INFO0013 Computer Vision

Camera Parameters and Their Calibration

Justus H. Plater

How to insert an artificial item into an image of a real scene?

Parameters of a Projective Camera

Reminder: Our Camera Matrix

\[ P = \begin{bmatrix} f' & 0 & 0 & 0 \\ 0 & f' & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \]

\[ x = P X \]

What do the other coefficients do? Do we need them?
**Internal Parameters**

Conventionally, one uses an *inverted pinhole model* with the image plane placed in front of the projection center, and positive z coordinates.

1. Principal point offset
2. Pixel aspect ratio
3. Skew

Matrix of intrinsic parameters $K = \text{camera calibration matrix}$

$x = K[I]_0 X$

See a demo.

---

**Positioning the Camera in Space**

Coordinate system transformations, first in 2D:

- Translation: $x' = x + t$
- Rotation: $x' = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} x$

In homogeneous coordinates?

**To map world points into the camera frame:**

- move the world frame into the camera frame
- *move the camera frame into the world frame*
**Pure Rotation in 3D**

![Diagram of Pure Rotation in 3D]

*Figure from Forsyth/Horn lecture notes*

**3 Elementary Axis Rotations [7]**

![Diagram of 3 Elementary Axis Rotations]

*Figure from Forsyth/Horn lecture notes*

**General Rigid Transformation**

![Diagram of General Rigid Transformation]

*Figure from Forsyth/Horn lecture notes*

**External Parameters**

- Position $c$ of the camera center
- Orientation $R$ of the camera

*Perspective Projection Matrix* $P = \text{Projective Camera}$

\[ x = P \chi \]

\[ P = KR[I - c] = K[R|t]\]

See a demo.

To align the camera frame with the world frame,

1. translate the camera to the origin of the world frame,
2. rotate it about the origin to align the $x$ and $z$ axes,
3. translate, scale and shear to make the $y$ axis and units match.
Affine Cameras

- Moving back along the principal axis by a distance factor of $k$ w.r.t.
  the world origin multiplies $p_{34}$ by $k$.
- Zooming in by a factor of $k$ multiplies $\mathbf{K}$ on the right
  by $\text{diag}(k, k, 1)$.
- If one then divides $P$ by $k$, the last row eventually
  becomes $[0, 0, 0, p_{34}]$.

Good approximation if all distances are large.

---

Geometric Camera Calibration

---

Calibration Rig

(Figure from Forsyth/Ponce lecture notes)

---

Estimating a Camera $P$ with the Direct Linear Transform (DLT)

- 11 degrees of freedom
- Minimal, linear solution: 5 1/2 point correspondences
  expressed as $x_i \times PX_i = 0$, giving rise to a system
  $Ap = 0$ with rank-11 coefficient matrix $A$.
  To help us write down $A$ concisely, let $p^T_i$ denote the
  $j$th row of matrix $P$, such that $PX_i = [p^T_1 X_1 \ldots p^T_n X_n]$.
- Overdetermined solution minimizing algebraic error
  $\|Ap\|^2 = \sum \sigma_i^3(x_i, PX_i) = \sum \|x_i \times PX_i\|^2$
**Error Management**

- Minimize geometric error by nonlinear optimization
  - transfer error: \( \sum d^2(\mathbf{x}_i, \mathbf{pX}_i) \)
  - reprojection error: \( \sum (d^2_{\text{mah}}(\mathbf{x}_i, \hat{\mathbf{x}}_i) + d^2_{\text{mah}}(\mathbf{x}_i, \tilde{\mathbf{x}}_i)) \)
    subject to \( \hat{\mathbf{x}}_i = \mathbf{p}\hat{\mathbf{X}}_i \)
- Before estimating, normalize (translate centroid to origin, scale such that the average norm is \( \sqrt{2} \)): \( \mathbf{T} \) and \( \mathbf{U} \) normalize the \( \mathbf{x}_i \) and the \( \hat{\mathbf{X}}_i \); \( \mathbf{P} = \mathbf{T}^{-1}\mathbf{pU} \).
- Incorporate known constraints on \( \mathbf{K} \): Parametrize \( \mathbf{P} \) appropriately.
- Watch out for degenerate point configurations!

**Determining the camera parameters**

\[
\mathbf{P} = [\mathbf{A}\mid \mathbf{b}] = \mathbf{A}[\mathbf{I} - \mathbf{c}] = \mathbf{K}[\mathbf{R}[\mathbf{I} - \mathbf{c}] \]

- Camera center: \( \mathbf{c} = -\mathbf{A}^{-1}\mathbf{b} \)
- \( \mathbf{K}, \mathbf{R} \): RQ-decomposition of \( \mathbf{A} \)

**Correcting for Radial Distortion**

\[
\hat{\mathbf{p}} = \mathbf{c} + L(r)|\mathbf{p} - \mathbf{c}|
\]

\[
r^2 = (\mathbf{p} - \mathbf{c})^T(\mathbf{p} - \mathbf{c})
\]

\[
L(r) = 1 + K_2r^2 + K_3r^3 + \ldots
\]

where \( \hat{\mathbf{p}} \) and \( \mathbf{p} \) are the corrected and measured coordinates, and \( \mathbf{c} \) is the center of radial distortion (which may not coincide with the principal point).

During nonlinear camera calibration, simply add all the \( K_i \) and perhaps \( \mathbf{c} \) to the unknown parameters!

**Note**

If the pixel aspect ratio is not unity, it must be taken into account when computing \( r^2 \).

**Calibration in Practice**

Instead of relying on a calibration rig with point features at known 3D coordinates,

- use auto-calibration to find the internal parameters by taking several pictures of any sufficiently richly structured scene,
- calibrate for radial distortion by choosing a scene that contains lots of straight lines, and arrange for these lines to come out straight in the image.
Solving an Overdetermined Homogeneous Linear System

Given the linear system $Ax = 0$, we seek the coefficients $x$ that minimize the squared-error function $E = e^T e$, where $e = Ax$.

To avoid the trivial solution $x = 0$, we need to impose a constraint on $x$, e.g., $x^T x = 1$.

In this case, the solution is given by the eigenvector corresponding to the minimum eigenvalue of $A^T A$.

Nice numerical methods exist. Most simply, $x$ is the last column of $V$, where $A = USV^T$ is the SVD of $A$.

Singular Value Decomposition

SVD decomposes any matrix $A$ into three matrices such that $A = USV^T$.

$S$ is diagonal; its elements $s_{ii}$ are called the singular values of $A$. They are generally arranged in nonincreasing order.

The columns $v_j$ of the orthonormal matrix $V$ are called the singular vectors of $A$.

It follows that $Av_j = s_{ii} u_j$, and in particular, if $s_{ii} = 0$, $Av_j = 0$.

Note

You don’t want to code sophisticated numerical methods yourself. See the Numerical Recipes, Matlab, vision libraries ...

Nonlinear Systems

Consider a nonlinear, differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}^p$ and the general system $f(x) = 0$.

- There is no general method for finding the global minimum of the squared error $E(x) = \sum_{i=1}^{p} f_i^2(x)$

- There are methods that look for local minima of the error function (by linearization using a first-order Taylor expansion around a current estimate $x$).
Methods for Nonlinear Systems

- For $p = q$, Newton's familiar method.
- For $p > q$, essentially do Newton on the gradient of the least-squares error function (which thus requires its Hessian).
- There are approximations to the latter method that do not require the Hessian and are thus more flexible and more robust, first of all the Levenberg-Marquardt algorithm.

Summary

- Knowing $P$, we can now render an artificial item and paste it into the scene!
- These are classical methods.
- There is lots of insight to be gained by deeper math.
- There are interesting practical methods (autocalibration without known world coordinates).