A Survey of Modern Camera Calibration Methods

Tim Yates
MSCS, St. Olaf College

March 15, 2012

1 Introduction

Camera calibration is the process of determining a map from points in the 3D space around a camera to 2D points on an image. In reality, this map is a function of the precise physical characteristics of the lenses and the image sensor, refracting photons onto a plane in a (mostly) predictable way. Because of the complexity, variety, and imprecision in camera construction, however, we usually assume much simpler models to estimate the true map. Once we choose a model, the task of calibration is determining a set of model parameters that give us the most likely explanation for the data that we see.

2 Models

2.1 Pinhole Camera Model

The most basic and widely used camera model is the pinhole model, which describes the image as the projection of light rays through the camera center (the pinhole) onto a plane at a certain distance (the focal length, f). We can equivalently describe the plane as being in front of the camera, and if we define a *camera coordinate system* with the camera center at the origin, the x and y axes aligned with the image axes, and the z axis passing through the nearest point from the origin to the image plane, we can say that:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{f}{z} \begin{bmatrix} x \\ y \end{bmatrix}$$

where (u, v) is the point on the image plane and (x, y, z) is the point in the camera coordinate system. This relationship can easily be proved with similar triangles (Lengyel 2002, p. 95). Note that dividing by z is equivalent to treating the point (x, y, z) as part of the projective space $\mathbb{P}^2$ and the point $(u, v, 1)$ as its corresponding affline coordinate (Mohr and Triggs 1996). If we consider the possibilities of rectangular pixels and skew (one axis not quite being orthogonal to the other) and add an offset so that the z axis does not need to pass through the center of the image coordinate system, we have the expanded relationship:

$$\underbrace{\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}}_{\tilde{\mathbf{u}}} \sim \underbrace{\begin{bmatrix} \alpha f & \gamma & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \underbrace{\begin{bmatrix} x \\ y \\ z \end{bmatrix}}_{\mathbf{x}}$$

where γ is the skew factor, α is the pixel aspect ratio, (u_0, v_0) is the image plane center or *principal point*, and $\sim$ means equality in projective space (i.e., equality up to a scale factor). $\mathbf{K}$ is known as the *camera matrix* and represents the parameters of the model, which has 5 degrees of freedom.

2.2 Distortion Models

The pinhole model gives a very simple projective relationship between 2D and 3D points that makes it easy to map from one to the other. It also produces an image that corresponds closely to the way we naturally perceive the world around us. And while the vast majority of cameras strive to attain the properties of a pinhole projection, they all fall short. Thanks to the basic properties of lenses, not to mention imperfections and misalignments, their images show some degree of *distortion*—deviation from an ideal projection, whose amount depends on the location in the image.

There are many models to describe these distortions. Some make strong assumptions about the causes or effects of distortion (for example, that distortion is radially symmetrical). Some just try to estimate a map directly from the data. These latter ones, the *nonparametric* or *model-free* approaches (Hartley and Kang 2005), have the advantage of flexibility, but the significant disadvantage that a large amount of high-precision data is required to make accurate estimations. *Parametric methods*, on the other hand, need less data, but if their assumptions are incorrect, they can be wildly inaccurate (Alpaydin 2010, p. 162–3).

The most widely used distortion model in calibration is the Brown-Conrady model, which has three components: *radial*, *decentering* (*tangential*), and *thin lens* distortion. Each of these distortion components has a physical basis, but they still simplify real lens systems (Weng et al. 1992). The combined distortion model can be expressed as:

$$\begin{aligned}\dot{\mathbf{u}}_d &= (1 + k_1\|\dot{\mathbf{u}}\|^2 + k_2\|\dot{\mathbf{u}}\|^4 + \cdots)\dot{\mathbf{u}} \\ &\quad + (1 + p_3\|\dot{\mathbf{u}}\|^2 + p_4\|\dot{\mathbf{u}}\|^4 + \cdots)\begin{bmatrix} 2\dot{u}^2 + \|\dot{\mathbf{u}}\|^2 & 2\dot{u}\dot{v} \\ 2\dot{u}\dot{v} & 2\dot{v}^2 + \|\dot{\mathbf{u}}\|^2 \end{bmatrix}\begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \\ &\quad + \|\dot{\mathbf{u}}\|^2 \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}\end{aligned}$$

where $\|\cdot\|$ denotes the ℓ^2 or Euclidean norm and k_i, p_i, and s_i are coefficients for the radial, decentering, and thin lens distortion, respectively. $\dot{\mathbf{u}}$ is the image coordinate with respect to the *distortion center*:

$$\dot{\mathbf{u}} = \mathbf{u} - \mathbf{u}_{d0}$$

which may or may not be the same as the principal point (Hartley and Kang 2005). $\dot{\mathbf{u}}_d$ is its corresponding distorted point.

As is apparent from the equation above, the Brown-Conrady model can have any number of radial and decentering distortion coefficients. Often, only the first two decentering coefficients are used (if any), and the thin lens distortion is left out completely. One or two radial coefficients are typical. This is partly for convenience and simplicity, but also because researchers have historically assumed that higher order terms would be too miniscule to matter (Zhang 2000). With higher-resolution cameras, increased processing power, and better estimation methods, some recent studies have called this into question. For instance, De Villiers et al. (2008) saw order-of-magnitude accuracy increases with models using 5 or more parameters—but only with the right minimization algorithms (discussed later).

A few parametric methods are variations on the Brown-Conrady model. They often ignore all effects but radial distortion. The *division model* uses the reciprocal of the radial power series (Fitzgibbon 2001):

$$\hat{\mathbf{u}}_d = \frac{1}{1 + k_1\|\hat{\mathbf{u}}\|^2 + \cdots}\hat{\mathbf{u}}$$

OpenCV[1] uses a combination of the traditional BC and division models from version 2.2 onward.

The *rational function model* (Claus and Fitzgibbon 2005) uses three polynomials:

$$\hat{u}_d = \frac{\mathcal{U}\mathbf{a}}{\mathcal{U}\mathbf{c}} = \frac{a_1\hat{u}^2 + a_2\hat{u}\hat{v} + \cdots + a_5\hat{v} + a_6}{c_1\hat{u}^2 + c_2\hat{u}\hat{v} + \cdots + c_5\hat{v} + c_6}$$
$$\hat{v}_d = \frac{\mathcal{U}\mathbf{b}}{\mathcal{U}\mathbf{c}} = \frac{b_1\hat{u}^2 + b_2\hat{u}\hat{v} + \cdots + b_5\hat{v} + b_6}{c_1\hat{u}^2 + c_2\hat{u}\hat{v} + \cdots + c_5\hat{v} + c_6}$$

where $\mathcal{U} = \begin{bmatrix}\hat{u}^2 & \hat{u}\hat{v} & \hat{v}^2 & \hat{u} & \hat{v} & 1\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ are coefficient vectors. An advantage of this model is that it can be solved linearly once $\mathcal{U}$ is computed.

Generally, a *polynomial model* uses polynomials in two variables. A common choice is the *bicubic model* (Grompone von Gioi et al. 2011):

$$\hat{u}_d = \mathcal{U}\mathbf{a} = a_1\hat{u}^3 + a_2\hat{u}^2\hat{v} + a_3\hat{u}\hat{v}^2 + a_4\hat{v}^3 + a_5\hat{u}^2 + a_6\hat{u}\hat{v} + a_7\hat{v}^2 + a_8\hat{u} + a_9\hat{v} + a_{10}$$
$$\hat{v}_d = \mathcal{U}\mathbf{b} = b_1\hat{u}^3 + b_2\hat{u}^2\hat{v} + b_3\hat{u}\hat{v}^2 + b_4\hat{v}^3 + b_5\hat{u}^2 + b_6\hat{u}\hat{v} + b_7\hat{v}^2 + b_8\hat{u} + b_9\hat{v} + b_{10}$$

where $\mathcal{U}$, $\mathbf{a}$, and $\mathbf{b}$ are defined in a similar manner. Like the polynomials of the rational function model, these can also be solved linearly.

3 Parameter Estimation

Once a model is chosen, the problem of calibration becomes one of estimating the parameters of the model as closely as possible. The goal of parameter estimation is to find the *most likely* set of parameters given a set of observations. If we have a model $\boldsymbol{\beta} = f(\boldsymbol{\theta}, \boldsymbol{\alpha})$ where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are vectors of inputs and outputs, and $\boldsymbol{\theta}$ is a set of parameters that determine the model,[2] we first find a set of known inputs $\boldsymbol{\alpha}_i$ and known outputs $\boldsymbol{\beta}_i$. Assuming the output is corrupted by normally-distributed noise[3], the most likely estimation of the parameters $\hat{\boldsymbol{\theta}}$ is the value that minimizes the *mean squared error* (Alpaydin 2010, p. 73–4):

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_i \left[\boldsymbol{\beta}_i - f(\boldsymbol{\theta}, \boldsymbol{\alpha}_i)\right]^2$$

The method used to find such parameters depends on the model. Some models have closed-form algebraic solutions, which are (usually) straightforward to compute and (usually) have predictable properties. This class includes functions that can be expressed as a

[1] http://opencv.itseez.com/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html

[2] Mathematically speaking, the difference between "parameters" and "inputs" is arbitrary. For our purposes, inputs are what we know, and parameters are what we want to determine. Also, inputs are likely to vary, while we assume that parameters are constant.

[3] This assumption is important. If, for instance, the output (or input) is corrupted by uniformly-distributed noise because of false positives, then the least squares method does *not* give the most likely estimation. In these cases, a more robust method like RANSAC should be used (Claus and Fitzgibbon 2005).

set of linear equations in their inputs—or in a function of their inputs, as with polynomial models (Alpaydin 2010, p. 74–5). For other models, no tractable closed-form solution exists, or the solution is undesirable for some reason. In these cases, iterative techniques are used in order to successively refine estimates (Weng et al. 1992).

3.1 Inputs and Outputs

In camera calibration, the inputs and outputs are typically *point correspondences*. If we know something about the 3D geometry of the scene that our camera is viewing, and if we can identify points in our image that relate to that geometry, then we can use 3D points as our inputs and image points as our output.

One way to know something about the geometry of the scene is to construct that geometry yourself. This is the idea behind *calibration targets*: for example, a grid of equally spaced lines or a set of regular solids with known measurements. Other aspects of geometry can be implicit: parallel lines in a room, corresponding points in a stereo images, or abstract features like the "image of the absolute conic" (Mohr and Triggs 1996; Zhang 2000). While actual points might not be known in these cases, they can provide constraints on the possible parameters that can be combined into a direct solution.

3.2 Solving the Pinhole Model

Zhang (2000) gives a simple approach (based on Tsai 1987) that is widely used. Taking multiple images of a flat calibration target with equally-spaced features (usually corners of a checkerboard or grid), we can position the world coordinates so that the origin is at one of the feature points and the target lies on the plane $z = 0$. We can then add an unknown rigid body transformation for each image to map these points into the camera coordinate system required by the pinhole model:

$$\tilde{\mathbf{u}} = \mathbf{K}\mathbf{T}\tilde{\mathbf{x}}$$

where $\tilde{\mathbf{x}} = \begin{bmatrix} x & y & 0 & 1 \end{bmatrix}^{\mathrm{T}}$ and $\mathbf{T}$ is a 3×4 transformation matrix that can be broken into rotation and translation components:

$$\mathbf{T} = \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \end{bmatrix}$$

But since the third element of $\tilde{\mathbf{x}}$ is zero, neither it nor $\mathbf{r}_3$ ever affects the result, so we can drop both from the variables' definitions. Combining the two matrix multiplications into one gives us an unknown *collineation* or *homography* (Mohr and Triggs 1996) $\mathbf{H} \equiv \mathbf{K}\mathbf{T} = \begin{bmatrix} \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \end{bmatrix}$. Since the model is now a simple linear system $\tilde{\mathbf{u}} = \mathbf{H}\tilde{\mathbf{x}}$, we can solve the homography directly with three point correspondences or use multivariate regression with more of them (Alpaydin 2010, p. 103–5).

Because the rotation vectors are orthonormal (Lengyel 2002, p. 55–6), we can show that:

$$\mathbf{h}_1^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\mathbf{h}_2 = 0$$
$$\mathbf{h}_1^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\mathbf{h}_1 = \mathbf{h}_2^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\mathbf{h}_2$$

Through some trickery involving expanding and regrouping these constraints into an alternative form, Zhang shows that these relationships are enough to recover the parameters of $\mathbf{K}$ when $\mathbf{H}_i$ are solved from $i \geq 3$ images of the same points under different transformations. Once $\mathbf{K}$ is known, $\mathbf{T}_i$ can be teased out as well. The details of this process are not important since, as we will see, this solution is only used as an estimate.

3.3 Solving Nonlinear Models

While this (and many other) algebraic solutions to the pinhole model are elegant and efficient, they do not, in general, work in the presence of distortion. This is because adding distortion is equivalent to composing the pinhole and distortion models:

$$\dot{\mathbf{u}}_d = d(\boldsymbol{\theta}_d, p(\boldsymbol{\theta}_p, \mathbf{x}))$$

If we find corresponding points in the image and scene, our image points are actually outputs from the distortion model, not the pinhole model. In this case, we have two options:

- Solve the distortion model separately, then reverse the distortion to find the undistorted points (outputs from the pinhole model). This is theoretically possible because the pinhole model always maps straight lines in space to straight lines in an image. Any curvature observed in these lines is due to distortion, a fact that many methods exploit to find the parameters using implicit inputs (Bailey 2002; Hartley and Kang 2005; Wang et al. 2006).
- Combine the pinhole and distortion models into a single model. For some distortion models, these combination models can be solved directly (Claus and Fitzgibbon 2005; Fitzgibbon 2001). For others, we need to use iterative methods (Zhang 2000).

The most general iterative technique is to use a numerical algorithm designed for minimizing nonlinear equations. These algorithms are treated as "black boxes" in the sense that they find parameters that minimize any well-behaved function that you give them—in this case, the mean-squared error of the combined model. They are typically variations on using Newton's method to find roots in the first derivative. As such, they are only able to find local minima, so we need to give an initial guess in the neighborhood of the global minimum (Weng et al. 1992). This is the purpose of directly solving the pinhole model (Zhang 2000).

Historically, the algorithm of choice has been Levenberg-Marquardt (Marquardt 1963). However, De Villiers et al. (2008) found that, of several alternatives that they tested, Levenberg-Marquardt gave the poorest accuracy when minimizing the Brown-Conrady distortion model. Other algorithms like Fletcher-Reeves (1964) and Leapfrog (Snyman 1983) gave much better results, especially with high-order models.

3.4 Numerical Stability

One significant caveat when applying any of these methods is the possibility of numerical instability. When a parameter vector or a matrix being solved includes values in a wide range (e.g., distortion coefficients and translations), this poor *conditioning* can magnify errors in the input and produce inaccurate results (Zhang 2000). One strategy that mitigates this problem is normalizing input values to make their expected variances equal, then scaling them in the output (Claus and Fitzgibbon 2005; De Villiers et al. 2008).

Numerical instability can also be a problem with high-order distortion models, particularly when combined with unknown transformations (Weng et al. 1992). However, many good results with high-order models show that this is not always an issue (De Villiers et al. 2008; Grompone von Gioi et al. 2011).

Another potential problem is dependence in the parameters. Both the general technique of maximum likelihood estimation (Alpaydin 2010, p. 62) and the numerical methods (Marquardt 1963) assume that parameters are identically and independently distributed. It is

important to construct the parameters so that they are as independent as possible. For example, a rotation should be parameterized as a 3-vector, since it only has 3 degrees of freedom (Zhang 2000).

References

Ethem Alpaydin. *Introduction to Machine Learning*. Adaptive Computation and Machine Learning. MIT, Cambridge, MA, 2 edition, 2010. ISBN 970262012430. 2, 3, 4, 5

D.G. Bailey. A new approach to lens distortion correction. *Proceedings Image and Vision Computing New Zealand 2002*, page 59–64, 2002. 5

D. Claus and A.W. Fitzgibbon. A rational function lens distortion model for general cameras. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, page 213–219, 2005. 3, 5

J.P. De Villiers, F.W. Leuschner, and R. Geldenhuys. Centi-pixel accurate real-time inverse distortion correction. In *Proceedings of SPIE*, 2008. doi: 10.1117/12.804771. 2, 5

A.W. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, page I–125, 2001. 3, 5

R. Fletcher and CM Reeves. Function minimization by conjugate gradients. *The computer journal*, 7(2):149–154, 1964. 5

R. Grompone von Gioi, P. Monasse, J.-M. Morel, and Z. Tang. Self-consistency and universality of camera lens distortion models, May 2011. Prepublication. 3, 5

R.I. Hartley and S.B. Kang. Parameter-free radial distortion correction with centre of distortion estimation. In *Tenth IEEE International Conference on Computer Vision*, volume 2, page 1834–1841, 2005. 2, 5

Eric Lengyel. *Mathematics for 3D Game Programming and Computer Graphics*. Game Development Series. Charles River Media, Hingham, MA, 2002. ISBN 1584500379. 1, 4

D.W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963. 5

Roger Mohr and Bill Triggs. Projective geometry for image analysis, July 1996. URL `http://www.cse.iitd.ernet.in/~suban/vision/tutorial/isprs96.html`. 1, 4

J.A. Snyman. An improved version of the original leap-frog dynamic method for unconstrained minimization: LFOP1(b). *Applied Mathematical Modelling*, 7(3):216–218, June 1983. ISSN 0307-904X. doi: 10.1016/0307-904X(83)90011-2. URL `http://www.sciencedirect.com/science/article/pii/0307904X83900112`. 5

R. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, 1987. 4

J. Wang, F. Shi, J. Zhang, and Y. Liu. Camera calibration from a single frame of planar pattern. In *Advanced concepts for intelligent vision systems*, page 576–587, 2006. 5

J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10): 965–980, 1992. 2, 4, 5

Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000. 2, 4, 5, 6