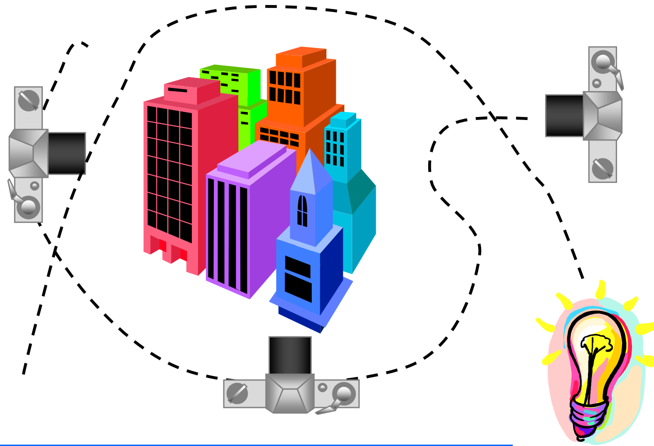
Today: Calibration



- What are the camera parameters?
- Where are the light sources?
- What is the mapping from radiance to pixel color?

Why Calibrate?

Want to solve for 3D geometry

Alternative approach

- Solve for 3D shape without known cameras
  - Structure from motion (unknown extrinsics)
  - Self calibration (unknown intrinsics & extrinsics)

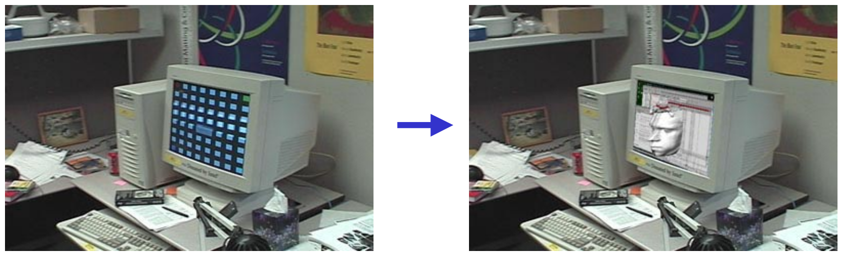
Why bother pre-calibrating the camera?

- Simplifies the 3D reconstruction problem
  - fewer parameters to solve for later on
- Improves accuracy
- Not too hard to do
- Eliminates certain ambiguities (scale of scene)

Applications

3D Modeling

Match Move



Images courtesy of Brett Allen ("Vision for Graphics", winter `01)

Image-Based Rendering

Light field capture
and rendering
(Levoy & Hanrahan, 96)

Camera Parameters

So far we've talked about:

- focal length
- principal (and nodal) point
- radial distortion
- CCD dimensions
- aperture

There is also

- optical center
- orientation
- digitizer parameters

Do we need all this stuff?

Usually simplify to "computable stuff"

- Intrinsics:
  - scale factor ("focal length")
  - aspect ratio
  - principle point
  - radial distortion
- Extrinsics
  - optical center
  - camera orientation

How does this relate to projection matrix?

$$\mathbf{p} = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{MP}$$

Projection Models

Orthographic $\mathbf{M} = \begin{bmatrix} i_x & i_y & i_z & t_x \\ j_x & j_x & j_x & t_y \end{bmatrix}$ $\vec{\text{i}}$ and $\vec{\text{j}}$ orthonormal

Weak Perspective $\mathbf{M} = f\begin{bmatrix} i_x & i_y & i_z & t_x \\ j_x & j_x & j_x & t_y \end{bmatrix}$ $\vec{\text{i}}$ and $\vec{\text{j}}$ orthonormal

Affine $\mathbf{M} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \end{bmatrix}$

Perspective $\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}$

Projective $\mathbf{M} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix}$

The Projection Matrix

Matrix Projection: $\mathbf{p} = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{MP}$

$\mathbf{M}$ can be decomposed into $\mathbf{t} \rightarrow \mathbf{R} \rightarrow$ project $\rightarrow \mathbf{A}$

$$\mathbf{M} = \underbrace{\begin{bmatrix} fa & c & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix}}_{\text{intrinsics } (\mathbf{A})} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{projection}} \underbrace{\begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{0}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}}_{\text{orientation}} \underbrace{\begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{t}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}}_{\text{position}}$$

Goal of Calibration

Learn mapping from 3D to 2D

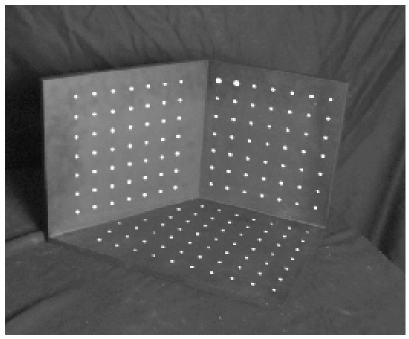
Can take different forms:

- Projection matrix: $\mathbf{p} = \begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{MP}$
- Camera parameters: $\mathbf{p} = \mathbf{f}(X, Y, Z, \mathbf{A}, \mathbf{R}, \mathbf{t})$
- General mapping $\Re^3 \rightarrow \Re^2$

Calibration: Basic Idea

Place a known object in the scene

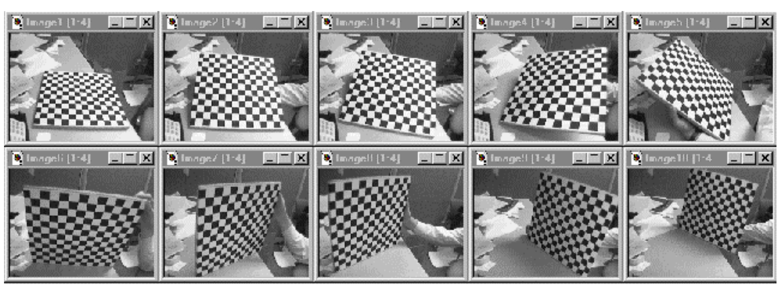
- identify correspondence between image and scene
- compute mapping from scene to image



Problem: must know geometry very accurately

- how to get this info?

Alternative: Multi-plane calibration



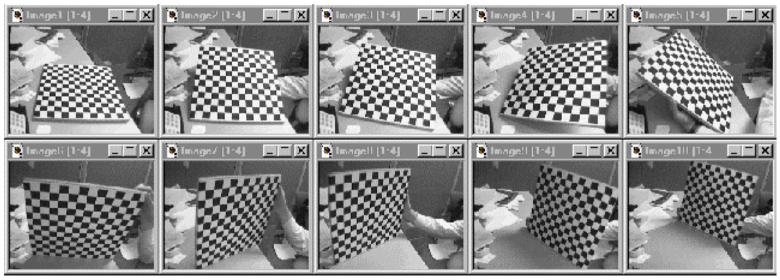
Images courtesy Jean-Yves Bouguet, Intel Corp.

Advantage

Disadvantages?

- 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 Bouget: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

Alternative: Multi-plane calibration



Images courtesy Jean-Yves Bouguet, Intel Corp.

Need 3D -> 2D correspondence

- User provided (lots 'O clicking)
- User seeded (some clicking)
- Fully automatic?

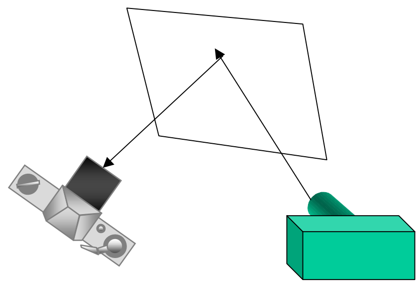
Chromaglyphs

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

Projector Calibration

A projector is the "inverse" of a camera

- has the same parameters, light just flows in reverse
- how to figure out where the projector is?



Basic idea

1. first calibrate the camera wrt. projection screen
2. now we can compute 3D coords of each projected point
3. use standard camera calibration routines to find projector parameters since we known 3D -> projector mapping

Calibration Approaches

Possible approaches (not comprehensive!)

- Experimental design
  - planar patterns
  - non-planar grids
- Optimization techniques
  - direct linear regression
  - non-linear optimization
- Cues
  - 3D -> 2D
  - vanishing points
  - special camera motions
    - » panorama stitching
    - » circular camera movement

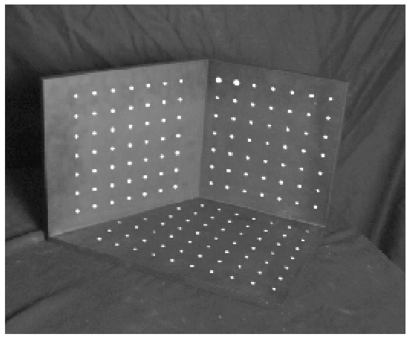
Want

- accuracy
- ease of use
- usually a trade-off

Estimating the Projection Matrix

Place a known object in the scene

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



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

Direct Linear Calibration

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

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

$$u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}$$
$$v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}$$

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

Direct Linear Calibration

$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & u_iX_1 & u_iY_1 & u_iZ_i \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & v_1X_1 & v_1Y_1 & v_1Z_1 \\ & & & & & \vdots & & & & & \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & u_nX_n & u_nY_n & u_nZ_n \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & v_nX_n & v_nY_n & v_nZ_n \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_n \\ v_n \end{bmatrix}$$

Can solve for m_{ij} by linear least squares

$$minimize \left\| \begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & u_iX_1 & u_iY_1 & u_iZ_i \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & v_1X_1 & v_1Y_1 & v_1Z_1 \\ & & & & & \vdots & & & & & \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & u_nX_n & u_nY_n & u_nZ_n \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & v_nX_n & v_nY_n & v_nZ_n \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \end{bmatrix} - \begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_n \\ v_n \end{bmatrix} \right\|$$

What error function are we minimizing?

Nonlinear estimation

Feature measurement equations

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

Minimize "image-space error"

$$e(\mathrm{M}) = \sum_i [(u_i - \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1})^2 + (v_i - \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1})^2]$$

How to minimize e(**M**)?

- Non-linear regression (least squares),
- Popular choice: Levenberg-Marquardt [Press'92]

Camera matrix calibration

Advantages:

- very simple to formulate and solve
- can recover **K** [**R** | **t**] from **M** using *RQ* decomposition [Golub & VanLoan 96]

Disadvantages?

- doesn't model radial distortion
- more unknowns than true degrees of freedom (sometimes)
- need a separate camera matrix for each new view

Separate intrinsics / extrinsics

New feature measurement equations

$$\begin{aligned} \hat{u}_{ij} &= f(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \\ \hat{v}_{ij} &= g(\mathbf{K}, \mathbf{R}_j, \mathbf{t}_j, \mathbf{x}_i) \end{aligned}$$

i – features
j – images

Use non-linear minimization

- e.g., Levenberg-Marquardt [Press'92]

Standard technique in photogrammetry, computer vision, computer graphics

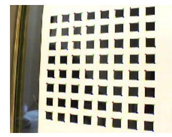
- [Tsai 87] – also estimates κ_1 (freeware @ CMU)
  - http://www.cs.cmu.edu/afs/cs/project/cil/ftp/html/v-source.html
- [Zhang 99] – estimates κ_1, κ_2, easier to use than Tsai
  - code available from Zhang's web site and in Intel's OpenCV
  - http://research.microsoft.com/~zhang/Calib/
  - http://www.intel.com/research/mrl/research/opencv/
  - Matlab version by Jean-Yves Bouget: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

Calibration from (unknown) Planes

What's the image of a plane under perspective?

- a homography (3x3 projective transformation)

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



- preserves lines, incidence, conics

H depends on camera parameters (**A**, **R**, **t**)

$$\mathbf{H} = \mathbf{A}\lfloor \mathbf{r_1} \quad \mathbf{r_2} \quad \mathbf{t} \rfloor$$

where

$$\mathbf{A} = \begin{bmatrix} fa & c & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{R} = \lfloor \mathbf{r_1} \quad \mathbf{r_2} \quad \mathbf{r_3} \rfloor$$

Given 3 homographies, can compute A, R, t

Calibration from Planes

1. Compute homography $\mathbf{H}^i$ for 3+ planes
   - Doesn't require knowing 3D
   - Does require mapping between at least 4 points on plane and in image (both expressed in 2D plane coordinates)
2. Solve for **A, R, t** from $\mathbf{H}^1$, $\mathbf{H}^2$, $\mathbf{H}^3$
   - 1plane if only *f* unknown
   - 2 planes if (f, u_c, v_c) unknown
   - 3+ planes for full ***K***
3. Introduce radial distortion model

$$\begin{aligned} \hat{u} &= u + u(\kappa_1 r^2 + \kappa r^4) \\ \hat{v} &= v + v(\kappa_1 r^2 + \kappa r^4) \end{aligned}$$

where

$$r = \sqrt{(u - u_c)^2 + (v - v_c)^2}$$

Solve for **A**, **R**, **t**, κ_1, κ_2

- nonlinear optimization (using Levenberg-Marquardt)