Projective Geometry and Camera Models

Computer Vision

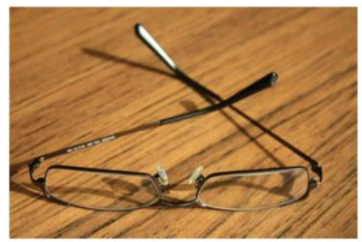
CS 543 / ECE 549

University of Illinois

Derek Hoiem

HWs

- HW 1 back today
  - Solutions are posted
  - Frequent mistake on question about shadow on specular surface

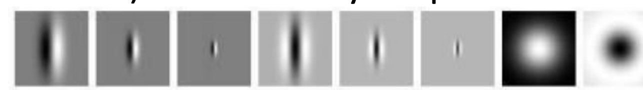


- HW 2 due next Tues

- HW 3 should be out by end of week

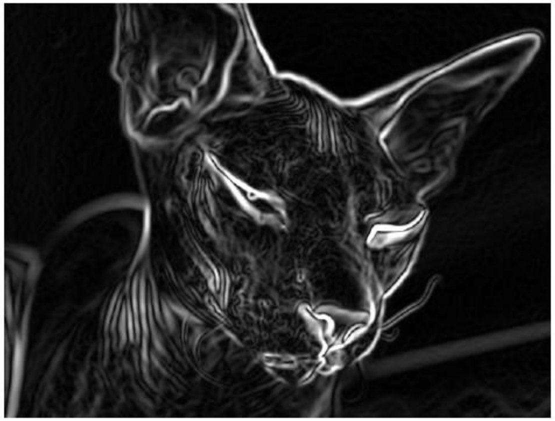
Top edge methods

- Huy Le (0.673):
  - Find edge magnitudes using sobel
  - Suppress using canny on R, G, B (OR edge maps) max pooling
  - Get max filter response of applying RFS filters (from Oxford) to boundary map



Top edge methods

- Austin Walters (0.673):
  - convert to LAB
  - compute gradient magnitudes
  - take max over channels



- Yang Xu (0.646)
  - oriented filter within RGB+HSV
  - group edges using connected components
  - threshold based on edge length/intensity

Think about your final projects

- Strongly encouraged to work in groups of 2-4 (but if you have a good reason to work by self, could be ok)
- Projects don’t need to be of publishable originality but should evince independent effort to learn about a new topic, try something new, or apply to an application of interest
- Proposals will be due after Spring Break

Last notes on registration

- Thin-plate splines: combines global affine warp with smooth local deformation

$$E_{TPS}(f) = \sum_{a=1}^{K} ||y_a - f(v_a)||^2 + \lambda \int\int \left[(\frac{\partial^2 f}{\partial x^2})^2 + 2(\frac{\partial^2 f}{\partial x \partial y})^2 + (\frac{\partial^2 f}{\partial y^2})^2 \right] dxdy$$

Diff of predicted vs. actual position

Smoothness cost for local warps

There is a closed form solution for parameter estimation and warping

$$f(v_a, d, w) = v_a \cdot d + \phi(v_a) \cdot w$$

Affine warp

Local deformation according to distance from control points

- Robust non-rigid point matching: http://noodle.med.yale.edu/~chui/tps-rpm.html (includes code, demo, paper)
  - Thin-plate spline registration with robustness to outliers

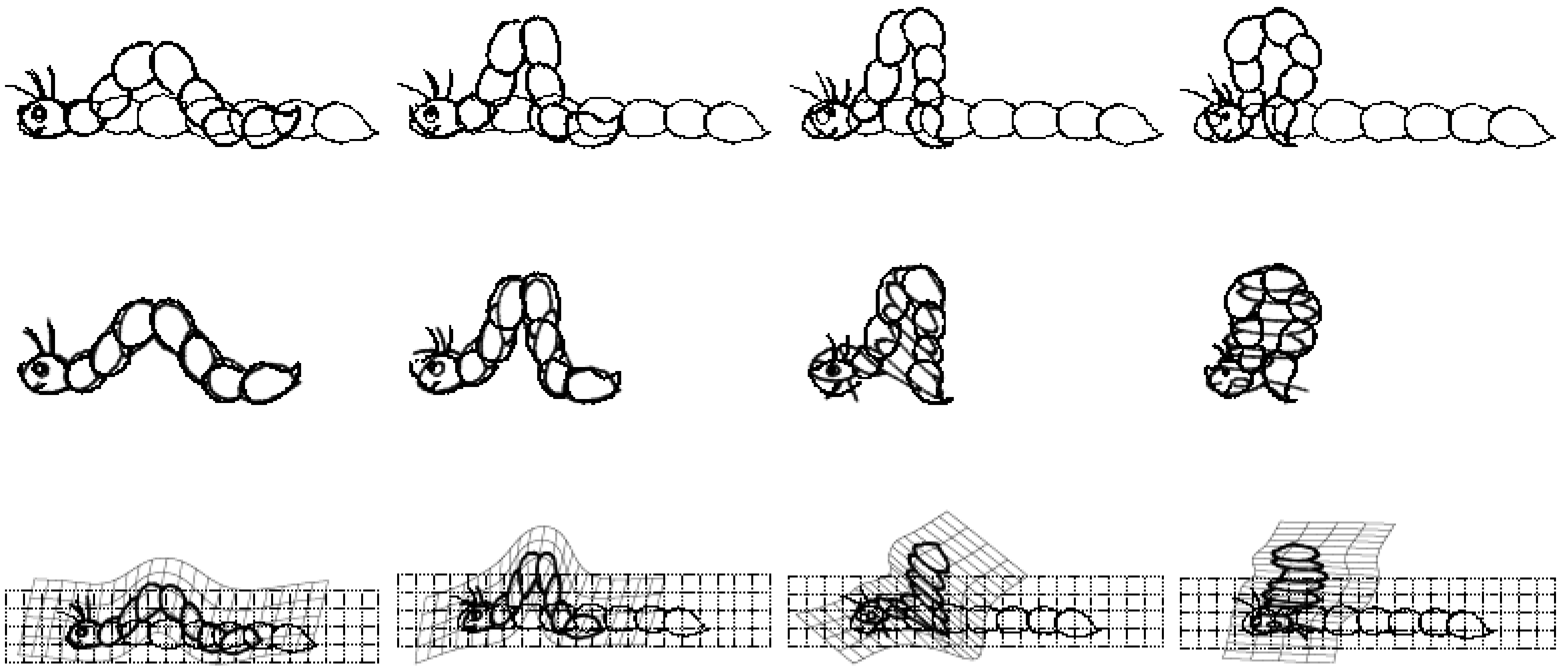
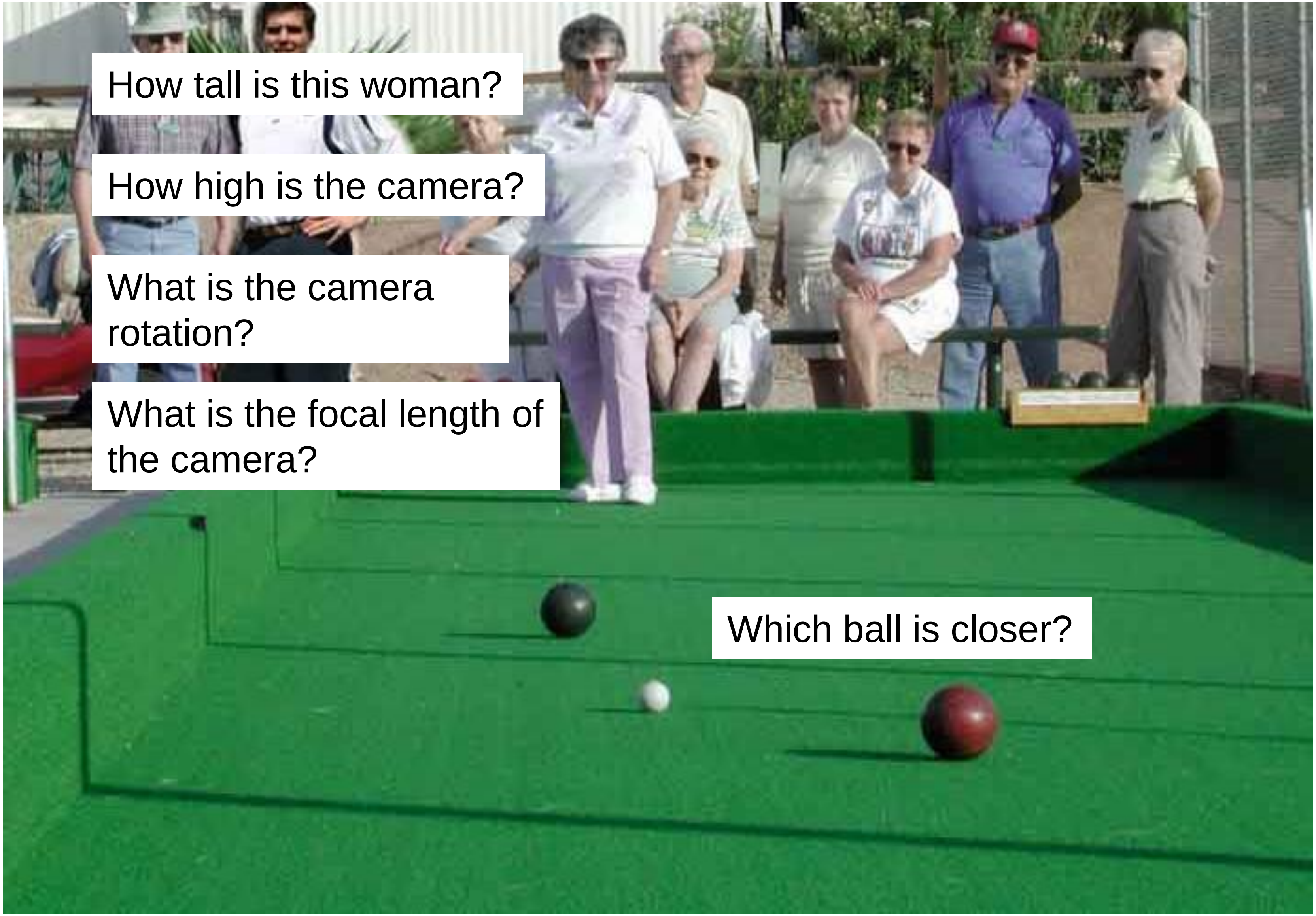


Fig. 12. **Large Deformation—Caterpillar Example.** From left to right, matching frame 1 to frame 5, 7, 11 and 12. **Top:** Original location. **Middle:** matched result. **Bottom:** deformation found.

Next two classes: Single-view Geometry

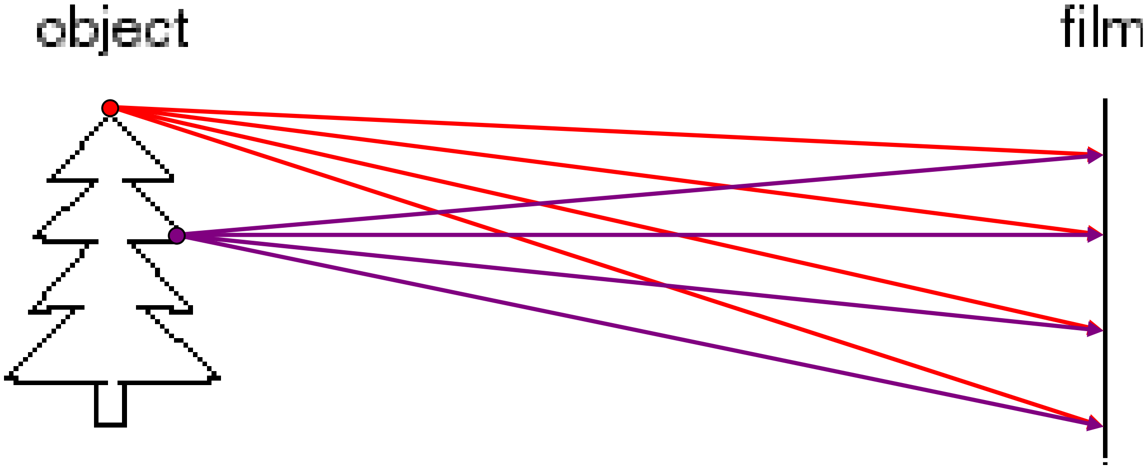


Today's class

Mapping between image and world coordinates

- Pinhole camera model
- Projective geometry
  - Vanishing points and lines
- Projection matrix

Image formation

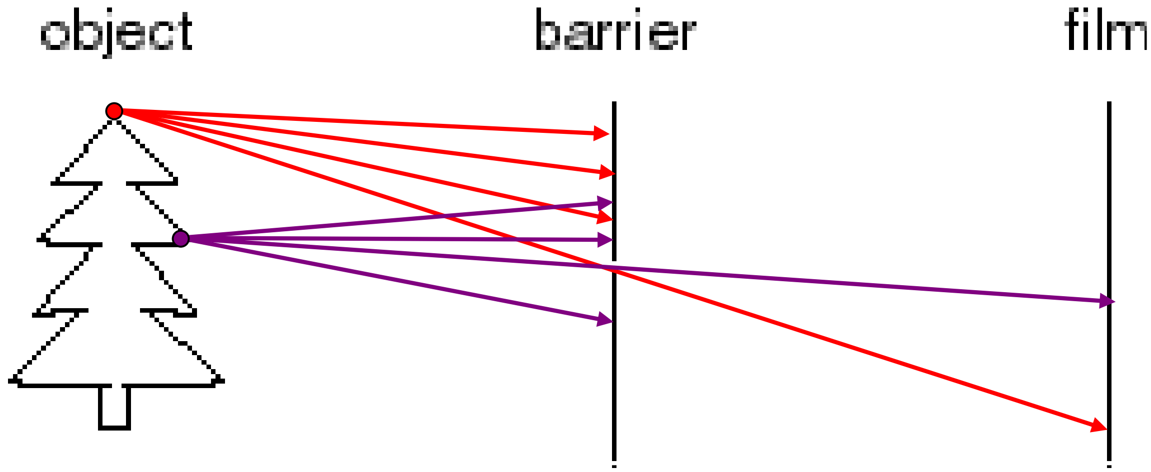


Let's design a camera

- Idea 1: put a piece of film in front of an object
- Do we get a reasonable image?

Slide source: Seitz

Pinhole camera

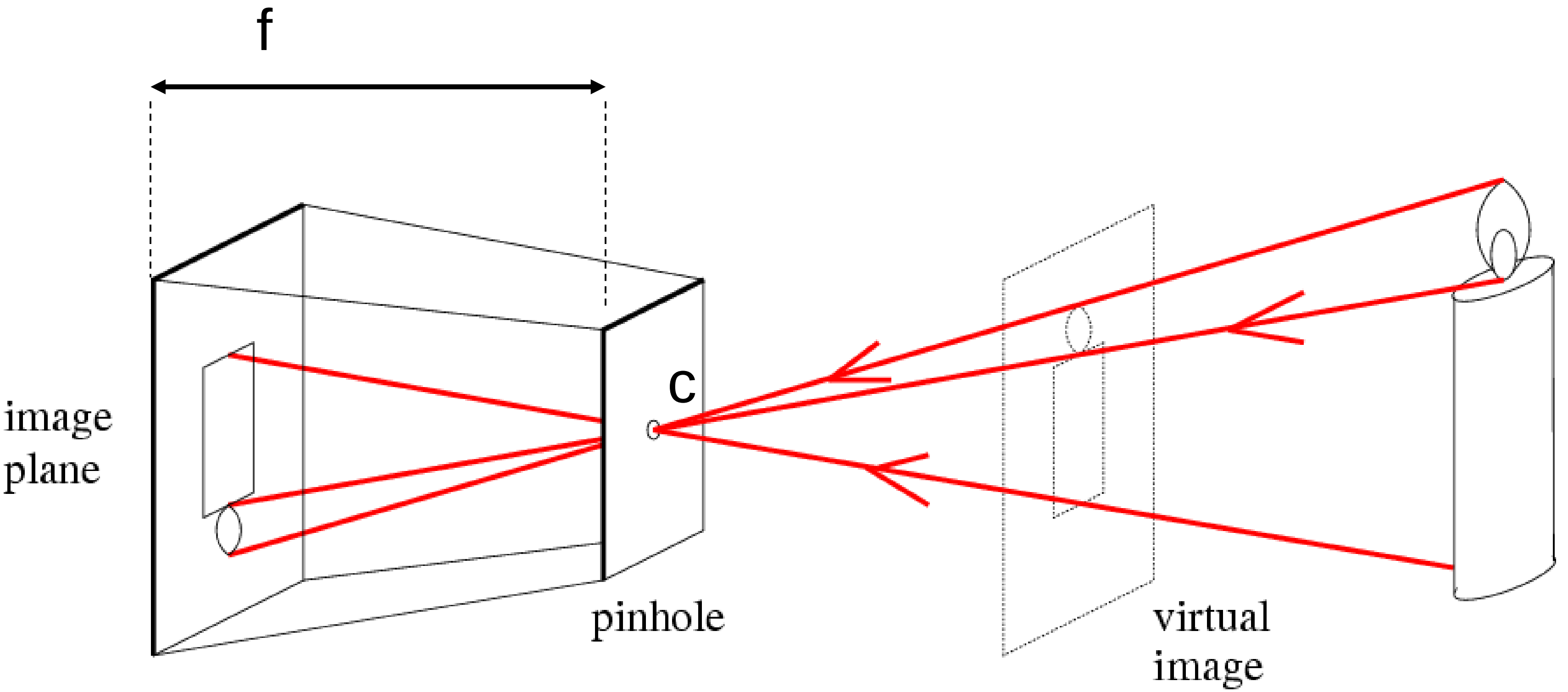


Idea 2: add a barrier to block off most of the rays

- This reduces blurring
- The opening known as the **aperture**

Slide source: Seitz

Pinhole camera



f = focal length
c = center of the camera

Figure from Forsyth

Camera obscura: the pre-camera

- First idea: Mo-Ti, China (470BC to 390BC)

- First built: Alhazen, Iraq/Egypt (965 to 1039AD)

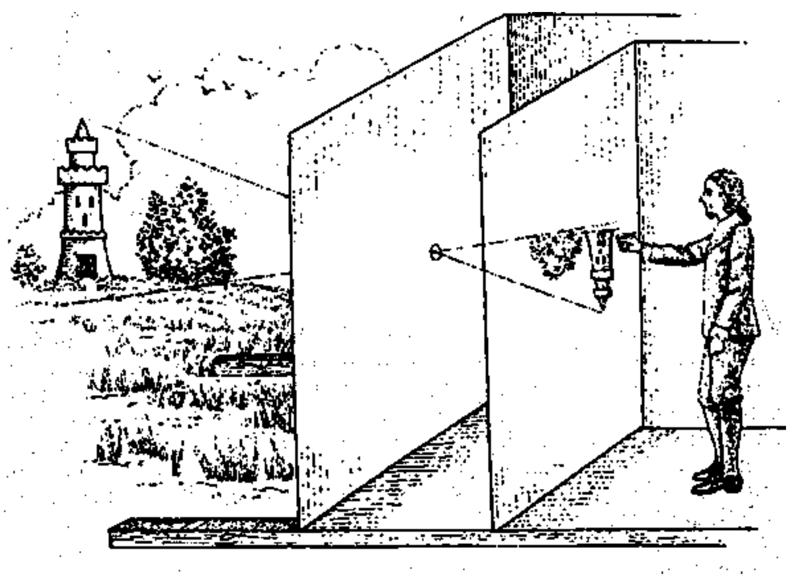
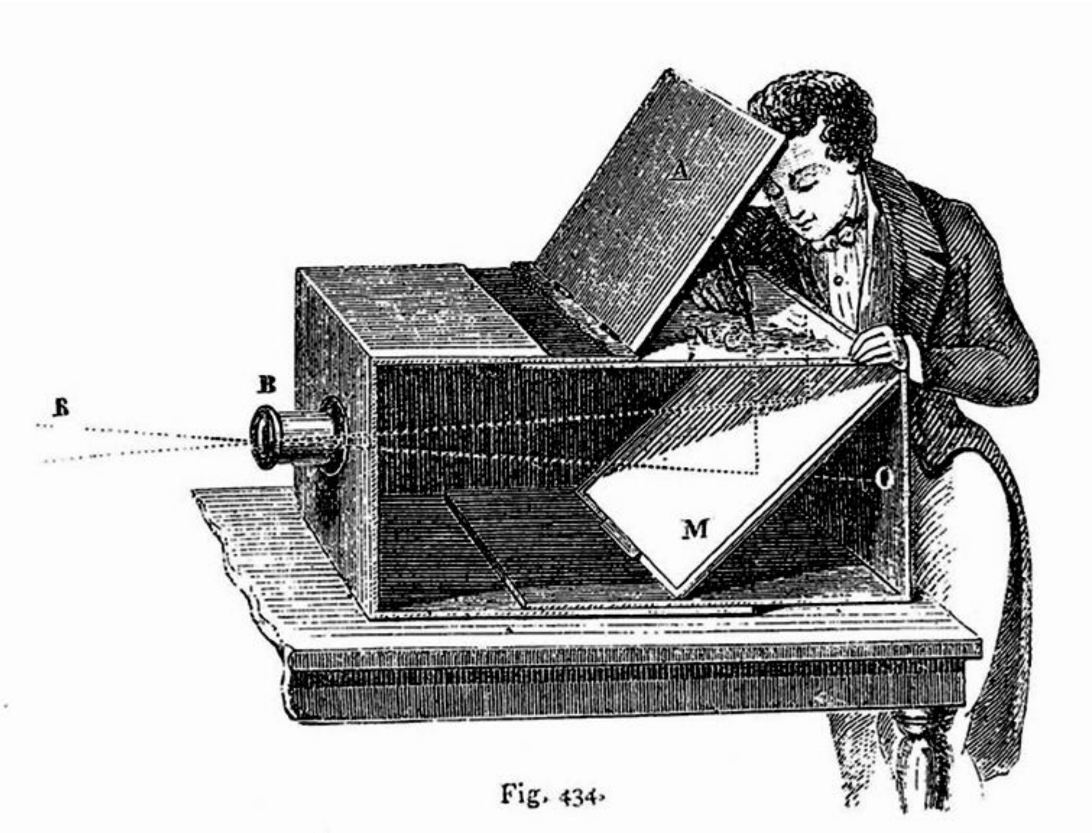


Illustration of Camera Obscura

Freestanding camera obscura at UNC Chapel Hill

Photo by Seth Ilys

Camera Obscura used for Tracing



Lens Based Camera Obscura, 1568

First Photograph

Oldest surviving photograph

- Took 8 hours on pewter plate

Joseph Niepce, 1826

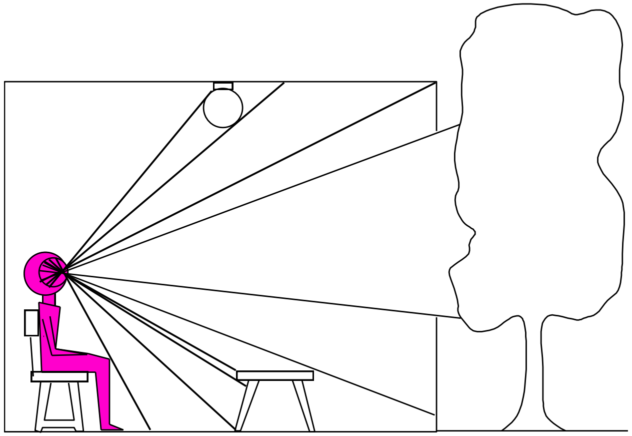
Photograph of the first photograph

Stored at UT Austin

Niepce later teamed up with Daguerre, who eventually created Daguerrotypes

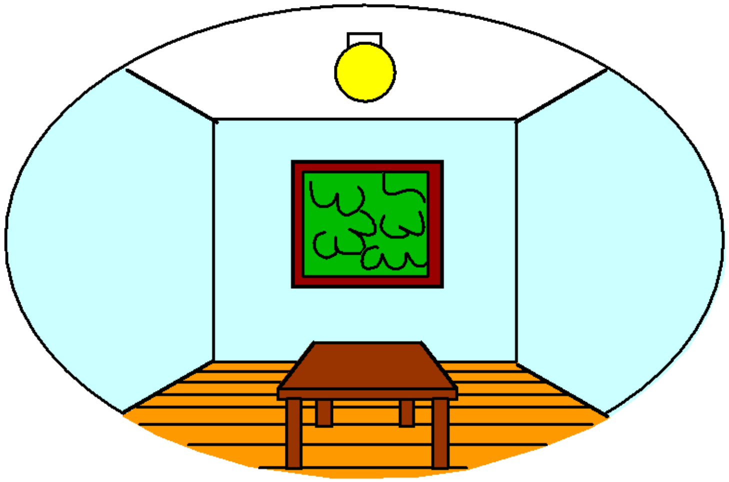
Dimensionality Reduction Machine (3D to 2D)

3D world



Point of observation

2D image



Slide source: Seitz

Projection can be tricky…

Slide source: Seitz

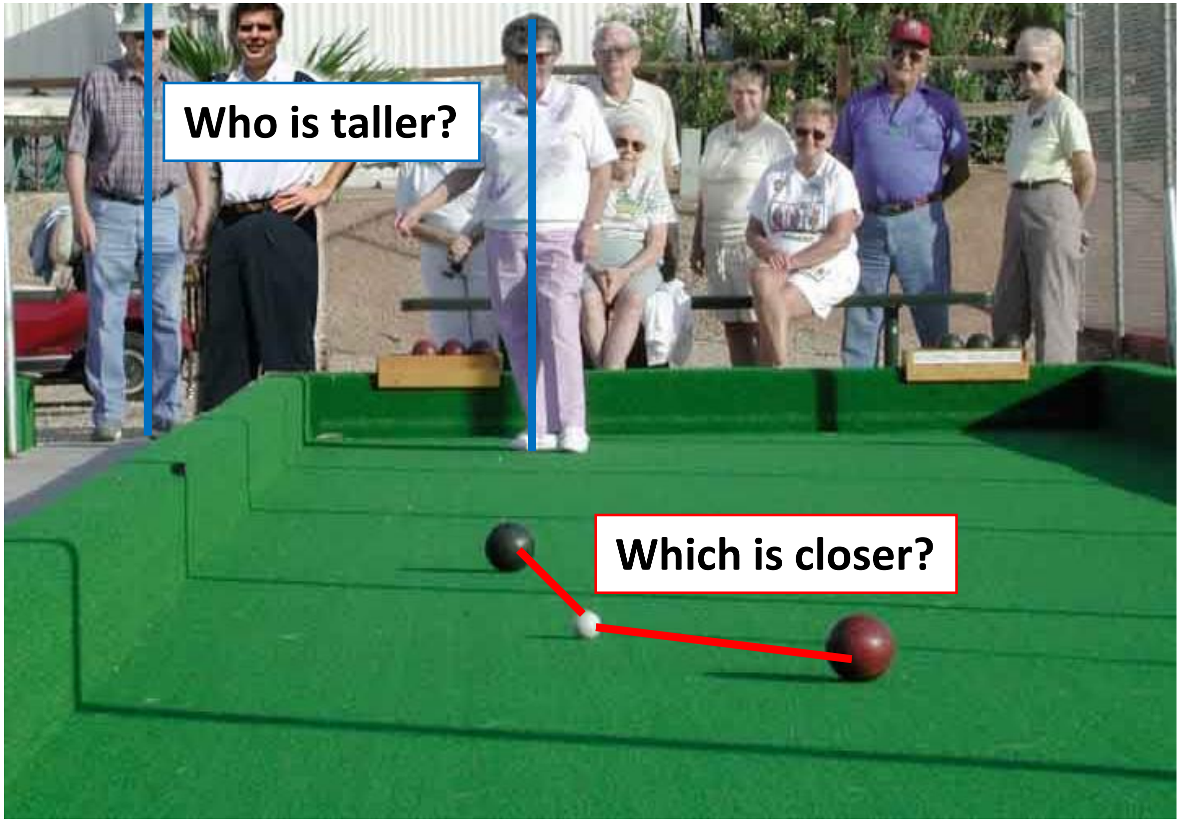
Projection can be tricky…

Making of 3D sidewalk art: http://www.youtube.com/watch?v=3SNYtd0Ayt0

Projective Geometry

What is lost?

- Length



Length is not preserved

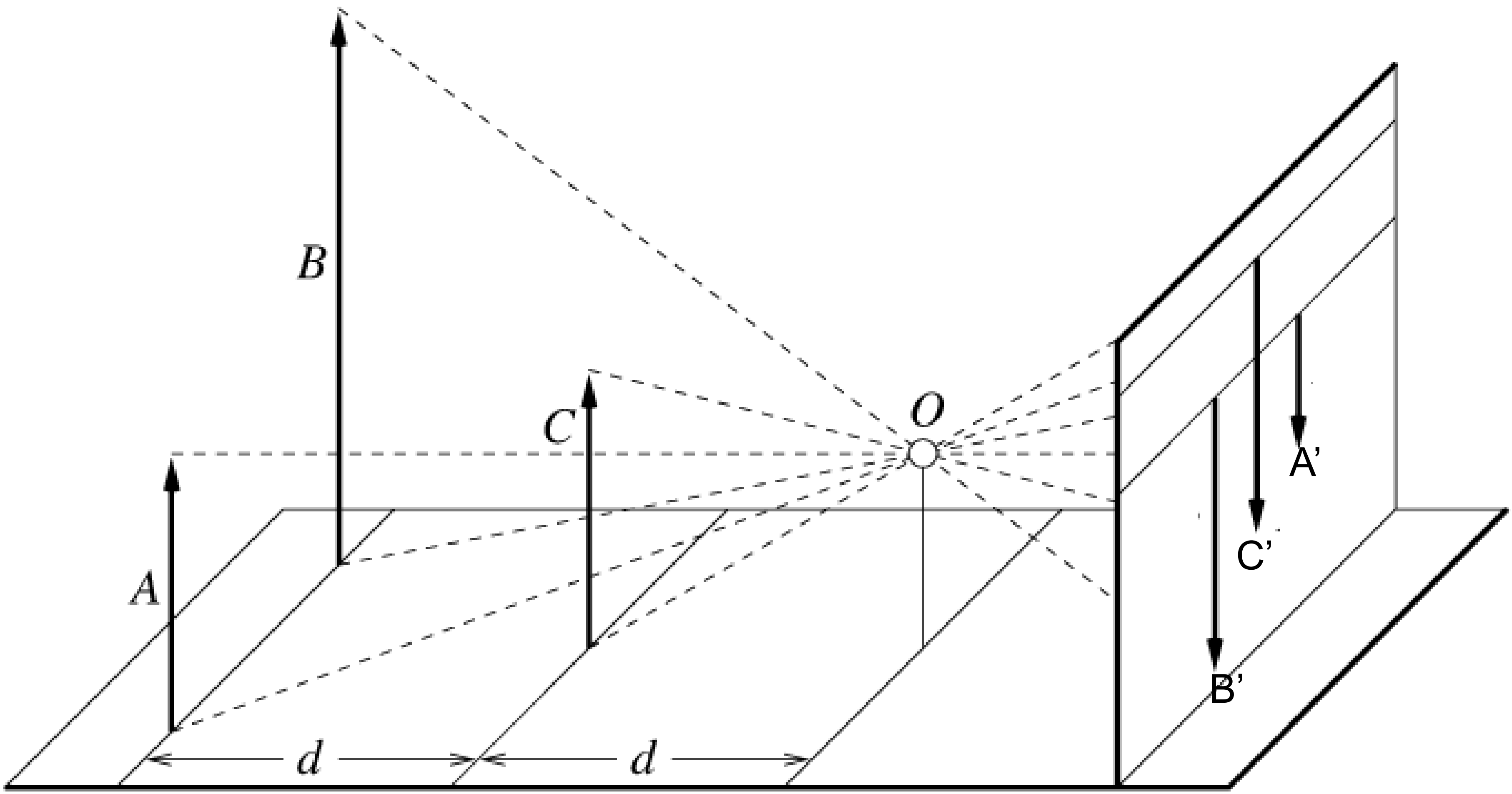
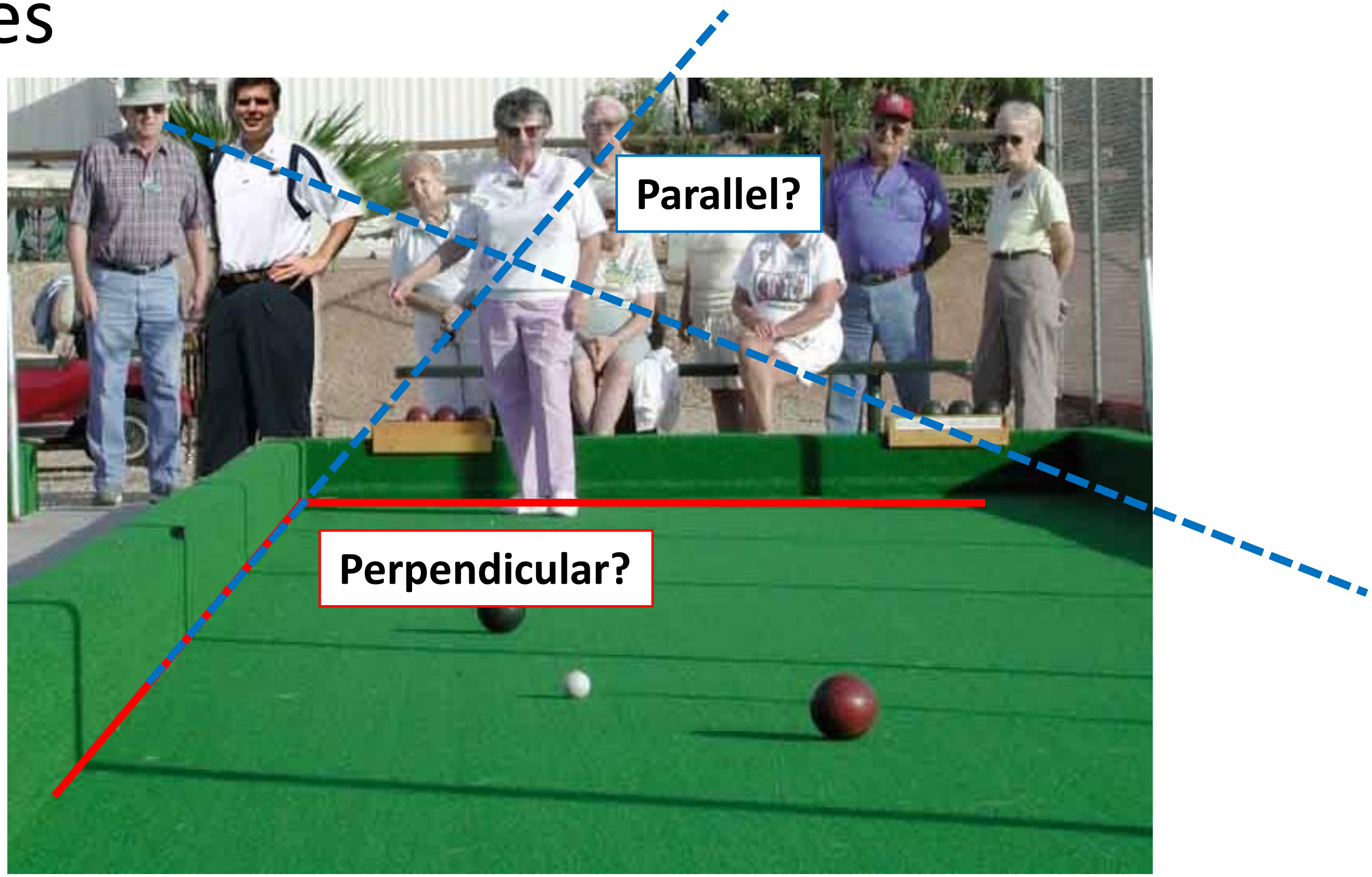


Figure by David Forsyth

Projective Geometry

What is lost?

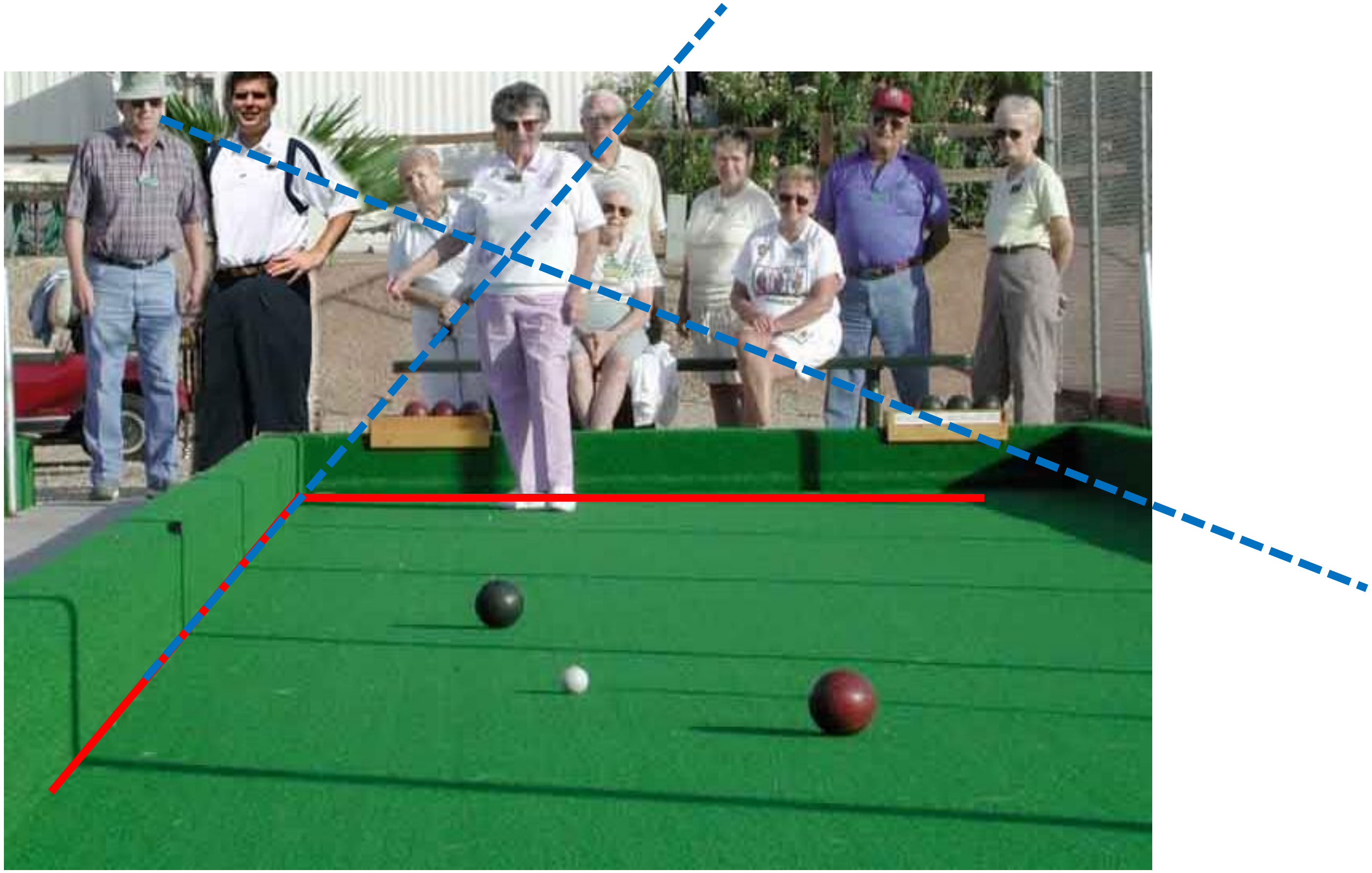
- Length
- Angles



Projective Geometry

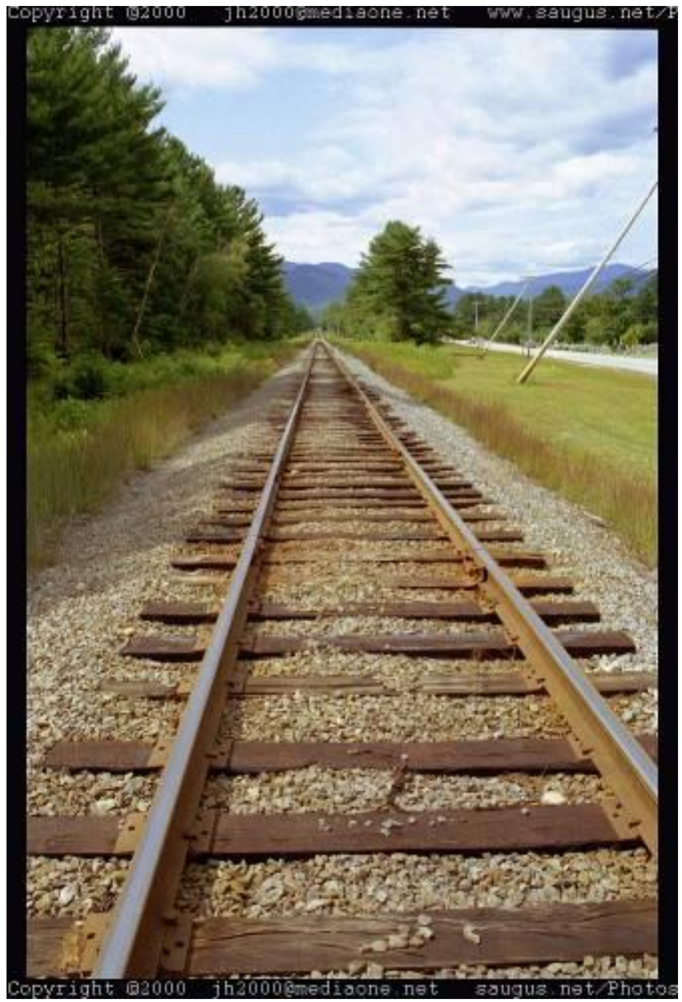
What is preserved?

- Straight lines are still straight

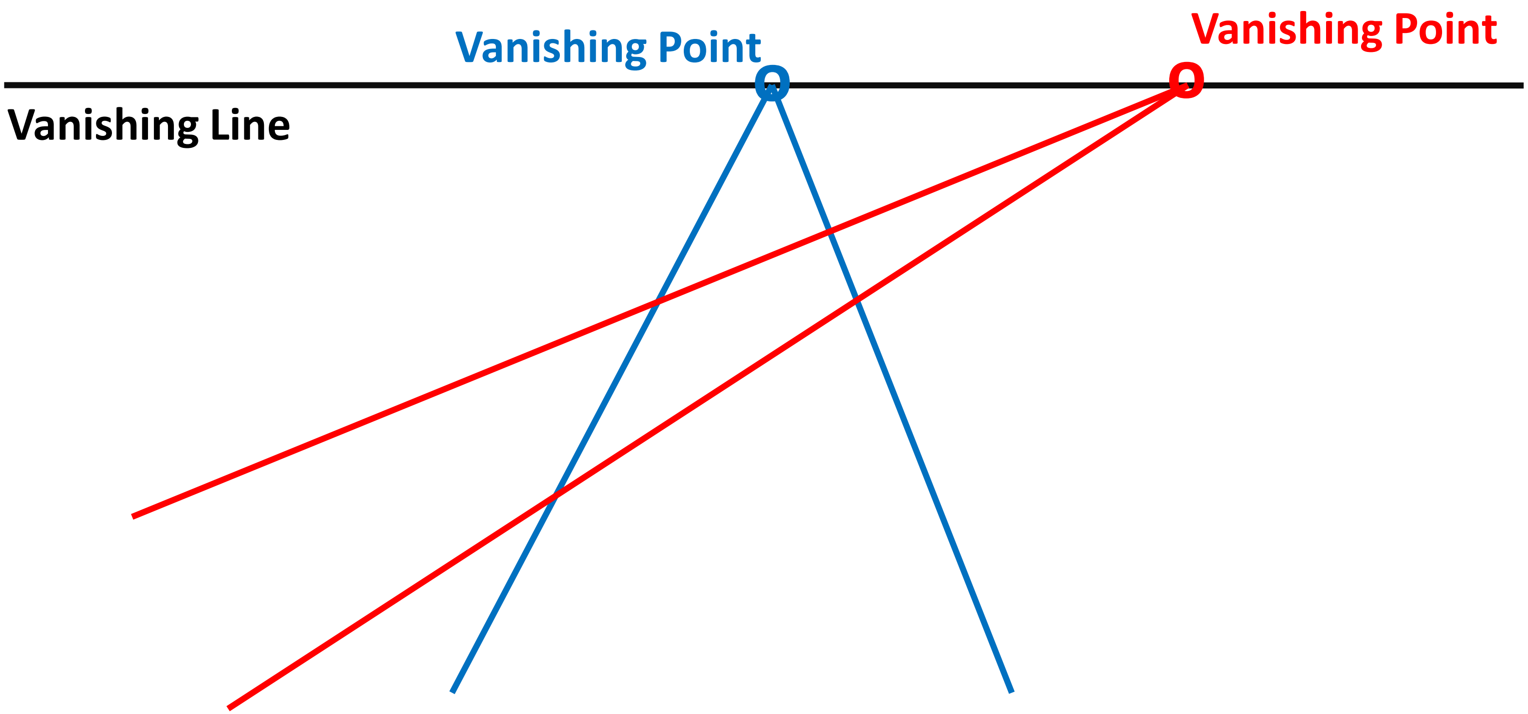


Vanishing points and lines

Parallel lines in the world intersect in the image at a "vanishing point"

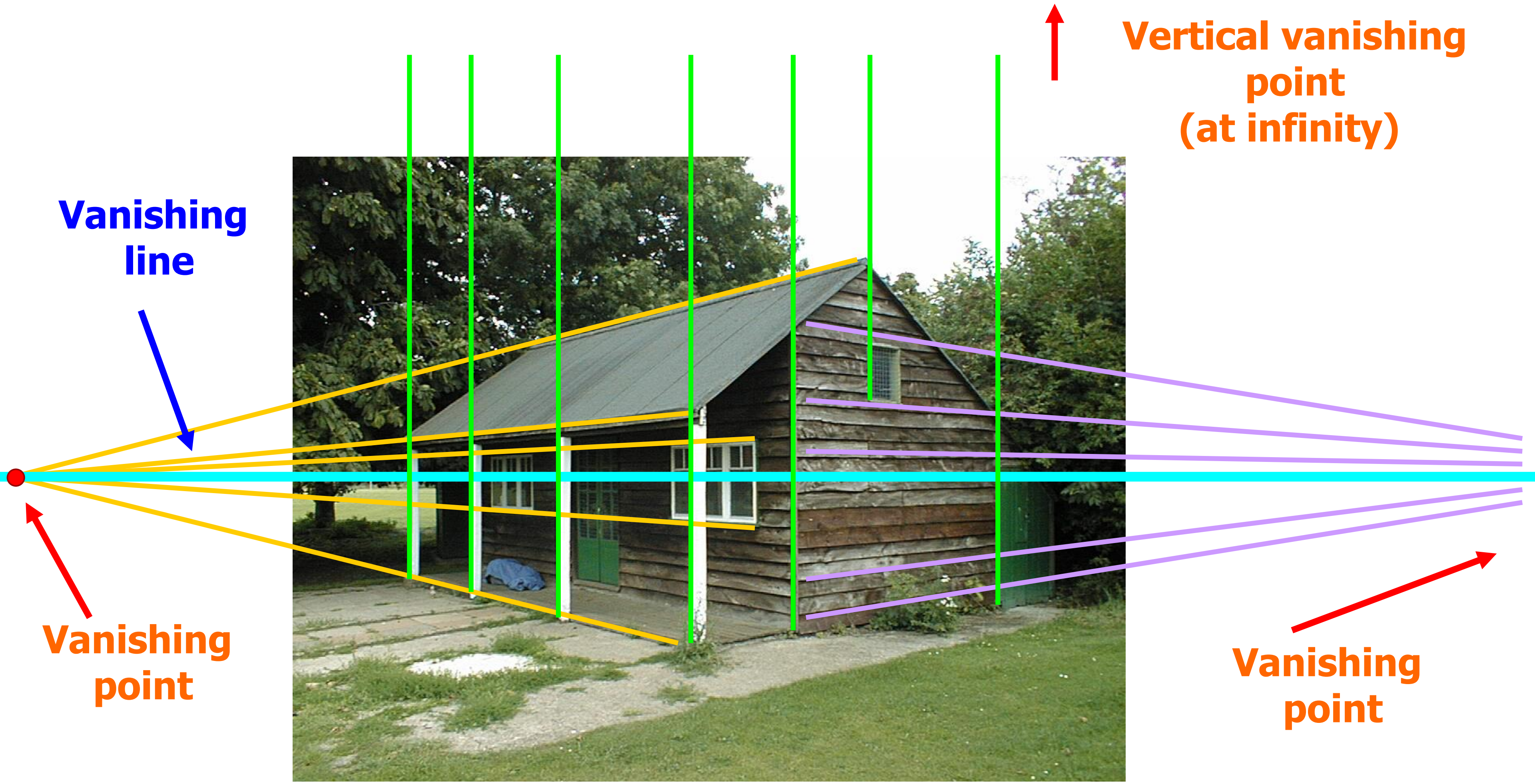


Vanishing points and lines



- The projections of parallel 3D lines intersect at a **vanishing point**
- The projection of parallel 3D planes intersect at a **vanishing line**
- If a set of parallel 3D lines are also parallel to a particular plane, their vanishing point will lie on the vanishing line of the plane
- Not all lines that intersect are parallel
- Vanishing point <-> 3D direction of a line
- Vanishing line <-> 3D orientation of a surface

Vanishing points and lines

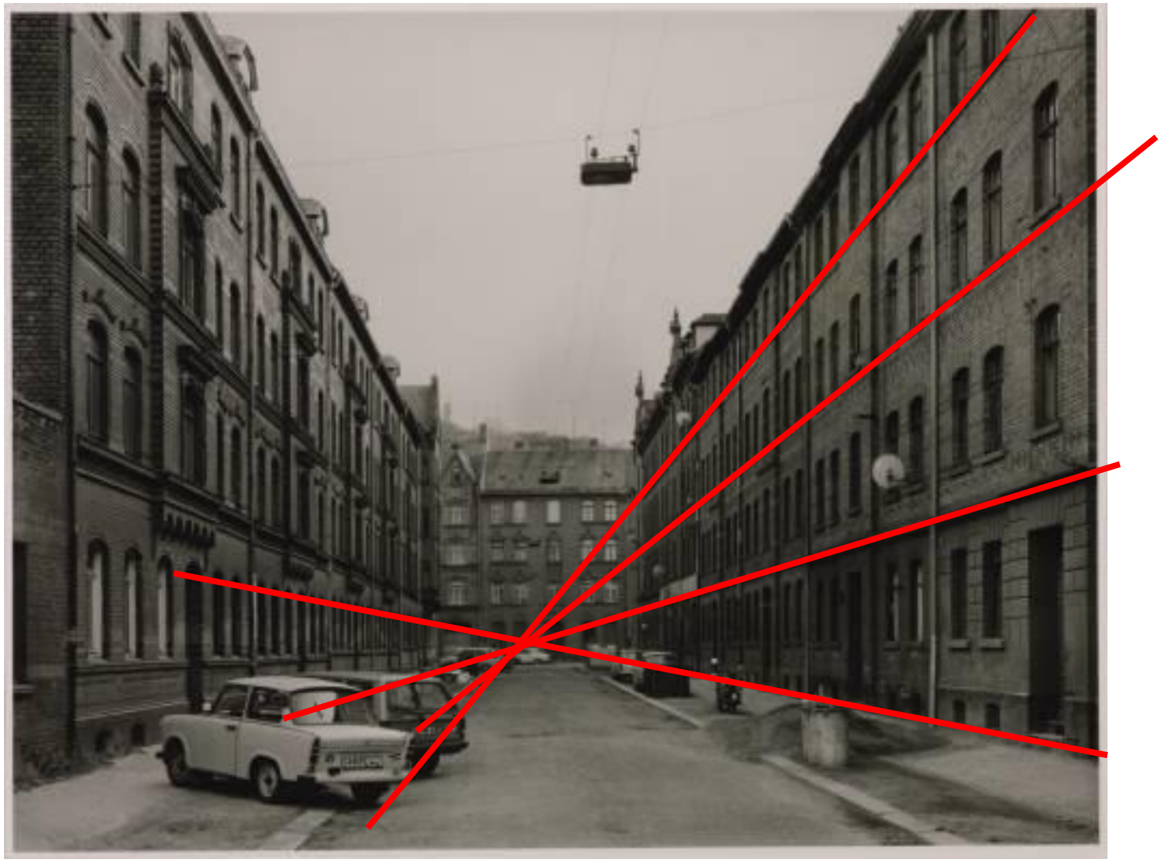


Slide from Efros, Photo from Criminisi

Vanishing points and lines

Photo from online Tate collection

Note on estimating vanishing points



Use multiple lines for better accuracy

… but lines will not intersect at exactly the same point in practice

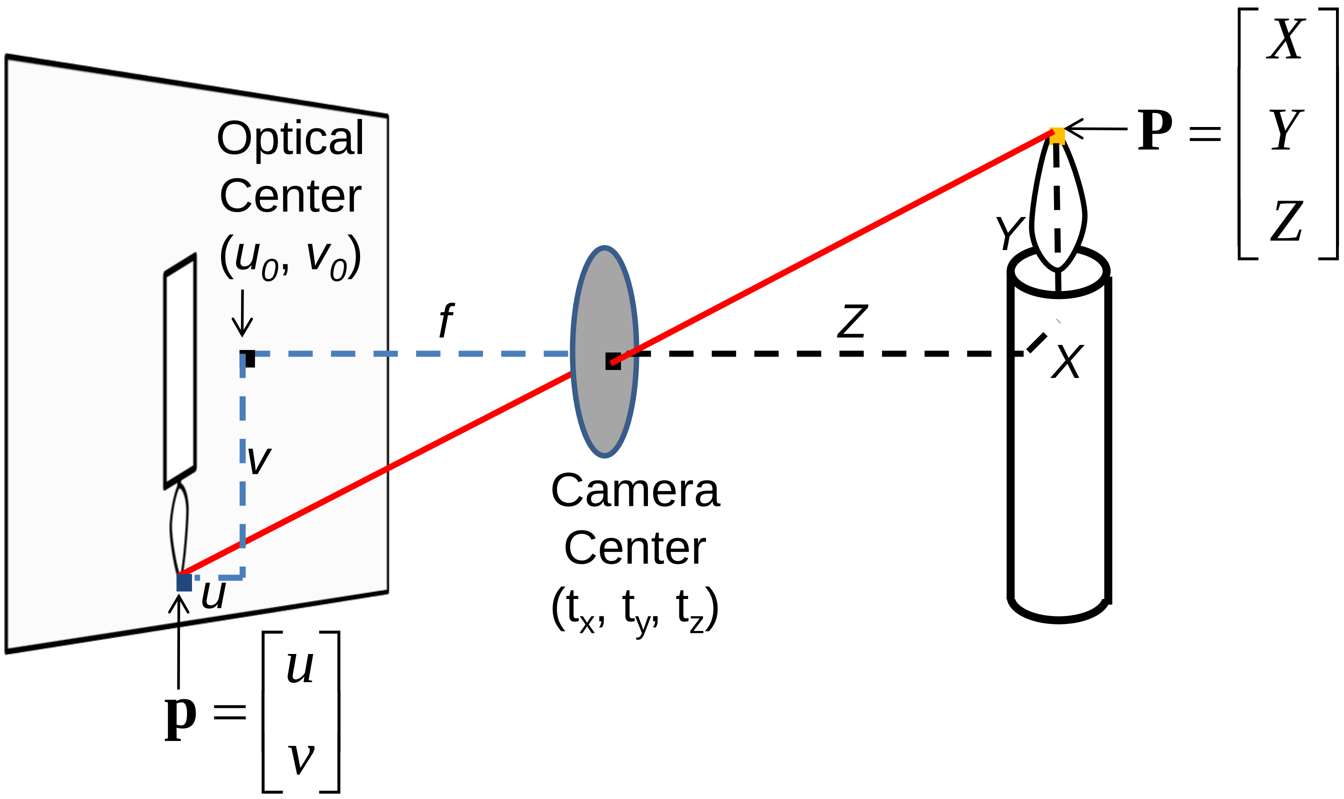
One solution: take mean of intersecting pairs

… bad idea!

Instead, minimize angular differences

Vanishing objects

Projection: world coordinates→image coordinates



Homogeneous coordinates

Conversion

Converting to *homogeneous* coordinates

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

homogeneous image coordinates

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

homogeneous scene coordinates

Converting *from* homogeneous coordinates

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

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

Homogeneous coordinates

Invariant to scaling

$$k\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} kx \\ ky \\ kw \end{bmatrix} \Rightarrow \begin{bmatrix} \frac{kx}{kw} \\ \frac{ky}{kw} \end{bmatrix} = \begin{bmatrix} \frac{x}{w} \\ \frac{y}{w} \end{bmatrix}$$

Homogeneous Coordinates

Cartesian Coordinates

Point in Cartesian is ray in Homogeneous

Basic geometry in homogeneous coordinates

- Line equation: ax + by + c = 0

$$line_i = \begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix}$$

- Append 1 to pixel coordinate to get homogeneous coordinate

$$p_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}$$

- Line given by cross product of two points

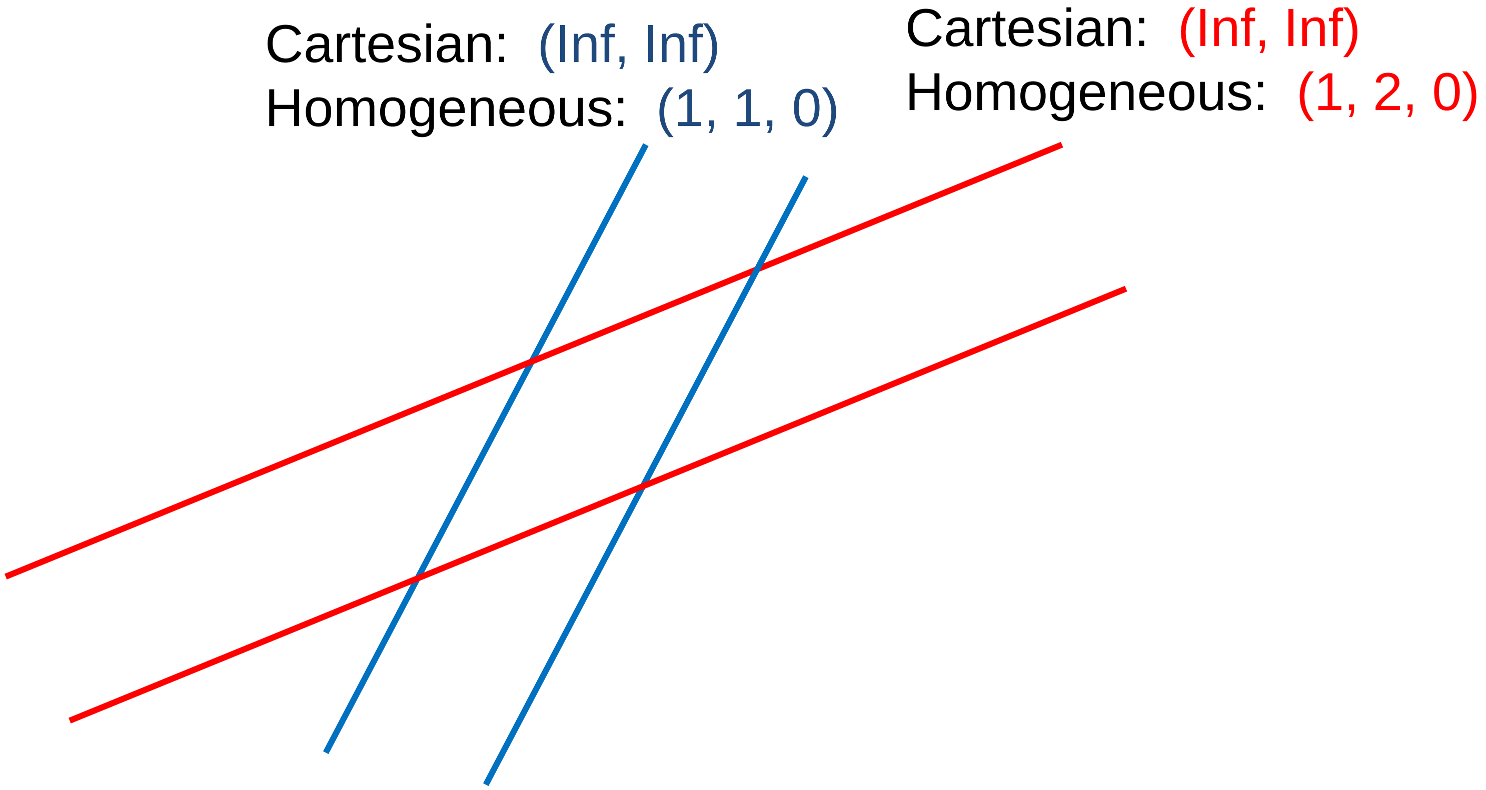
$$line_{ij} = p_i \times p_j$$

- Intersection of two lines given by cross product of the lines

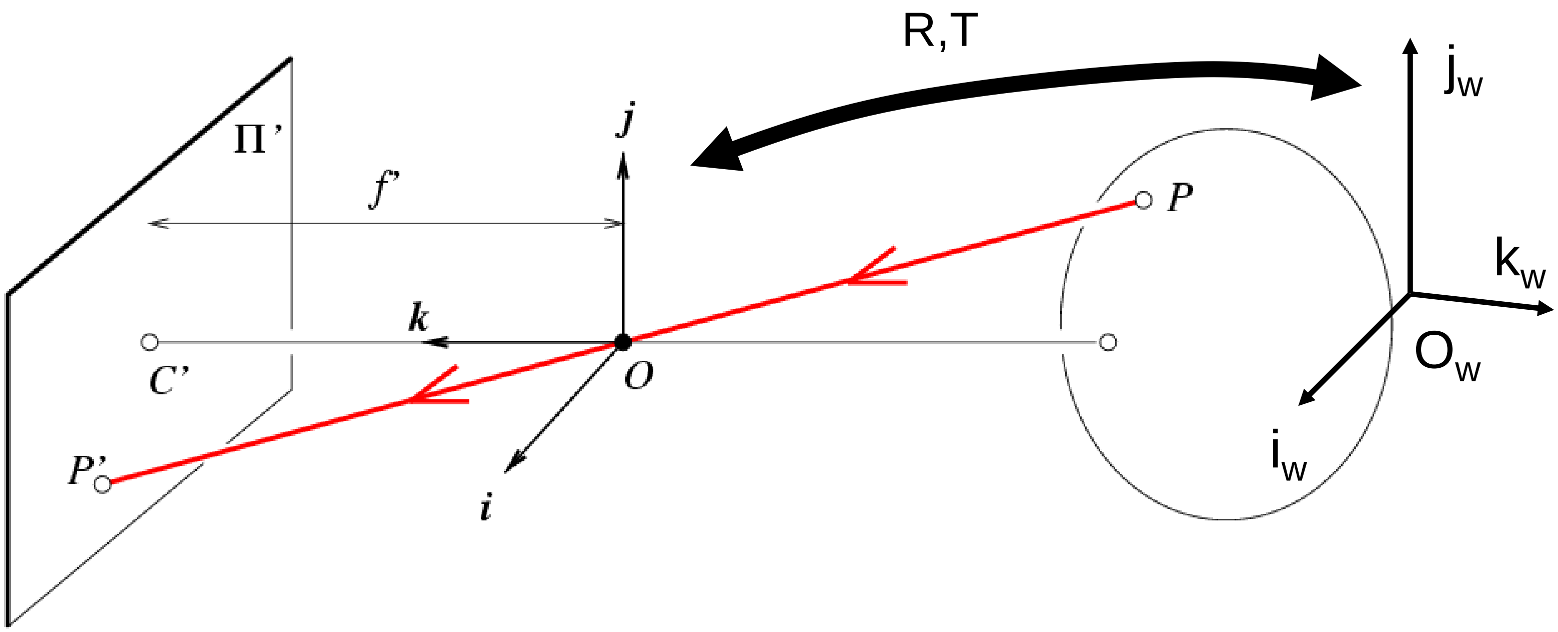
$$q_{ij} = line_i \times line_j$$

Another problem solved by homogeneous coordinates

Intersection of parallel lines



Projection matrix



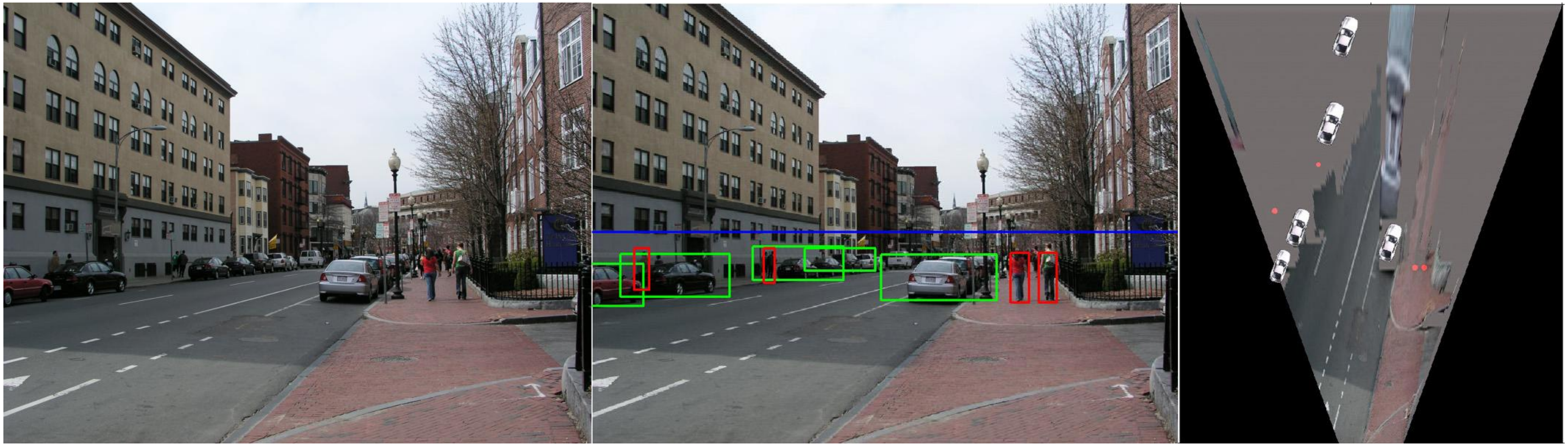
$$\mathbf{x} = \mathbf{K}\left[\mathbf{R} \quad \mathbf{t}\right]\mathbf{X}$$

x: Image Coordinates: (u,v,1)
K: Intrinsic Matrix (3x3)
R: Rotation (3x3)
t: Translation (3x1)
X: World Coordinates: (X,Y,Z,1)

Interlude: when have I used this stuff?

When have I used this stuff?

Object Recognition (CVPR 2006)

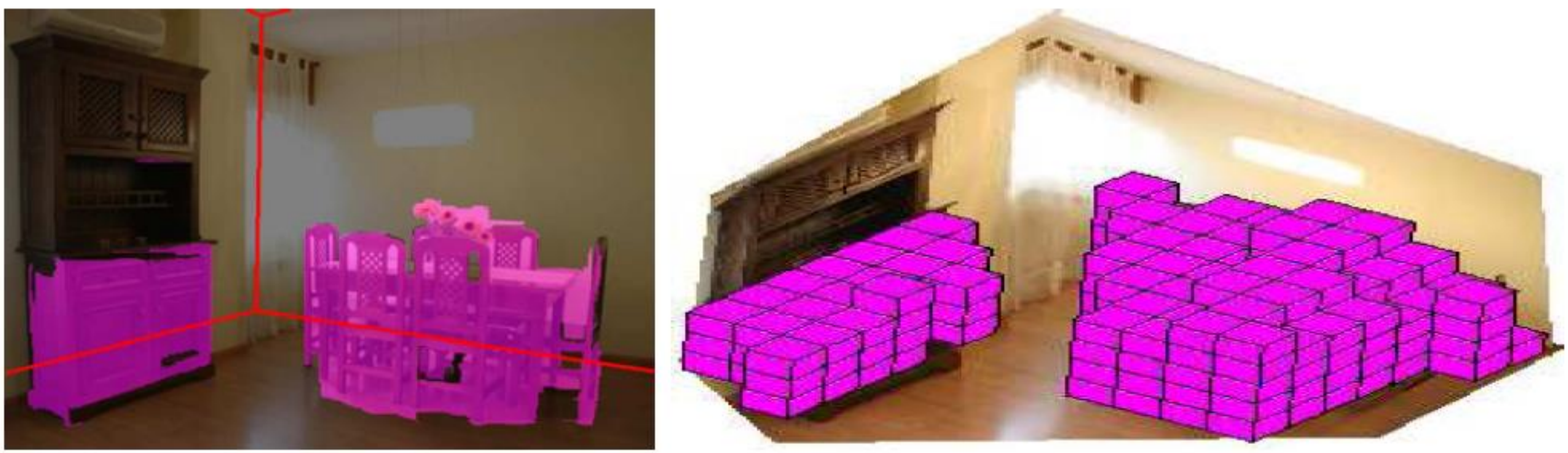


When have I used this stuff?

Single-view reconstruction (SIGGRAPH 2005)

When have I used this stuff?

Getting spatial layout in indoor scenes (ICCV 2009)

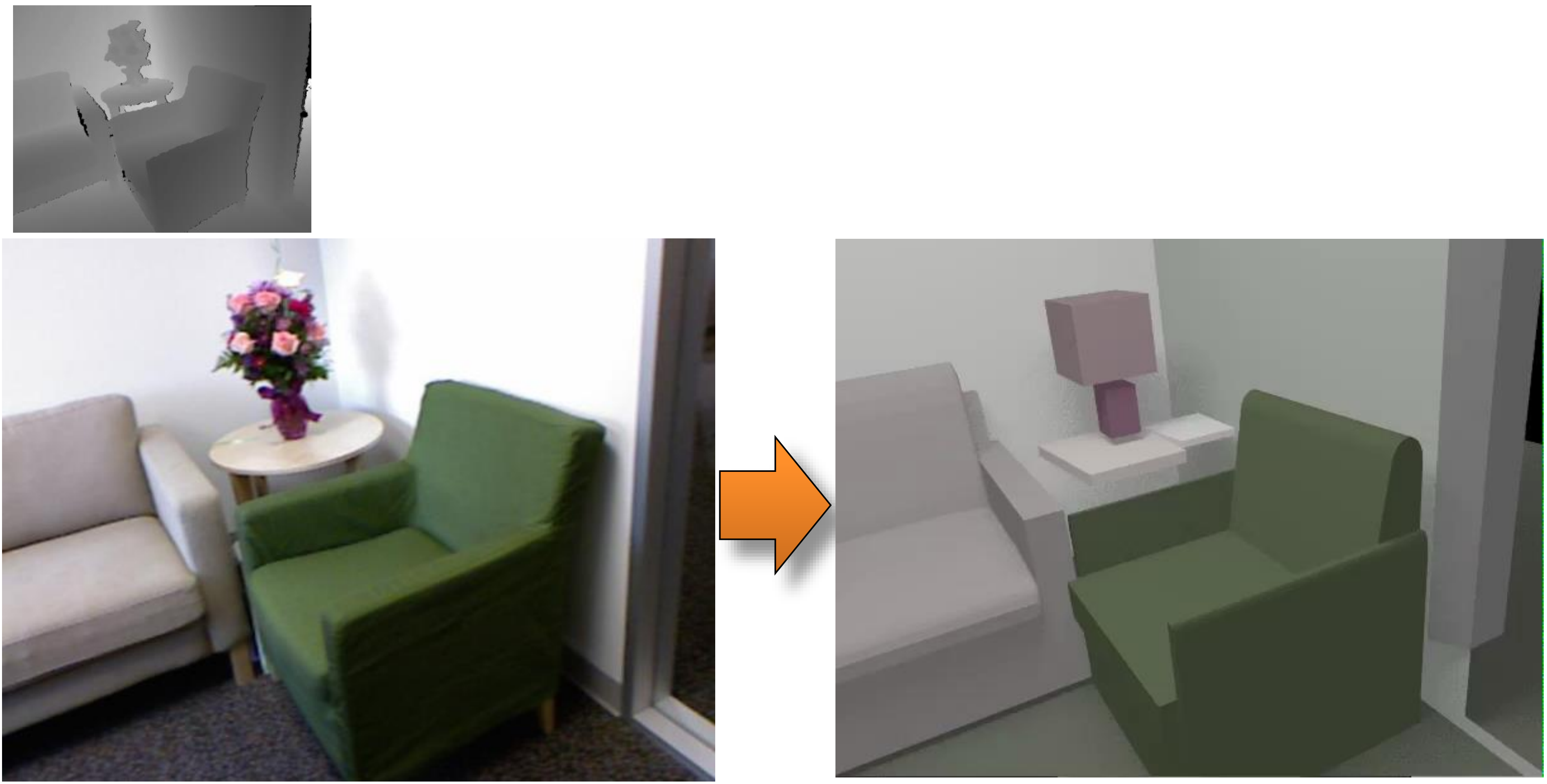


When have I used this stuff?

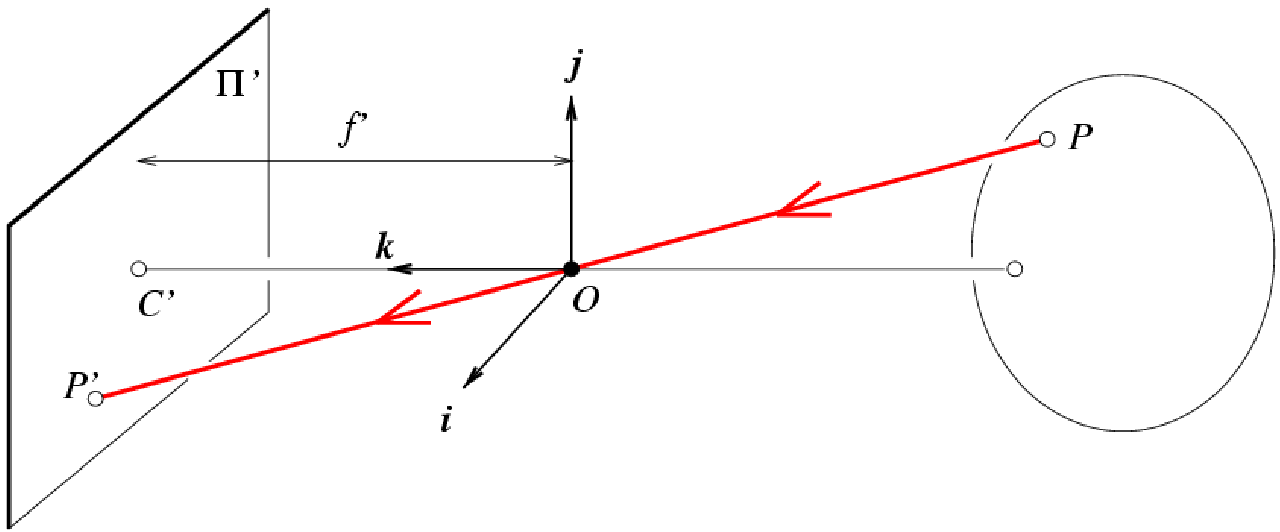
Inserting synthetic objects into images: http://vimeo.com/28962540

When have I used this stuff?

Creating detailed and complete 3D scene models from a single view (ongoing)



Projection matrix



Intrinsic Assumptions

- Unit aspect ratio
- Optical center at (0,0)
- No skew

Extrinsic Assumptions

- No rotation
- Camera at (0,0,0)

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X} \Rightarrow w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \underbrace{\begin{matrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{matrix}}_{\mathbf{K}} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Remove assumption: known optical center

Intrinsic Assumptions

- Unit aspect ratio
- No skew

Extrinsic Assumptions

- No rotation
- Camera at (0,0,0)

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X} \Rightarrow w\begin{bmatrix}u \\ v \\ 1\end{bmatrix} = \begin{bmatrix}f & 0 & u_0 & 0 \\ 0 & f & v_0 & 0 \\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x \\ y \\ z \\ 1\end{bmatrix}$$

Remove assumption: square pixels

Intrinsic Assumptions
- No skew

Extrinsic Assumptions
- No rotation
- Camera at (0,0,0)

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X} \Rightarrow w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Remove assumption: non-skewed pixels

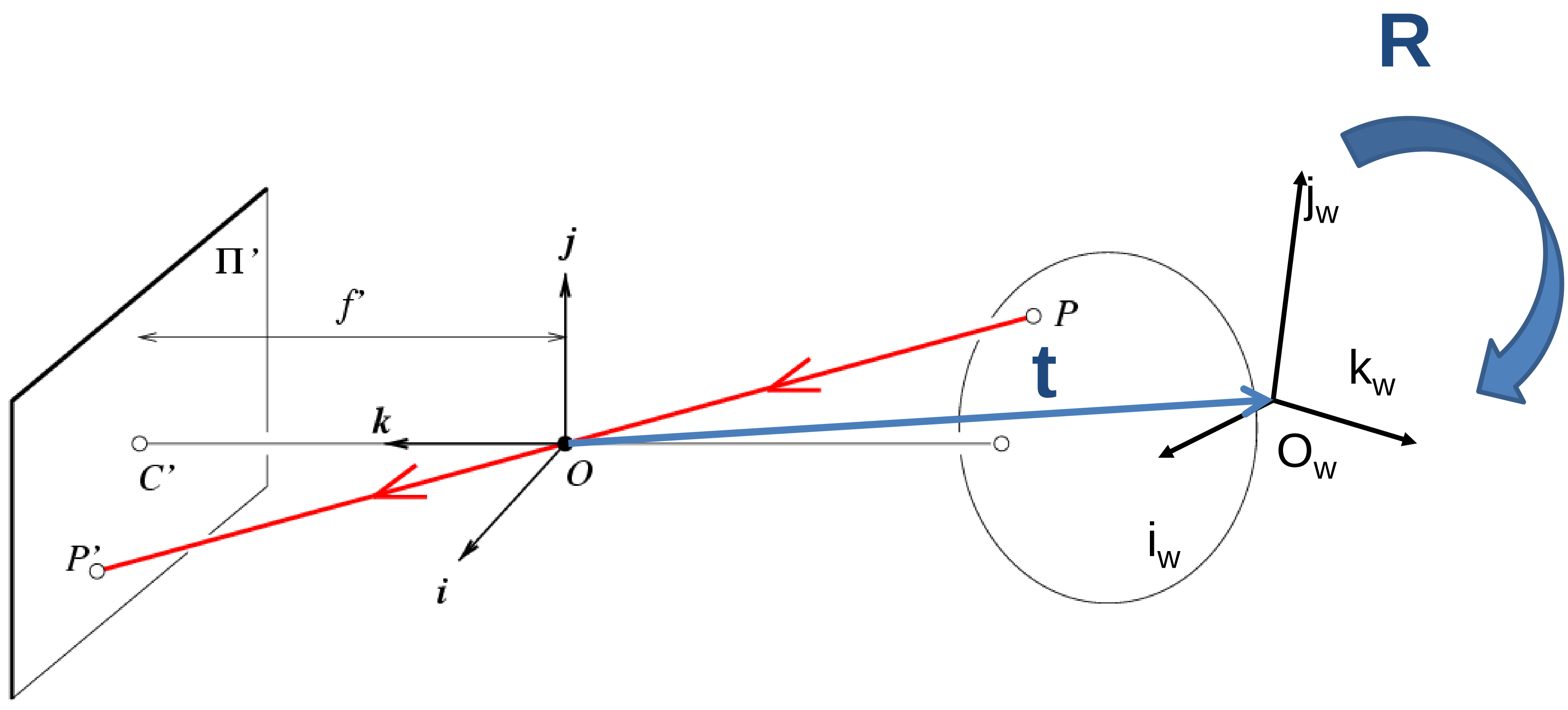
Intrinsic Assumptions

Extrinsic Assumptions
- No rotation
- Camera at (0,0,0)

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X} \Rightarrow w\begin{bmatrix}u\\v\\1\end{bmatrix} = \begin{bmatrix}\alpha & s & u_0 & 0\\0 & \beta & v_0 & 0\\0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$$

Note: different books use different notation for parameters

Oriented and Translated Camera



Allow camera translation

Intrinsic Assumptions

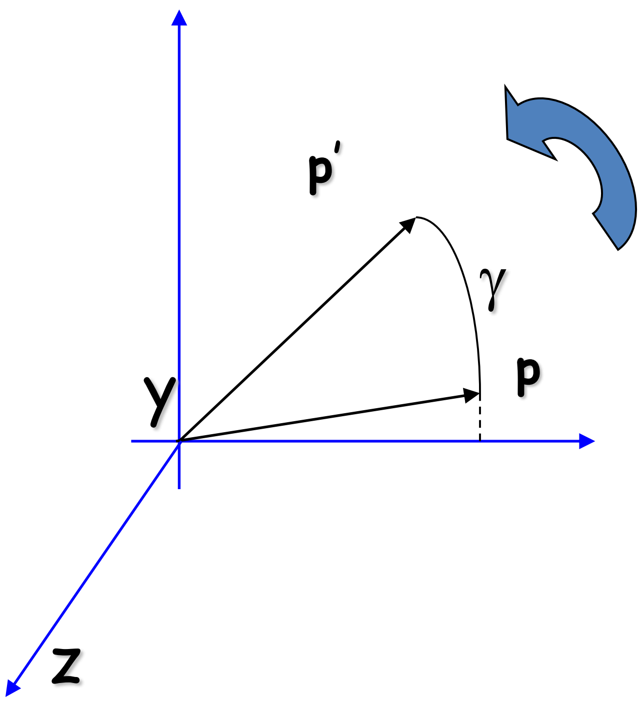
Extrinsic Assumptions
- No rotation

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{t}\end{bmatrix}\mathbf{X} \Rightarrow w\begin{bmatrix}u \\ v \\ 1\end{bmatrix} = \begin{bmatrix}\alpha & 0 & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z\end{bmatrix}\begin{bmatrix}x \\ y \\ z \\ 1\end{bmatrix}$$

Slide Credit: Saverese

3D Rotation of Points

Rotation around the coordinate axes, counter-clockwise:



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

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

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

Allow camera rotation

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Degrees of freedom

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \overset{5}{\begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}} \overset{6}{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

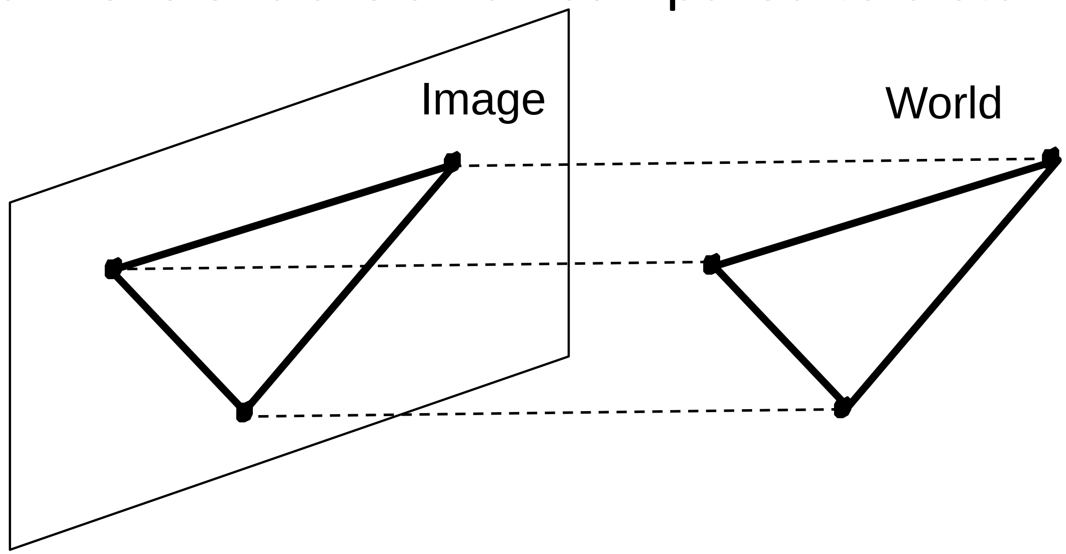
Vanishing Point = Projection from Infinity

$$\mathbf{p} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} \Rightarrow \mathbf{p} = \mathbf{KR}\begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \mathbf{p} = \mathbf{K}\begin{bmatrix} x_R \\ y_R \\ z_R \end{bmatrix}$$

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x_R \\ y_R \\ z_R \end{bmatrix} \Rightarrow \begin{matrix} u = \dfrac{f x_R}{z_R} + u_0 \\ \\ v = \dfrac{f y_R}{z_R} + v_0 \end{matrix}$$

Scaled Orthographic Projection

- Special case of perspective projection
  - Object dimensions are small compared to distance to camera



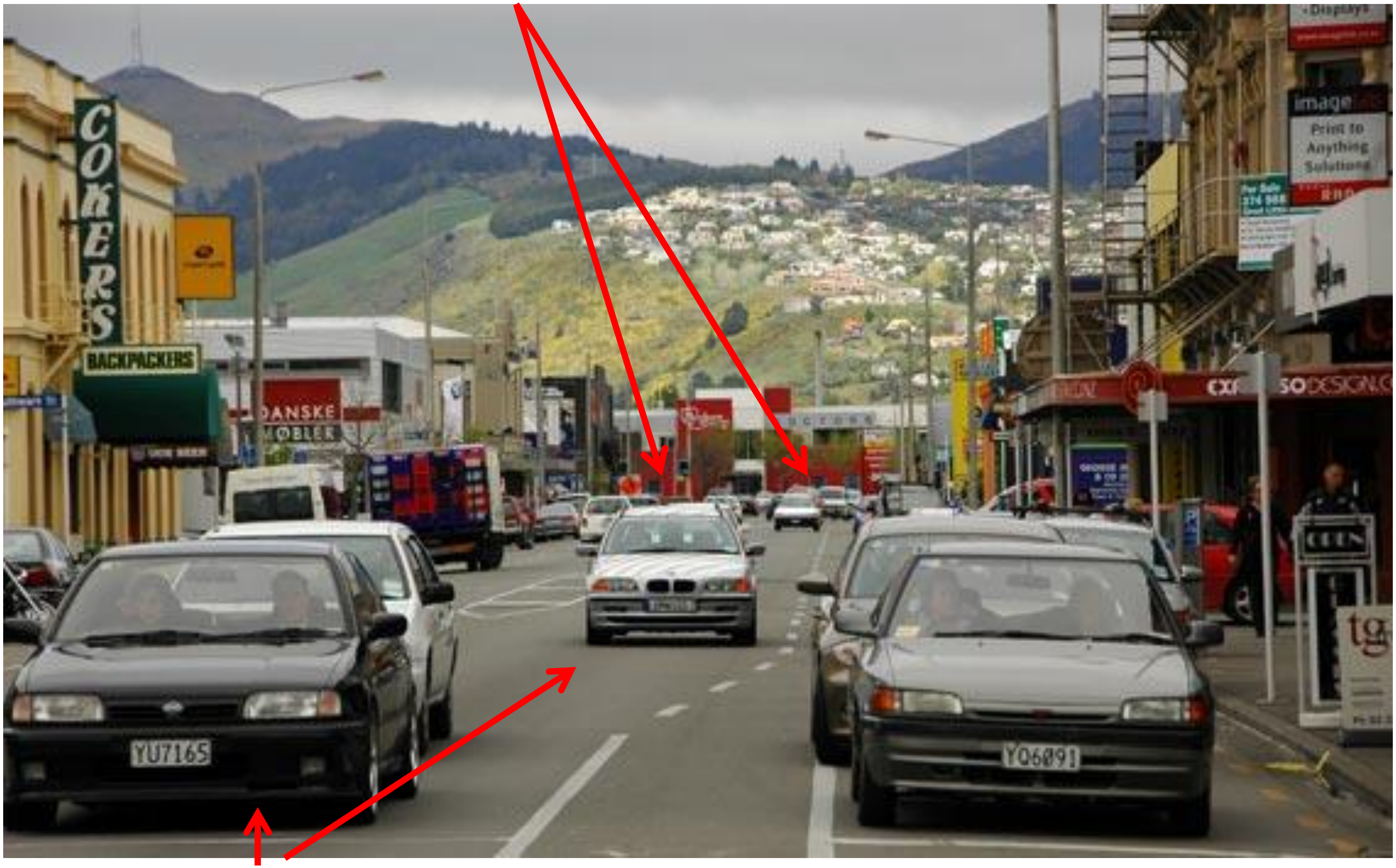
  - Also called "weak perspective"

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 0 & s \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Slide by Steve Seitz

Example

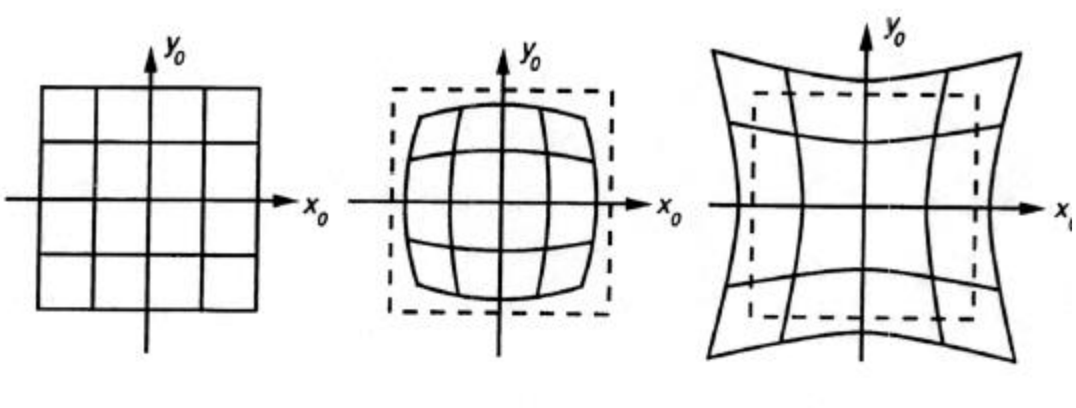
Far field: object appearance doesn't change as objects translate



Near field: object appearance changes as objects translate

Beyond Pinholes: Radial Distortion

- Common in wide-angle lenses or for special applications (e.g., security)
- Creates non-linear terms in projection
- Usually handled by through solving for non-linear terms and then correcting image



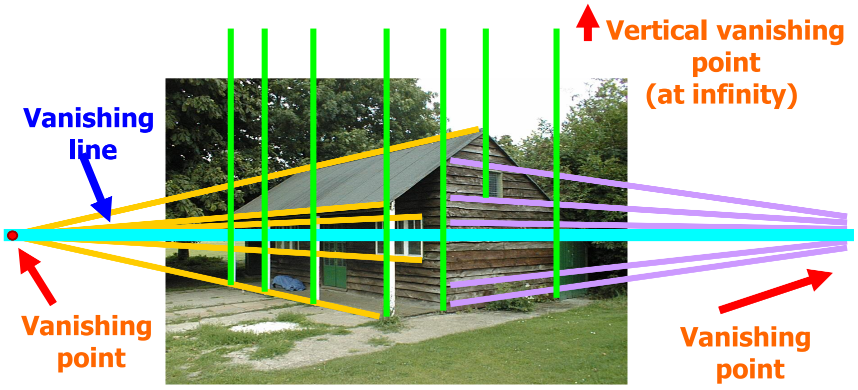
No Distortion Barrel Distortion Pincushion Distortion

Corrected Barrel Distortion

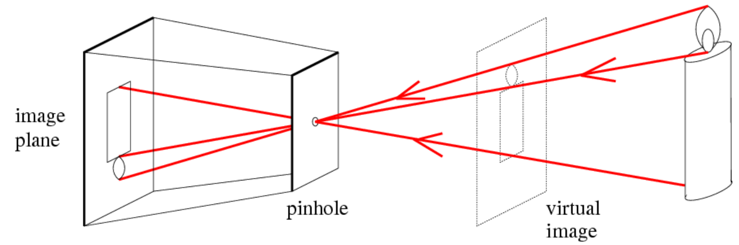
Image from Martin Habbecke

Things to remember

- Vanishing points and vanishing lines



- Pinhole camera model and camera projection matrix



$$\mathbf{x} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\mathbf{X}$$

- Homogeneous coordinates

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

Next class

- Applications of camera model and projective geometry
  - Recovering the camera intrinsic and extrinsic parameters from an image
  - Recovering size in the world
  - Projecting from one plane to another