

CS5670: Computer Vision

Single-View Modeling



Single-View Modeling

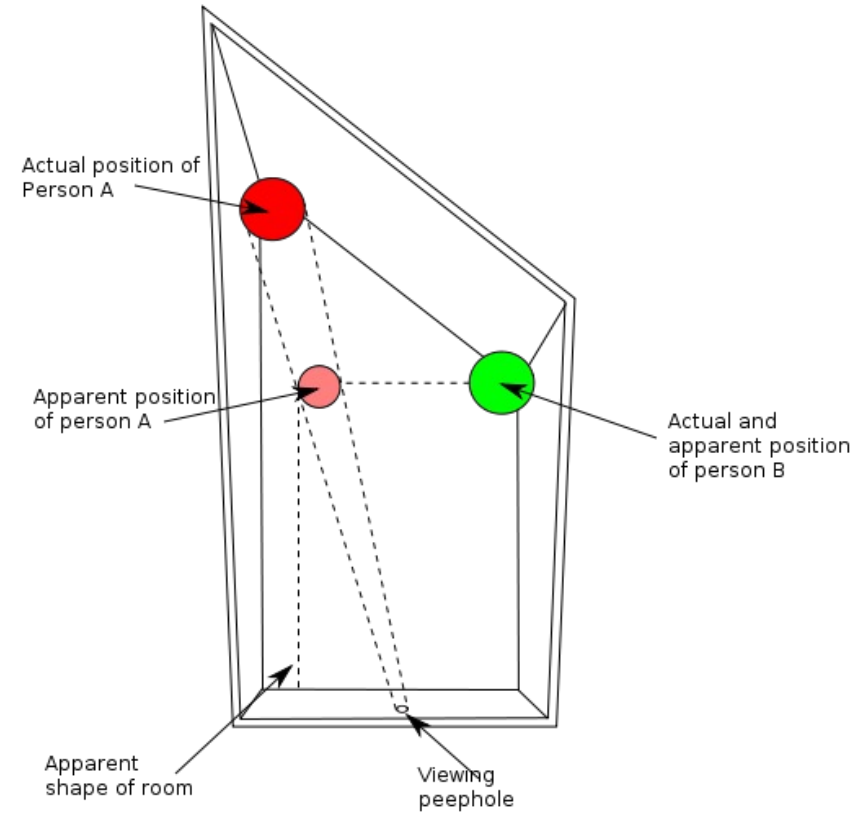


[Ames Room](#)

- Readings

- Mundy and Zisserman. *Geometric Invariance in Computer Vision*, Appendix: Projective Geometry for Machine Vision, MIT Press, 1992, (**read 23.1-23.5, 23.10**)
 - available online: <http://www.cs.cmu.edu/~ph/869/papers/zisser-mundy.pdf>

Ames Room



Announcements

- Project 3: Autostitch (Panorama Stitching)
 - Due on Friday, March 17, by 8pm
 - To be done in groups of 2
 - If you need help finding a team member, let us know

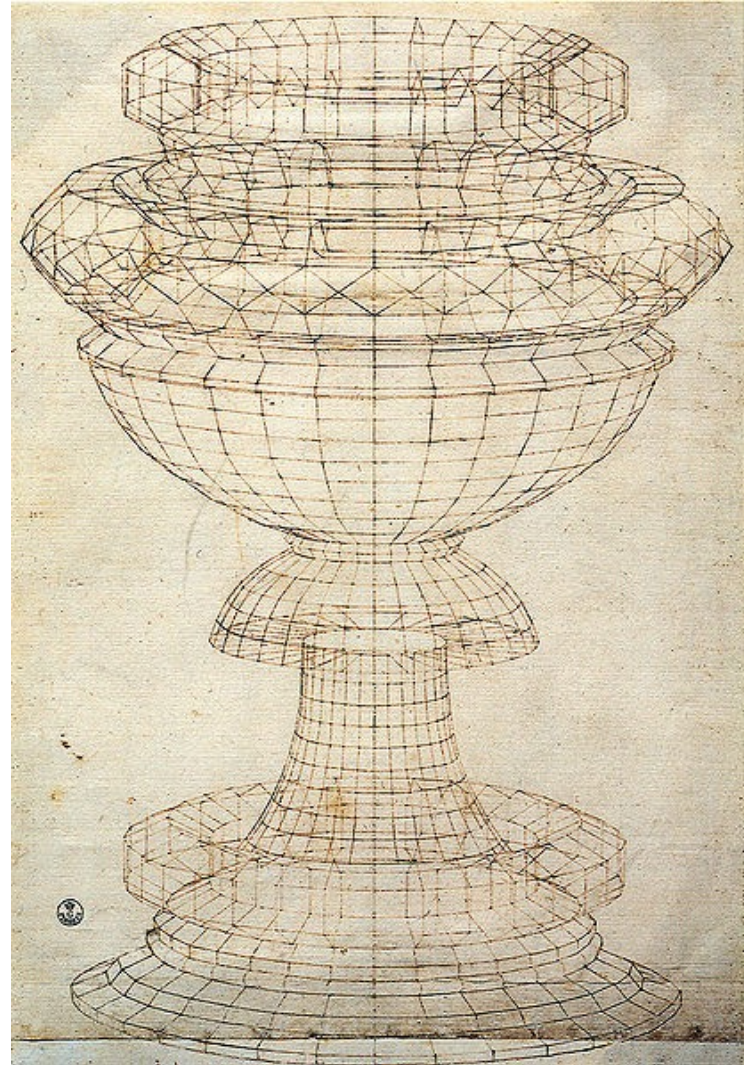
Forced perspective in film



How Lord of the Rings used forced perspective shots with a moving camera
https://www.youtube.com/watch?v=QWMFpxkGO_s

Projective geometry—what's it good for?

- Uses of projective geometry
 - Drawing
 - Measurements
 - Mathematics for projection
 - Undistorting images
 - Camera pose estimation
 - **Object recognition**

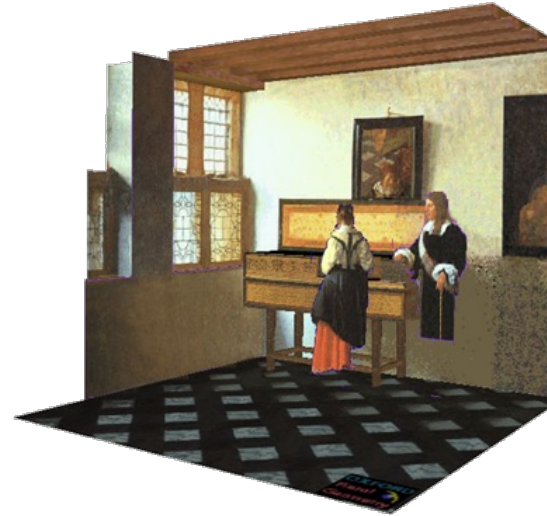


[Paolo Uccello](#)

Applications of projective geometry



Vermeer's *Music Lesson*



Reconstructions by Criminisi et al.

Making measurements in images

WARBY PARKER

Measure your pupillary distance (PD)

Your PD is the distance between your pupils. To measure it, follow the instructions below — once you submit your photo, our team of experts will determine your PD and email you once we've applied it to your order.

1



**Wearing glasses?
Take 'em off before you get started.**

2



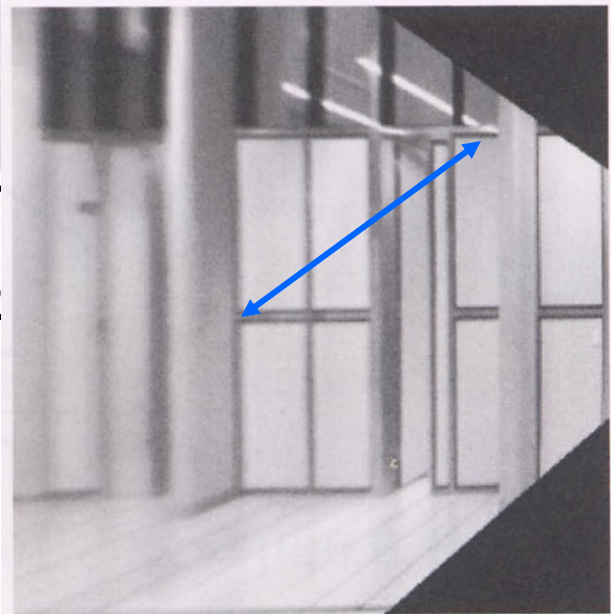
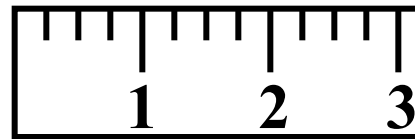
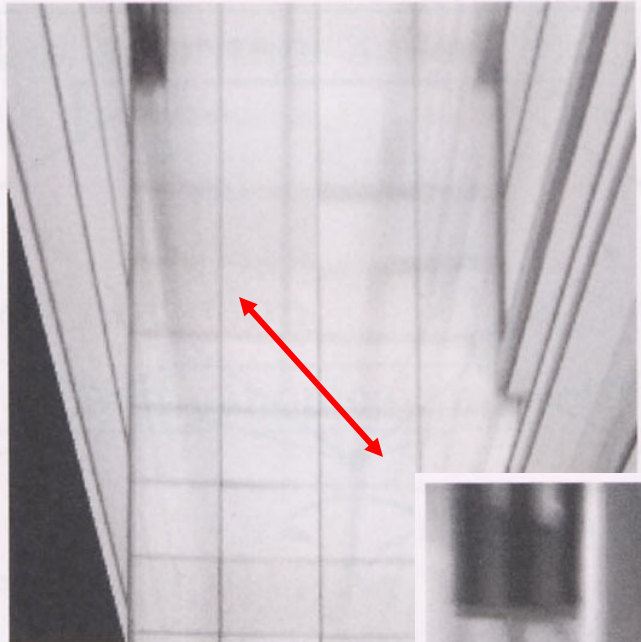
**Hold up any card with a magnetic
strip (we use this for scale).**

3



**Look straight ahead
and snap a photo.**

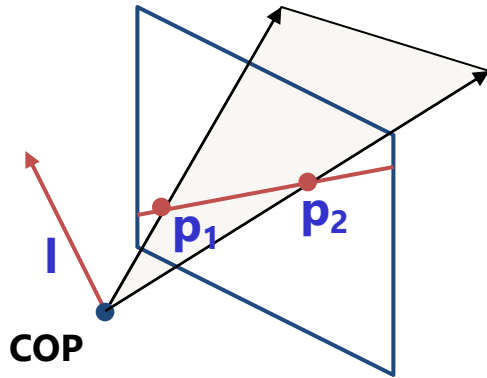
Measurements on planes



Approach: unwarp then measure

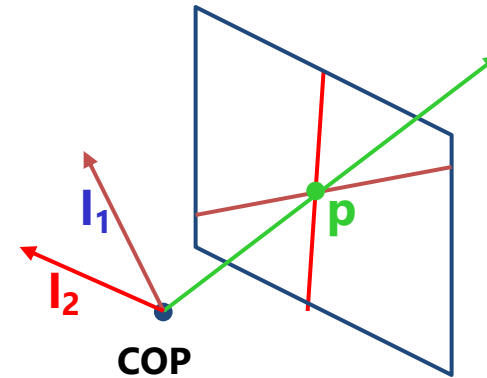
Point and line duality

- A line \mathbf{l} is a homogeneous 3-vector
- It is \perp to every point (ray) \mathbf{p} on the line: $\mathbf{l} \cdot \mathbf{p} = 0$



What is the line \mathbf{l} spanned by points \mathbf{p}_1 and \mathbf{p}_2 ?

- \mathbf{l} is \perp to \mathbf{p}_1 and $\mathbf{p}_2 \Rightarrow \mathbf{l} = \mathbf{p}_1 \times \mathbf{p}_2$
- \mathbf{l} can be interpreted as a *plane normal*

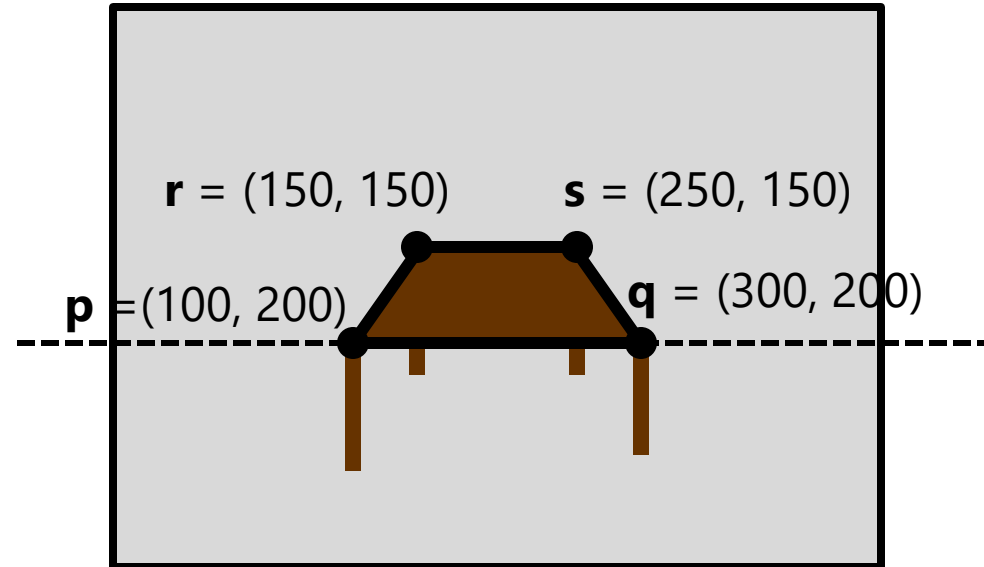


What is the intersection of two lines \mathbf{l}_1 and \mathbf{l}_2 ?

- \mathbf{p} is \perp to \mathbf{l}_1 and $\mathbf{l}_2 \Rightarrow \mathbf{p} = \mathbf{l}_1 \times \mathbf{l}_2$

Points and lines are *dual* in projective space

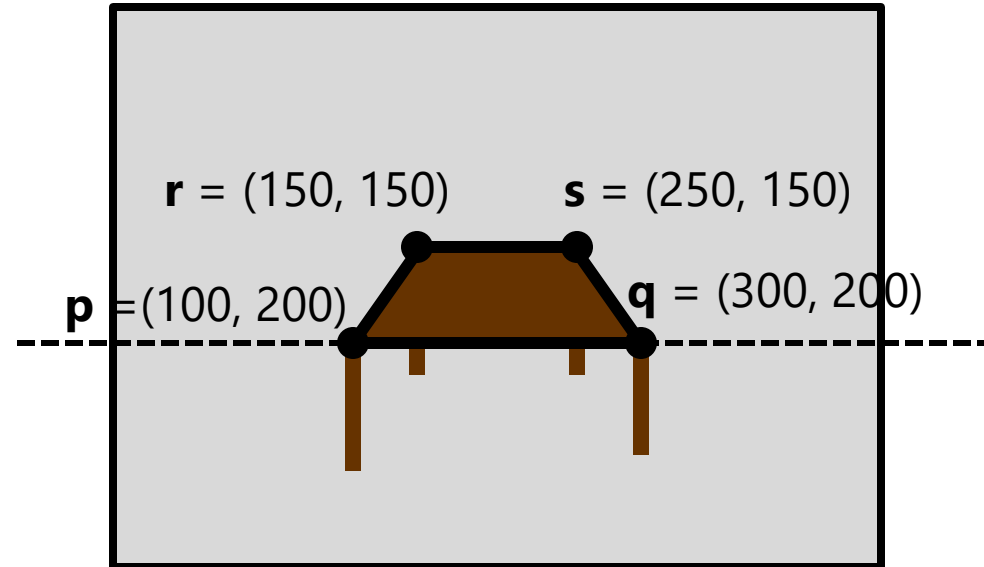
Example



What is the line passing through points \mathbf{p} and \mathbf{q} ?

$$\mathbf{p} \times \mathbf{q}$$

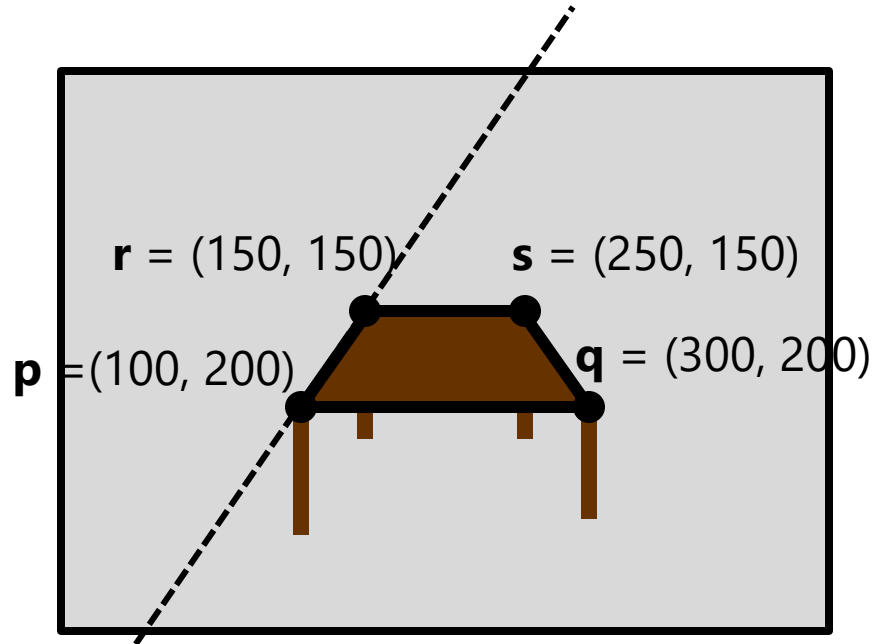
Example



How do we interpret the line $\ell = \begin{bmatrix} 0 \\ 1 \\ -200 \end{bmatrix}$?

Answer: the set of points (x, y) such that $\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0$, $y - 200 = 0$
i.e.,

Example

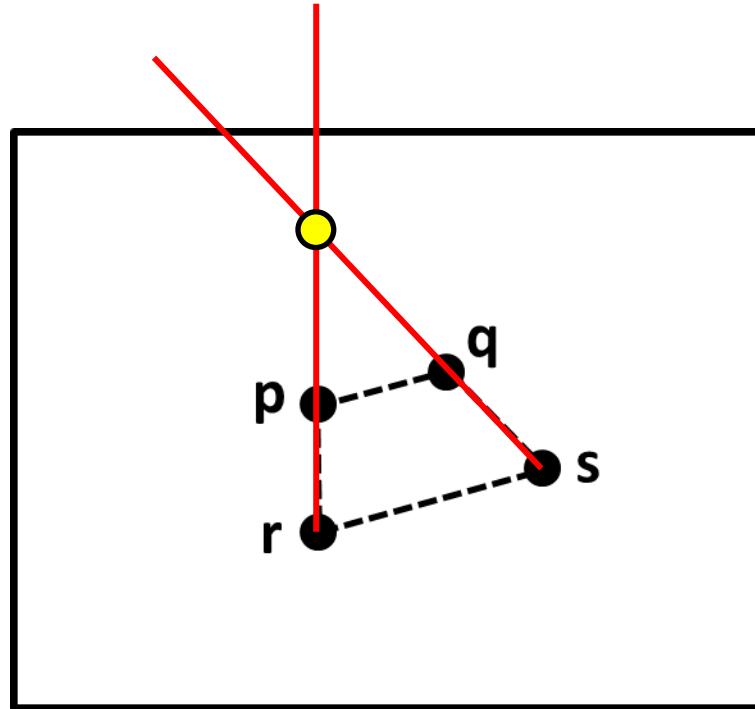


What is the line passing through points \mathbf{p} and \mathbf{r} ?

$$\mathbf{p} \times \mathbf{r} = \begin{bmatrix} 100 \\ 200 \\ 1 \end{bmatrix} \times \begin{bmatrix} 150 \\ 150 \\ 1 \end{bmatrix} = \begin{bmatrix} 200 \cdot 1 - 150 \cdot 1 \\ 150 \cdot 1 - 100 \cdot 1 \\ 100 \cdot 150 - 150 \cdot 200 \end{bmatrix} = \begin{bmatrix} 50 \\ 50 \\ -15000 \end{bmatrix} \sim \begin{bmatrix} 1 \\ 1 \\ -300 \end{bmatrix}$$

i.e., all points (x, y) such that $x + y = 300$

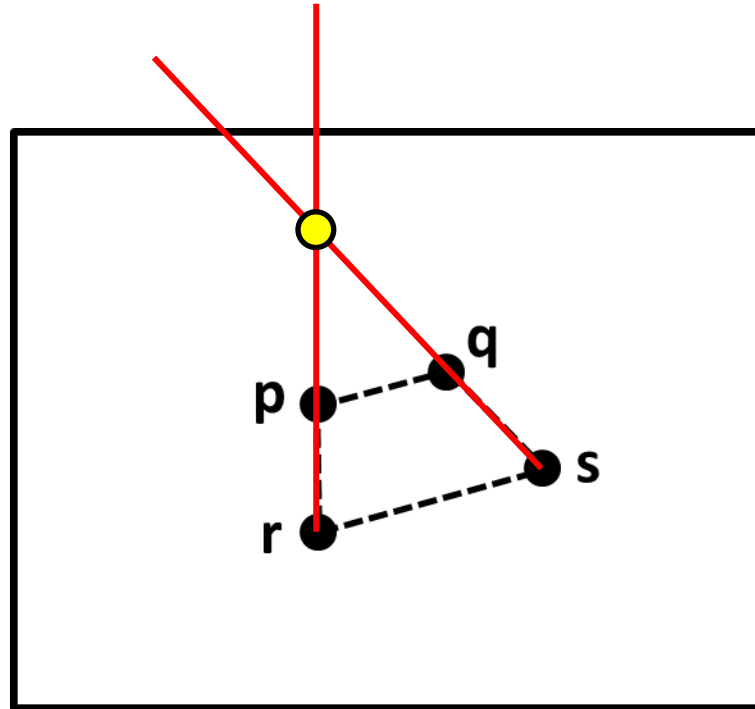
Question time



Consider the above image, with four points \mathbf{p} , \mathbf{q} , \mathbf{r} , \mathbf{s} , labeled (assume these are 2D homogeneous points).

What is a simple expression for the point of intersection between the line through \mathbf{p} and \mathbf{r} and the line through \mathbf{q} and \mathbf{s} ?

Question time

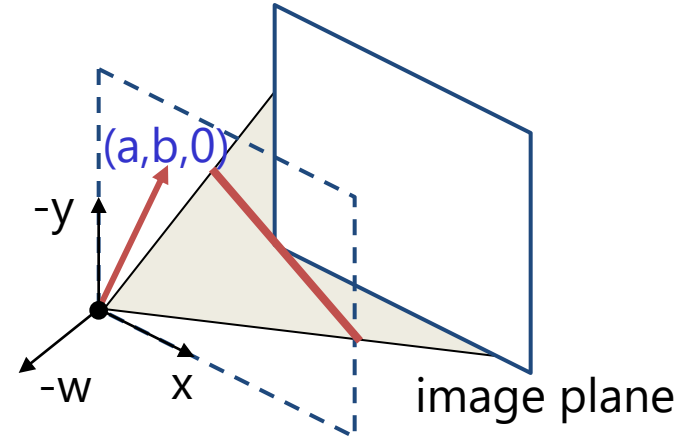
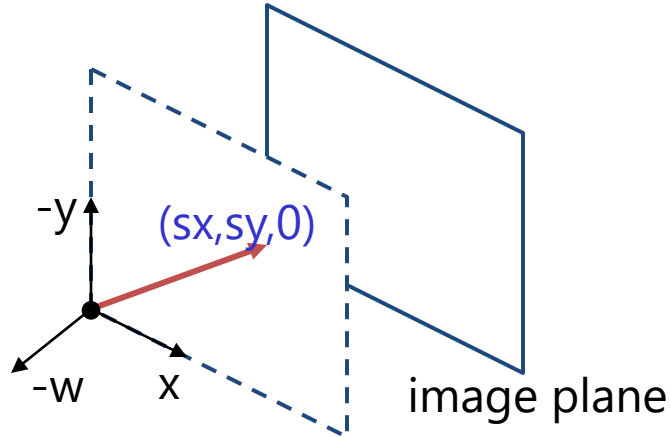


Consider the above image, with four points \mathbf{p} , \mathbf{q} , \mathbf{r} , \mathbf{s} , labeled (assume these are 2D homogeneous points).

What is a simple expression for the point of intersection between the line through \mathbf{p} and \mathbf{r} and the line through \mathbf{q} and \mathbf{s} ?

Answer: $(\mathbf{p} \times \mathbf{r}) \times (\mathbf{q} \times \mathbf{s})$

Ideal points and lines



- Ideal point (“point at infinity”)
 - $p \cong (x, y, 0)$ – parallel to image plane
 - It has infinite image coordinates
- Ideal line
 - $l \cong (a, b, 0)$ – parallel to image plane
 - Corresponds to a line in the image (finite coordinates)
 - goes through image origin (*principal point*)

3D projective geometry

- These concepts generalize naturally to 3D
 - Homogeneous coordinates
 - Projective 3D points have four coords: $\mathbf{P} = (X, Y, Z, W)$
 - Duality
 - A plane \mathbf{N} is also represented by a 4-vector
 - Points and planes are dual in 3D: $\mathbf{N} \mathbf{P} = 0$
 - Three points define a plane, three planes define a point

3D to 2D: perspective projection

Projection:

$$\mathbf{p} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{\Pi P}$$

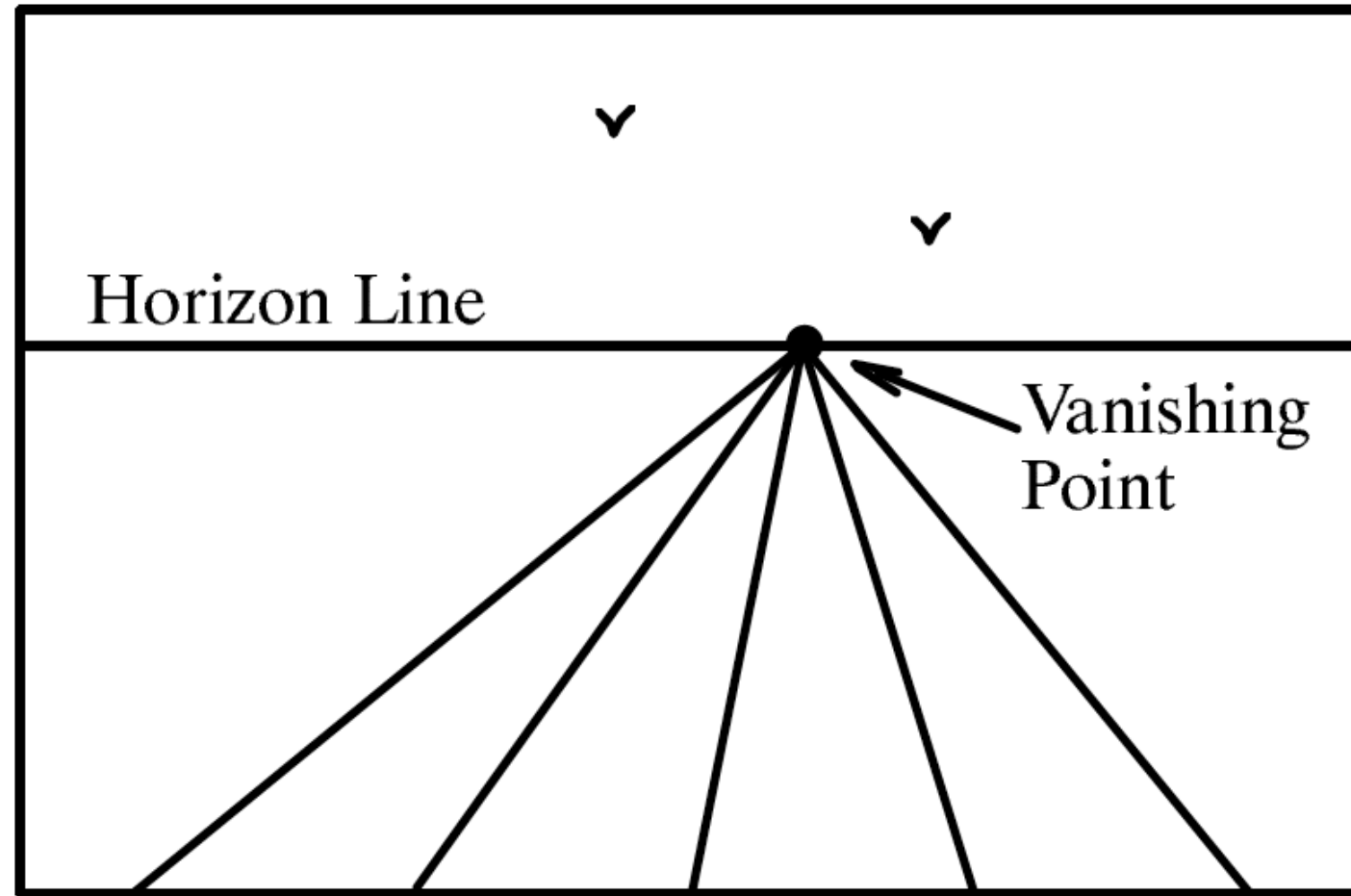
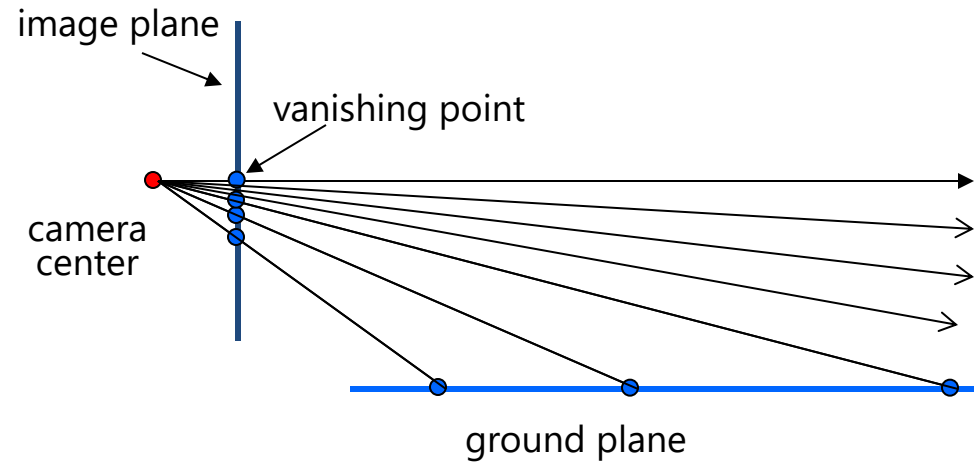


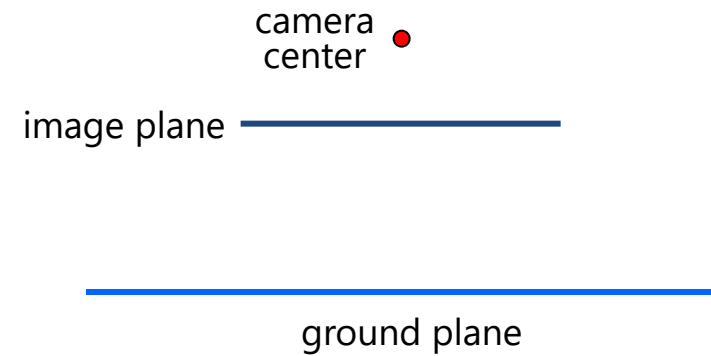
Figure 23.4

A perspective view of a set of parallel lines in the plane. All of the lines converge to a single vanishing point.

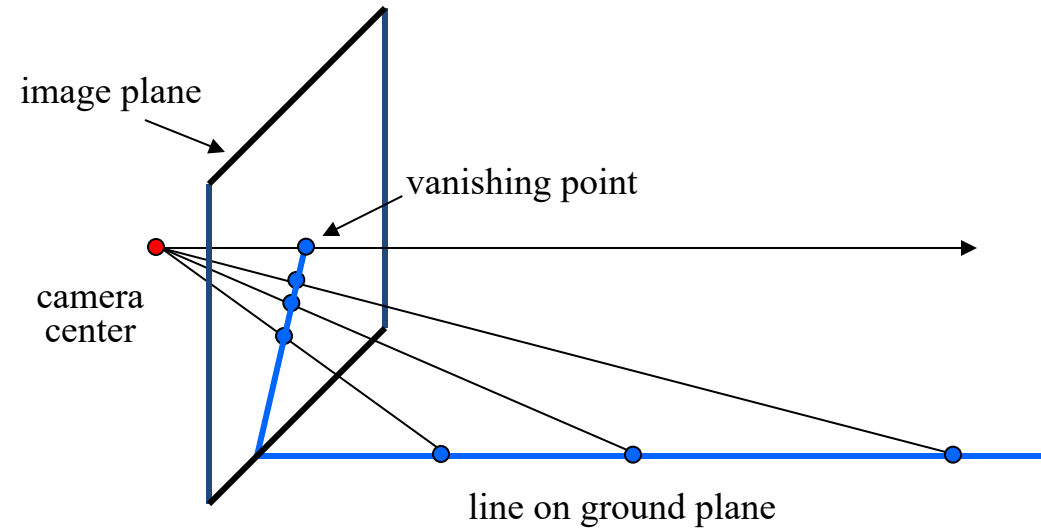
Vanishing points (1D)



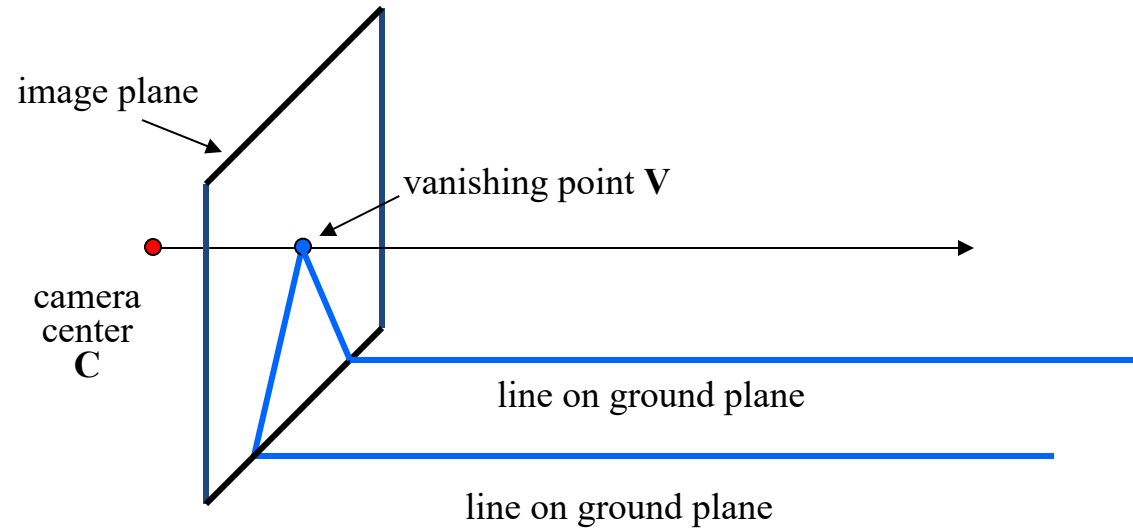
- Vanishing point
 - projection of a point at infinity
 - can often (but not always) project to a finite point in the image



Vanishing points (2D)

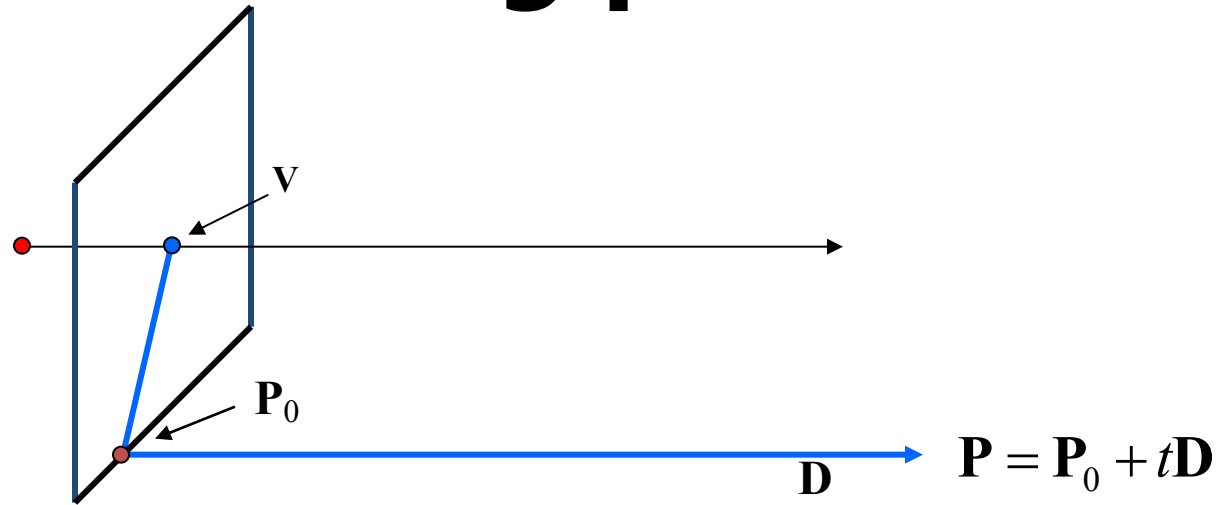


Vanishing points



- Properties
 - Any two parallel lines (in 3D) have the same vanishing point \mathbf{v}
 - The ray from \mathbf{C} through \mathbf{v} is parallel to the lines
 - An image may have more than one vanishing point
 - in fact, every image point is a potential vanishing point

Computing vanishing points



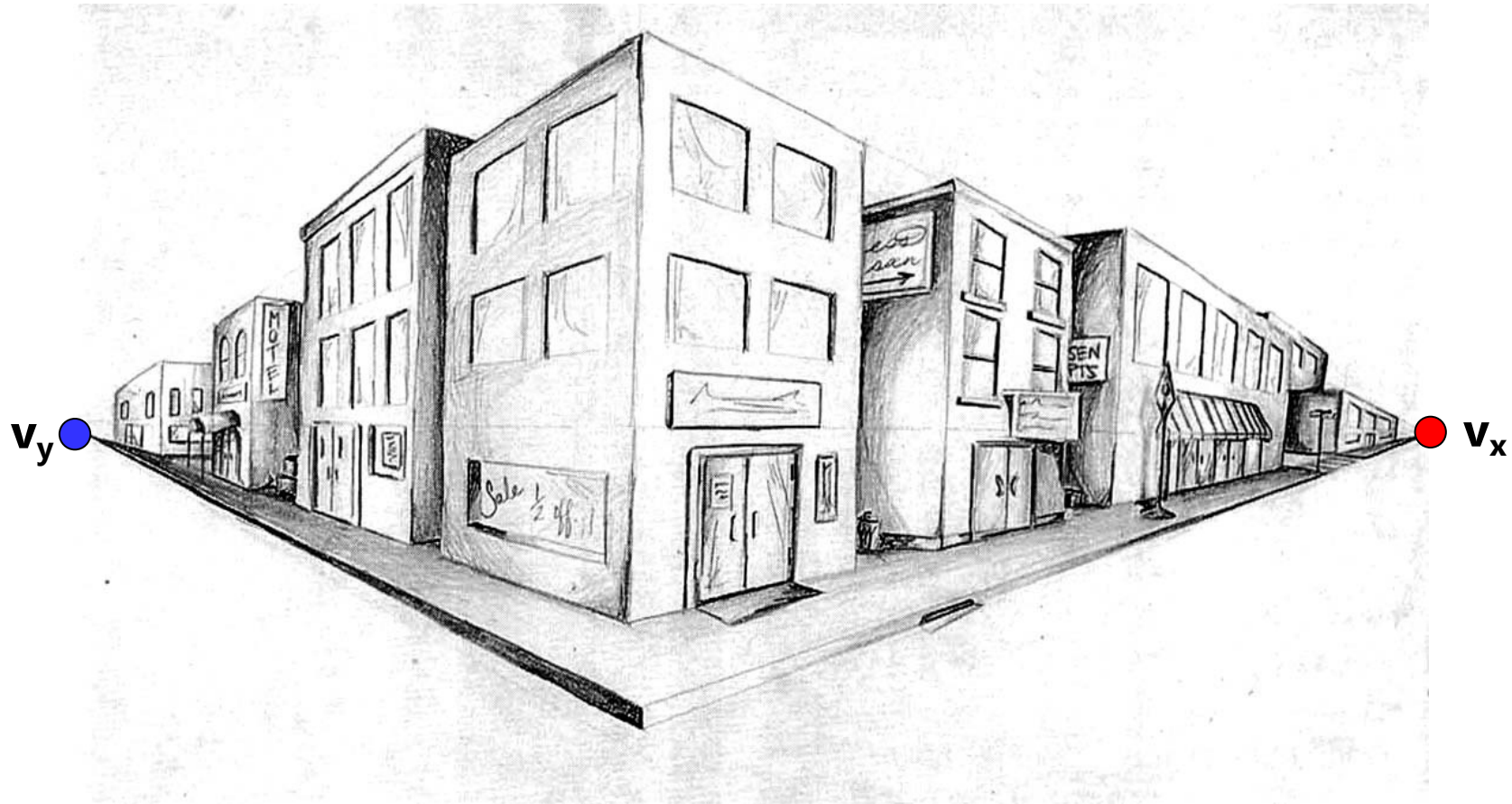
$$\mathbf{P}_t = \begin{bmatrix} P_X + tD_X \\ P_Y + tD_Y \\ P_Z + tD_Z \\ 1 \end{bmatrix} \cong \begin{bmatrix} P_X / t + D_X \\ P_Y / t + D_Y \\ P_Z / t + D_Z \\ 1/t \end{bmatrix}$$

- Properties $\mathbf{v} = \mathbf{IIP}_\infty$
 - \mathbf{P}_∞ is a point at *infinity*, \mathbf{v} is its projection
 - Depends only on line *direction*
 - Parallel lines $\mathbf{P}_0 + t\mathbf{D}$, $\mathbf{P}_1 + t\mathbf{D}$ intersect at \mathbf{P}_∞

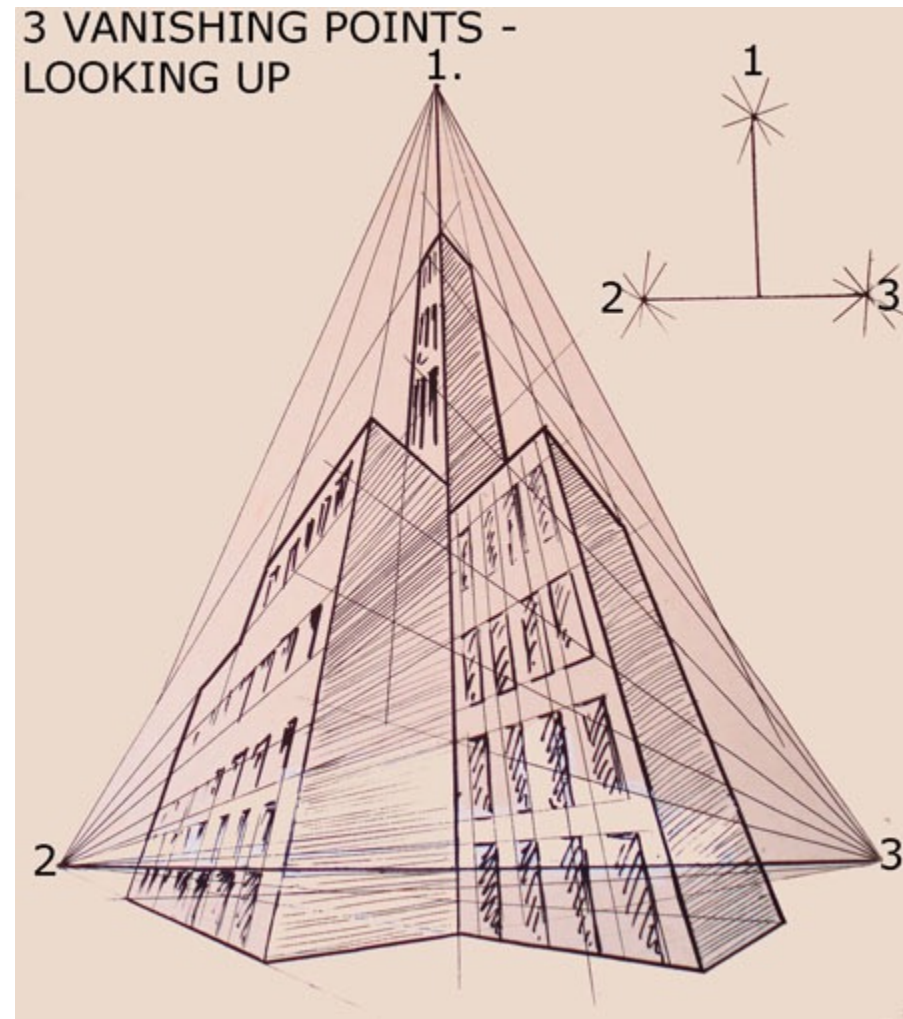
One-point perspective



Two-point perspective

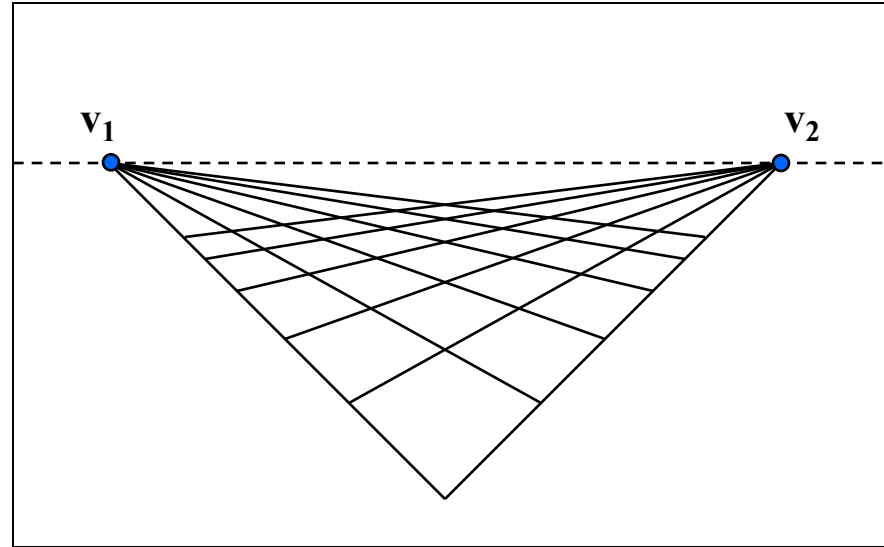


Three-point perspective



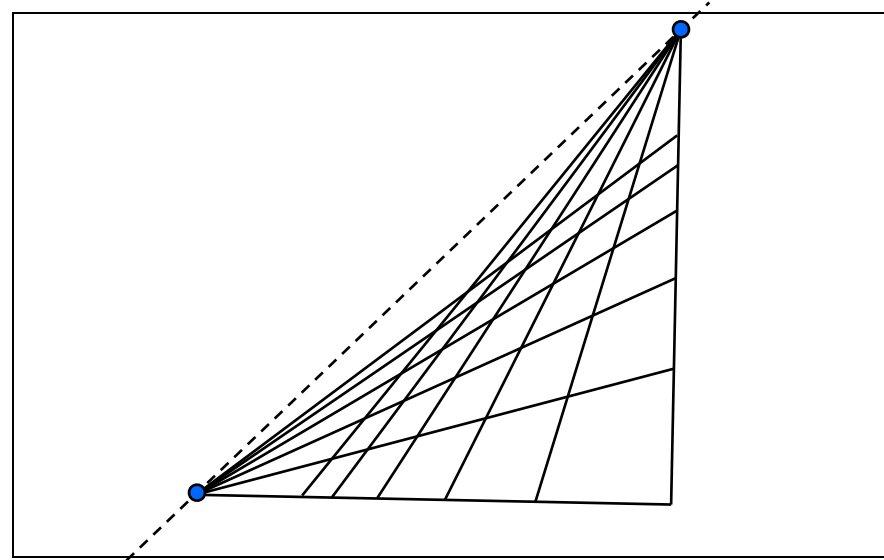
Questions?

Vanishing lines



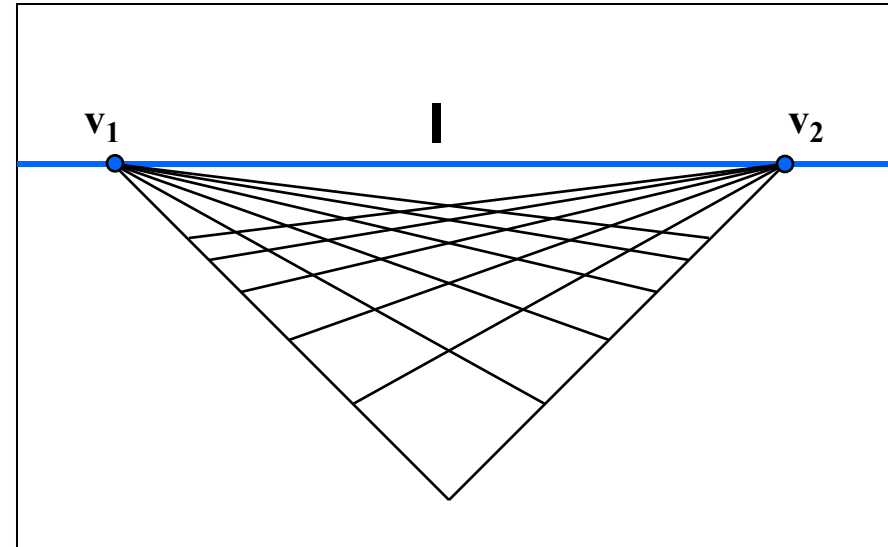
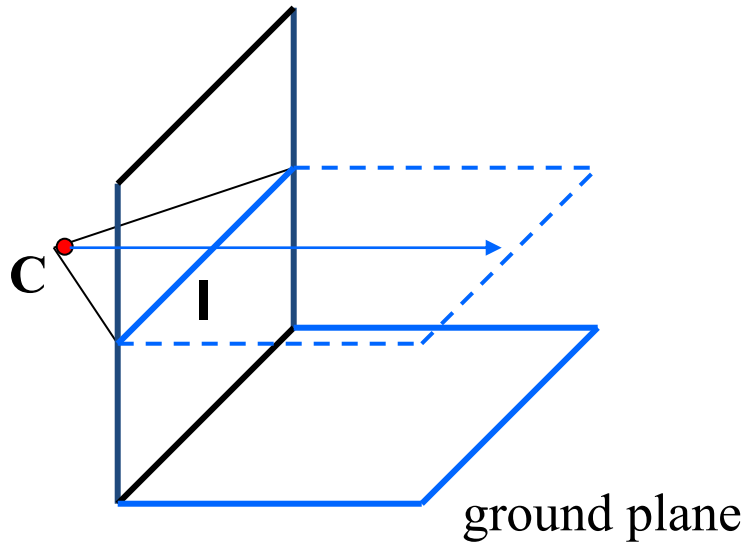
- Multiple Vanishing Points
 - Any set of parallel lines on the plane define a vanishing point
 - The union of all of these vanishing points is the *horizon line*
 - also called *vanishing line*
 - Note that different planes (can) define different vanishing

Vanishing lines



- Multiple Vanishing Points
 - Any set of parallel lines on the plane define a vanishing point
 - The union of all of these vanishing points is the *horizon line*
 - also called *vanishing line*
 - Note that different planes (can) define different vanishing

Computing vanishing lines



- **Properties**

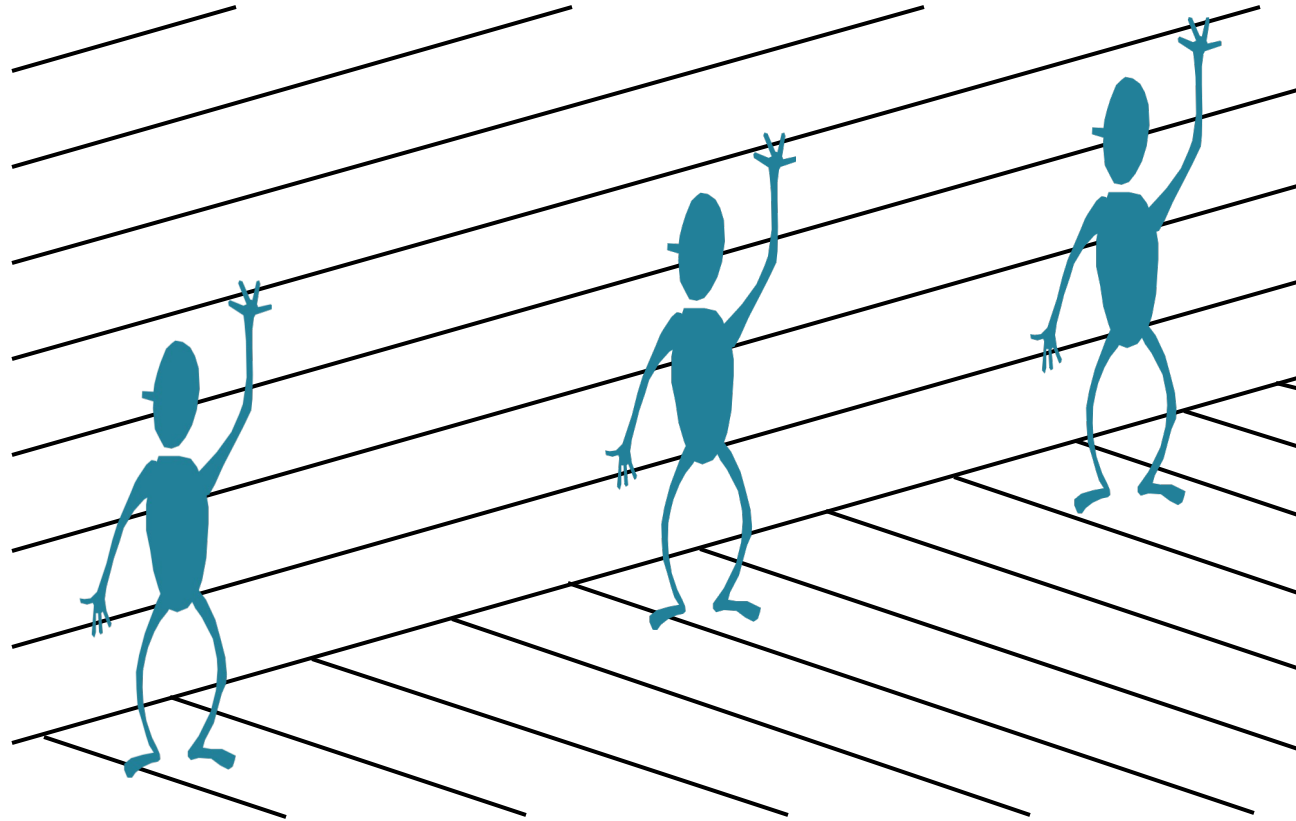
- **I** is intersection of horizontal plane through **C** with image plane
- Compute **I** from two sets of parallel lines on ground plane
- All points at same height as **C** project to **I**
 - points higher than **C** project above **I**
- Provides way of comparing height of objects in the scene



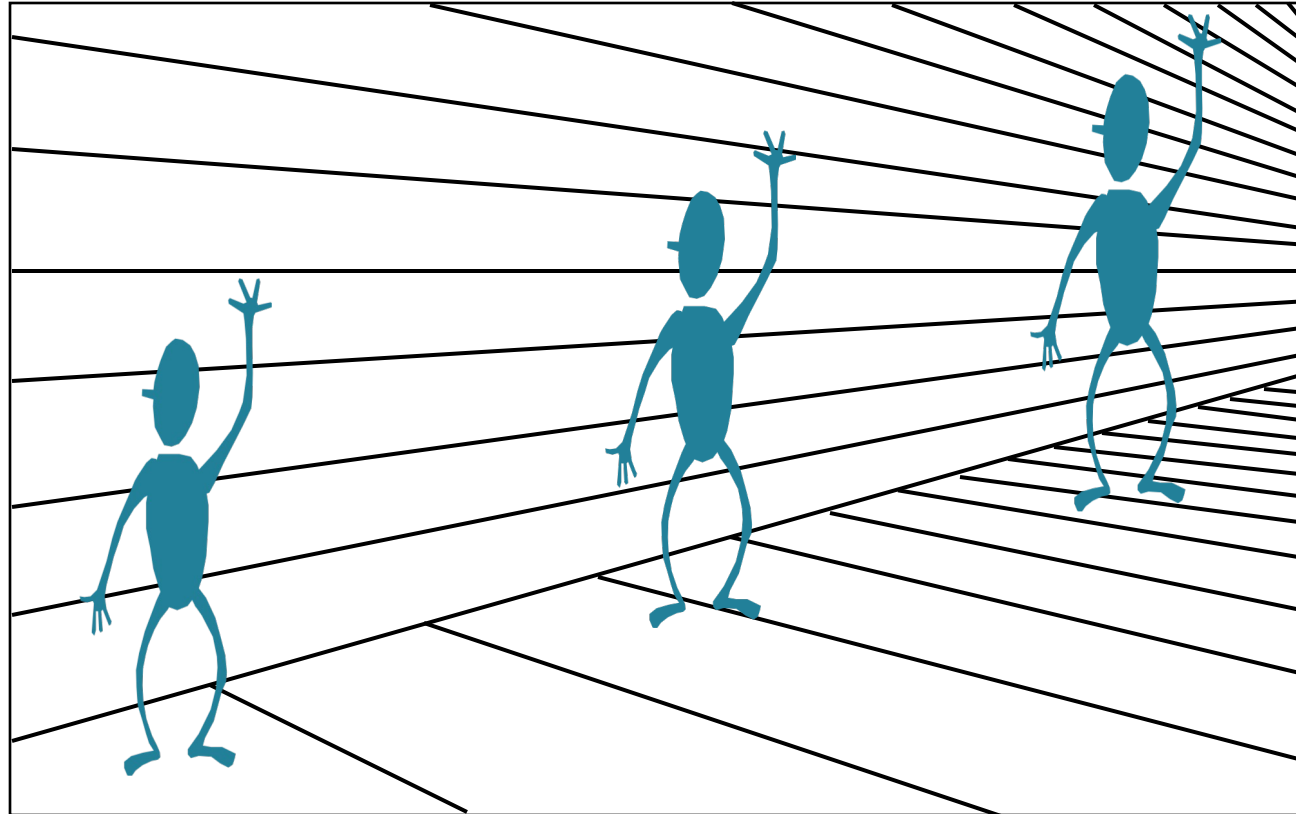
Lots of fun with vanishing points



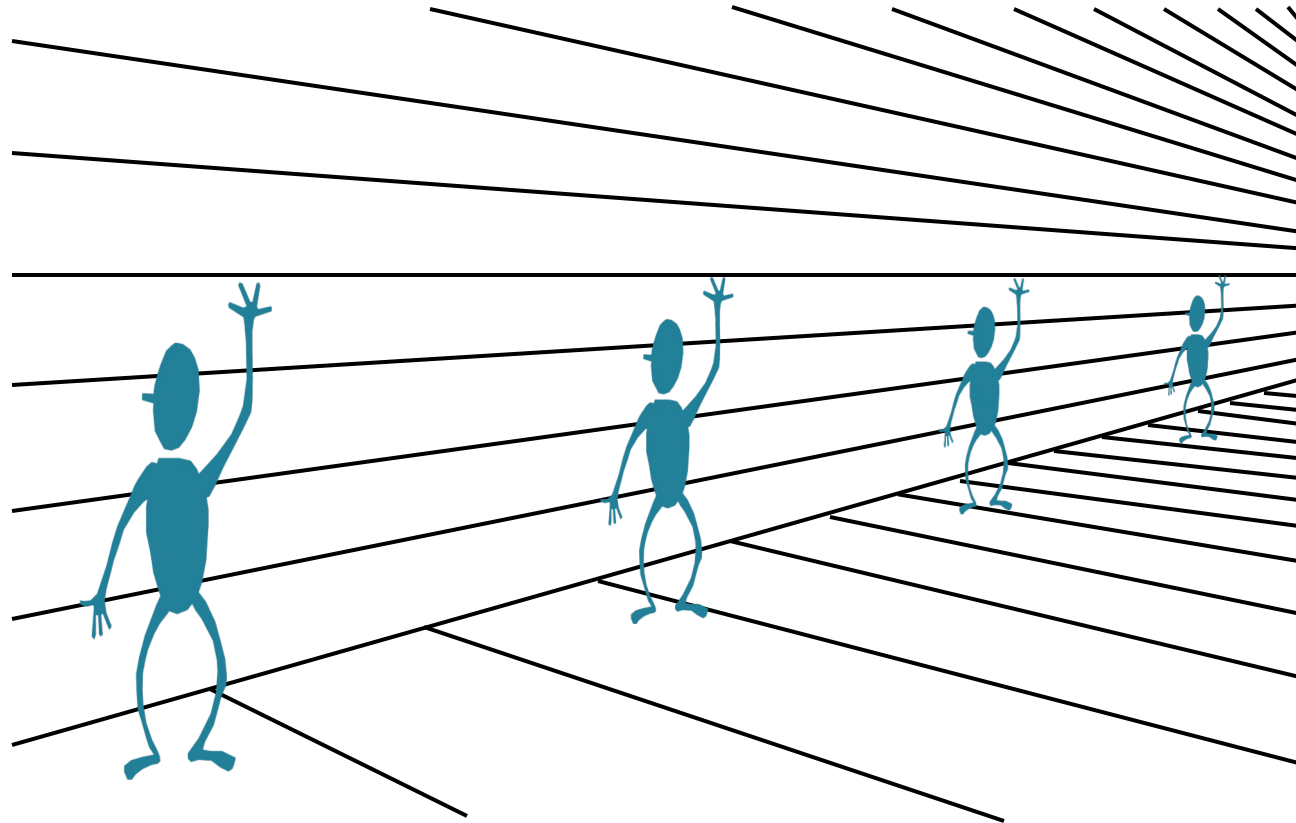
Perspective cues



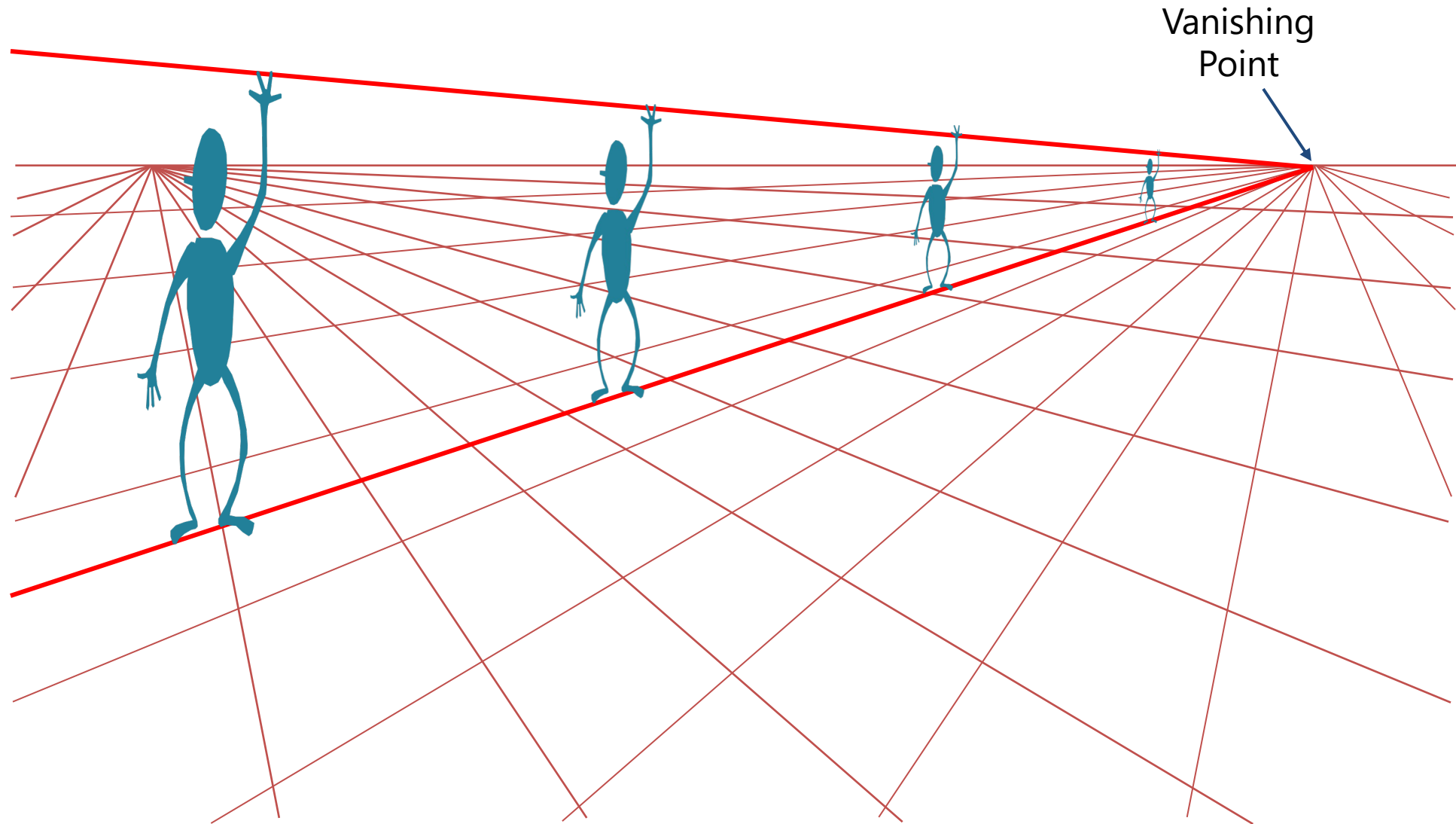
Perspective cues



Perspective cues

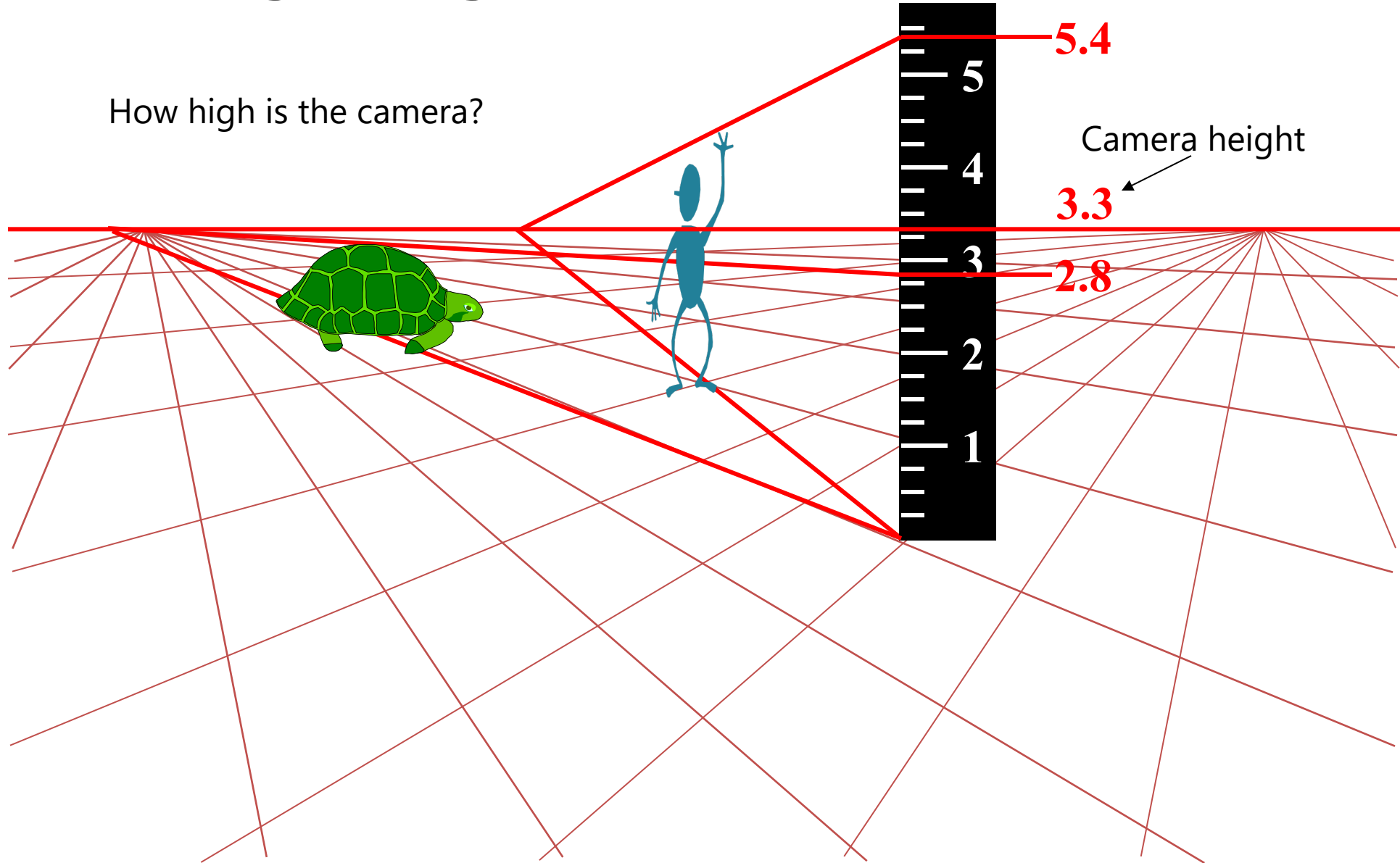


Comparing heights

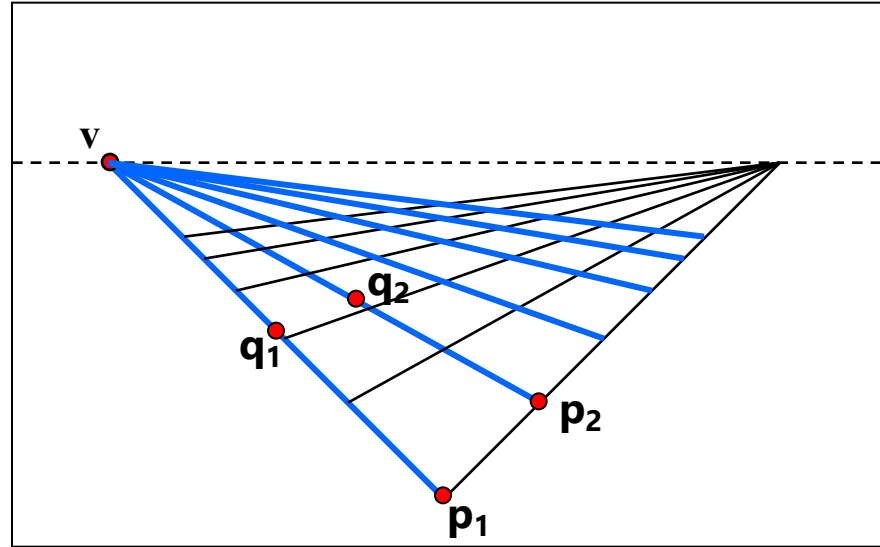


Measuring height

How high is the camera?



Computing vanishing points (from lines)



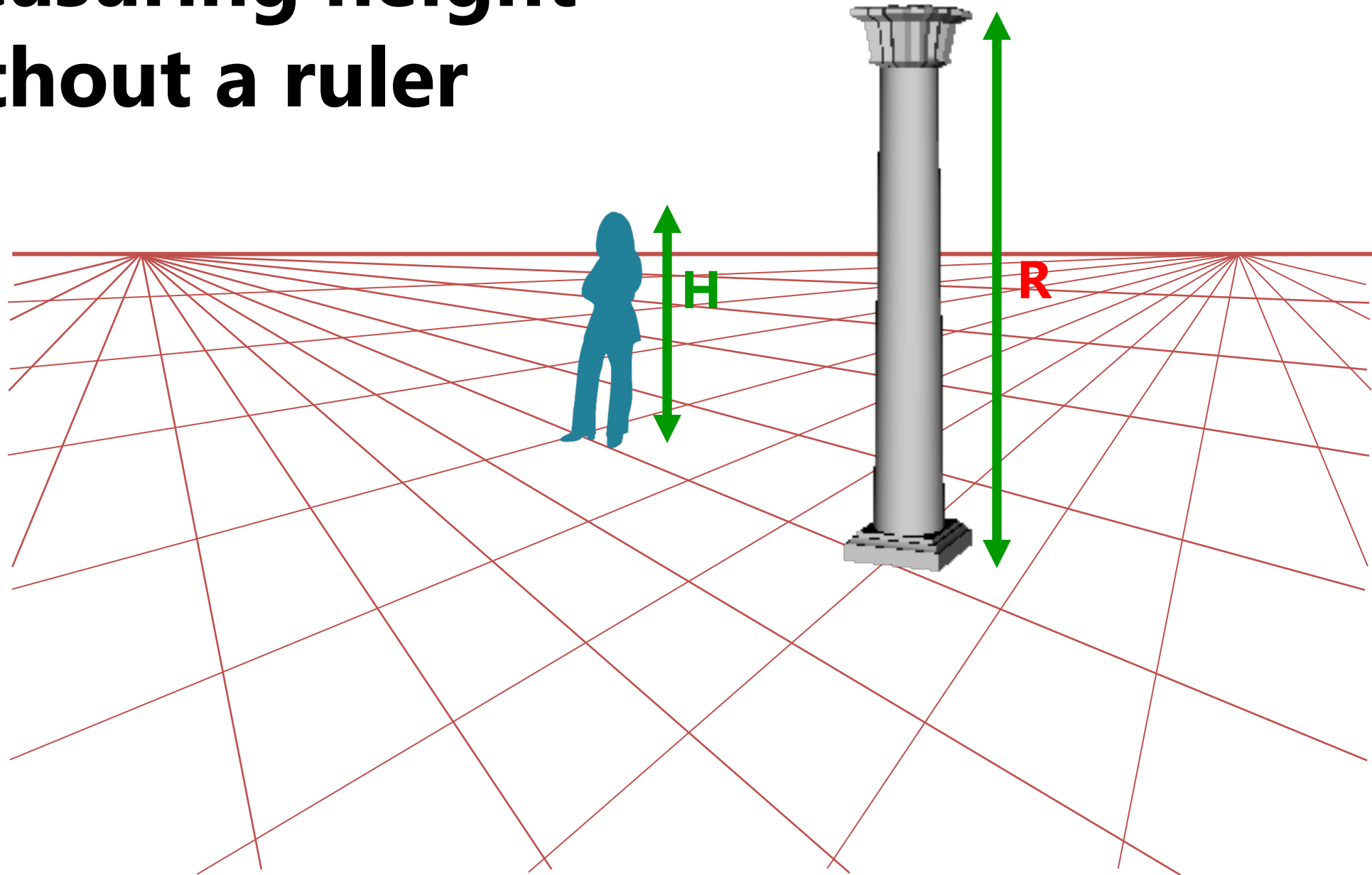
Intersect p_1q_1 with p_2q_2

$$v = (p_1 \times q_1) \times (p_2 \times q_2)$$

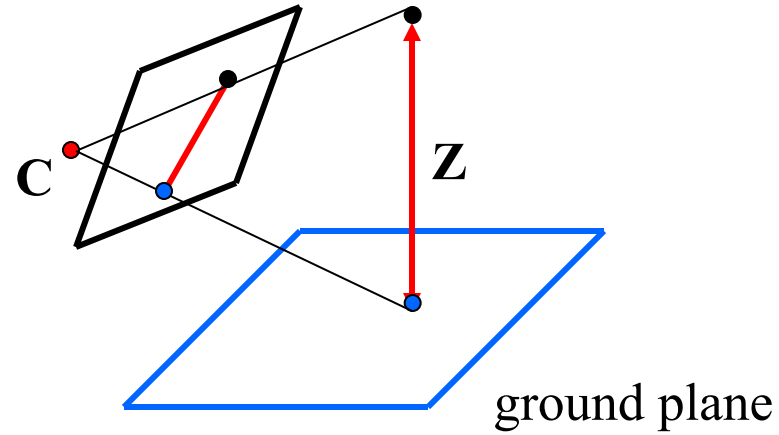
Least squares version

- Better to use more than two lines and compute the “closest” point of intersection
- See notes by [Bob Collins](#) for one good way of doing this:
 - <http://www-2.cs.cmu.edu/~ph/869/www/notes/vanishing.txt>

Measuring height without a ruler



Measuring height without a ruler



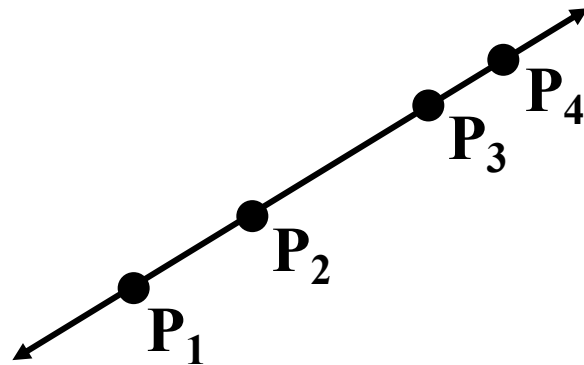
Compute Z from image measurements

- 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



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

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

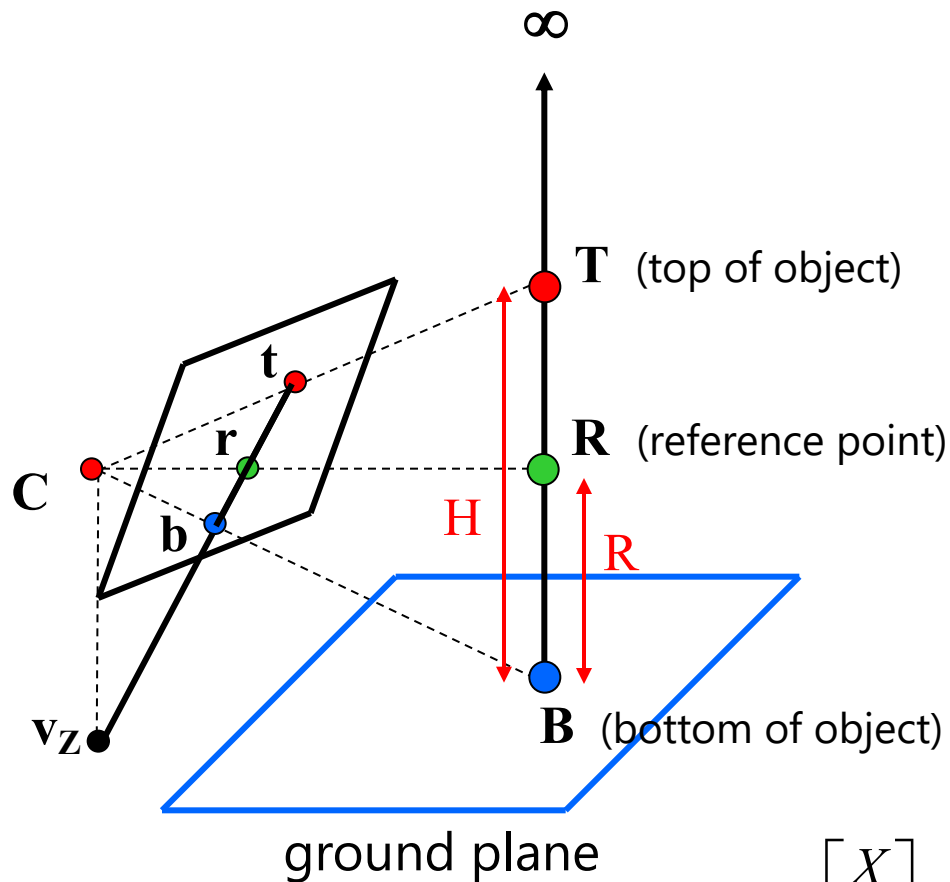
Can permute the point ordering

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

This is the fundamental invariant of projective geometry

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

Measuring height



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

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

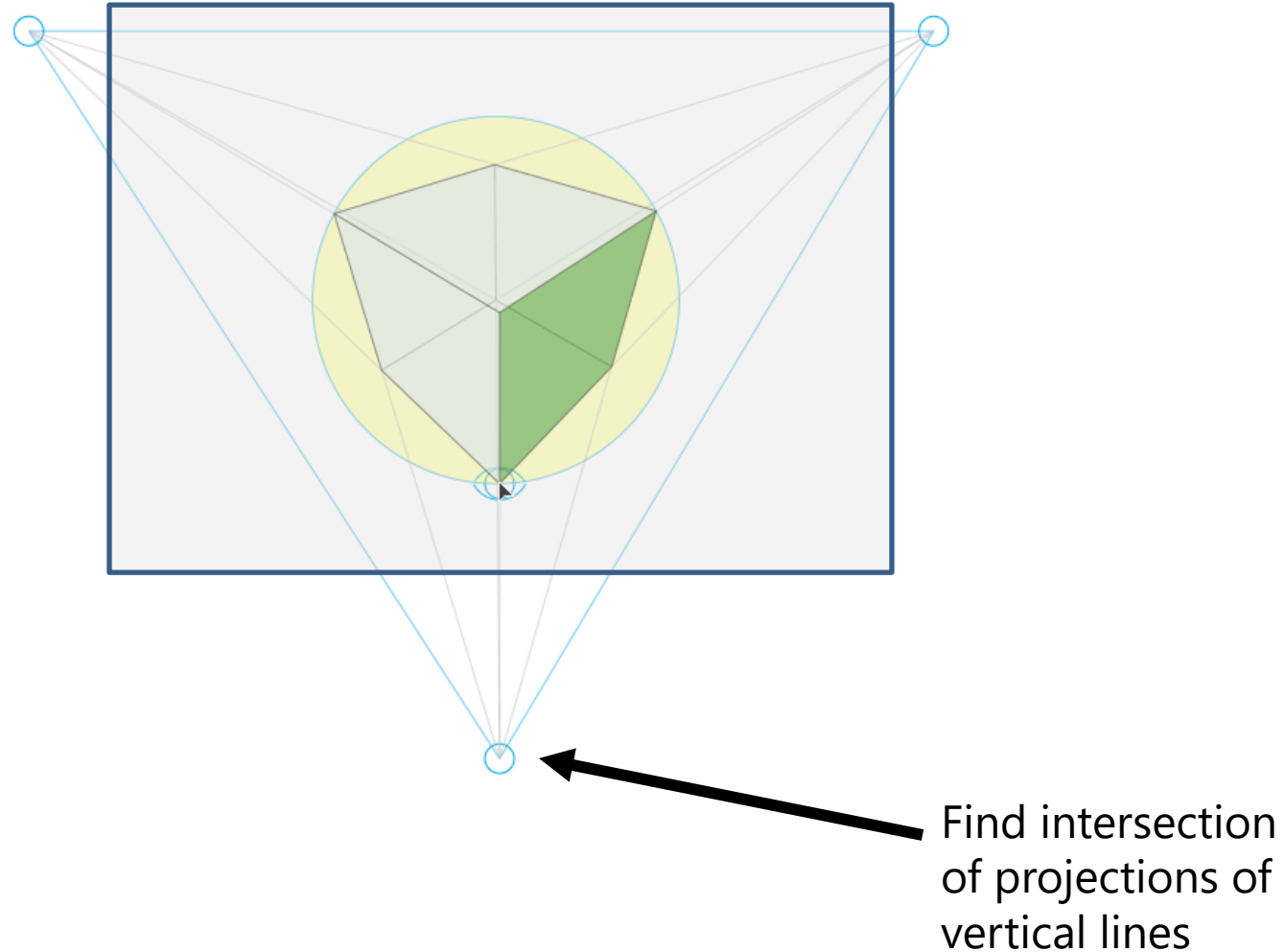
$$\frac{\|\mathbf{T} - \mathbf{B}\| \|\infty - \mathbf{R}\|}{\|\mathbf{R} - \mathbf{B}\| \|\infty - \mathbf{T}\|} = \frac{H}{R}$$

scene cross ratio

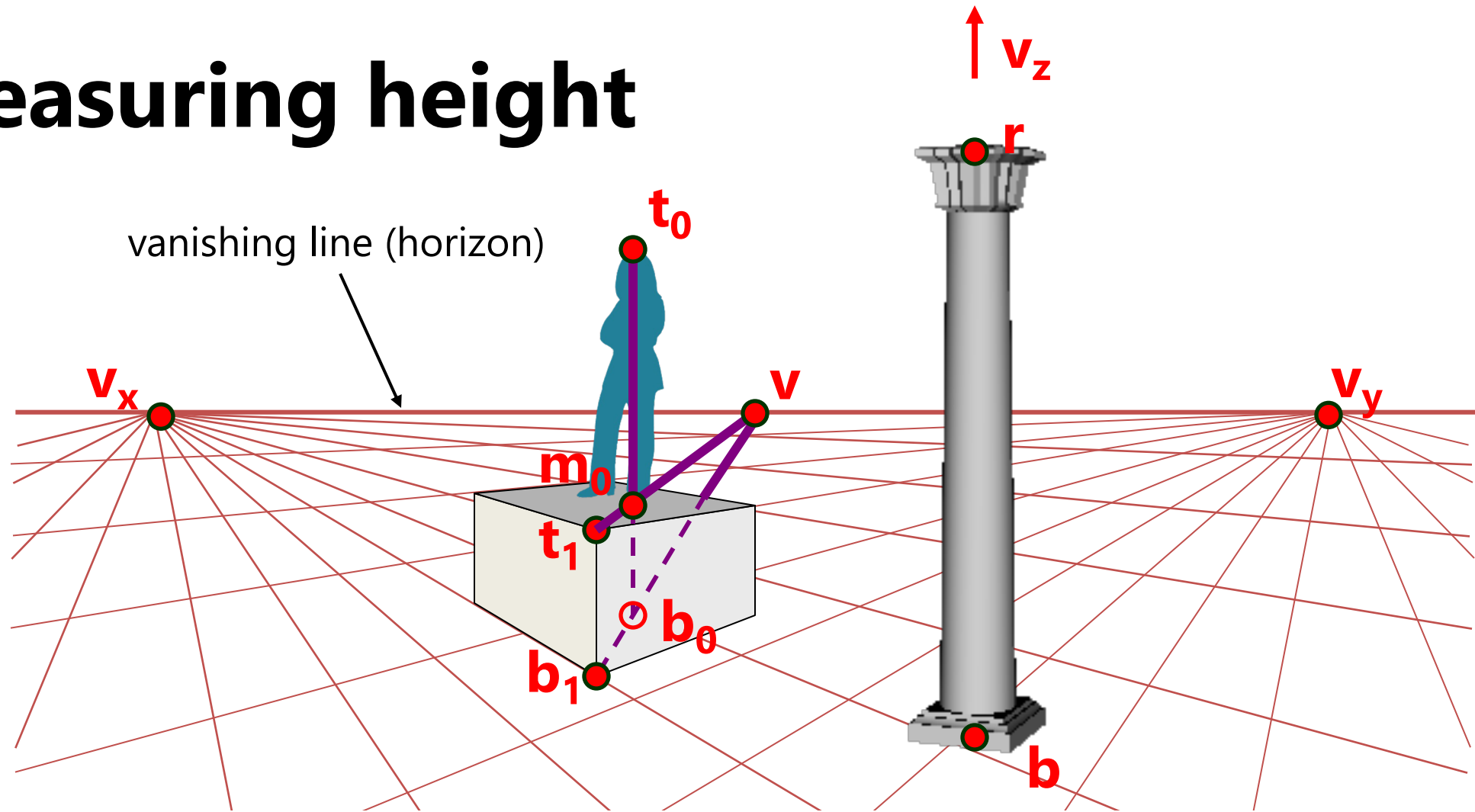
$$\frac{\|\mathbf{t} - \mathbf{b}\| \|\mathbf{v}_Z - \mathbf{r}\|}{\|\mathbf{r} - \mathbf{b}\| \|\mathbf{v}_Z - \mathbf{t}\|} = \frac{H}{R}$$

image cross ratio

Finding the vertical (z) vanishing point



Measuring height



What if the point on the ground plane b_0 is not known?

- Here the person is standing on the box, height of box is known
- Use one side of the box to help find b_0 as shown above

3D modeling from a photograph



St. Jerome in his Study, H. Steenwick

Bringing Pictorial Space to Life: Computer Techniques for the Analysis of Paintings.
Antonio Criminisi, Martin Kemp, Andrew Zisserman. 2002.

3D modeling from a photograph

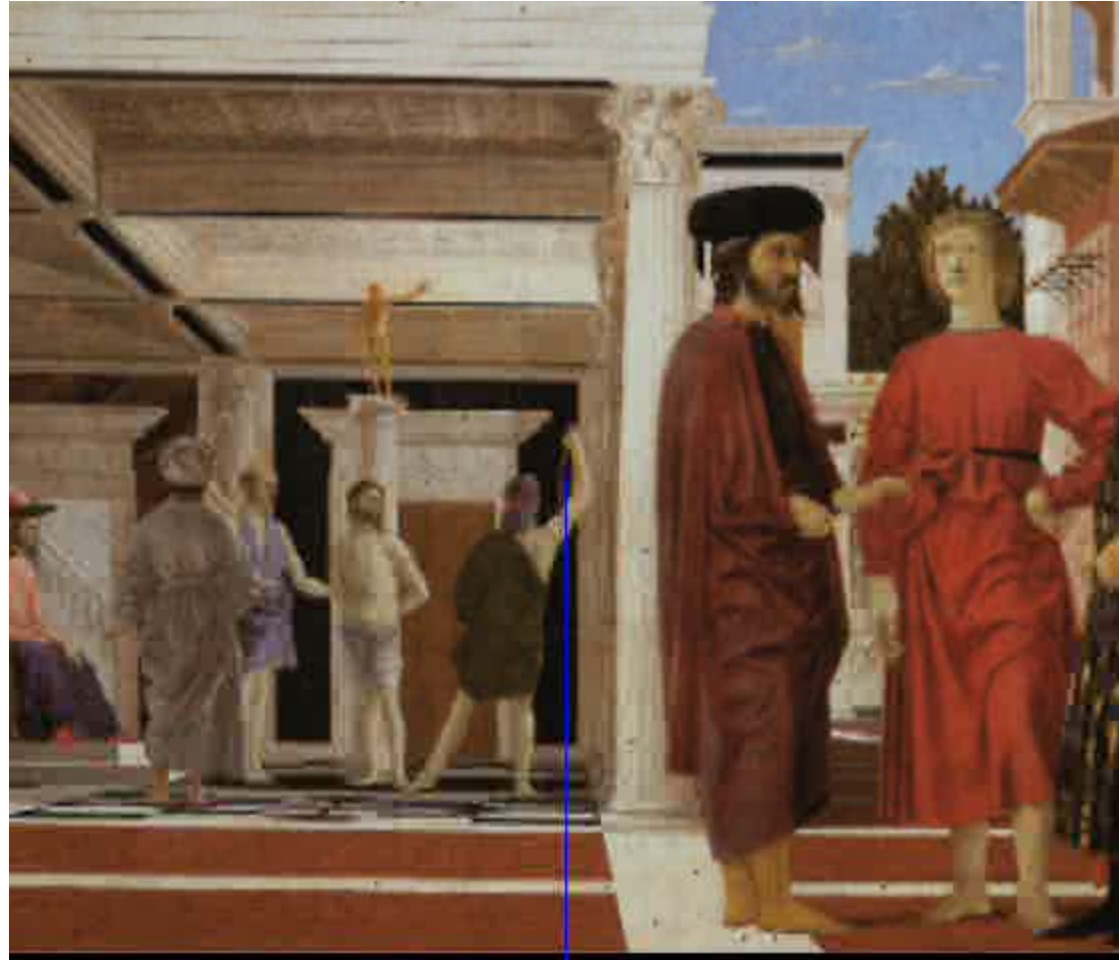


3D modeling from a photograph



Flagellation, Piero della Francesca

3D modeling from a photograph



video by Antonio Criminisi

3D modeling from a photograph



Flagellation. Piero della Francesca. c1453.

Related problem: camera calibration

- Goal: estimate the camera parameters
 - Version 1: solve for 3x4 projection matrix

$$\mathbf{X} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{P}\mathbf{X}$$

- Version 2: solve for camera parameters separately
 - intrinsics (focal length, principal point, pixel size)
 - extrinsics (rotation angles, translation)
 - radial distortion

Vanishing points and projection matrix

$$\mathbf{\Pi} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = \begin{bmatrix} \boldsymbol{\pi}_1 & \boldsymbol{\pi}_2 & \boldsymbol{\pi}_3 & \boldsymbol{\pi}_4 \end{bmatrix}$$

- $\boldsymbol{\pi}_1 = \mathbf{\Pi} \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T = \mathbf{v}_x$ (X vanishing point)
- similarly, $\boldsymbol{\pi}_2 = \mathbf{v}_y$, $\boldsymbol{\pi}_3 = \mathbf{v}_z$
- $\boldsymbol{\pi}_4 = \mathbf{\Pi} \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T =$ projection of world origin

$$\mathbf{\Pi} = \begin{bmatrix} \mathbf{v}_X & \mathbf{v}_Y & \mathbf{v}_Z & \mathbf{0} \end{bmatrix}$$

Not So Fast! We only know \mathbf{v} 's up to a scale factor

$$\mathbf{\Pi} = \begin{bmatrix} a \mathbf{v}_X & b \mathbf{v}_Y & c \mathbf{v}_Z & \mathbf{0} \end{bmatrix}$$

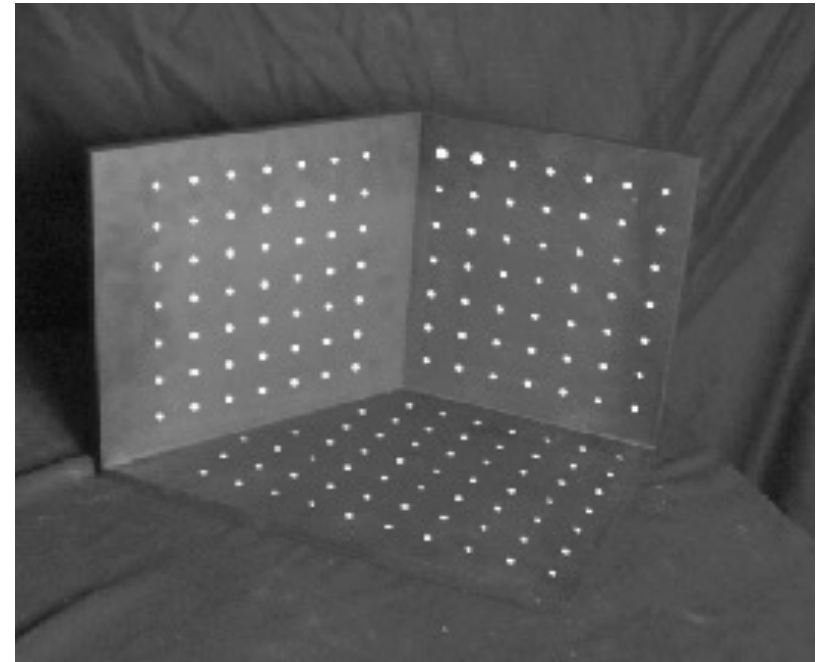
- Can fully specify by providing 3 reference points with known coordinates

Calibration using a reference object

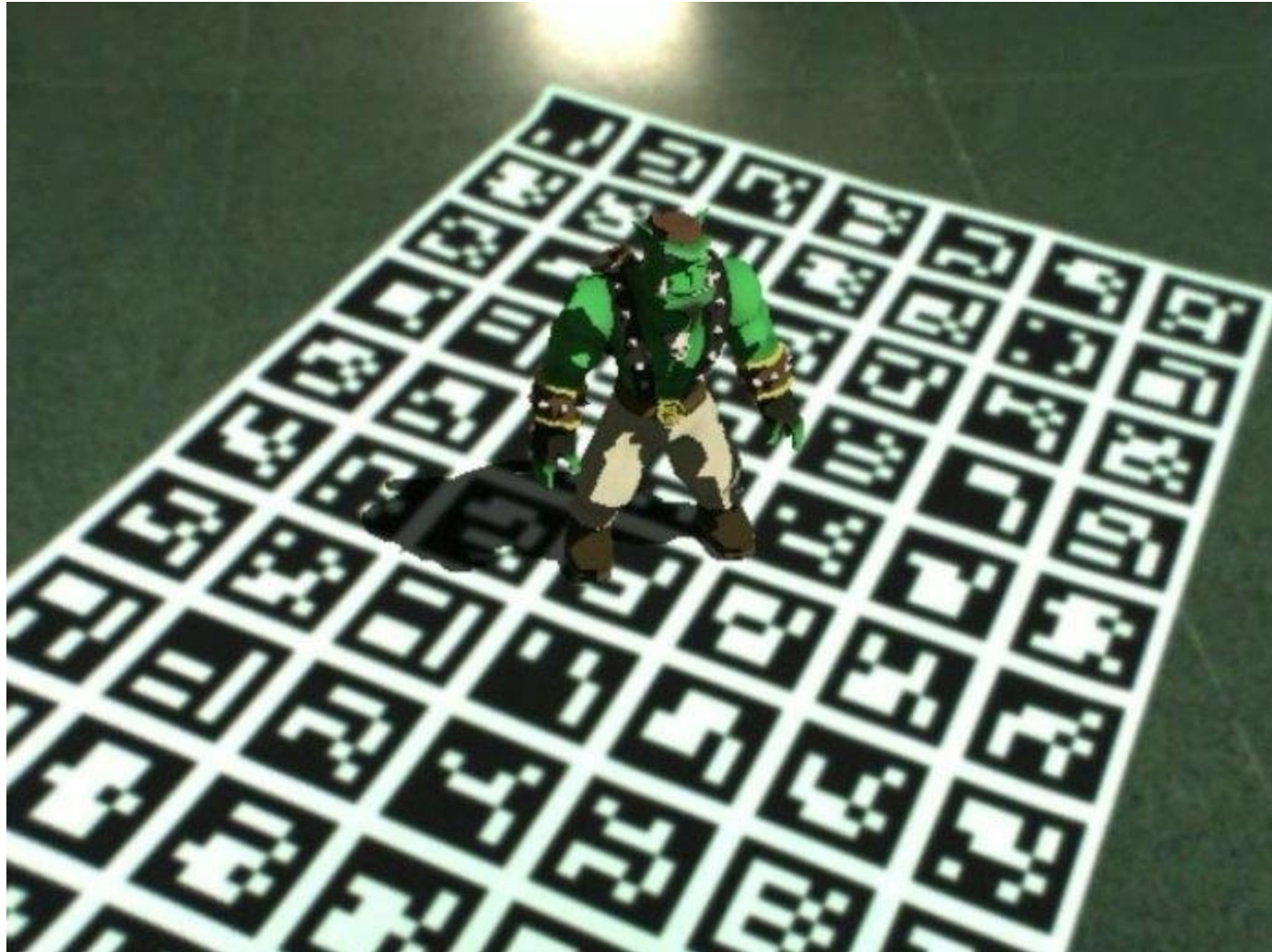
- Place a known object in the scene
 - identify correspondence between image and scene
 - compute mapping from scene to image

Issues

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



AR codes

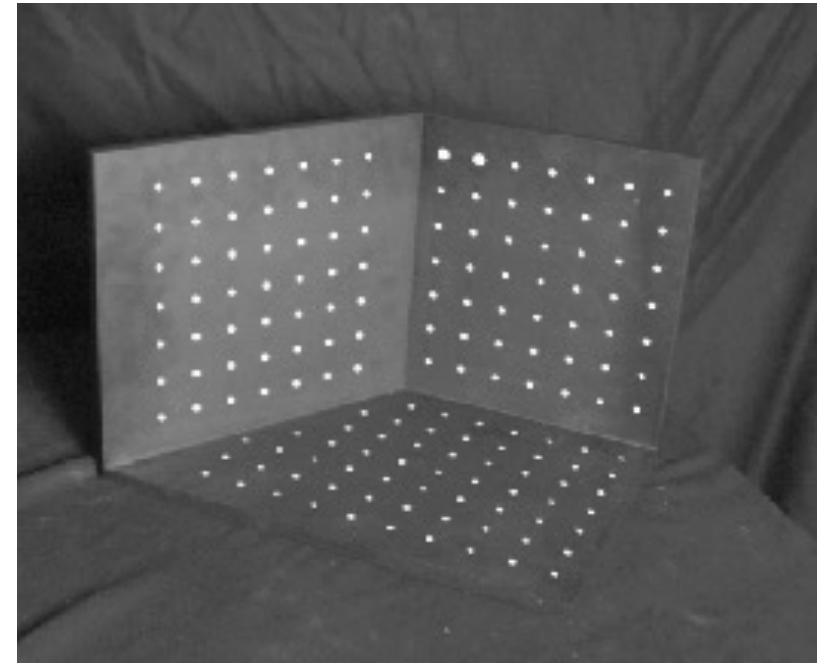


ArUco

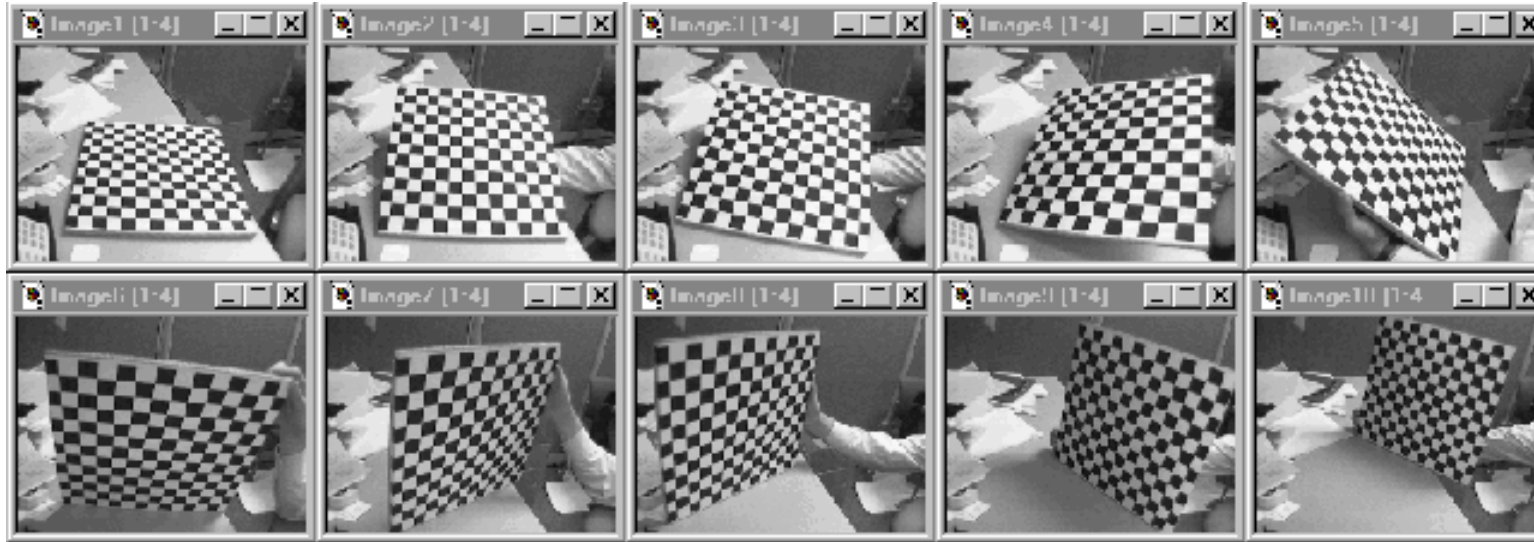
Estimating the projection matrix

- Place a known object in the scene
 - identify correspondence between image and scene
 - compute mapping from scene to image

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



Alternative: multi-plane calibration

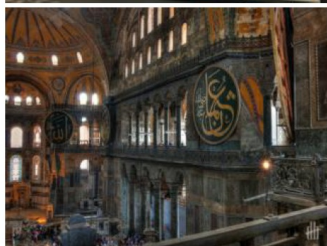


Images courtesy Jean-Yves Bouguet

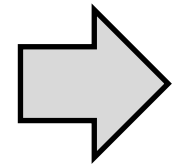
Advantage

- Only requires a plane
- Don't have to know positions/orientations
- Good code available online! (including in OpenCV)
 - Matlab version by Jean-Yves Bouguet: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html
 - Amy Tabb's camera calibration software: <https://github.com/amy-tabb/basic-camera-calibration>

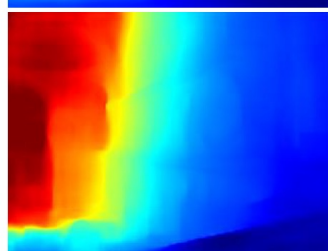
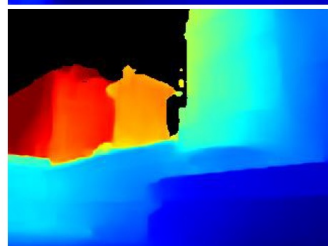
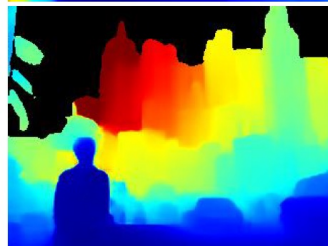
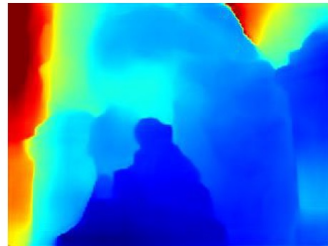
Single-image depth prediction using deep learning



Image



Deep learning

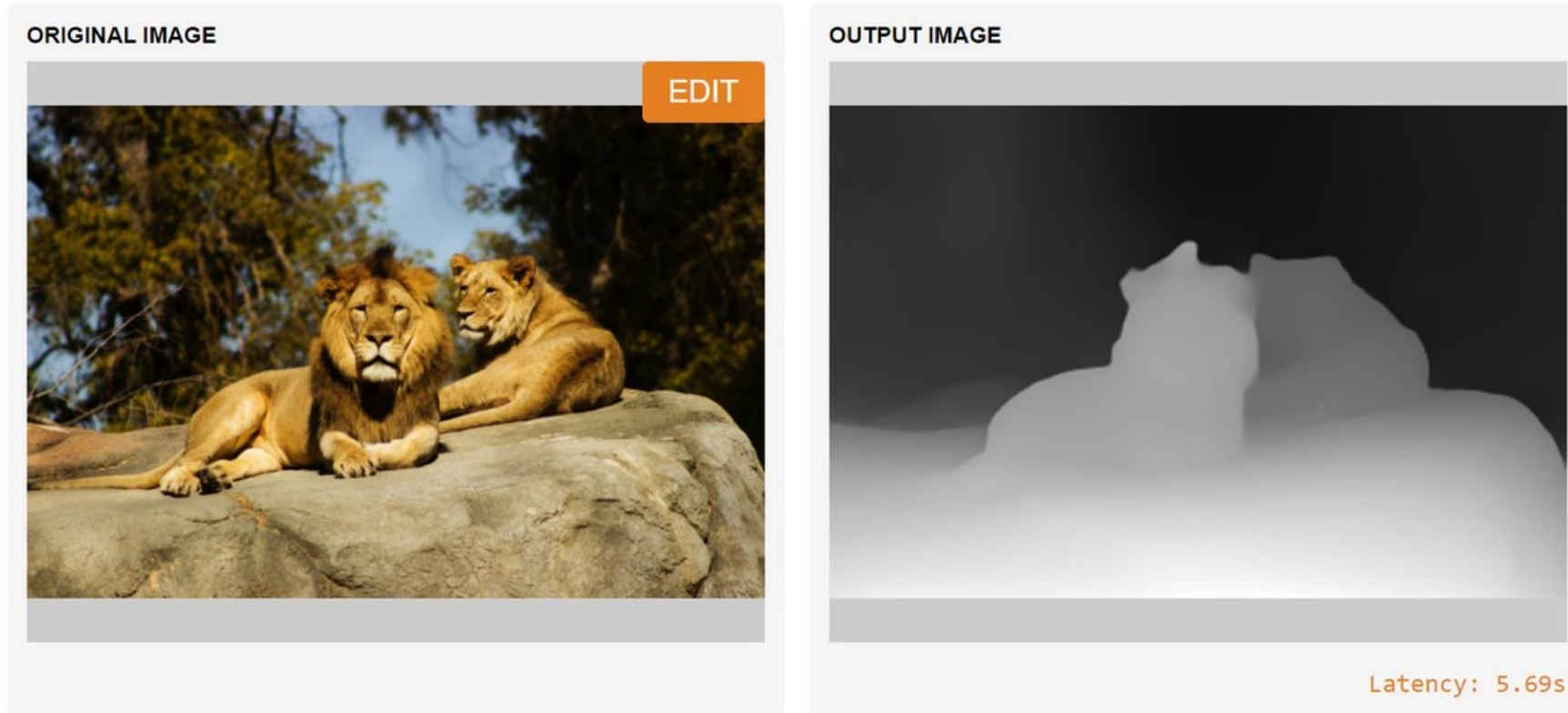


Depth map

Li and Snavely. Megadepth: Learning single-view depth prediction from internet photos. CVPR 2018.

MiDaS depth prediction

Ranftl et al. *Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer.*



<https://gradio.app/g/AK391/MiDaS>

<https://github.com/intel-isl/MiDaS>

Single-image depth prediction



Picture credit: [Magritte, The Treachery of Images](#), and [the Berkeley Computer Vision Group](#)

Miangoleh*, Dille*, Mai, Paris, and Aksoy.

Boosting Monocular Depth Estimation Models to High-Resolution via Content-Adaptive Multi-Resolution Merging.

CVPR, 2021

Deep geometry prediction

- More on this topic later!

Questions?