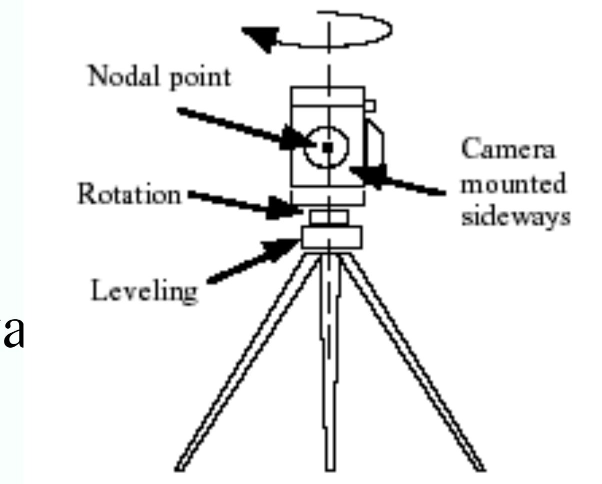
Image Based Rendering

D.A. Forsyth, with slides from John Hart

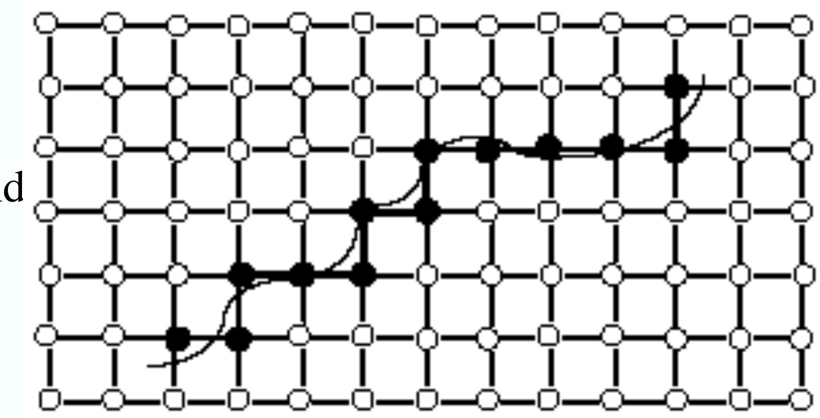
Topics

- Mosaics
  - translating cameras reveal extra information, break occlusion
- Optical flow
  - for very small movements of the camera
- Explicit image based rendering
  - multiple calibrated cameras yield a system of rays that models objects
- Camera calibration
  - postrender things into pictures
- Stereopsis
  - two cameras reveal a lot of geometry
- Structure from motion
  - more cameras yield even more geometry

Implicit example: Quicktime VR

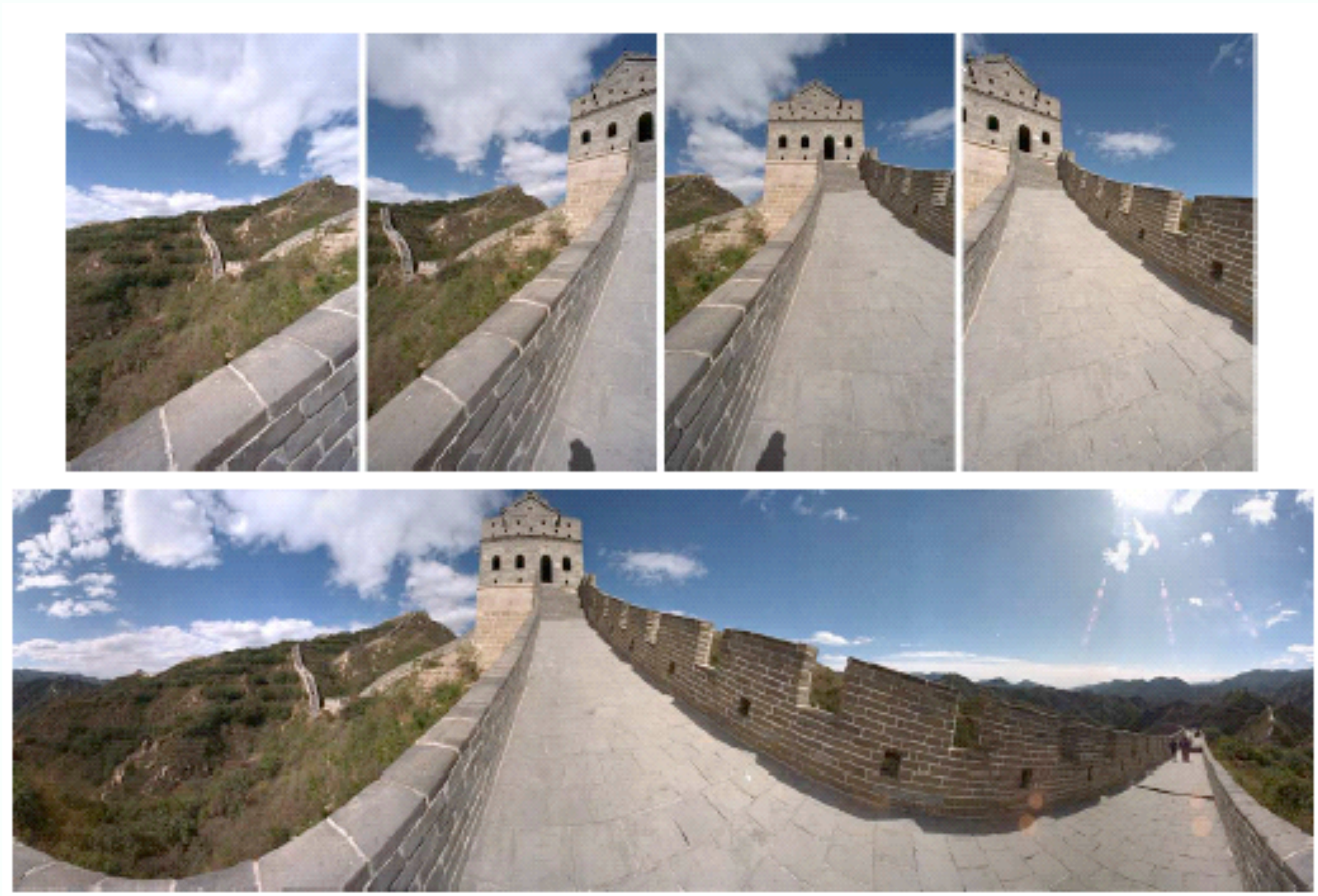


- Construct a mosaic that can provide va
  at various points
- Issues:
  - recovering the mosaics
    - specialised hardware
    - correlation based mosaicing
  - structuring the representation for fast rend



Figures from "QuickTime VR – An Image-Based Approach to Virtual Environment Navigation", Shenchang Eric Chen, SIGGRAPH 95

Figures from “QuickTime VR – An Image-Based Approach to Virtual Environment Navigation”, Shenchang Eric Chen, SIGGRAPH 95



Figures from “QuickTime VR – An Image-Based Approach
to

Matching points is important

M. Brown and D. Lowe, “Recognising Panoramas”, ICCV 2003

Matching points

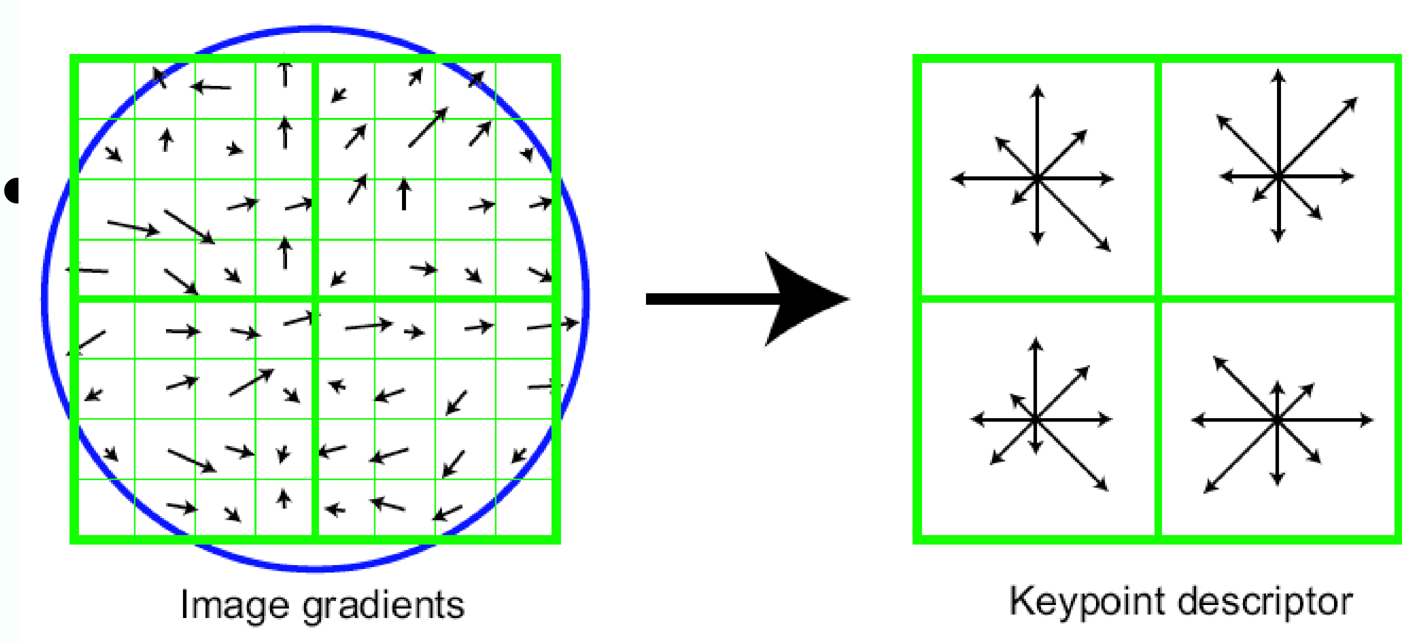
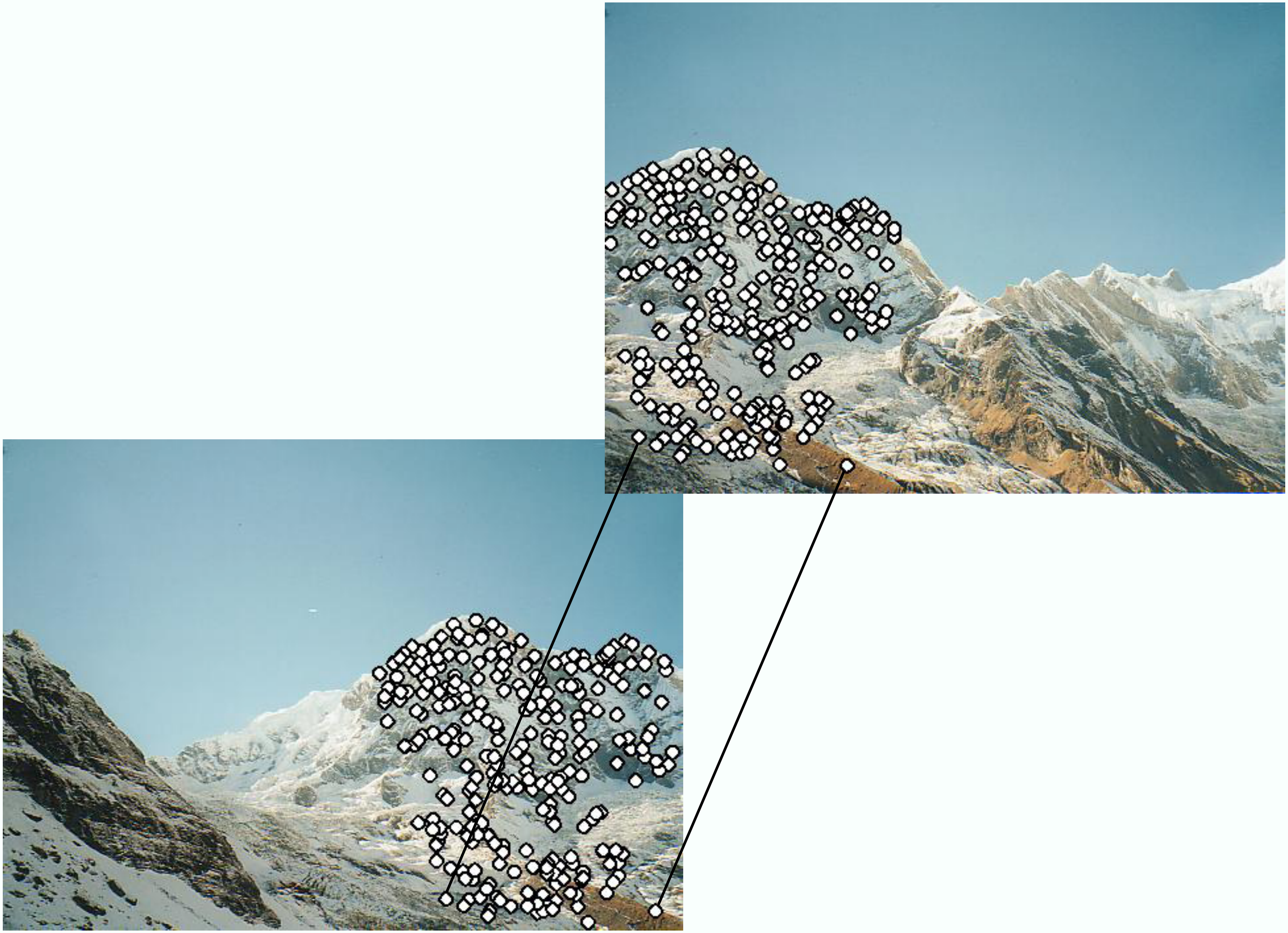


Fig 7 from:
Distinctive image features from scale-invariant keypoints
David G. Lowe, International Journal of Computer Vision, 60, 2 (2004), pp. 91-110.



M. Brown and D. Lowe, “Recognising Panoramas”, ICCV 2003

Translation isn't enough to align the images - we need to use a homography

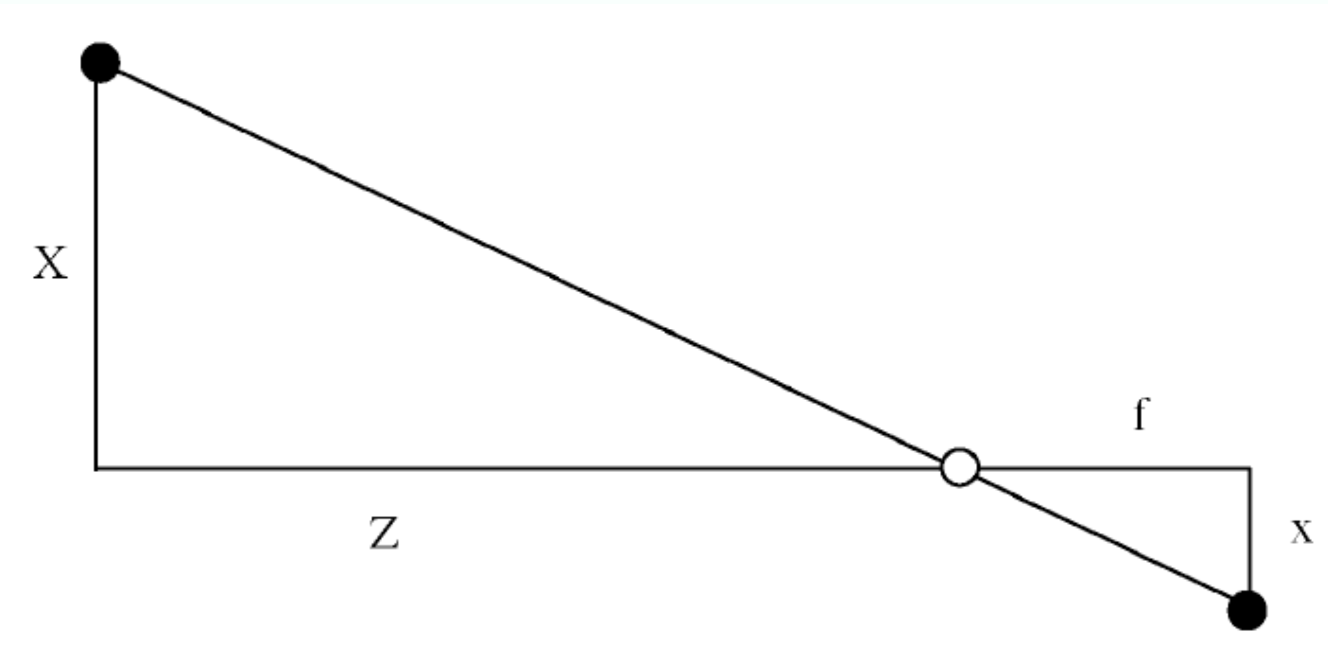
M. Brown and D. Lowe, “Recognising Panoramas”, ICCV 2003

Homographies

- Assume camera rotates about focal point
    - what happens to the image?
        - write camera as matrix, assume infinite image plane at z=-f

Projection in Coordinates

- From the drawing, we have $X/Z = -x/f$
- Generally



A perspective camera as a matrix

- Turn previous expression into HC's
  - HC's for 3D point are (X,Y,Z,T)
  - HC's for point in image are (U,V,W)

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix}$$

$$\mathcal{C}$$

A general perspective camera - I

- Can place a perspective camera at the origin, then rotate and translate coordinate system
- In homogeneous coordinates, rotation, translation are:

$$\mathcal{E} = \begin{pmatrix} \mathcal{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}$$

- So rotated, translated camera is:

$$\mathcal{C}\mathcal{E}$$

A general perspective camera - II

- In the camera plane, there can be a change of coordinates
  - choice of origin
    - there is a “natural” origin --- the camera center
      - where the perpendicular passing through the focal point hits the image plane
  - rotation
  - pixels may not be square
  - scale

- Camera becomes

$$\mathcal{MCE}$$

Intrinsics - typically come with the camera

Extrinsics - change when you move around

What are the transforms?

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} \text{Transform} \\ \text{representing} \\ \text{intrinsic parameters} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{f} & 0 \end{pmatrix} \begin{pmatrix} \text{Transform} \\ \text{representing} \\ \text{extrinsic parameters} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix}$$

$$\begin{pmatrix} s & 0 & c_x \\ 0 & sa & c_y \\ 0 & 0 & s/f \end{pmatrix}$$

cx, cy - location of camera center
s - scale
a - aspect ratio
f - focal length

Homographies

- Camera 1 is

$$\begin{pmatrix} s & 0 & c_x \\ 0 & sa & c_y \\ 0 & 0 & s/f \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

- Camera 2 is

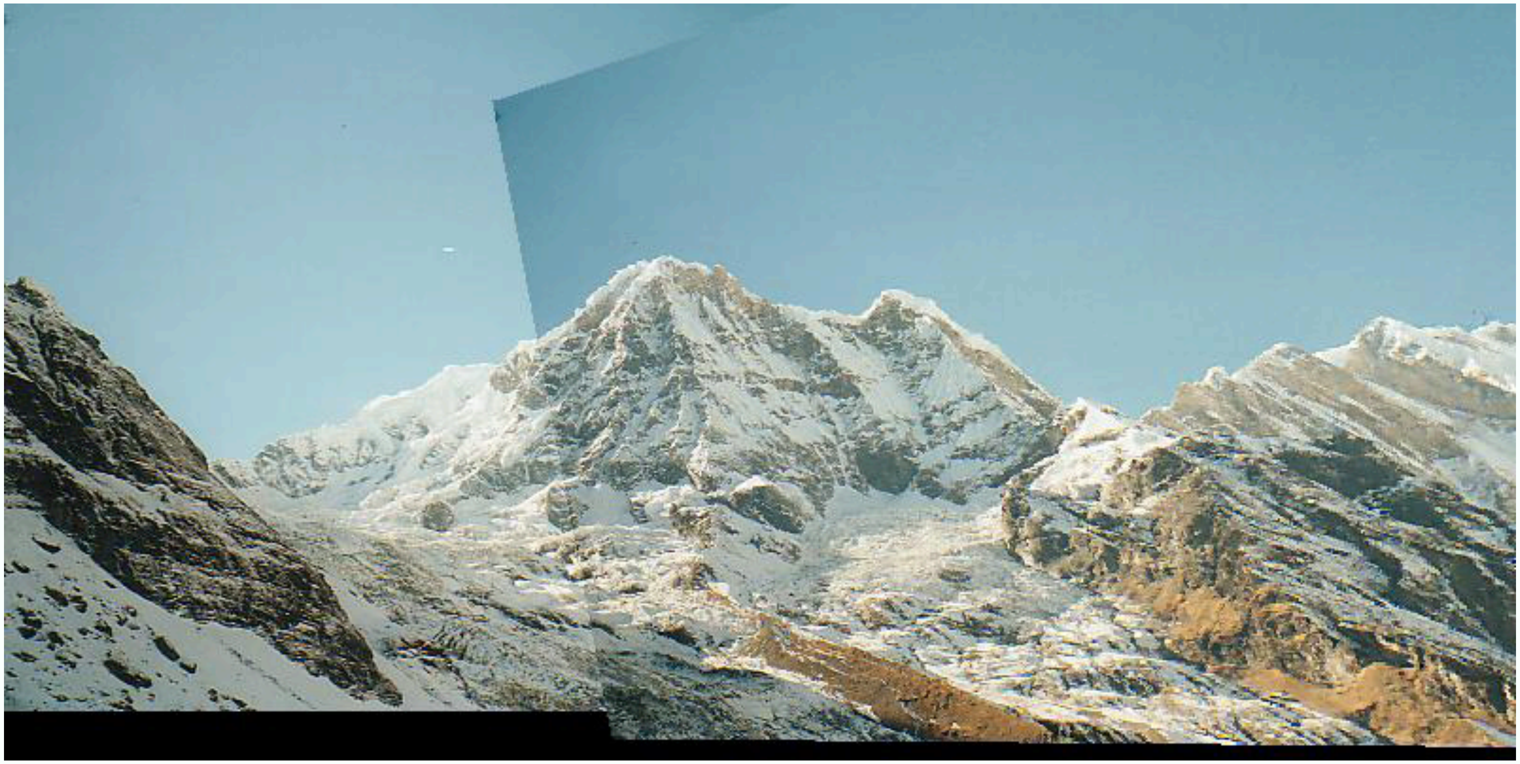
$$\begin{pmatrix} s & 0 & c_x \\ 0 & sa & c_y \\ 0 & 0 & s/f \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathcal{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix}$$

Homographies

- There isn't any translation, so 1 -> 2 is

$$\begin{pmatrix} s & 0 & c_x \\ 0 & sa & c_y \\ 0 & 0 & s/f \end{pmatrix} \mathcal{R} \begin{pmatrix} s & 0 & c_x \\ 0 & sa & c_y \\ 0 & 0 & s/f \end{pmatrix}^{-1}$$

- How do we estimate?
  - linear least squares, followed by nonlinear least squares



M. Brown and D. Lowe, “Recognising Panoramas”, ICCV 2003

M. Brown and D. Lowe, “Recognising Panoramas”, ICCV 2003

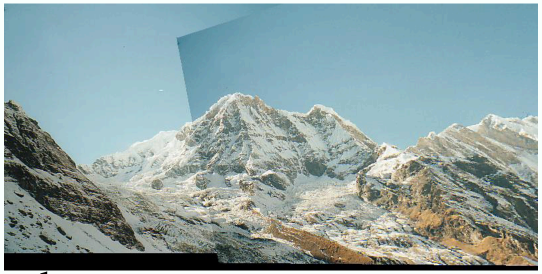
M. Brown and D. Lowe, “Recognising Panoramas”, ICCV 2003

Bundle adjustment

- Errors accumulate
  - so pairwise homographies will not join up to make a cylindrical mosaic
- Minimize all errors for all pairs of corresponding points
  - as a function of all parameters
  - start with pairwise estimates, use newton's method

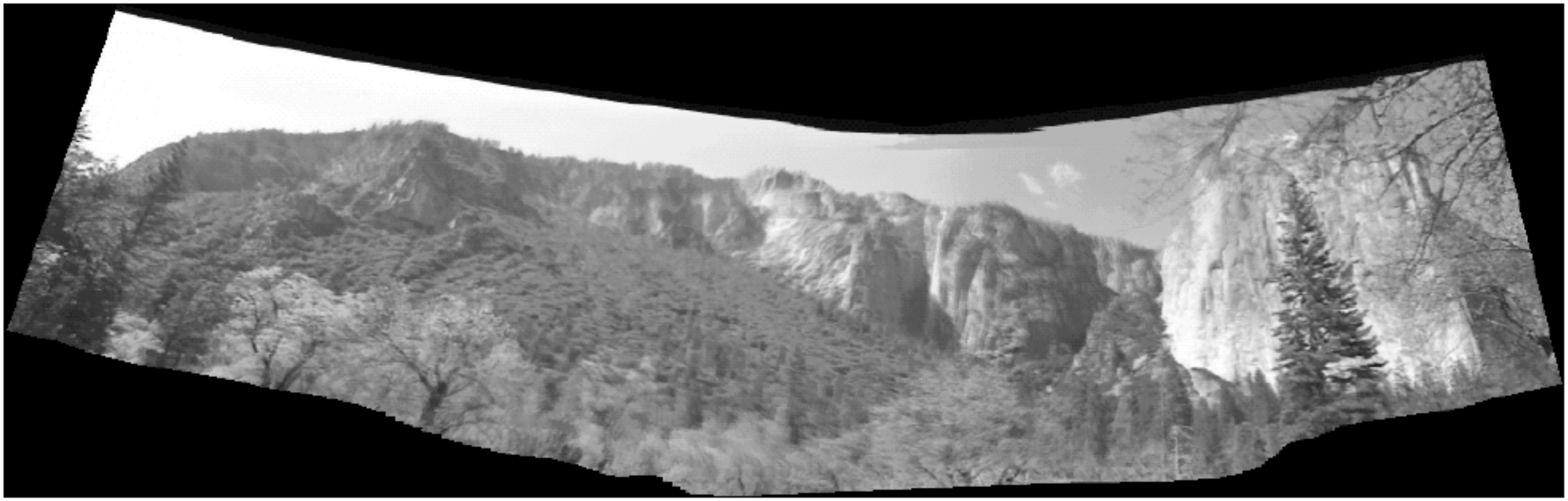
Blending

- Corresponding pixels aren't always same color
  - aperture, sensitivity, etc., etc.
- Blend for consistency
  - pixels "far" from camera center are less reliable
  - Strategy:
    - weight with distance from camera center, then blend
      - fuzzes out small details
  - Strategy
    - separate bands
    - blend low spatial frequencies like this
    - high spatial frequencies from image with most weight



M. Brown and D. Lowe, “Recognising Panoramas”, ICCV 2003


MAY 19 1995



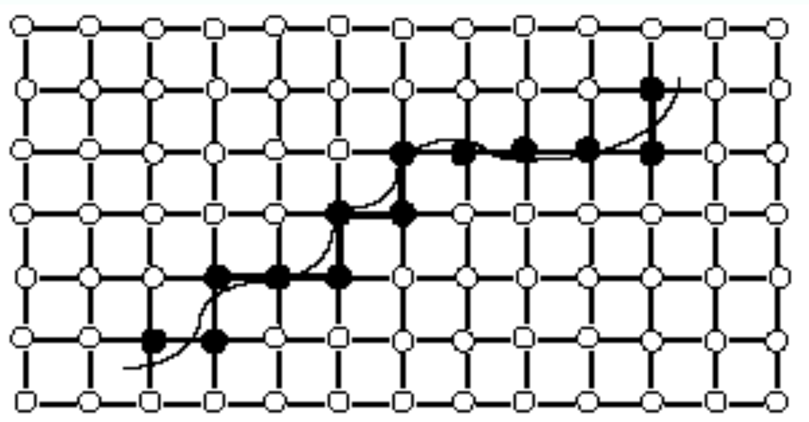
)

)

)

Optical Flow

- Local motion "at a pixel"
  - Arrow joins pixel in this frame to corresponding pixel in next frame
    - hard to estimate accurately from images
      - but easy to predict for small movements of the head, known geom



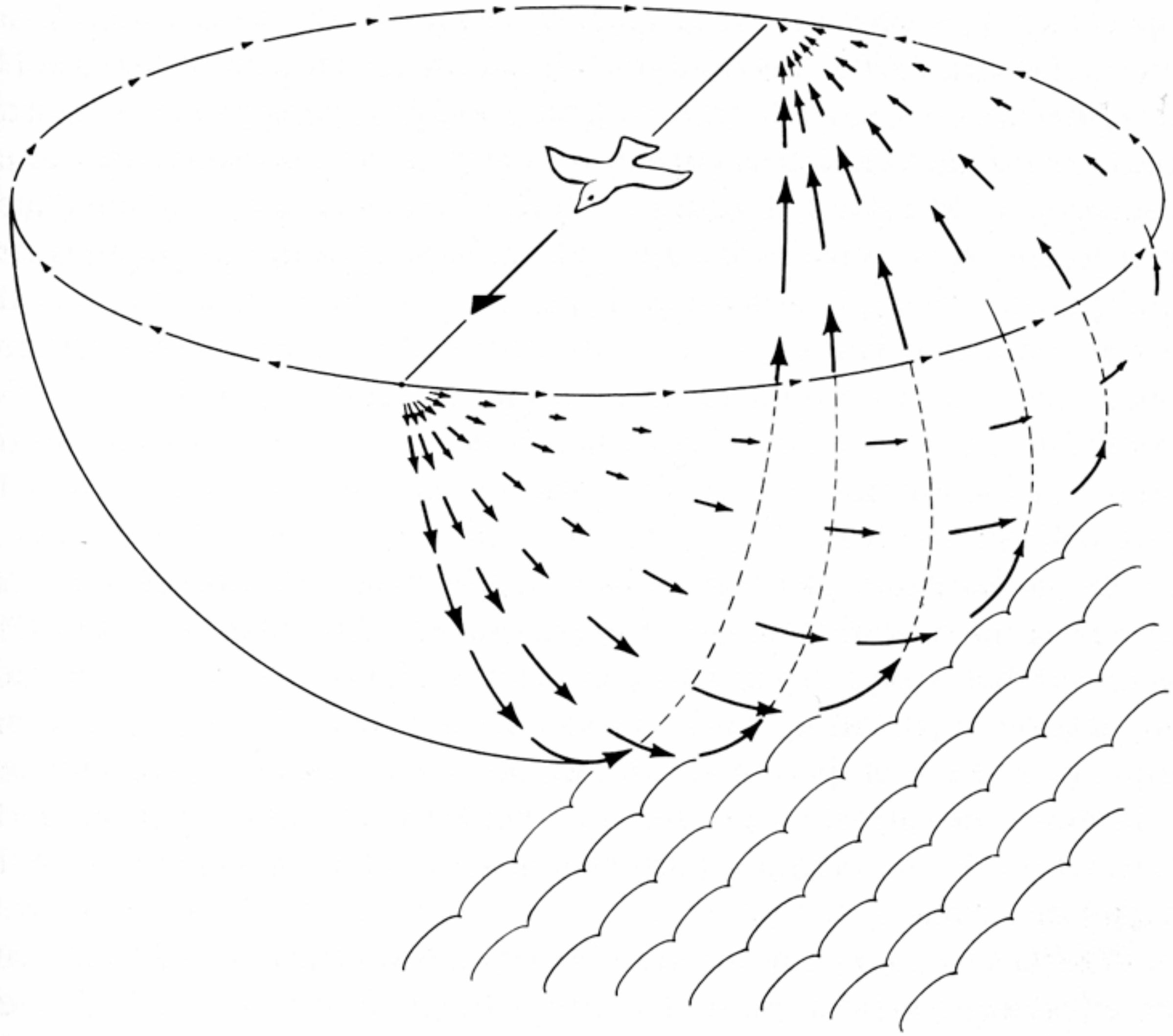
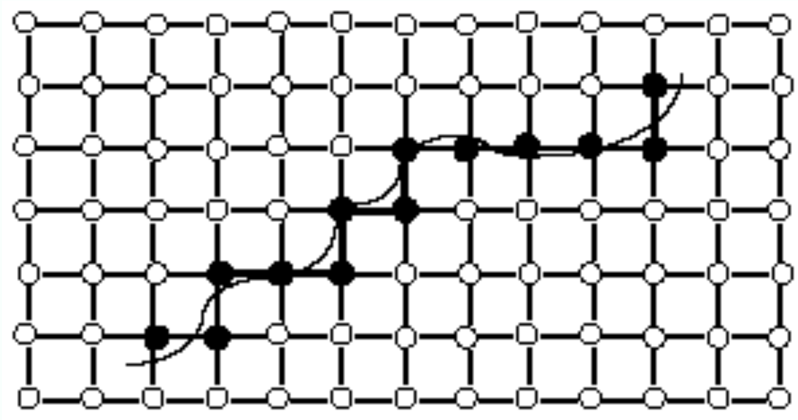


Figure 3–55. Gibson's example of flow induced by motion. The arrows represent angular velocities, which are zero directly ahead and behind. (Reprinted from J. J. Gibson, *The Senses Considered as Perceptual Systems,* Houghton Mifflin, Boston, 1966, fig. 9.3. Copyright © 1966 Houghton Mifflin Company. Used by permission.)

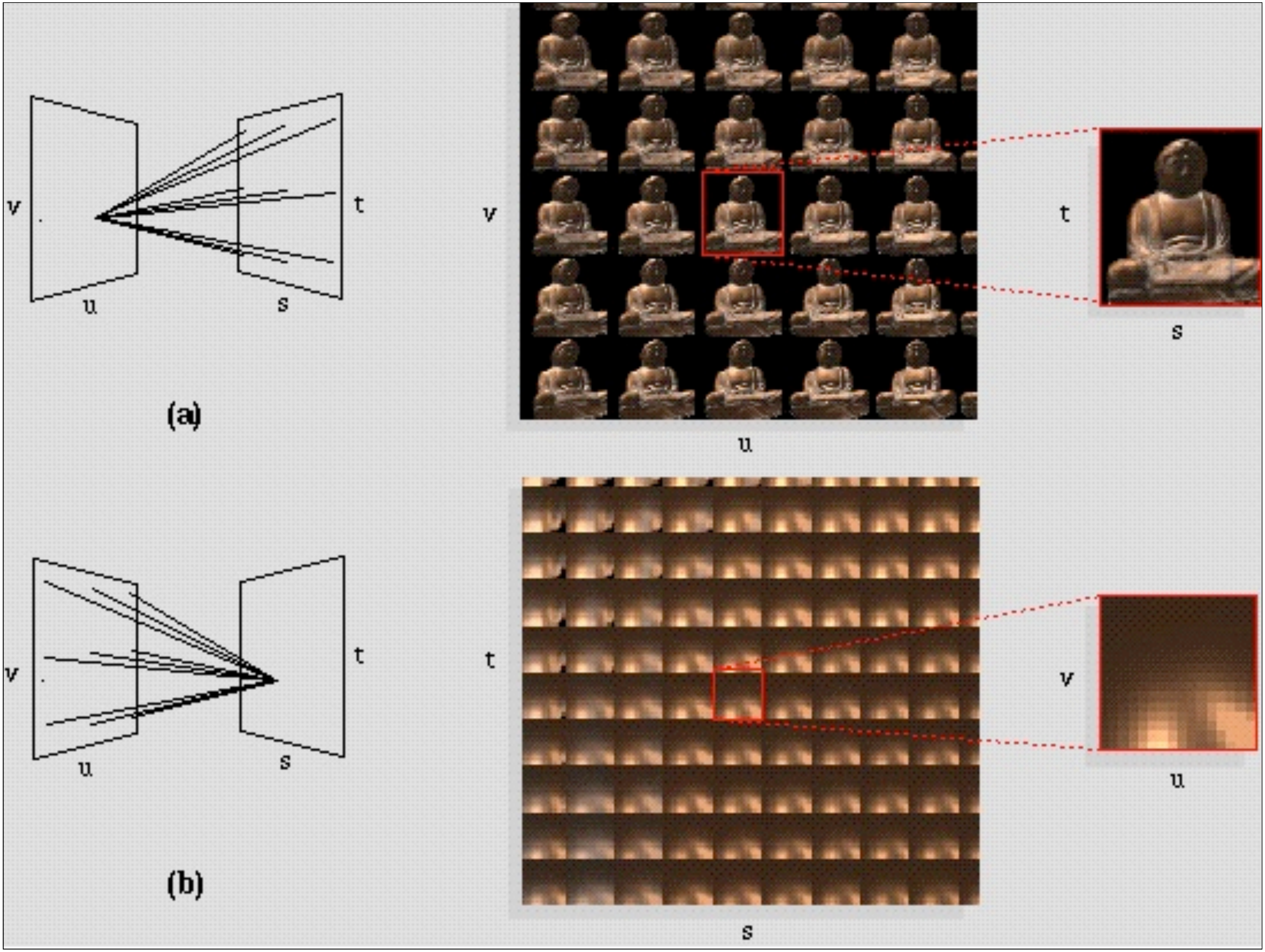
Optical flow



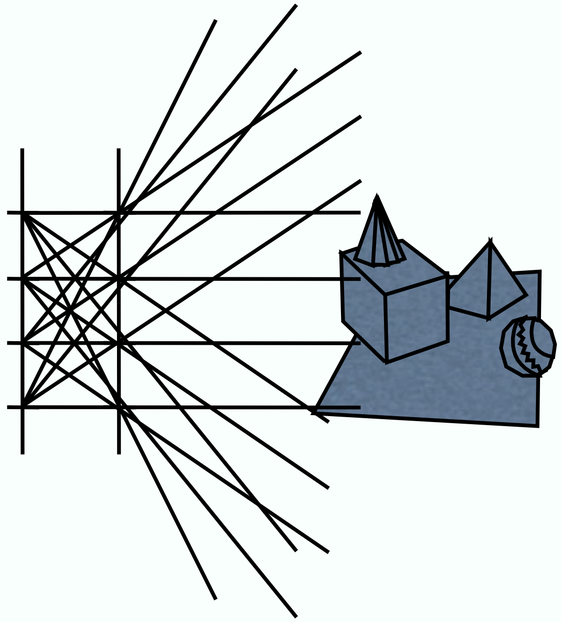
- Compute flows produced by moving
  - with vision methods, using geometry constraints we haven’t done yet
  - interpolate along flow to produce intermediate images

Explicit image based rendering

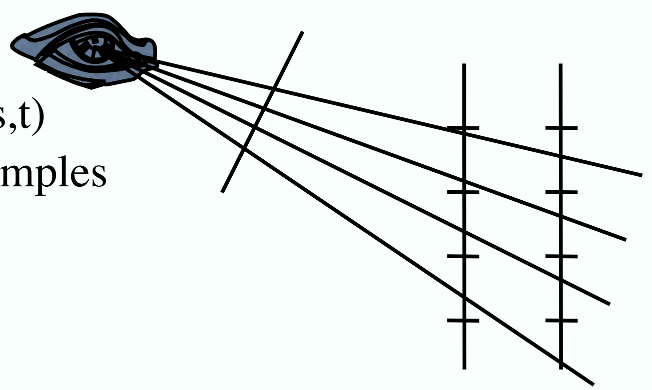
- Put object “in a box”
- Evaluate every light ray through the box
  - four dimensional family
  - by taking lots of photographs
- Render
  - query this structure
  - using any ray tracing alg we know


v
u
s
t
(a)
v
u
t
s
t
s
v
u
v
u
s
(b)
t
s
v
u

Rendering and light fields



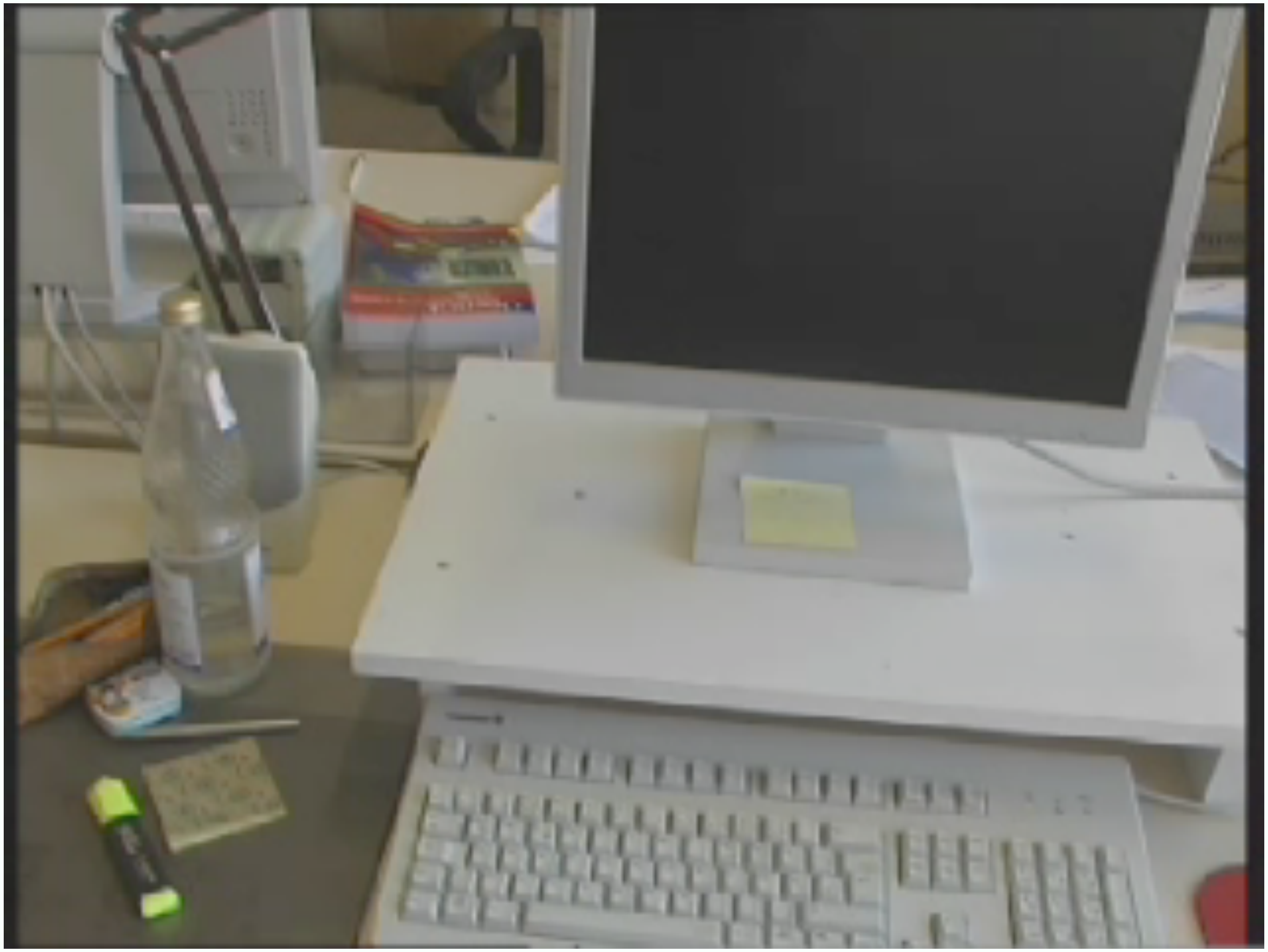
- Rendering into a light field
  - Cast rays between all pairs of points in panes
  - Store resulting radiance at (u,v,s,t)
- Rendering from a light field
  - Cast rays through pixels into light field
  - Compute two ray-plane intersections to find (u,v,s,t)
  - Interpolate u,v and s,t to find radiance between samples
  - Plot radiance in pixel

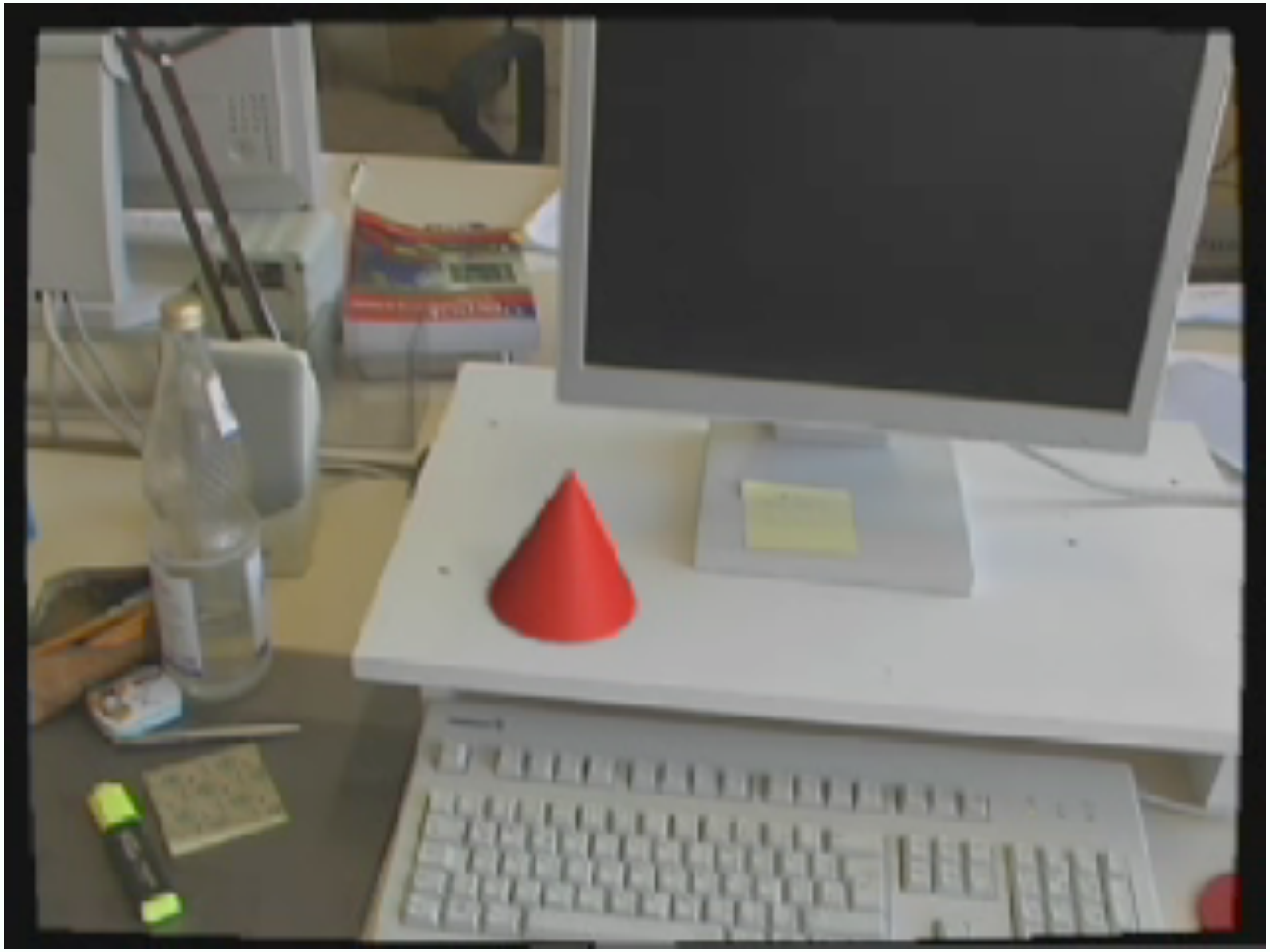


[redacted]

Postrendering into images, video

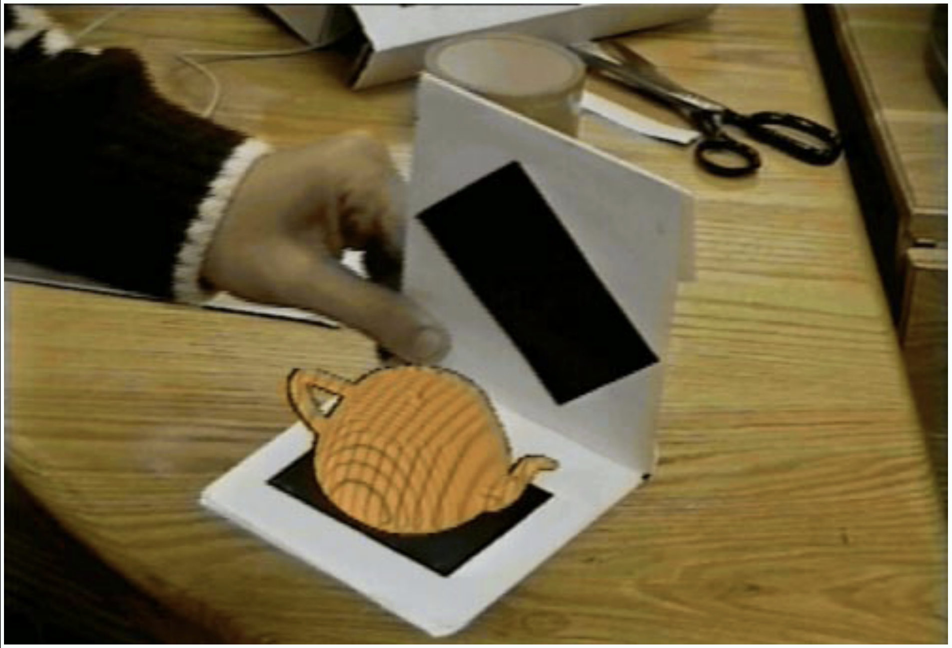
- Options
  - Insert a calibration object, calibrate camera, use this info to render
    - problem: calibration object in picture
  - (video) reconstruct world points, camera, render using camera
    - we'll discuss this shortly

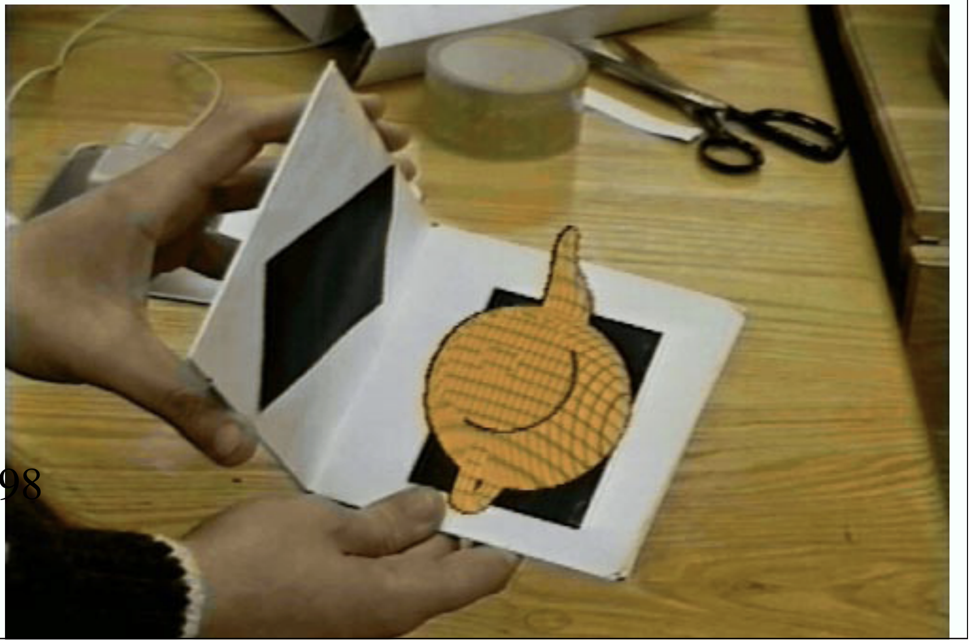




Camera calibration

- Two strategies:
  - Perspective cameras
    - calibration object has known points in 3D
    - find projections
    - compute camera using least squares
  - Scaled Orthography
    - projection is linear (no division, no H.C.'s)
    - world points as unique linear combination of calibration points
    - image projection is same linear combination of projected calibration points





Calibration-Free Augmented Reality
Kiriakos N. Kutulakos and James R. Vallino, 1998