Introduction to Computer Vision

Instructors: Jean Ponce and Matthew Trager
jean.ponce@inria.fr, matthew.trager@cims.nyu.edu

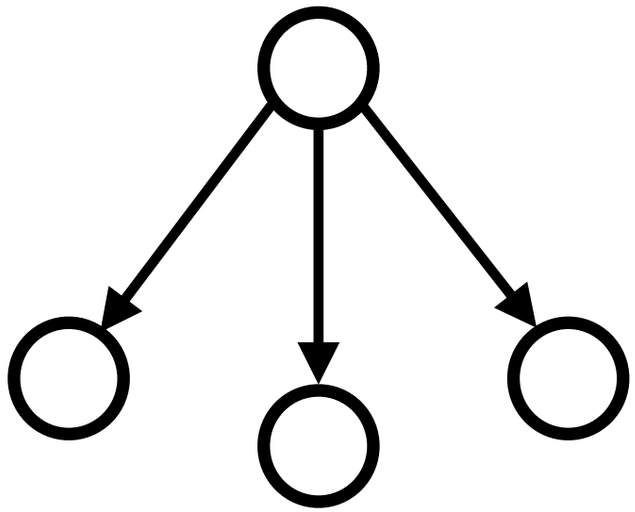
TAs: Jiachen (Jason) Zhu and Sahar Siddiqui
jiachen.zhu@nyu.edu, ss12414@nyu.edu

Topics

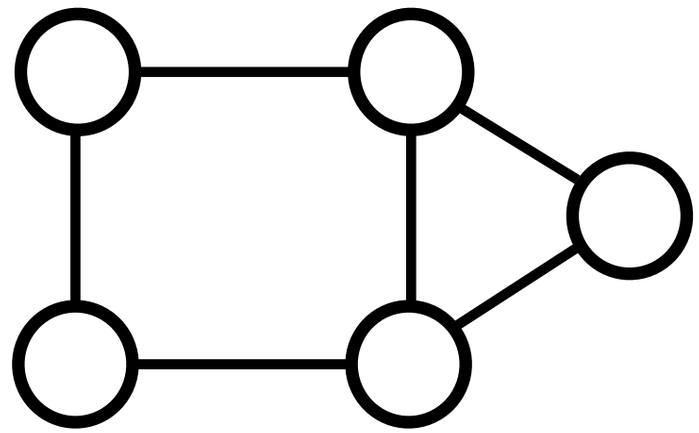
- Brief intro to MRF, graph cuts, and segmentation.
- Introduction to visual geometry, single-view geometry, image stitching.

Probabilistic Graphical Models

A graphical model is a collection of probability distributions that can be associated with a graph.



Directed Graphical Models
Bayesian Networks



Undirected Graphical Models
Markov Random Fields

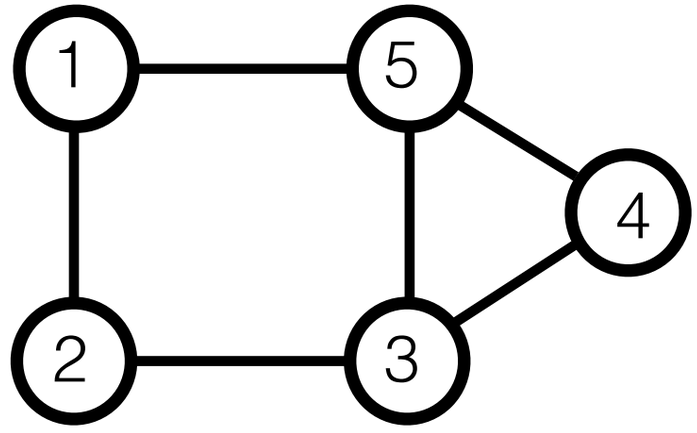
Nodes ↔ Random variables

Graph reachability↔ (In)dependence of r.v.

Markov Random Fields

- $G = (V, E)$ undirected graph
- $\{Y_s\}_{s \in V}$ random variables
- If $\mathscr{C}$ is the set of cliques of G, then

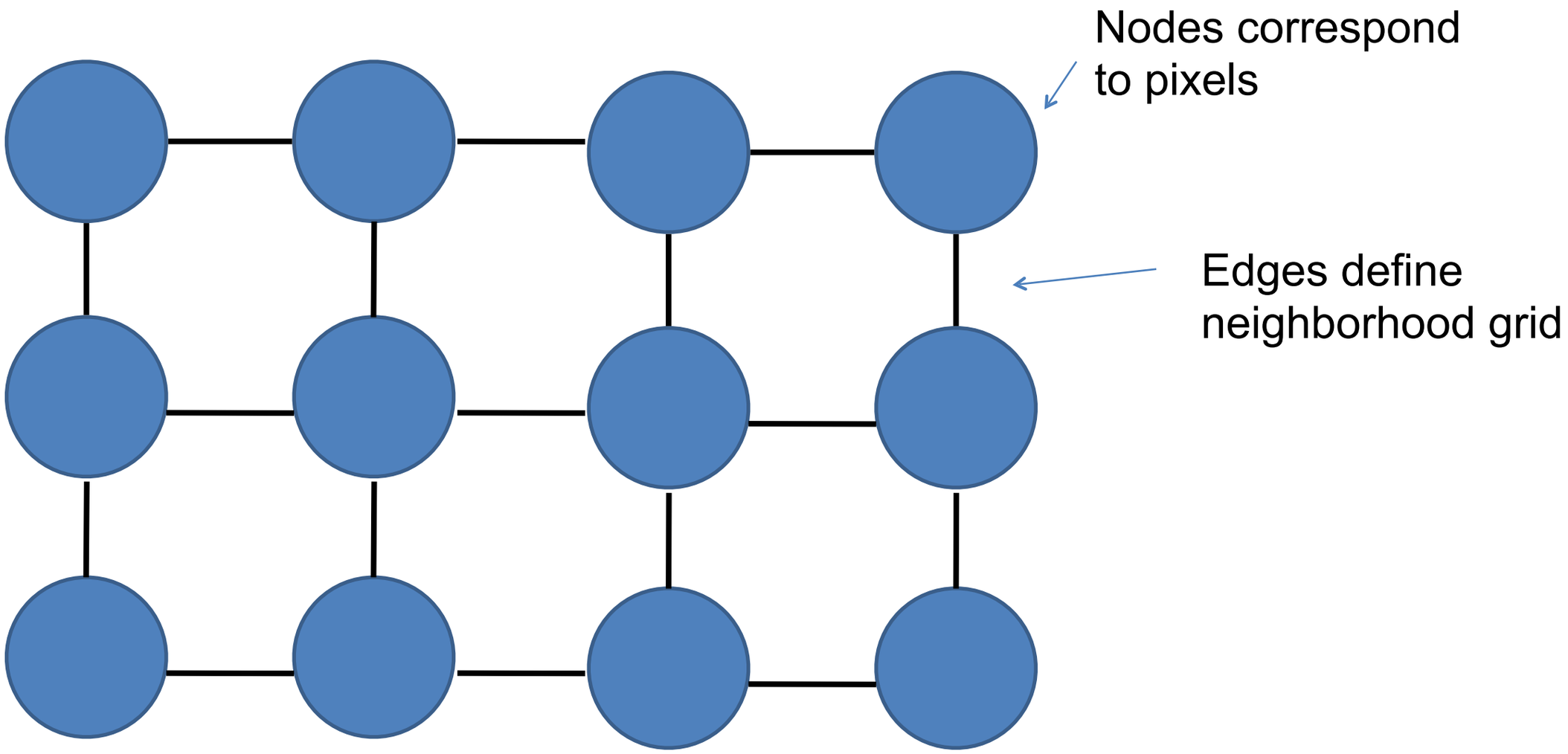
$$p\left(y_1, y_2, \ldots, y_m\right) = \frac{1}{Z} \prod_{C \in \mathscr{C}} \psi_C\left(y_C\right)$$



partition constant $Z = \sum_{y_i} \prod_{c \in \mathscr{C}} \psi_C(y_c)$

variables associated with a clique c

Binary image segmentation



Each node is a binary r.v. (background/forground).

Conditional Markov Random Fields

- X observed random variables
- $\{Y_s\}_{s \in V}$ random variables
- If $\mathscr{C}$ is the set of cliques of G, then

$$p\left(y_1, \ldots, y_n | X\right) = \frac{1}{Z(X)} \prod_{C \in \mathscr{C}} \psi_C\left(y_C; X\right)$$

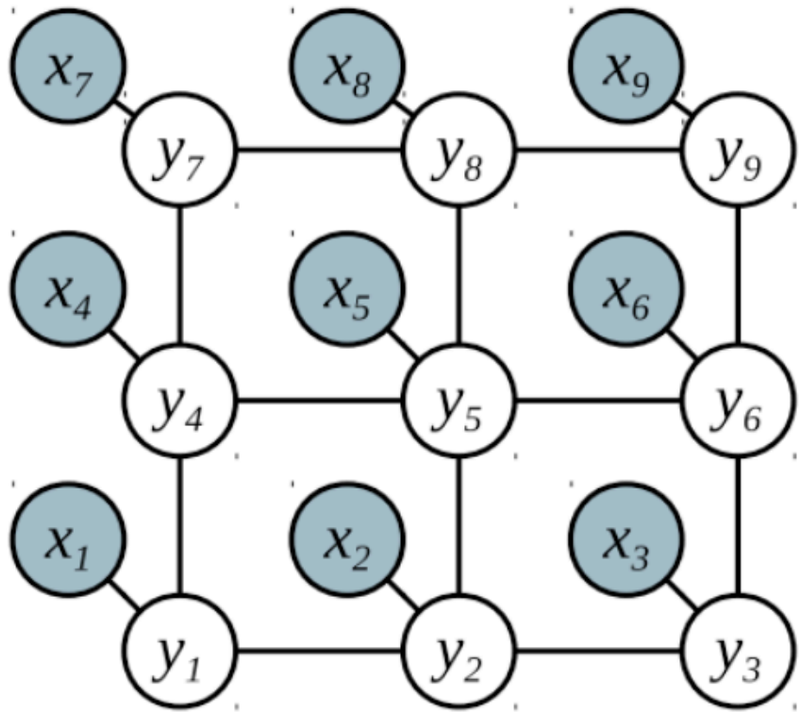
partition function $Z(X) = \sum_{y_i} \prod_{c \in \mathscr{C}} \psi_C(y_c; X)$

variables associated with a clique c

Segmentation

- x_i are observed variables (pixel values)
- $y_i \in \{-1,1\}$ are unknown labels.
- We would like to solve the Maximum a Posteriori (MAP) problem:

$$Y^\star = \underset{Y \in \mathcal{Y}}{\operatorname{argmax}}\, P(Y|X)$$



Maximum a Posteriori (MAP) inference

$$
\begin{aligned}
Y^\star &= \operatorname*{argmax}_{Y \in \mathcal{Y}} P(Y \mid X) \\
&= \operatorname*{argmax}_{Y \in \mathcal{Y}} \frac{1}{Z(X)} \prod_c \psi_c\left(Y_c; X\right) \\
&= \operatorname*{argmax}_{Y \in \mathcal{Y}} \log\left(\frac{1}{Z(X)} \prod_c \psi_c\left(Y_c; X\right)\right) \\
&= \operatorname*{argmax}_{Y \in \mathcal{Y}} \sum_c \log \psi_c\left(Y_c; X\right) - \log Z(X) \\
&= \operatorname*{argmax}_{Y \in \mathcal{Y}} \sum_c \log \psi_c\left(Y_c; X\right) \qquad \leftarrow -E(Y; X)
\end{aligned}
$$

Maximum a Posteriori (MAP) inference

$$Y^\star = \underset{Y \in \mathcal{Y}}{\operatorname{argmax}}\, P(Y \mid X) = \underset{Y \in \mathcal{Y}}{\operatorname{argmax}} \sum_c \log \psi_c \left(Y_c; X\right) = \underset{Y \in \mathcal{Y}}{\operatorname{argmin}}\, E(Y; X)$$

MAP inference ↔ Energy minimization

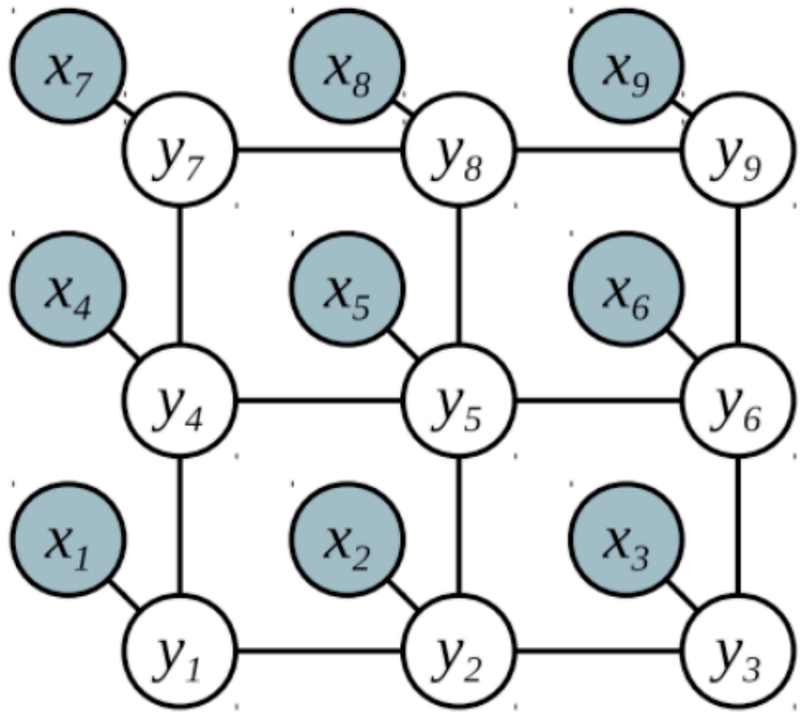
The energy function is

$$E(Y; X) = \sum_{c \in \mathscr{C}} \phi(Y_c; X)$$

where $\phi(Y_c; X) = -\log \psi(Y_c, X)$.

Segmentation

- x_i are observed variables (pixel values)
- $y_i \in \{-1,1\}$ are unknown labels.
- $d_f(x_i), d_b(x_i)$ are functions the pixel x_i with foreground and background models.
- $B(x_i, x_j)$ is a non-negative symmetric function that compares two pixels ,e.g., $k_1 + k_2 \exp(-\|x_i - x_j\|^2)$.



$$E(Y;X) = \frac{1}{2}\left[\sum_i d_f(x_i)(1+y_i) + d_b(x_i)(1-y_i) + \sum_i \sum_{j \in \mathcal{N}(i)} B(x_i, x_j)(1 - y_i y_j)\right]$$

Energy minimization problem

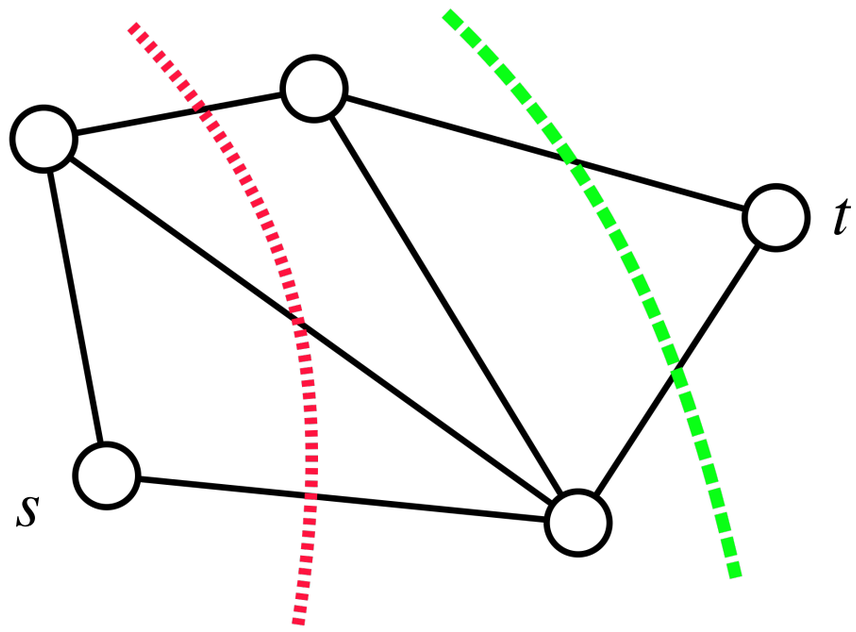
- We need to solve a discrete optimization problem:

$$\min_{Y \in \{0,1\}^N} E(Y; X)$$

- For general MRF, many strategies (belief propagation, simulated annealing, etc.) but the problem is NP-hard.
- In the case of binary segmentation, the problem reduces computing a minimizing cut in a graph, which can be solved in polynomial time.

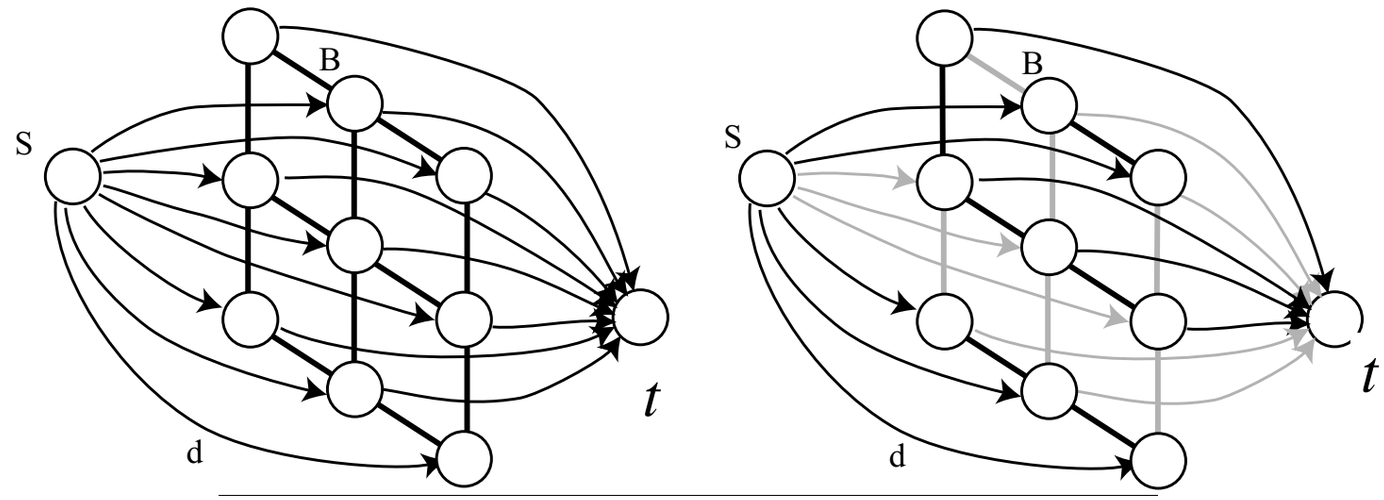
Min-Cut problem

- Let $G = (V, E)$ be a graph with two special nodes s and t.
- A cut of G is a partition of V into disjoint subsets S and T such that $s \in S, t \in T$.
- The cut set is $C(S, T) = \{(v, w) \in E \mid v \in S, w \in T\}$.
- Given a cost function $c : E \to \mathbb{R}^+$, we wish to minimize the cost of the cut: $\mathscr{E}(S, T) = \sum_{(v,w) \in C(S,T)} c(v, w)$



Graph cuts for segmentation

$$E(Y;X) = \frac{1}{2}\left[\sum_i d_f\left(x_i\right)\left(1+y_i\right) + d_b\left(x_i\right)\left(1-y_i\right) + \sum_i \sum_{j\in\mathcal{N}(i)} B\left(x_i,x_j\right)\left(1-y_i y_j\right)\right]$$



- $c(i,j) = B(x_i, x_j)$
- $c(s,i) = d_f(x_i)$ if label unknown, otherwise 0 or ∞.
- $c(j,t) = d_b(x_j)$ if label unknown, otherwise 0 or ∞.

Application: GrabCut

- User provides a bounding box for fg/bg model.
- Algorithm iteratively applies graph cut to re-estimate region statistics.

Rother, Kolmogorov, and Blake (2004)

Summary

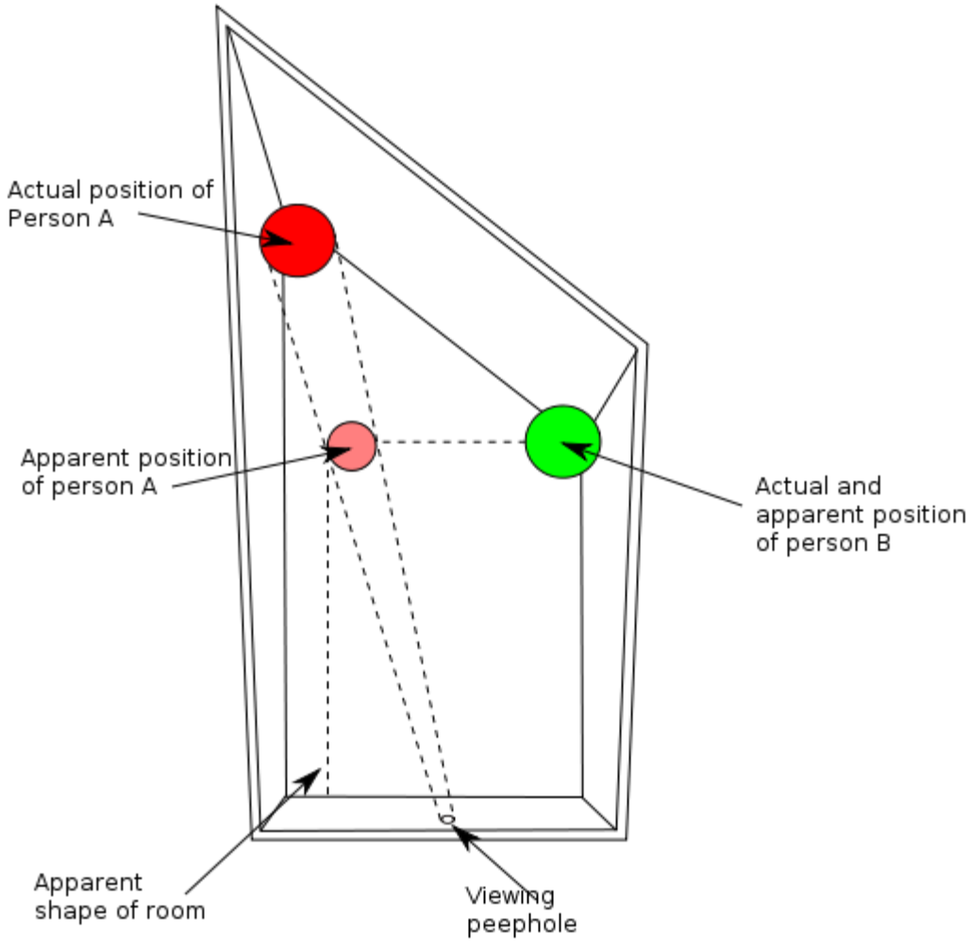
- MRF are a useful tool in computer vision (segmentation, but also stereo, pose estimation…)
- The MAP inference problem is a discrete optimization problem that can be NP-hard.
- For segmentation, reduces to graph min-cut which can be solved efficiently.

Other references:

- Book on graphical models: Koller and Friedman 2009
- Concise online lecture notes by Stefano Ermon
- Classic paper: What Energy Functions can be Minimized via Graph Cuts? (Kolmogorov and Zabih)

Visual Geometry

- What can we say about the 3D world from pictures?
  - Using one picture, many ambiguities



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

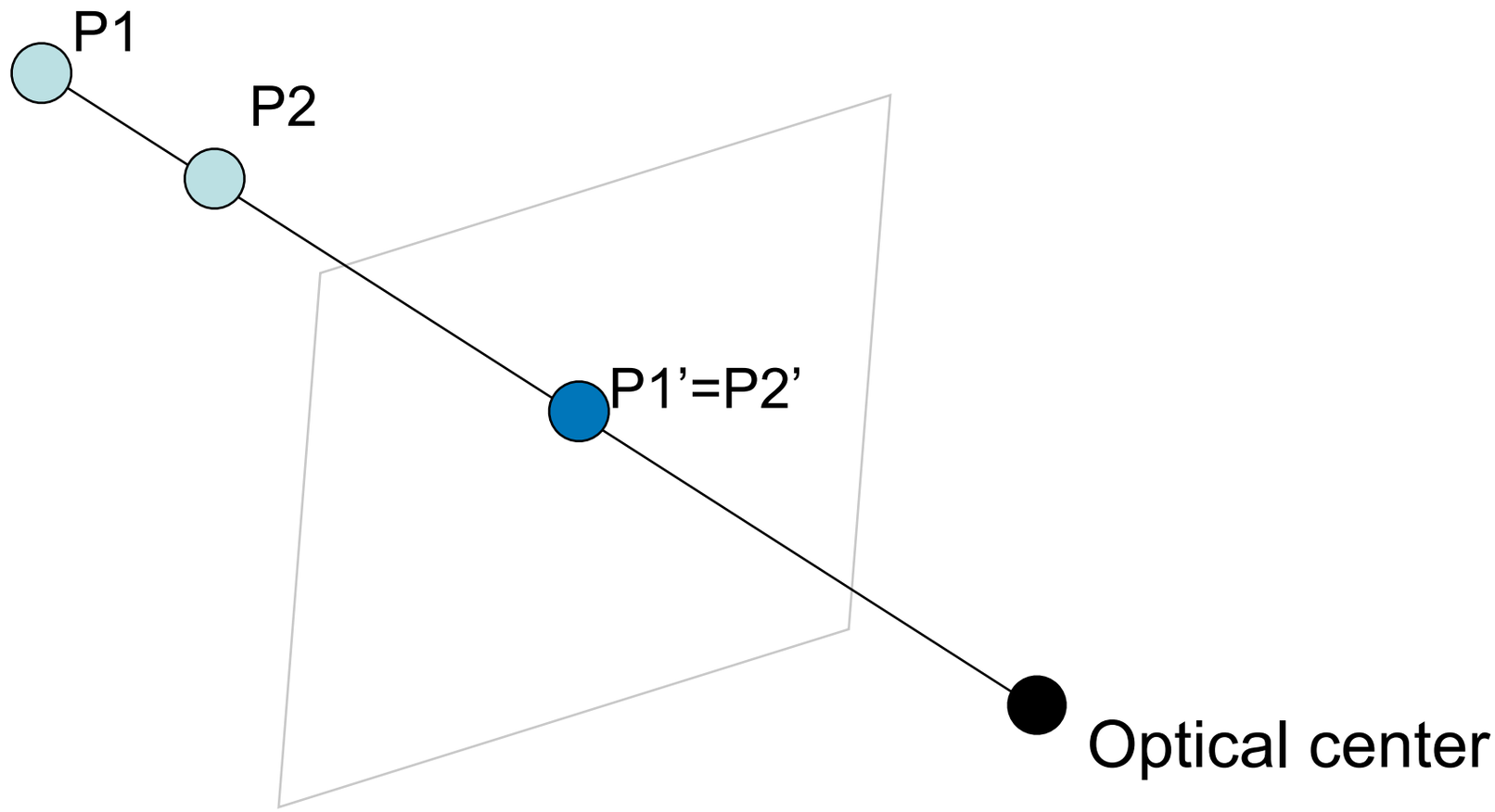
Visual Geometry

- What can we say about the 3D world from 2D pictures?
  - Using one picture, many ambiguities

Images from S. Lazebnik

Why the ambiguity?

- Structure and depth are ambiguous from single views.



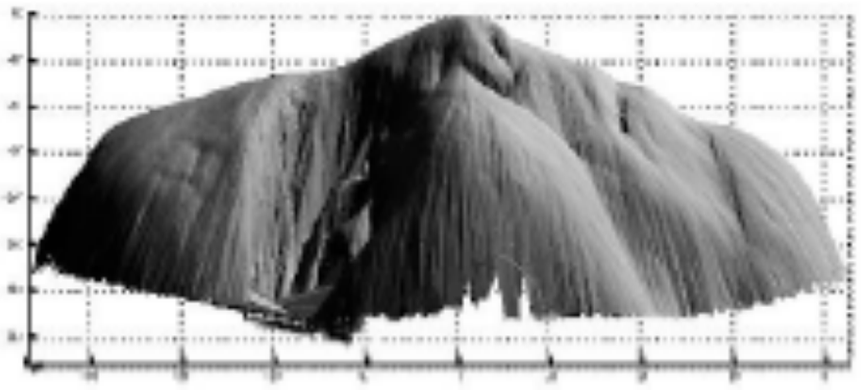
What cues help us to perceive 3d shape and depth?

If stereo were critical for depth perception, navigation, recognition, etc., then this would be a problem

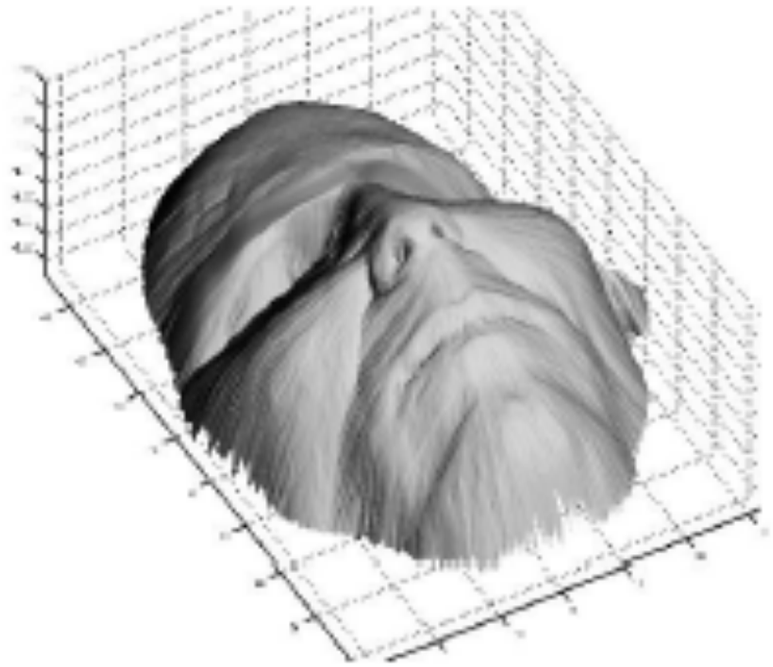
Source J. Hays

Shading

a)



b)

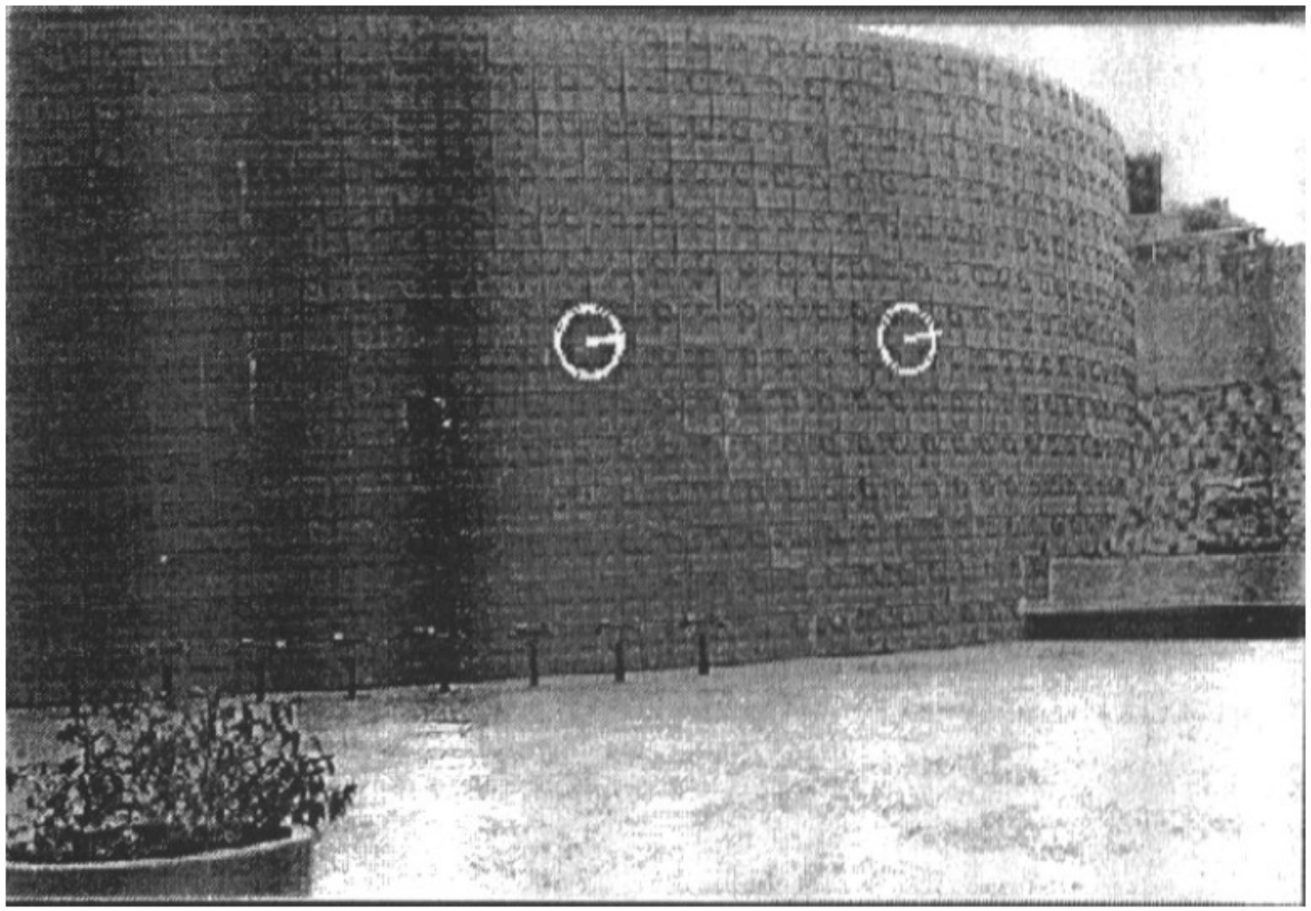


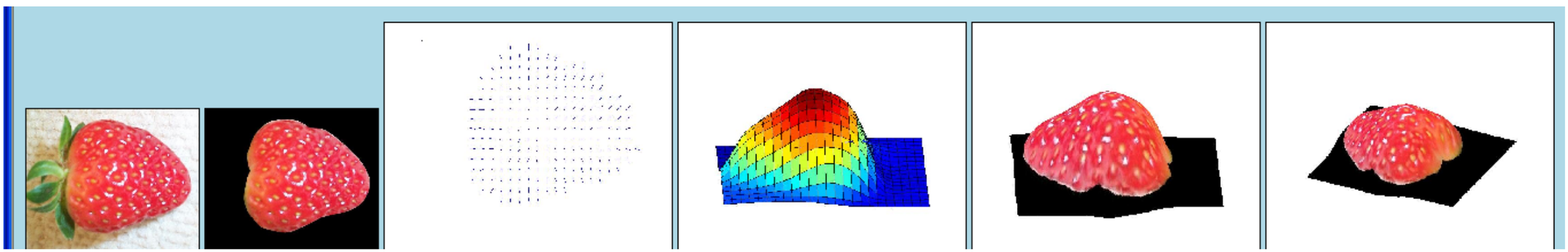
c)

[Figure from Prados & Faugeras 2006]

Source J. Hays

Texture

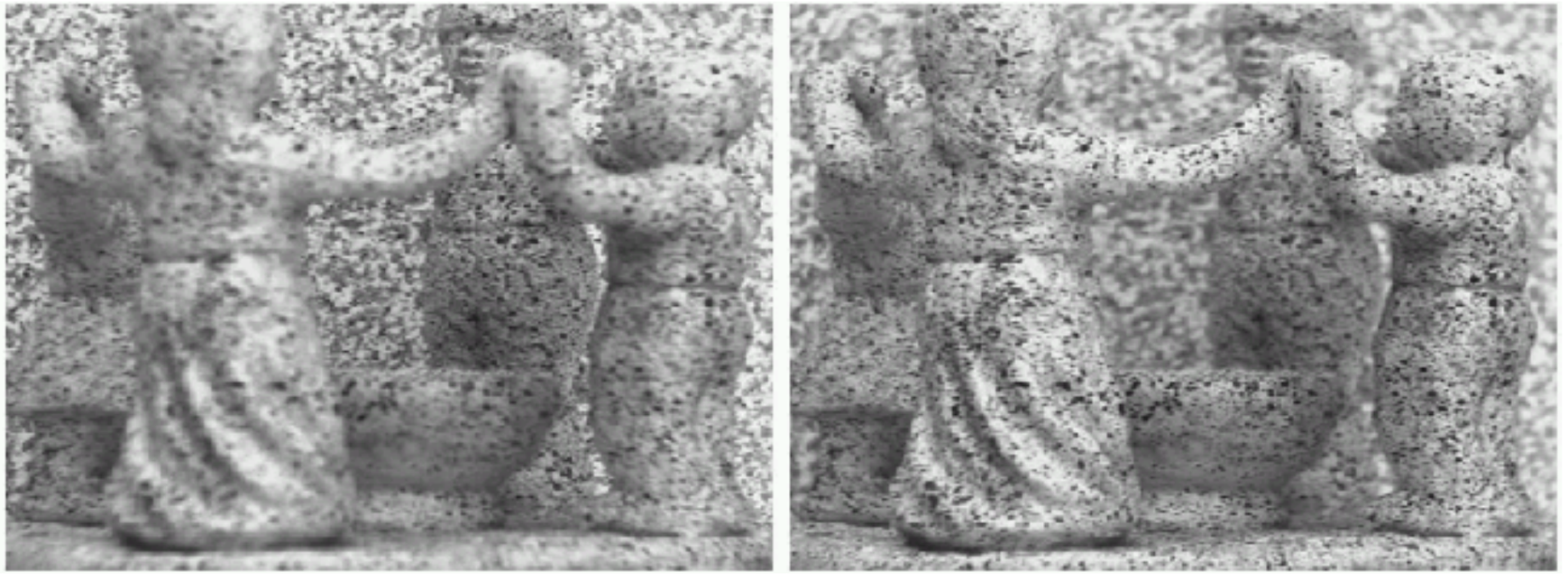




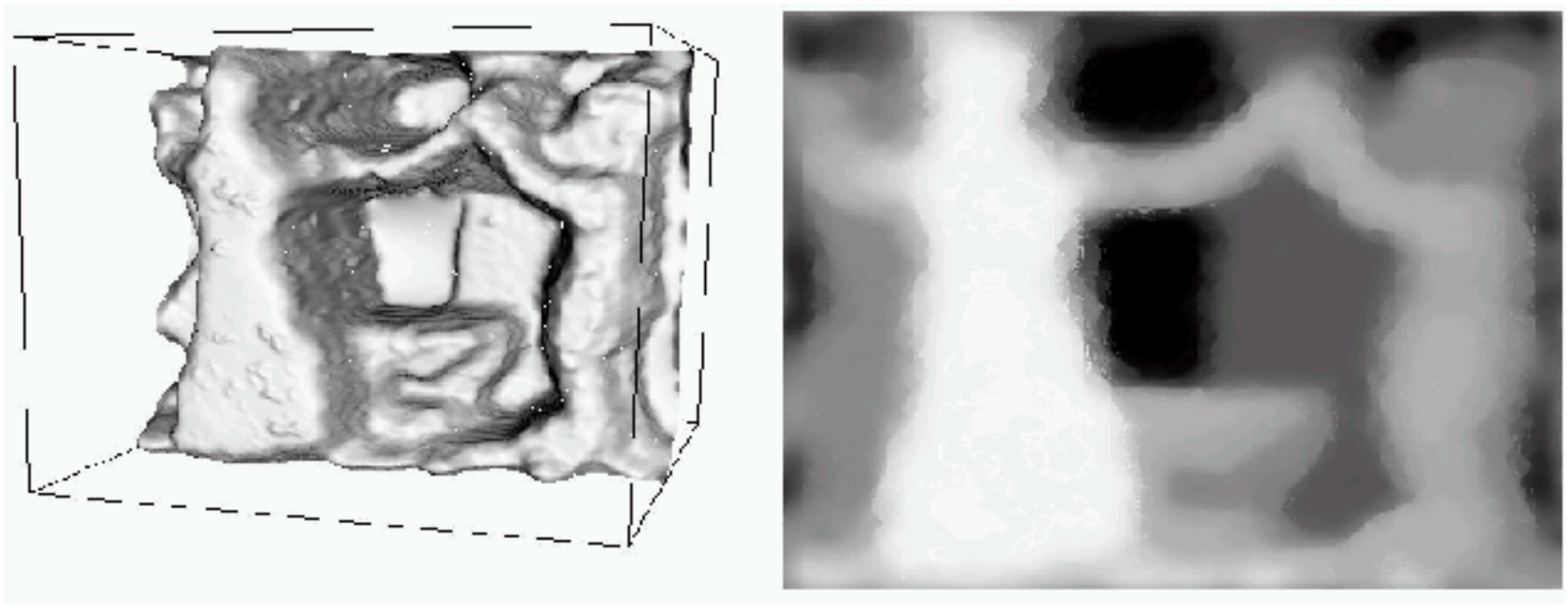
[From A.M. Loh. The recovery of 3-D structure using visual texture patterns. PhD thesis]

Source J. Hays

Focus/defocus



Images from same point of view, different camera parameters



3d shape / depth estimates

[figs from H. Jin and P. Favaro, 2002]

Occlusion

Rene Magritt'e famous painting *Le Blanc-Seing,* 1965

Source J. Hays

Perspective effects

Image credit: S. Seitz

Motion

Figures from L. Zhang

3D reconstruction

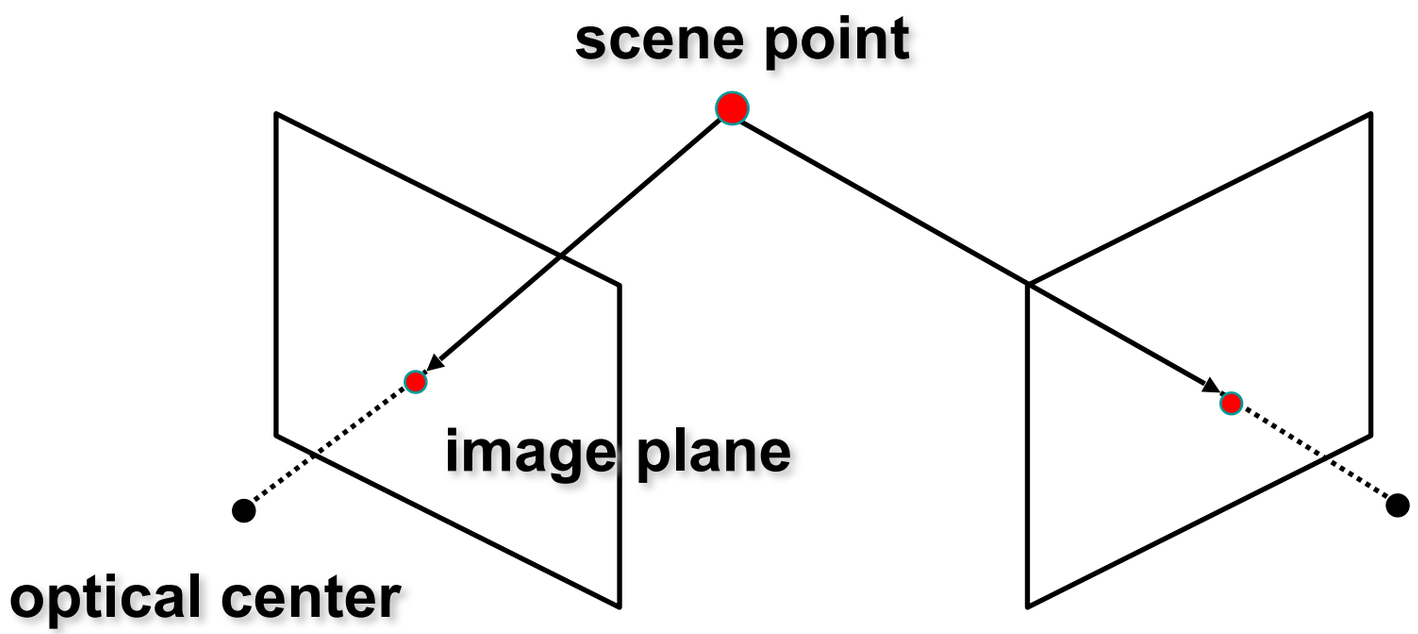
- Typically 3D reconstruction is computed from multiple photographs.
  - Humans (usually) can use one view but it's very difficult.
  - Reconstructing from many views and unknown camera locations is still difficult!
  - Many variations: structure-from-motion, multi-view stereo, etc.

3D reconstruction from Internet photos

BigSFM, Snavely at al. (2013)

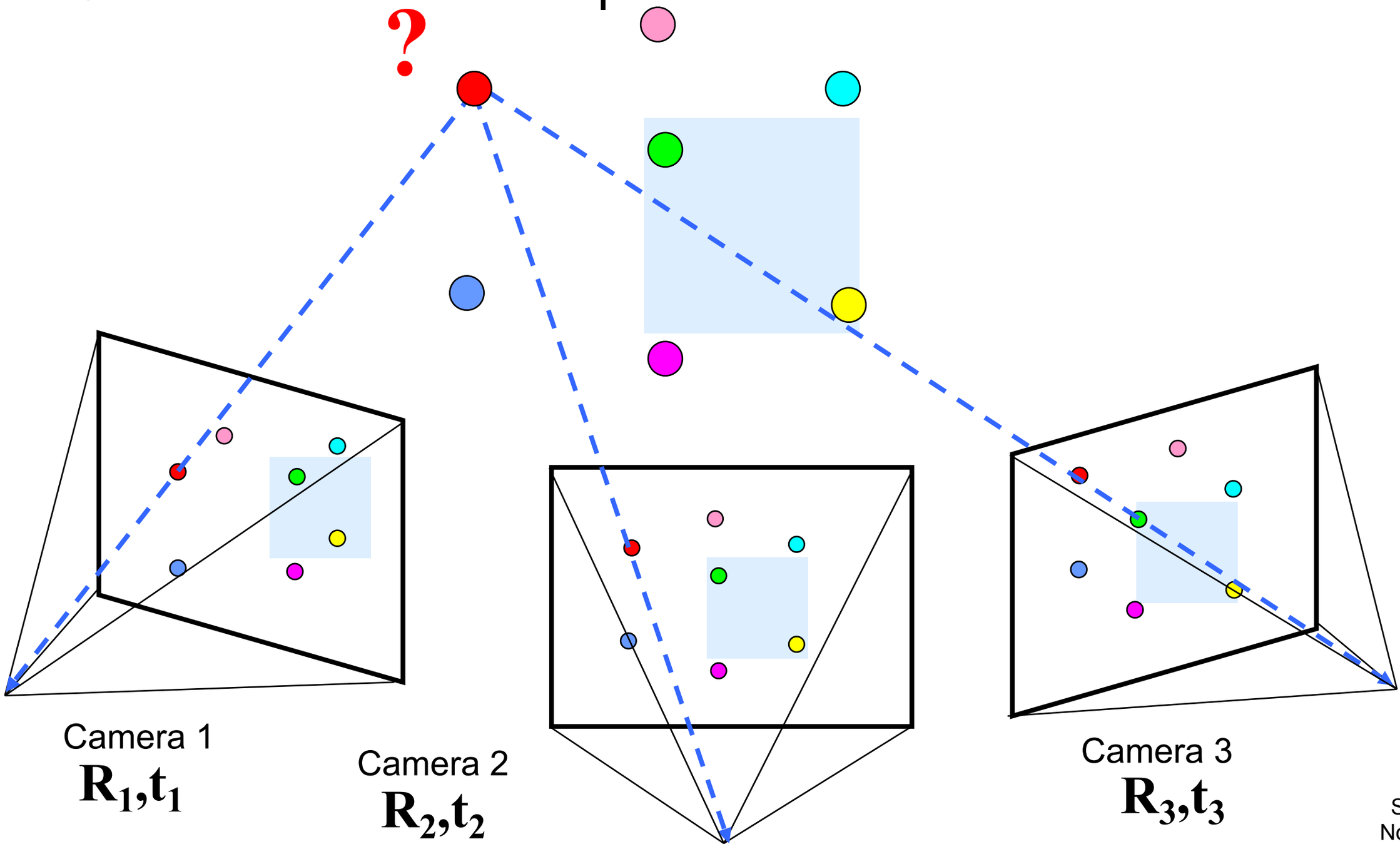
Problems in visual geometry

- Camera calibration: use 3D-2D corrs
- Structure-from-motion: use 2D-2D corrs
- (Multi-view) stereo: use motion and images to create 3D model
- “Shape from X”: use Shading, Texture, Focus



Multi-view geometry problems

- Given projections of the same 3D points in two or more images, compute the unknown camera parameters and 3D coordinates of the points



Slide credit: Noah Snavely

What is preserved in images?

- Distances?
  – no!
- Parallelism
  – no!
- Angles?
  – no!
- Collinearity?
  – Yes!

Projective Geometry

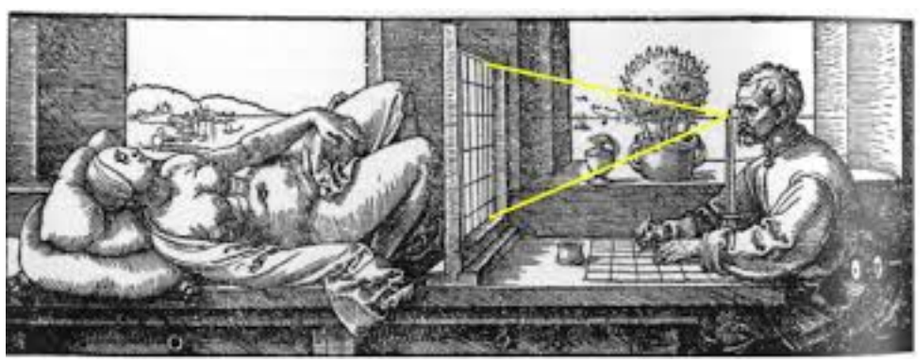
Projective geometry is a classical topic in mathematics. Like Euclidean geometry but with:

- *Points at infinity*: where parallel lines meet.
- *Projective transformations*: maps that preserve lines but not angles and distances.

Vanishing points are images of points at infinity.

Angles and distances are not preserved in images.

Emerged from the study of *perspectivity* in the Renaissaince.



Projective Geometry

The n-dimensional projective space $\mathbb{P}^n$ is defined as

$$\mathbb{P}^n = (\mathbb{R}^{n+1} \backslash \{0\}) / \sim ,$$

where $v \sim w$ if $v = \lambda w$ with $\lambda \in \mathbb{R} \backslash \{0\}$. Equivalence classes of vectors are “projective points”.

Concretely: represent points with homogeneous coordinates!

- Affine space can be mapped into projective space: $x \in \mathbb{R}^n \mapsto [x; 1] \in \mathbb{P}^n$.
- “Points at infinity” are extra points: $[x; 0] \in \mathbb{P}^n$.
- No projective notion of parallelism: in a projective plane ($n = 2$), lines always intersect.
- Projective transformations (“homographies”) are linear transformations on homogeneous coordinates. Bigger group of transformations than affine maps!

Euclidean to Homogeneous coordinates

- From Euclidean to homogeneous coordinates in 2D:

$$\begin{pmatrix} x \\ y \end{pmatrix} \Longrightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- From Euclidean to homogeneous coordinates in 3D:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \Longrightarrow \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Homogeneous to Euclidean coordinates

(for non-zero vectors)

- From homogeneous to Euclidean coordinates in 2D:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \Rightarrow \begin{cases} c = 0 \Rightarrow \text{Point at infinity in the direction } (a, b) \\ c \neq 0 \Rightarrow \begin{pmatrix} \frac{a}{c} \\ \frac{b}{c} \end{pmatrix} \end{cases}$$

- From homogeneous to Euclidean coordinates in 3D:

$$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} \Rightarrow \begin{cases} d = 0 \Rightarrow \text{Point at infinity in the direction } (a, b, c) \\ d \neq 0 \Rightarrow \begin{pmatrix} \frac{a}{d} \\ \frac{b}{d} \\ \frac{c}{d} \end{pmatrix} \end{cases}$$

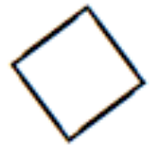
Homogeneous coordinates

- Two set of coordinates that differ by a multiplicative constant denote the same cartesian vector.

$$\lambda \neq 0 \quad \Longrightarrow \quad \begin{pmatrix} a \\ b \\ c \\ \vdots \end{pmatrix} \sim \begin{pmatrix} \lambda a \\ \lambda b \\ \lambda c \\ \vdots \end{pmatrix}$$

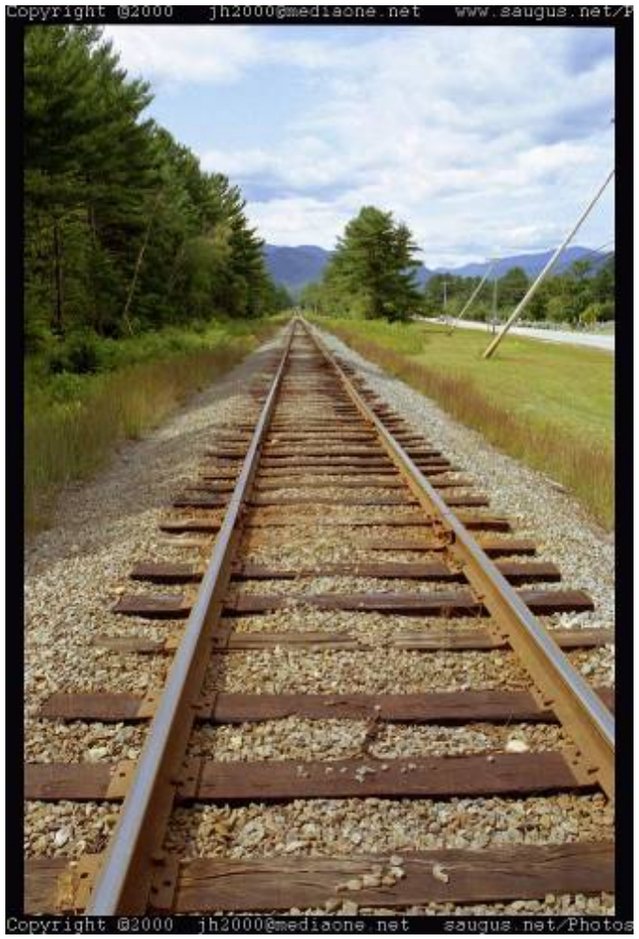
Point in Cartesian is ray in Homogeneous

3D transformations and homogeneous coordinates

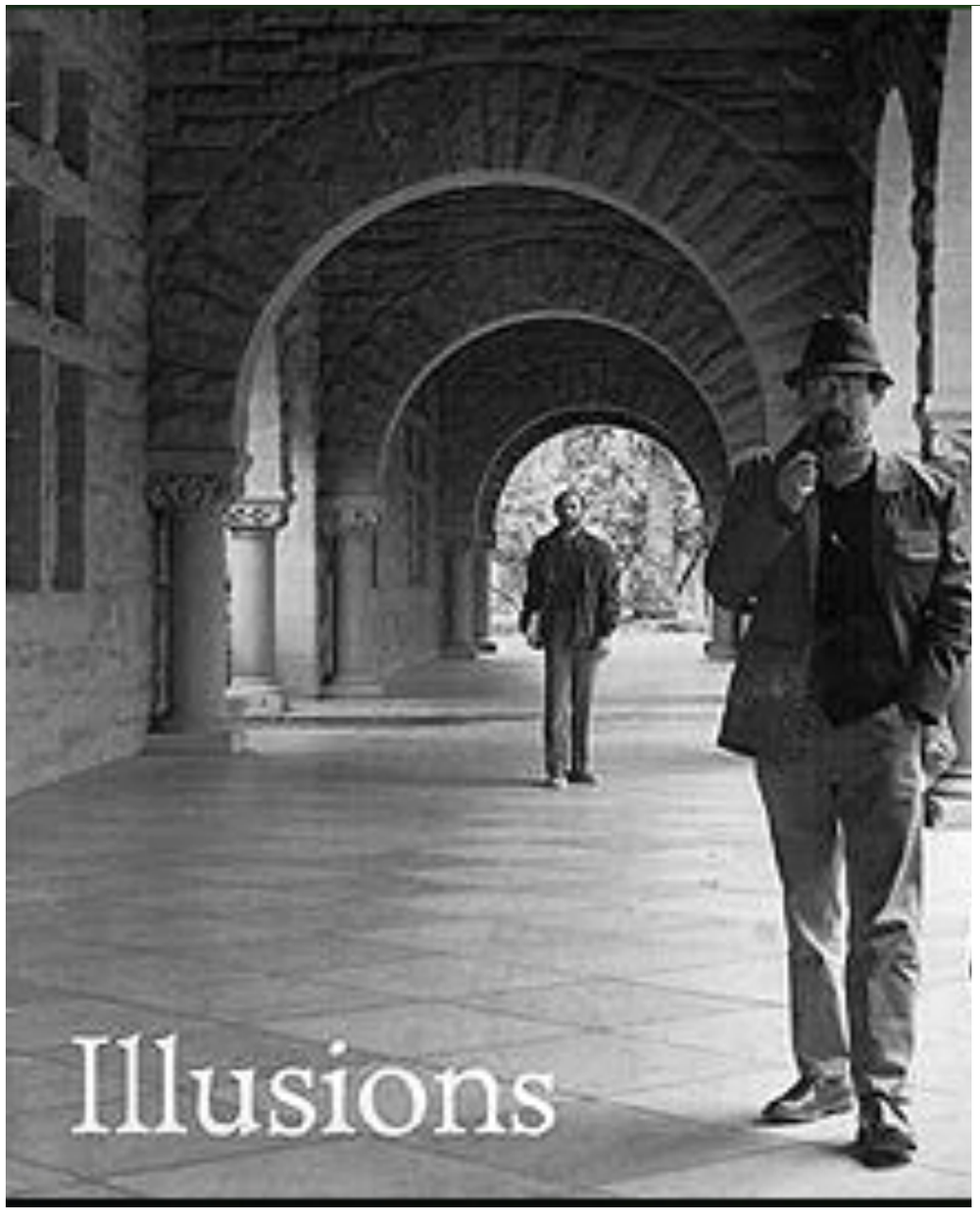
Transformation	Matrix	# DoF	Preserves	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{3\times4}$	3	orientation	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{3\times4}$	6	lengths	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{3\times4}$	7	angles	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{3\times4}$	12	parallelism	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{4\times4}$	15	straight lines	

Vanishing points and lines

Parallel lines in the world intersect in the image at a “vanishing point”

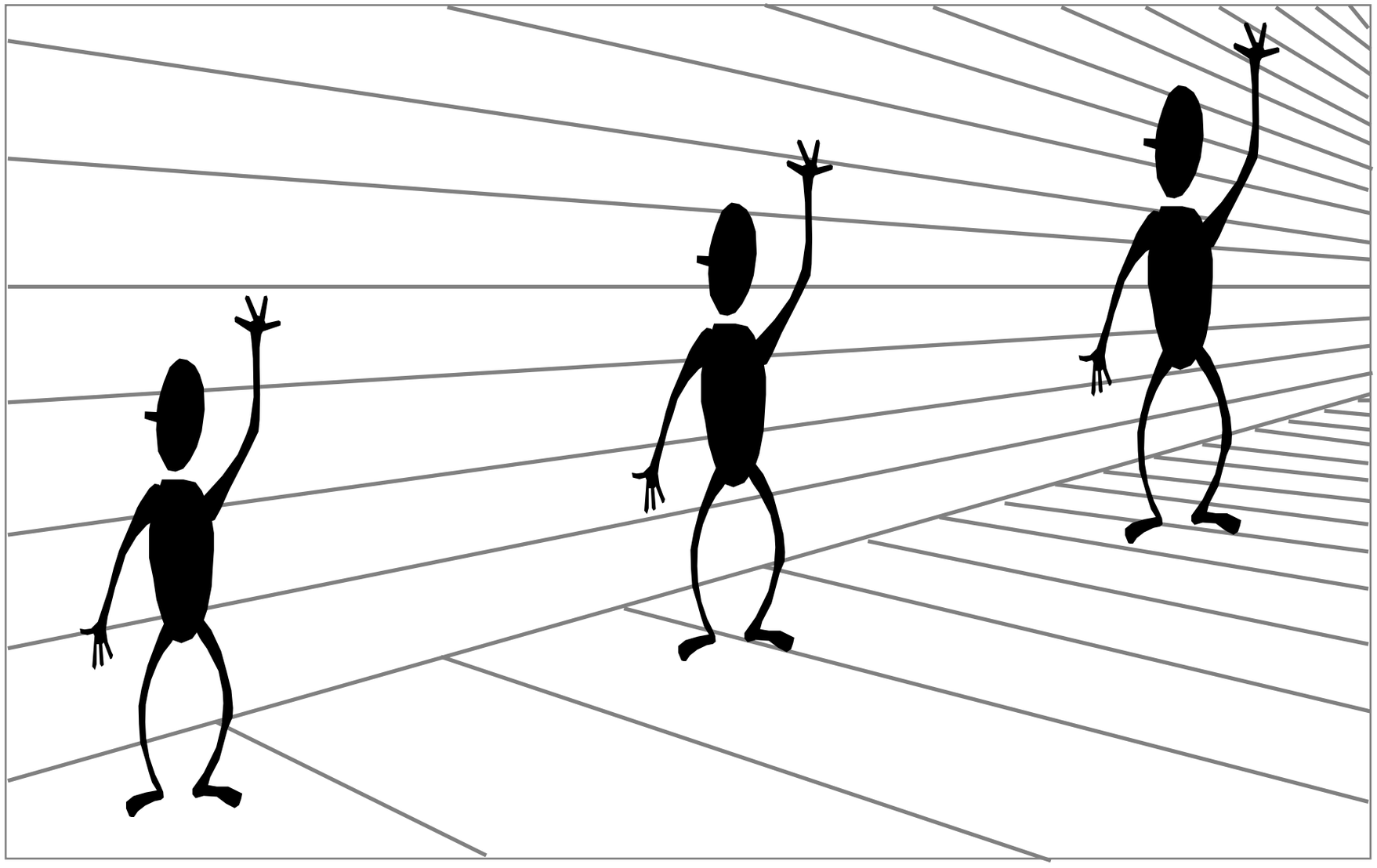


How can we measure the size of 3D objects from an image?

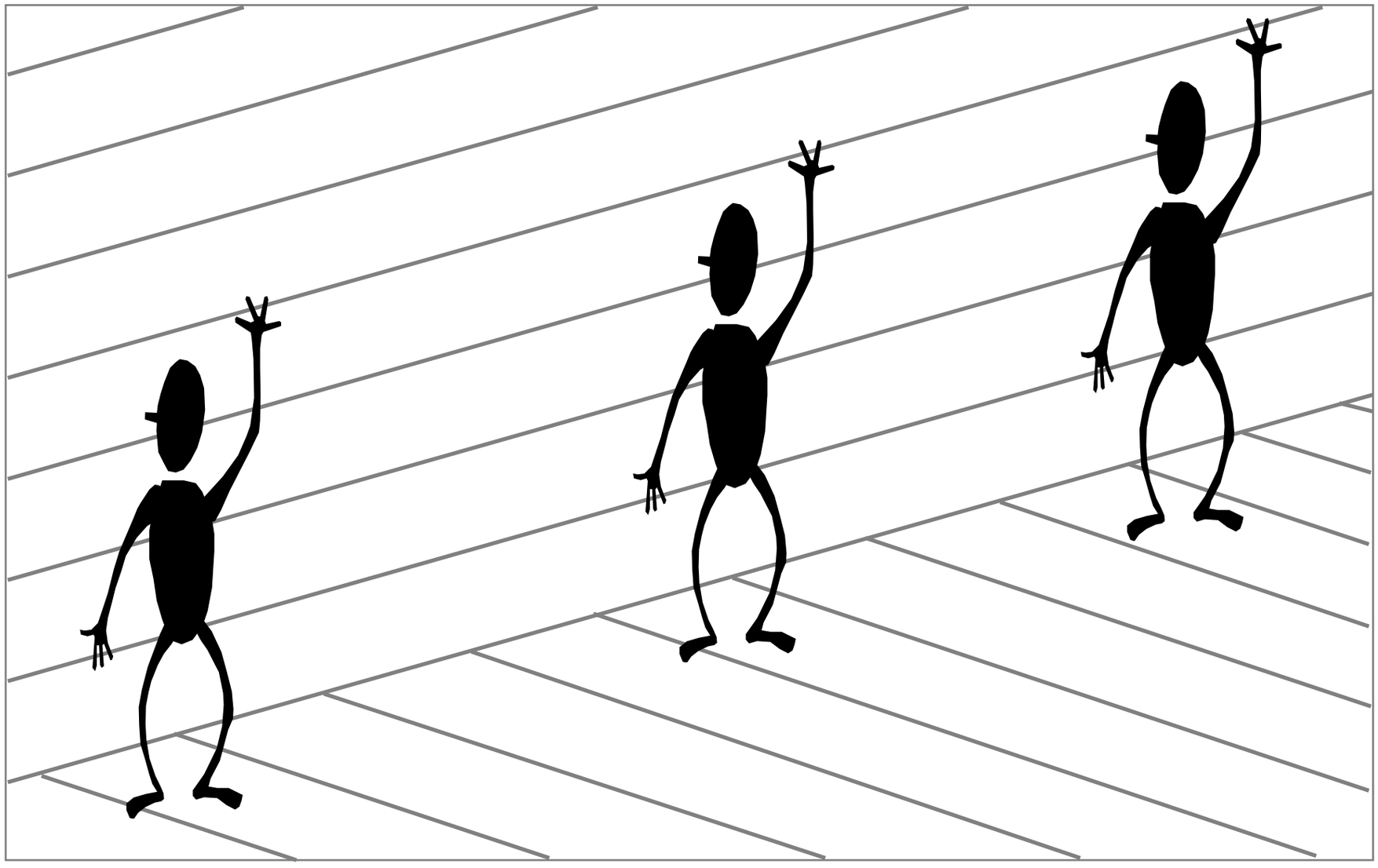


Slide by Steve Seitz

Perspective cues

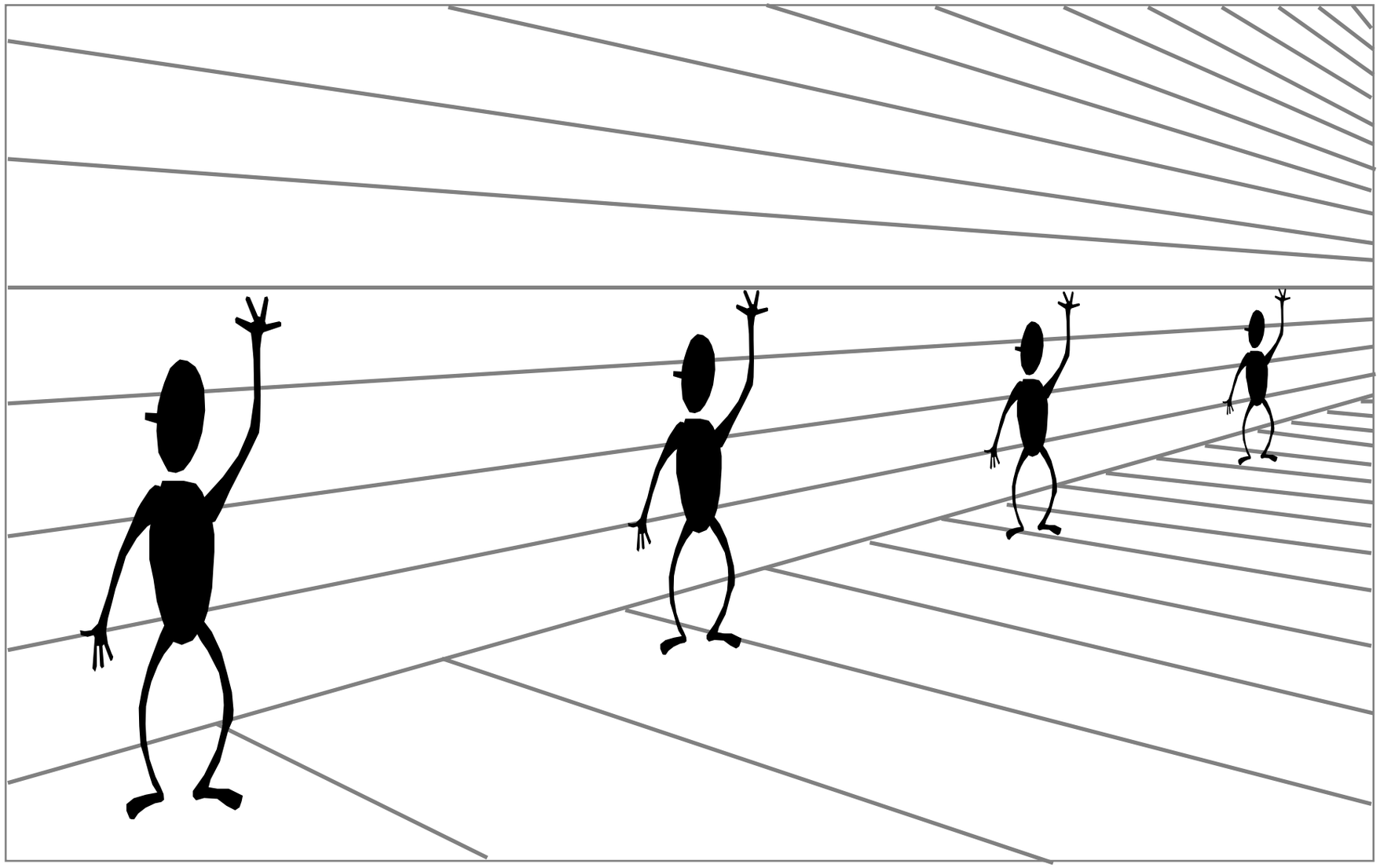


Perspective cues

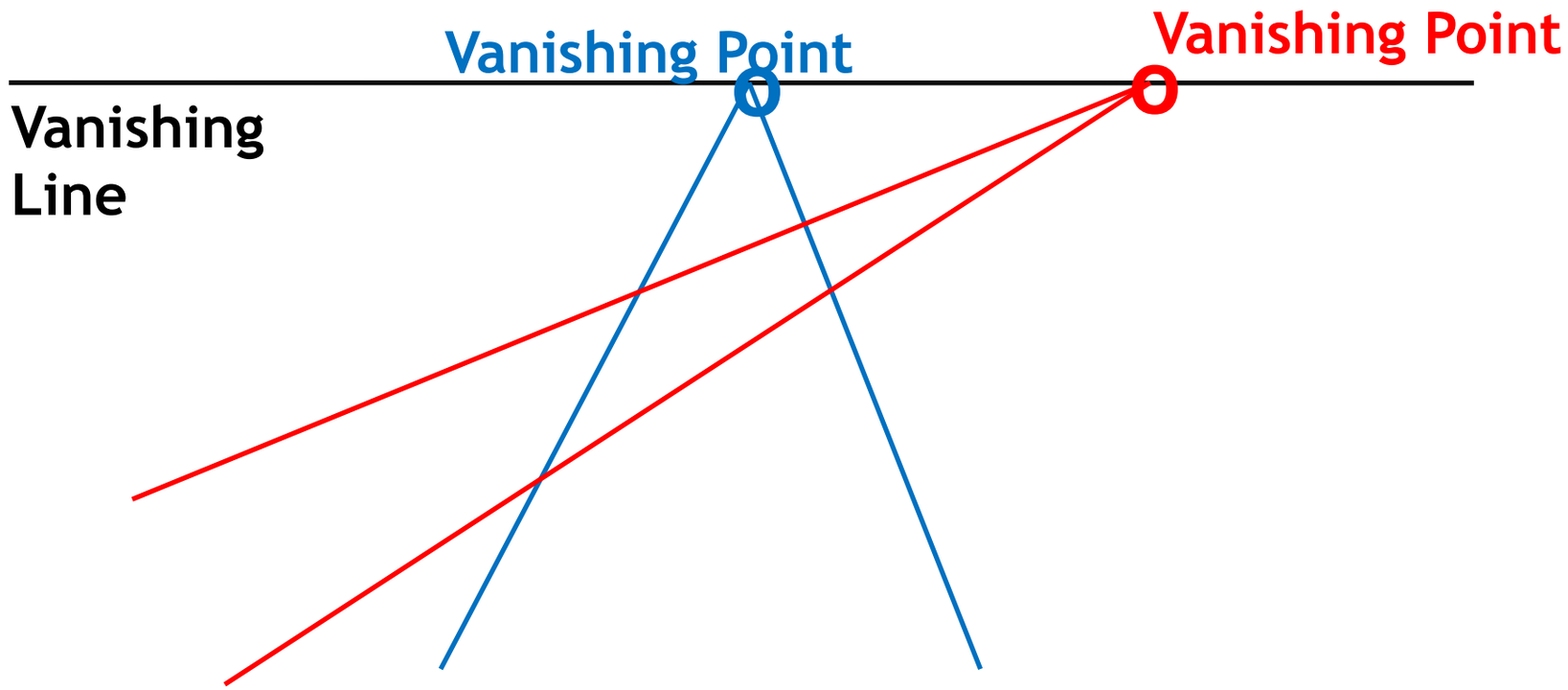


Slide by Steve Seitz

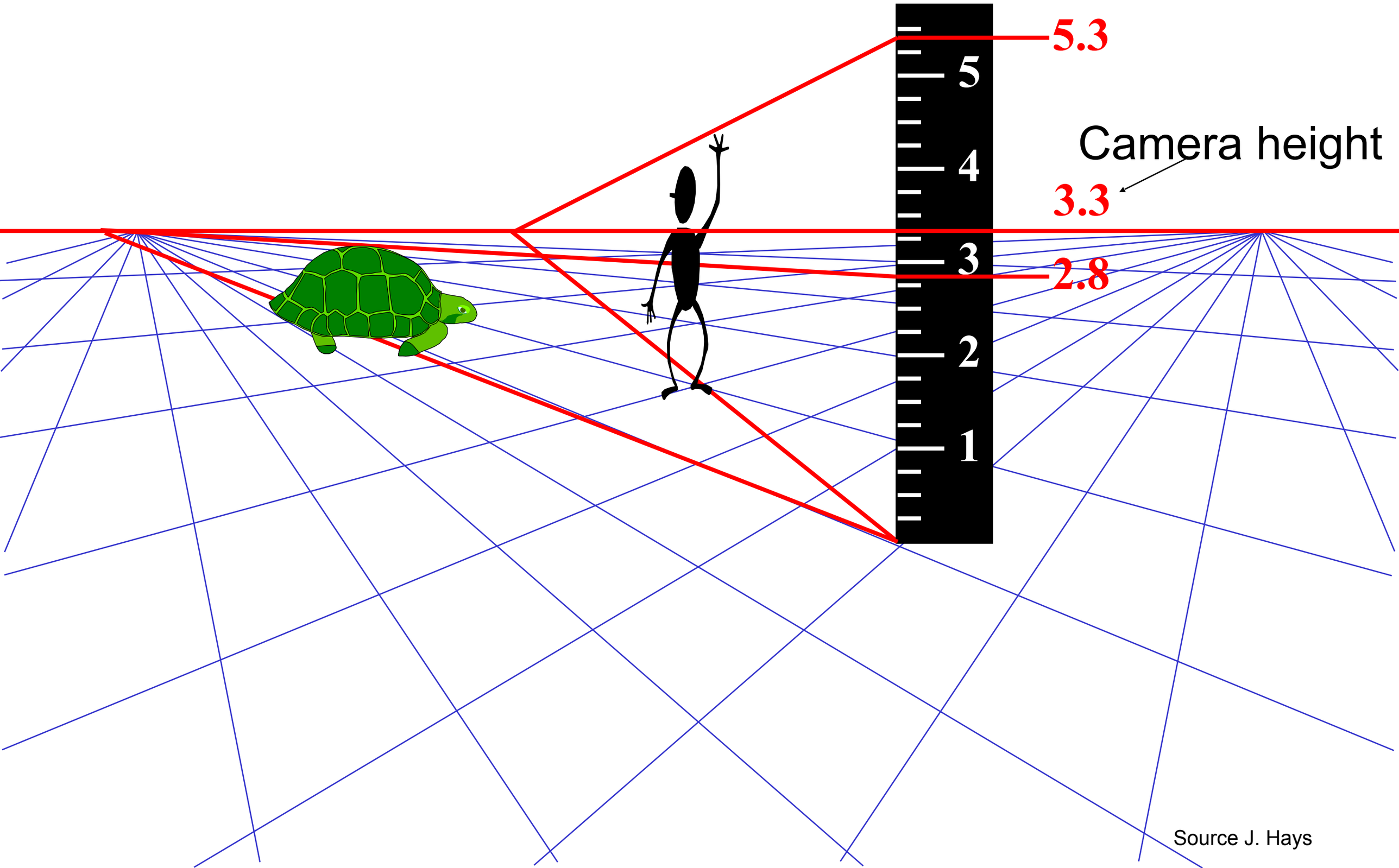
Perspective cues



Vanishing points and lines

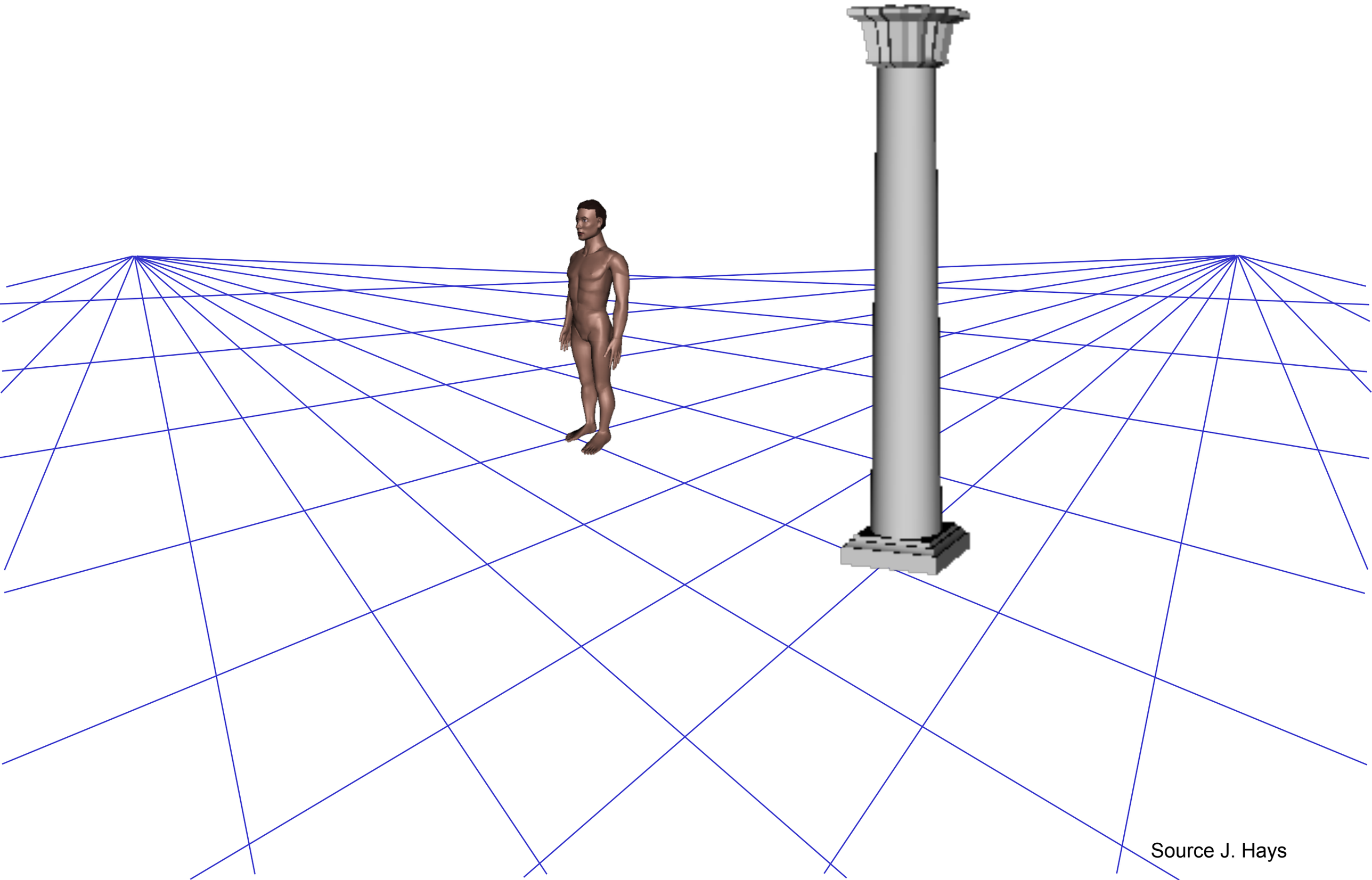


Measuring height

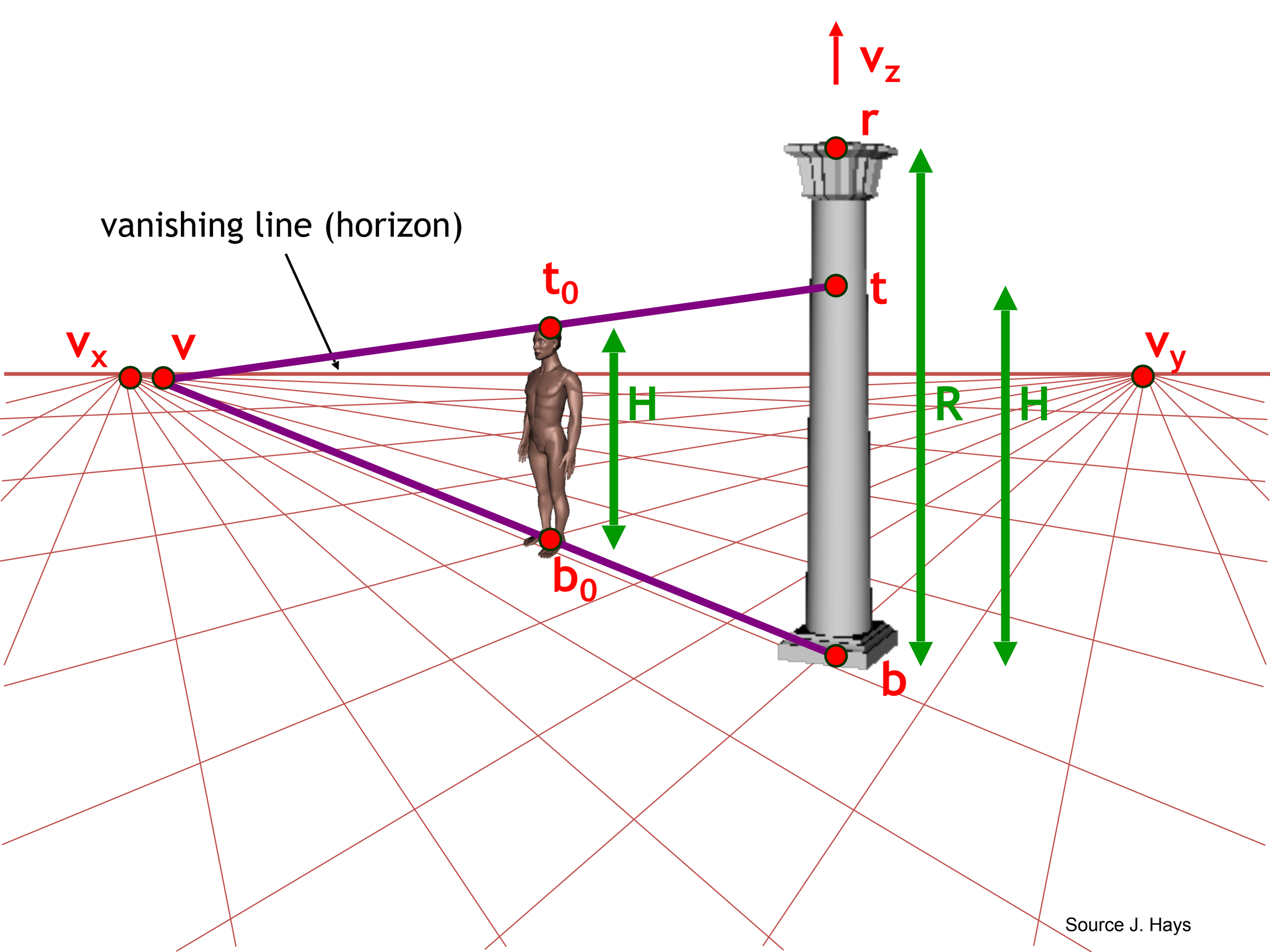


Source J. Hays

Measuring height without a ruler

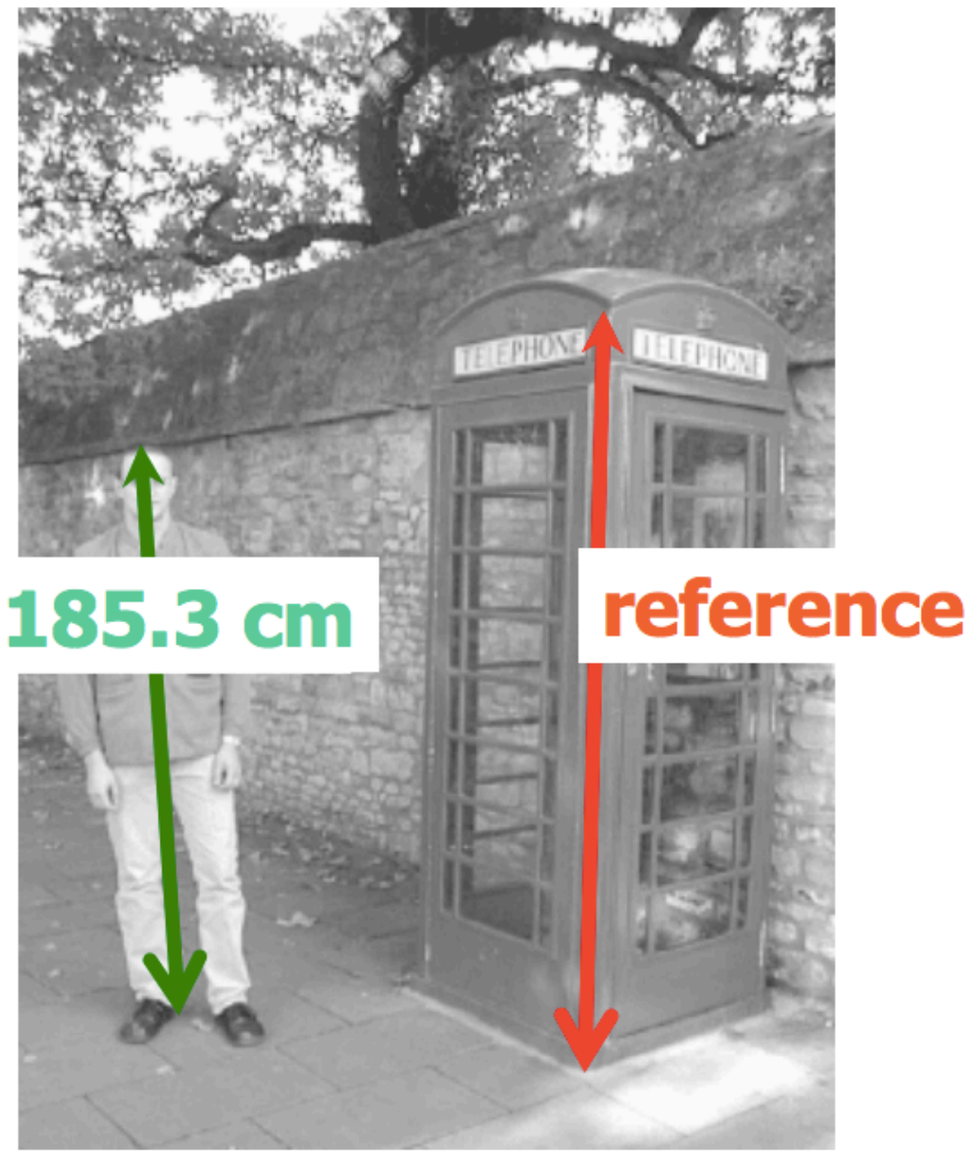


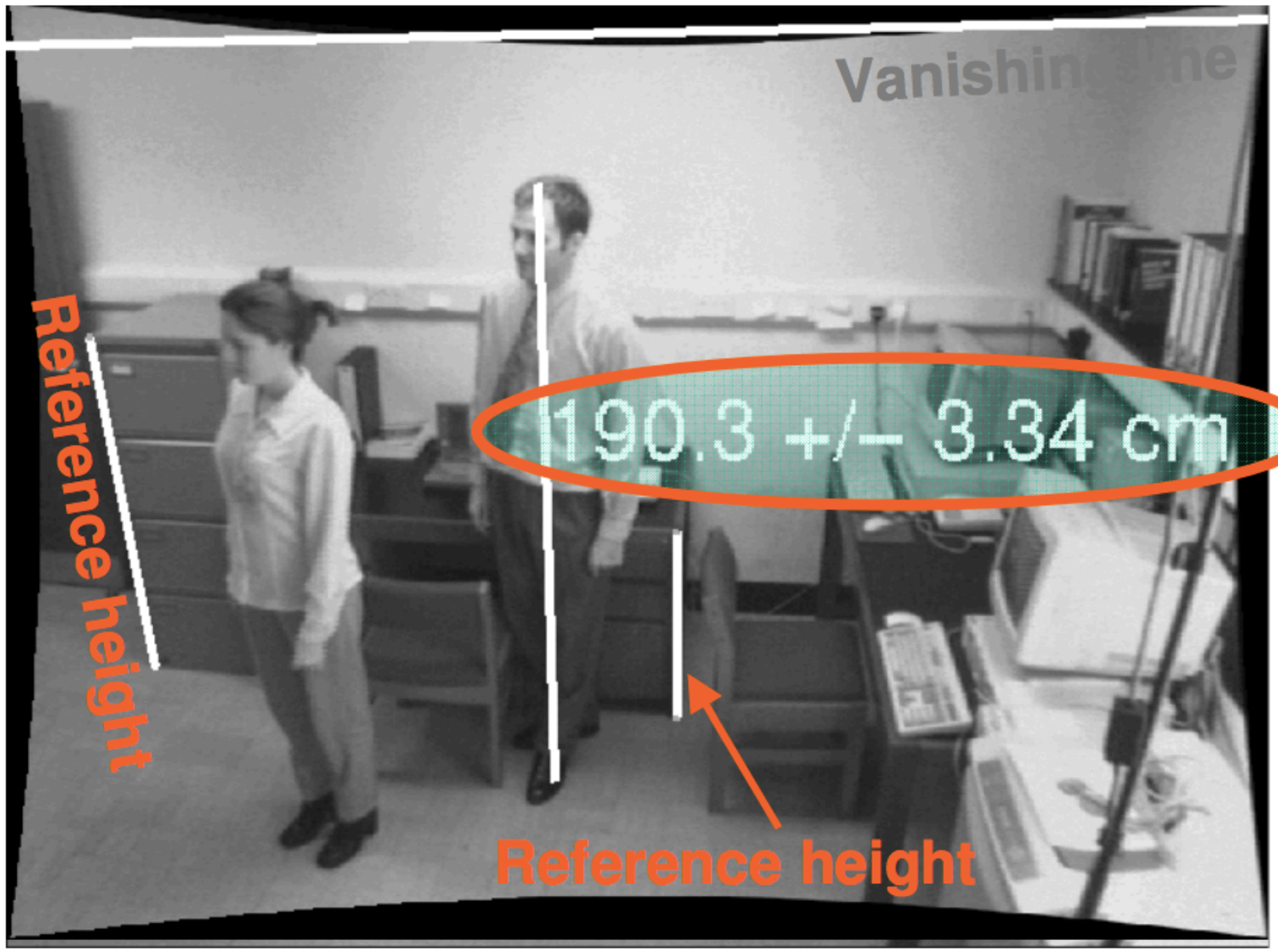
Source J. Hays



Source J. Hays

Examples





A. Criminisi, I. Reid, and A. Zisserman, Single View Metrology, IJCV 2000
Figure from UPenn CIS580 slides

Source S. Lazebnik

Another example

- Are the heights of the two groups of people consistent with one another?
  - Measure heights using Christ as reference

Piero della Francesca, *Flagellation*, ca. 1455

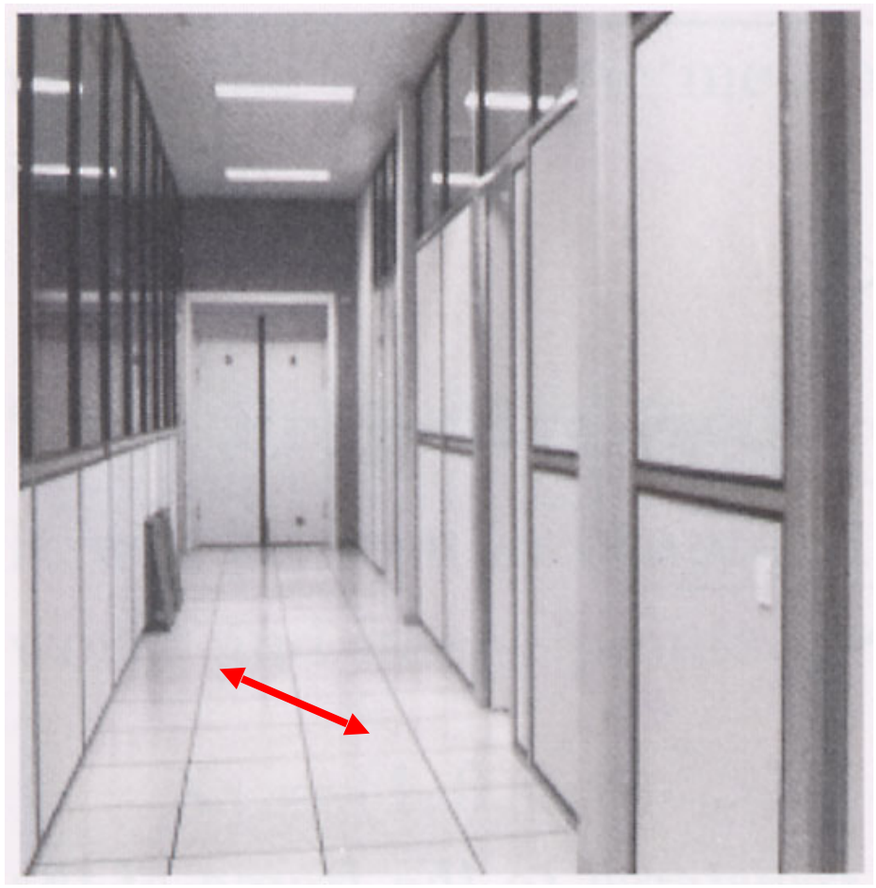
A. Criminisi, M. Kemp, and A. Zisserman, Bringing Pictorial Space to Life: computer techniques for the analysis of paintings,
Proc. Computers and the History of Art, 2002

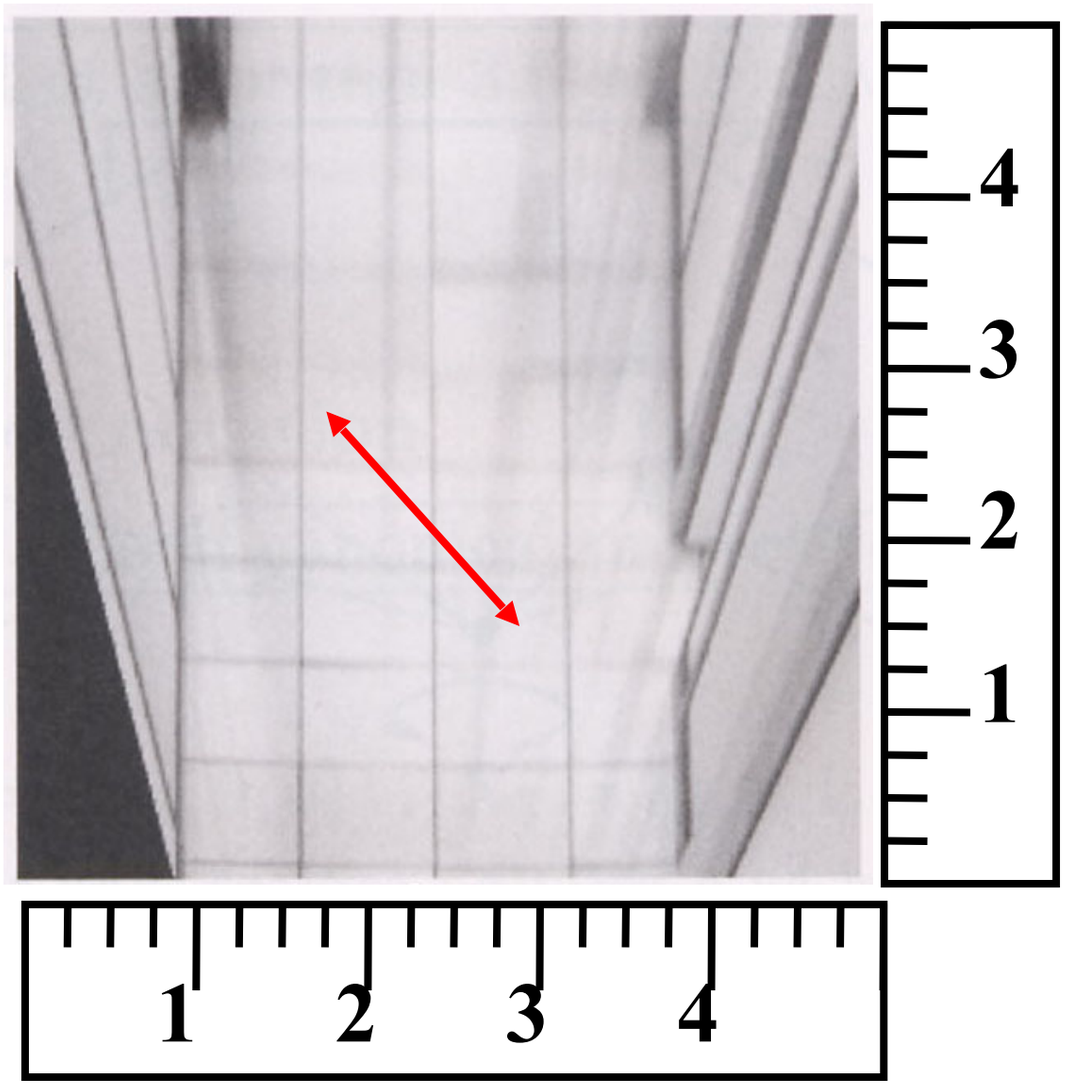
Source S. Lazebnik

Which is higher – the camera or the man in the parachute?

Source J. Hays

Measurements on planes



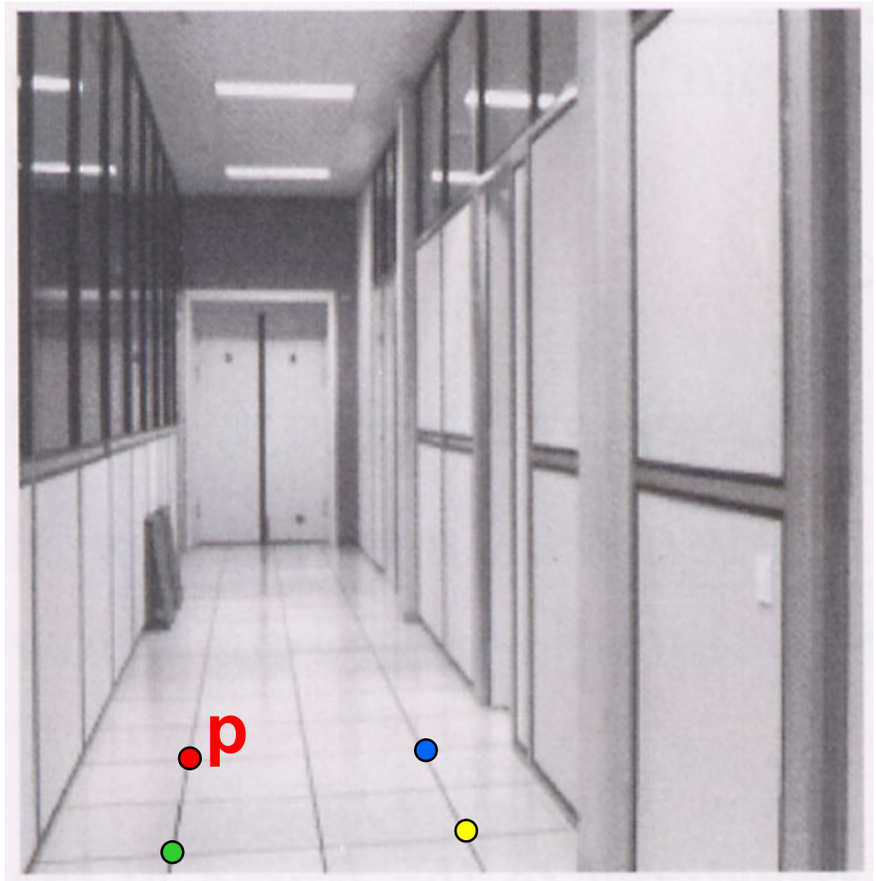


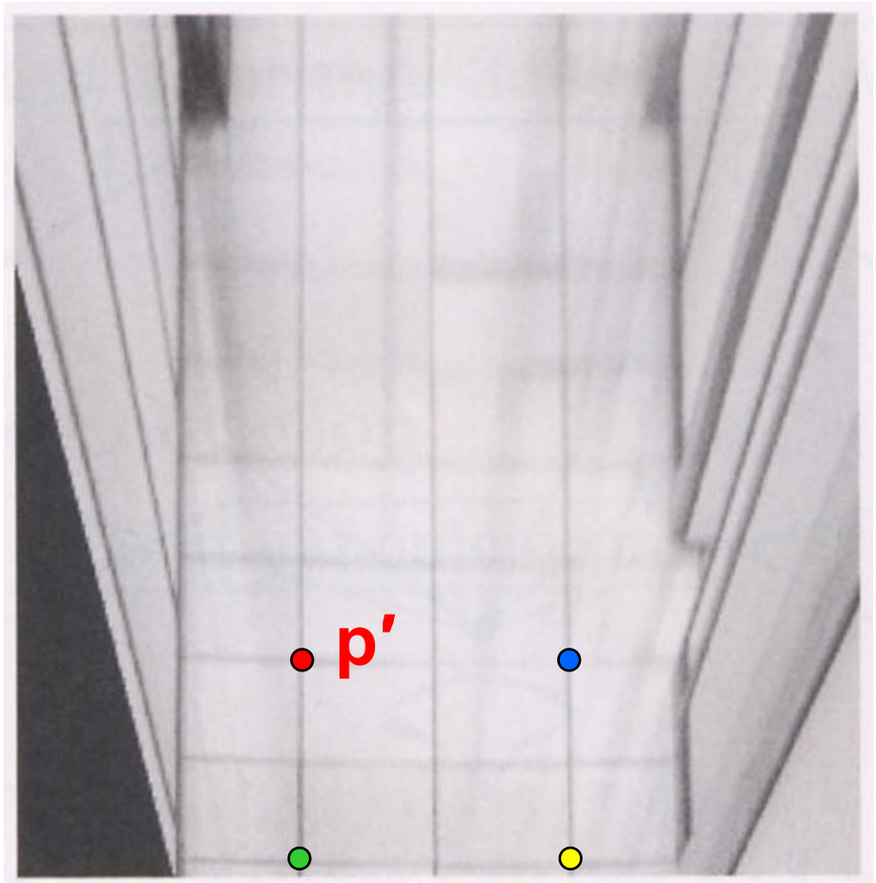
Approach: unwarp then measure

What kind of warp is this?

Source S. Lazebnik

Image rectification





To unwarp (rectify) an image

- solve for homography **H** given **p** and **p′**
  - how many points are necessary to solve for **H**?

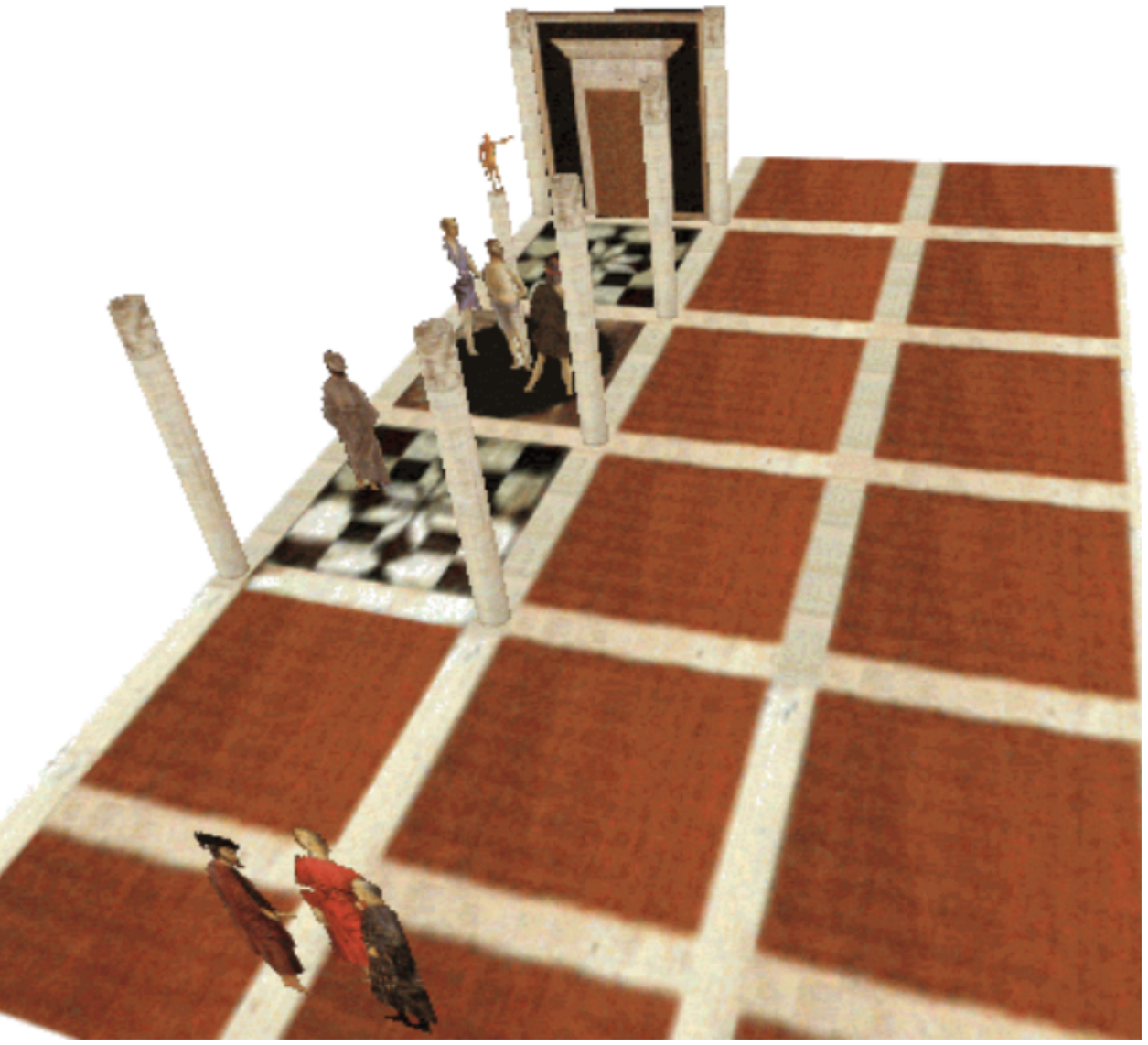
Source S. Lazebnik

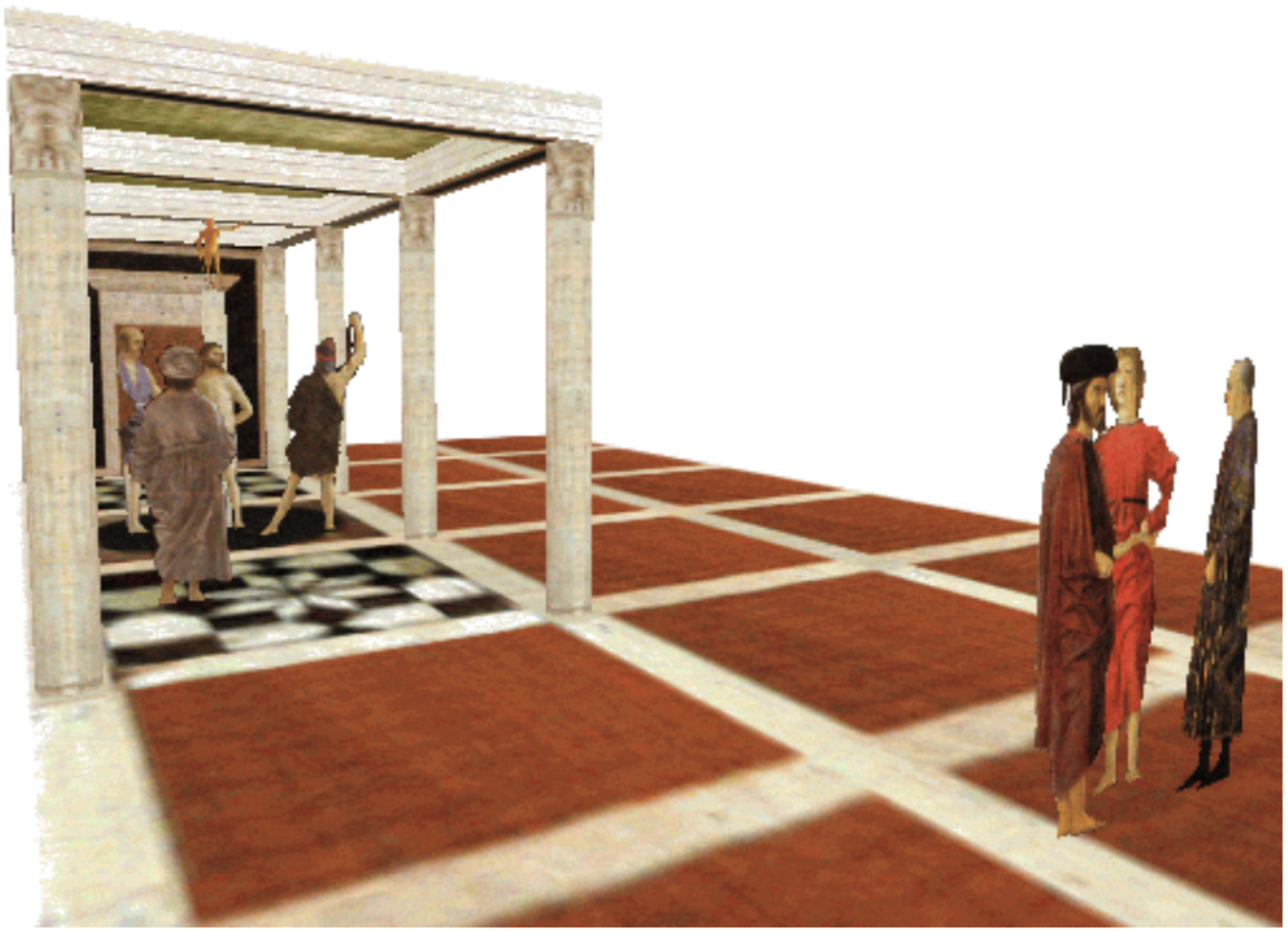
Image rectification: example

Piero della Francesca, *Flagellation*, ca. 1455

Source S. Lazebnik

Application: 3D modeling from a single image





A. Criminisi, M. Kemp, and A. Zisserman,[Bringing Pictorial Space to Life: computer techniques for the analysis of paintings],
Proc. Computers and the History of Art, 2002

Source S. Lazebnik

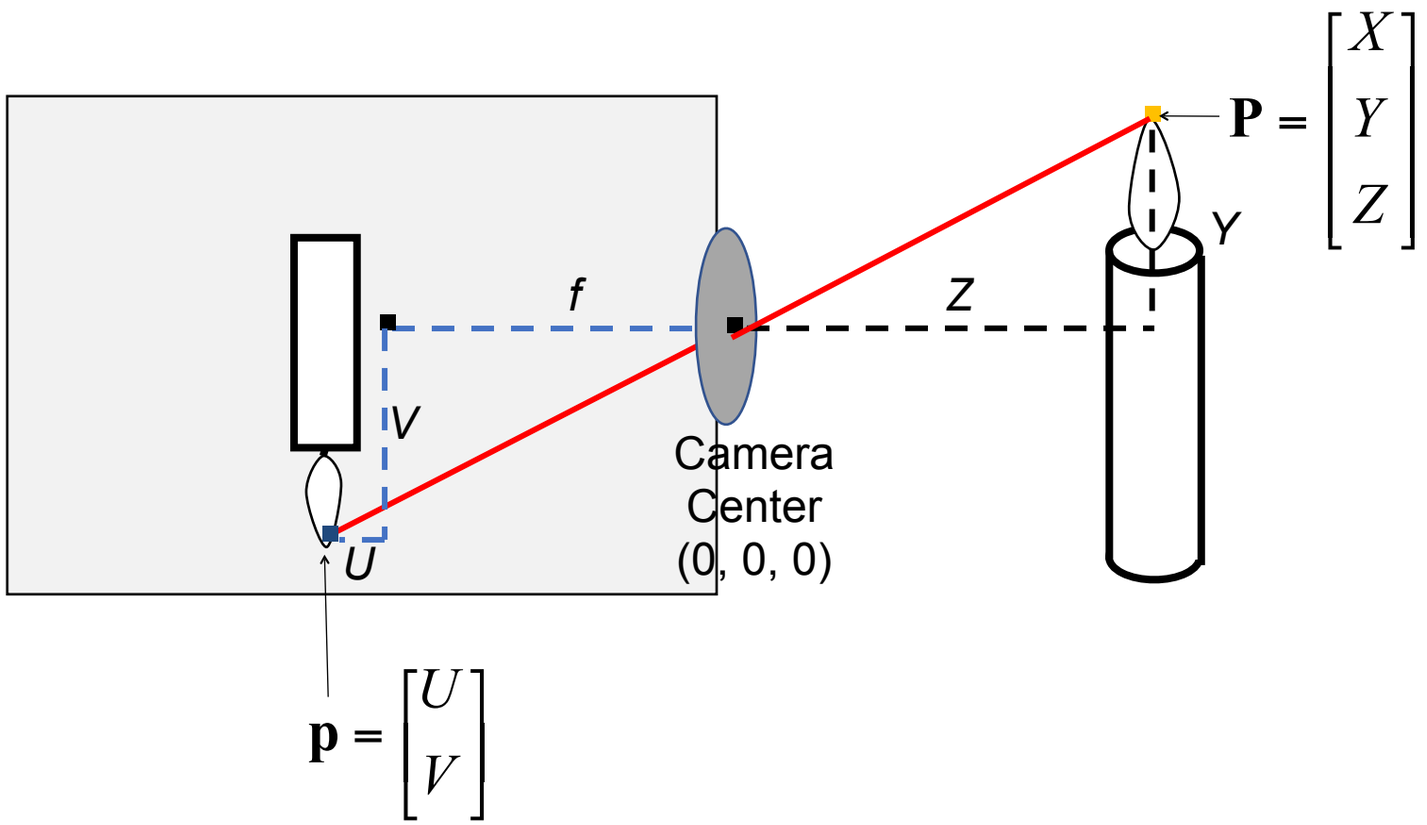
Application: 3D modeling from a single image

J. Vermeer, *Music Lesson*, 1662

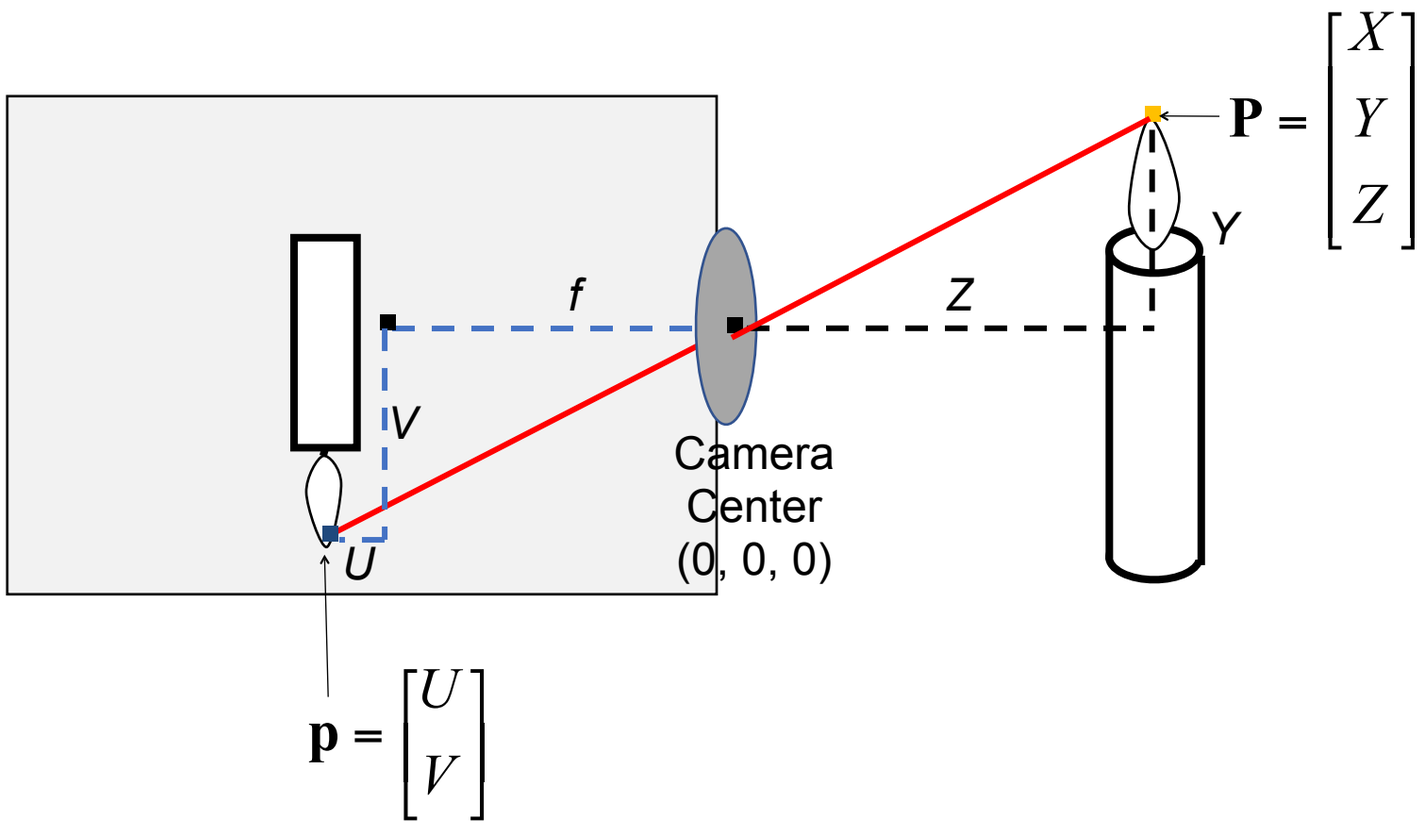
A. Criminisi, M. Kemp, and A. Zisserman, Bringing Pictorial Space to Life: computer techniques for the analysis of paintings,
Proc. Computers and the History of Art, 2002

Source S. Lazebnik

Pinhole cameras again



Pinhole cameras again

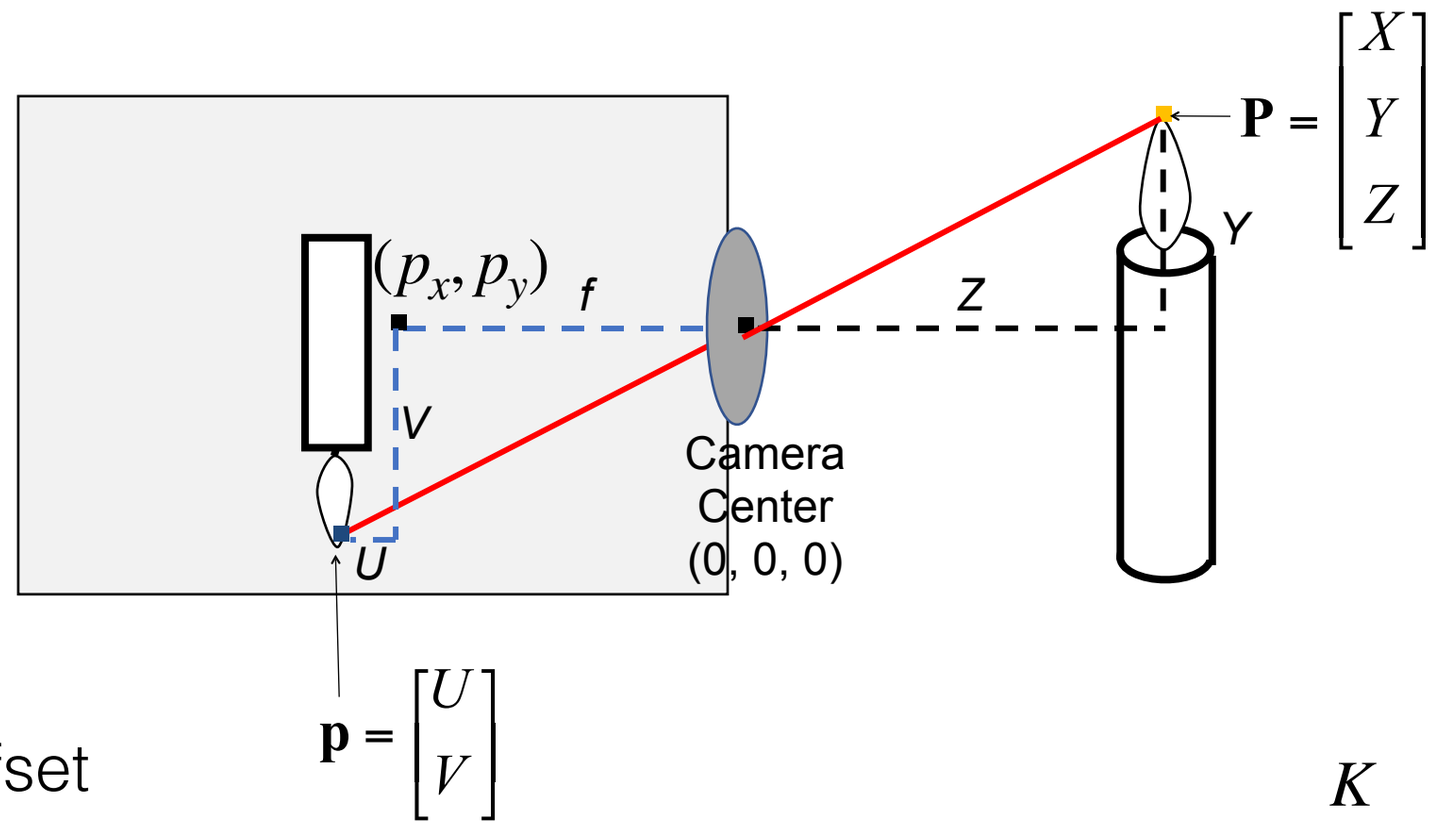


$$U = f\frac{X}{Z}$$

$$V = f\frac{Y}{Z}$$

$$\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Pinhole cameras again

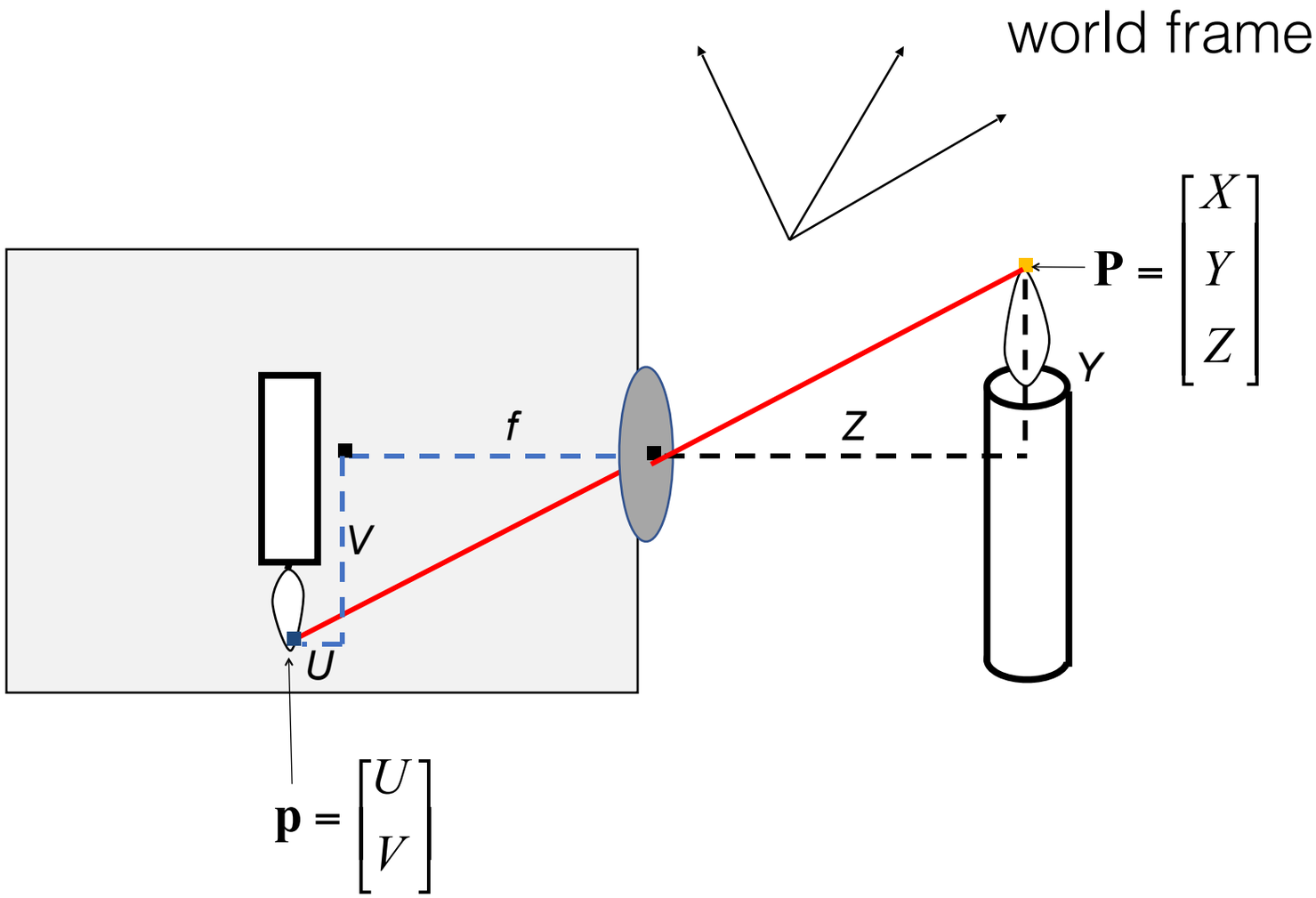


principal point offset

$$U = f\frac{X}{Z} + p_x$$
$$V = f\frac{Y}{Z} + p_y$$

$$\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & p_1 \end{bmatrix}}_{K} [Id|0] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Pinhole cameras again



$$P_{cam} = RP + t \longrightarrow \begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = K \begin{bmatrix} Id \mid 0 \end{bmatrix} \begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{bmatrix} = K[R \mid t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

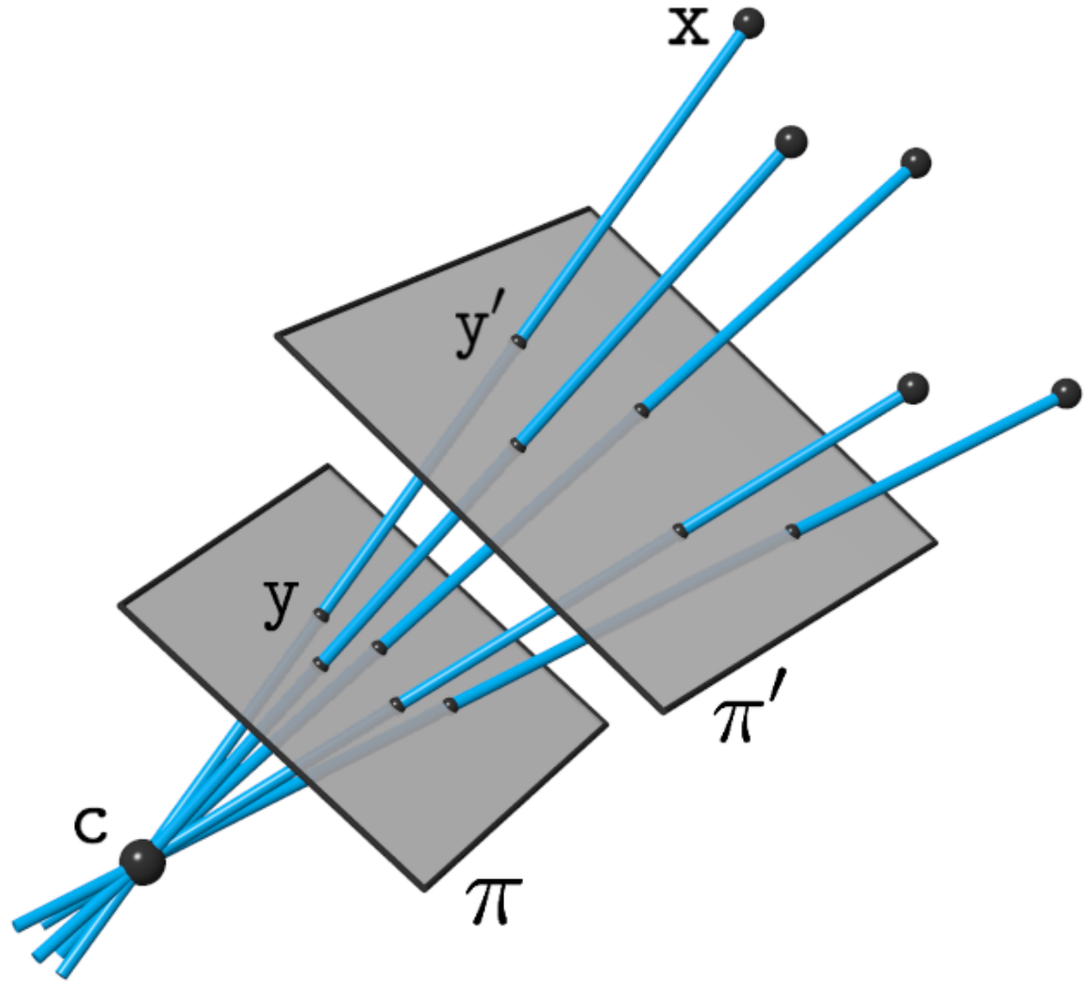
Pinhole camera model

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \underbrace{\begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}}_{[R\,|\,t]} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- 11 total parameters: 5 intrinsic + 6 extrinsic
- $M = K[R\,|\,t]$ is a 3x4 matrix
- Fact: $M \in \mathbb{R}^{3\times4}$ is a valid camera iff left 3x3 block is invertible.
- Projective camera model: 3x4 matrix of full rank.

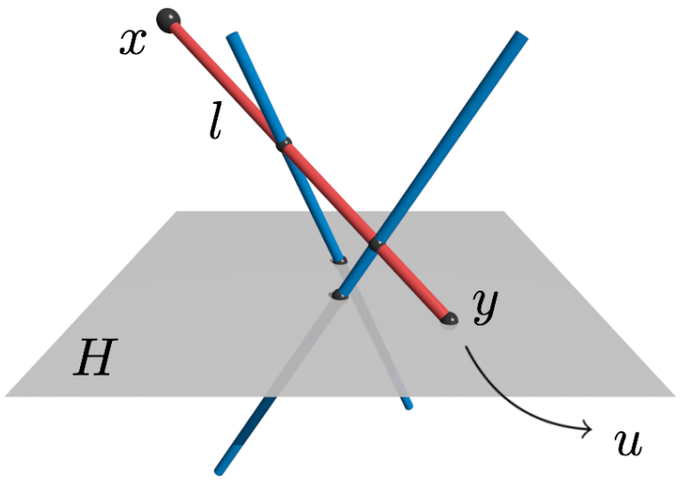
Geometry of pinhole cameras

- The viewing lines associated with a pinhole camera all pass through the pinhole.
- The retinal plane (and image coordinates) are less important than the pinhole.
- The recorded lines form a two-dimensional family.

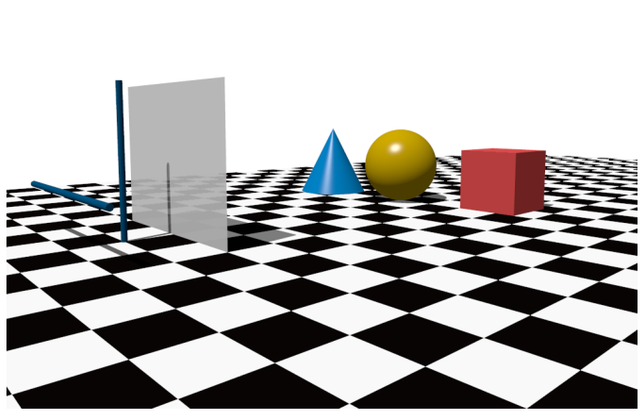


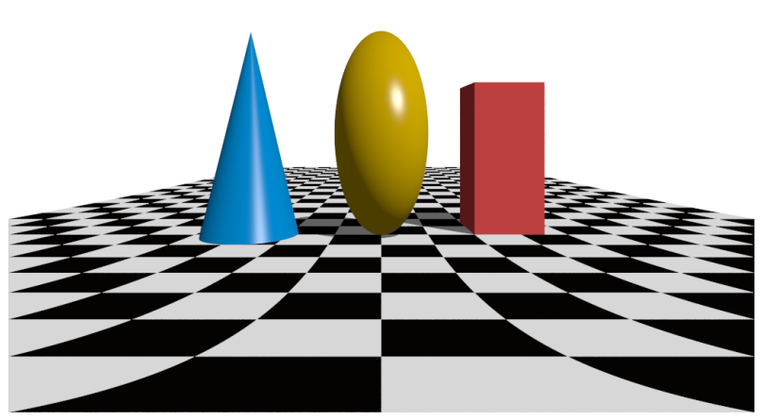
Exotic cameras

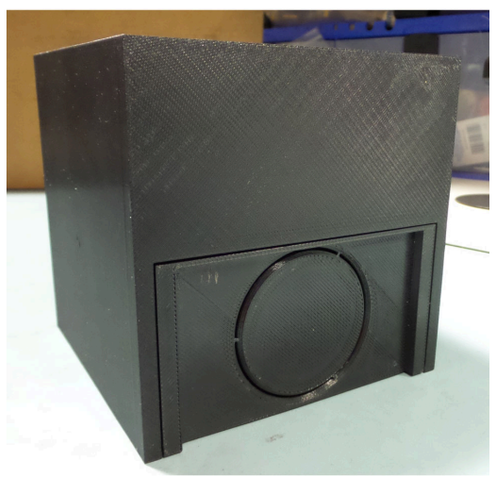
- A camera is a machine that records light and creates 2D picture of a 3D world.
- It “samples” a two-dimensional set of lines.

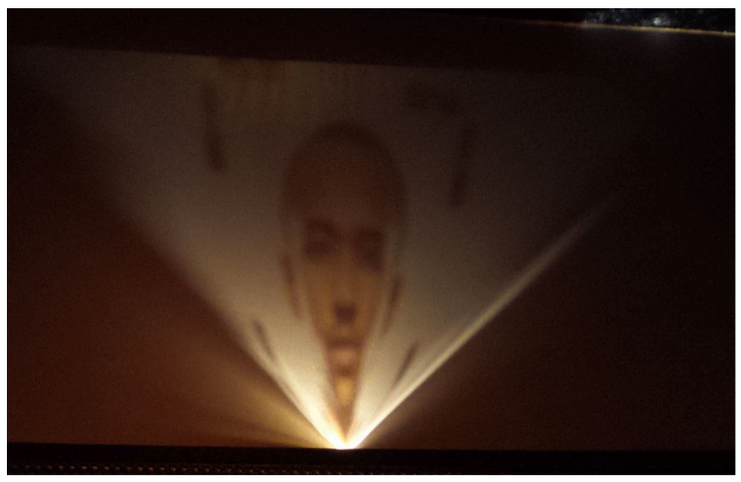


“Two-slit camera”









Thinking in projective space

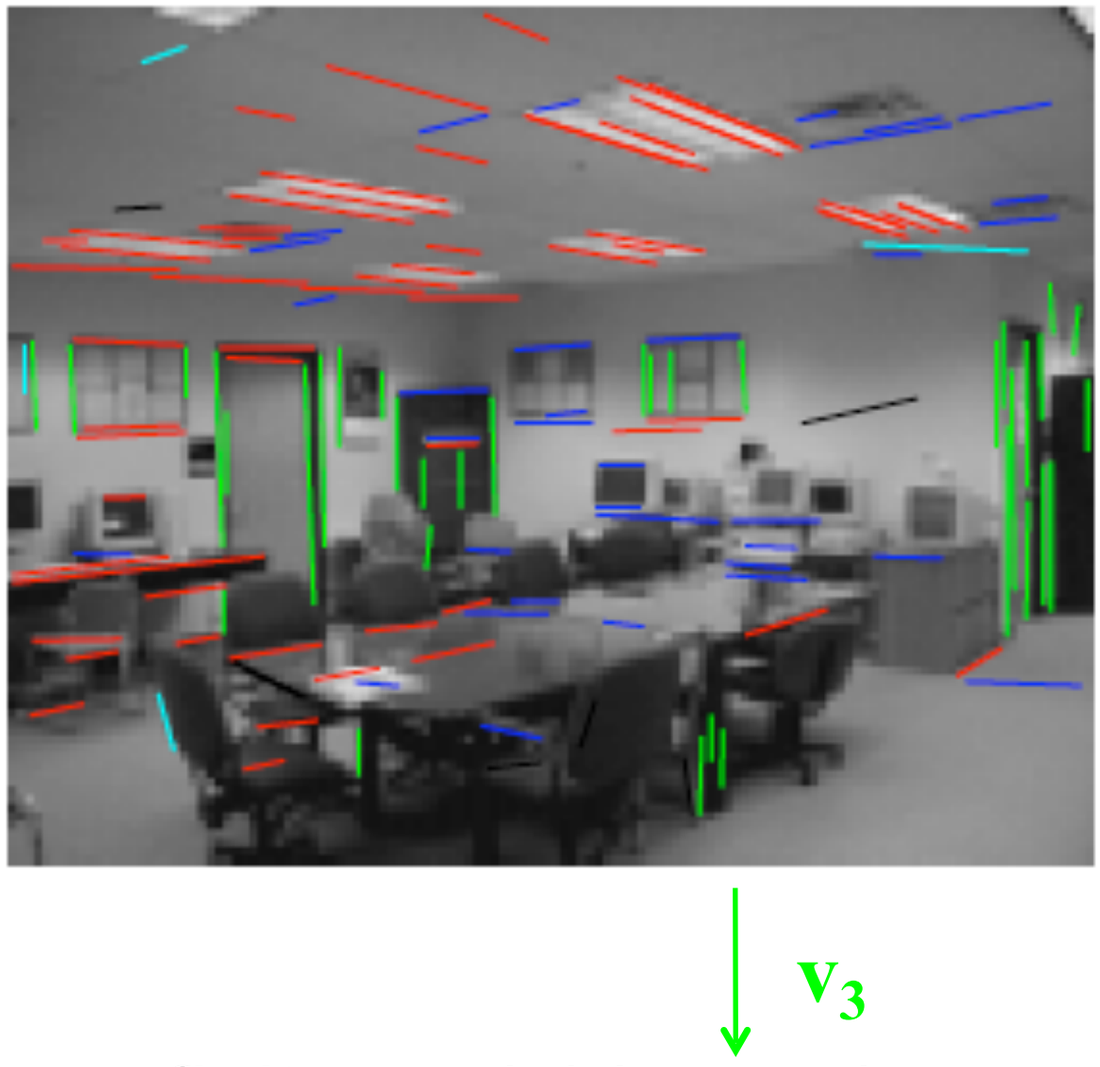
- It is often convenient to “forget” that the world is Euclidean (simplifies calculations and removes assumptions).
- Projective reconstruction yields parameters of $\boldsymbol{M} \in \mathbb{R}^{3\times4}$ and 3D reconstruction up to projective transformations.

Hartley Zisserman 04

- To “upgrade” a transformation, we need the camera’s internal parameters ($\boldsymbol{K}$ matrix).

Calibration from vanishing points

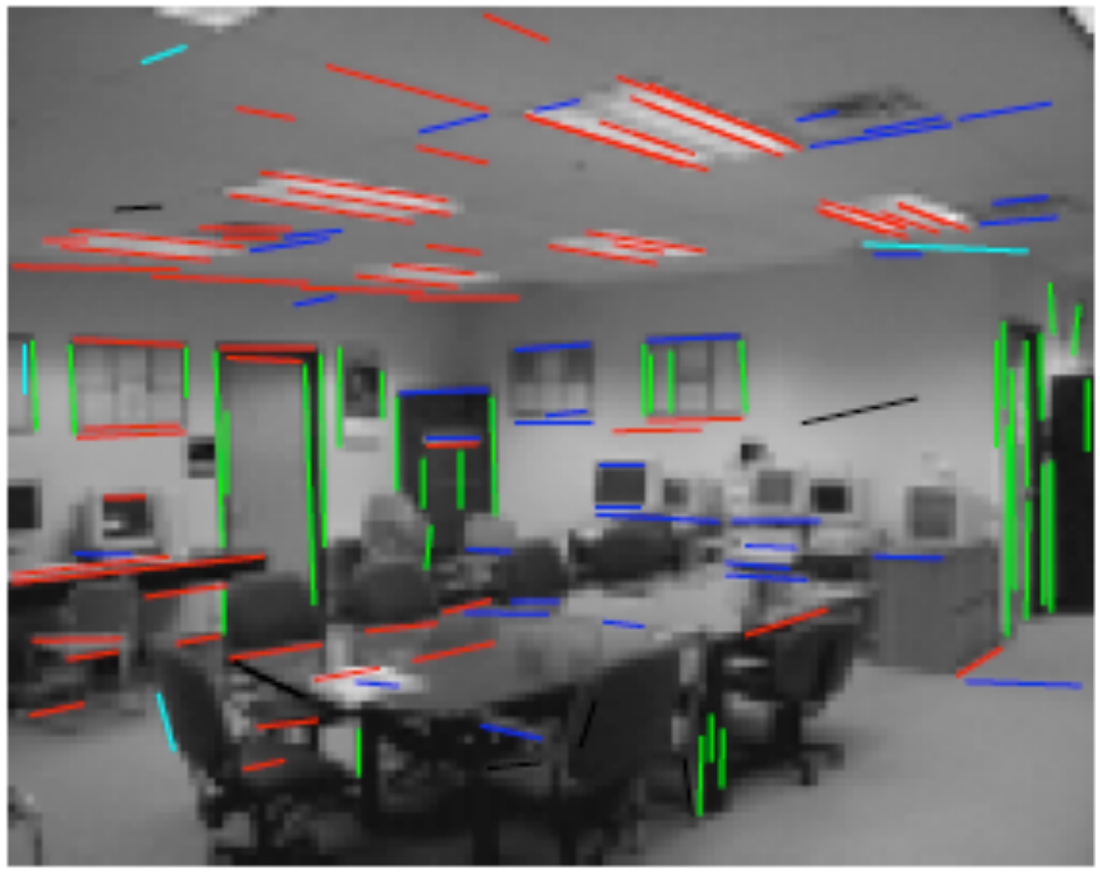
- Consider a scene with three orthogonal vanishing directions:



- Note: $\mathbf{v}_1$, $\mathbf{v}_2$ are *finite* vanishing points and $\mathbf{v}_3$ is an *infinite* vanishing point

Calibration from vanishing points

- Consider a scene with three orthogonal vanishing directions:



- We can align the world coordinate system with these directions

Calibration from vanishing points

- Let us align the world coordinate system with three orthogonal vanishing directions in the scene:

$$\mathbf{e}_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}, \quad \mathbf{e}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}, \quad \mathbf{e}_3 = \begin{bmatrix}0\\0\\1\end{bmatrix} \qquad \lambda_i \mathbf{v}_i = \mathbf{K}\begin{bmatrix}\mathbf{R} \mid \mathbf{t}\end{bmatrix}\begin{bmatrix}\mathbf{e}_i\\0\end{bmatrix} = \mathbf{K}\mathbf{R}\mathbf{e}_i$$

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

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

- Each pair of vanishing points gives us a constraint on the focal length and principal point.

Single-view geometry summary

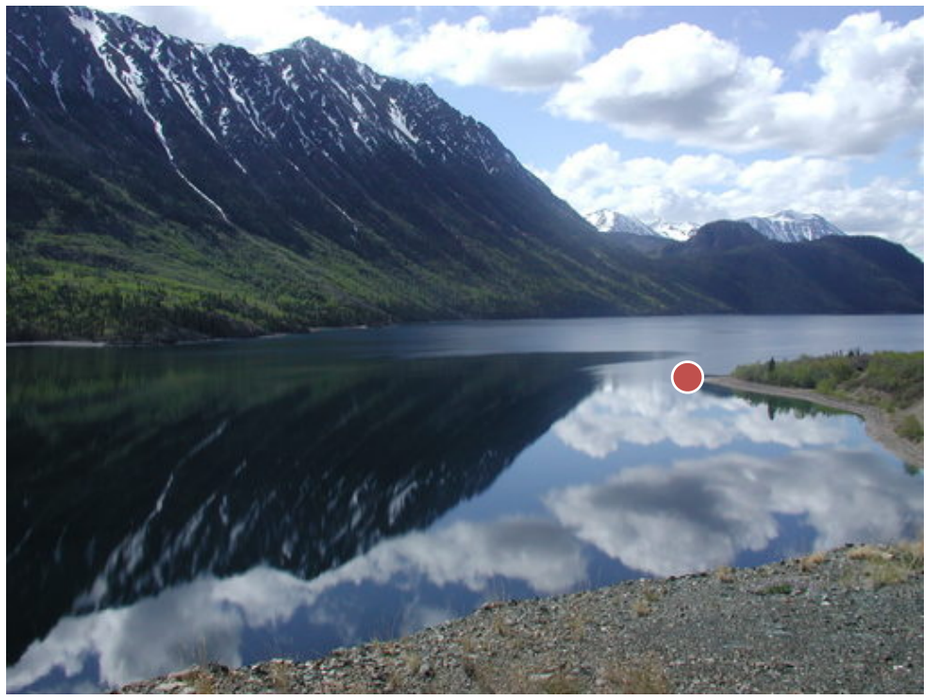
- Projective geometry is a natural language for vision (angles and distances not preserved in images).
- Vanishing points are projections of points at infinity.
- Pinhole camera model is simpler in projective setting.
- Orthogonal vanishing points can be used to recover calibration matrix.

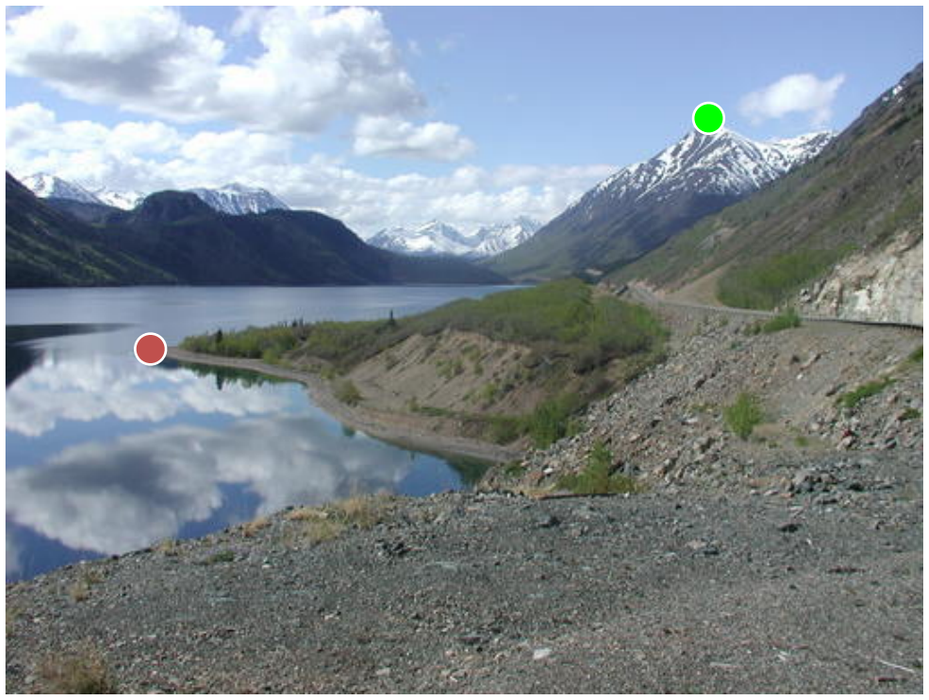
Image Stitching

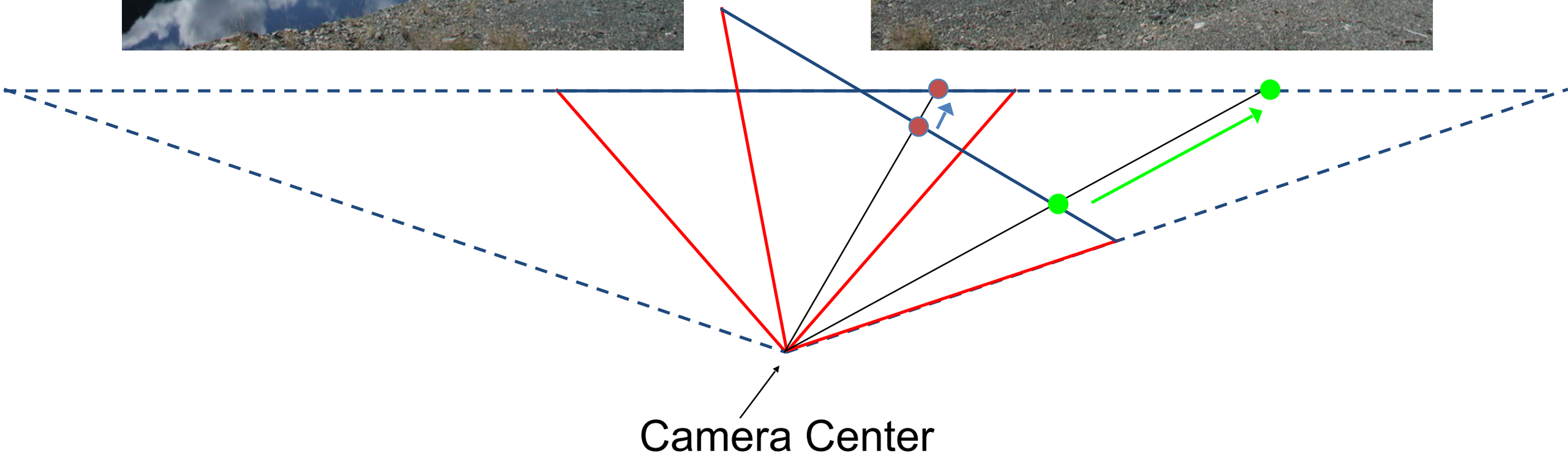
- Combine two or more overlapping images to make one larger image

Slide credit: Vaibhav Vaish

Illustration



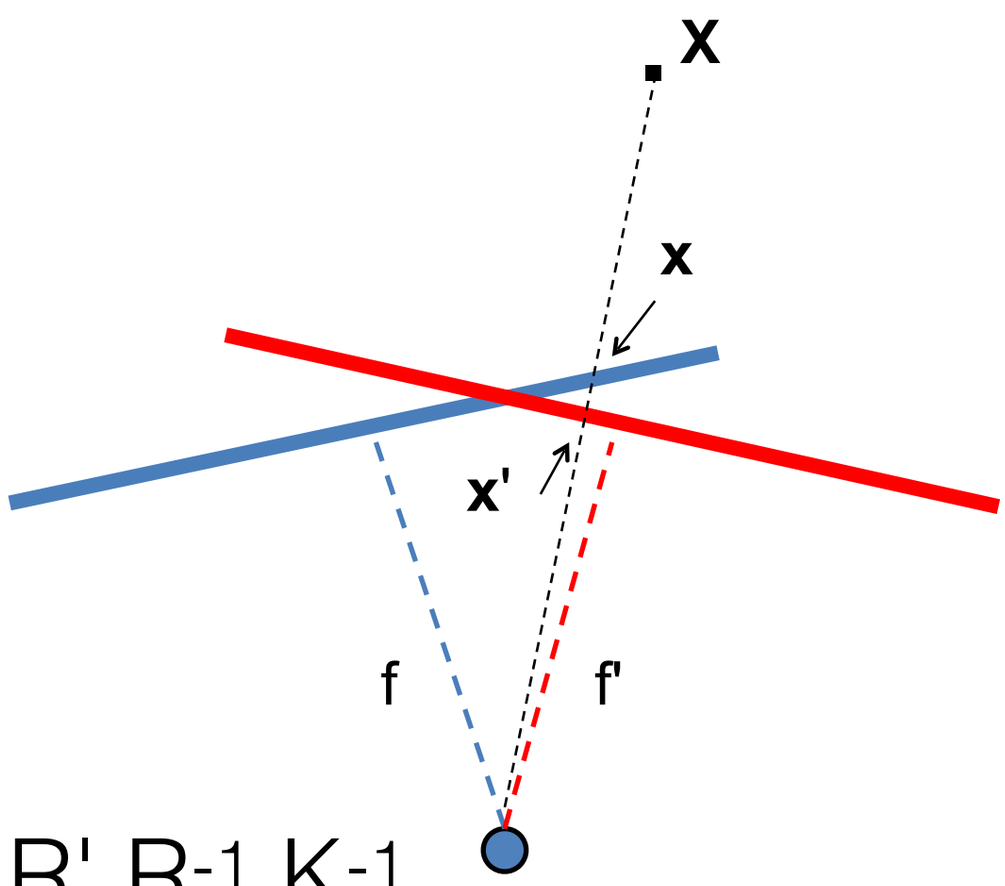




Following slides from D. Hoiem

Problem set-up

- x = K [R t] X
- x' = K' [R' t'] X
- t=t'=0



- x'=Hx where $H = K' R' R^{-1} K^{-1}$
- Typically only R and f will change (4 parameters), but, in general, H has 8 parameters

Image Stitching Algorithm Overview

1. Detect keypoints (e.g., SIFT)
2. Match keypoints (e.g., $1^{st}/2^{nd}$ NN < thresh)
3. Estimate homography with four matched keypoints (using RANSAC)
4. Combine images

Computing homography

Assume we have four matched points: How do we compute the homography **H**?

Direct Linear Transformation (DLT)

$$\mathbf{x}' \sim \mathbf{H}\mathbf{x} \qquad \mathbf{x} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \qquad \mathbf{x}' = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}$$

$$\begin{bmatrix} -u & -v & -1 & 0 & 0 & 0 & uu' & vu' & u' \\ 0 & 0 & 0 & -u & -v & -1 & uv' & vv' & v' \end{bmatrix} \mathbf{h} = \mathbf{0} \qquad \mathbf{h} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix}$$

Computing homography

Direct Linear Transform

$$\begin{bmatrix} -u_1 & -v_1 & -1 & 0 & 0 & 0 & u_1u'_1 & v_1u'_1 & u'_1 \\ 0 & 0 & 0 & -u_1 & -v_1 & -1 & u_1v'_1 & v_1v'_1 & v'_1 \\ & & & \vdots & & & & & \\ 0 & 0 & 0 & -u_n & -v_n & -1 & u_nv'_n & v_nv'_n & v'_n \end{bmatrix} \mathbf{h} = \mathbf{0} \Rightarrow \mathbf{Ah} = \mathbf{0}$$

- Apply SVD: $\boldsymbol{UDV}^T = \boldsymbol{A}$
- $\boldsymbol{h} = \boldsymbol{V}_{\text{smallest}}$ (column of $\boldsymbol{V}$ corr. to smallest singular value)

Computing homography

- Assume we have four matched points: How do we compute homography **H**?

Normalized DLT

1. Normalize coordinates for each image
   a) Translate for zero mean
   b) Scale so that average distance to origin is ~sqrt(2)

$$\tilde{\mathbf{x}} = \mathbf{T}\mathbf{x} \qquad \tilde{\mathbf{x}}' = \mathbf{T}'\mathbf{x}'$$

   - This makes problem better behaved numerically (see HZ p. 107-108)
2. Compute $\tilde{\mathbf{H}}$ using DLT in normalized coordinates
3. Unnormalize: $\mathbf{H} = \mathbf{T}'^{-1}\tilde{\mathbf{H}}\mathbf{T}$

$$\mathbf{x}'_i = \mathbf{H}\mathbf{x}_i$$

Computing homography

- Assume we have matched points with outliers: How do we compute homography **H**?

Automatic Homography Estimation with RANSAC

1. Choose number of samples ***N***
2. Choose 4 random potential matches
3. Compute **H** using normalized DLT
4. Project points from **x** to **x**’ for each potentially matching pair: $\mathbf{x}_i' = \mathbf{H}\mathbf{x}_i$
5. Count points with projected distance < t
   - E.g., t = 3 pixels
6. Repeat steps 2-5 ***N*** times
   - Choose **H** with most inliers

HZ Tutorial ‘99

Automatic Image Stitching

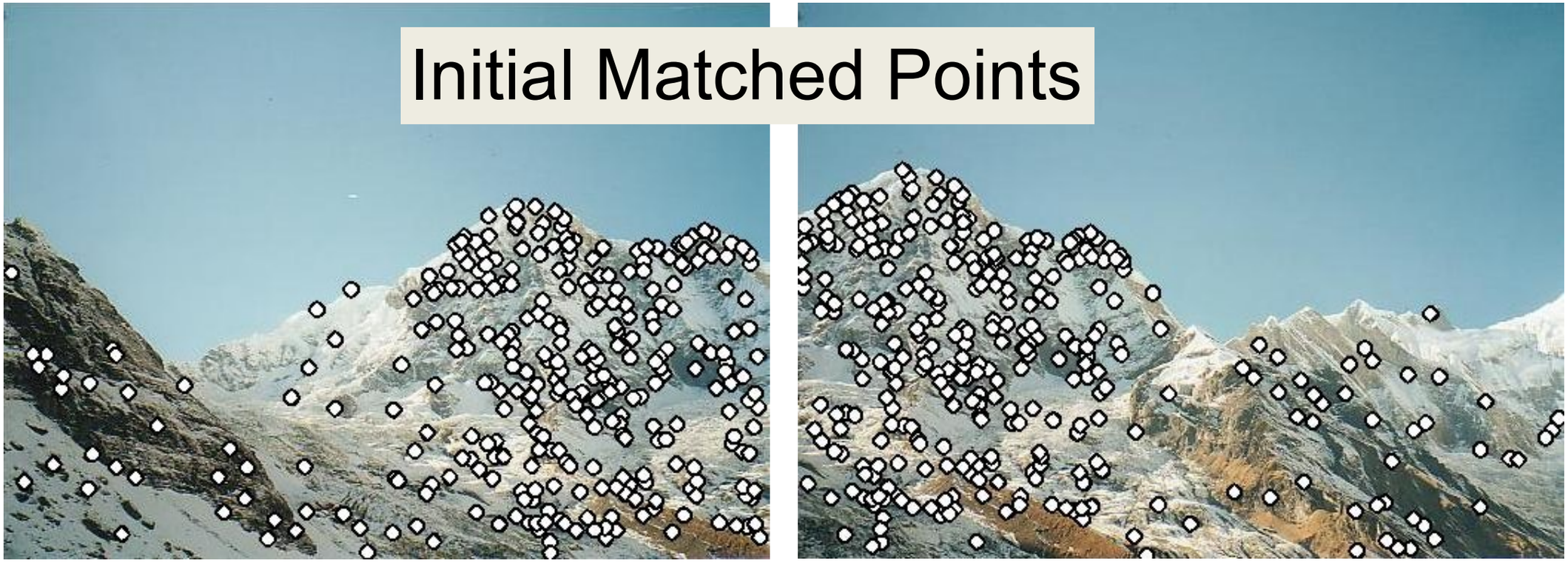
1. Compute interest points on each image

2. Find candidate matches

3. Estimate homography **H** using matched points and RANSAC with normalized DLT

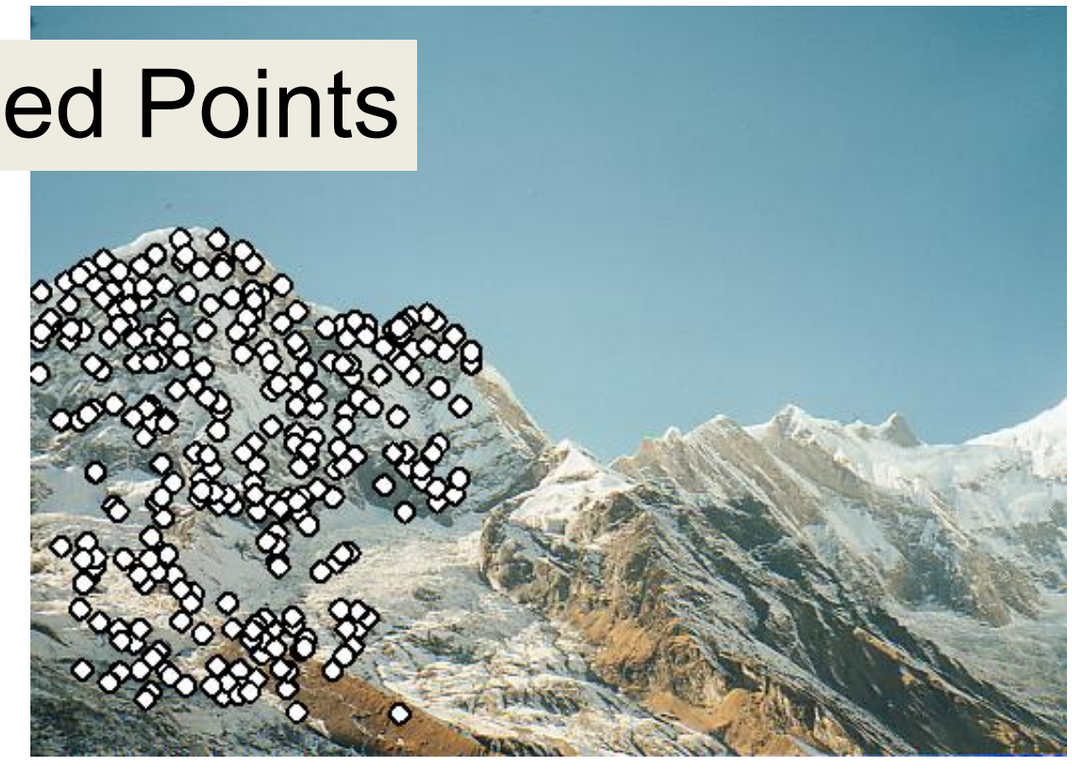
4. Project each image onto the same surface and blend

RANSAC for Homography

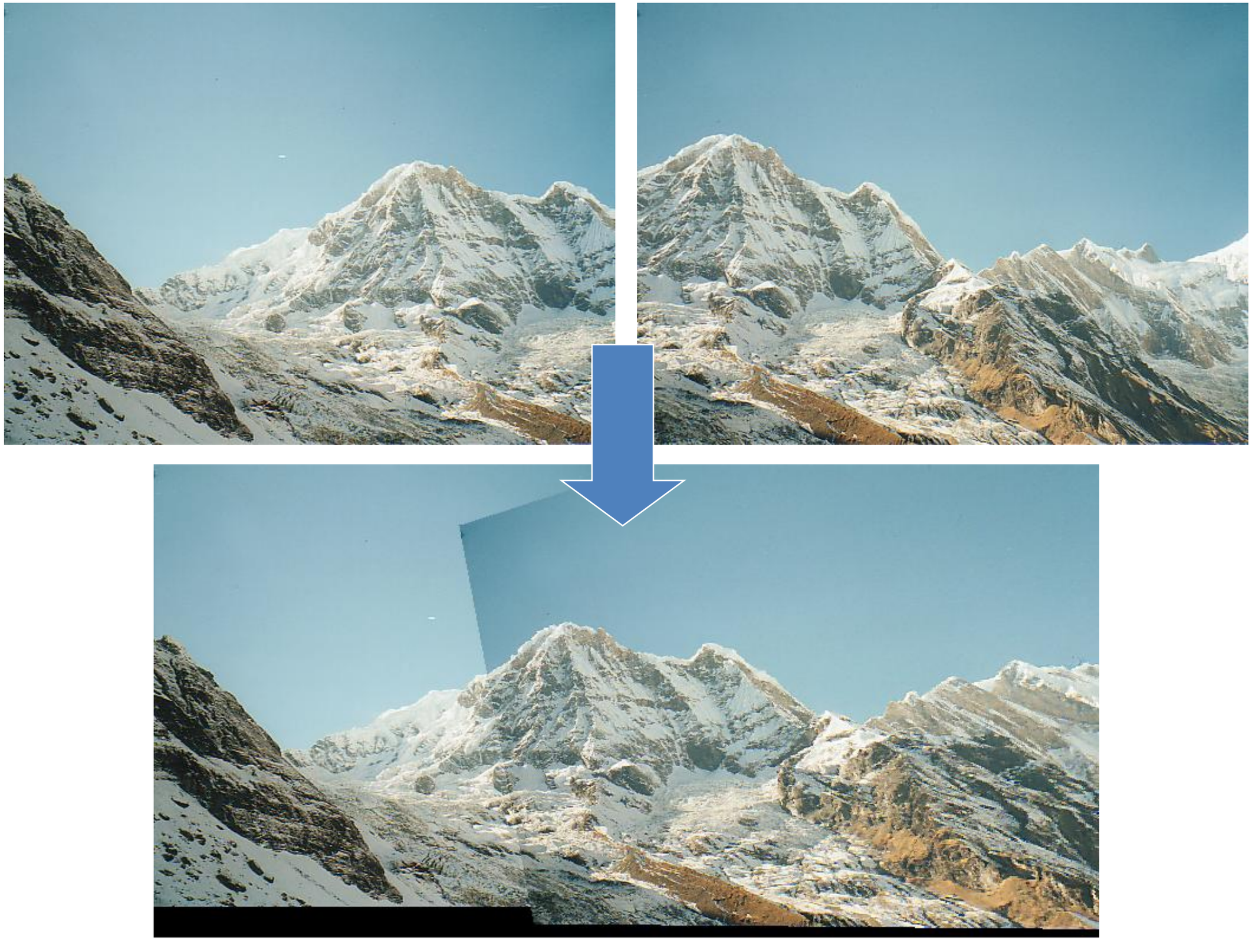


RANSAC for Homography

Final Matched Points

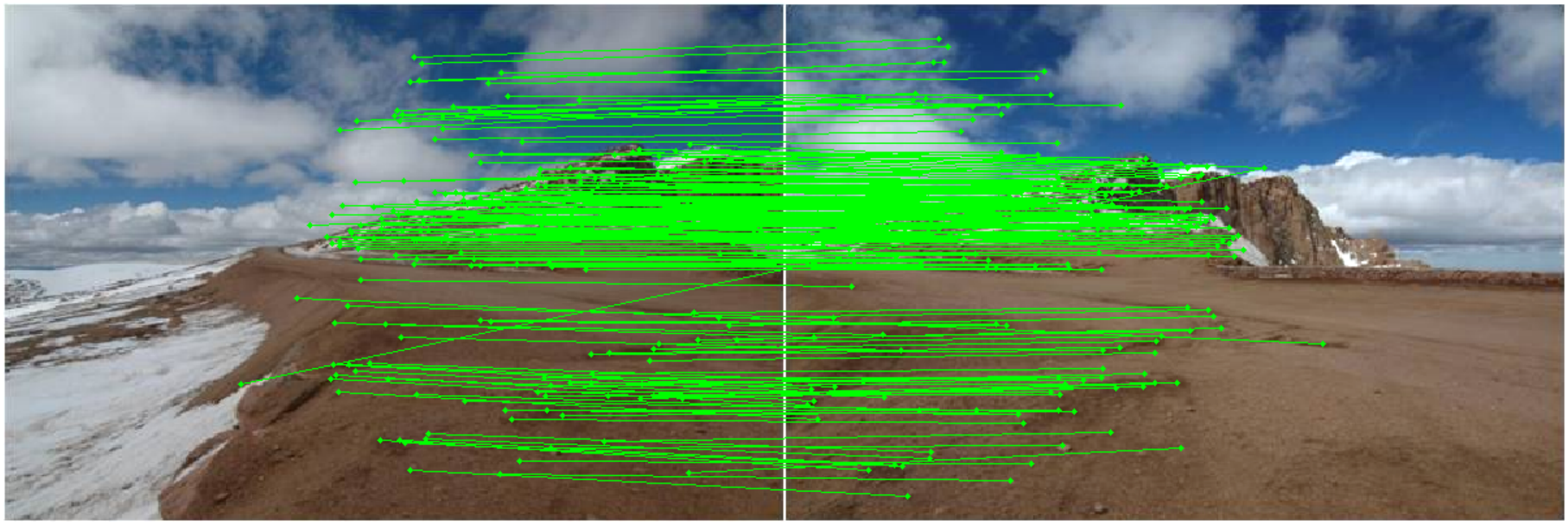


RANSAC for Homography

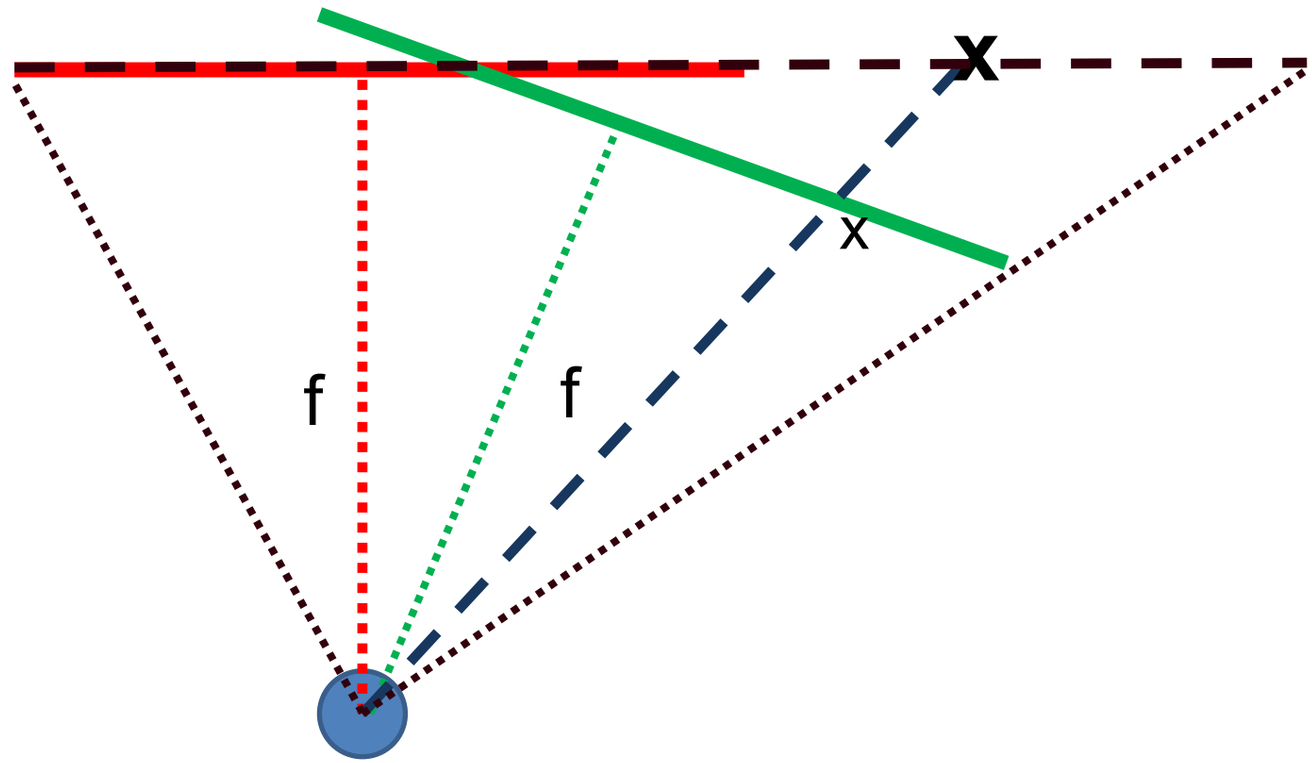


Choosing a Projection Surface

Many to choose: planar, cylindrical, spherical, etc.

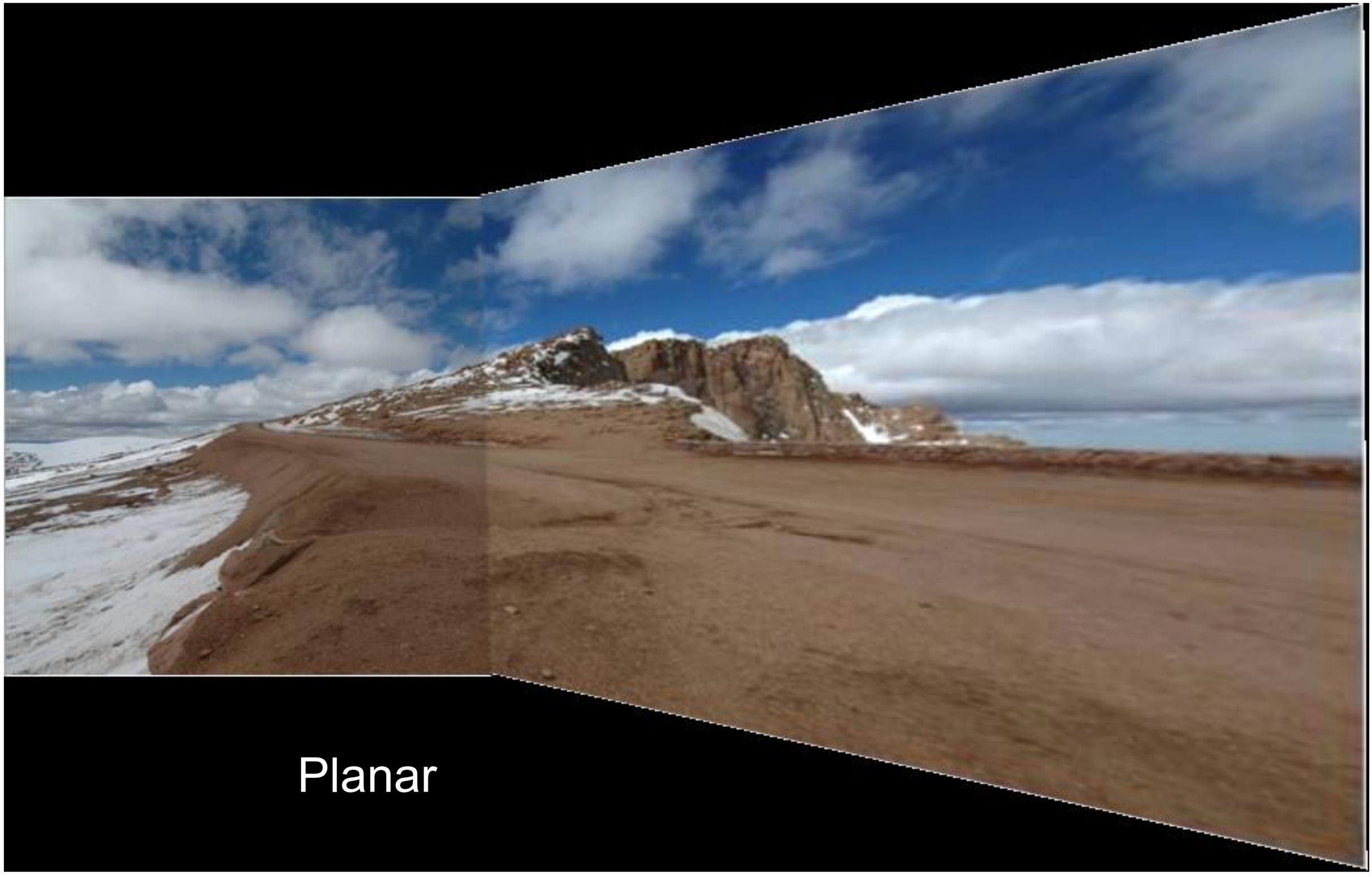


Planar Mapping



1) For red image: pixels are already on the planar surface
2) For green image: map to first image plane

Planar Projection

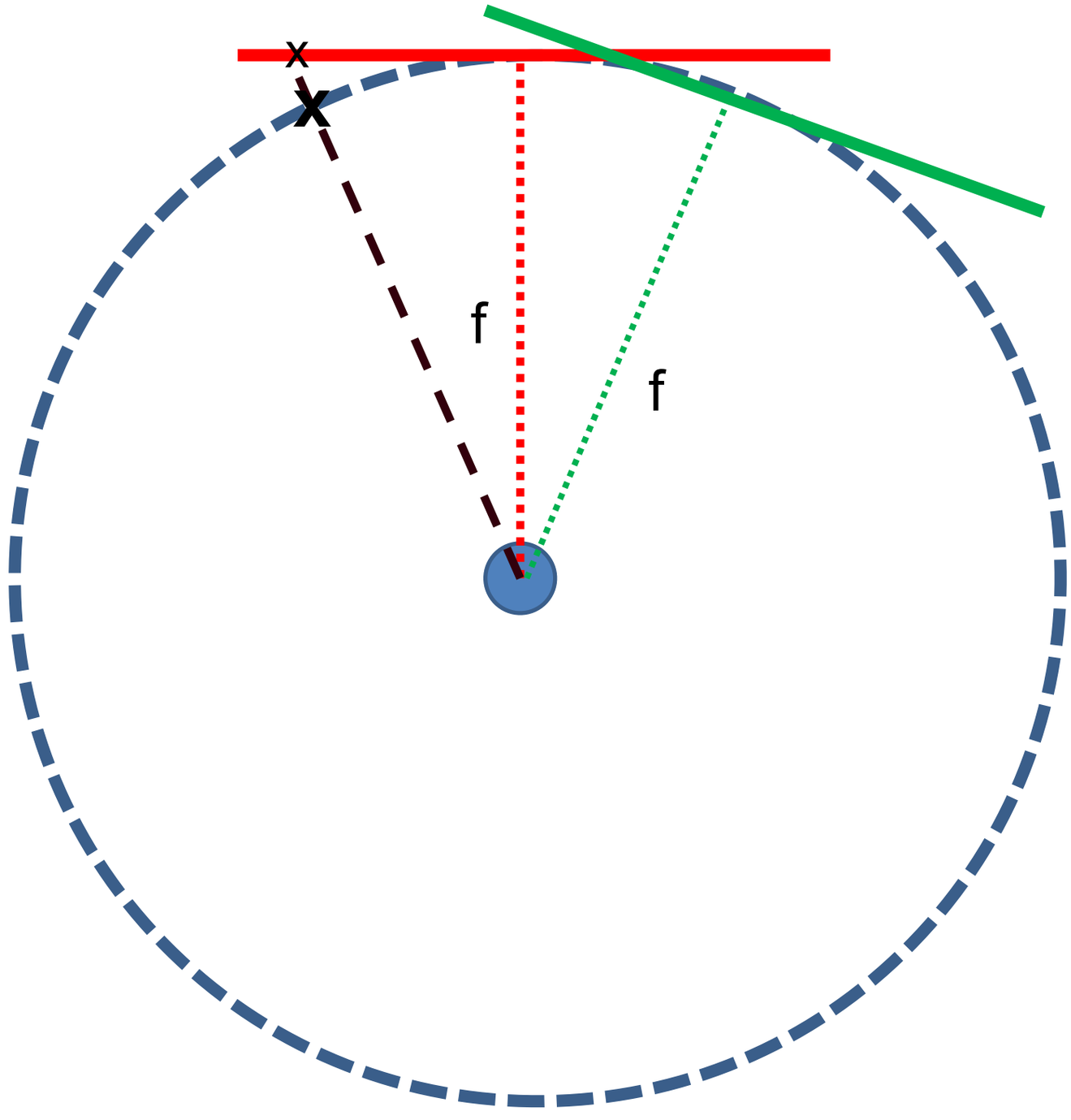


Photos by Russ Hewett

Planar Projection

Planar

Cylindrical Mapping



1) For red image: compute h, theta on cylindrical surface from (u, v)
2) For green image: map to first image plane, than map to cylindrical surface

Cylindrical Projection

Cylindrical

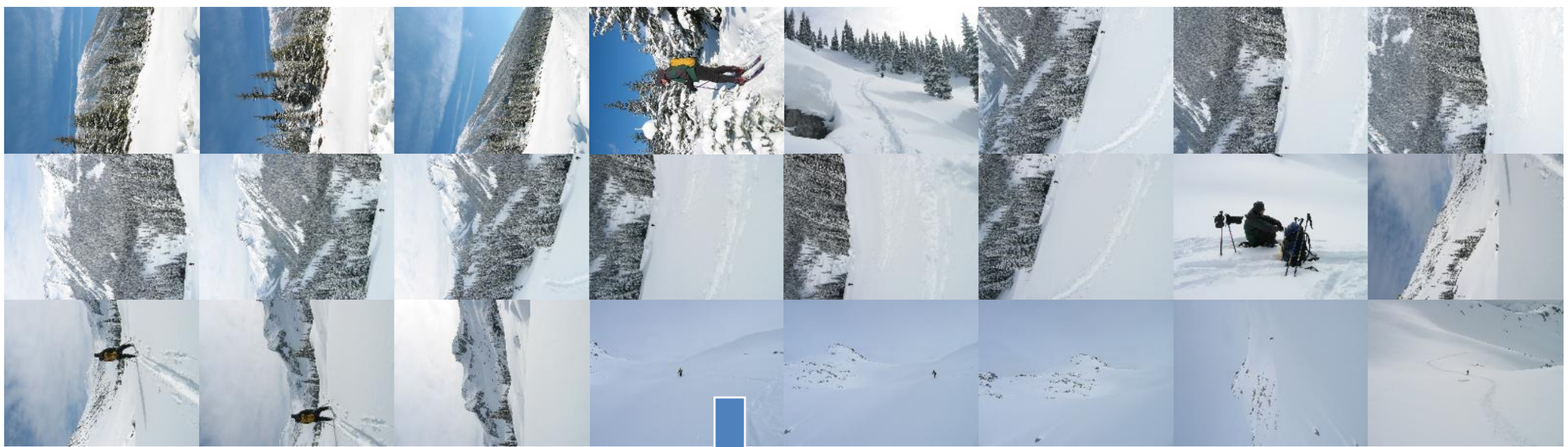
Cylindrical Projection

Cylindrical


Planar


Cylindrical

Recognizing Panoramas



Some of following material from Brown and Lowe 2003 talk

Brown and Lowe 2003, 2007

Recognizing Panoramas

Input: N images

1. Extract SIFT points, descriptors from all images
2. Find K-nearest neighbors for each point (K=4)
3. For each image
   a) Select m candidate matching images by counting matched keypoints (m=6)
   b) Solve homography $\mathbf{H}_{ij}$ for each matched image

Recognizing Panoramas

Input: N images

1. Extract SIFT points, descriptors from all images
2. Find K-nearest neighbors for each point (K=4)
3. For each image
   a) Select m candidate matching images by counting matched keypoints (m=6)
   b) Solve homography $\mathbf{H}_{ij}$ for each matched image
   c) Decide if match is valid ($n_i > 8 + 0.3\ n_f$)

inliers

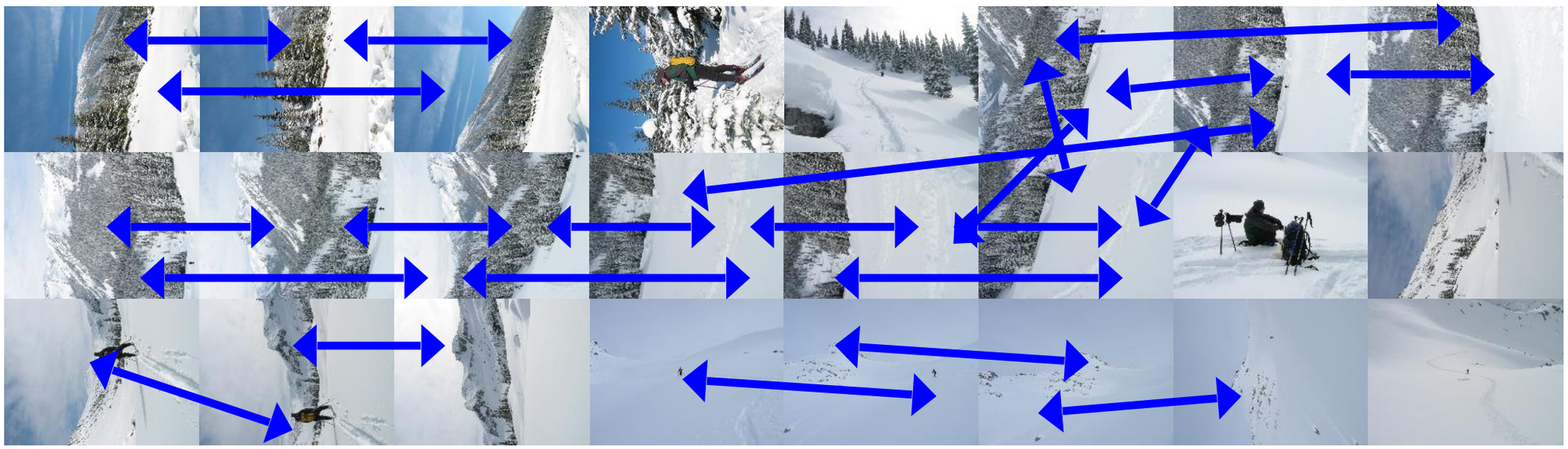
keypoints in overlapping area

Recognizing Panoramas (cont.)

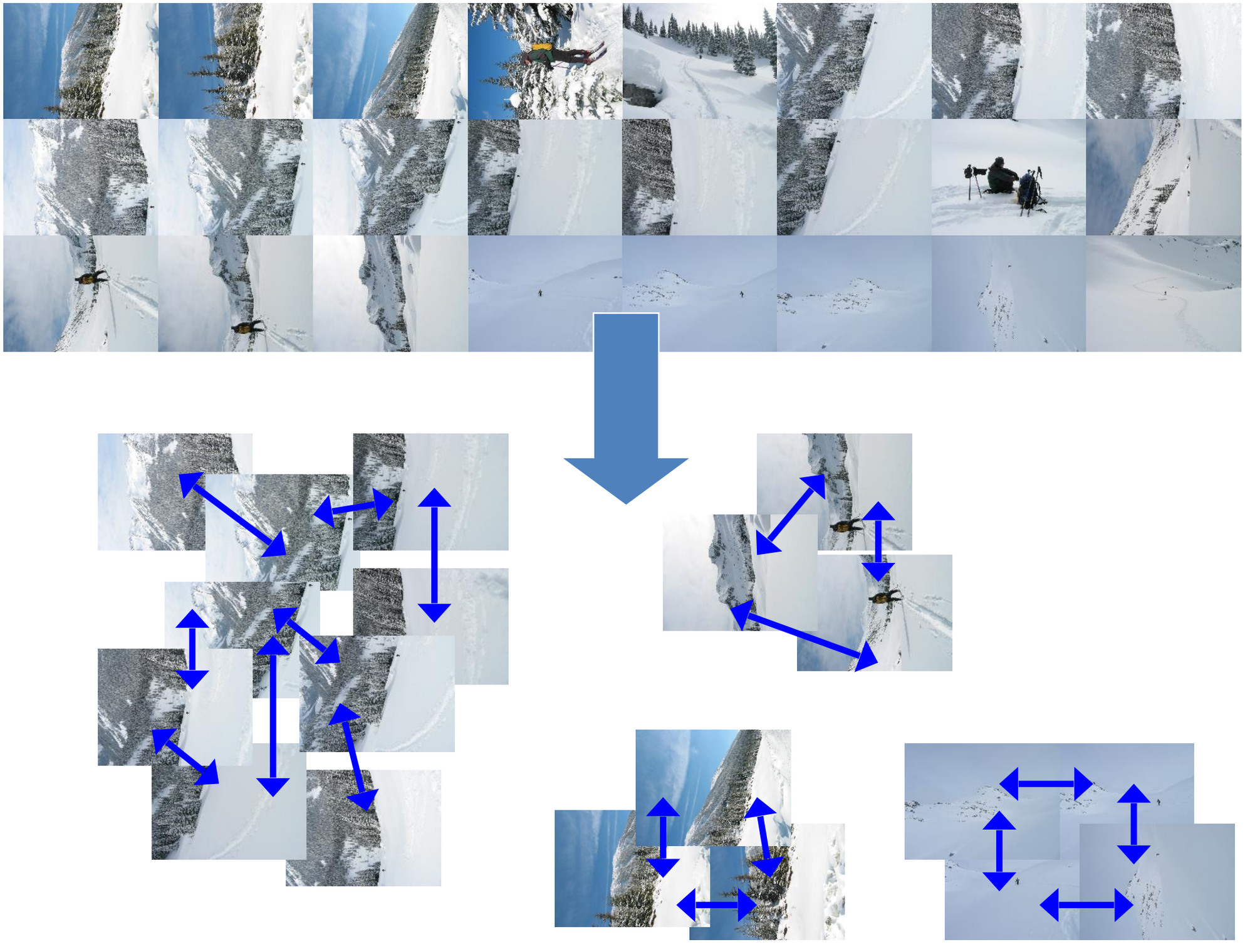
(now we have matched pairs of images)

4. Find connected components

Finding the panoramas



Finding the panoramas



Recognizing Panoramas (cont.)

(now we have matched pairs of images)

4. Find connected components
5. For each connected component
   a) Perform bundle adjustment to solve for rotation $(\theta_1, \theta_2, \theta_3)$ and focal length f of all cameras
   b) Project to a surface (plane, cylinder, or sphere)
   c) Render with multiband blending

Bundle adjustment for stitching

- Non-linear minimization of re-projection error

$$\mathbf{R}_i = e^{[\boldsymbol{\theta}_i]_\times}, \;\; [\boldsymbol{\theta}_i]_\times = \begin{bmatrix} 0 & -\theta_{i3} & \theta_{i2} \\ \theta_{i3} & 0 & -\theta_{i1} \\ -\theta_{i2} & \theta_{i1} & 0 \end{bmatrix}$$

- $\hat{\mathbf{x}}' = \mathbf{H}\mathbf{x}$ where $\mathbf{H} = \mathbf{K}' \mathbf{R}' \mathbf{R}^{-1} \mathbf{K}^{-1}$

$$\mathbf{K}_i = \begin{bmatrix} f_i & 0 & 0 \\ 0 & f_i & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$error = \sum_{1}^{N} \sum_{j}^{M_i} \sum_{k} dist(\mathbf{x}', \hat{\mathbf{x}}')$$

- Solve non-linear least squares (Levenberg-Marquardt algorithm)
  - See paper for details

Bundle Adjustment

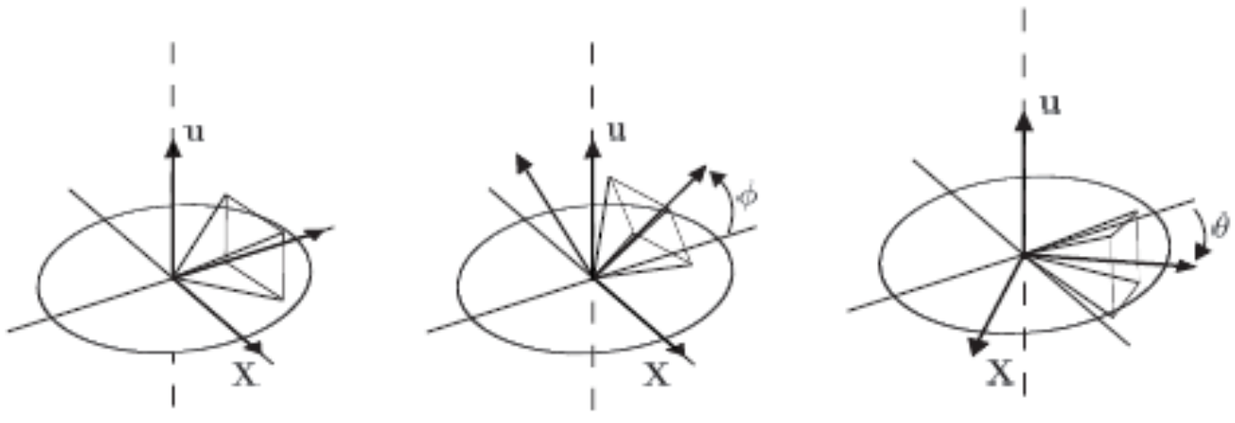
- New images initialised with rotation, focal length of best matching image

Bundle Adjustment

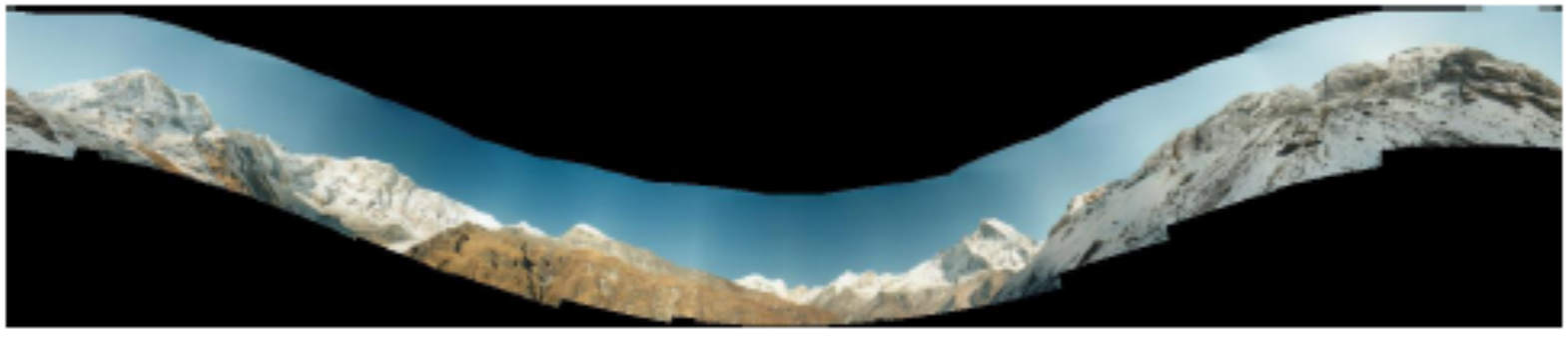
- New images initialised with rotation, focal length of best matching image

Straightening

- Rectify images so that “up” is vertical



(a) (b) (c)



(a) Without automatic straightening

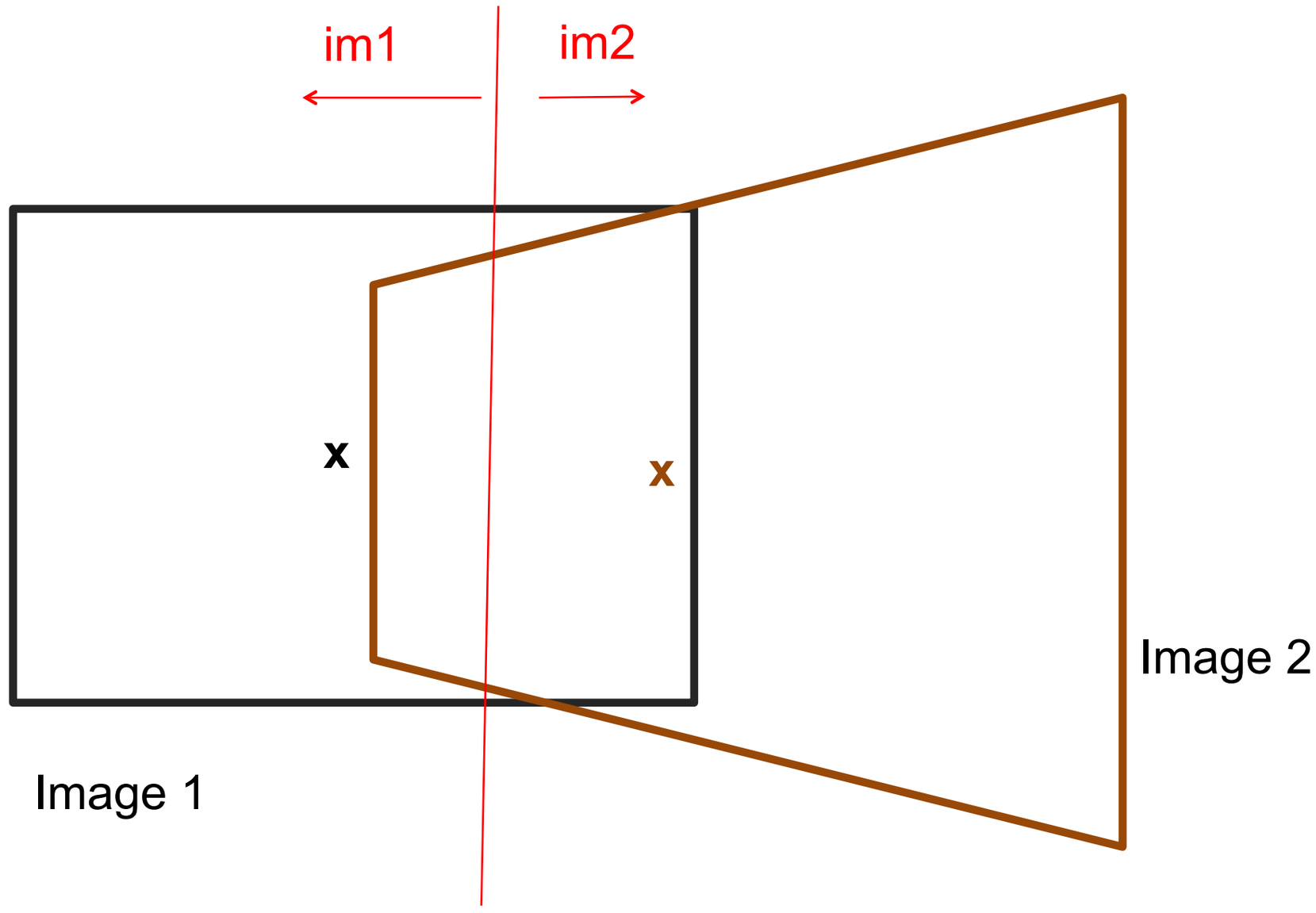
(b) With automatic straightening

Details to make it look good

- Choosing seams
- Blending

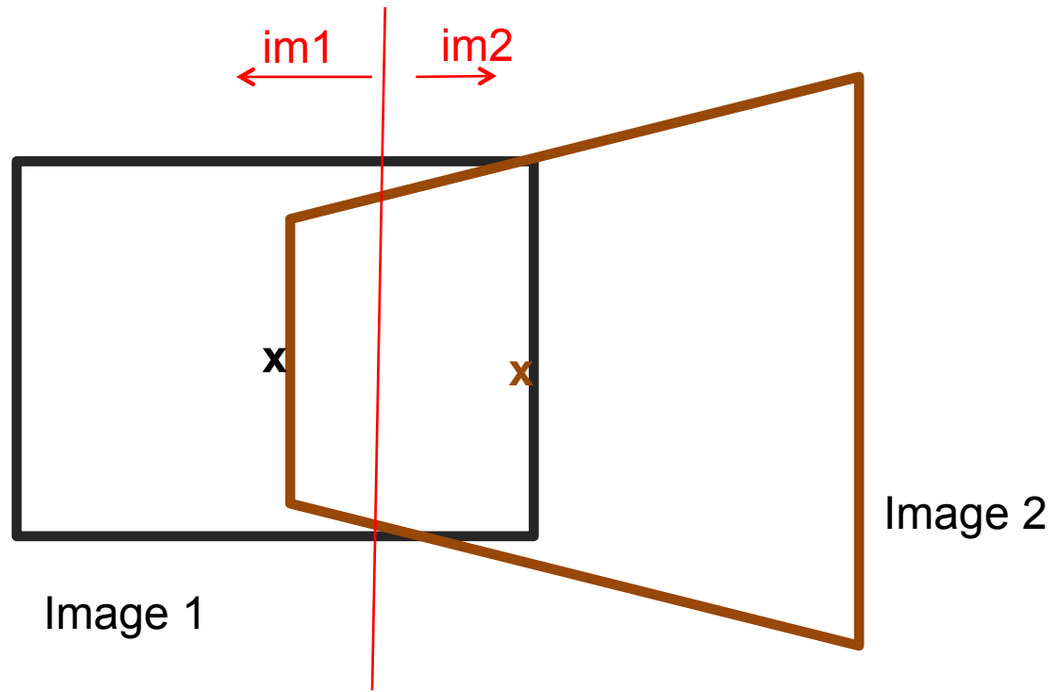
Choosing seams

- Easy method
  - Assign each pixel to image with nearest center



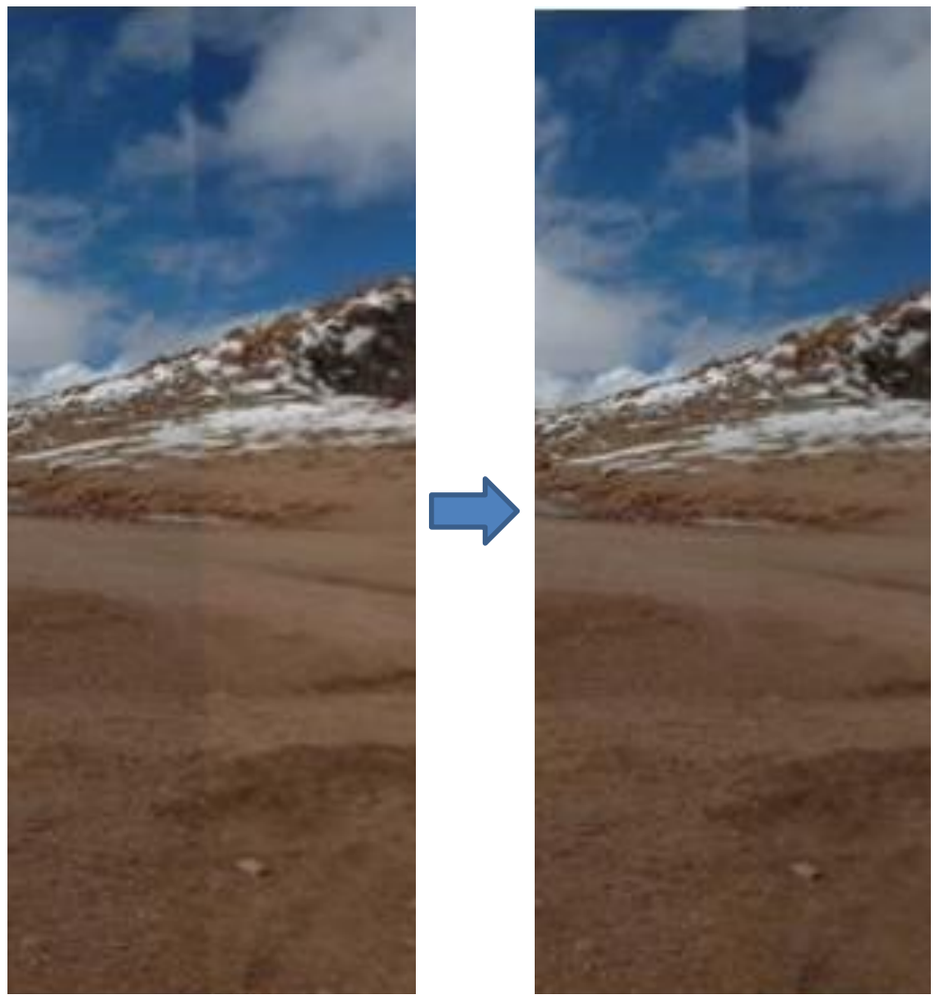
Choosing seams

- Easy method
  - Assign each pixel to image with nearest center
  - Create a mask:
    - `mask(y, x) = 1` **iff pixel should come from im1**
  - Smooth boundaries (called “feathering”):
    - `mask_sm = imfilter(mask, gausfil)`
  - Composite
    - `imblend = im1_c.*mask + im2_c.*(1-mask)`



Gain compensation

- Simple gain adjustment
  - Compute average RGB intensity of each image in overlapping region
  - Normalize intensities by ratio of averages



Multi-band Blending

- Burt & Adelson 1983
  - Blend frequency bands over range $\propto \lambda$

Multiband blending

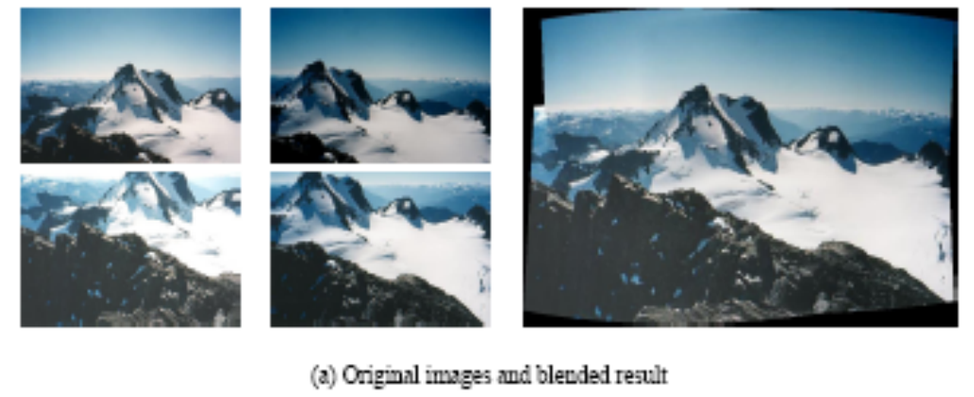
At low frequencies, blend slowly
At high frequencies, blend quickly

1. Compute Laplacian pyramid of images and mask

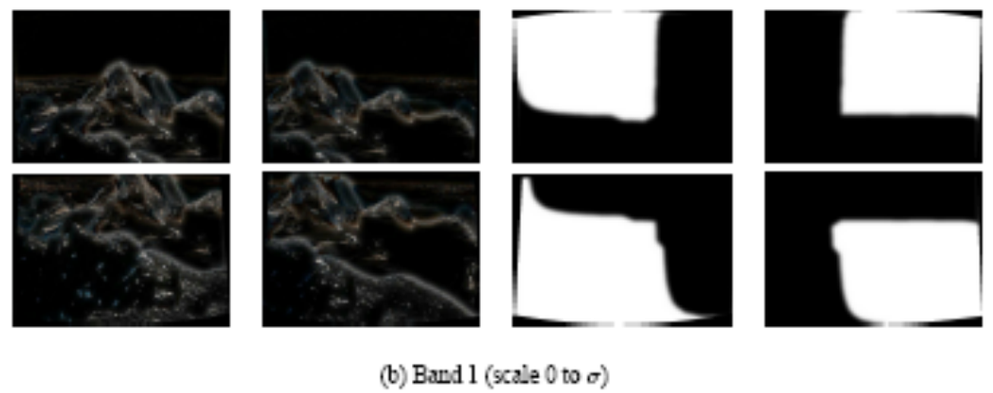
2. Create blended image at each level of pyramid

3. Reconstruct complete image

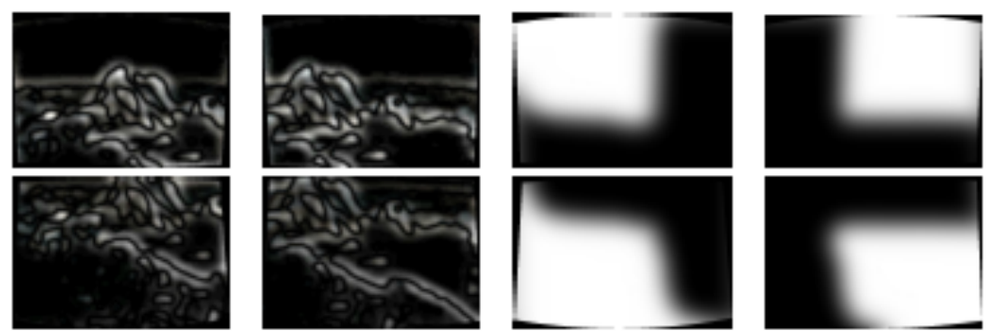
Laplacian pyramids



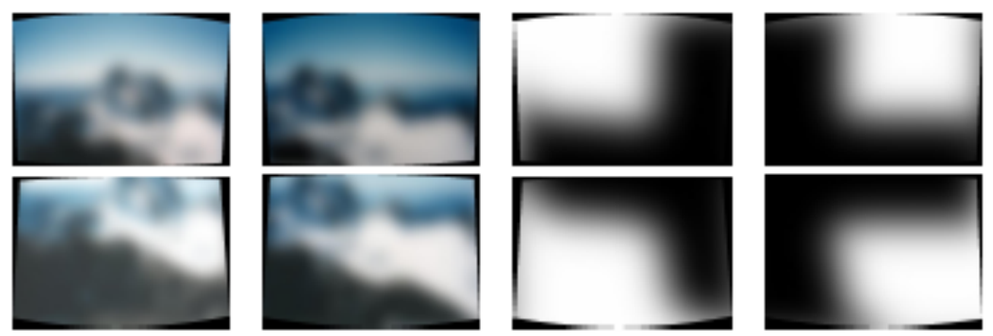
(a) Original images and blended result



(b) Band 1 (scale 0 to σ)

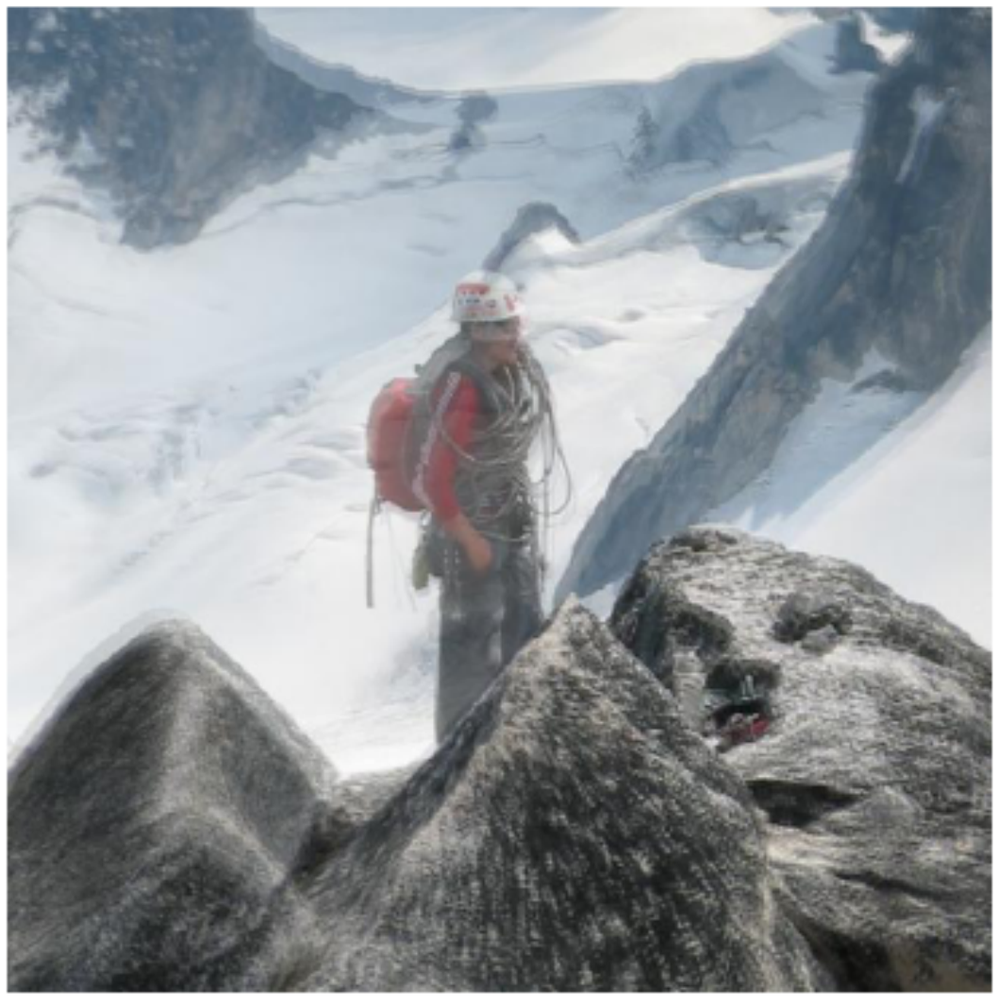


(c) Band 2 (scale σ to 2σ)



(d) Band 3 (scale lower than 2σ)

Blending comparison (IJCV 2007)

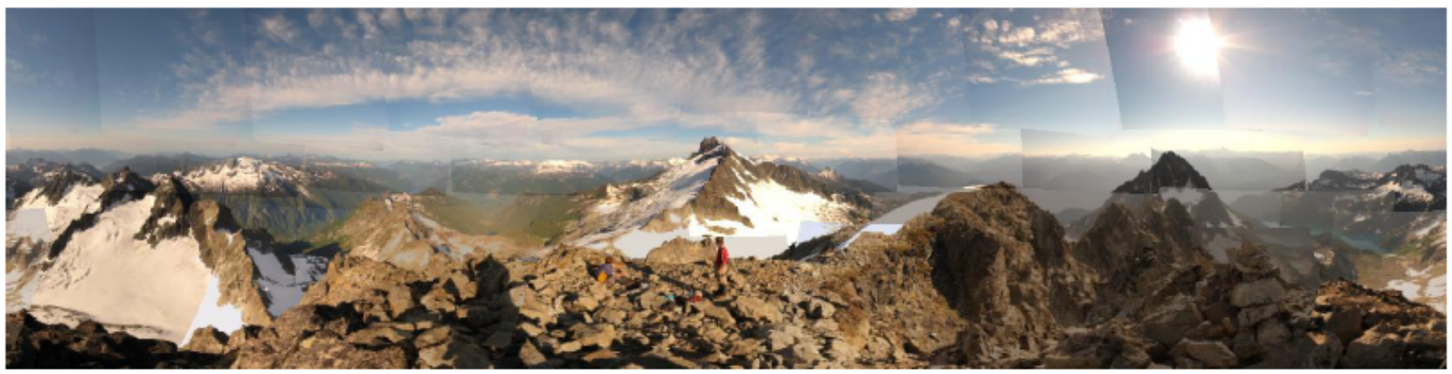


(a) Linear blending

(b) Multi-band blending

Blending Comparison

(b) Without gain compensation



(c) With gain compensation

(d) With gain compensation and multi-band blending

Further reading

- DLT algorithm: HZ p. 91 (alg 4.2), p. 585
- Normalization: HZ p. 107-109 (alg 4.2)
- RANSAC: HZ Sec 4.7, p. 123, alg 4.6

- Rick Szeliski's alignment/stitching tutorial
- Recognising Panoramas: Brown and Lowe, IJCV 2007 (also bundle adjustment)

Stitching summary

- Homography relates rotating cameras

- Recover homography using RANSAC and normalized DLT

- Bundle adjustment (global optimization) minimizes reprojection error for set of related images

- Details to make it look nice (blending).

Next class

- Stereo and epipolar geometry