

Announcements

- Panorama signups
- Panorama project issues?
- Nalwa handout

Projective geometry



[Ames Room](#)

Readings

- Mundy, J.L. and Zisserman, A., Geometric Invariance in Computer Vision, Chapter 23: Appendix: Projective Geometry for Machine Vision, MIT Press, Cambridge, MA, 1992, pp. 463-534 (for this week, read 23.1 - 23.5, 23.10)
- available online: <http://www.cs.cmu.edu/~ph/869/papers/zisser-mundy.pdf>

Projective geometry—what's it good for?

Uses of projective geometry

- Drawing
- Measurements
- Mathematics for projection
- Undistorting images
- Focus of expansion
- Camera pose estimation, match move
- Object recognition

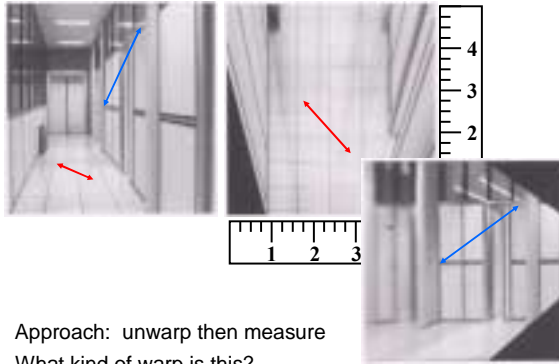
Applications of projective geometry



Vermeer's Music Lesson

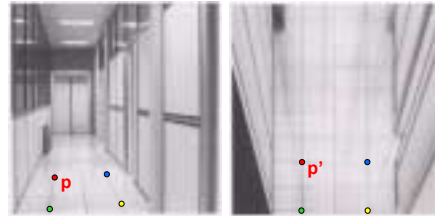


Measurements on planes



Approach: unwarped then measure
What kind of warp is this?

Image rectification



To unwarped (rectify) an image

- solve for homography \mathbf{H} given \mathbf{p} and \mathbf{p}'
- solve equations of the form: $\mathbf{w}\mathbf{p}' = \mathbf{H}\mathbf{p}$
 - linear in unknowns: w and coefficients of \mathbf{H}
 - \mathbf{H} is defined up to an arbitrary scale factor
 - how many points are necessary to solve for \mathbf{H} ?

work out on board

Solving for homographies

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

$$x'_i = \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

$$y'_i = \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

$$x'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{00}x_i + h_{01}y_i + h_{02}$$

$$y'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{10}x_i + h_{11}y_i + h_{12}$$

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

Solving for homographies

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A} \quad \mathbf{h} \quad \mathbf{0}$$

$2n \times 9 \quad 9 \quad 2n$

Linear least squares

- Since \mathbf{h} is only defined up to scale, solve for unit vector $\hat{\mathbf{h}}$
- Minimize $\|\mathbf{A}\hat{\mathbf{h}}\|^2$

$$\|\mathbf{A}\hat{\mathbf{h}}\|^2 = (\mathbf{A}\hat{\mathbf{h}})^T \mathbf{A}\hat{\mathbf{h}} = \hat{\mathbf{h}}^T \mathbf{A}^T \mathbf{A} \hat{\mathbf{h}}$$
- Solution: $\hat{\mathbf{h}}$ = eigenvector of $\mathbf{A}^T \mathbf{A}$ with smallest eigenvalue
- Works with 4 or more points

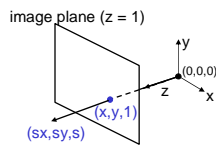
The projective plane

Why do we need homogeneous coordinates?

- represent points at infinity, homographies, perspective projection, multi-view relationships

What is the geometric intuition?

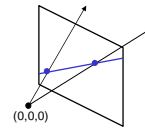
- a point in the image is a *ray* in projective space



- Each point (x,y) on the plane is represented by a ray (sx, sy, s)
 - all points on the ray are equivalent: $(x, y, 1) \equiv (sx, sy, s)$

Projective lines

What is a line in projective space?



- A line is a *plane* of rays through origin
 - all rays (x,y,z) satisfying: $ax + by + cz = 0$

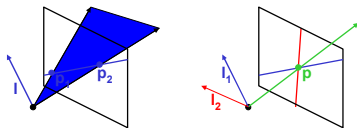
$$\text{in vector notation: } 0 = \begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$\mathbf{l} \quad \mathbf{p}$

- A line is also represented as a homogeneous 3-vector \mathbf{l}

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 rays \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} is the plane normal

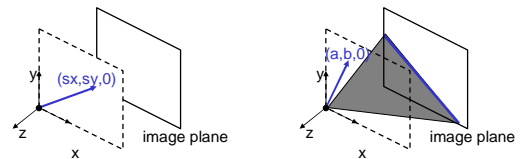
What is the intersection of two lines \mathbf{l}_1 and \mathbf{l}_2 ?

- \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

- every property of points also applies to lines

Ideal points and lines



Ideal point ("point at infinity")

- $\mathbf{p} \equiv (x, y, 0)$ – parallel to image plane
- It has infinite image coordinates

Ideal line

- $\mathbf{l} \equiv (a, b, 0)$ – parallel to image plane
- Corresponds to a line in the image (finite coordinates)

Homographies of points and lines

Computed by 3x3 matrix multiplication

- To transform a point: $\mathbf{p}' = \mathbf{H}\mathbf{p}$
- To transform a line: $\mathbf{l}\mathbf{p}=0 \rightarrow \mathbf{l}'\mathbf{p}'=0$
 - $0 = \mathbf{l}\mathbf{p} = \mathbf{l}\mathbf{H}^{-1}\mathbf{H}\mathbf{p} = \mathbf{l}\mathbf{H}^{-1}\mathbf{p}' \Rightarrow \mathbf{l}' = \mathbf{l}\mathbf{H}^{-1}$
 - lines are transformed by postmultiplication of \mathbf{H}^{-1}

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: $\mathbf{N}\mathbf{P}=0$
- Projective transformations
 - Represented by 4x4 matrices \mathbf{T} : $\mathbf{P}' = \mathbf{T}\mathbf{P}$, $\mathbf{N}' = \mathbf{N}\mathbf{T}^{-1}$

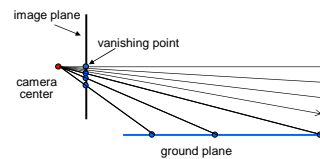
3D to 2D: “perspective” projection

Matrix Projection:
$$\mathbf{p} = \begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{M}\mathbf{P}$$

What is not preserved under perspective projection?

What IS preserved?

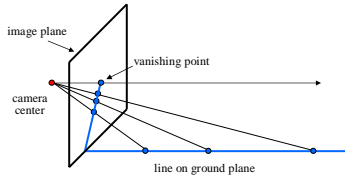
Vanishing points



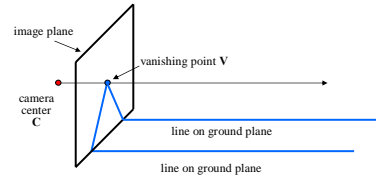
Vanishing point

- projection of a point at infinity

Vanishing points (2D)



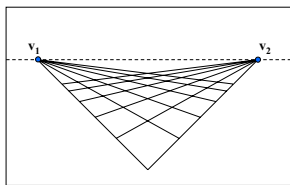
Vanishing points



Properties

- Any two parallel lines have the same vanishing point
- The ray from **C** through **v** point is parallel to the lines
- An image may have more than one vanishing point

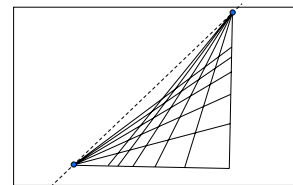
Vanishing lines



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 define different vanishing lines

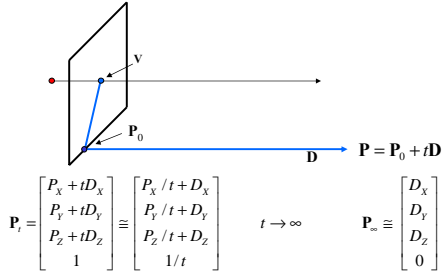
Vanishing lines



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 define different vanishing lines

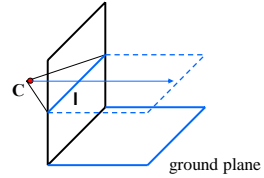
Computing vanishing points



Properties $v = \Pi P_\infty$

- P_∞ is a point at *infinity*, v is its projection
- They depend only on line *direction*
- Parallel lines $P_0 + tD$, $P_1 + tD$ intersect at P_∞

Computing vanishing lines



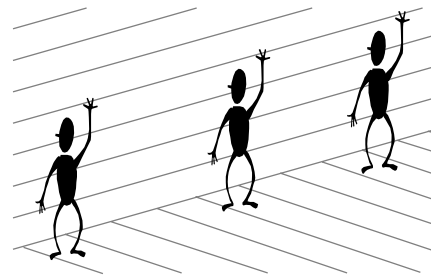
Properties

- 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
- Provides way of comparing height of objects in the scene

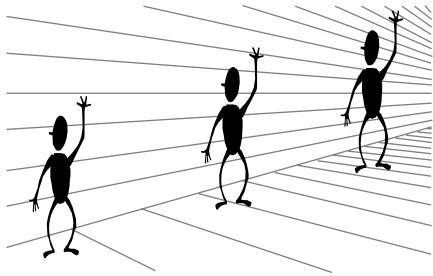
Fun with vanishing points



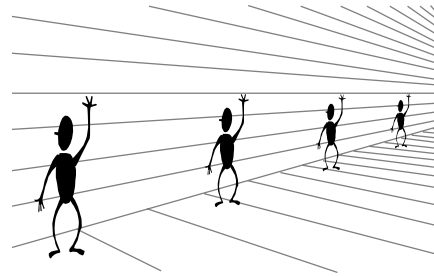
Perspective cues



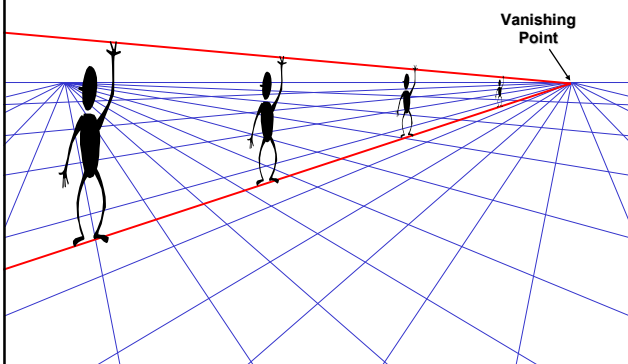
Perspective cues



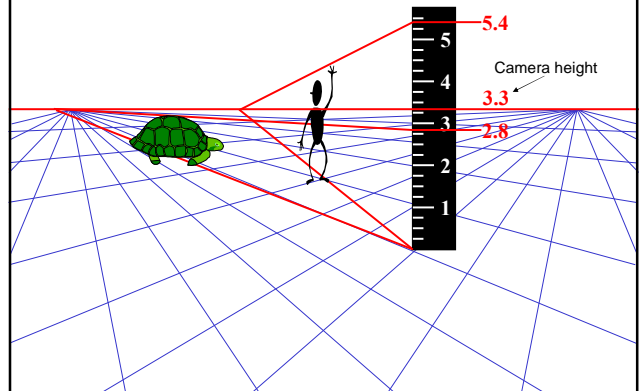
Perspective cues



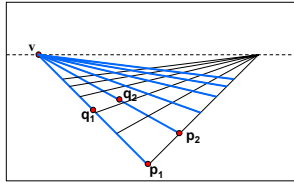
Comparing heights



Measuring height



Computing vanishing points (from lines)



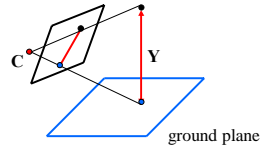
Intersect p_1q_1 with p_2q_2

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

Least squares version

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

Measuring height without a ruler



Compute Y from image measurements

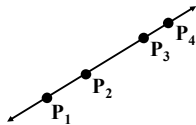
- Need more than vanishing points to do this

The cross ratio

A Projective Invariant

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

The cross-ratio of 4 collinear points



$$\frac{\|P_3 - P_1\| \|P_4 - P_2\|}{\|P_3 - P_2\| \|P_4 - P_1\|}$$

$$P_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

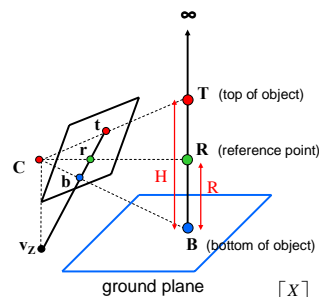
$$\frac{\|P_1 - P_3\| \|P_4 - P_2\|}{\|P_1 - P_2\| \|P_4 - P_3\|}$$

Can permute the point ordering

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

This is the fundamental invariant of projective geometry

Measuring height



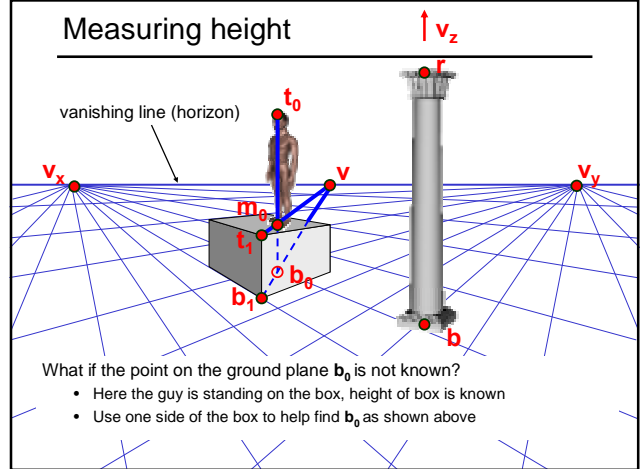
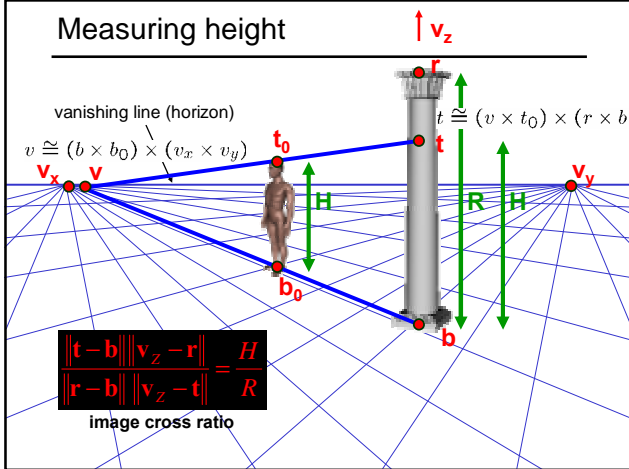
$$\frac{\|T - B\| \|\infty - R\|}{\|R - B\| \|\infty - T\|} = \frac{H}{R}$$

scene cross ratio

$$\frac{\|t - b\| \|v_z - r\|}{\|r - b\| \|v_z - t\|} = \frac{H}{R}$$

image cross ratio

scene points represented as $P = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$ image points as $p = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$



Measurements within reference plane

Solve for homography H relating reference plane to image plane

- H maps reference plane (X,Y) coords to image plane (x,y) coords
- Fully determined from 4 known points on ground plane
 - Option A: physically measure 4 points on ground
 - Option B: find a square, guess the dimensions
 - Option C: Note $H =$ columns 1,2,4 projection matrix
 - » derive on board
- Given (x, y) , can find (X,Y) by H^{-1}

Criminisi et al., ICCV 99

Complete approach

- Load in an image
- Click on lines parallel to X axis
 - repeat for Y, Z axes
- Compute vanishing points
- Specify 3D and 2D positions of 4 points on reference plane
- Compute homography H
- Specify a reference height
- Compute 3D positions of several points
- Create a 3D model from these points
- Extract texture maps
- Output a VRML model

Vanishing points and projection matrix

$$\mathbf{\Pi} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ \pi_1 & \pi_2 & \pi_3 & \pi_4 \end{bmatrix} = [\pi_1 \quad \pi_2 \quad \pi_3 \quad \pi_4]$$

- $\pi_1 = \mathbf{\Pi} [1 \ 0 \ 0 \ 0]^T = \mathbf{v}_x$ (X vanishing point)
- similarly, $\pi_2 = \mathbf{v}_y$, $\pi_3 = \mathbf{v}_z$
- $\pi_4 = \mathbf{\Pi} [0 \ 0 \ 0 \ 1]^T$ = projection of world origin

$$\mathbf{\Pi} = [\mathbf{v}_x \quad \mathbf{v}_y \quad \mathbf{v}_z \quad \mathbf{o}]$$

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

$$\mathbf{\Pi} = [a \mathbf{v}_x \quad b \mathbf{v}_y \quad c \mathbf{v}_z \quad \mathbf{o}]$$

- Can fully specify by providing 3 reference points

3D Modeling from a photograph



3D Modeling from a photograph



Camera calibration

Goal: estimate the camera parameters

- Version 1: solve for projection matrix

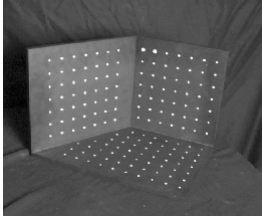
$$\mathbf{X} = \begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{\Pi X}$$

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

Calibration: Basic Idea

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

Chromaglyphs

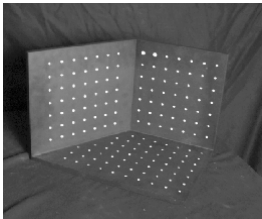


Courtesy of Bruce Culbertson, HP Labs
http://www.hpl.hp.com/personal/Bruce_Culbertson/ibr98/chromagl.htm

Estimating the Projection Matrix

Place a known object in the scene

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



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

Direct Linear Calibration

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

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}$$

$$v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}$$

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

Direct Linear Calibration

$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1 X_1 & -u_1 Y_1 & -u_1 Z_1 & -u_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1 X_1 & -v_1 Y_1 & -v_1 Z_1 & -v_1 \\ & & & & & & & & & & & \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -u_n X_n & -u_n Y_n & -u_n Z_n & -u_n \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -v_n X_n & -v_n Y_n & -v_n Z_n & -v_n \end{bmatrix} \begin{bmatrix} m_{100} \\ m_{101} \\ m_{102} \\ m_{103} \\ m_{110} \\ m_{111} \\ m_{112} \\ m_{113} \\ m_{200} \\ m_{201} \\ m_{202} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Can solve for m_{ij} by linear least squares

- use eigenvector trick that we used for homographies

Direct linear calibration

Advantages:

- Very simple to formulate and solve
- Once you know the projection matrix, can compute intrinsics and extrinsics using matrix factorizations

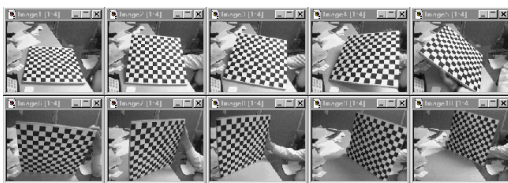
Disadvantages?

- 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 (e.g., Levenberg Marquart)

Alternative: Multi-plane calibration



Images courtesy Jean-Yves Bouguet, Intel Corp.

Advantage

- Only requires a plane
- Don't have to know positions/orientations
- Good code available online!
 - Zhengyou Zhang's web site: <http://research.microsoft.com/~zhang/Calib/>
 - Intel's OpenCV library: <http://www.intel.com/research/mrl/research/opencv/>
 - Matlab version by Jean-Yves Bouguet: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

Summary

Things to take home from this lecture

- Homogeneous coordinates and their geometric intuition
- Homographies
- Points and lines in projective space
 - projective operations: line intersection, line containing two points
 - ideal points and lines (at infinity)
- Vanishing points and lines and how to compute them
- Single view measurement
 - within a reference plane
 - height
- Cross ratio
- Camera calibration
 - using vanishing points
 - direct linear method