MIT CSAIL

MIT COMPUTER VISION

6.819 / 6.869: Advances in Computer Vision

Antonio Torralba and Bill Freeman

Lecture 11

Geometry, Camera Calibration, and Stereo.

2d from 3d;
3d from multiple 2d measurements

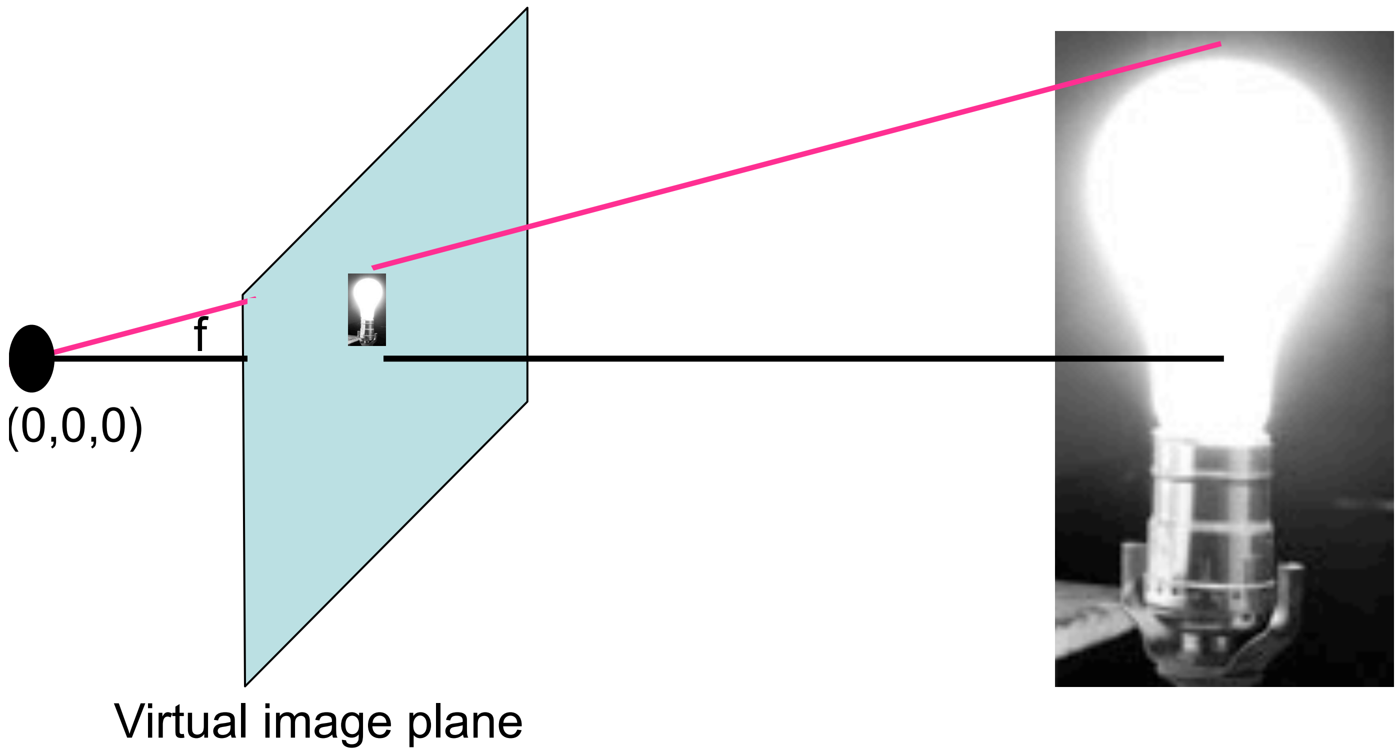
?

2d

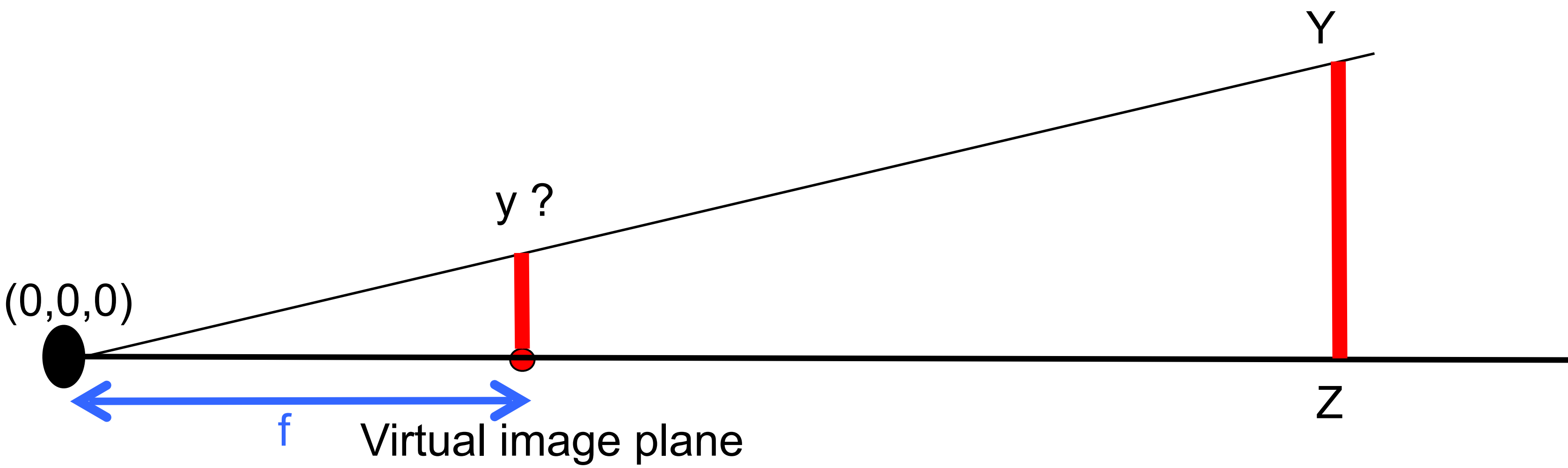
3d

?

Perspective projection



Perspective projection



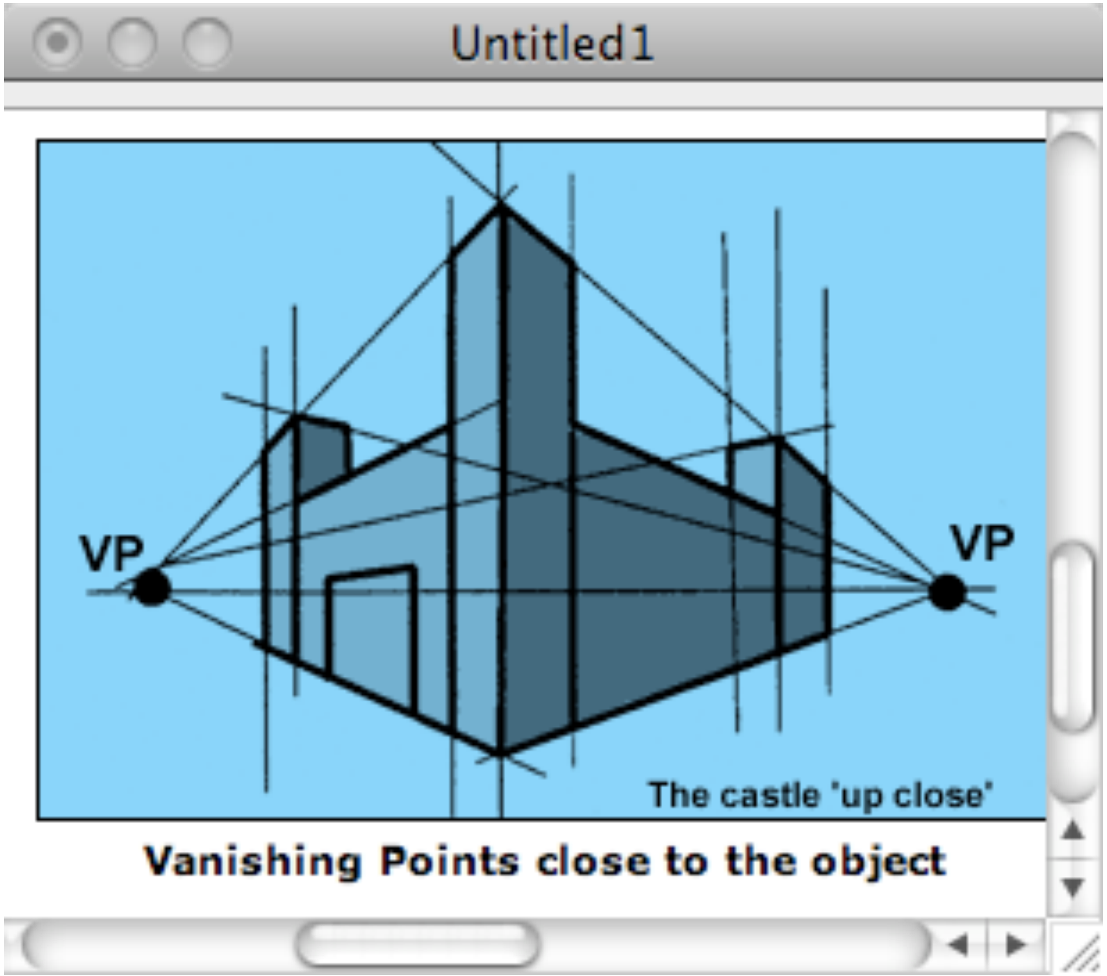
Similar triangles: y / f = Y / Z

y = f Y/Z

Perspective projection:

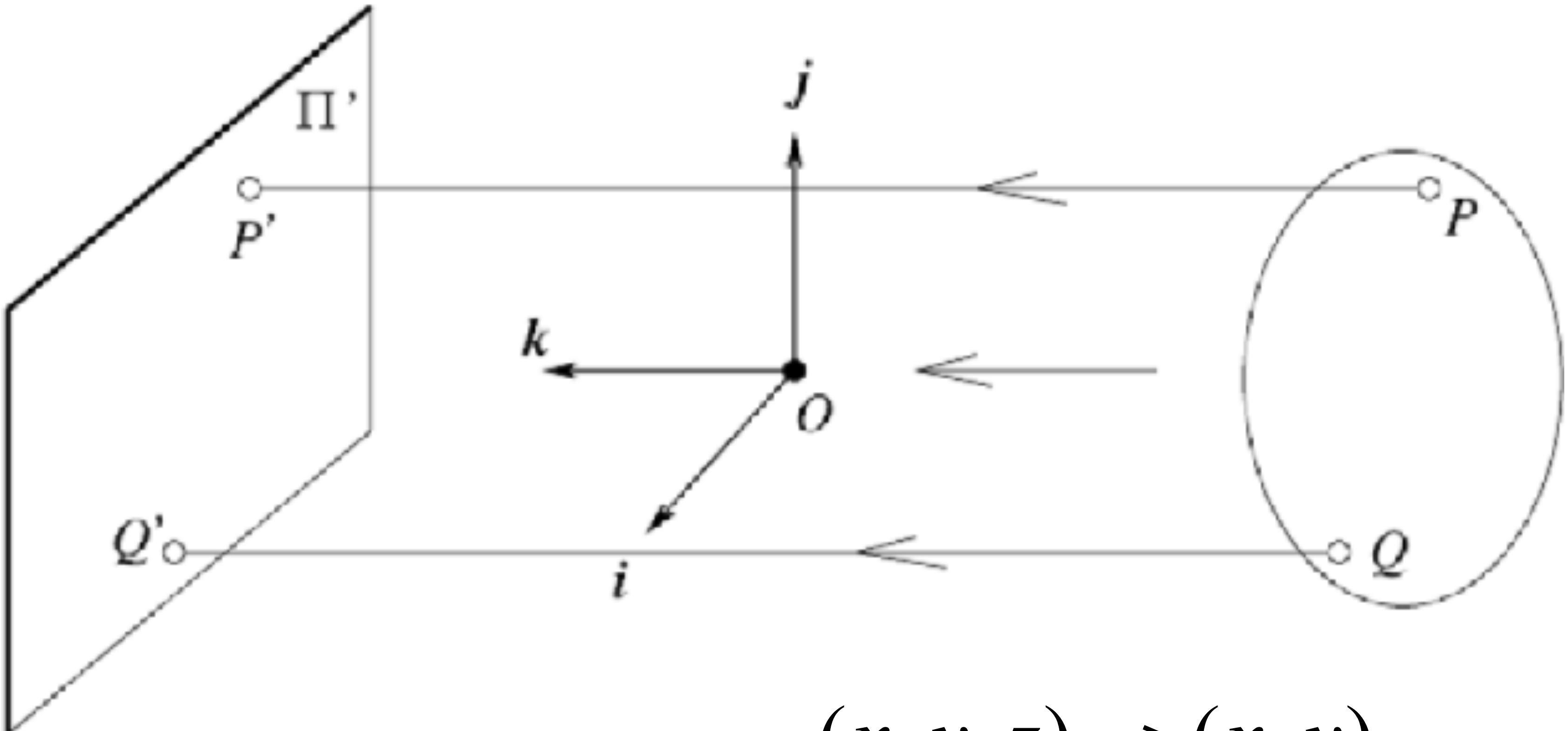
$$(X,Y,Z) \Rightarrow \left(f\frac{X}{Z}, f\frac{Y}{Z} \right)$$

Vanishing points



http://www.ider.herts.ac.uk/school/courseware/graphics/two_point_perspective.html

Other projection models: Orthographic projection



$$(x, y, z) \rightarrow (x, y)$$

Three camera projections

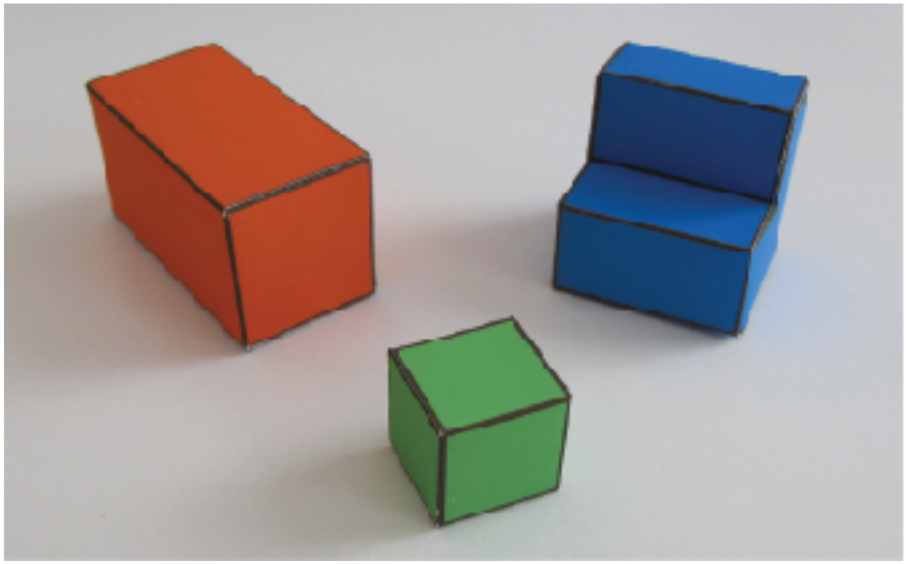
3-d point **2-d image position**

(1) Perspective: $$(x, y, z) \rightarrow \left(\frac{fx}{z}, \frac{fy}{z}\right)$$

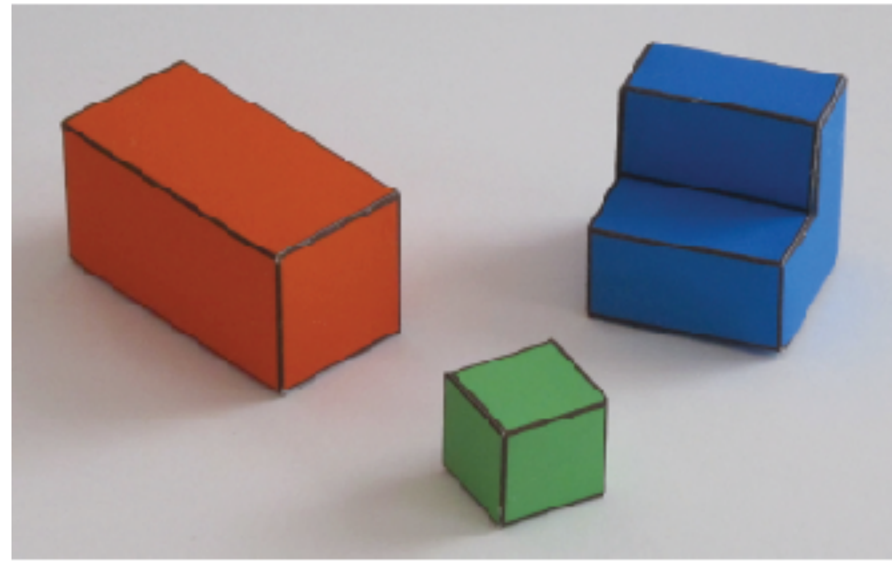
(2) Weak perspective: $$(x, y, z) \rightarrow \left(\frac{fx}{z_0}, \frac{fy}{z_0}\right)$$

(3) Orthographic: $$(x, y, z) \rightarrow (x, y)$$

Three camera projections



Perspective projection



Parallel (orthographic) projection

Weak perspective?

Homogeneous coordinates

Is the perspective projection a linear transformation?

- **no—division by z is nonlinear**

Trick: add one more coordinate:

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

homogeneous image coordinates

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

homogeneous world coordinates

Converting *from* homogeneous coordinates

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

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

Slide by Steve Seitz

Perspective Projection

- Projection is a matrix multiply using homogeneous coordinates:

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

This is known as perspective projection

- **The matrix is the projection matrix**

Slide by Steve Seitz

Perspective Projection

How does scaling the projection matrix change the transformation?

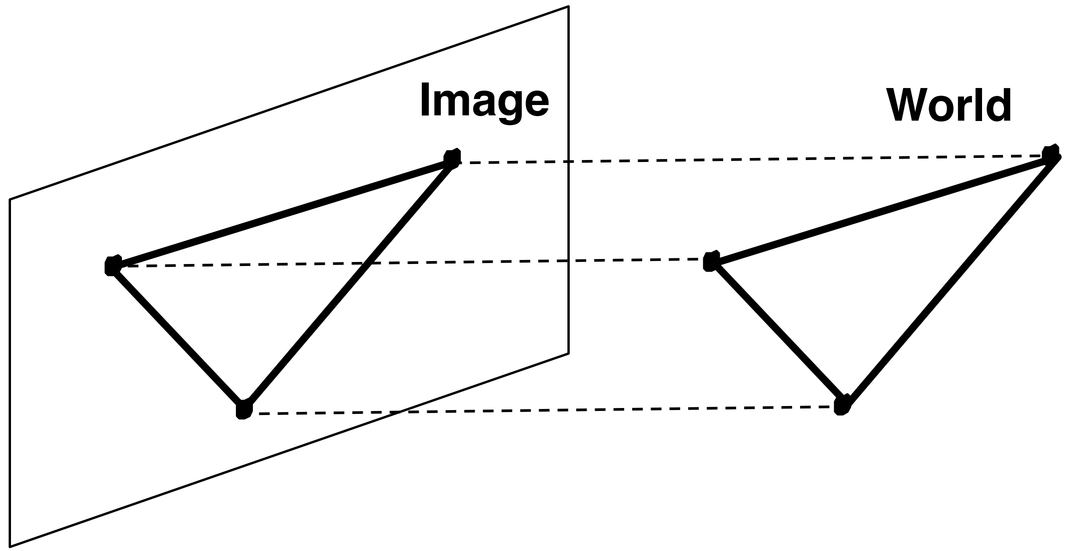
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z/f \end{bmatrix} \Rightarrow \left(f\frac{x}{z}, f\frac{y}{z} \right)$$

$$\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} fx \\ fy \\ z \end{bmatrix} \Rightarrow \left(f\frac{x}{z}, f\frac{y}{z} \right)$$

Slide by Steve Seitz

Orthographic Projection

Special case of perspective projection



- Also called “parallel projection”
- What’s the projection matrix?

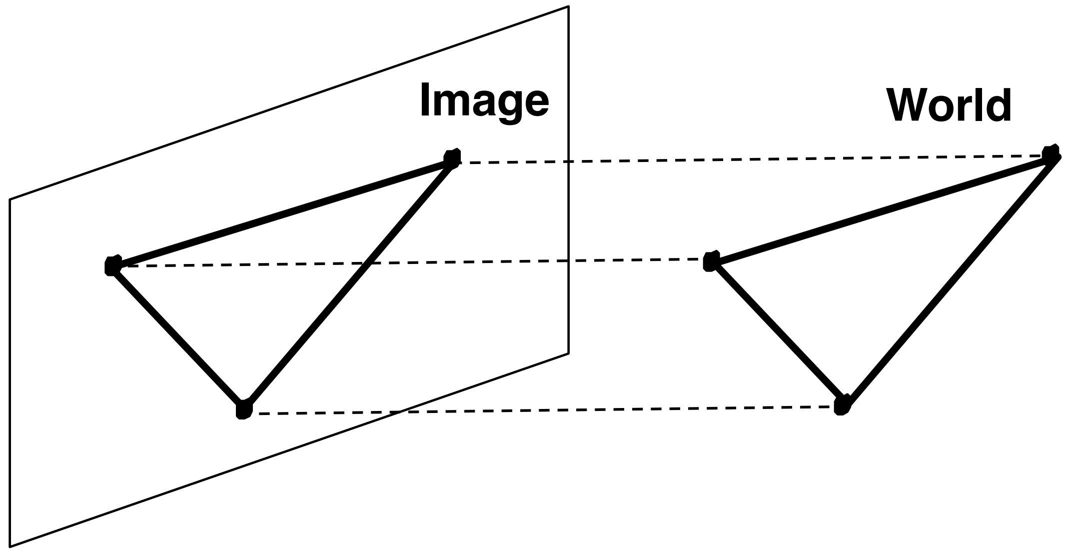
$$\boxed{\quad ? \quad}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow (x, y)$$

Slide by Steve Seitz

Orthographic Projection

Special case of perspective projection

- Distance from the COP to the PP is infinite

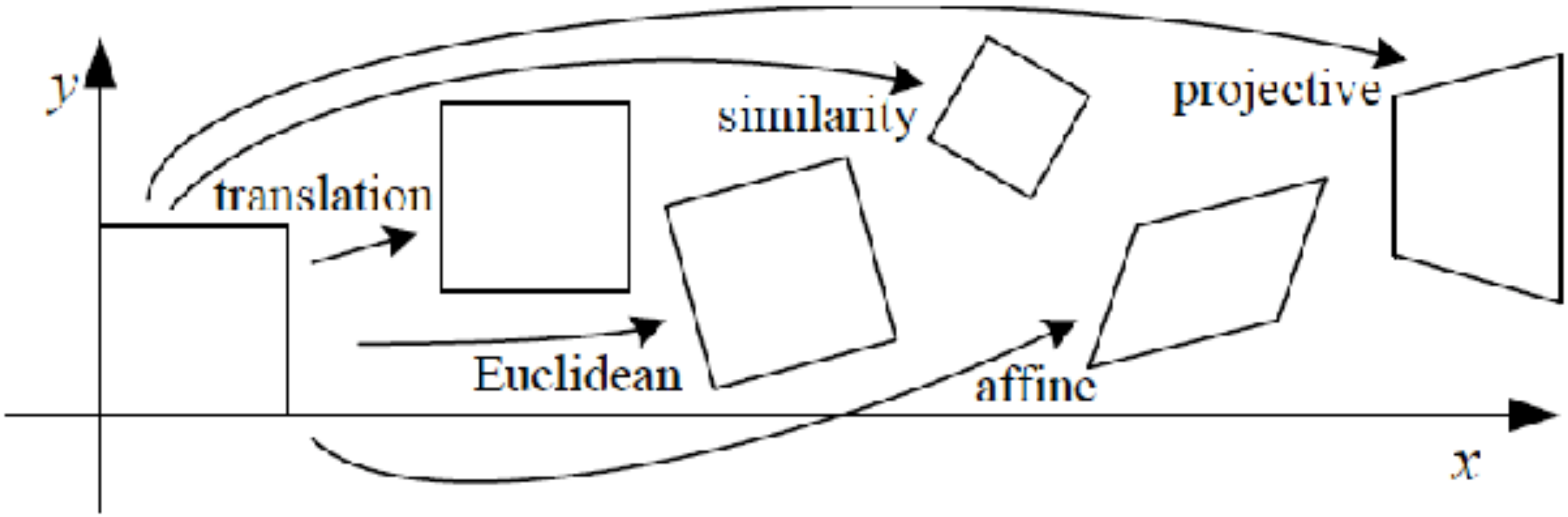


- Also called “parallel projection”
- What’s the projection matrix?

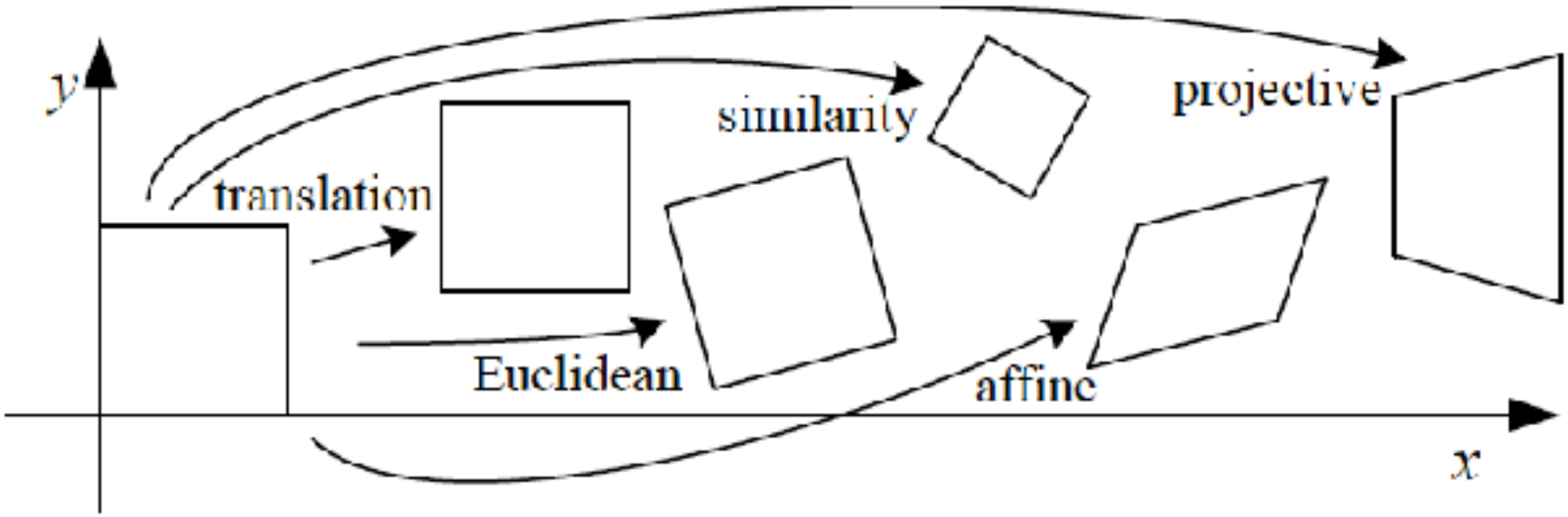
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow (x, y)$$

Slide by Steve Seitz

2D Transformations



2D Transformations

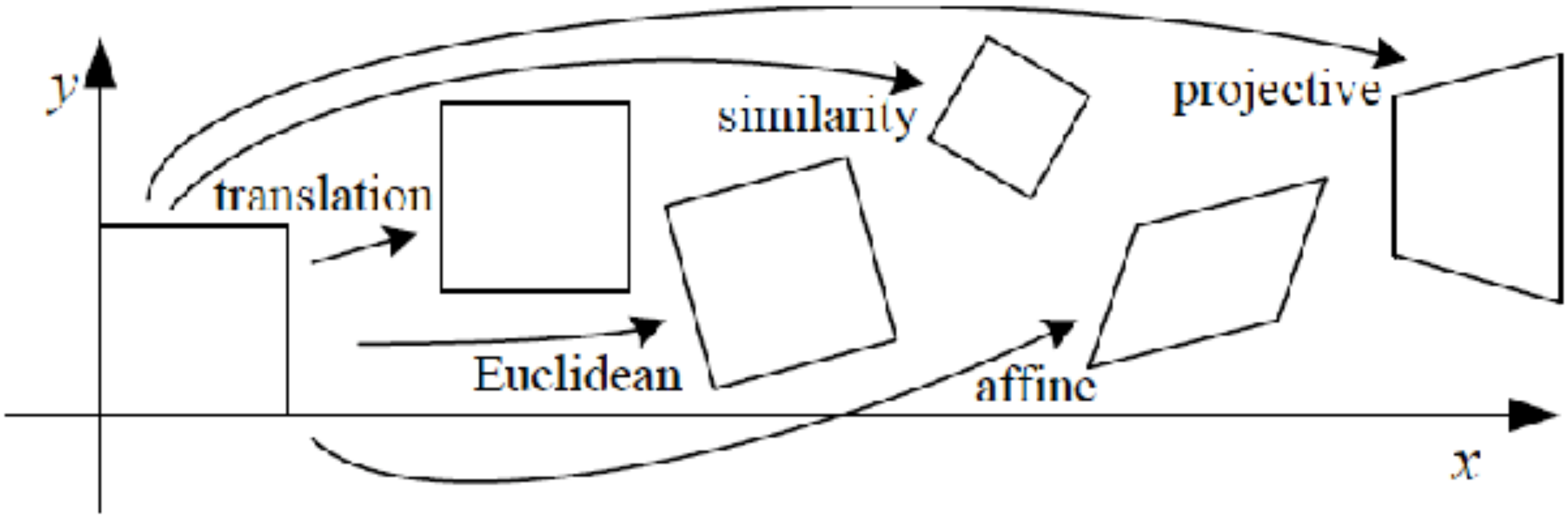


Example: translation

$$x' = x + t$$

$$\begin{bmatrix} \\ \end{bmatrix} = \begin{bmatrix} \\ \end{bmatrix} + \begin{bmatrix} tx \\ ty \end{bmatrix}$$

2D Transformations

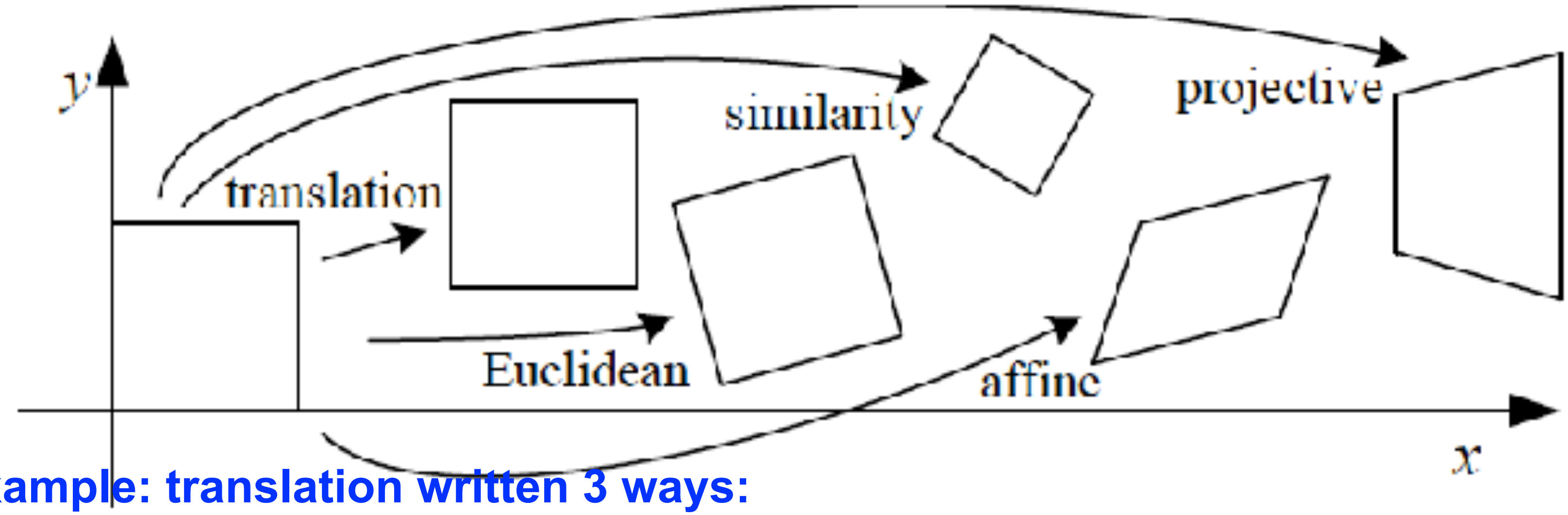


Example: translation

$$x' = x + t \qquad x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x}$$

$$\begin{bmatrix} \; \\ \; \end{bmatrix} = \begin{bmatrix} \; \\ \; \end{bmatrix} + \begin{bmatrix} tx \\ ty \end{bmatrix} \qquad \begin{bmatrix} \; \\ \; \end{bmatrix} = \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \end{bmatrix} \cdot \begin{bmatrix} \; \\ \; \\ 1 \end{bmatrix}$$

2D Transformations



Example: translation written 3 ways:

non-homog

$$x' = x + t$$

$$\begin{bmatrix} \ \\ \ \end{bmatrix} = \begin{bmatrix} \ \\ \ \end{bmatrix} + \begin{bmatrix} tx \\ ty \end{bmatrix}$$

homog in, non-h out,

$$x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x}$$

$$\begin{bmatrix} \ \\ \ \end{bmatrix} = \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \end{bmatrix} . \begin{bmatrix} \ \\ \ \\ 1 \end{bmatrix}$$

homog in, homog out

$$\bar{x}' = \begin{bmatrix} I & t \\ 0^T & 1 \end{bmatrix} \bar{x}$$

$$\begin{bmatrix} \ \\ \ \\ \ \end{bmatrix} = \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \ \\ \ \\ \ \end{bmatrix}$$

Now we can chain transformations

Translation and rotation, written in each set of coordinates

from

Non-homogeneous coordinates

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

to

Homogeneous coordinates

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

where

$$ {}^{B}_{A}C = \begin{pmatrix} \boxed{\begin{matrix} - & - & - \\ - & {}^{B}_{A}R & - \\ - & - & - \end{matrix}} & \boxed{\begin{matrix} | \\ {}^{B}_{A}\vec{t} \\ | \end{matrix}} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{pmatrix} $$

Camera calibration

Use the camera to tell you things about the world:

- Relationship between coordinates in the world and coordinates in the image: *geometric camera calibration, see* Szeliski, section 5.2, 5.3 for references
- (Relationship between intensities in the world and intensities in the image: *photometric image formation*, see Szeliski, sect. 2.2.)

Camera calibration

- Intrinsic parameters

Image coordinates relative to camera ←→ Pixel coordinates

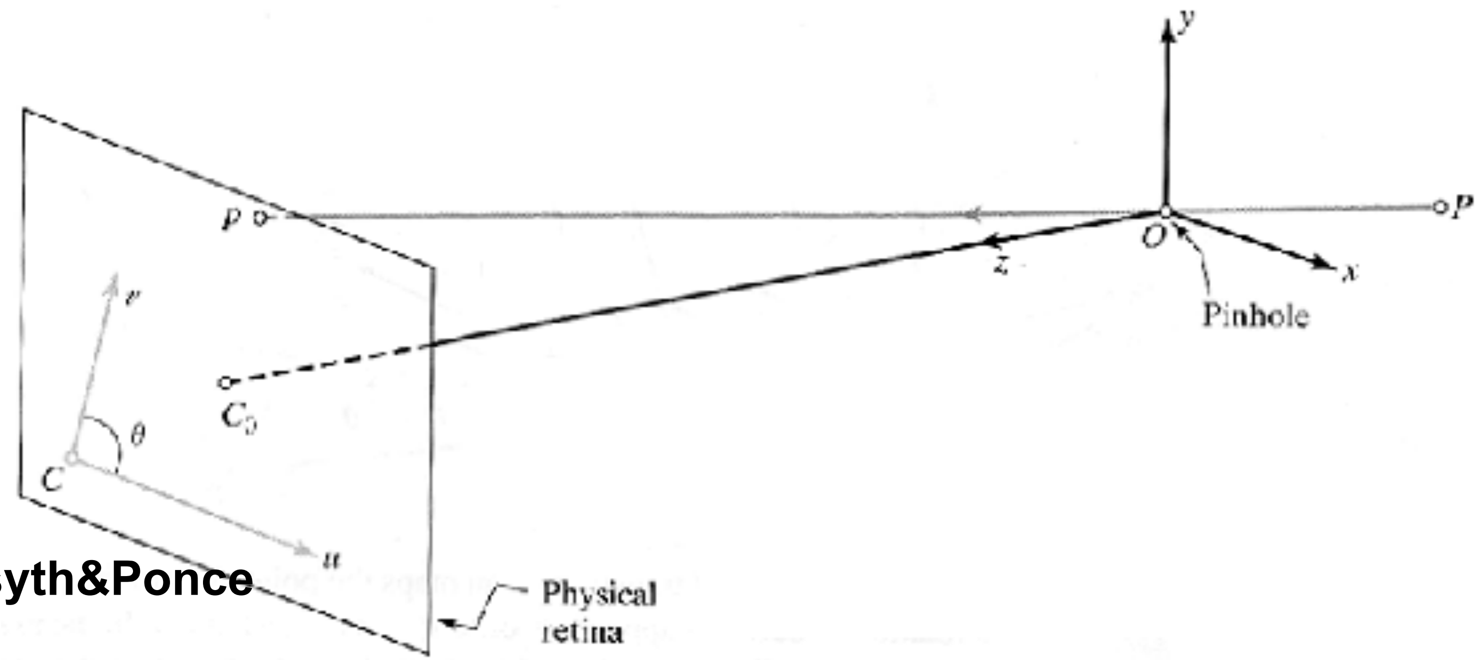
- Extrinsic parameters

Camera frame 1 ←→ Camera frame 2

Camera calibration

- Intrinsic parameters
- Extrinsic parameters

Intrinsic parameters: from idealized world coordinates to pixel values



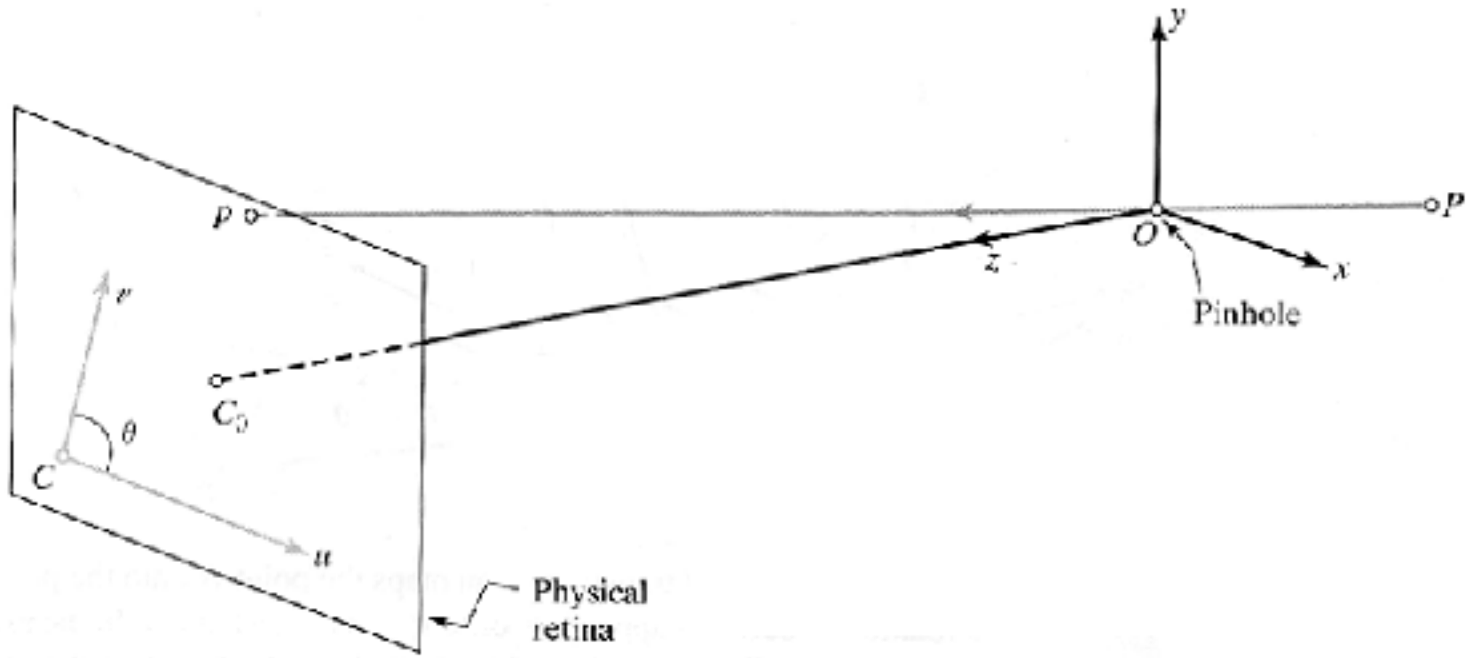
Forsyth&Ponce

Perspective projection

$$u = f\frac{x}{z}$$

$$v = f\frac{y}{z}$$

Intrinsic parameters

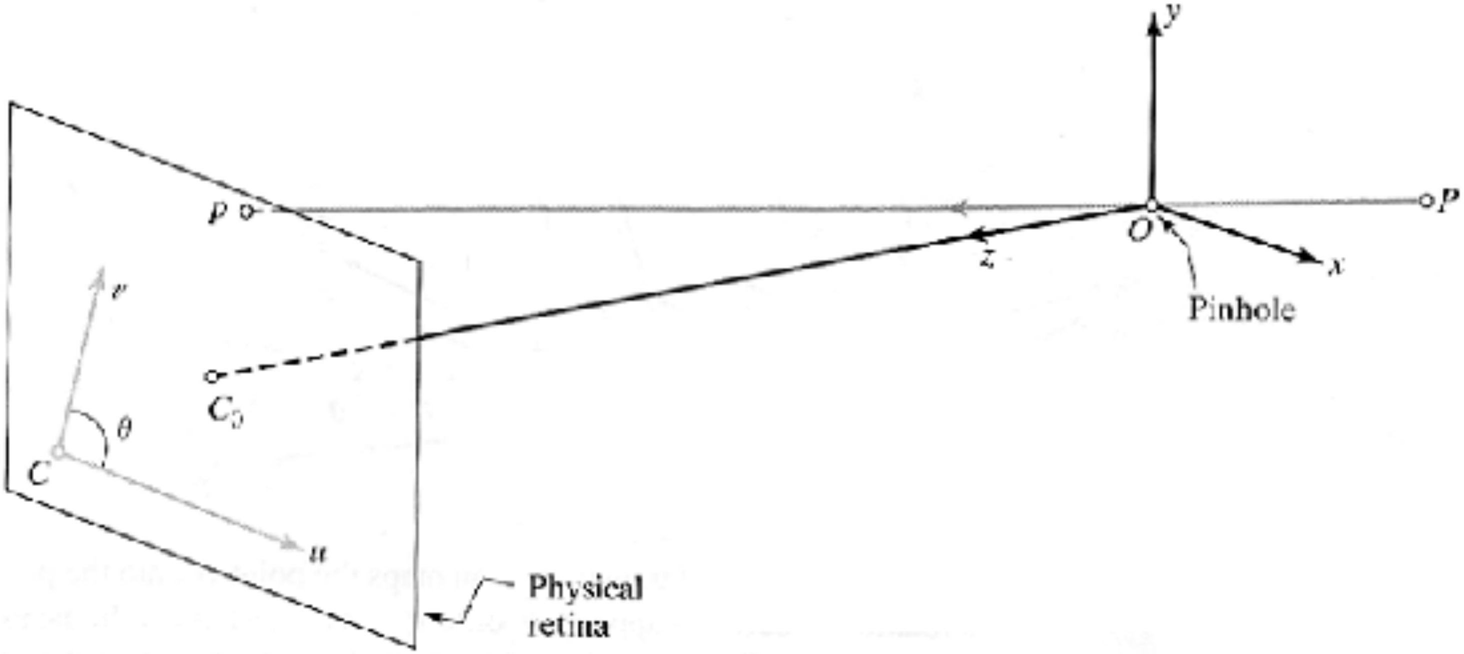


But "pixels" are in some arbitrary spatial units

$$u = \alpha \frac{x}{z}$$

$$v = \alpha \frac{y}{z}$$

Intrinsic parameters

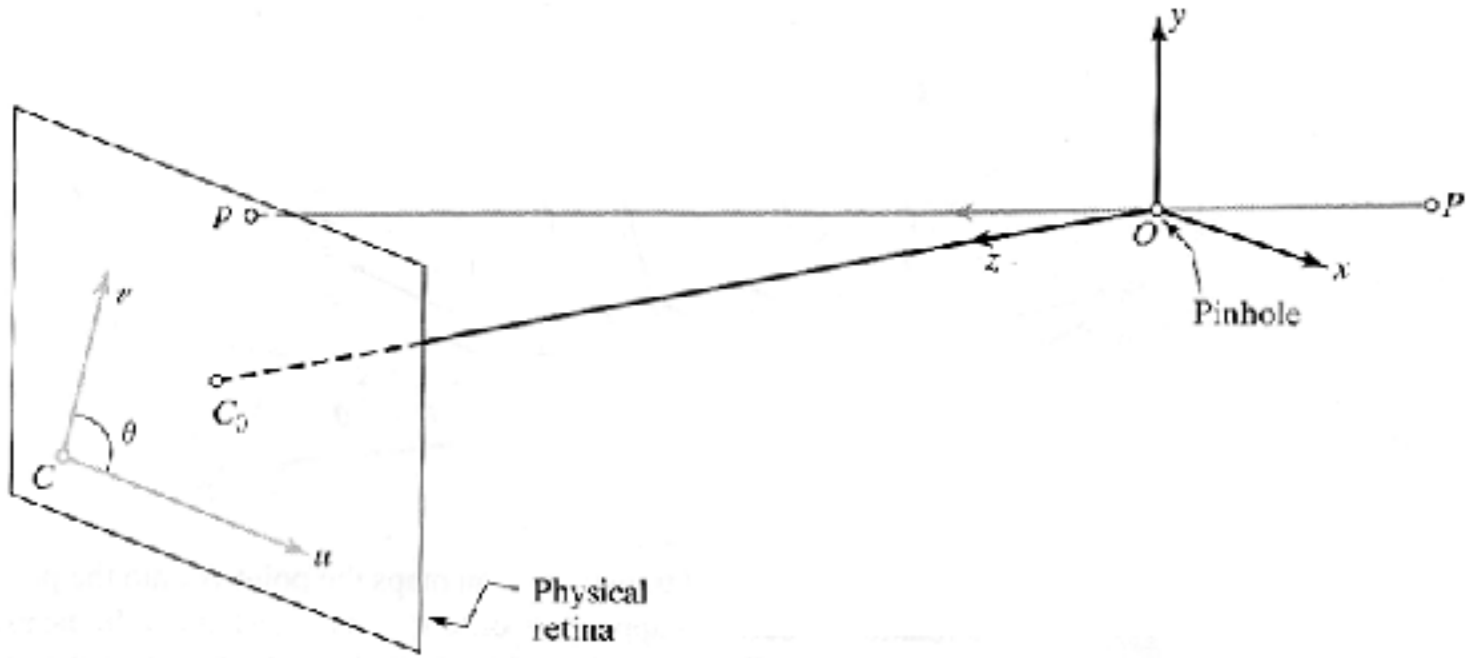


Maybe pixels are not square

$$u = \alpha \frac{x}{z}$$

$$v = \beta \frac{y}{z}$$

Intrinsic parameters

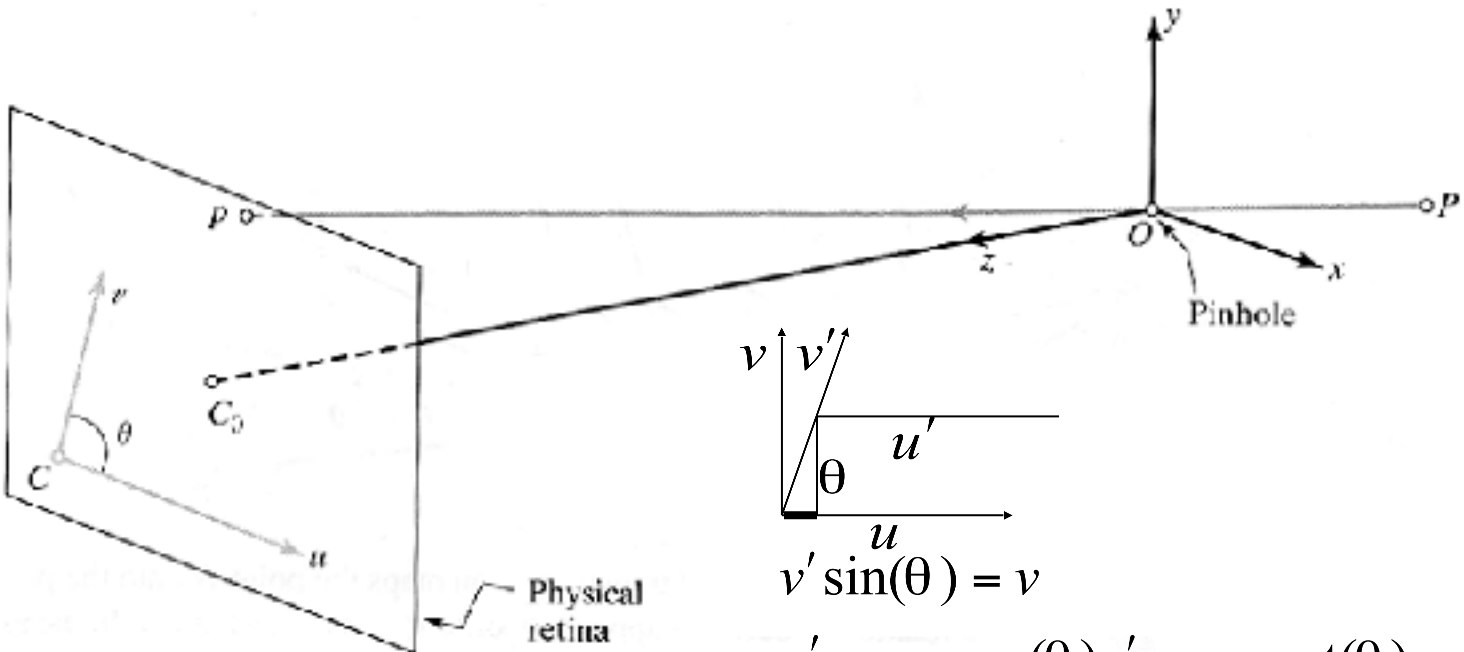


The origin of our camera pixel coordinates may be somewhere other than under the camera optical axis.

$$u = \alpha \frac{x}{z} + u_0$$

$$v = \beta \frac{y}{z} + v_0$$

Intrinsic parameters



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

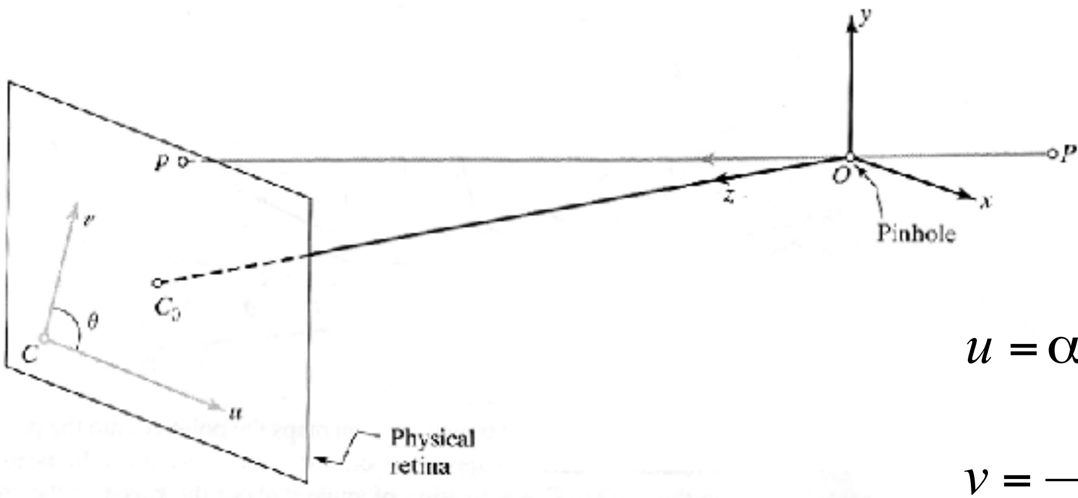
$$u' = u - \cos(\theta) v' = u - \cot(\theta) v$$

May be skew between camera pixel axes (but usually this angle is 90 deg).

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

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

Intrinsic parameters, homogeneous coordinates



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

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

Using homogenous coordinates, we can write this as:

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

or:

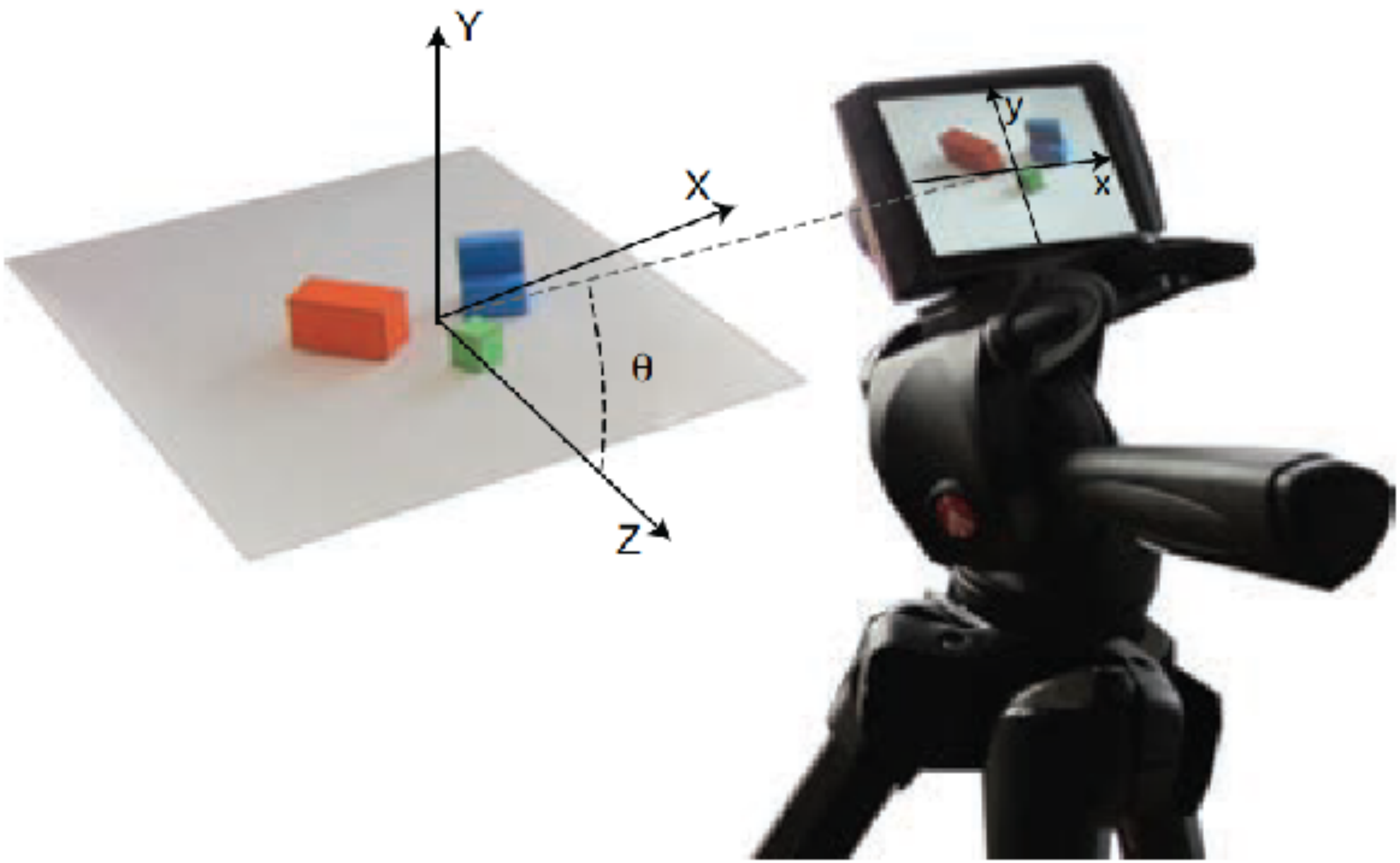
In pixels $\longrightarrow$ $\vec{p} = \mathrm{K} \; {}^{C}\vec{p}$

In camera-based coords

Camera calibration

- Intrinsic parameters
- Extrinsic parameters

World and camera coordinate systems



In the first lecture, we placed the world coordinates in the center of the scene.

Extrinsic parameters: translation and rotation of camera frame

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

Non-homogeneous coordinates

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

Homogeneous coordinates

pixels

Intrinsic

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

World coordinates

Camera coordinates

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

Extrinsic

Combining extrinsic and intrinsic calibration parameters, in homogeneous coordinates

$$\vec{p} = K \underbrace{\begin{pmatrix} {}^{C}_{W}R & {}^{C}_{W}\vec{t} \\ 0\,0\,0 & 1 \end{pmatrix}} \; {}^{W}\vec{p}$$

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

Forsyth&Ponce

Other ways to write the same equation

pixel coordinates

world coordinates

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

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

Conversion back from homogeneous coordinates leads to:

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

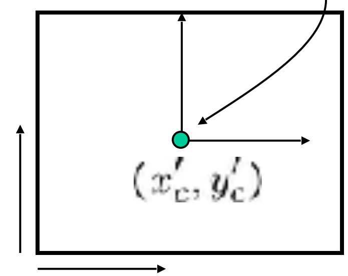
Summary camera parameters

A camera is described by several parameters

- Translation T of the optical center from the origin of world coords
- Rotation R of the image plane
- focal length f, principle point (x'_c, y'_c), pixel size (s_x, s_y)
- blue parameters are called "extrinsics," red are "intrinsics"

Projection equation

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



- The projection matrix models the cumulative effect of all parameters
- Useful to decompose into a series of operations

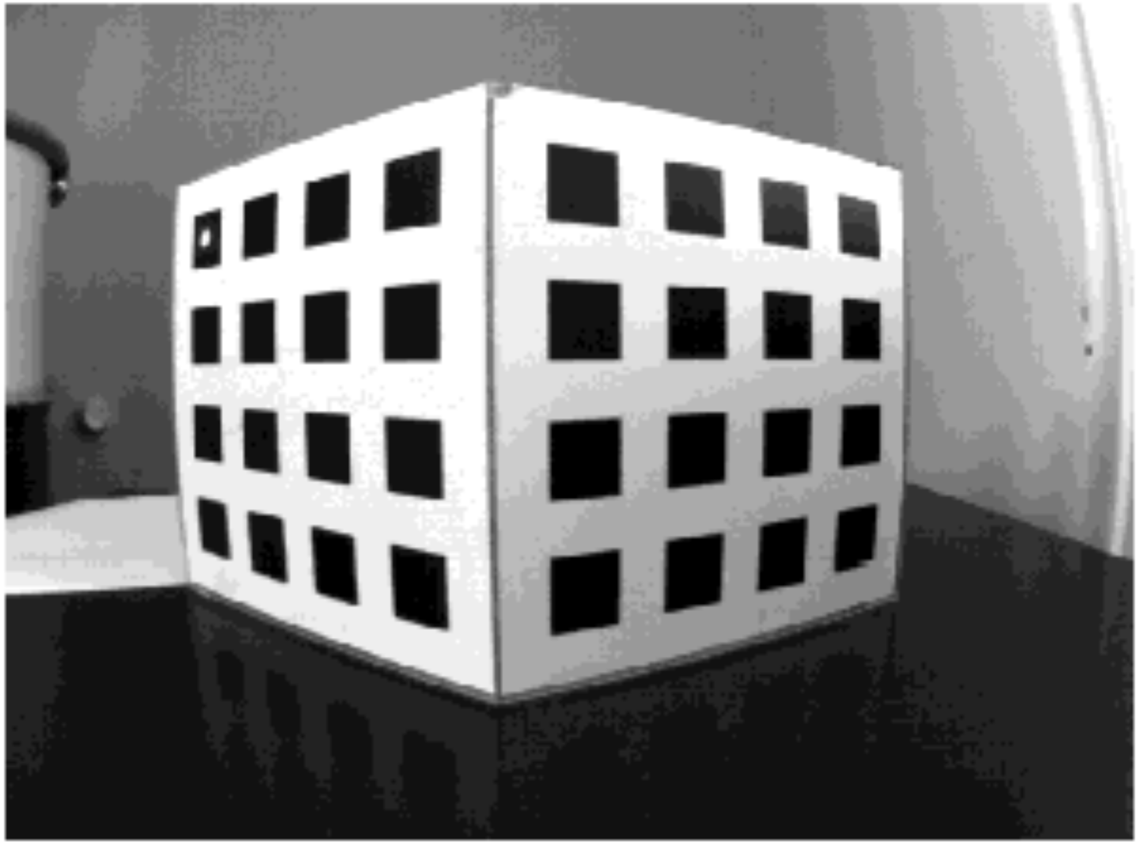
identity matrix

$$\Pi = \underbrace{\begin{bmatrix} -fs_x & 0 & x'_c \\ 0 & -fs_y & y'_c \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsics}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{0}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{T}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}}_{\text{translation}}$$

- The definitions of these parameters are not completely standardized
  - especially intrinsics—varies from one book to another

do we calculate the camera's calibration matrix, or measure?

Calibration target



The Opti-CAL Calibration Target Image

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

Camera calibration

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

$$u = \frac{\vec{m}_1 \cdot \vec{P}}{\vec{m}_3 \cdot \vec{P}}$$

$$v = \frac{\vec{m}_2 \cdot \vec{P}}{\vec{m}_3 \cdot \vec{P}}$$

So for each feature point, i, we have:

$$(\vec{m}_1 - u_i \vec{m}_3) \cdot \vec{P}_i = 0$$

$$(\vec{m}_2 - v_i \vec{m}_3) \cdot \vec{P}_i = 0$$

Camera calibration

Stack all these measurements of i=1…n points

$$(\vec{m}_1 - u_i \vec{m}_3) \cdot \vec{P}_i = 0$$

$$(\vec{m}_2 - v_i \vec{m}_3) \cdot \vec{P}_i = 0$$

into a big matrix (cluttering vector arrows omitted from P and m):

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

Camera calibration

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

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

Camera calibration

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

Q m = 0

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

The minimum eigenvector of the matrix Q^TQ gives us that (see Forsyth&Ponce, 3.1), because it is the unit vector x that minimizes $x^T Q^TQ\ x$.

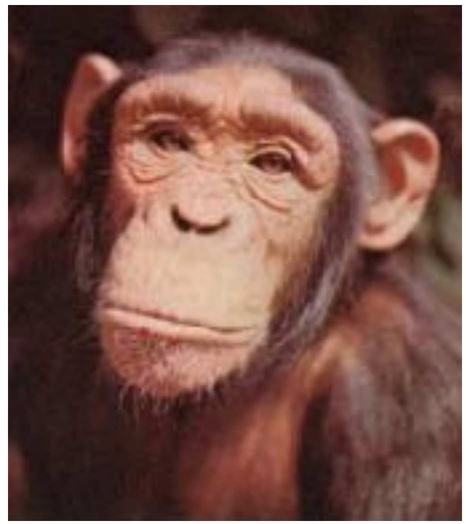
Once you have the M matrix, can recover the intrinsic and extrinsic parameters.

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

Vision systems

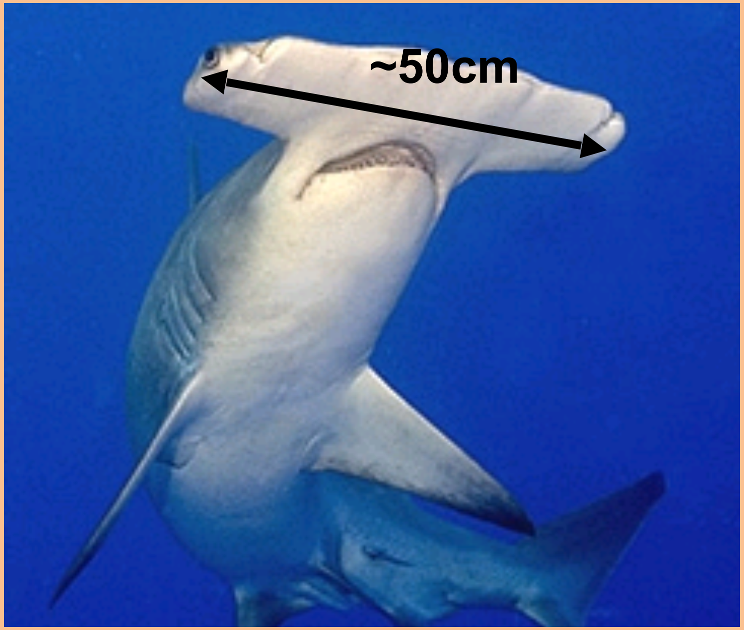
One camera

Two cameras



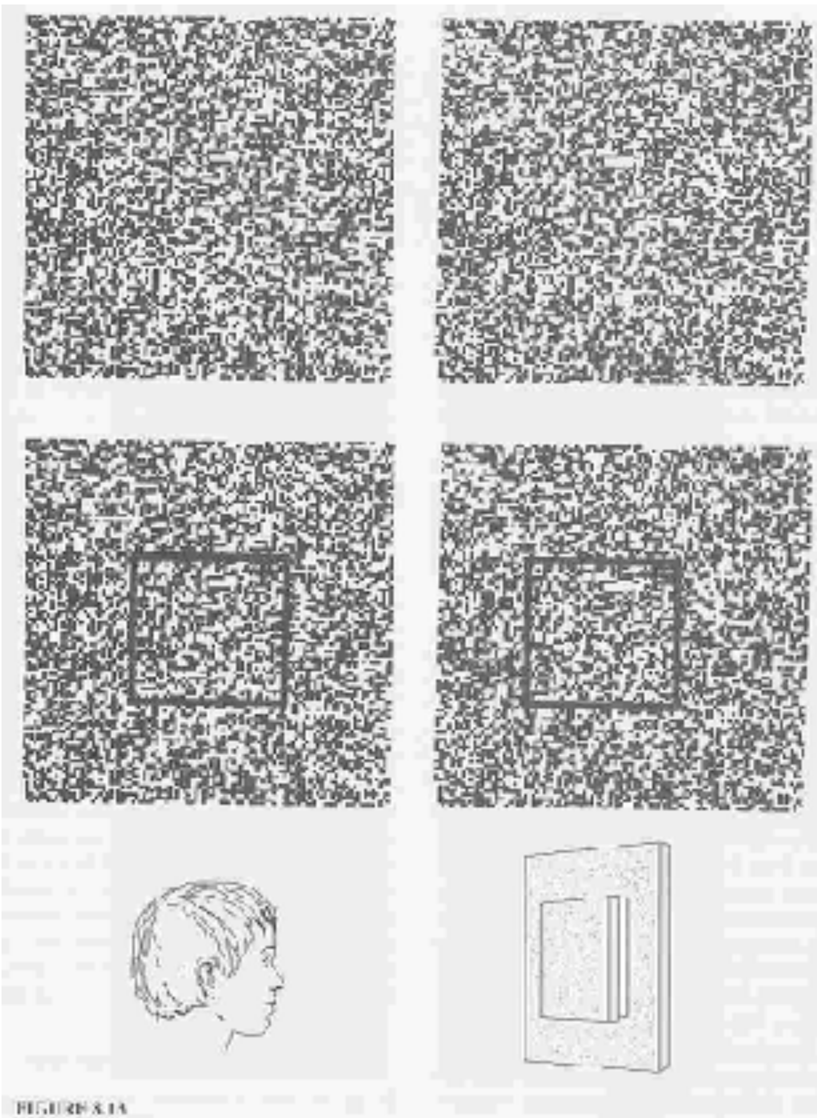
N cameras

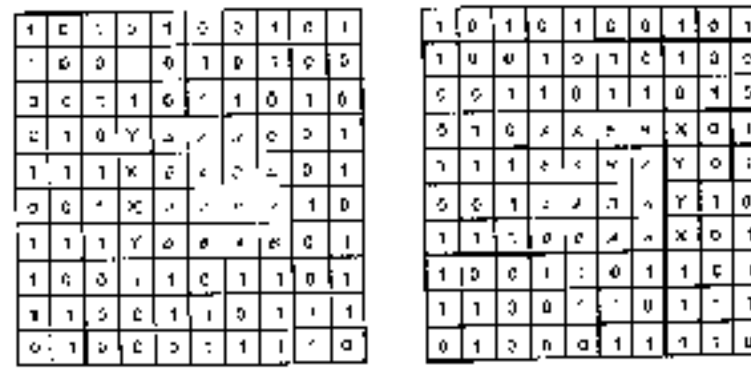
Stereo vision



Depth without objects

Random dot stereograms (Bela Julesz)





Julesz, 1971

Stereo photography and stereo viewers

Take two pictures of the same subject from two slightly different viewpoints and display so that each eye sees only one of the images.

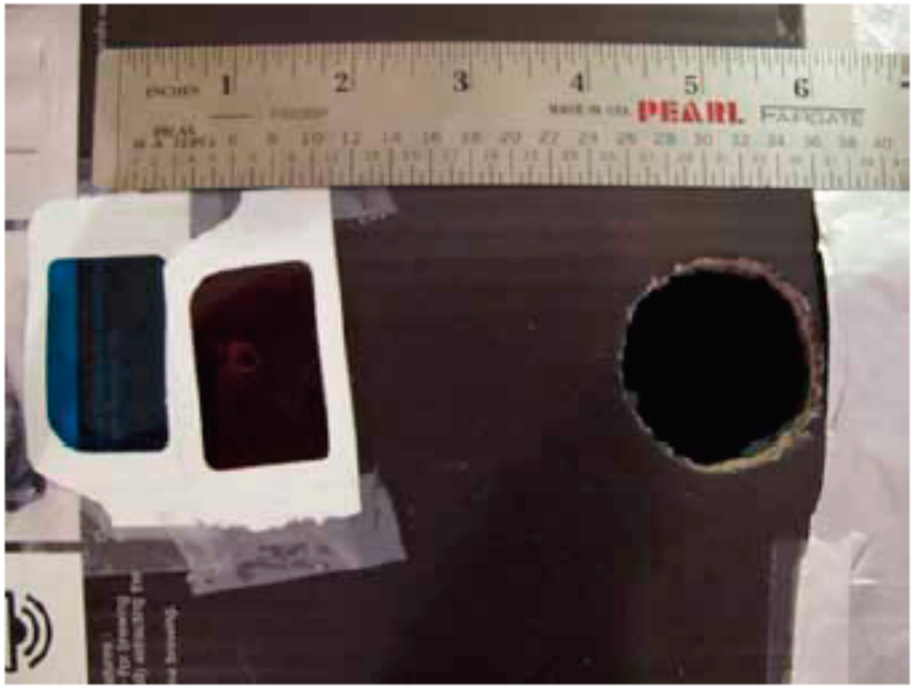
Image courtesy of fisher-price.com

Slide credit: Kristen Grauman

Public Library, Stereoscopic Looking Room, Chicago, by Phillips, 1923

de credit: Kristen Grauman

Anaglyph pinhole camera



Autostereograms

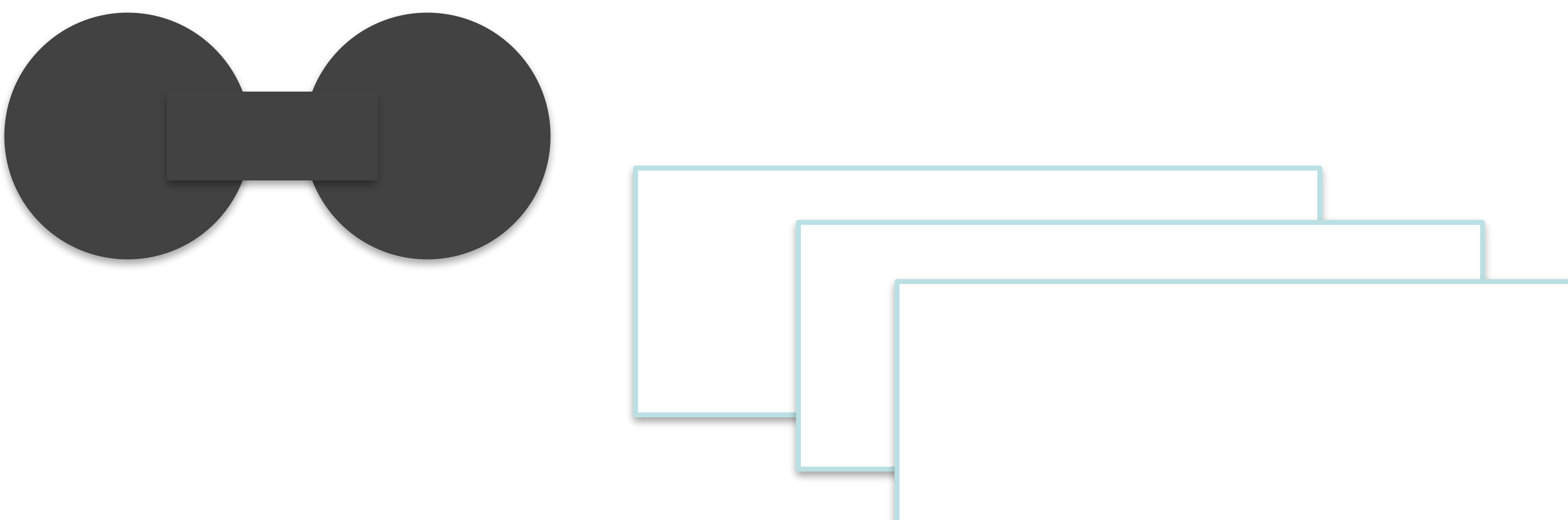
Exploit disparity as depth cue using single image.

(Single image random dot stereogram, Single image stereogram)

Slide credit: Kristen Grauman

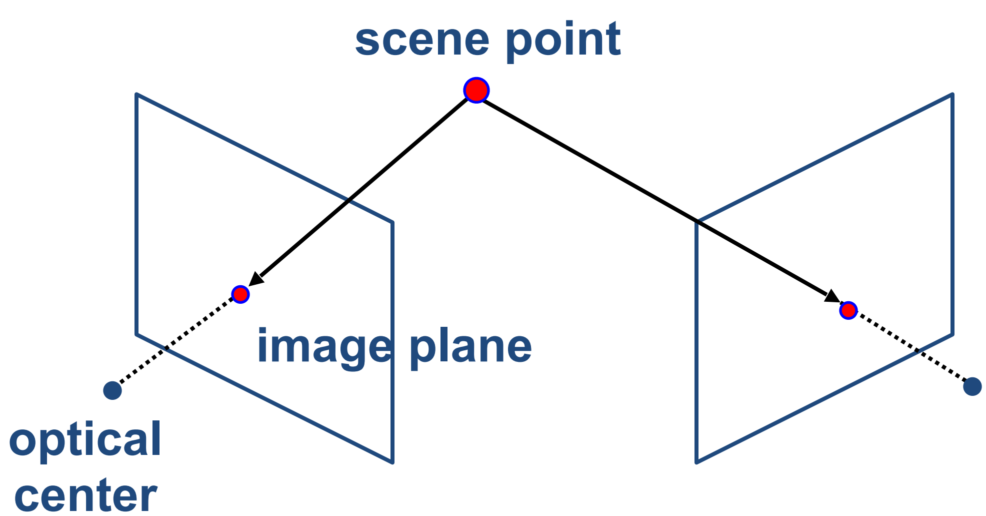
My conundrum regarding stereo displays

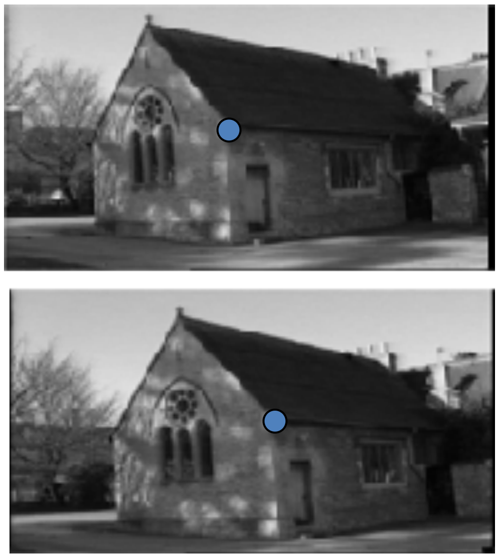
Real 3d scenes often look to me like thin, flat layers, stacked in depth. Why is that?



Estimating depth with stereo

- **Stereo**: shape from disparities between two views
- We'll need to consider:
  - Info on camera pose ("calibration")
  - Image point correspondences

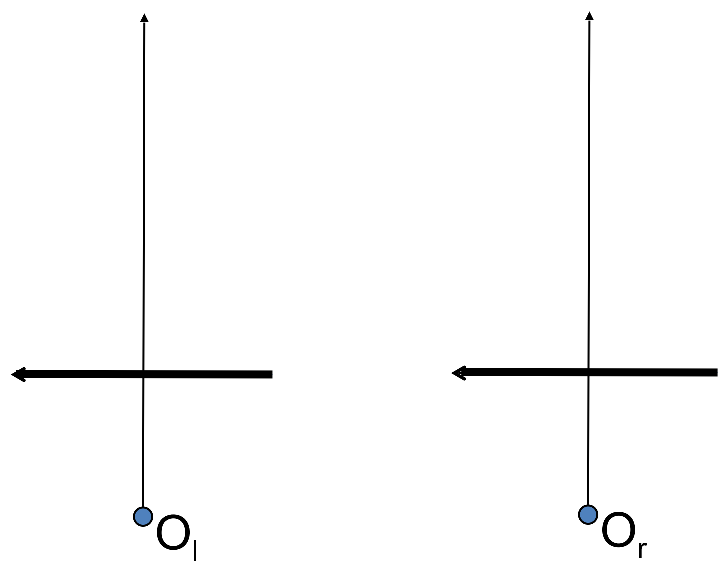


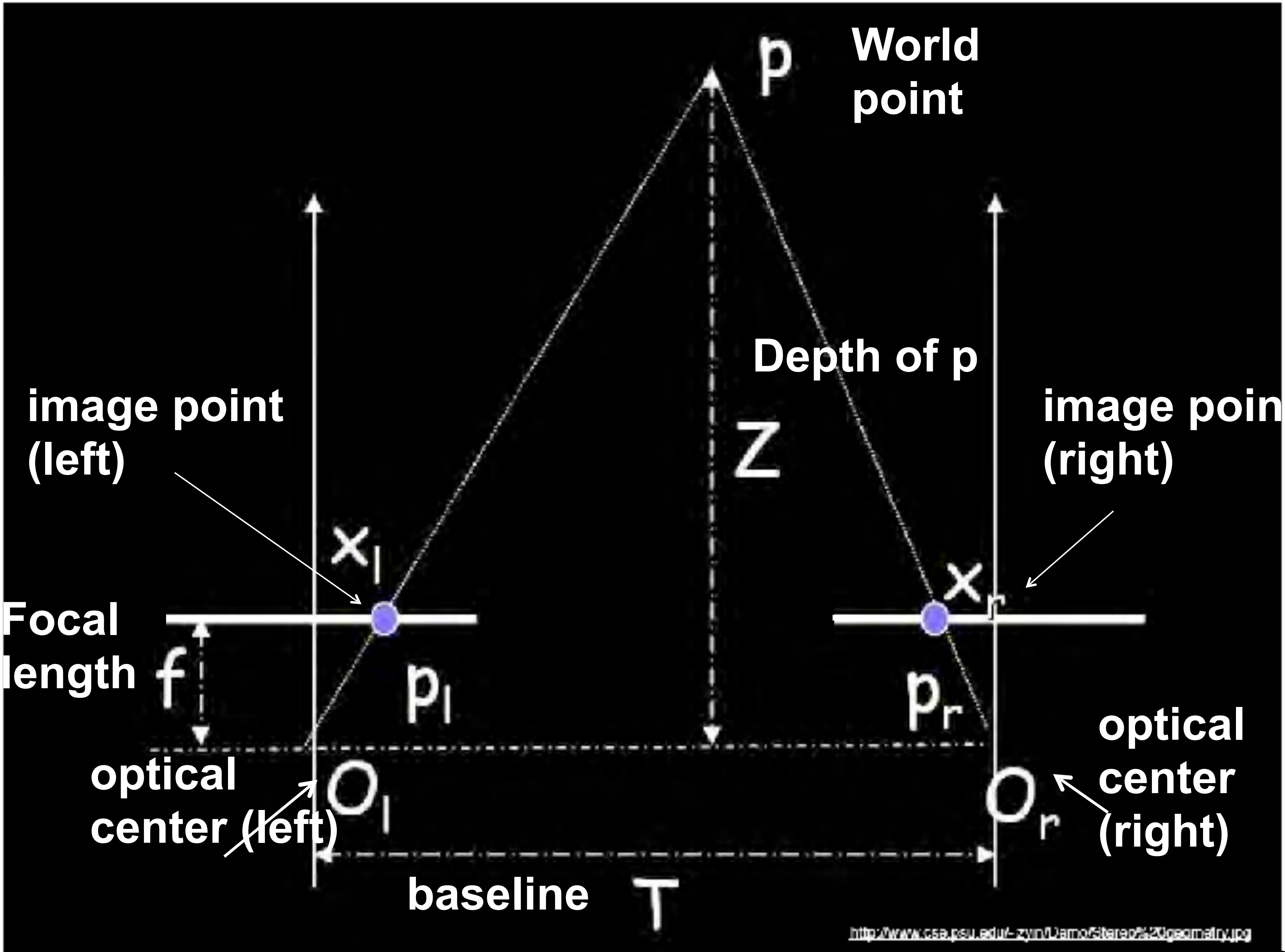


Slide credit: Kristen Grauman

Geometry for a simple stereo system

- Assume a simple setting:
  - Two identical cameras
  - parallel optical axes
  - known camera parameters (i.e., calibrated cameras).

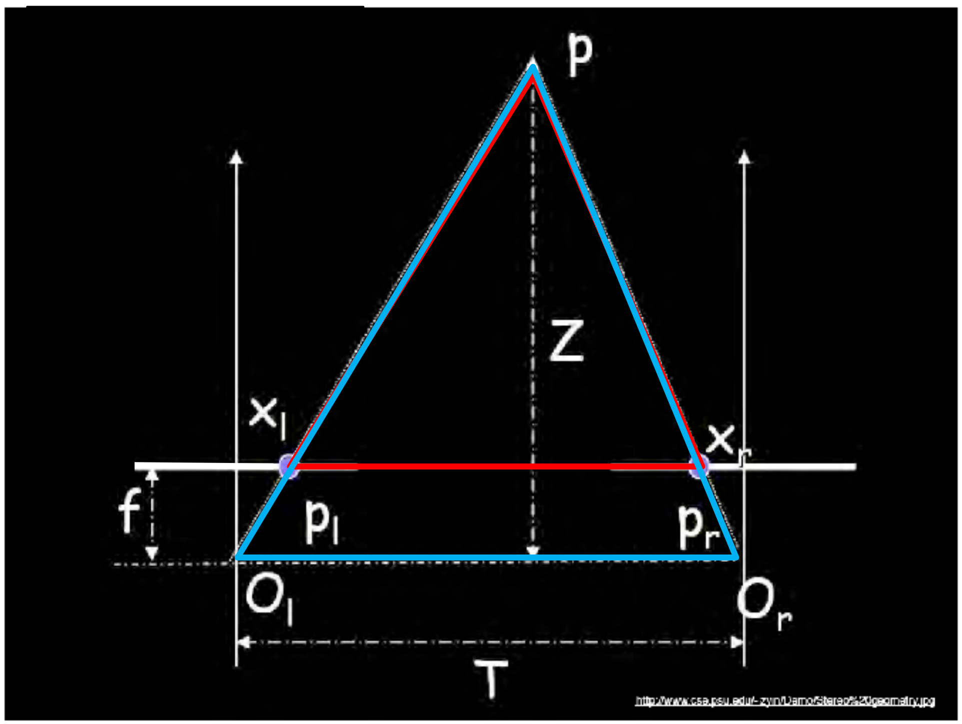




Slide credit: Kristen Grauman

Geometry for a simple stereo system

- Assume parallel optical axes, known camera parameters (i.e., calibrated cameras). We can triangulate via:



Similar triangles (p_l, P, p_r) and (O_l, P, O_r):

$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}$$

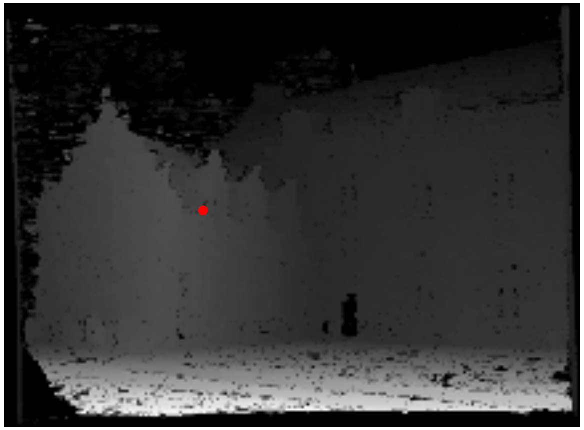
$$Z = f \frac{T}{x_r - x_l}$$

disparity

Slide credit: Kristen Grauman

Depth from disparity

image I(x,y) **Disparity map D(x,y)** **image I´(x´,y´)**

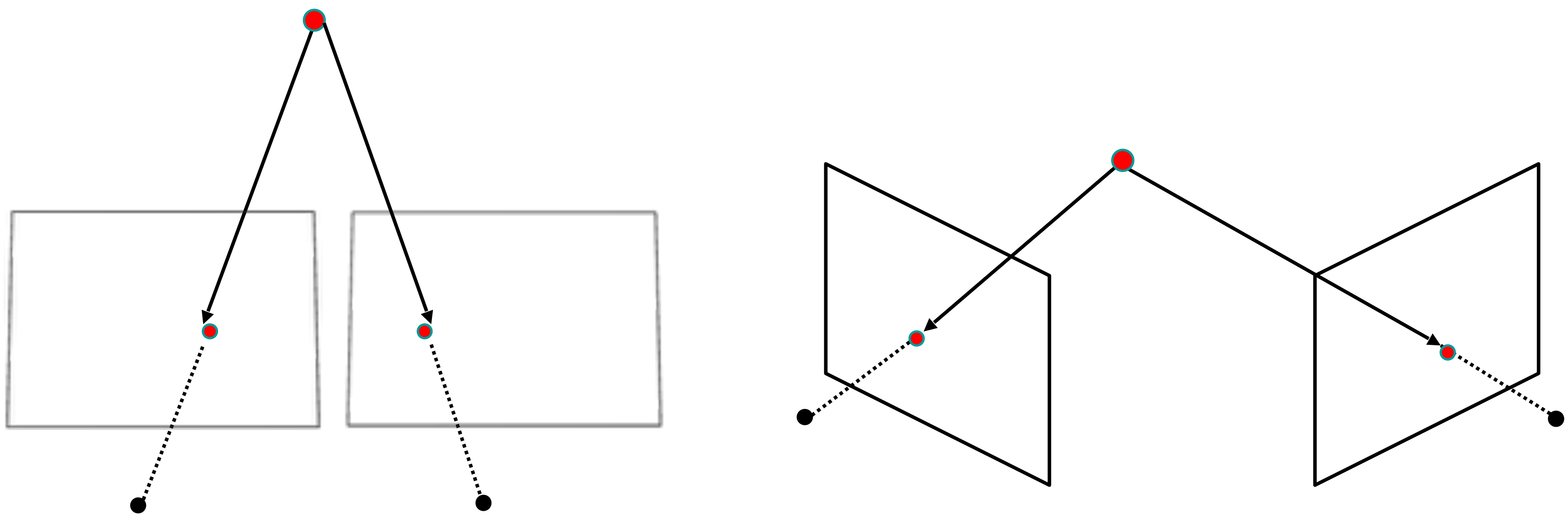


(x´,y´)=(x+D(x,y), y)

Slide credit: Kristen Grauman

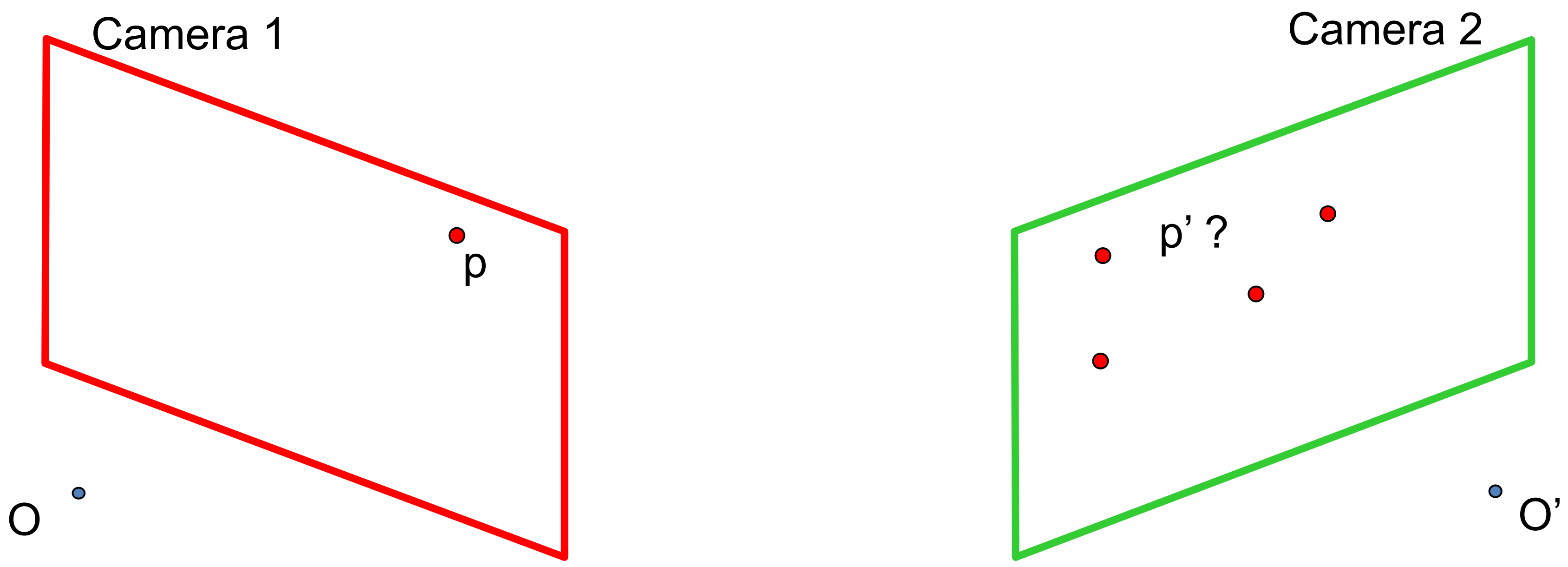
General case, with calibrated cameras

- The two cameras need not have parallel optical axes.



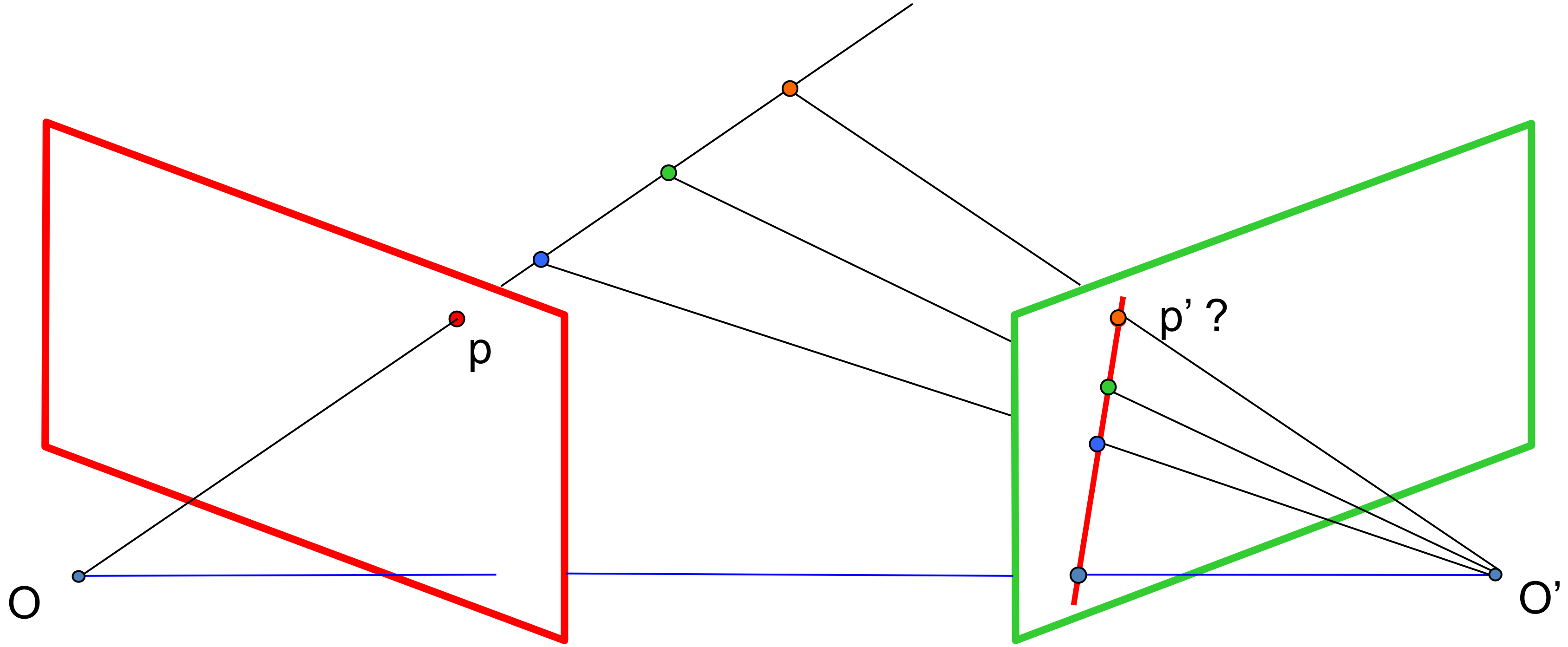
Vs.

Stereo correspondence constraints



If we see a point in camera 1, are there any constraints on where we will find it on camera 2?

Epipolar constraint



Geometry of two views constrains where the corresponding pixel for some image point in the first view must occur in the second view:

It must be on the line carved out by a plane connecting the world point and optical centers.

Why is this useful?

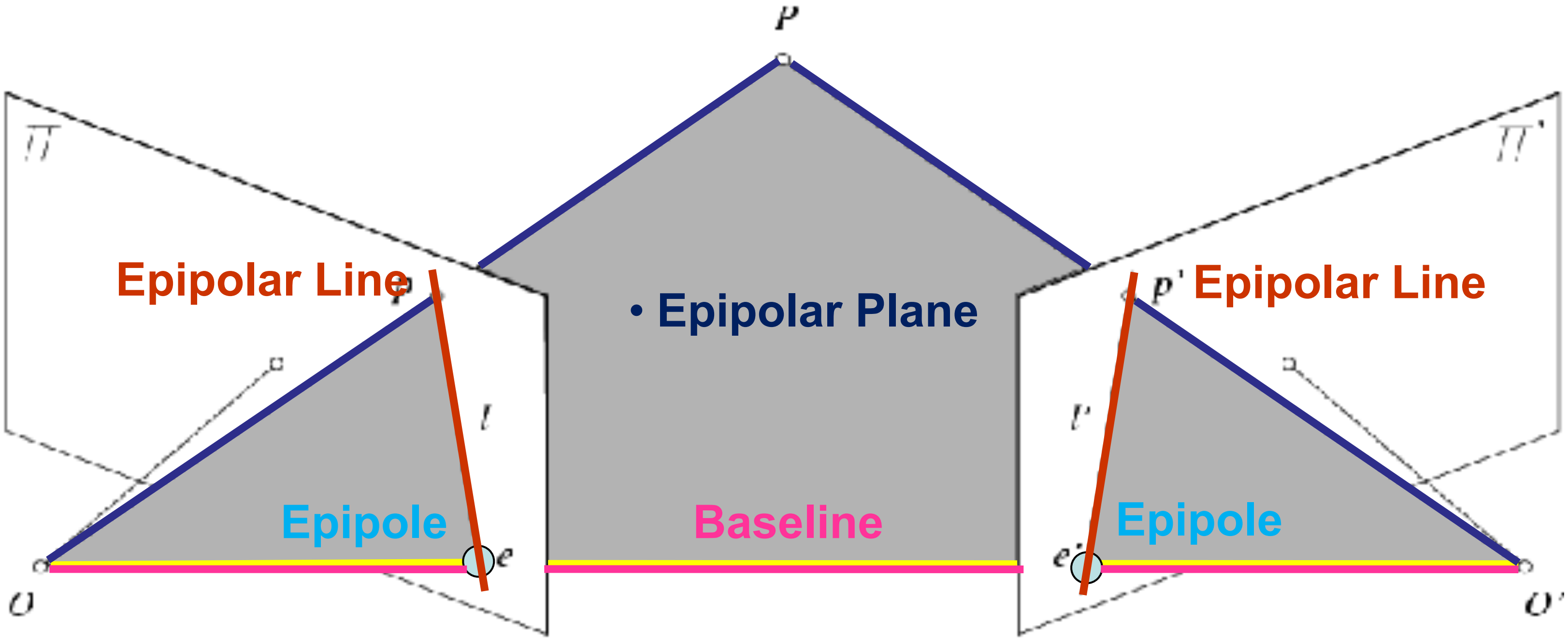
Epipolar constraint

This is useful because it reduces the correspondence problem to a 1D search along an epipolar line.

Image from Andrew Zisserman

Slide credit: Kristen Grauman

Epipolar geometry



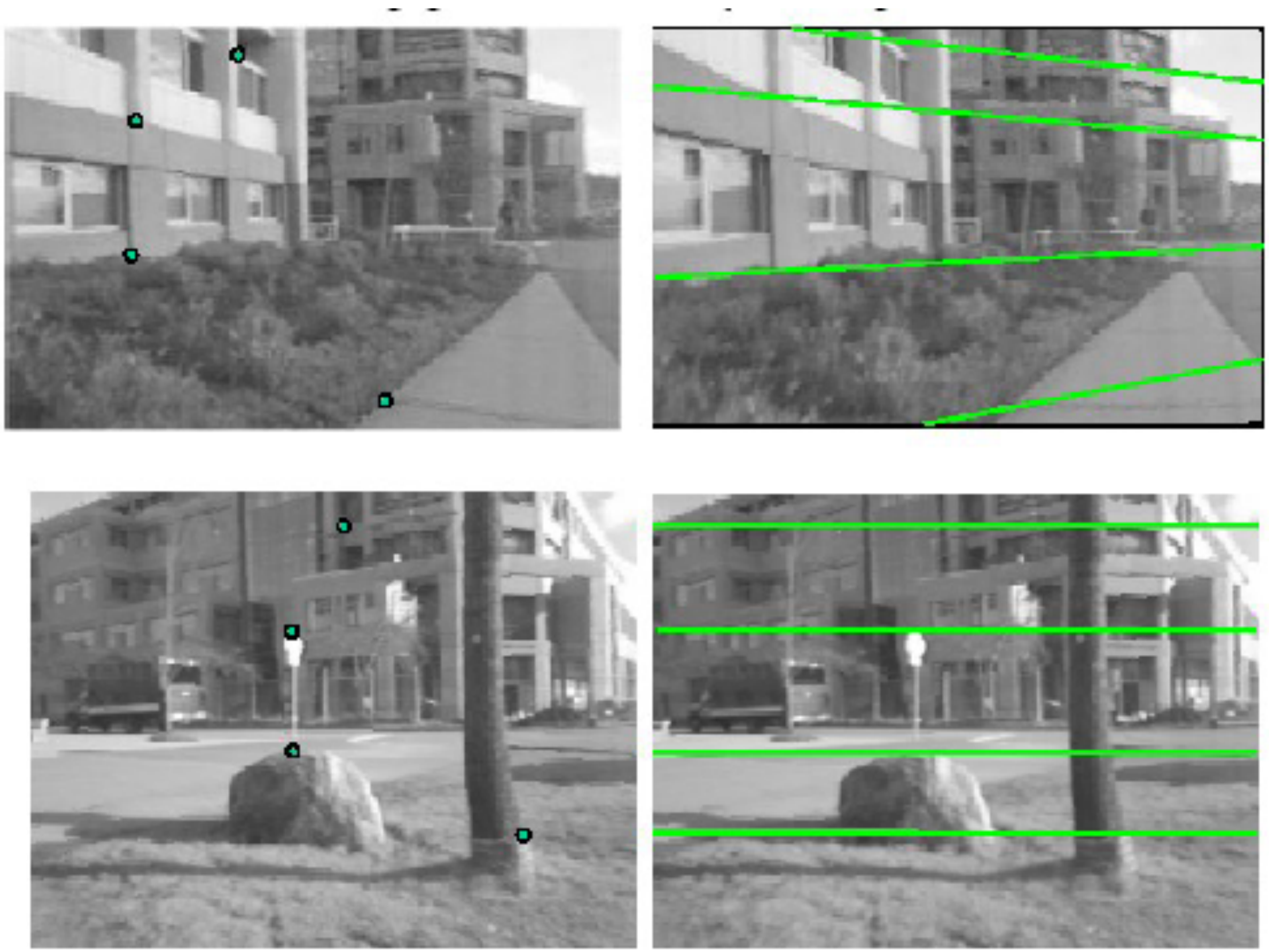
- **Epipolar plane**: plane containing baseline and world point
- **Epipole**: point of intersection of baseline with the image plane
- **Epipolar line**: intersection of epipolar plane with the image plane
- **Baseline**: line joining the camera centers

- All epipolar lines intersect at the epipole
- An epipolar plane intersects the left and right image planes in epipolar lines

http://www.ai.sri.com/~luong/research/Meta3DViewer/EpipolarGeo.html

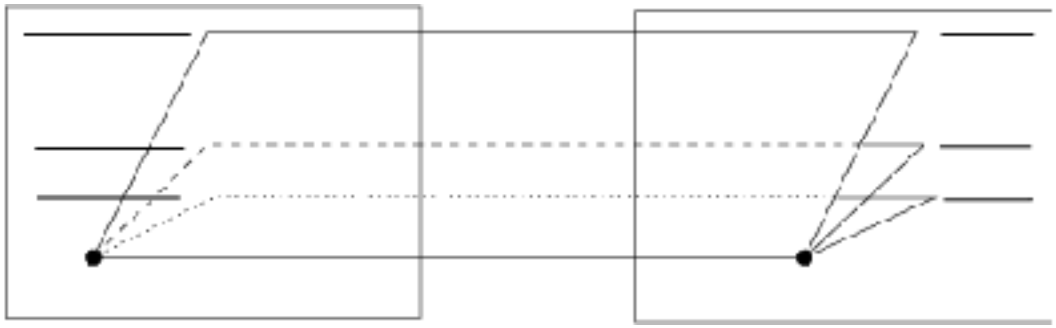
Slide credit: Kristen Grauman

Example

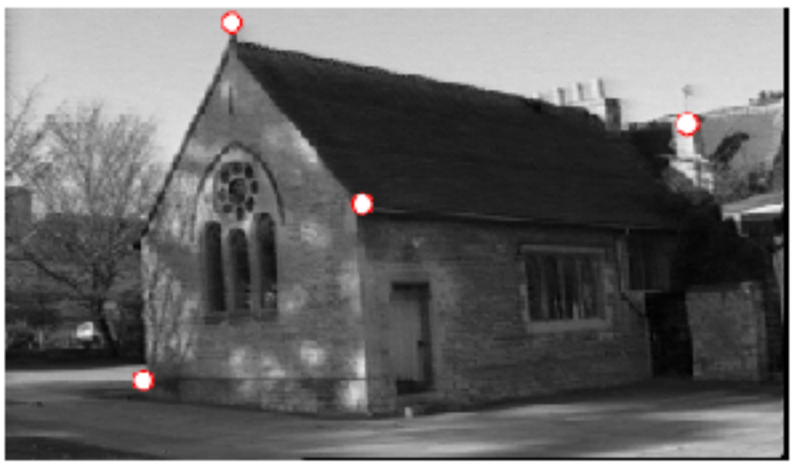


Slide credit: Kristen Grauman

Example: parallel cameras



Where are the epipoles?



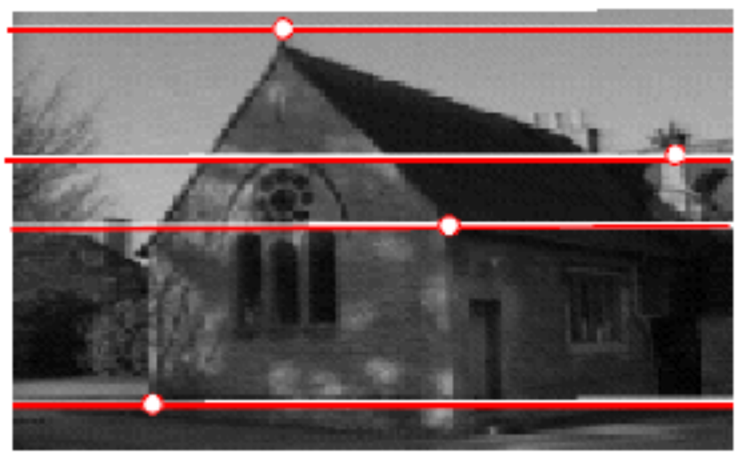


Figure from Hartley & Zisserman

Slide credit: Kristen Grauman

Example: converging cameras

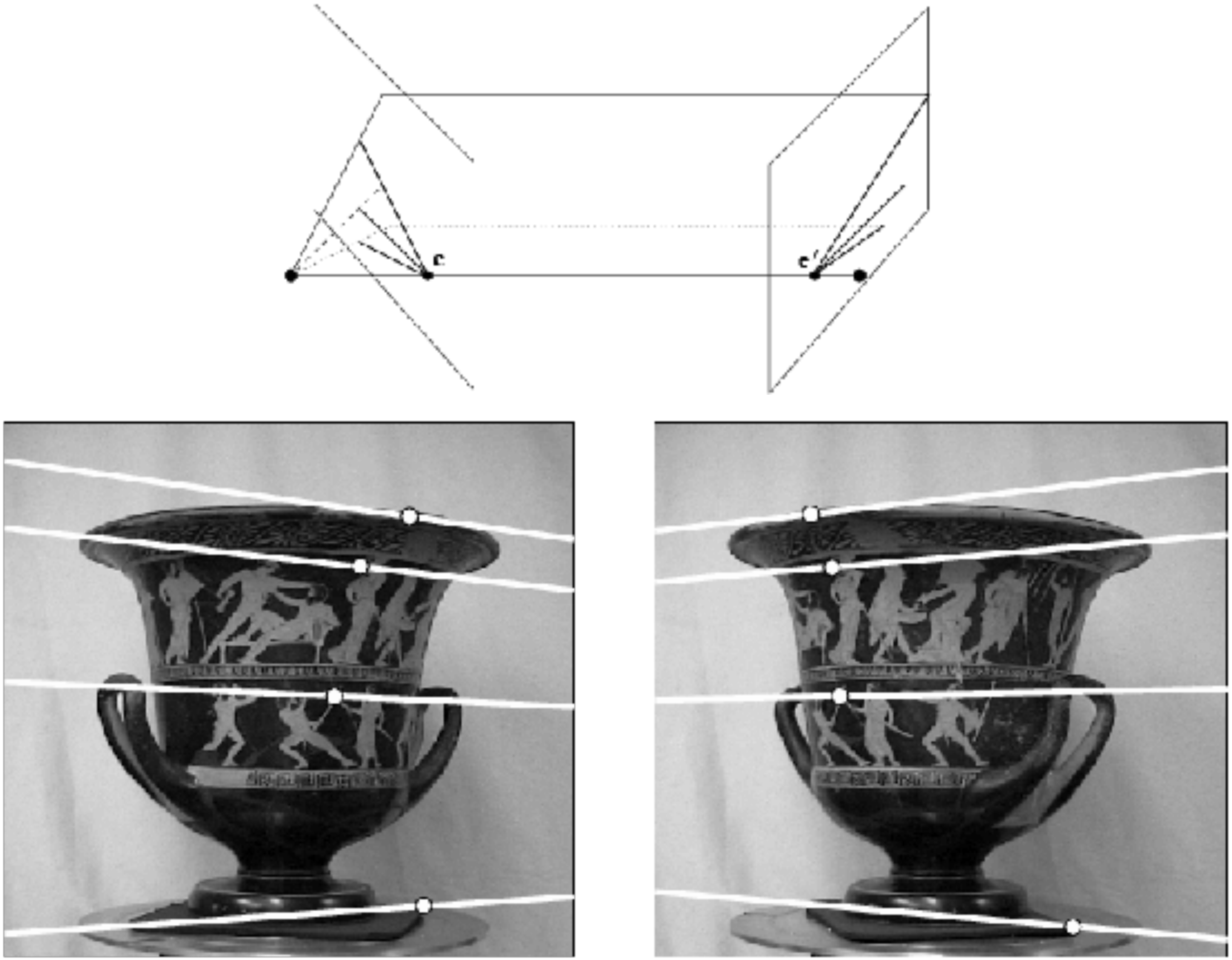


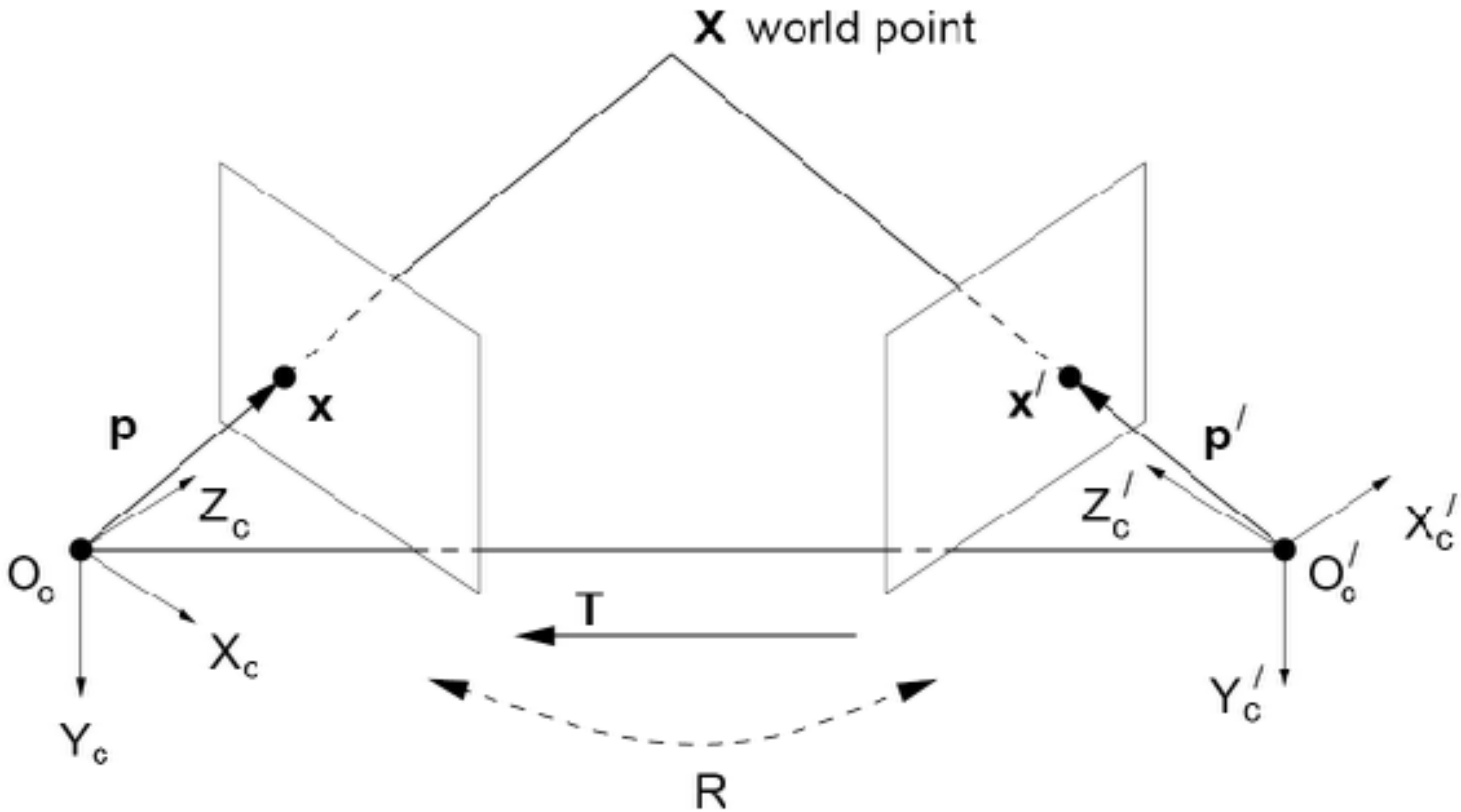
Figure from Hartley & Zisserman

Slide credit: Kristen Grauman

- So far, we have the explanation in terms of geometry.
- Now, how to express the epipolar constraints algebraically?

Slide credit: Kristen Grauman

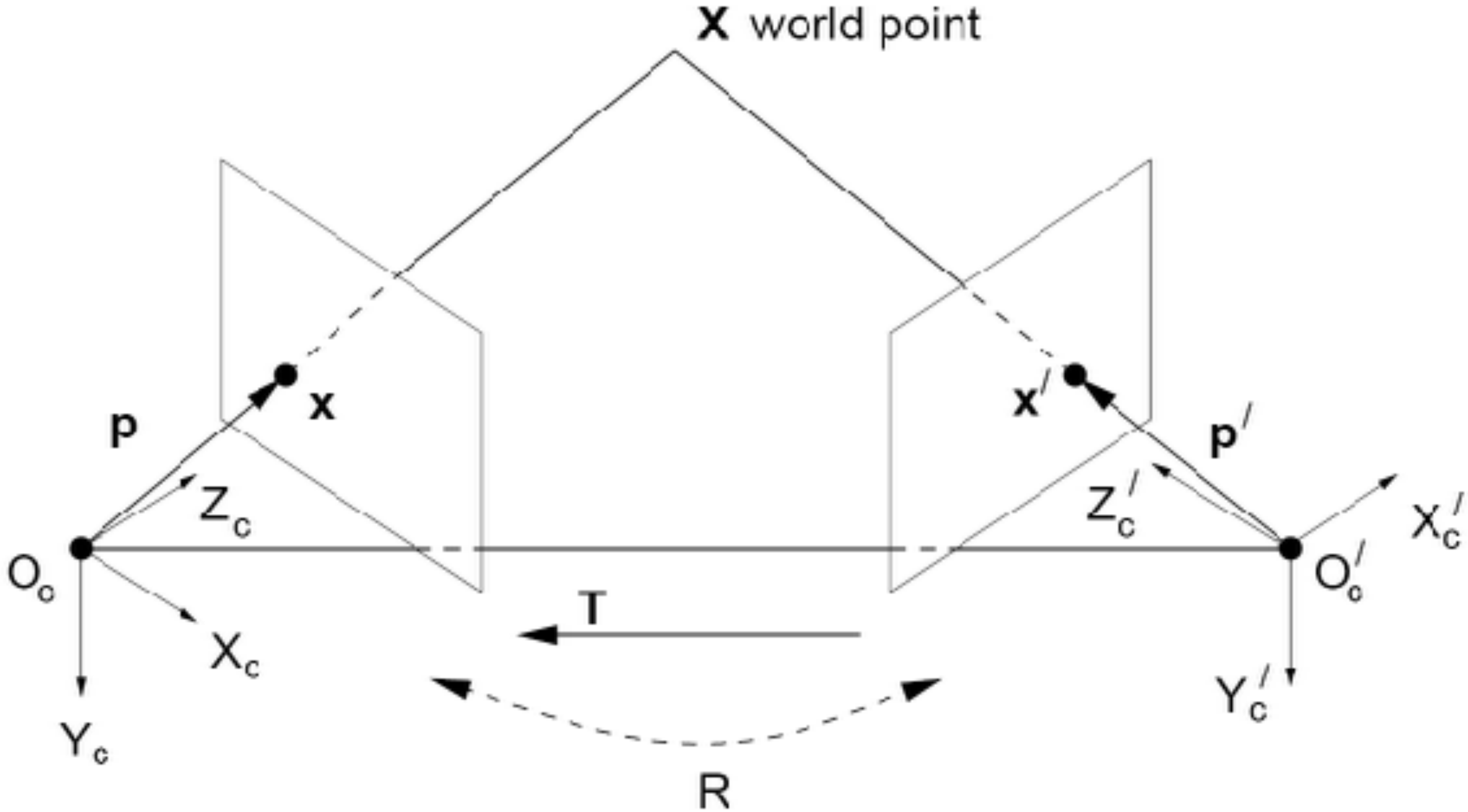
Stereo geometry, with calibrated cameras



Main idea

Slide credit: Kristen Grauman

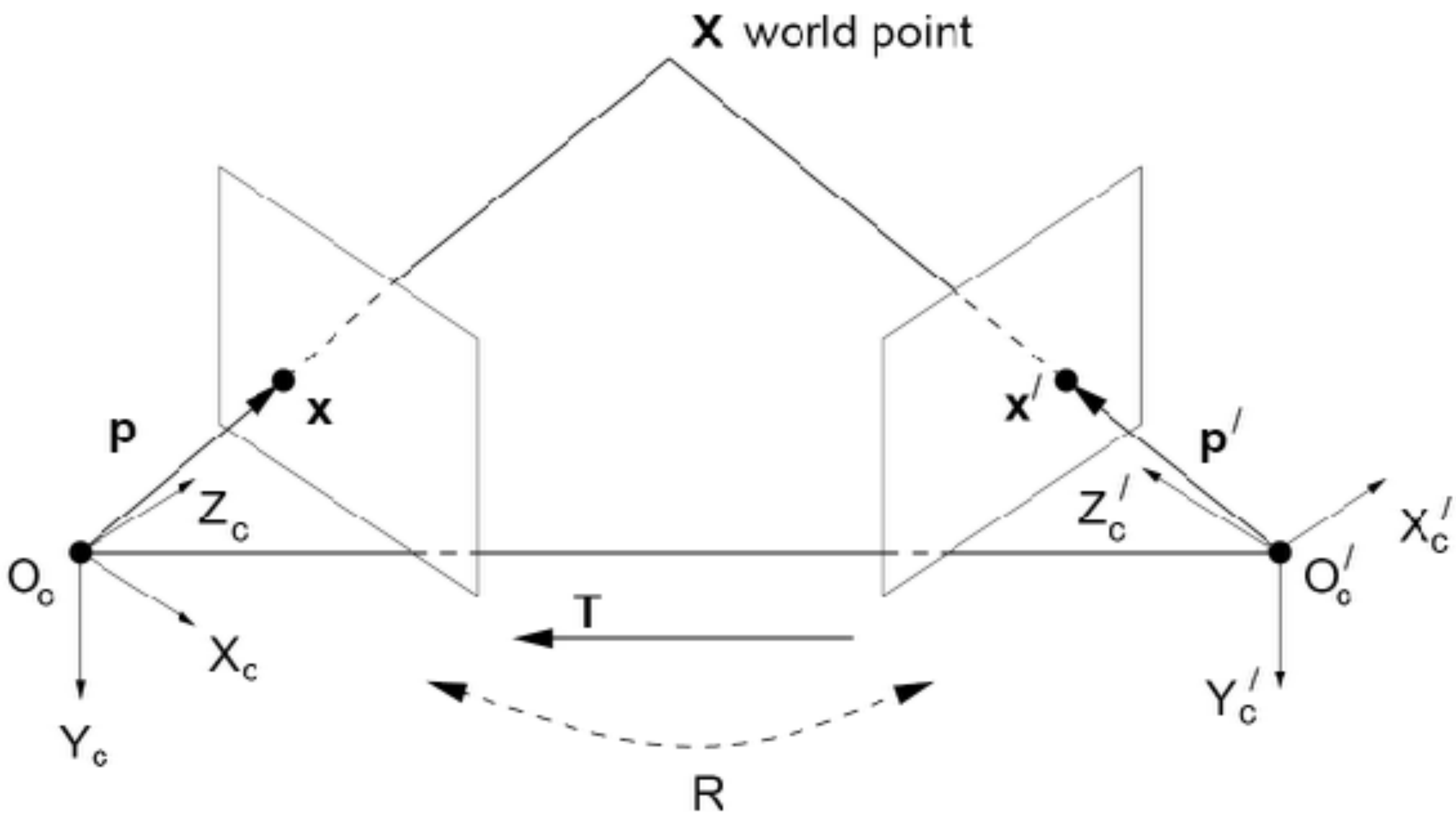
Stereo geometry, with calibrated cameras



If the stereo rig is calibrated, we know :
how to rotate and translate camera reference frame 1 to get to camera reference frame 2.
Rotation: 3 x 3 matrix R; translation: 3 vector T.

Slide credit: Kristen Grauman

Stereo geometry, with calibrated cameras

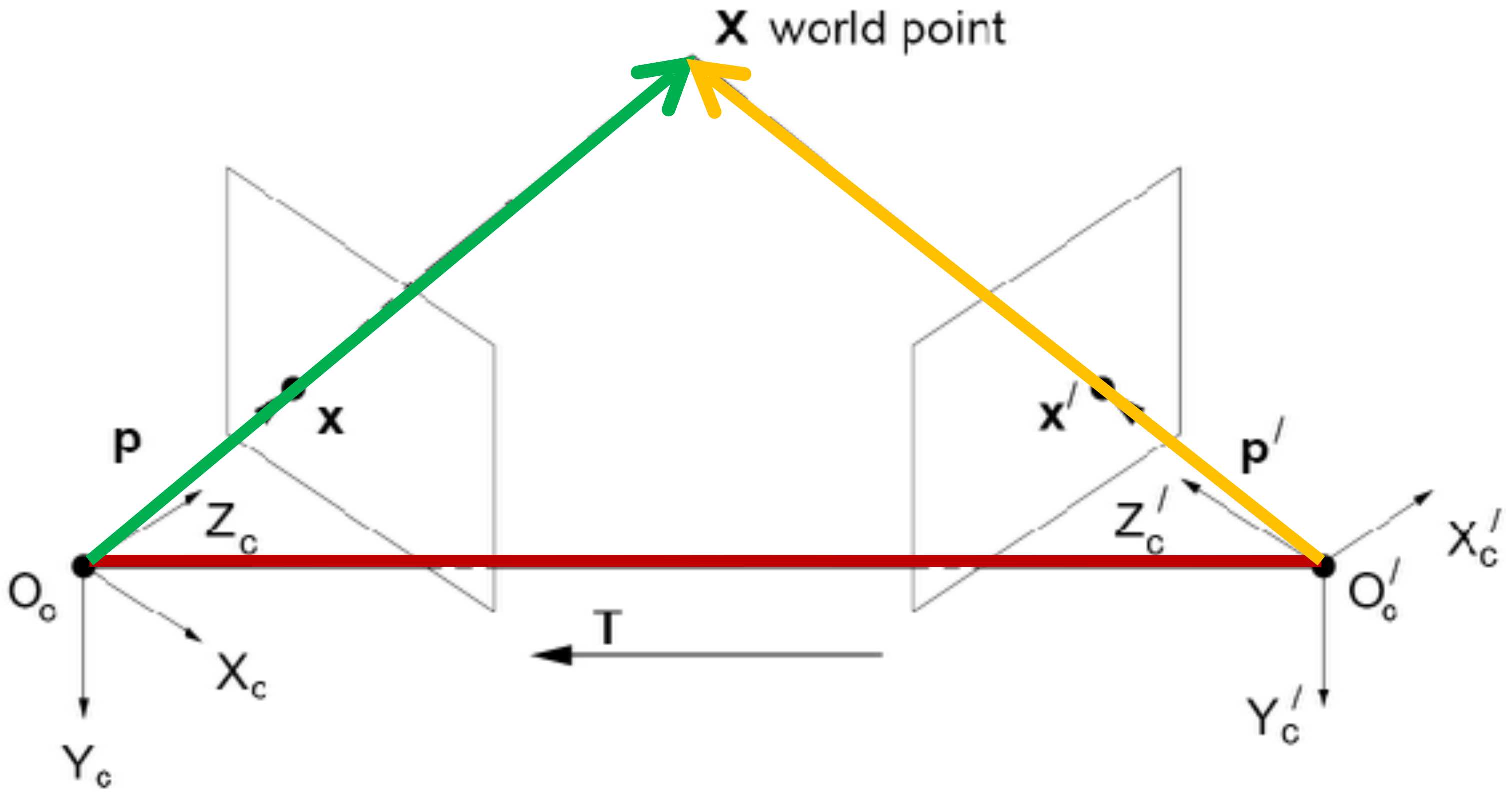


If the stereo rig is calibrated, we know :
how to rotate and translate camera reference frame 1 to get to camera reference frame 2.

$$X'_c = RX_c + T'$$

Slide credit: Kristen Grauman

From geometry to algebra



$$\boxed{\mathbf{X}'} = \mathbf{R}\boxed{\mathbf{X}} + \boxed{\mathbf{T}}$$

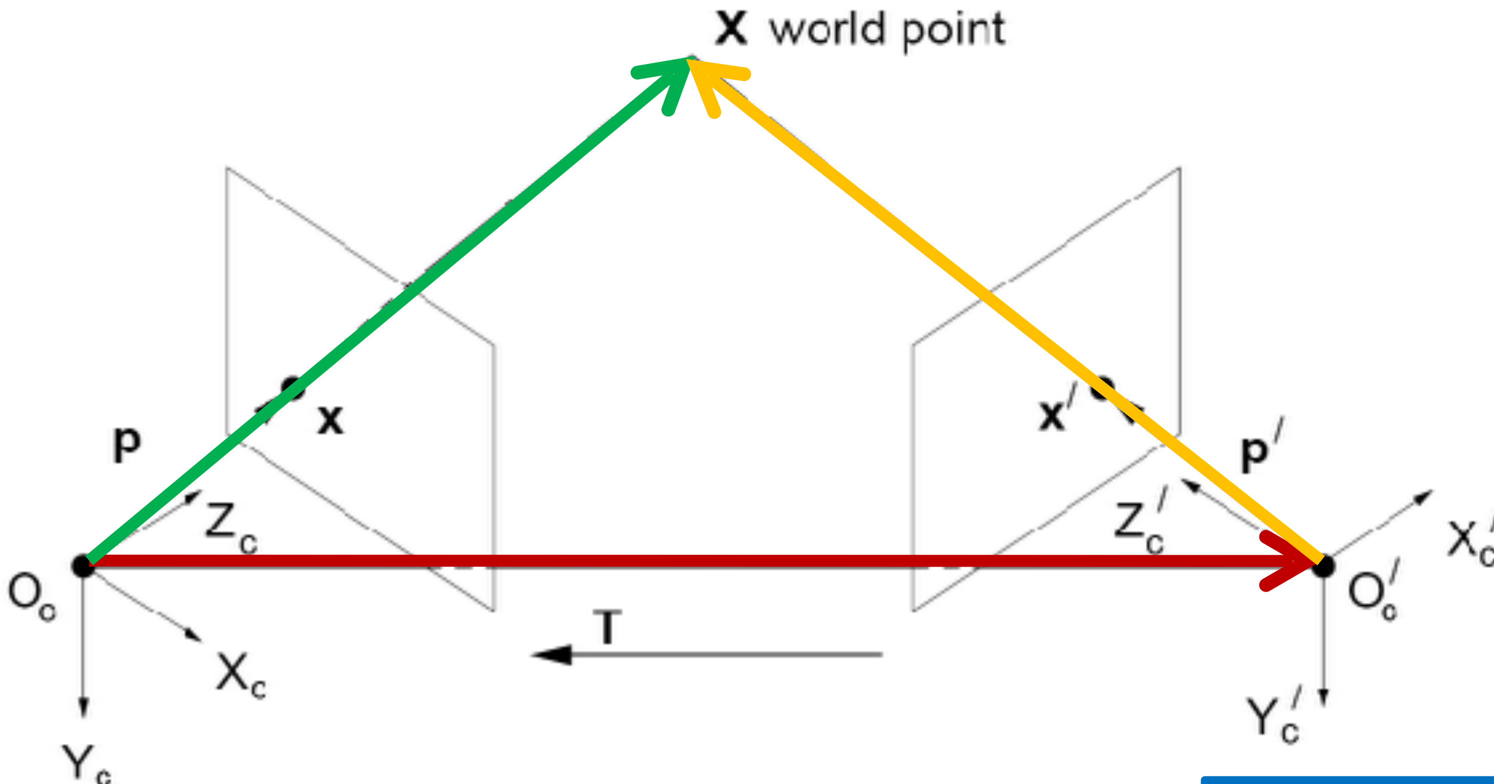
$$\underbrace{\mathbf{T} \times \mathbf{X}'}_{\text{Normal to the plane}} = \mathbf{T} \times \mathbf{R}\mathbf{X}$$

$$\mathbf{X}' \cdot \left(\mathbf{T} \times \mathbf{X}'\right) = \mathbf{X}' \cdot \left(\mathbf{T} \times \mathbf{R}\mathbf{X}\right)$$

$$= 0$$

Slide credit: Kristen Grauman

From geometry to algebra



$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{T}$$

$$\underbrace{\mathbf{T} \times \mathbf{X}'}_{\text{Normal to the plane}} = \mathbf{T} \times \mathbf{R}\mathbf{X} + \mathbf{T} \times \mathbf{T}$$

$$= \mathbf{T} \times \mathbf{R}\mathbf{X}$$

$$\mathbf{X}' \cdot (\mathbf{T} \times \mathbf{X}') = \boxed{\mathbf{X}' \cdot (\mathbf{T} \times \mathbf{R}\mathbf{X}) = 0}$$

Slide credit: Kristen Grauman

Aside: cross product

$$\vec{a} \times \bar{b} = \vec{c}$$

$$\bar{a} \cdot \bar{c} = 0$$

$$\bar{b} \cdot \bar{c} = 0$$

Vector cross product takes two vectors and returns a third vector that's perpendicular to both inputs.

So here, c is perpendicular to both a and b, which means the dot product = 0.

Slide credit: Kristen Grauman

Matrix form of cross product

$$\vec{a} \times \vec{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \vec{c}$$

$$\vec{a} \cdot \vec{c} = 0$$

$$\vec{b} \cdot \vec{c} = 0$$

Can be expressed as a matrix multiplication.

$$[a_x] = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

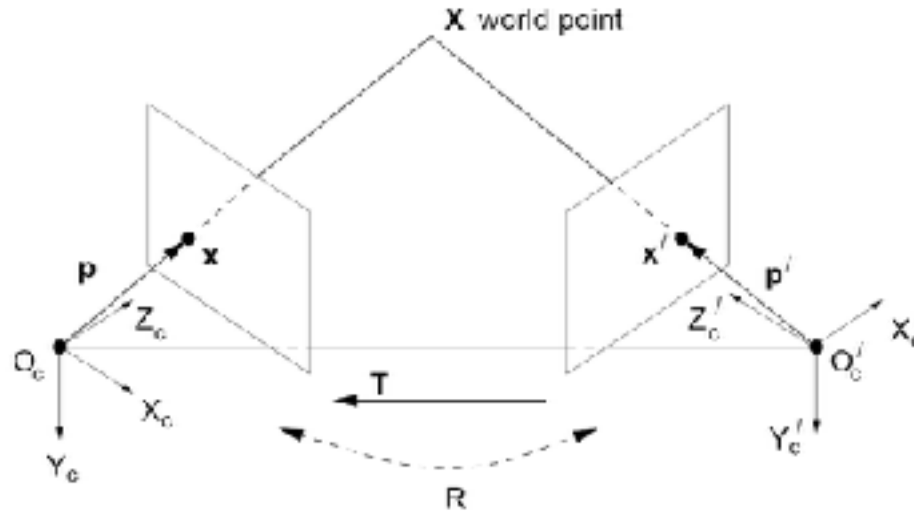
$$\boxed{\vec{a} \times \vec{b} = [a_x]\vec{b}}$$

Slide credit: Kristen Grauman

Essential matrix

$$\boxed{\mathbf{X}' \cdot \left(\mathbf{T} \times \mathbf{R}\mathbf{X}\right) = 0}$$

$$\mathbf{X}' \cdot \left(\mathbf{T}_x \; \mathbf{R}\mathbf{X}\right) = 0$$



Let $\mathbf{E} = \mathbf{T}_x\mathbf{R}$

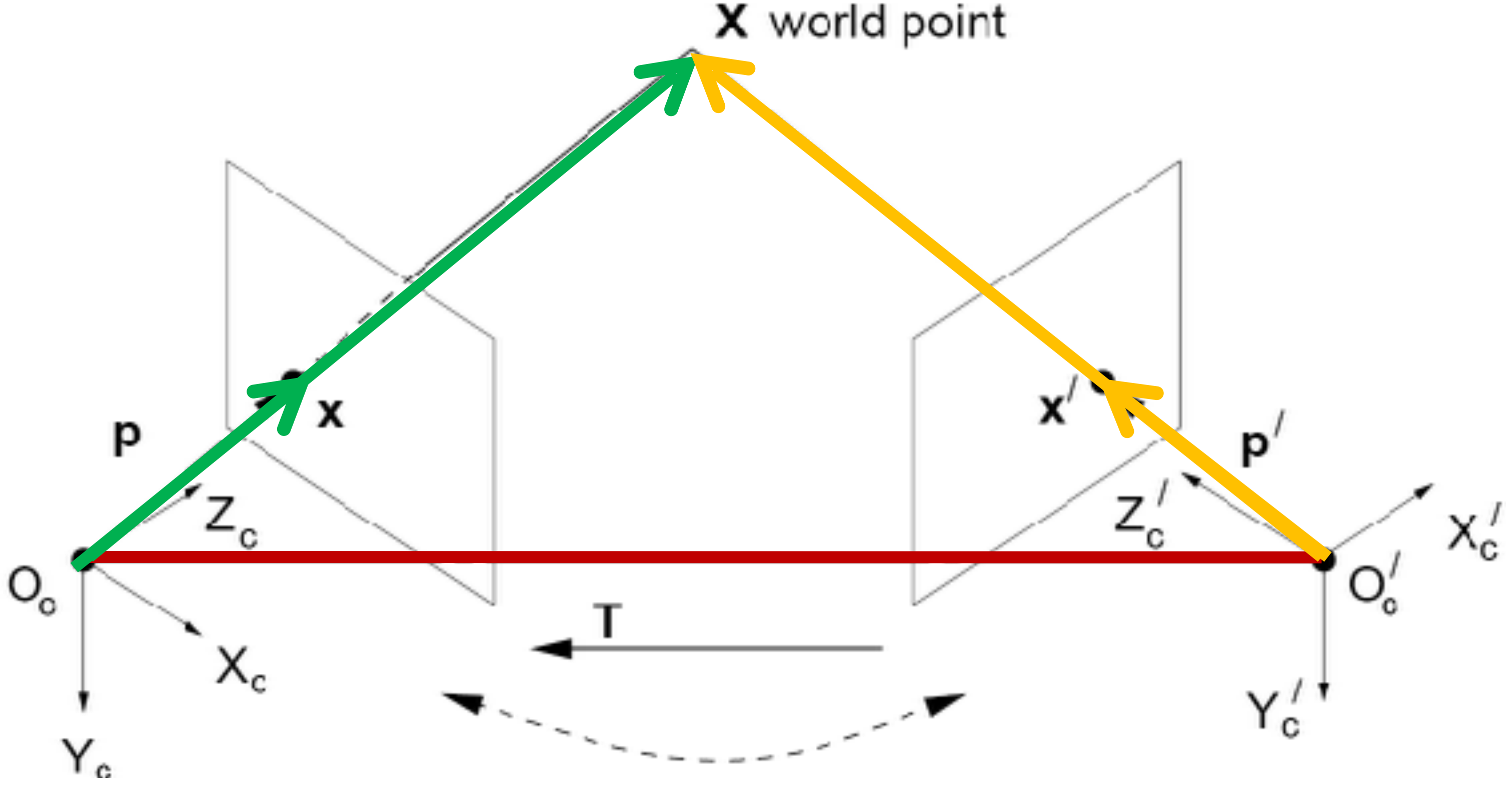
$$\mathbf{X}'^T \mathbf{E}\mathbf{X} = 0$$

E is called the essential matrix, and it relates corresponding image points between both cameras, given the rotation and translation.

If we observe a point in one image, its position in the other image is constrained to lie on line defined by above.

Note: these points are in camera coordinate systems.

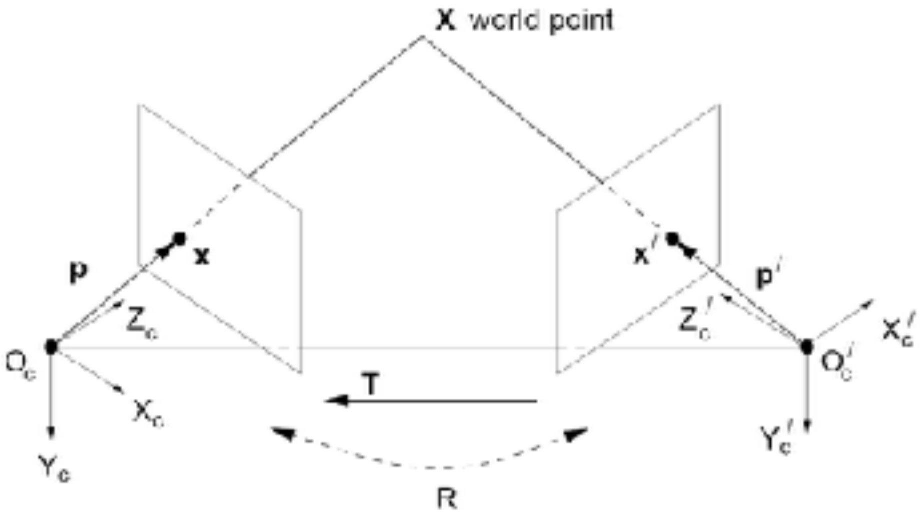
x and x' are scaled versions of X and X'



$$X' \cdot (T' \times RX) = 0$$

$$X' \cdot (T'_x RX) = 0$$

Let $$E = T'_x R$$



$$\mathbf{X}'^T \mathbf{E} \mathbf{X} = 0$$

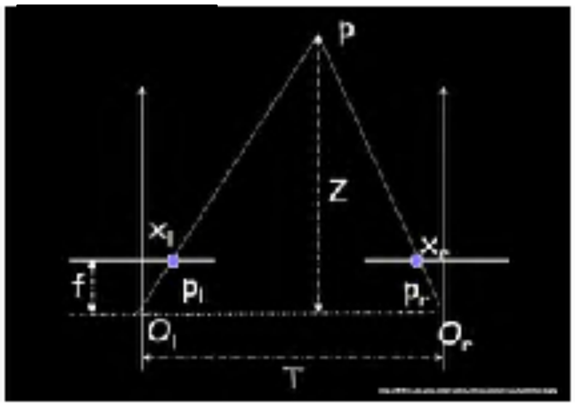
$$x'^T E x = 0$$ pts x and x' in the image planes are scaled versions of X and X'.

E is called the essential matrix, and it relates corresponding image points between both cameras, given the rotation and translation.

If we observe a point in one image, its position in the other image is constrained to lie on line defined by above.

Note: these points are in camera coordinate systems.

Essential matrix example: parallel cameras



$$\mathbf{R} =$$

$$\mathbf{T} =$$

$$\mathbf{E} = [\mathbf{T}_\mathbf{x}]\mathbf{R} =$$

$$\mathbf{p} = [x, y, f]$$

$$\mathbf{p}' = [x', y', f]$$

$$\mathbf{p}'^{\mathrm{T}}\mathbf{E}\mathbf{p} = 0$$

For the parallel cameras, image of any point must li on same horizontal line in each image plane.

Slide credit: Kristen Grauman

image I(x,y)

Disparity map D(x,y)

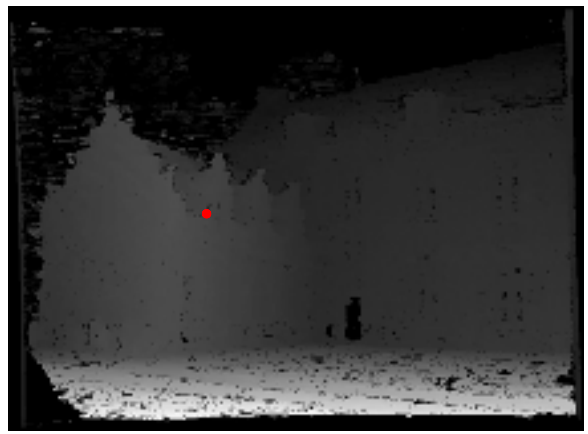


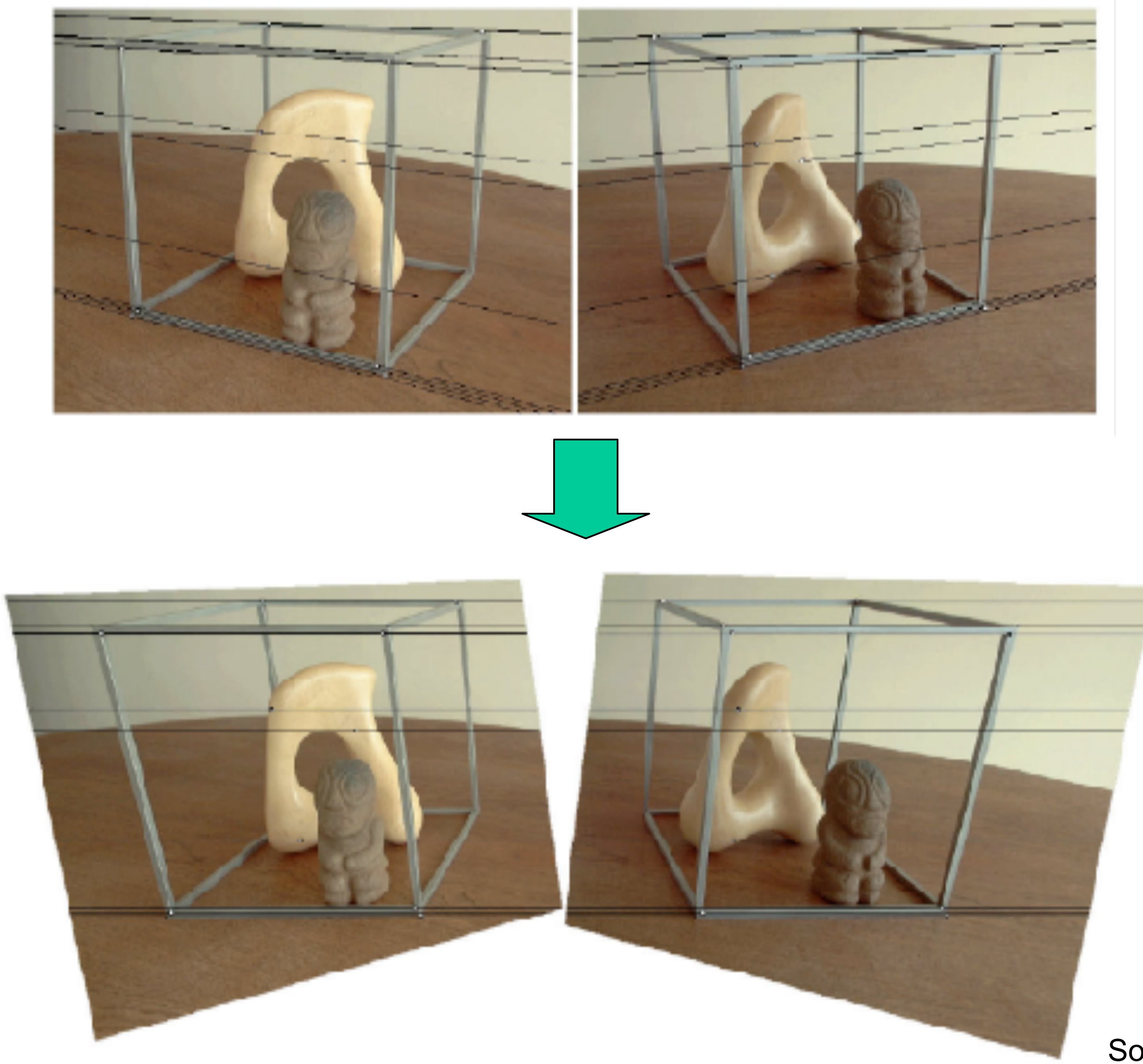
image I´(x´,y´)

$$(x´,y´)=(x+D(x,y),y)$$

What about when cameras' optical axes are not parallel?

Slide credit: Kristen Grauman

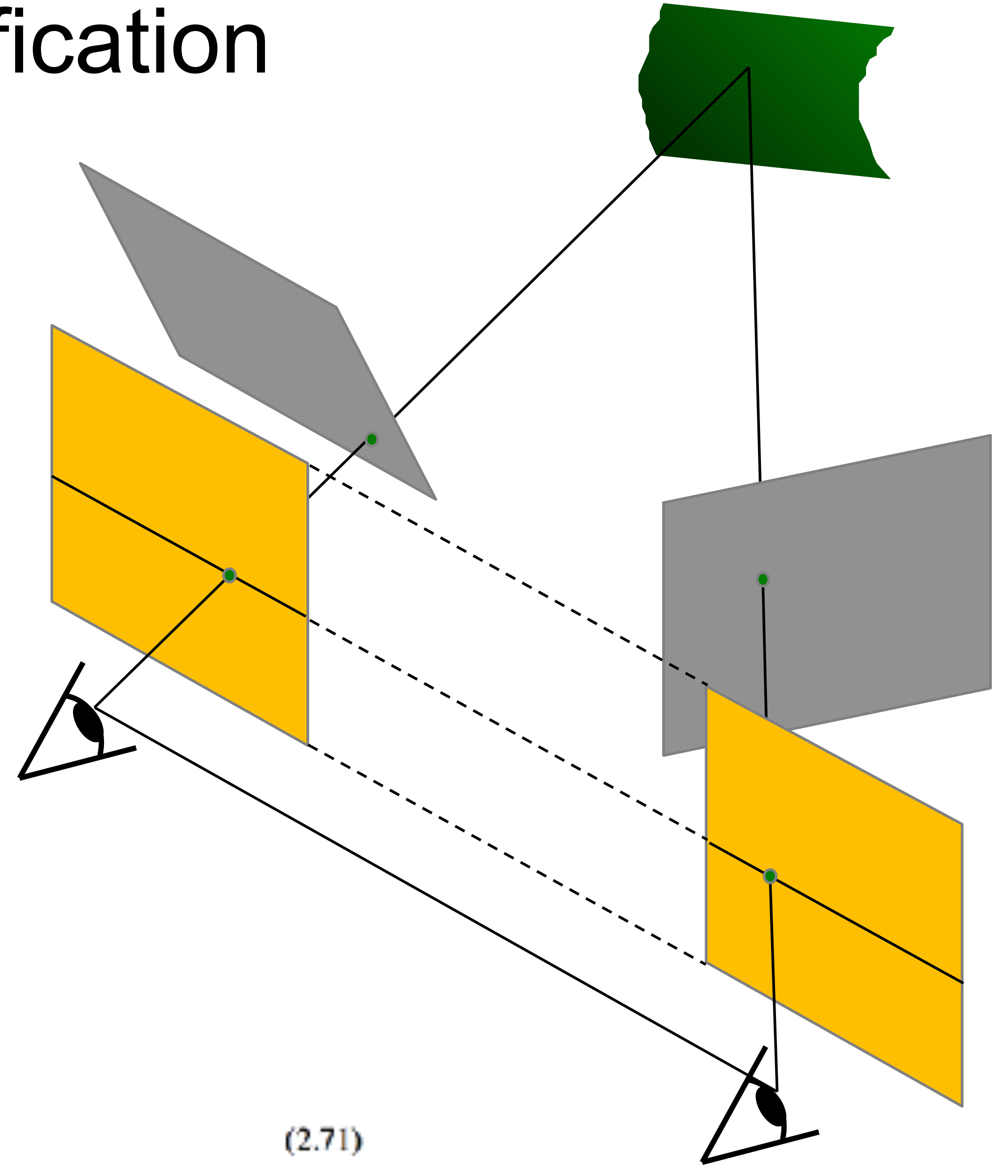
Stereo image rectification: example



Source: Alyosha Efros

Stereo image rectification

In practice, it is convenient if image scanlines (rows) are the epipolar lines.



- Reproject image planes onto a common plane parallel to the line between optical centers
- Pixel motion is horizontal after this transformation
- Two homographies (3x3 transforms), one for each input image reprojection
- See Szeliski book, Sect. 2.1.5, Fig. 2.12, and “Mapping from one camera to another” p. 56:

The mapping equation (2.70) thus reduces to

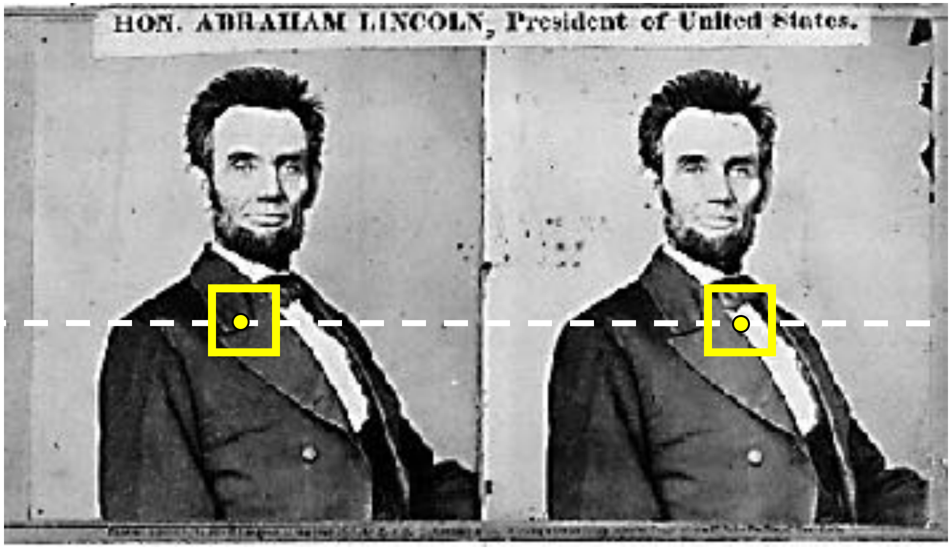
$$\tilde{x}_1 \sim \tilde{H}_{10}\tilde{x}_0, \qquad (2.71)$$

where $\tilde{H}_{10}$ is a general 3×3 homography matrix and $\tilde{x}_1$ and $\tilde{x}_0$ are now 2D homogeneous coordinates (i.e., 3-vectors) (Szeliski 1996).

Adapted from Li Zhang

Slide credit: Kristen Grauman

Your basic stereo algorithm



For each epipolar line

For each pixel in the left image

- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

Improvement: match ***windows***

Slide credit: Rick Szeliski

Image block matching

How do we determine correspondences?

- *block matching* or *SSD* (sum squared differences)

$$E(x,y;d) = \sum_{(x',y') \in N(x,y)} [I_L(x'+d,y') - I_R(x',y')]^2$$

d is the *disparity* (horizontal motion)

How big should the neighborhood be?

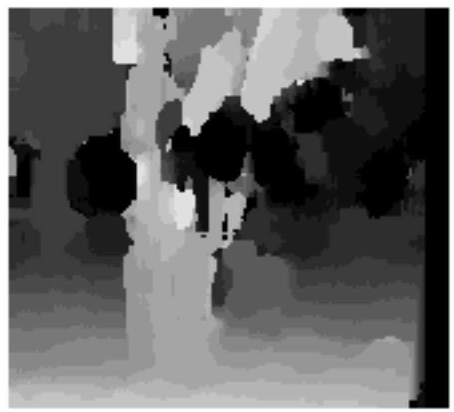
Slide credit: Rick Szeliski

Neighborhood size

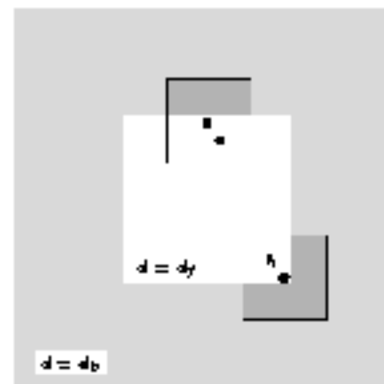
Smaller neighborhood: more details

Larger neighborhood: fewer isolated mistakes

w = 3



w = 20



Slide credit: Rick Szeliski

Matching criteria

Raw pixel values (correlation)

Band-pass filtered images [Jones & Malik 92]

“Corner” like features [Zhang, …]

Edges [many people…]

Gradients [Seitz 89; Scharstein 94]

Rank statistics [Zabih & Woodfill 94]

Slide credit: Rick Szeliski

Local evidence framework

For every disparity, compute *raw* matching costs

$$E_0(x, y; d) = \rho(I_L(x' + d, y') - I_R(x', y'))$$

Why use a robust function?

- occlusions, other outliers

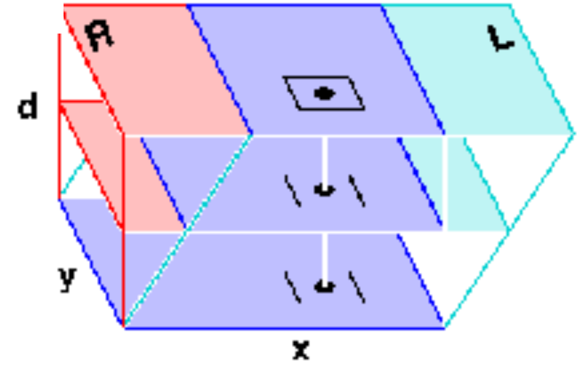
Can also use alternative match criteria

Slide credit: Rick Szeliski

Local evidence framework

Aggregate costs spatially

$$E(x, y; d) = \sum_{(x',y') \in N(x,y)} E_0(x', y', d)$$



Here, we are using a *box filter* (efficient moving average implementation)

Can also use weighted average, [non-linear] diffusion…

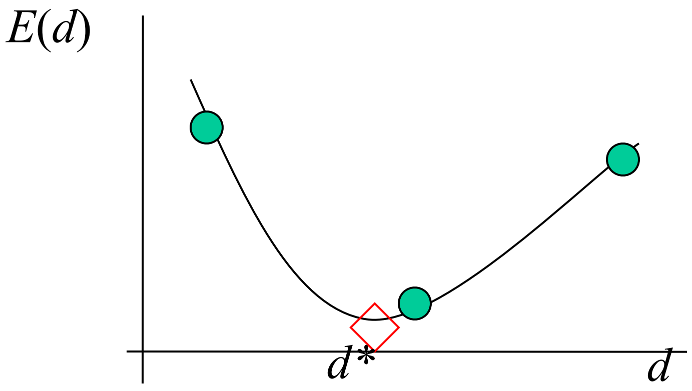
Slide credit: Rick Szeliski

Local evidence framework

Choose winning disparity at each pixel

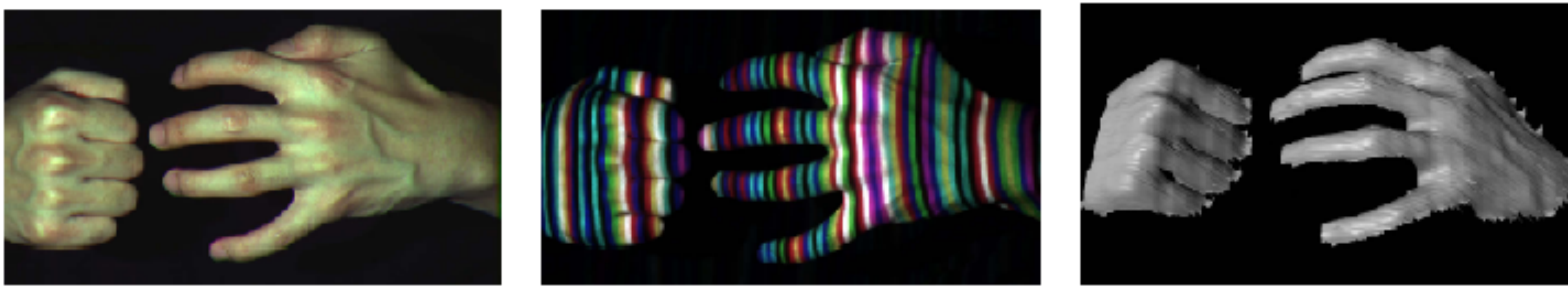
$$d(x, y) = \arg\min_{d} E(x, y; d)$$

Interpolate to *sub-pixel* accuracy

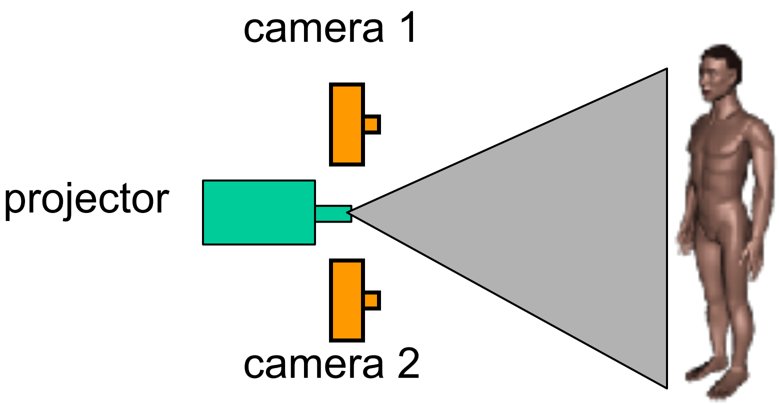


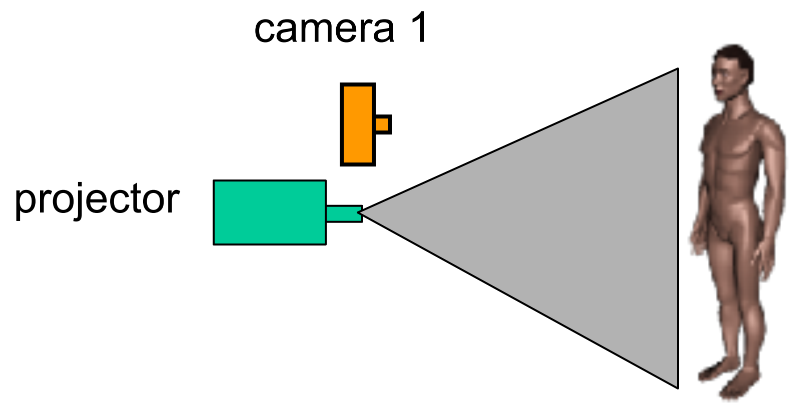
Slide credit: Rick Szeliski

Active stereo with structured light



Li Zhang's one-shot stereo





Project "structured" light patterns onto the object

- simplifies the correspondence problem

Li Zhang, Brian Curless, and Steven M. Seitz. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. In *Proceedings of the 1st International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)*, Padova, Italy, June 19-21, 2002, pp. 24-36.

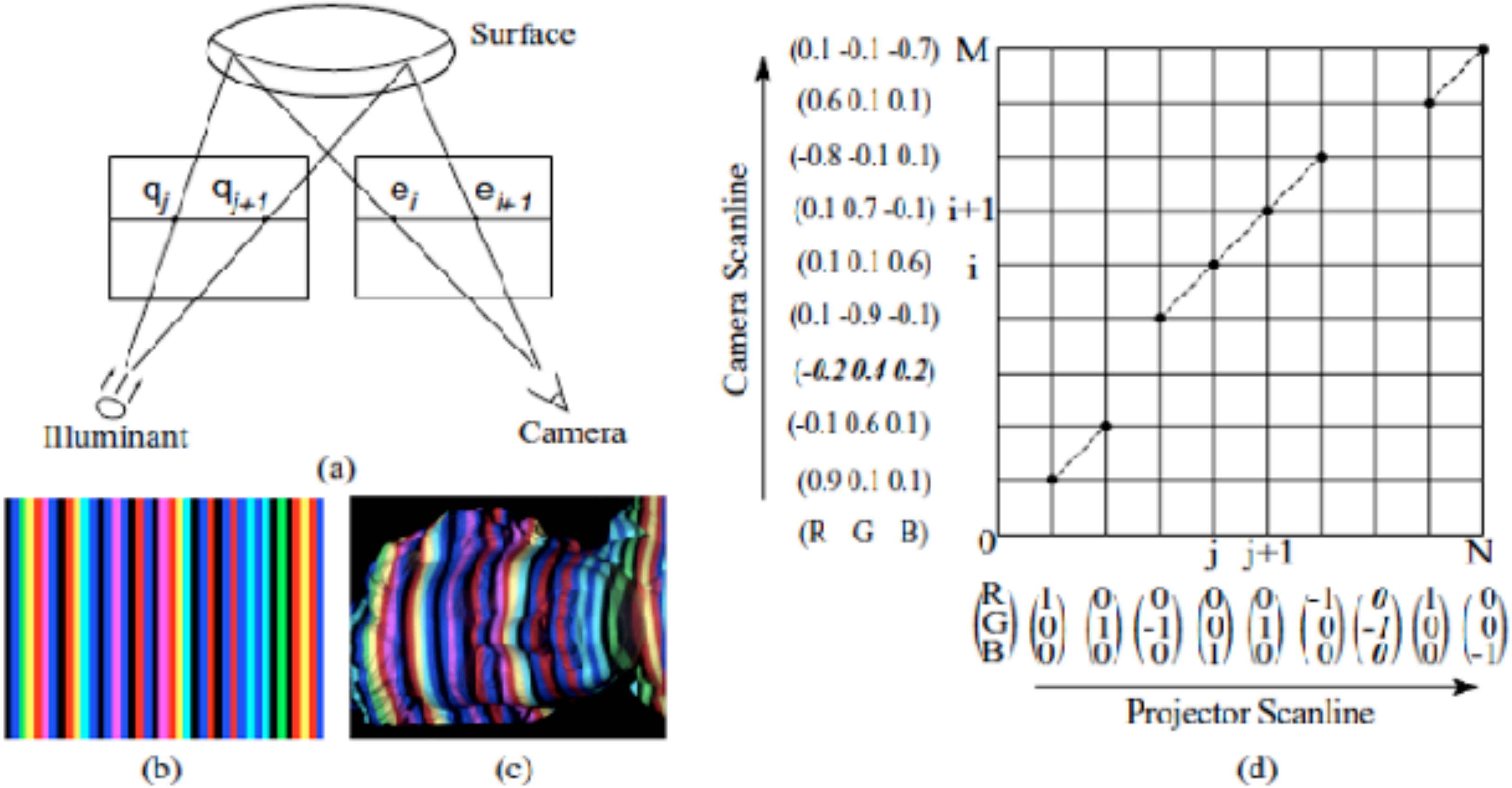
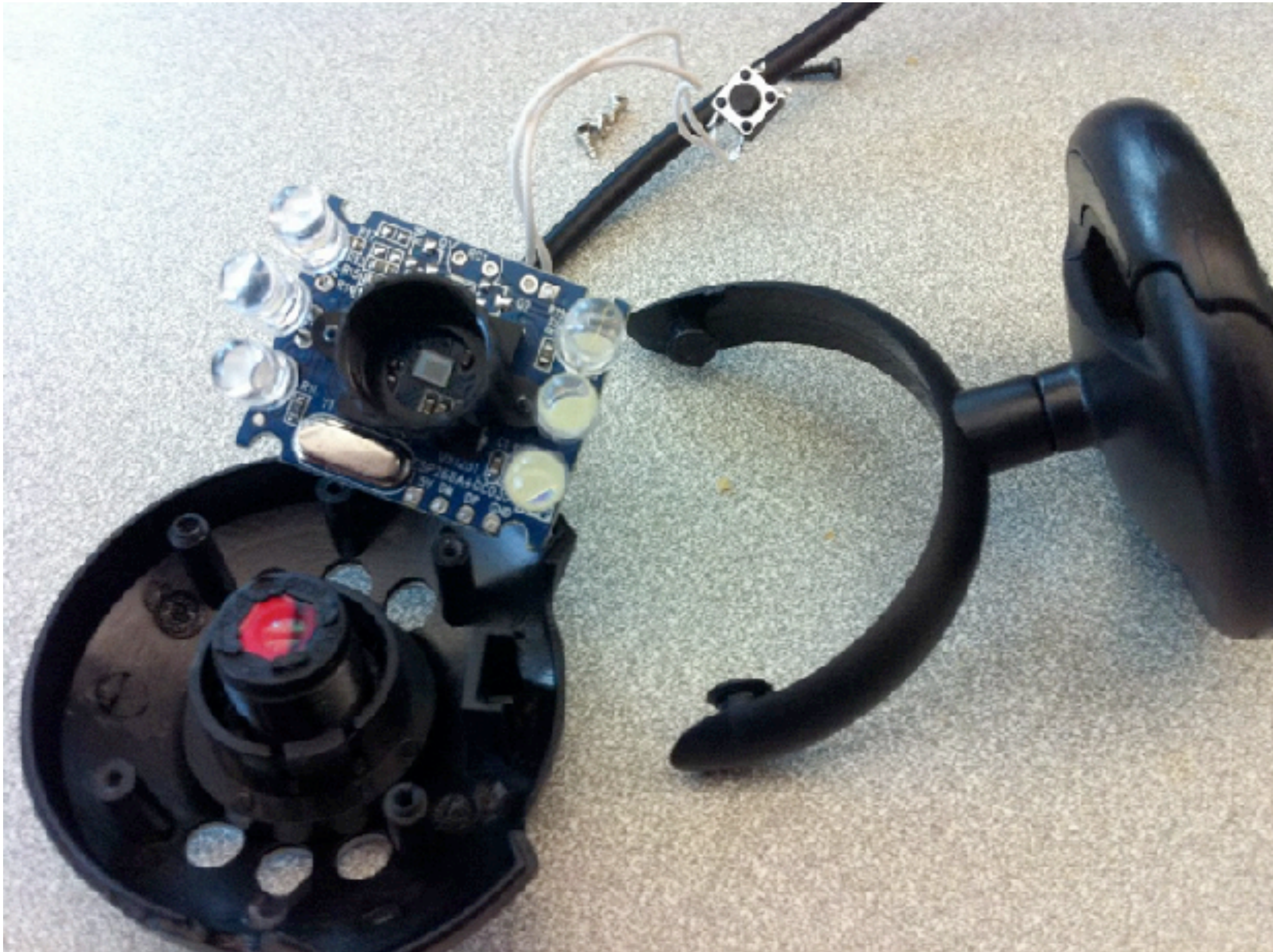
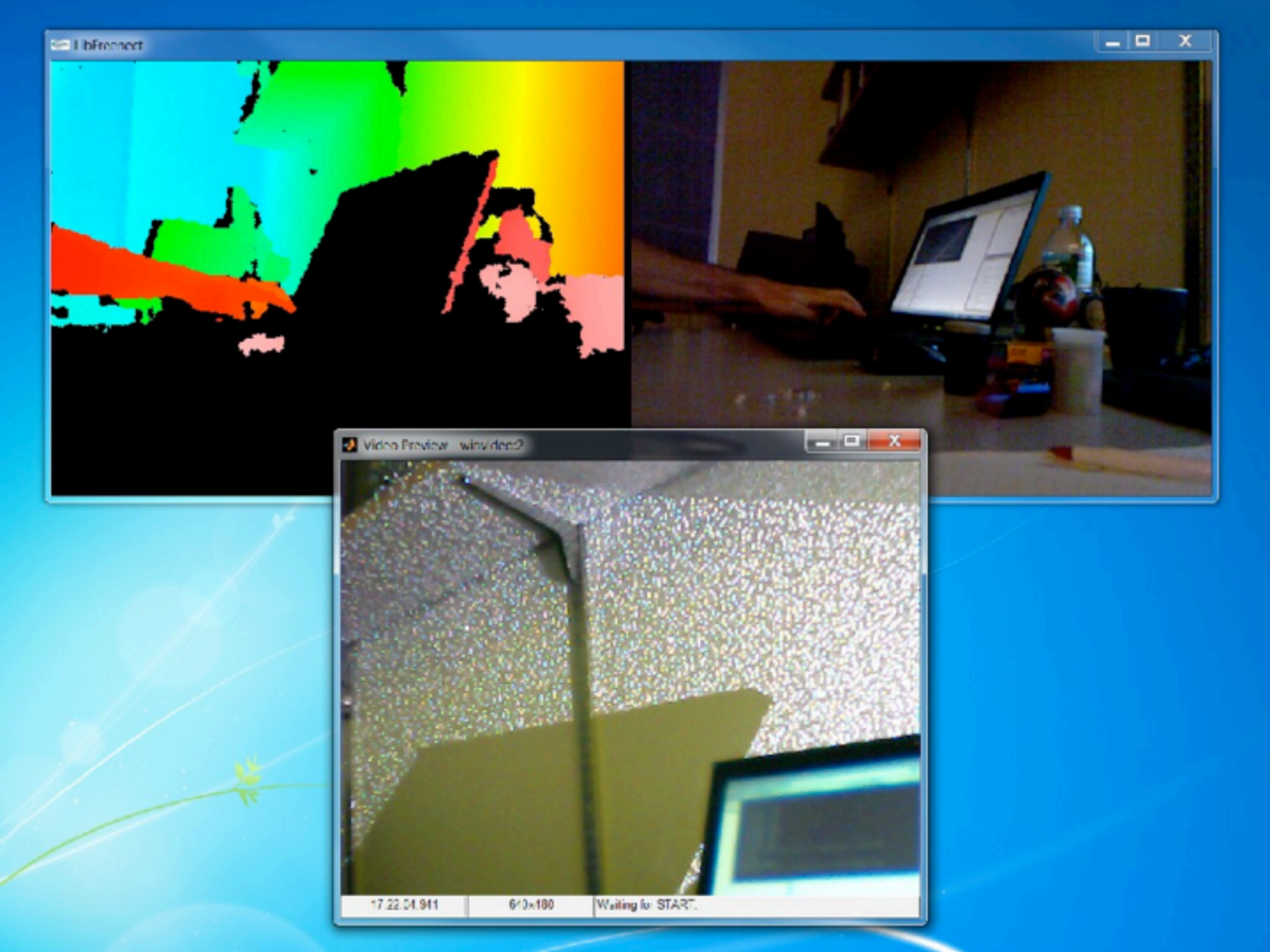


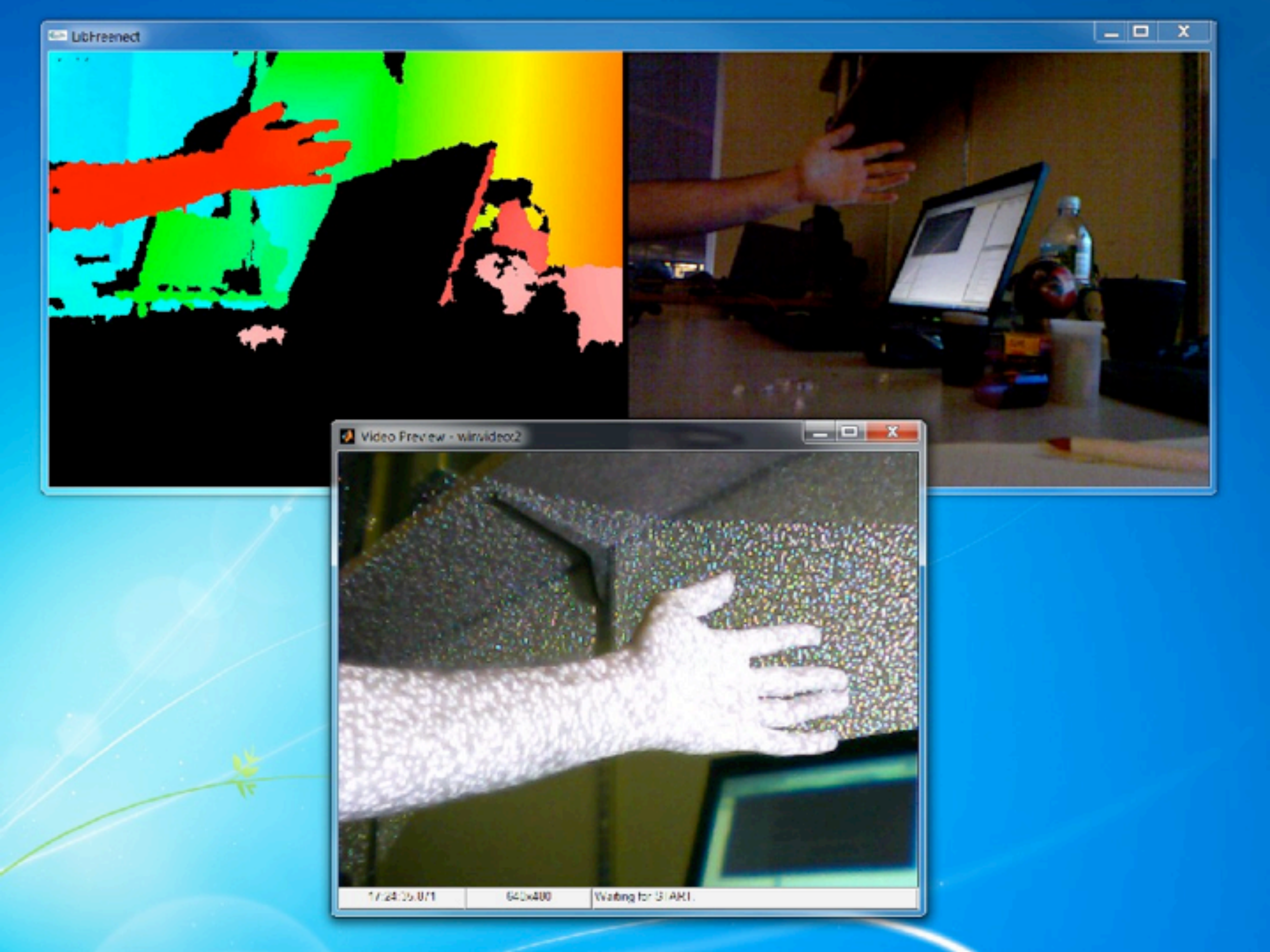
Figure 2. Summary of the one-shot method. (a) In optical triangulation, an illumination pattern is projected onto an object and the eflected light is captured by a camera. The 3D point is reconstructed from the relative displacement of a point in the pattern and mage. If the image planes are rectified as shown, the displacement is purely horizontal (one-dimensional). (b) An example of he projected stripe pattern and (c) an image captured by the camera. (d) The grid used for multi-hypothesis code matching. The orizontal axis represents the projected color transition sequence and the vertical axis represents the detected edge sequence, oth taken for one projector and rectified camera scanline pair. A match represents a path from left to right in the grid. Each ertex (j, i) has a score, measuring the consistency of the correspondence between e_i, the color gradient vectors shown by the ertical axis, and q_j, the color transition vectors shown below the horizontal axis. The score for the entire match is the summation f scores along its path. We use dynamic programming to find the optimal path. In the illustration, the camera edge in bold italics orresponds to a false detection, and the projector edge in bold italics is missed due to, e.g., occlusion.

Li Zhang, Brian Curless, and Steven M. Seitz



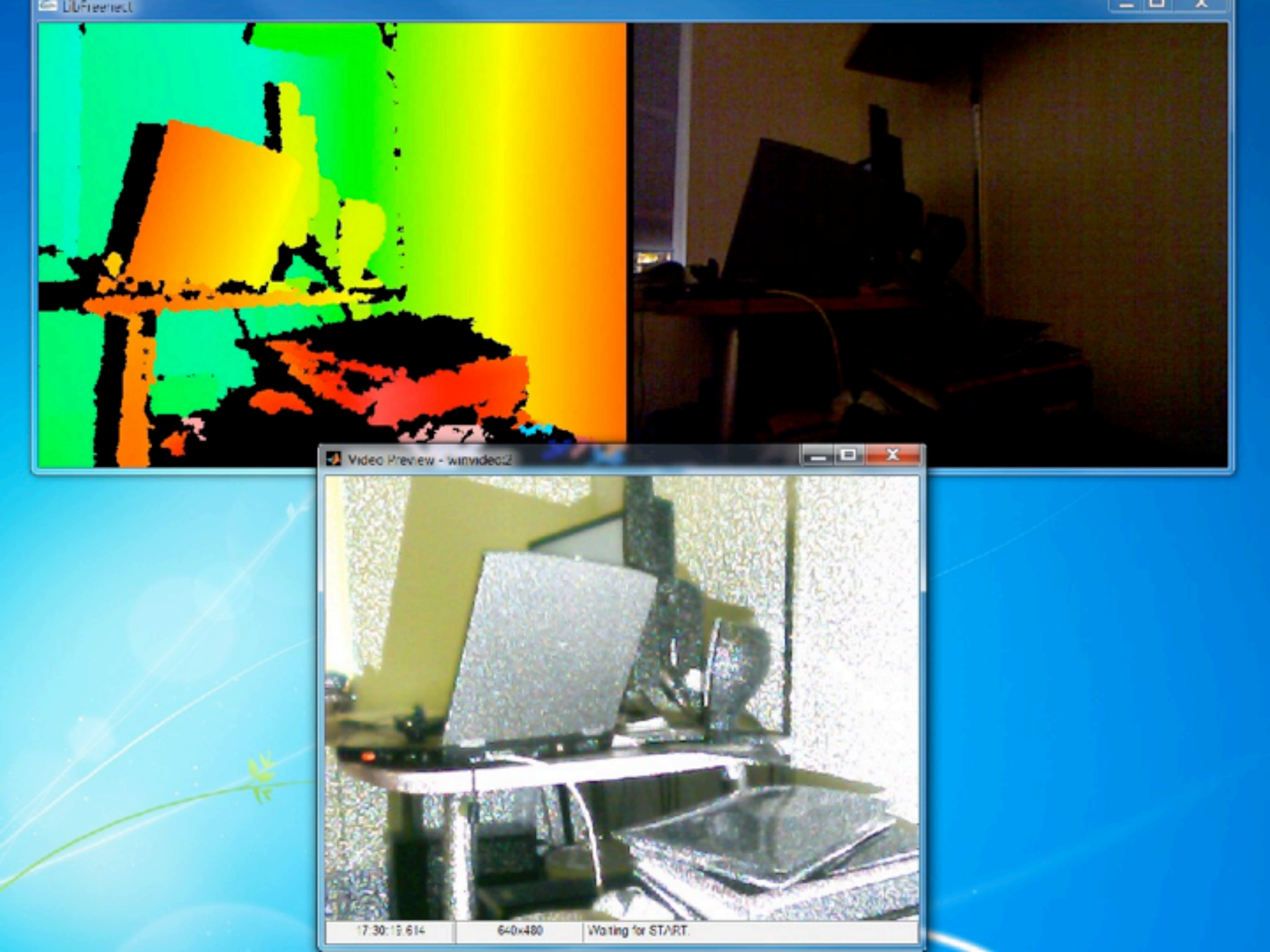

Megapixel 10X Digital Zoom f=3.85mm


Waiting for START.



LibFreenect
Video Preview - winvideo2


LibFreenect
Video Preview - winvideo2
640x480
Waiting for START


LibFreenect
Video Preview - winvideo2
17:30:19.614
640x480
Waiting for START.

Bibliography

D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision, 47(1):7-42, May 2002.

R. Szeliski. Stereo algorithms and representations for image-based rendering. In British Machine Vision Conference (BMVC'99), volume 2, pages 314-328, Nottingham, England, September 1999.

G. M. Nielson, Scattered Data Modeling, IEEE Computer Graphics and Applications, 13(1), January 1993, pp. 60-70.

S. B. Kang, R. Szeliski, and J. Chai. Handling occlusions in dense multi-view stereo. In CVPR'2001, vol. I, pages 103-110, December 2001.

Y. Boykov, O. Veksler, and Ramin Zabih, Fast Approximate Energy Minimization via Graph Cuts, Unpublished manuscript, 2000.

A.F. Bobick and S.S. Intille. Large occlusion stereo. International Journal of Computer Vision, 33(3), September 1999. pp. 181-200

D. Scharstein and R. Szeliski. Stereo matching with nonlinear diffusion. International Journal of Computer Vision, 28(2):155-174, July 1998

Slide credit: Rick Szeliski

Bibliography

Volume Intersection

- Martin & Aggarwal, “Volumetric description of objects from multiple views”, Trans. Pattern Analysis and Machine Intelligence, 5(2), 1991, pp. 150-158.
- Szeliski, “Rapid Octree Construction from Image Sequences”, Computer Vision, Graphics, and Image Processing: Image Understanding, 58(1), 1993, pp. 23-32.

Voxel Coloring and Space Carving

- Seitz & Dyer, “Photorealistic Scene Reconstruction by Voxel Coloring”, Proc. Computer Vision and Pattern Recognition (CVPR), 1997, pp. 1067-1073.
- Seitz & Kutulakos, “Plenoptic Image Editing”, Proc. Int. Conf. on Computer Vision (ICCV), 1998, pp. 17-24.
- Kutulakos & Seitz, “A Theory of Shape by Space Carving”, Proc. ICCV, 1998, pp. 307-314.

Slide credit: Rick Szeliski

Bibliography

Related References

- Bolles, Baker, and Marimont, “Epipolar-Plane Image Analysis: An Approach to Determining Structure from Motion”, International Journal of Computer Vision, vol 1, no 1, 1987, pp. 7-55.
- Faugeras & Keriven, “Variational principles, surface evolution, PDE's, level set methods and the stereo problem", IEEE Trans. on Image Processing, 7(3), 1998, pp. 336-344.
- Szeliski & Golland, “Stereo Matching with Transparency and Matting”, Proc. Int. Conf. on Computer Vision (ICCV), 1998, 517-524.
- Roy & Cox, “A Maximum-Flow Formulation of the N-camera Stereo Correspondence Problem”, Proc. ICCV, 1998, pp. 492-499.
- Fua & Leclerc, “Object-centered surface reconstruction: Combining multi-image stereo and shading", International Journal of Computer Vision, 16, 1995, pp. 35-56.
- Narayanan, Rander, & Kanade, “Constructing Virtual Worlds Using Dense Stereo”, Proc. ICCV, 1998, pp. 3-10.

Slide credit: Rick Szeliski