Robot Aerobics: Four Easy Steps to a More Flexible Calibration

Daniel E. Stevenson and Margaret M. Fleck
Department of Computer Science
University of Iowa
Iowa City, IA 52242, USA

Abstract

In this paper, we present a method for calibrating intrinsic and extrinsic camera parameters. This algorithm can easily be modified by other users to suit their particular calibration needs, without requiring a high precision calibration target or complicated linear algebra. The algorithm uses controlled motions and a single light source to simulate calibration targets in convenient 3D locations. These convenient calibration targets enable us to simplify the calibration algorithm and gather dense data for lens distortion. Dense data makes the distortion correction more accurate than traditional low-order polynomial fits, and allows us to calibrate wide-angle lenses (> 70° field of view).

1 Introduction

A wide variety of computer vision algorithms require some type of camera calibration. Most applications do not require high precision but, rather, a flexible, easily-understood algorithm which delivers moderately good precision for a wide range of cameras and lenses. In particular, the algorithm should handle wide-angle lenses because these are useful in navigation, analysis of egomotion, and photography in confined spaces. (The human eye has a 180° horizontal field of view.) Finally, the method should not require specialized optical equipment (as in [12, 20]).

Previous image-based calibration algorithms fall into two classes. The *complex target* algorithms [11, 17, 18, 19] compute calibration from a target containing many features whose 3D configuration is precisely known. For example, Tsai [17] uses a set of corners on a planar target, translated using a precision stage. These algorithms are fast, but they require special precision targets and are "black boxes" which the typical user can neither understand nor modify.

The *moving camera* algorithms calibrate camera parameters by observing the motion of features as the camera is moved in a known direction. All existing algorithms in this class are seriously flawed. Many [1, 3, 5, 7, 8, 9] assume there is no radial distortion and, thus, cannot be used with wide-angle lenses. Some [1, 3, 4, 13] cannot calibrate extrinsic parameters: they are set by manual alignment, introducing unknown errors. Algorithms based on zooming [11] will not work on non-zoom lenses.

Finally, previous algorithms provide only sketchy information about radial distortion. Only Oh and Hall [13] gather dense data on the distortion function. Other methods calculate one [4, 19, 20], two [17], or three [11] parameters of a polynomial approximation. These algorithms have been tested only on low-distortion lenses: field of view less than 65° or precision fisheye lenses [6, 13, 19]. There is no reason to believe the wide variety of distortion functions for wide-angle lenses [15] (Figure 1) can be accurately modeled by low-order polynomials. For example, Figure 2 shows the systematic errors between two Tsai-type fifth-order (2-parameter) polynomials and measured data for our 2.6mm lens (116.5° field of view).

Previous algorithms cannot compute dense distortion data because they cannot choose where their features lie in the field of view. We propose a new moving camera algorithm which dynamically positions a feature within the field of view, choosing locations convenient for calibration. This allows us to make do with a single feature and decompose the calibration process into a series of simple adjustments to individual camera parameters. Both intrinsic and extrinsic parameters can be calibrated, including dense data on radial distortion and the camera's intensity response.

2 Method Overview

The standard perspective camera model is not convenient for representing wide-angle lenses, and not capable of representing lenses with a field of view ≥ 180°. Therefore, we use a more general two-stage imaging model [2]: perspective projection onto a sphere, followed by a radially symmetric projection onto the image plane.

The camera has five intrinsic parameters: the location of the *principal point* in the digitized image,

0-8186-7042-8/95 $4.00 © 1995 IEEE

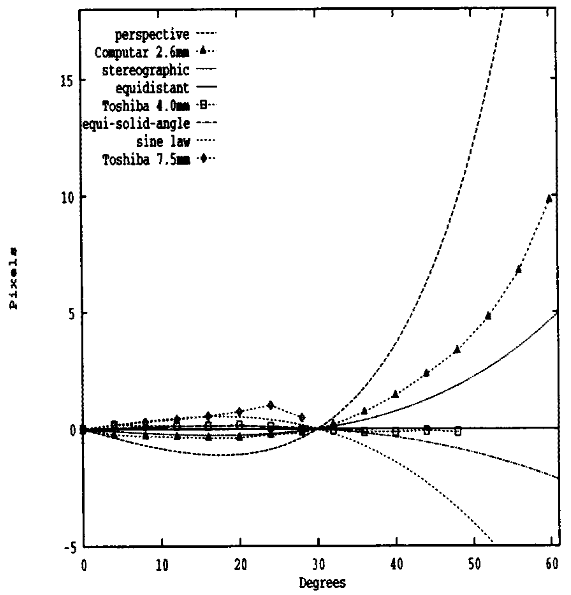

Figure 1: Radial projection functions for three lenses compared to five idealized projection models. Each was normalized to have the value 30 at 30 degrees and the line $y = x$ was subtracted.

the ***aspect ratio*** relating horizontal and vertical image distances, the ***baseline response*** at each image location when there is no incoming light, the ***radial projection function***, and the ***radial intensity drop-off function.***

The camera is mounted on a 6 degree of freedom positioning device (a Puma 562 robot arm). Six extrinsic parameters relate the robot's coordinate system to that of the camera. R_x, R_y and R_z are Euler angles giving the relative orientation of the two coordinate systems. The translation between the origins of the two coordinate systems is given by T_x, T_y and T_z. The z-axis of the camera coordinate system is the camera's optical axis.

We assume that a cooperative user has supplied approximate values for the extrinsic camera parameters and for the principal point. We expect errors of approximately ±2*cm* in the translational parameters and ±5° in the orientation of the camera axes, though our algorithm can correct larger errors (Section 5). We expect the principal point to lie within about 30 pixels of the center of the digitized image.

Throughout the calibration process, motion commands are issued in camera coordinates. A commanded camera motion f is translated into a motion g of the robot's wrist as follows:

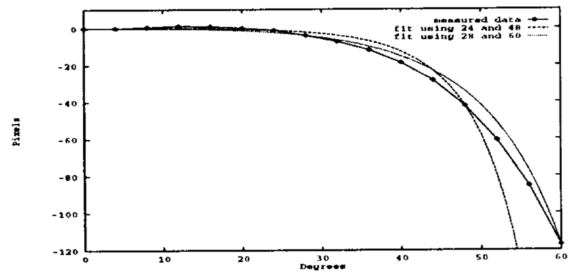

Figure 2: Tsai-type fifth-order polynomials poorly fit the radial projection function of our 2.6mm lens.

$$
\begin{aligned}
g &= M^{-1} \circ f \circ M \\
M &= T_{xyz} \circ R_z \circ R_y \circ R_x
\end{aligned}
$$

Our calibration algorithm has four steps:

0: Measure the zero baseline response of the camera.

A: Adjust the extrinsic parameters of the camera: R_x, R_y, R_z, T_x, T_y and T_z. This step assumes that the input principal point is correct.

B: Measure the remaining intrinsic parameters of the camera: aspect ratio, radial projection function, and intensity drop-off.

C: Examine the radial projection function and intensity drop-off for asymmetries. If the principal point is wrong, correct it and return to step A.

The zero baseline response of the camera is measured by taking an image in the dark with the lens cap on. Our full-size CCD cameras have a constant offset of 30-50 units (out of the usual $[0, 255]$ intensity range). The miniature cameras used for mounting on the robot arm have baseline responses that vary in an irregular way across the CCD array, with a noticeable slope. It is essential to calibrate the baseline response if you ever intend to (explicitly or implicitly) divide one intensity value by another.

The other parameters are calibrated by moving the camera relative to a single spherical light source fixed in space, which appears as a bright dot in the camera image. To calibrate the intrinsic parameters of a large camera, the roles of camera and light could be reversed: fix the camera in space and mount the light on the end of the robot arm.

3 Experimental setup

The algorithm was tested using two Toshiba 1K-M40A miniature color CCD cameras, one with a Toshiba JK-L75M 7.5mm lens and one with an Elmo 4mm lens (horizontal fields of view 49° and 92°). The camera heads were mounted on a second-hand, 6 degree of freedom PUMA 562 robot arm, using about

$5 worth of brackets from the local hardware store. Our calibration code, partly LISP and partly C, runs on an IBM RS/6000 320H, communicating with the robot controller via a serial port.

Images are acquired using a Datacube digicolor framegrabber, which returns 505 by 470 pixel images. Our calibration target is a frosted white spherical Christmas tree light 1.5cm in diameter, mounted on a black background. The dot is located in the image by finding the center of mass of all pixels whose intensity is above a set threshold.

The weak link in this system is the robot arm. Its positioning accuracy, though acceptable for ordinary robotics applications, creates small but visible errors in camera calibration (compare [9]). We are currently designing a hand-operated calibration mount for higher-precision calibration of intrinsic parameters. The camera should be calibrated on a mount at least as precise as the mounts used in applications.

4 Step A: Extrinsic Parameters

The extrinsic parameters are calibrated in the following order:

1: The direction of the camera z-axis: R_x and R_y

2: The direction of the camera x-axis: R_z

3: The location of the camera center along the x- and y-axes: T_x and T_y

4: The location of the camera center along the z-axis: T_z

Throughout this step, we assume that the input principal point is correct.

Because the lens may have considerable radial distortion, extrinsic parameters must be calibrated with only limited information about image geometry. We can use angles about the image center (as in [17]). Because radial projection functions are monotonic and symmetric about the principal point, we can also determine which of two points is further from the principal point (not used by [17]).

Controlled motion of the camera is used to reveal errors in each parameter. We command a motion which should have a particular effect on the position of the calibration dot. The difference between the actual motion of the dot and the ideal motion is measured by an *evaluation function*. Gradient descent is then used to find the parameter values which minimize the evaluation function.

Previous calibration algorithms use gradient descent to fit an analytic model to a set of pre-measured features. By contrast, the gradient descent in our algorithm makes a new measurement of the physical world whenever it needs to compute the evaluation function for new parameter settings. So the robot repeatedly executes almost identical motions (doing its "aerobics"), searching for the best parameter values. This allows us to make effective use of the monotonicity constraint on radial projection functions.

We cannot rotate the camera without knowing the location of the camera center. Therefore, steps A1 and A2 can use only translational motions. Conversely, since no translational motion can reveal errors in the location of the camera center, steps A3 and A4 must use motions involving rotation (and perhaps translation).

Step A1: R_x and R_y

1: Center the dot at the principal point.

2: Translate parallel to the estimated camera z-axis.

3: Find the dot pixel position: (x, y)

Evaluation Function: $x^2 + y^2$

Discussion: Translation parallel to the actual camera z-axis should leave the dot fixed at the principal point; translation in other directions moves it.

Step A2: R_z

1: Center the dot at the principal point.

2: Translate parallel to the estimated camera x-axis.

3: Find the dot pixel position: (x, y)

Evaluation Function: y^2

Discussion: Translation parallel to the actual camera x-axis should move the dot along the horizontal line through the principal point. The evaluation function measures the distance between the dot and this line.

Step A3: T_x and T_y

1: Center the dot at the principal point.

2: Rotate around the estimated camera z-axis.

3: Find the dot pixel position: (x, y)

Evaluation Function: $x^2 + y^2$

Discussion: Rotation about the actual camera z-axis should keep the dot at the principal point; rotation about other axes parallel to the z-axis moves the dot in a circle.

Step A4: T_z

1: Center the dot at the principal point.

2: Rotate $\theta°$ around the estimated camera x-axis.

3: Find the first dot pixel position: (x_1, y_1)

4: Rotate $-\theta°$ around the estimated camera x-axis.

5: Translate along the camera z-axis.

6: Rotate $\theta°$ around the estimated camera x-axis.

7: Find the second dot pixel position: (x_2, y_2)

Evaluation Function: $(y_1 - y_2)^2$

Discussion: The motion of the dot under rotation about the actual x-axis should not depend on the distance between the light and the camera. Therefore, when T_z is correct, y_1 and y_2 should be the same.

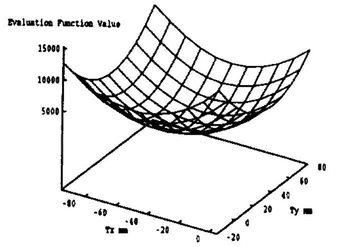

Figure 3: The measured error surface of T_x and T_y has a smooth slope and a single local minimum.

Table 1: Variation in Output Values

	Constant Start		Variable Start		
Parameter	#	σ	#	Range	σ
R_x	50	0.02°	81	±20°	0.06°
R_y	50	0.06°	81	±20°	0.06°
R_z	50	< 0.01°	41	±20°	0.03°
T_x	50	0.02mm	81	±100mm	2.32mm
T_y	50	0.15mm	81	±100mm	2.22mm
T_z	50	1.11mm	41	±100mm	3.23mm

The adjustment in Step A4 requires virtual features in particular positions relative to one another and relative to the camera. Our controlled motions allow us to simulate features at any desired 3D location. Because a fixed calibration target may not contain a suitable set of features, and because he did not use the monotonicity of radial projection functions, Tsai [17] was unable to compute T_z before the focal length and distortion parameters.

5 Accuracy of Extrinsic Parameters

The implementation of Step A uses a simplified version of the steepest descent algorithm. Because our estimated parameter values are typically close to the final estimates, this algorithm usually converges in less than 10 steps and requires less than 15 minutes to calibrate all extrinsic parameters. Our measured error surfaces have the smooth slope and single local minimum required for gradient descent, as illustrated in Figure 3.

To assess sensitivity to noise, we ran the calibration algorithm for each parameter repeatedly, both from a constant starting position and from a range of starting positions. Output values vary little (Table 1) even when the initial parameter estimates are as much as 20° or 10cm from the correct values.

6 Step B: Intrinsic Parameters

Step A has established a camera coordinate system based on to the input estimate of the principal point. We now use motions in this coordinate system to calibrate the intrinsic camera parameters: aspect ratio, radial projection function, and intensity drop-off.

Step B1: Aspect Ratio

1: Center the dot at the principal point.
2: Rotate θ° around the camera x-axis.
3: Find the first dot pixel position: $(x1, y1)$
4: Rotate $-\theta$° around the camera x-axis.
5: Rotate θ° around the camera y-axis.
6: Find the second dot pixel position: $(x2, y2)$

Horizontal image distances are then expanded by $(y1/x2)$.

Step B2: Radial Projection Function

We next calibrate the radial projection function, $p(\alpha)$, mapping angular positions in the field of view to pixel distances from the principal point.

1: Center the dot at the principal point.
2: Compute the angle ϕ required to move the dot into a corner of the image.
3: Rotate in direction ϕ by small angular steps (currently 1°). At each position, record the distance between the dot and the principal point, stopping when the dot reaches the corner of the image.

It is important to rotate into the corners of the image in order to measure $p(\alpha)$ for the full range of angles present in the image. Figure 1 shows the measured values for three lenses.

Previous calibration algorithms compute radial distortion, i.e. the difference between the measured radial projection function and ideal perspective projection $p(\alpha) = k\tan(\alpha)$ (where k is a scaling constant). However, this representation works poorly on wide-angle lenses. Because the $\cos^4$ intensity drop-off makes design of perspective lenses difficult for wide fields of view and impossible for fields of view above 140°, many wide-angle lenses are designed with "fisheye" projection functions: $k\sin(\alpha)$, $k(\alpha)$, and $k\sin(\frac{\alpha}{2})$ [10, 14] (Figure 1). Stereographic projection ($k\tan(\frac{\alpha}{2})$), which has convenient geometric properties, approximates the fisheye projections [2].

Direct representation of the radial projection function offers several advantages for wide-angle lenses. The projection function values are always relatively small, whereas the distortion between a fisheye lens and perspective projection becomes infinite as the field of view approaches 180°. The radial projection function can describe lenses with field of view $\geq 180°$ [10]. There is no need for a separate focal length parameter. Finally, a table lookup algorithm can be used to undistort a camera image into any of the ideal projection models.

Step B3: Intensity Drop-off

In an ideal perspective lens, the apparent brightness of an object decreases by approximately $\cos^4(\alpha)$, where α is the angular distance from the optical axis [10]. Because much of the drop-off is due to variation in how many pixels represent each solid angle, this effect is greatly reduced or eliminated in fisheye lenses [10]. Our algorithm measures the effect as follows:

1: Center the dot at the principal point.

2: Compute the angle ϕ required to move the dot into a corner of the image.

3: Rotate in direction ϕ by small angular steps (currently 1°). At each position, record the integrated intensity of the dot, stopping when the dot reaches the corner of the image.

The integrated intensity is the sum of the intensity values at all pixels contained in the dot, minus the corresponding values in zero baseline image. Images can be corrected to remove intensity drop-off using these integrated intensities, together with the changes in the dot's size (which can be computed from the radial projection function).

7 Step C: Look for Asymmetries

The extrinsic and intrinsic camera parameters were calibrated using an estimate of the principal point which may or may not have been accurate. There could also be tangential distortion, the lens might be tilted relative to the image plane, the components within the lens might not be aligned with one another, and/or the radial distortion might not be symmetrical about the optical axis. All of these defects, if present, create asymmetries in the radial projection function and/or the intensity drop-off. Following previous authors, we assume that most or all observed errors are due to a displaced principal point.

No calibration algorithm can detect errors in these alignment parameters unless they induce asymmetries which are larger than those created by robot positioning errors, uneven CCD response, and the like. To detect asymmetries, we repeat calibration steps B2 and B3, rotating the camera in eight directions (left, right, up, down, and four diagonals). Ideally, data from all eight directions should be identical: any differences indicate errors in alignment parameters.

In some lenses (e.g. [20]), the drop-off in intensity response towards the edges of the image is large enough that it could be used to detect a mis-placed principal point. Because our lenses are closer to fisheye projection than perspective, intensities vary little across our images. The function is so flat that it would be difficult to locate the peak accurately even with high-precision measurement of intensity values.

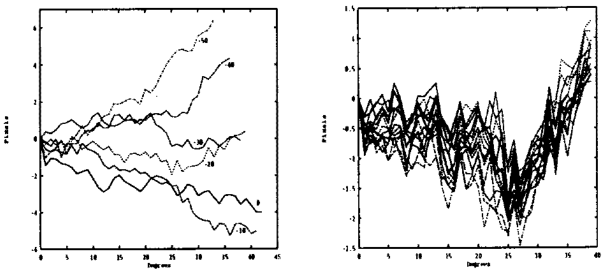

Figure 4: Left: Differences between the left and right radial projection functions, for various horizontal displacements of the principal point from the center of the digitized image. A displacement of −20 pixels seems to minimize the differences, although a displacement of −30 is similar in quality. Right: Repeated measurements (for displacement −20) of these differences vary by up to 2 pixels.

Asymmetry can also be measured by comparing values of the radial projection function on opposite sides of the alleged principal point. Figure 4 shows the differences between left and right values for various horizontal displacements of the principal point. Because the radial projection function has a simple shape, the sign of these differences indicates the direction in which the principal point is displaced. The magnitude of the displacement could be estimated by dividing the left-right differences by the slope of the radial projection function (cf. optical flow).

The variation between the lines in Figure 4 (left) is ≤ 6 pixels even when the principal point is displaced 20 pixels. This is consistent with the theoretical analysis in [17]: the appearance of the image is not very sensitive to changes in the principal point. The principal point can probably be localized to within ±5 to 10 pixels, by integrating the radial projection function differences from Figure 4 across the entire field of view. This should be sufficient precision for most practical applications.

Notice that the principal point cannot be found by translating along an estimate of the camera axis and detecting the focus of expansion as proposed in [6] (cf. also [1, 3, 4]). The focus of expansion is determined by which axis you translate along: pick a (slightly) different axis direction and you get a (substantially) different focus of expansion and thus a different hypothesis for the principal point. The focus of expansion algorithm merely makes the estimated axis and the estimated principal point consistent with one another (cf. step A1 of our algorithm). It has the insidious property that if you choose the axis estimate in a consistent way (e.g. parallel to the camera body) it will measure an image position with high repeatability but that position is not necessarily the principal point.

While we are obtaining radial distortion data in a given direction, we monitor the dot's angular position about the principal point. Variations in this angle indicate tangential camera distortion and/or inaccuracy in the commanded camera motions. For our mount, this typically indicates that the robot arm is approaching a singularity.

8 Conclusions

By using the full power of controlled motion along with a single light source we can simulate calibration features at any desired 3D location. This allowed us to design a camera calibration algorithm which is very general (usable even on wide-angle lenses) and easy to understand. These simulated calibration features also allow us to densely measure intensity drop-off and measure the radial projection function more accurately the traditional low-order polynomial models.

The algorithm can be adapted (by a non-expert) for use on a variety of controllable mounts. Its precision is limited primarily by the precision of the mount. On a high-precision mount (e.g. a set of optical stages), it could produce very high-precision measurements of intrinsic camera parameters. Even on a robot arm whose positioning is definitely suspect, it produces results precise enough for most vision applications. See [16] for additional details.

Acknowledgments

We would like to thank Terry Boult, Michael Covington, Brian Madden, Howard Moraff, Steve Osborne, and Mike Wall.

References

[1] Basu, Anup (1993) "Active Calibration: Alternative Strategy and Analysis," *IEEE CVPR* 1993, 495–500.

[2] Margaret M. Fleck (1995) "Perspective Projection: the Wrong Imaging Model," TR 95-01, Comp. Sci., U. Iowa,

[3] Dron, Lisa (1993) "Dynamic Camera Self-Calibration from Controlled Motion Sequences," *IEEE CVPR* 1993, 501–506.

[4] Du, Fenglei and Michael Brady (1993) "Self-Calibration of the Intrinsic Parameters of Cameras for Active Vision Systems," *IEEE CVPR* 1993, 477–482.

[5] Faugeras, Olivier (1992) "Camera Self-Calibration: Theory and Experiment," *European Conf. Comp. Vision* 1992, 321–334.

[6] Fiala, John C., Ronald Lumia, Karen J. Roberts, and Albert J. Wavering (1994) "TRICLOPS: A Tool for Studying Active Vision," *Intern. J. of Comp. Vision* 12/2-3, 231–250.

[7] Hartley, Richard I. (1992) "Estimation of Relative Camera Positions for Uncalibrated Cameras," *European Conf. Comp. Vision* 1992, 579–587.

[8] Hartley, Richard I. (1994) "Self-Calibration from Multiple Views with a Rotating Camera," *European Conf. Comp. Vision* 1994, vol. I., 471–478.

[9] Horaud, R., F. Dornaika, B. Boufama, and R. Mohr. (1994) "Self Calibration of a Stereo Head Mounted onto a Robot Arm," *European Conf. Comp. Vision* 1994, vol. I., 455–462.

[10] Kingslake, Rudolf, *A History of the Photographic Lens,* Academic Press, San Diego, 1989.

[11] Li, Mengxiang (1994) "Camera Calibration of a Head-Eye System for Active Vision," *European Conf. Comp. Vision* 1994, 543–554.

[12] Lenz, Reimar K. and Roger Y. Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology," *IEEE PAMI* 10/5 (1988) 713–720.

[13] Oh, Sung Jun and Ernest L. Hall (1989) "Calibration of an Omnidirectional Vision Navigation System Using an Industrial Robot," *Optical Engineering* 28/9, 955–962.

[14] Ray, Sidney F. (1994) *Applied Photographic Optics,* second edition, Focal Press, Oxford.

[15] Smith, Warren J. (1992) *Modern Lens Design: A Resource Manual,* McGraw-Hill, New York.

[16] Stevenson, Daniel E. and Margaret M. Fleck (1994) "Robot Aerobics: Four Easy Steps to a More Flexible Calibration," TR 94-09, Comp. Sci., U. of Iowa.

[17] Tsai, Roger Y., "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision," *IEEE CVPR* 1986, 364–374.

[18] Wei, Guo-Qing and Song De Ma (1994) "Implicit and Explicit Camera Calibration: Theory and Experiments," *IEEE PAMI* 16/5, 469–480.

[19] Weng, Juyang, P. Cohen, and M. Herniou (1992) "Camera Calibration with Distortion Models and Accuracy Evaluation," *IEEE PAMI* 14/10, 965–980.

[20] Willson, Reg G. (1994) "Modeling and Calibration of Automated Zoom Lenses," Ph.D. thesis, ECE, Carnegie-Mellon, CMU-RI-TRI-94-03.