Calibration and Single view meterology

Lecture slides by Steve Seitz (mostly)
Lecture presented by Varsha

CS 543/ ECE 549

26th January 2010

Todays class

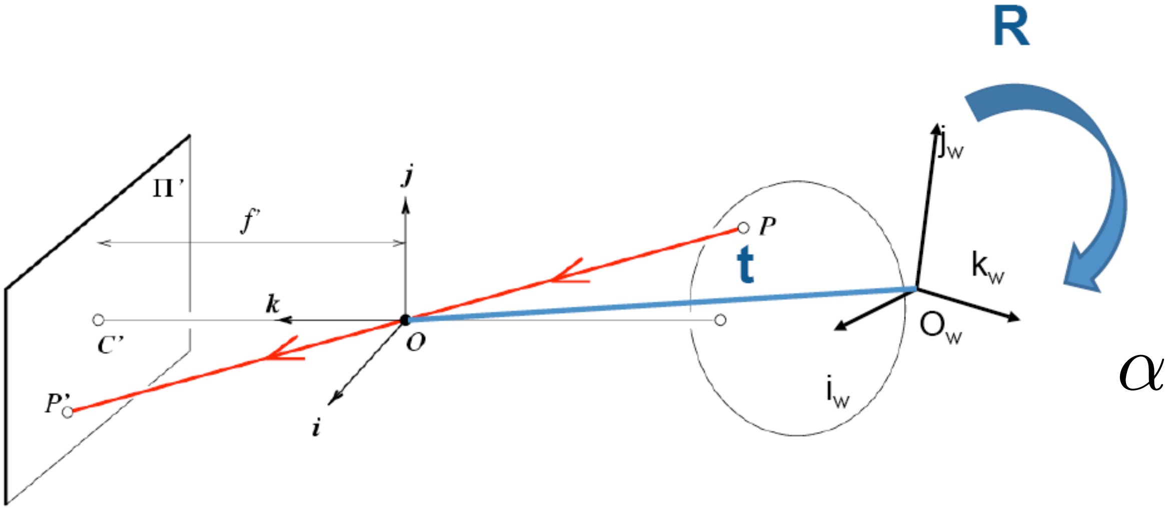
- Recovering the camera intrinsic and extrinsic parameters from an image
- Recovering size in the world
- Projecting from one plane to another (if time allows)

Recovering the camera intrinsic and extrinsic parameters from an image

Recap 3D to 2D: “perspective” projection

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

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



C’-(u0,v0) optical center

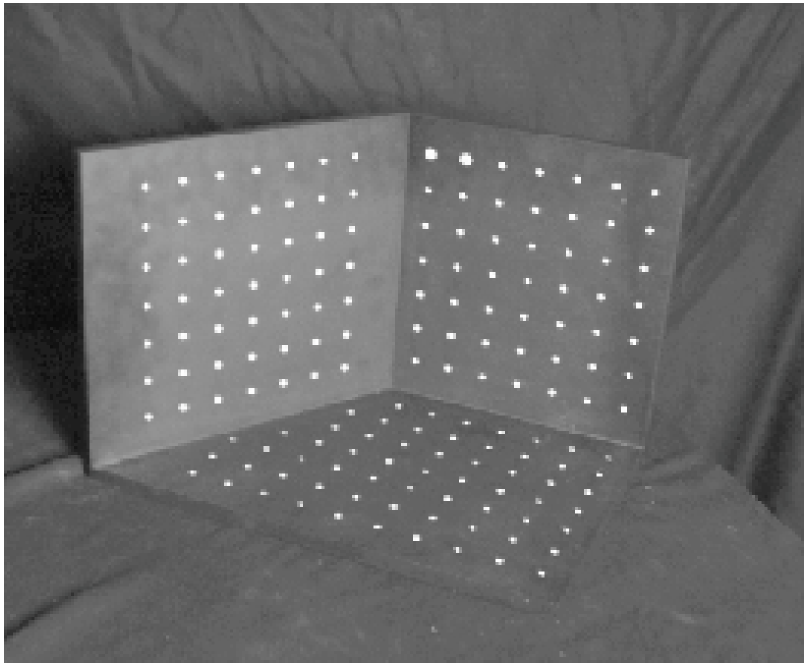
α, β - focal length in x and y direction

s - skew

Calibration using a reference object

Place a known object in the scene

- identify correspondence between image and scene
- compute mapping from scene to image



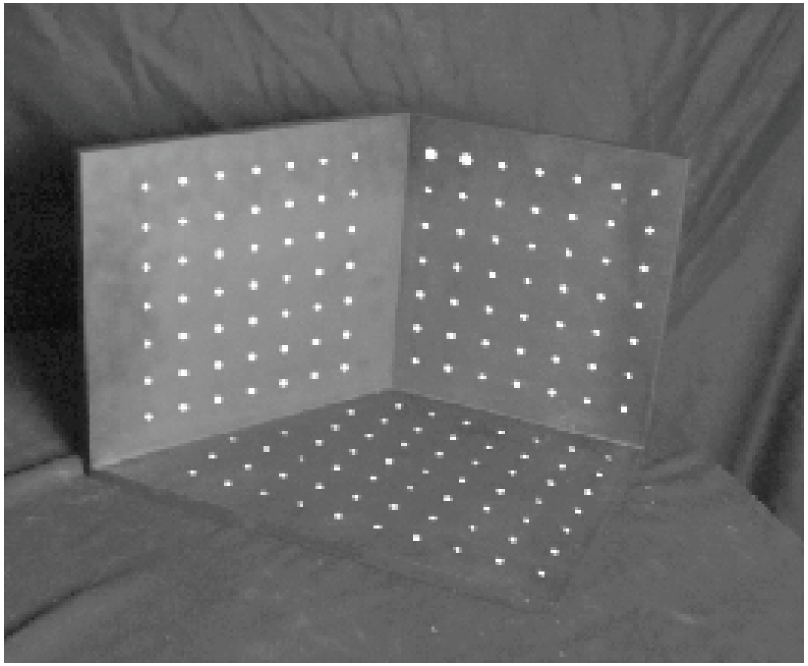
Issues

- must know geometry very accurately
- must know 3D->2D correspondence

Estimating the projection matrix

Place a known object in the scene

- identify correspondence between image and scene
- compute mapping from scene to image



$$\lambda \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$$\Downarrow$$

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

Direct linear calibration

$$\lambda \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$$\begin{aligned} u_i &= \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}} \\ v_i &= \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}} \end{aligned}$$

Direct linear calibration

$$\lambda \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$$\begin{aligned} u_i &= \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}} \\ v_i &= \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}} \end{aligned}$$

$$\begin{aligned} u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) &= m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03} \\ v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) &= m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13} \end{aligned}$$

Direct linear calibration

$$\lambda \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$$\begin{aligned} u_i &= \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}} \\ v_i &= \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}} \end{aligned}$$

$$\begin{aligned} u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) &= m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03} \\ v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) &= m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13} \end{aligned}$$

$$\begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -u_iX_i & -u_iY_i & -u_iZ_i & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -v_iX_i & -v_iY_i & -v_iZ_i & -v_i \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Ax=0 form

Direct linear calibration

$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 & -u_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 & -v_1 \\ & & & & & & \vdots & & & & & \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -u_nX_n & -u_nY_n & -u_nZ_n & -u_n \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -v_nX_n & -v_nY_n & -v_nZ_n & -v_n \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

Ax=0 form

Can solve for m_{ij} by linear least squares

Direct linear calibration

Advantage:

- Very simple to formulate and solve

Disadvantages:

- Doesn’t tell you the camera parameters
- Doesn’t model radial distortion
- Hard to impose constraints (e.g., known focal length)
- Doesn’t minimize the right error function

For these reasons, *nonlinear methods* are preferred

- Define error function E between projected 3D points and image positions
  - E is nonlinear function of intrinsics, extrinsics, radial distortion
- Minimize E using nonlinear optimization techniques
  - e.g., variants of Newton’s method

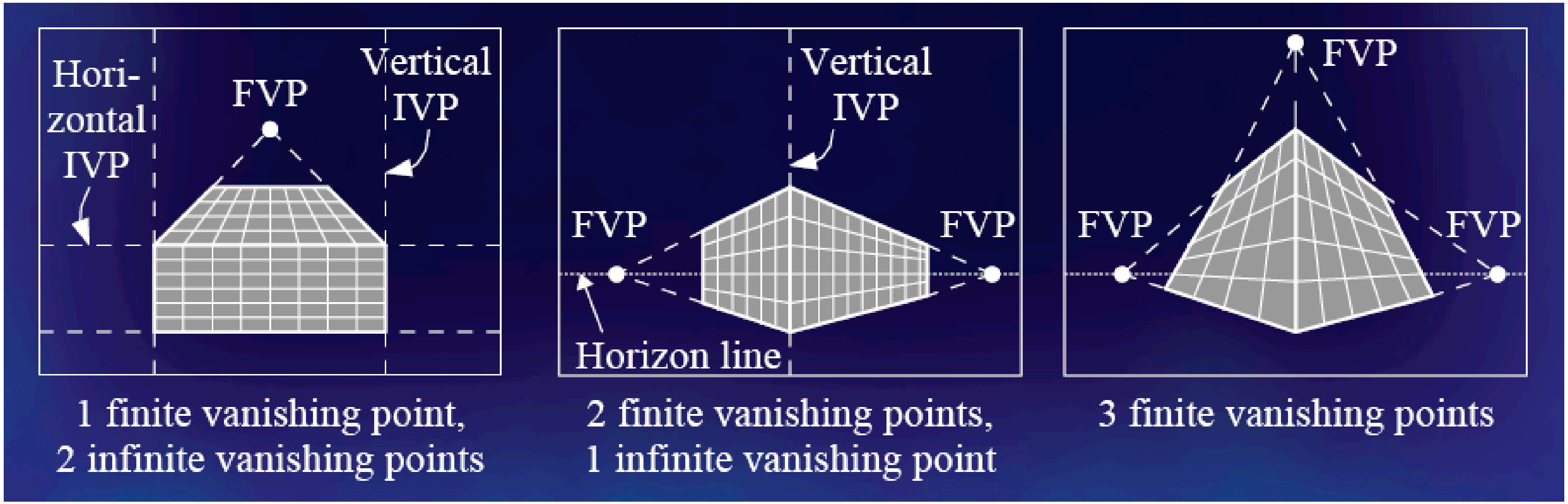
Summary

- Known correspondences (ui,vi) ~ (Xi,Yi,Zi)
- Compute mij solving system of linear equations
- May use this to initialize non linear error minimization problem to recover more accurate mij
- Note that some normalization of points is necessary for result to be invariant to different choices of coordinate systems.

Camera calibration from an image

- What if world coordinates are not known?
- Can we use scene features(vanishing points)?

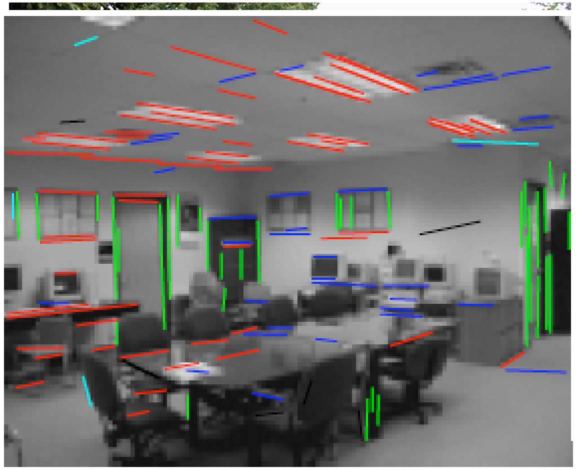
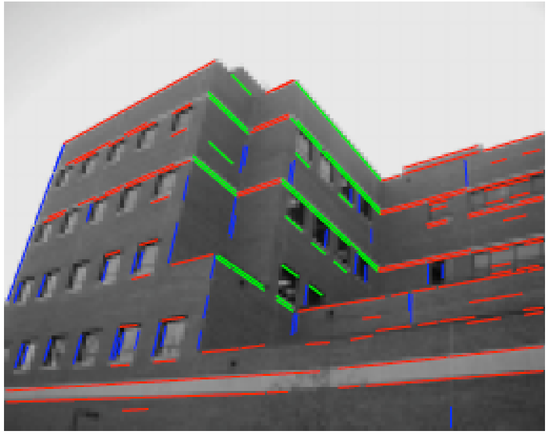
- Scene contains lines along directions that are orthogonal



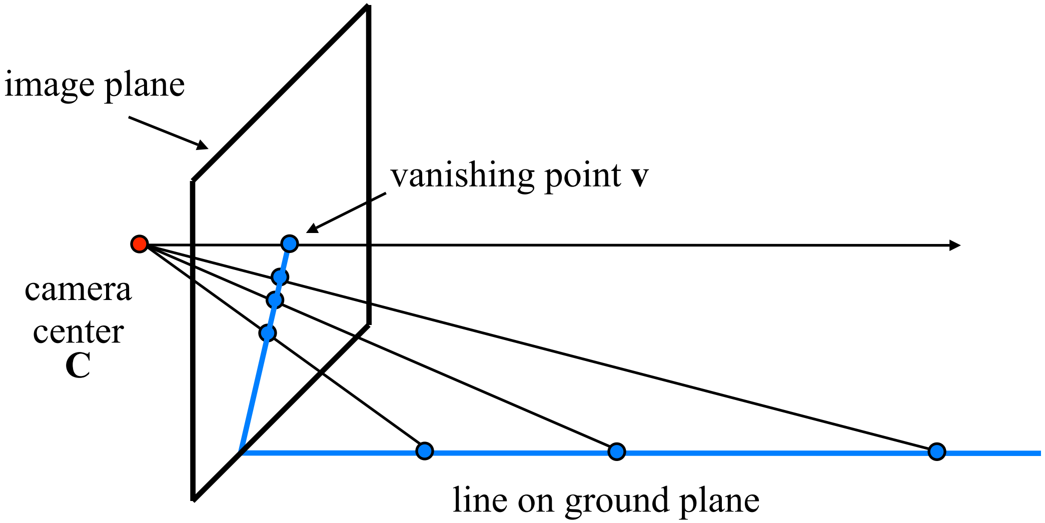
1 finite vanishing point,
2 infinite vanishing points

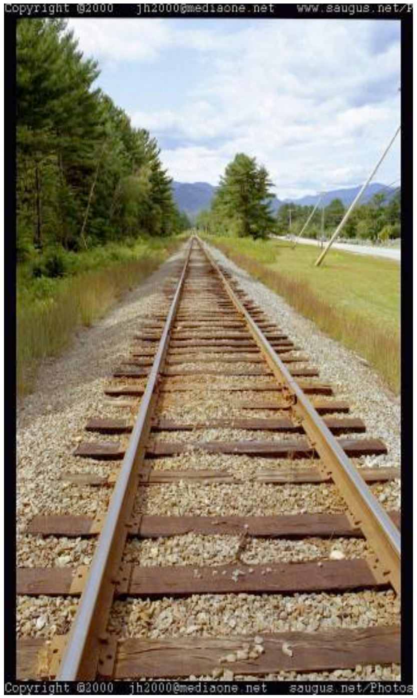
2 finite vanishing points,
1 infinite vanishing point

3 finite vanishing points

Recap: Vanishing points

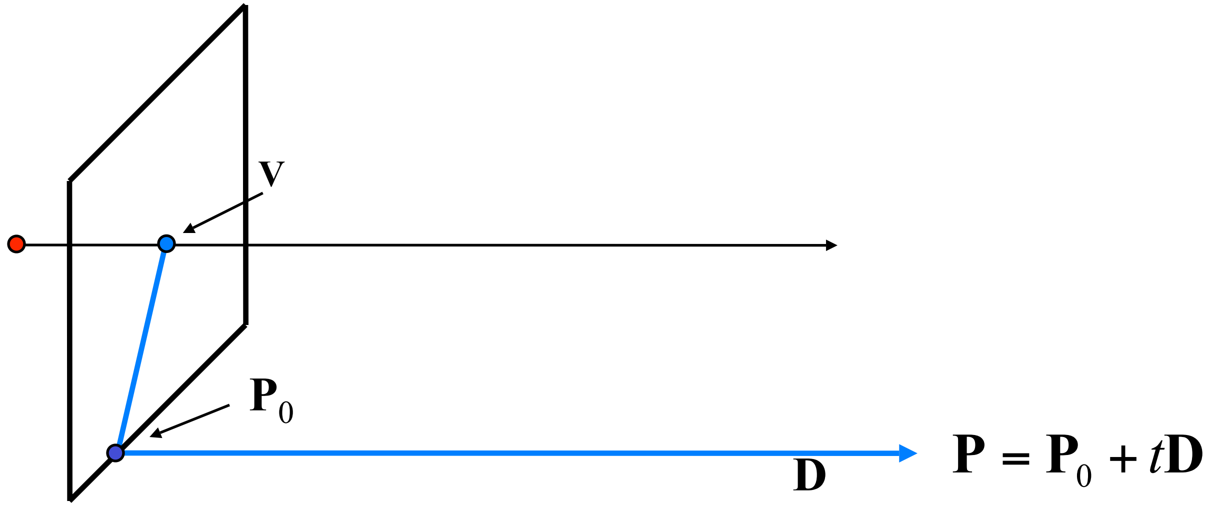



Copyright ©2000 jh2000@mediaone.net www.saugus.net/P
Copyright ©2000 jh2000@mediaone.net saugus.net/Photos

Vanishing point

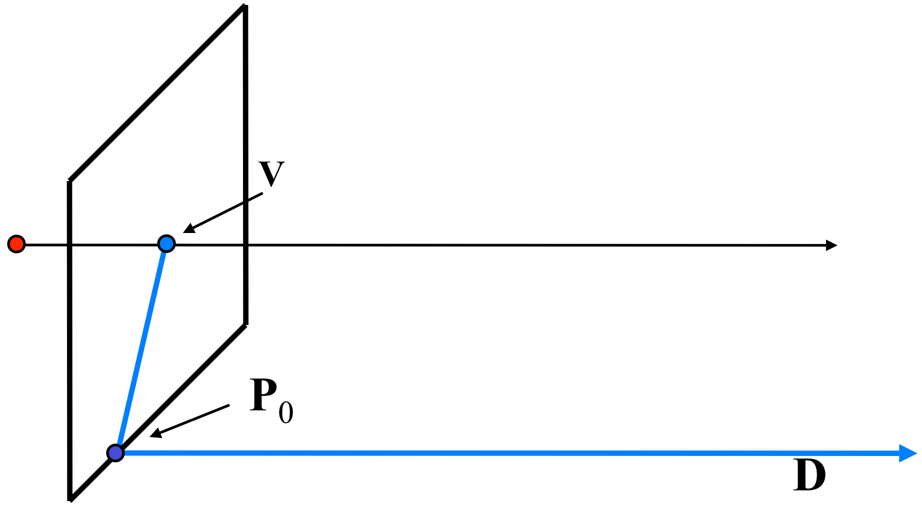
- projection of a point at infinity

vanishing points



$$\mathbf{P}_t = \begin{bmatrix} P_X + tD_X \\ P_Y + tD_Y \\ P_Z + tD_Z \\ 1 \end{bmatrix}$$

vanishing points



$$\mathbf{P} = \mathbf{P}_0 + t\mathbf{D}$$

$$\mathbf{P}_t = \begin{bmatrix} P_X + tD_X \\ P_Y + tD_Y \\ P_Z + tD_Z \\ 1 \end{bmatrix} \cong \begin{bmatrix} P_X / t + D_X \\ P_Y / t + D_Y \\ P_Z / t + D_Z \\ 1 / t \end{bmatrix} \qquad t -> \infty \qquad \mathbf{P}_\infty \cong \begin{bmatrix} D_X \\ D_Y \\ D_Z \\ 0 \end{bmatrix}$$

Properties $\boldsymbol{v} = K\left[R \quad \boldsymbol{t}\right] \boldsymbol{P}_\infty = KR\boldsymbol{D}$

- $\mathbf{P}_\infty$ is a point at *infinity*, **v** is its projection
- They depend only on line *direction*

Camera calibration

- From vanishing points corresponding to 3 orthogonal directions of world

$$e_i = [1,0,0]^T,\ e_j = [0,1,0]^T,\ e_k = [0 \quad 0 \quad 1]^T$$

$$\mathbf{v}_i = KRe_i, \mathbf{v}_j = KRe_j, \mathbf{v}_k = KRe_k.$$

$$e_i^T e_j = 0$$

$$\mathbf{v}_i^T K^{-T} R R^T K^{-1} \mathbf{v}_j = \mathbf{v}_i^T K^{-T} K^{-1} \mathbf{v}_j \quad = 0$$

- zero skew, unit aspect ratio

$$K = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad K^{-1} = \begin{bmatrix} 1/f & 0 & -u_0/f \\ 0 & 1/f & -v_0/f \\ 0 & 0 & 1 \end{bmatrix}$$

$$v_i^T K^{-T} K^{-1} v_j = 0$$
$$v_j^T K^{-T} K^{-1} v_k = 0$$
$$v_i^T K^{-T} K^{-1} v_k = 0$$

- 3 finite vanishing points: get f, u0, v0
- 2 finite and one infinite : u0,v0 as point on vf1 vf2 closest to image center, get f
- 2 infinite vanishing points : f cant be recovered u0, v0 is at the third vanishing point

Rotation from vanishing points

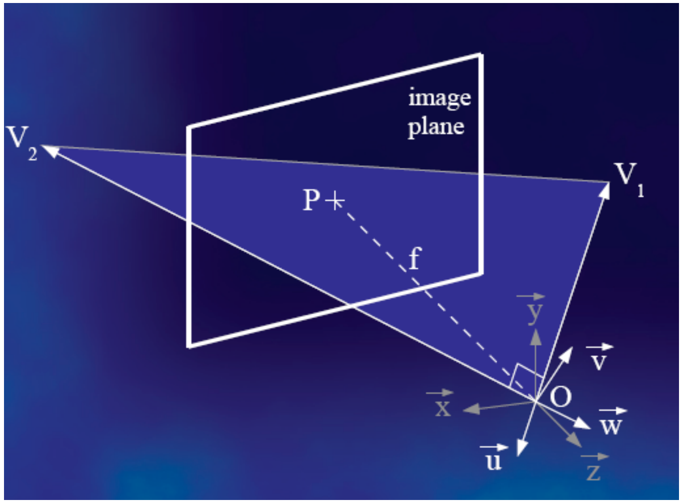
- R_{1c} 1st column vector of Rotation matrix

$$R = \begin{bmatrix} R_{1c} & R_{2c} & R_{3c} \end{bmatrix}$$

$$\lambda v_i = KRe_i \qquad e_i = [1, 0, 0]^T$$

$$R_{1c} = \lambda K^{-1} v_i$$

- λ from $||R_{1c}||_2 = 1$



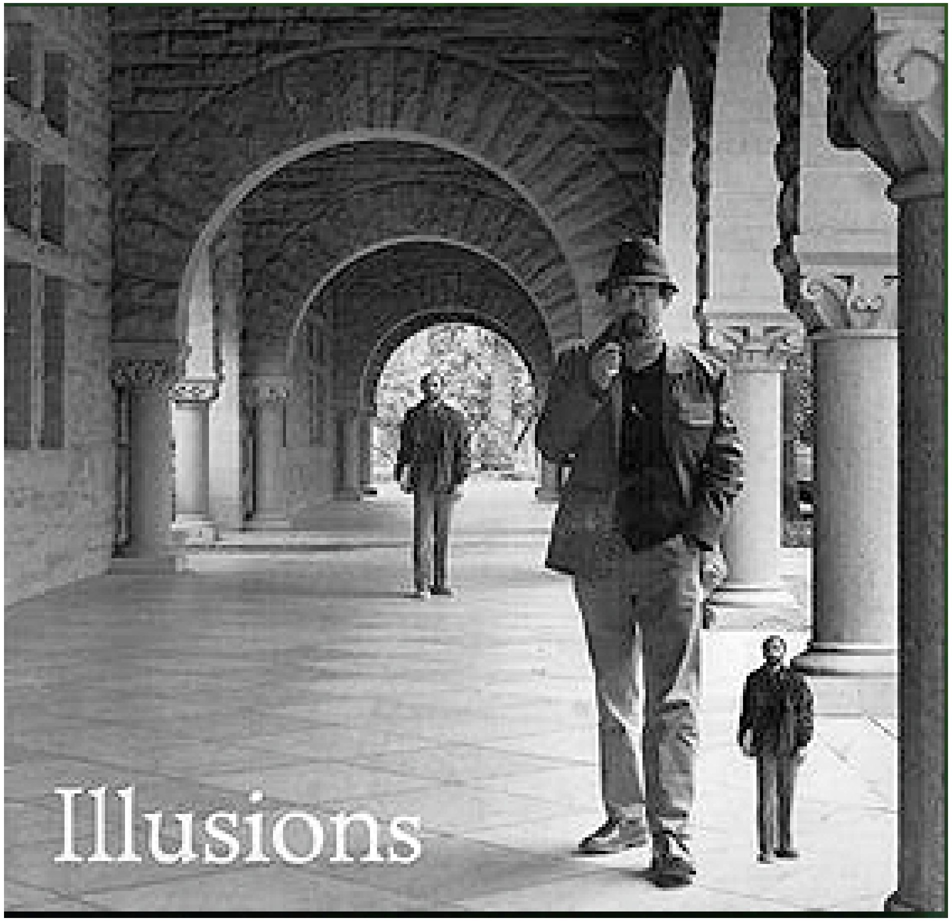
Summary

- Orthogonality constraints for solving K
- Rotation directly from vanishing points if camera is known

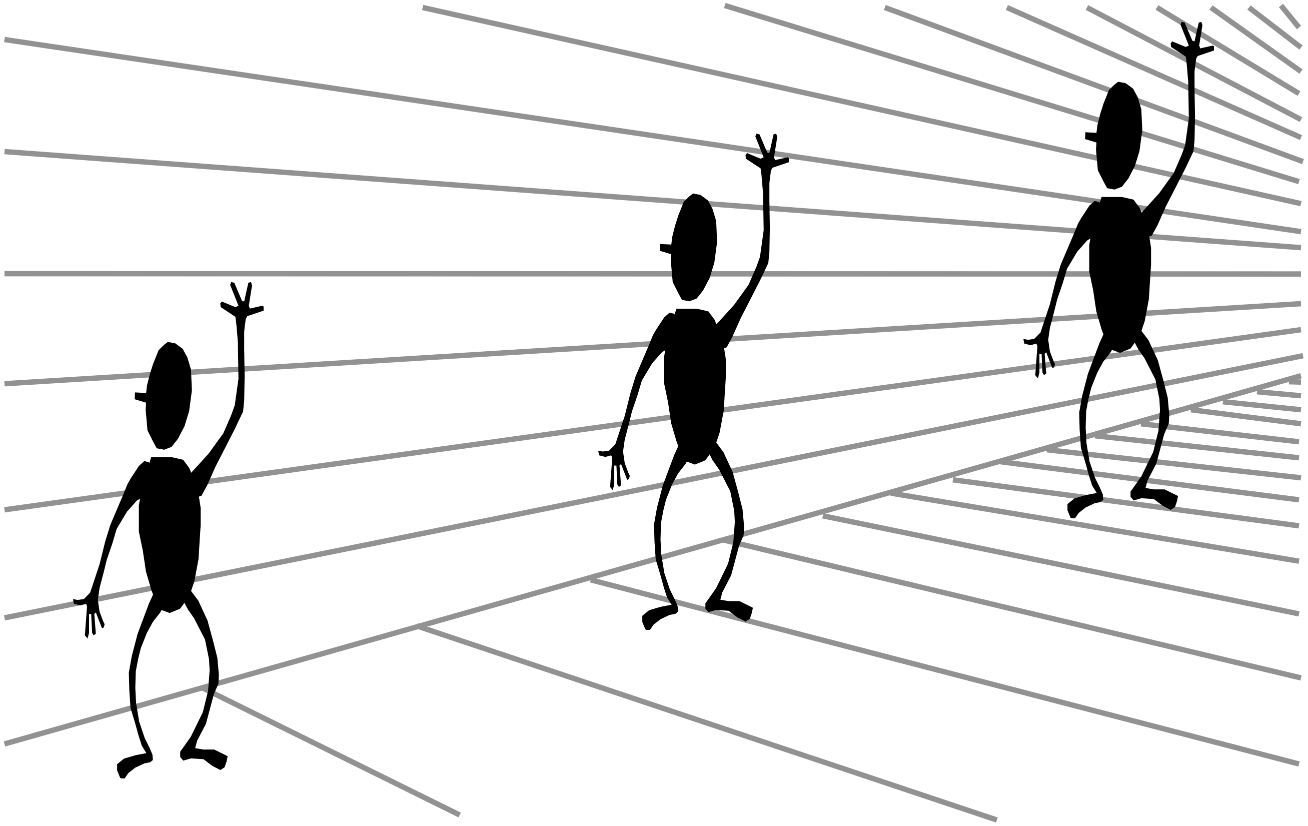
- Calibrating camera using vanishing points
- Advantages
  - No need to specify point correspondences
  - Could be completely automatic
- Disadvantages
  - Problems due to infinite vanishing points
  - inaccuracies in computation of vanishing points

Recovering size in the world

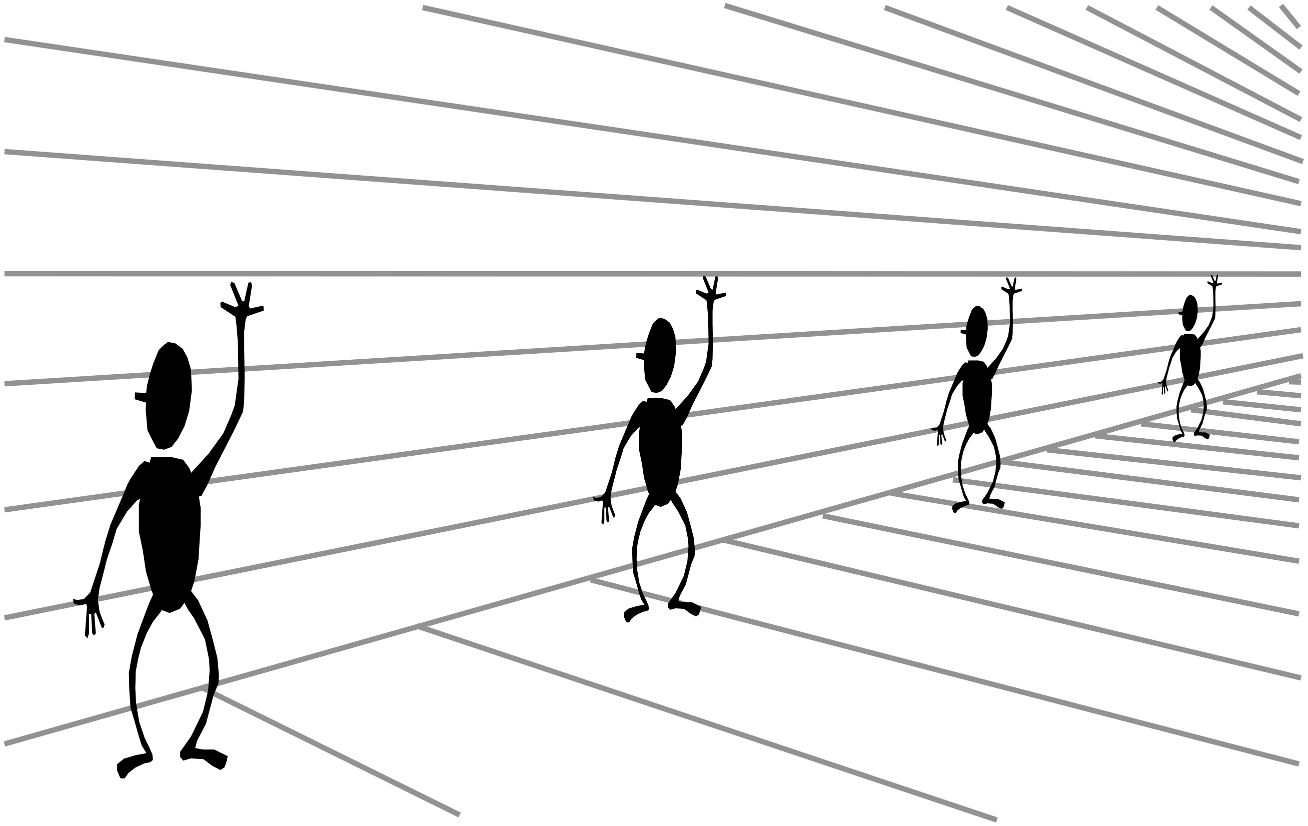
Fun with vanishing points



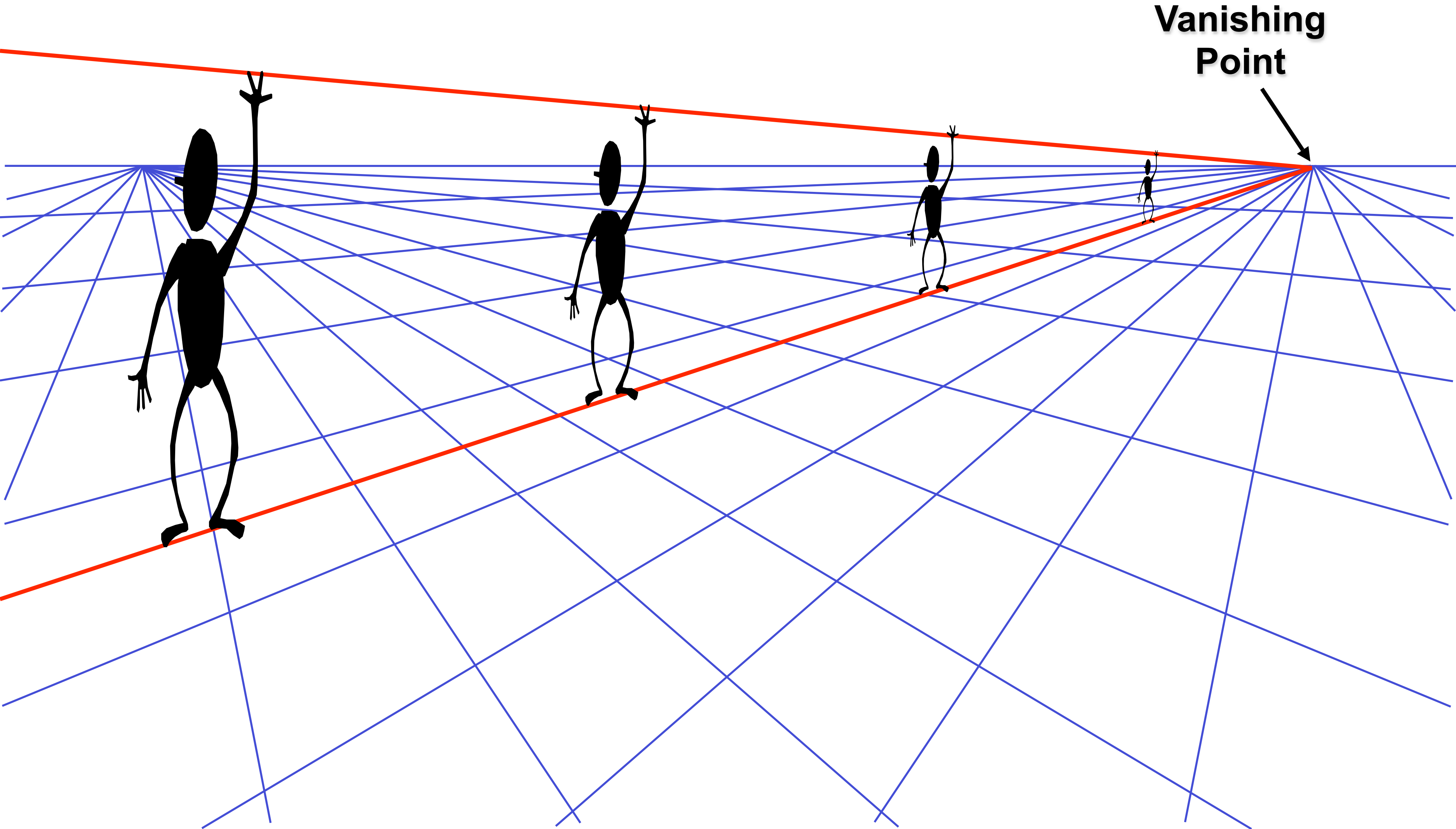
Perspective cues



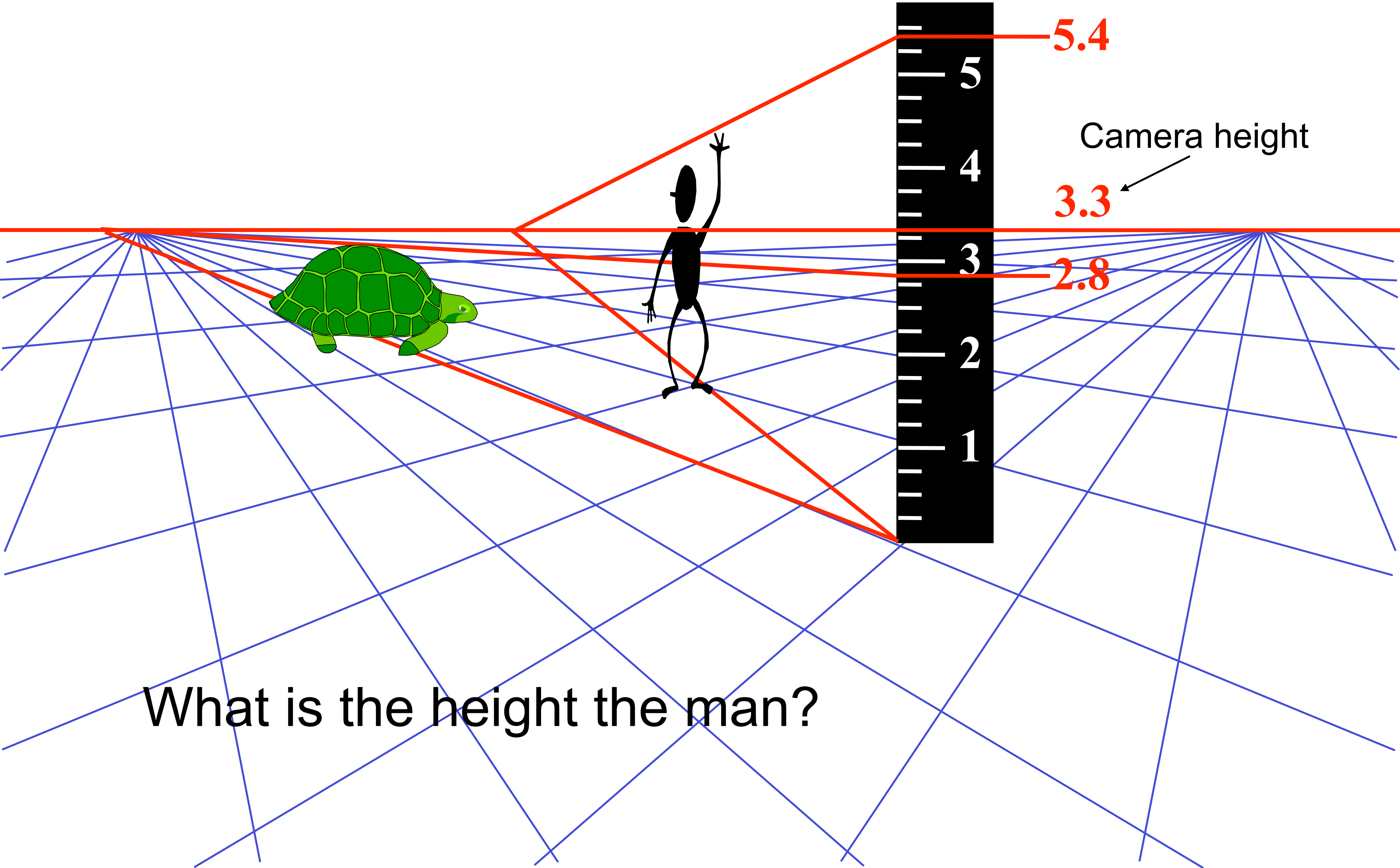
Perspective cues



Comparing heights

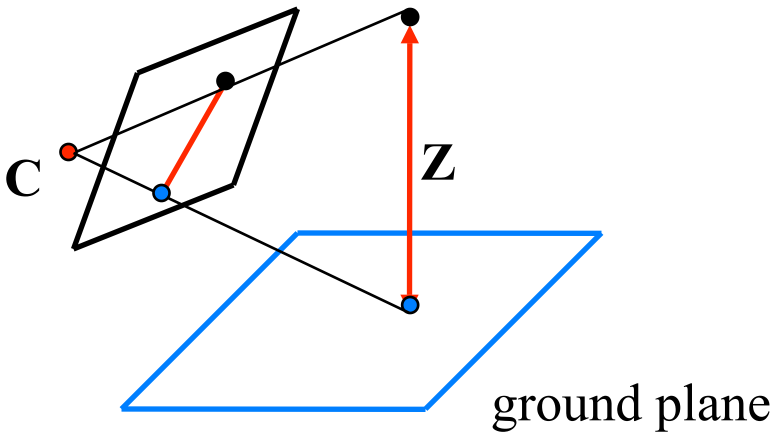


Measuring height



What is the height the man?

Measuring height without a ruler



Compute Z from image measurements

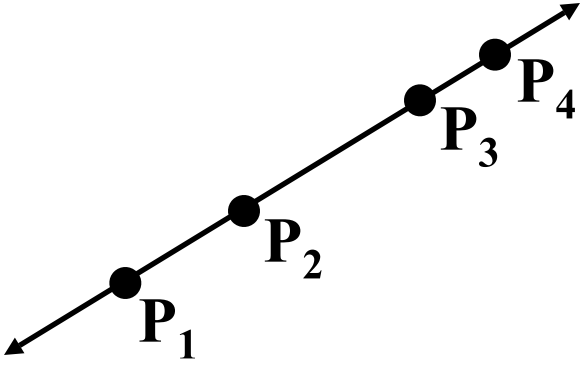
- Need more than vanishing points to do this

The cross ratio

A Projective Invariant

- Something that does not change under projective transformations (including perspective projection)

The cross-ratio of 4 collinear points



$$\frac{\|\mathbf{P}_3 - \mathbf{P}_1\| \, \|\mathbf{P}_4 - \mathbf{P}_2\|}{\|\mathbf{P}_3 - \mathbf{P}_2\| \, \|\mathbf{P}_4 - \mathbf{P}_1\|}$$

$$\mathbf{P}_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

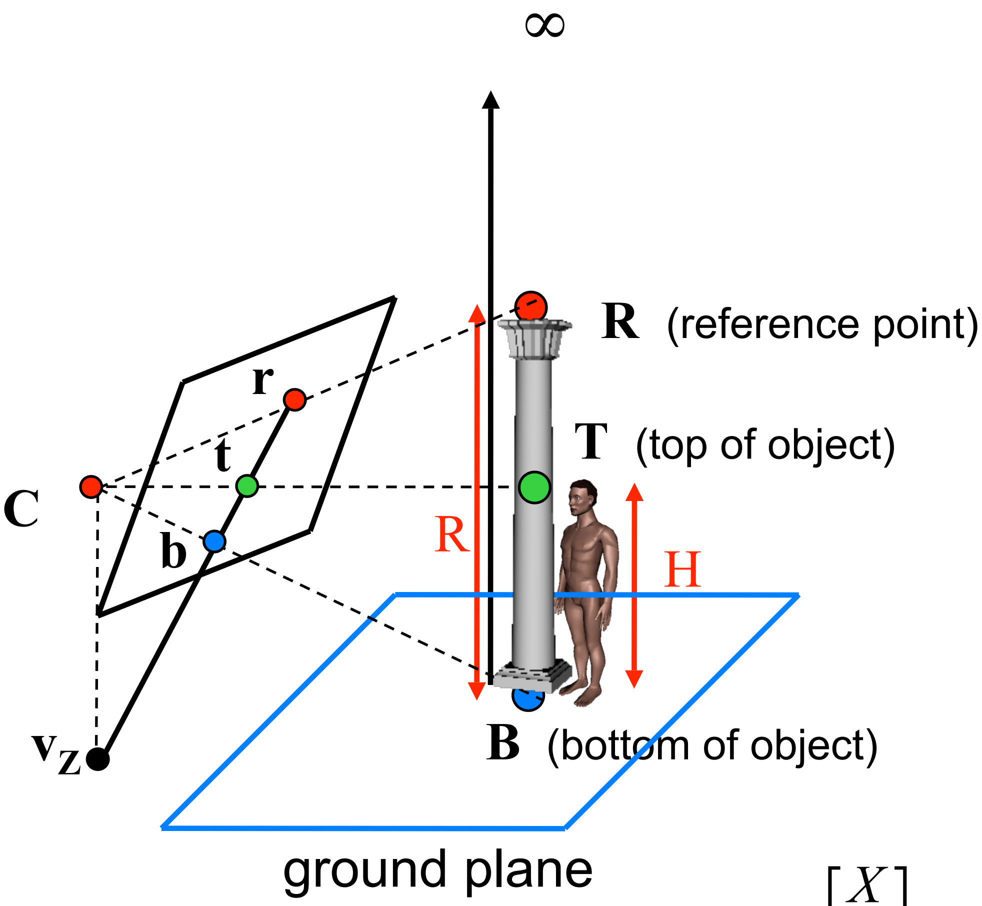
Can permute the point ordering

$$\frac{\|\mathbf{P}_1 - \mathbf{P}_3\| \, \|\mathbf{P}_4 - \mathbf{P}_2\|}{\|\mathbf{P}_1 - \mathbf{P}_2\| \, \|\mathbf{P}_4 - \mathbf{P}_3\|}$$

- 4! = 24 different orders (but only 6 distinct values)

This is the fundamental invariant of projective geometry

Measuring height



$$\frac{\|\mathbf{T}-\mathbf{B}\|\,\|\infty-\mathbf{R}\|}{\|\mathbf{R}-\mathbf{B}\|\,\|\infty-\mathbf{T}\|} = \frac{H}{R}$$

scene cross ratio

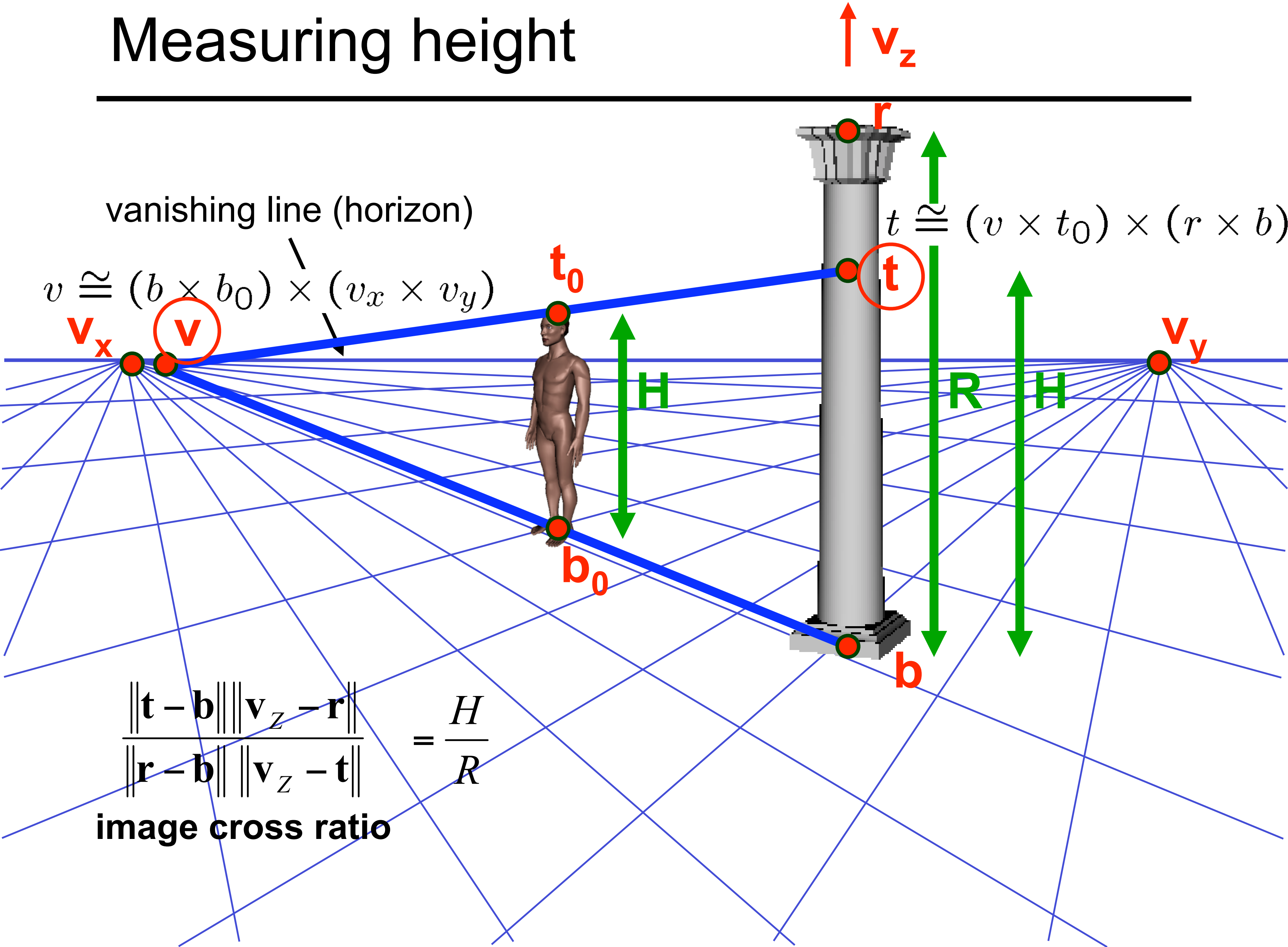
$$\frac{\|\mathbf{t}-\mathbf{b}\|\|\mathbf{v}_Z-\mathbf{r}\|}{\|\mathbf{r}-\mathbf{b}\|\,\|\mathbf{v}_Z-\mathbf{t}\|} = \frac{H}{R}$$

image cross ratio

scene points represented as $\mathbf{P} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$

image points as $\mathbf{p} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Measuring height



$$v \cong (b \times b_0) \times (v_x \times v_y)$$

$$t \cong (v \times t_0) \times (r \times b)$$

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

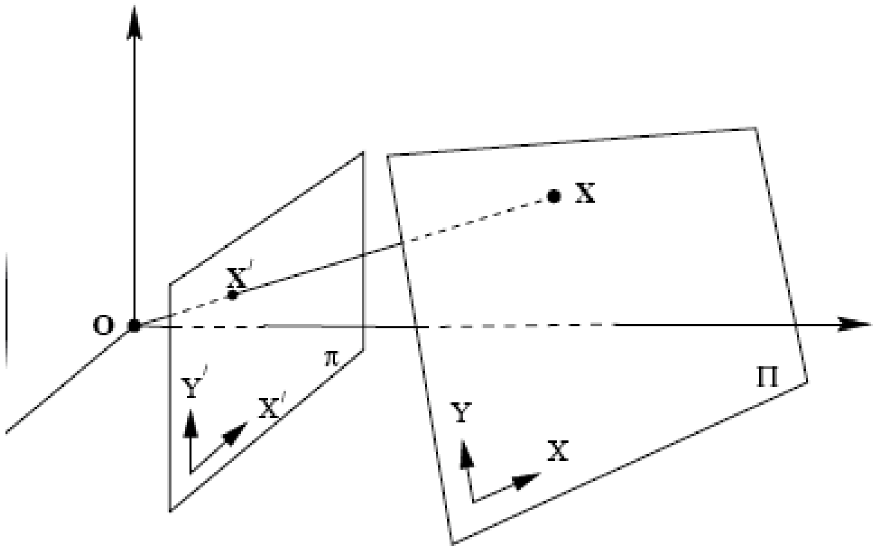
image cross ratio

Summary

- Cross ratio is invariant to projective transformation
- Apply this to get relative sizes of objects
- Perspective is important for infering object sizes

Projecting from one plane to another

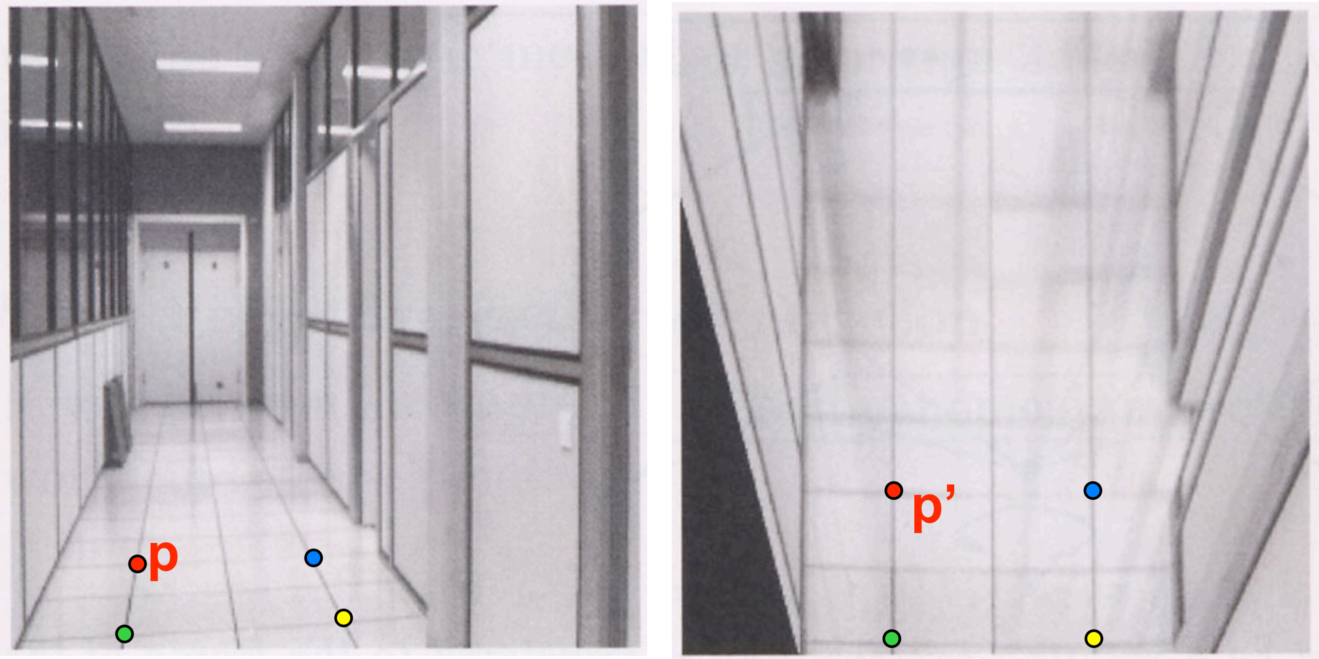
Homography



Homography is a mapping between points of one plane to another under projective transformation through a single point

x’=Hx H is 3X3 matrix, has 8 degrees of freedom

Examples: Image rectification



To unwarp (rectify) an image

- solve for homography **H** given **x** and **x’**
- solve equations of the form: w**x’** = **Hx**
  - linear in unknowns: w and coefficients of **H**
  - H is defined up to an arbitrary scale factor
  - how many points are necessary to solve for **H**?

Solving for homographies

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{aligned} x'_i &= \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}} \\ y'_i &= \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}} \end{aligned}$$

$$\begin{aligned} x'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\ y'_i(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12} \end{aligned}$$

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -y'_i x_i & -y'_i y_i & -y'_i \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Solving for homographies

$$\underset{\substack{\mathbf{A}\\ 2n \times 9}}{\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & & \vdots & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix}} \underset{\substack{\mathbf{h}\\ 9}}{\begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix}} = \underset{\substack{\mathbf{0}\\ 2n}}{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}}$$

Defines a least squares problem: $\text{minimize } \|\mathbf{Ah} - \mathbf{0}\|^2$

- Since **h** is only defined up to scale, solve for unit vector $\hat{\mathbf{h}}$
- Solution: $\hat{\mathbf{h}}$ = eigenvector of $\mathbf{A}^T\mathbf{A}$ with smallest eigenvalue
- Works with 4 or more points