.e. totally 2*N* equations), given *N* pairs of ***corresponding*** points in the two images

The equation **Ah** = **0** will be solved by computing the SVD of **A**, i.e. **A** = **USV**^{T}. The vector **h** will be given by the singular vector in **V**, corresponding to the null singular value (in the ideal case) or the least singular value. This is equivalent to the eigenvector of **A**^{T}**A** with the least eigenvalue. In the ideal case, **A** has rank 8 and hence the exact solution exists. In the non-ideal case, there is no **h** such that **Ah** = **0**, so we find an approximate solution, i.e. a solution for which **Ah** is small in magnitude.

Planar Homography

- Although there are 9 entries in the matrix, there are only 8 degrees of freedom. You can see this if you divide the numerator and denominator of the following equations by H^$_{33}$.

$$x_2 = \frac{\hat{H}_{11}x_1 + \hat{H}_{12}y_1 + \hat{H}_{13}}{\hat{H}_{31}x_1 + \hat{H}_{32}y_1 + \hat{H}_{33}} = \frac{\tilde{H}_{11}x_1 + \tilde{H}_{12}y_1 + \tilde{H}_{13}}{\tilde{H}_{31}x_1 + \tilde{H}_{32}y_1 + 1}$$

$$y_2 = \frac{\hat{H}_{21}x_1 + \hat{H}_{22}y_1 + \hat{H}_{23}}{\hat{H}_{31}x_1 + \hat{H}_{32}y_1 + \hat{H}_{33}} = \frac{\tilde{H}_{21}x_1 + \tilde{H}_{22}y_1 + \tilde{H}_{23}}{\tilde{H}_{31}x_1 + \tilde{H}_{32}y_1 + 1}$$

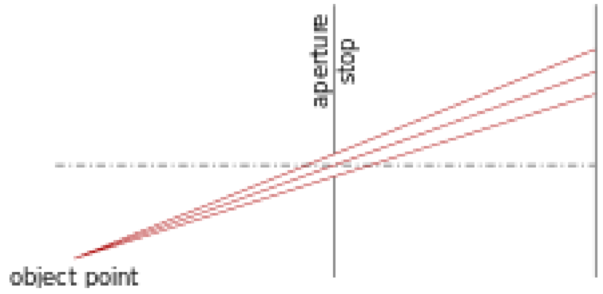
- The number of point correspondences required is at least 4.

Planar homography

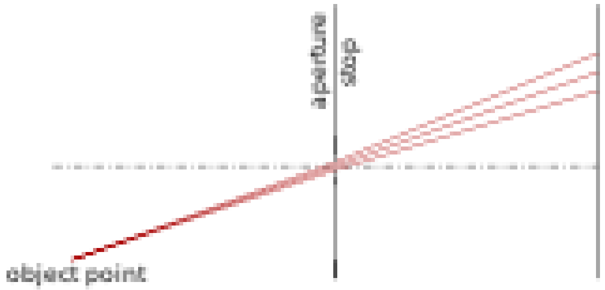
- It is a motion model that is more general than 2D affine transformations!
- Note: in the derivation, make sure you understand that the **planarity** of the object being imaged, is **critical**. Otherwise you need a motion model more complicated than a homography!
- Again note: you do not need calibrated cameras (i.e. the corresponding points are in terms of image coordinates, i.e. in pixels), and you do not need to know the intrinsic or extrinsic camera parameters in order to determine the homography.
- The homography transformation has 8 degrees of freedom and is more general than the affine transformation which has only 6 degrees of freedom. Observe the homography matrix on the previous slides – in an affine transformation, H_{31} and H_{32} would be 0.

Adding a Lens

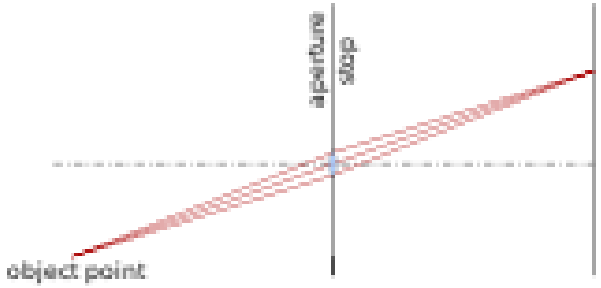
Pinhole Camera to Camera with Lens



With a large pinhole, the image spot is large, resulting in a blurry image.

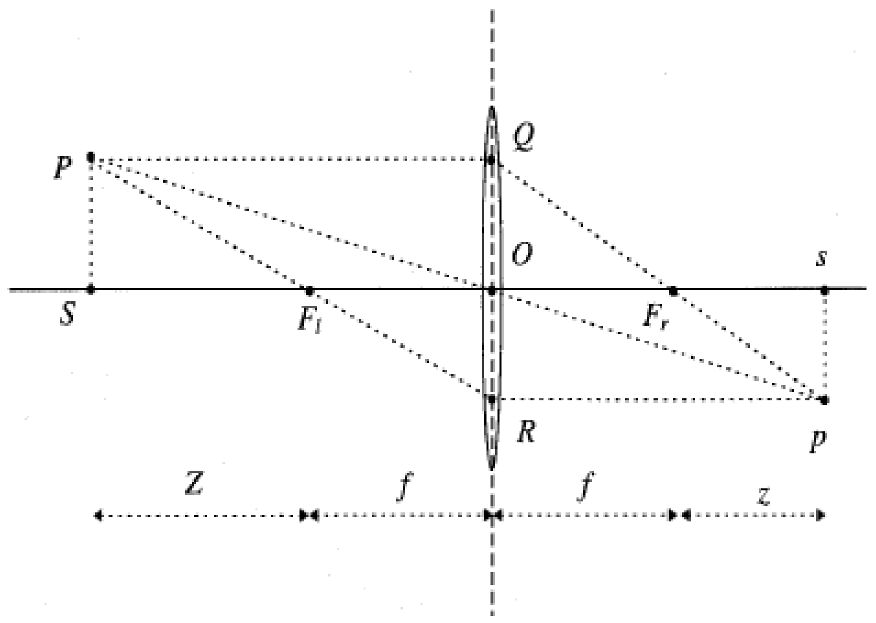


With a small pinhole, light is reduced resulting in a sharp but noisy image



With a simple lens, much more light can be brought into sharp focus.

http://en.wikipedia.org/wiki/Camera_lens

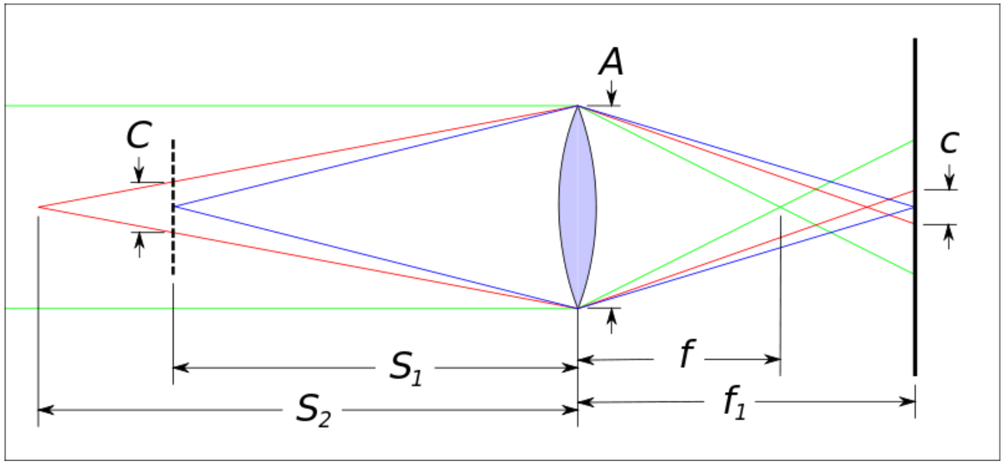


Thin convergent lens:

Optical axis Sos passing through the optical center O of the lens and normal to the plane of the lens

All rays of light from object point (say) P parallel to the optical axis pass through the principal focus Fr after refraction. The ray of light from P through O passes without refraction. But all rays of light emanating from P passing through the lens are focused onto a single point p – the image of P (if point P was in focus).

Camera Lens: Focal Length, Aperture

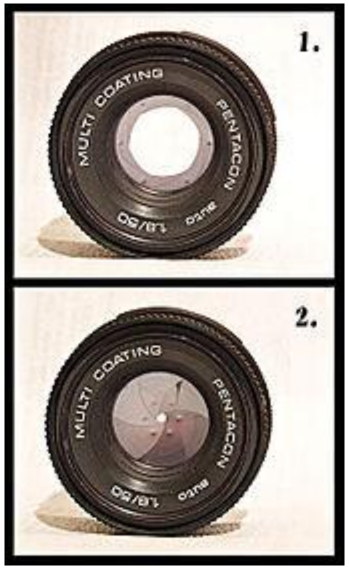


http://en.wikipedia.org/wiki/Circle_of_confusion

A lens focusses light onto a sensor (or photo-film). The distance between the center of the lens and the principal focus is called **focal length**. This distance is changed as the shape of the lens changes. Defocussed points map onto a **circle of confusion** on the sensor.

Focal length affects magnification. Large focal lengths yield smaller fields of view but more magnification. Small focal lengths yield large fields of view but less magnification.

Camera: Aperture

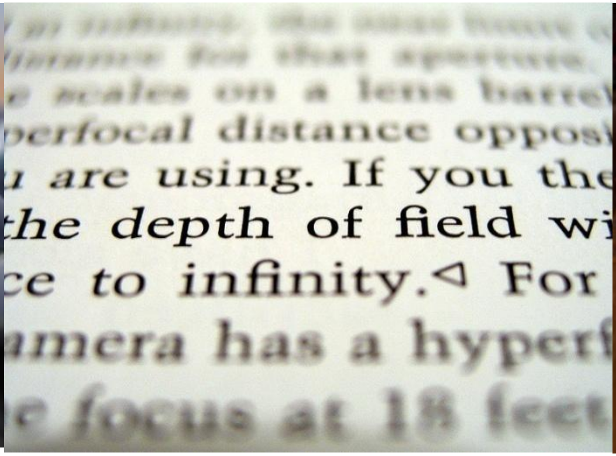


A wider aperture (above) allows for a larger shutter speed than a smaller aperture (below) for the same amount of exposure (or the same amount of light entering the camera).

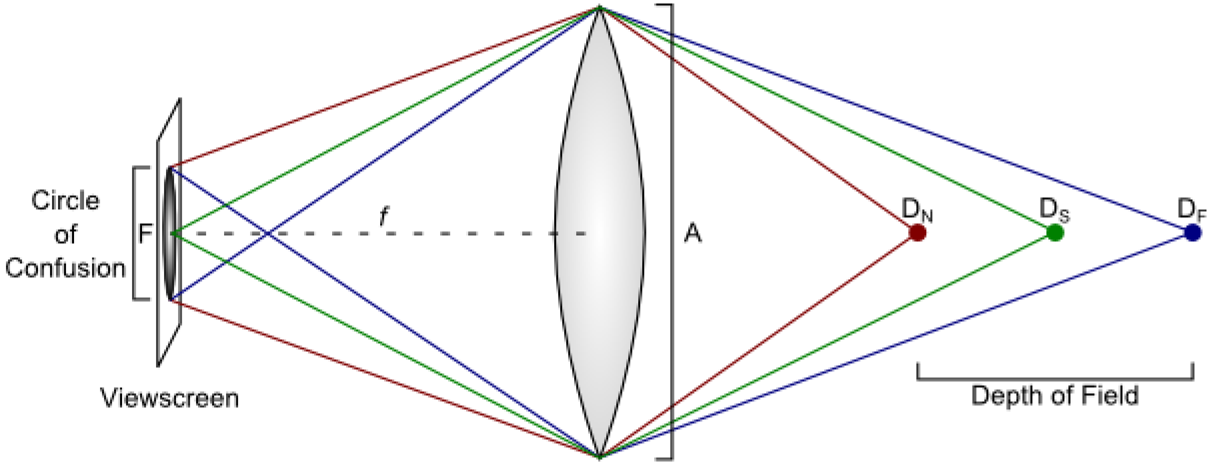
Lower shutter speed = susceptibility to motion blur

Depth of field

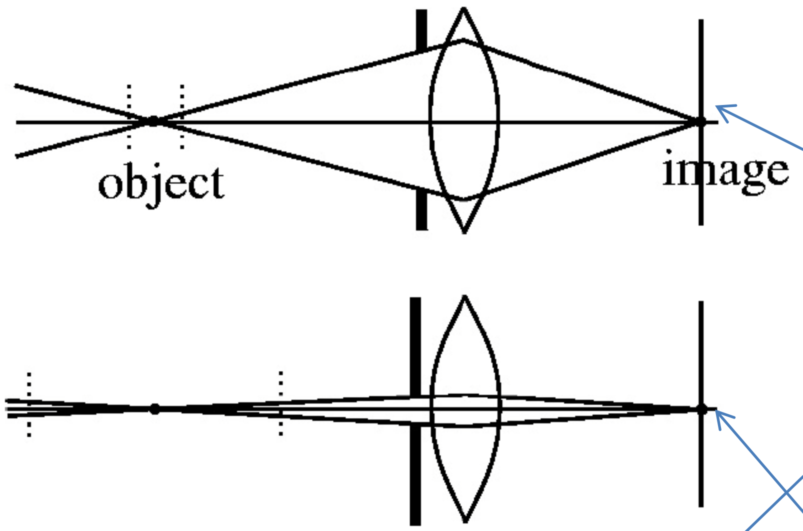
- The images of some object points can appear blurry. The amount of blur depends on the distance between the object point and the camera plane (i.e. depth of the point).
- The range of depths for which the level of blur is acceptable to the human eye (or which appear acceptably sharp) is called as **depth of field**.



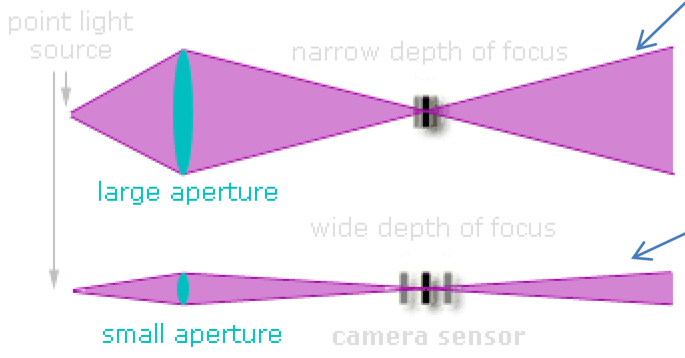
[illegible]
distance for that aperture.
e scales on a lens barrel
perfocal distance opposi
u are using. If you the
the depth of field wi
ce to infinity.◁ For
amera has a hyperf
e focus at 18 feet



Aperture and depth of field



Larger aperture = shallower DOF, as radii of circle of confusion increase rapidly

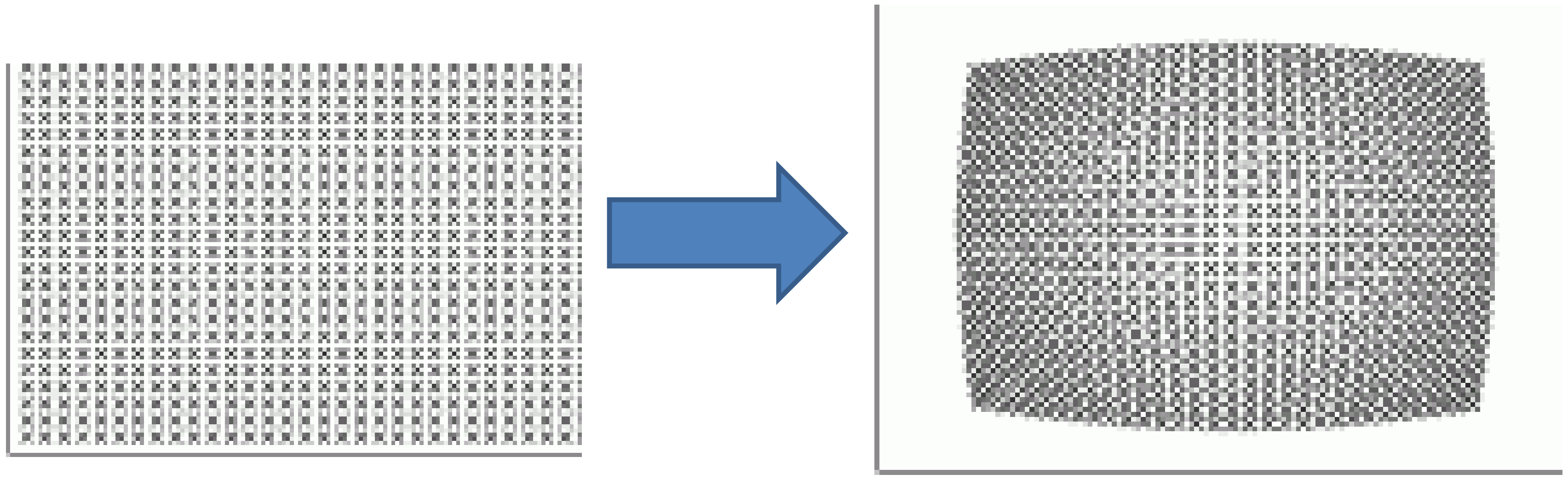


Smaller aperture = more extensive DOF, as radii of circle of confusion increase more slowly

http://www.cambridgeincolour.com/tutorials/depth-of-field.htm

Downside of using a lens ☹

- It causes radial distortion – straight lines in 3D can get mapped onto curved lines in 2D, and this effect is much more jarring near the image borders.



Downside of using a lens ☹

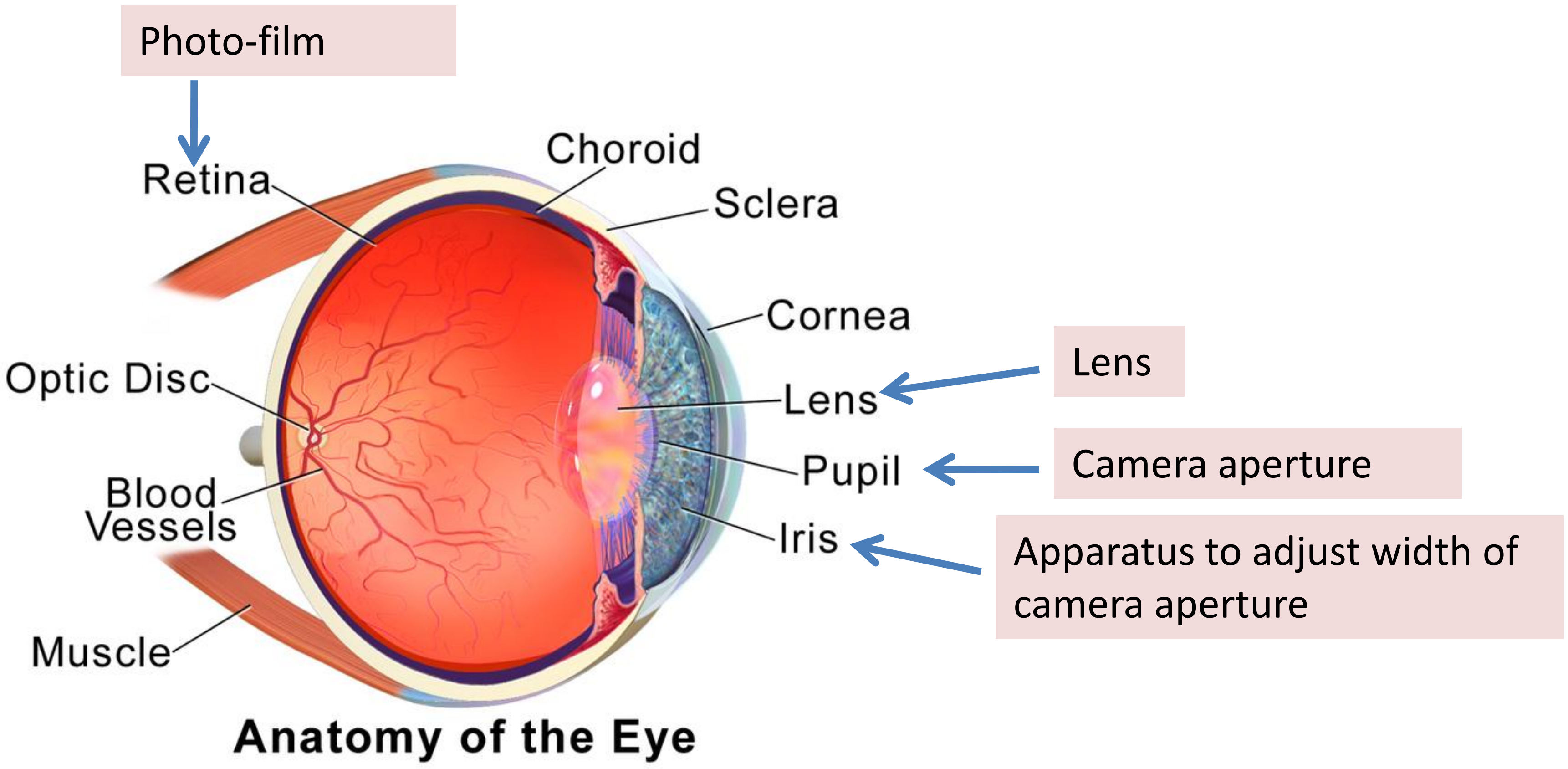
- The relationship between the distorted and undistorted coordinates is approximated by:

$$x_d - o_x = (x_u - o_x)(1 + k_1 r^2 + k_2 r^4)$$
$$y_d - o_y = (y_u - o_y)(1 + k_1 r^2 + k_2 r^4)$$
$$r^2 = (x_u - o_x)^2 + (y_u - o_y)^2$$

(x_u,y_u) = undistorted coordinates
(x_d,y_d) = distorted coordinates

- k_1 and k_2 are (extra!) intrinsic parameters of the camera. The associated calibration procedure is now more complicated.

The Human Eye as a Camera



http://en.wikipedia.org/wiki/File:Blausen_0388_EyeAnatomy_01.png

Summary

- Affine Transformations in 2D and 3D
- Pinhole camera, perspective projections
- Vanishing points
- Camera Calibration
- Cross-ratios
- Planar homography
- Adding in a lens