The Projection “Chain”

Points in the image

Conversion to Pixel Grid

Projection

Transform to camera coordinates

Points in the world
Projection Geometry: Standard Camera Coordinates

• By convention, we place the image in front of the optical center
  – typically we approximate by saying it lies one focal distance from the center
  – in reality this can’t be true for a finite size chip!

• Optical axis is z axis pointing outward

• X axis is parallel to the scanlines (rows) pointing to the right!

• By the right hand rule, the Y axis must point downward

• Note this corresponds with indexing an image from the upper left to the lower right, where the X coordinate is the column index and the Y coordinate is the row index.
Intrinsic Parameters

*Intrinsic Parameters* describe the conversion from unit focal length metric to pixel coordinates (and the reverse)

\[
x_{\text{mm}} = - (x_{\text{pix}} - o_x) s_x \quad \rightarrow \quad -1/s_x x_{\text{mm}} + o_x = x_{\text{pix}}
\]

\[
y_{\text{mm}} = - (y_{\text{pix}} - o_y) s_y \quad \rightarrow \quad -1/s_y y_{\text{mm}} + o_y = y_{\text{pix}}
\]

or

\[
\begin{pmatrix}
x \\
y \\
w
\end{pmatrix}_{\text{pix}} =
\begin{pmatrix}
-1/s_x & 0 & o_x \\
0 & -1/s_y & o_y \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x \\
y \\
w
\end{pmatrix}_{\text{mm}} = K_{\text{int}} p
\]

It is common to combine scale and focal length together as they are both scaling factors; note projection is unitless in this case!
Putting it All Together

Now, using the idea of *homogeneous transforms*, we can write:

\[ p' = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} p \]

R and T both require 3 parameters. These correspond to the 6 *extrinsic parameters* needed for camera calibration.

Then we can write

\[ q = \Pi p' \] for some projection model \( \Pi \)

Finally, we can write

\[ u = K q \] for intrinsic parameters \( K \)
Camera parameters

- Summary:
  - points expressed in external frame
  - points are converted to canonical camera coordinates
  - points are projected
  - points are converted to pixel units

\[
\begin{bmatrix}
U \\
V \\
W
\end{bmatrix} = \begin{bmatrix}
\text{Transformation representing intrinsic parameters} \\
\text{Transformation representing projection model} \\
\text{Transformation representing extrinsic parameters}
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
T
\end{bmatrix}
\]

- point in pixel coords.
- point in metric image coords.
- point in cam. coords.
- point in world coords.
Calibration using a 3D target

Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, pp. 364-374, 1986
Camera Calibration

Calibration = the computation of the camera intrinsic and extrinsic parameters

• General strategy:
  – view calibration object
  – identify image points
  – obtain camera matrix by minimizing error
  – obtain intrinsic parameters from camera matrix

• Error minimization:
  – Linear least squares
    • easy problem numerically
    • solution can be rather bad
  – Minimize image distance
    • more difficult numerical problem
    • solution usually rather good, but can be hard to find
      – start with linear least squares
  – Numerical scaling is an issue

• Most modern systems employ the multi-plane method
  – avoids knowing absolute coordinates of calibration points
Calibration Procedure

• Calibration target: 2 planes at right angle with checkerboard patterns (Tsai grid)
  ▪ We know positions of pattern corners only with respect to a coordinate system of the target
  ▪ We position camera in front of target and find images of corners
  ▪ We obtain equations that describe imaging and contain internal parameters of camera
    • As a side benefit, we find position and orientation of camera with respect to target (camera \textit{pose})
Image Processing of Image of Target

- Canny edge detection
- Straight line fitting to detected linked edges
- Intersecting the lines to obtain the image corners
- Matching image corners and 3D target checkerboard corners
  - By counting if whole target is visible in image
- We get pairs (image point)--(world point)
  \[(x_i, y_i) \rightarrow (X_i, Y_i, Z_i)\]
From Camera Coordinates to World Coordinates
From Camera Coordinates to World Coordinates 2

CM = CO + OM

\[ x_S i + y_S j + z_S k = T_x i + T_y j + T_z k + X_S I + Y_S J + Z_S K \]

\[ x_S = T_x + X_S I.i + Y_S J.i + Z_S K.i \]

\[
\begin{bmatrix}
  x_S \\
  y_S \\
  z_S \\
\end{bmatrix} =
\begin{bmatrix}
  T_x \\
  T_y \\
  T_z \\
\end{bmatrix} +
\begin{bmatrix}
  I.i & J.i & K.i \\
  I.j & J.j & K.j \\
  I.k & J.k & K.k \\
\end{bmatrix}
\begin{bmatrix}
  X_S \\
  Y_S \\
  Z_S \\
\end{bmatrix}
\]
From Camera Coordinates to World Coordinates 3

\[ \text{CM} = \text{OM} - \text{OC} \]
\[ x_s \mathbf{i} + y_s \mathbf{j} + z_s \mathbf{k} = (X_s - X_C) \mathbf{I} + (Y_s - Y_C) \mathbf{J} + (Z_s - Z_C) \mathbf{K} \]
\[ x_s = (X_s - X_C) \mathbf{I} \cdot \mathbf{i} + (Y_s - Y_C) \mathbf{J} \cdot \mathbf{i} + (Z_s - Z_C) \mathbf{K} \cdot \mathbf{i} \]
\[ x_{\text{cam}} = R(X - \tilde{C}) \quad (T = -R\tilde{C}) \]

\( \tilde{C} \) is vector OC expressed in world coordinate system
Homogeneous Coordinates 2

• Here we use \(- \mathbf{R \tilde{C}}\) instead of \(\mathbf{T}\)

\[
\begin{bmatrix}
  x_S \\
  y_S \\
  z_S \\
  1
\end{bmatrix} =
\begin{bmatrix}
  \mathbf{R} & - \mathbf{R \tilde{C}} \\
  \mathbf{0}_3^T & 1
\end{bmatrix}
\begin{bmatrix}
  X_S \\
  Y_S \\
  Z_S \\
  1
\end{bmatrix}
\]
Linear Transformation from World Coordinates to Pixels

- Combine camera projection and coordinate transformation matrices into a single matrix \( P \)

\[
\begin{bmatrix}
    u' \\
    v' \\
    w'
\end{bmatrix} = K \begin{bmatrix}
    I_3 & 0_3
\end{bmatrix}
\begin{bmatrix}
    x_s \\
    y_s \\
    z_s \\
    1
\end{bmatrix}
\Rightarrow
\begin{bmatrix}
    u' \\
    v' \\
    w'
\end{bmatrix} = K \begin{bmatrix}
    I_3 & 0_3
\end{bmatrix} \begin{bmatrix}
    R & -R\tilde{C}
\end{bmatrix}
\begin{bmatrix}
    X_S \\
    Y_S \\
    Z_S \\
    1
\end{bmatrix}
\Rightarrow
\begin{bmatrix}
    u' \\
    v' \\
    w'
\end{bmatrix} = P
\begin{bmatrix}
    X_S \\
    Y_S \\
    Z_S \\
    1
\end{bmatrix}
\Rightarrow \mathbf{x} = PX
Properties of Matrix $P$

• Further simplification of $P$:

$$x = K[I_3 \mid 0_3] \begin{bmatrix} R & -R\tilde{C} \\ 0^T_3 & 1 \end{bmatrix} X$$

$$[I_3 \mid 0_3] \begin{bmatrix} R & -R\tilde{C} \\ 0^T_3 & 1 \end{bmatrix} = [R - R\tilde{C}] = R[I_3 \mid -\tilde{C}]$$

$$x = KR[I_3 \mid -\tilde{C}] X$$

$$P = KR[I_3 \mid -\tilde{C}]$$

• $P$ has 11 degrees of freedom:
  • 5 from triangular calibration matrix $K$, 3 from $R$ and 3 from $\tilde{C}$
  • $P$ is a fairly general 3 x 4 matrix
    • left 3x3 submatrix $KR$ is non-singular
Calibration

1. Estimate matrix $P$ using scene points and their images
2. Estimate the interior parameters and the exterior parameters

$$P = KR[I_3 \mid -\tilde{C}]$$

- Left 3x3 submatrix of $P$ is product of upper-triangular matrix and orthogonal matrix
Finding Camera Translation

- Find homogeneous coordinates of $C$ in the scene
- $C$ is the null vector of matrix $P$
  \[ P \begin{bmatrix} C \end{bmatrix} = 0: \]
  \[
  P = K R \begin{bmatrix} I_3 & -\tilde{C} \end{bmatrix}
  \begin{bmatrix}
  1 & 0 & 0 & -X_c \\
  0 & 1 & 0 & -Y_c \\
  0 & 0 & 1 & -Z_c
  \end{bmatrix}
  \begin{bmatrix}
  X_c \\
  Y_c \\
  Z_c
  \end{bmatrix}
  = \begin{bmatrix} 0 \\
  0 \\
  0
  \end{bmatrix}
  \]
- Find null vector $C$ of $P$ using SVD
  - $C$ is the unit singular vector of $P$ corresponding to the smallest singular value (the last column of $V$, where $P = U D V^T$ is the SVD of $P$)
Finding Camera Orientation and Internal Parameters

• Left 3x3 submatrix $\mathbf{M}$ of $\mathbf{P}$ is of form $\mathbf{M} = \mathbf{K} \mathbf{R}$
  - $\mathbf{K}$ is an upper triangular matrix
  - $\mathbf{R}$ is an orthogonal matrix

• Any non-singular square matrix $\mathbf{M}$ can be decomposed into the product of an upper-triangular matrix $\mathbf{K}$ and an orthogonal matrix $\mathbf{R}$ using the RQ factorization
  - Similar to QR factorization but order of 2 matrices is reversed
RQ Factorization of $\mathbf{M}$

\[
\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & -s \\ 0 & s & c \end{bmatrix}, \quad \mathbf{R}_y = \begin{bmatrix} c' & 0 & s' \\ 0 & 1 & 0 \\ -s' & 0 & c' \end{bmatrix}, \quad \mathbf{R}_z = \begin{bmatrix} c'' & -s'' & 0 \\ s'' & c'' & 0 \\ 0 & 0 & 1 \end{bmatrix}
\]

- Compute $c = -\frac{m_{33}}{(m_{32}^2 + m_{33}^2)^{1/2}}$, $s = \frac{m_{32}}{(m_{32}^2 + m_{33}^2)^{1/2}}$

- Multiply $\mathbf{M}$ by $\mathbf{R}_x$. The resulting term at (3,2) is zero because of the values selected for $c$ and $s$

- Multiply the resulting matrix by $\mathbf{R}_y$, after selecting $c'$ and $s'$ so that the resulting term at position (3,1) is set to zero

- Multiply the resulting matrix by $\mathbf{R}_z$, after selecting $c''$ and $s''$ so that the resulting term at position (2,1) is set to zero

$\mathbf{M} \mathbf{R}_x \mathbf{R}_y \mathbf{R}_z = \mathbf{K} \Rightarrow \mathbf{M} = \mathbf{K} \mathbf{R}_z^T \mathbf{R}_y^T \mathbf{R}_x^T = \mathbf{K} \mathbf{R}$
Computing Matrix $P$

- Use corresponding image and scene points
  - 3D points $X_i$ in world coordinate system
  - Images $x_i$ of $X_i$ in image
- Write $x_i = P X_i$ for all $i$
- Similar problem to finding projectivity matrix $H$ (i.e. homography)
Computation of $P$

- $x_i = P X_i$ involves homogeneous coordinates, thus $x_i$ and $P X_i$ just have to be proportional: $x_i \times P X_i = 0$

- Let $p_1^T, p_2^T, p_3^T$ be the 3 row vectors of $P$

\[
P X_i = \begin{bmatrix}
p_1^T X_i \\
p_2^T X_i \\
p_3^T X_i
\end{bmatrix}
\]

\[
x_i \times P X_i = \begin{bmatrix}
v'_i p_3^T X_i - w'_i p_2^T X_i \\
w'_i p_1^T X_i - u'_i p_3^T X_i \\
u'_i p_2^T X_i - v'_i p_1^T X_i
\end{bmatrix}
\]

\[
\Rightarrow \begin{bmatrix}
0_4^T \\
w'_i X_i^T \\
- v'_i X_i^T
\end{bmatrix}
\begin{bmatrix}
- w'_i X_i^T \\
v'_i X_i^T \\
- u'_i X_i^T
\end{bmatrix}
\begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix}
= 0
\]

$\begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix}$ is a $12 \times 1$ vector
Computation of \( P \)

- Third row can be obtained from sum of \( u'_i \) times first row - \( v'_i \) times second row

\[
\begin{bmatrix}
0^T_4 & -w'_i X_i^T & v'_i X_i^T \\
w'_i X_i^T & 0^T_4 & -u'_i X_i^T \\
-v'_i X_i^T & u'_i X_i^T & 0^T_4
\end{bmatrix}
\begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix} = 0
\]

- So we get 2 independent equations in 11 unknowns (ignoring scale)

- With 6 point correspondences, we get enough equations to compute matrix \( P \)

\[
A \ p = 0
\]
Solving $A \mathbf{p} = 0$

- Linear system $A \mathbf{p} = 0$
- When possible, have at least 5 times as many equations as unknowns (28 points)
- Minimize $\| A \mathbf{p} \|$ with the constraint $\| \mathbf{p} \| = 1$
  - $\mathbf{p}$ is the unit singular vector of $A$ corresponding to the smallest singular value (the last column of $V$, where $A = U \Sigma V^T$ is the SVD of $A$)
- Called Direct Linear Transformation (DLT)
Improving $P$ Solution with Nonlinear Minimization

- Find $p$ using DLT
- Use as initialization for nonlinear minimization of $\sum d(x_i, PX_i)^2$
  - Use Levenberg-Marquardt iterative minimization
Multi-planar calibration

Z. Zhang. A flexible new technique for camera calibration.
Alternative: multi-plane calibration

Advantage

• Only requires a plane
• Don’t have to know positions/orientations
• Good code available online!
  – 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
  – Zhengyou Zhang’s web site:  http://research.microsoft.com/~zhang/Calib/
Parameters

• (same model as before - just different notations)

Projection equation:

\[ s\tilde{m} = A[R \ t]\tilde{M} \]

Intrinsic parameters:

\[ A = \begin{bmatrix} \alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \]
Planar homography

- Assume world coordinate system is such that Z=0 on the calibration pattern

- Projected pattern point:

\[ s\tilde{m} = A[R \ t]\tilde{M} \]

\[ \begin{bmatrix} u \\ v \end{bmatrix} = A \begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \end{bmatrix} \]

\[ = A \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} . \]

Pattern and its image are related by homography:

\[ s\tilde{m} = H\tilde{M} \quad \text{with} \quad H = A \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \]
Homography H

• Compute H for each image using known correspondences
Constraints on internal parameters

• Noting

\[ H = [h_1 \ h_2 \ h_3] \]

• And using the equation

\[ [h_1 \ h_2 \ h_3] = \lambda A [r_1 \ r_2 \ t] \]

• One derives two equations on internal parameters only:

\[ h_1^T A^{-T} A^{-1} h_2 = 0 \]
\[ h_1^T A^{-T} A^{-1} h_1 = h_2^T A^{-T} A^{-1} h_2 \]
Computing Intrinsics

• Rotation Matrix is orthogonal....

\[ r_i^T r_j = 0 \]
\[ r_i^T r_i = r_j^T r_j \]

• Write the homography in terms of its columns...

\[ h_1 = sA r_1 \]
\[ h_2 = sA r_2 \]
\[ h_3 = sA t \]
Computing Intrinsics

• Derive the two constraints:

\[ h_1 = sA r_1 \]
\[ \frac{1}{s} A^{-1} h_1 = r_1 \]
\[ \frac{1}{s} A^{-1} h_2 = r_2 \]

\[ r_1^T r_2 = 0 \]
\[ h_1^T A^{-T} A^{-1} h_2 = 0 \]

\[ r_1^T r_1 = r_2^T r_2 \]
\[ h_1^T A^{-T} A^{-1} h_1 = h_2^T A^{-T} A^{-1} h_2 \]
Closed-Form Solution

Let $B = A^{-T}A^{-1} = \begin{bmatrix}
\frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2 \beta} & \frac{v_0 \gamma - u_0 \beta}{\alpha^2 \beta} \\
-\frac{\gamma}{\alpha^2 \beta} & \frac{\gamma^2}{\alpha^2 \beta^2} + \frac{1}{\beta^2} & \frac{\gamma (v_0 \gamma - u_0 \beta)}{\alpha^2 \beta^2} - \frac{v_0}{\beta^2} \\
\frac{v_0 \gamma - u_0 \beta}{\alpha^2 \beta} & \frac{\gamma (v_0 \gamma - u_0 \beta)}{\alpha^2 \beta^2} - \frac{v_0}{\beta^2} & \frac{(v_0 \gamma - u_0 \beta)^2}{\alpha^2 \beta^2} + \frac{v_0^2}{\beta^2} + 1
\end{bmatrix}$

- Notice $B$ is symmetric, 6 parameters can be written as a vector $b$.
- From the two constraints, we have $h_1^T B h_2 = v_{12} b$

\[
\begin{bmatrix}
v_{ij}^T \\
(v_{11} - v_{22})^T
\end{bmatrix} b = 0;
\]

- Stack up $n$ of these for $n$ images and build a $2n \times 6$ system.
- Solve with SVD.
- Intrinsic parameters “fall-out” of the result easily using algebra
Solve for extrinsic parameters

• Using equation (A is known)

\[
\begin{bmatrix}
  h_1 & h_2 & h_3
\end{bmatrix} = \lambda A
\begin{bmatrix}
  r_1 & r_2 & t
\end{bmatrix}
\]

• One obtains

\[
\lambda = 1/\|A^{-1}h_1\| = 1/\|A^{-1}h_2\|
\]

• Issue: due to noise \(r_1\) and \(r_2\) may not be orthogonal

\[
\begin{align*}
  r_1 &= \lambda A^{-1}h_1 \\
  r_2 &= \lambda A^{-1}h_2 \\
  r_3 &= r_1 \times r_2 \\
  t &= \lambda A^{-1}h_3
\end{align*}
\]

(-> board)
Non-linear Refinement

• Closed-form solution minimized algebraic distance.

• Since full-perspective is a non-linear model
  
  – Can in
  
  \[ 
  \sum_{i=1}^{n} \sum_{j=1}^{m} \left\| m_{ij} - \hat{m}(A, R_k, T_k, M_j) \right\|^2 
  \]
  
  – Use maximum likelihood inference for our estimated parameters.
Notes

• Improvement (like for previous method)
  – use non-linear optimization to refine the results
  – compute radial distortion

• Degenerate configurations?
  – Pure translation
  – Parallel planes