Bowling for Calibration: An Undemanding Camera Calibration Procedure Using a Sphere

Pietro Cerri, Oscar Gerelli, and Dario Lodi Rizzini

Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Parma
{cerri,gerelli,rizzini}@ce.unipr.it

Abstract. Camera calibration is a critical problem in computer vision. This paper presents a new method for extrinsic parameters computation: images of a ball rolling on a flat plane in front of the camera are used to compute roll and pitch angles. The calibration is achieved by an iterative Inverse Perspective Mapping (IPM) process that uses an estimation on ball gradient invariant as a stop condition. The method is quick and as easy to use as throw a ball and is particularly suited to be used to quickly calibrate vision systems in unfriendly environments where a grid is not available. The algorithm correctness is demonstrated and its accuracy is computed using both computer generated and real images.

1 Introduction

The recovery of 3D information from images is one of the main problems in computer vision, which needs the precise knowledge of both intrinsic and extrinsic parameters. While intrinsic parameters can be computed once for a given camera (with constant focal length), extrinsic parameters need a new measurement every time the camera is moved. For a camera mounted on a vehicle, the main problem is the computation of camera orientation whose drifts are mainly due to vibrations. Calibration is a critical and hard process which generally needs a large infrastructure such as a grid with markers at known positions. For example in the DARPA Grand Challenge project [1] (in which our team built a self-guided truck that completed the whole course using vision as primary sensor) we had the need to often recalibrate the cameras in the Mojave desert (Nevada). In such extreme environments a quick and easy calibration is mandatory. The use of a ball to calibrate is based on the assumption that a sphere has the same shape when framed from different points of view. The bowling ball is chosen because no deformations are possible, but any other ball can be used, assuming that deformations will be ignored. A number of images with the ball in different positions must be acquired for the calibration. These images are obtained throwing a ball on a flat plane, avoiding bounces. Intrinsic parameters of the camera and its position together with ball size must be known to apply this method. This kind of method can be used to compute pitch and roll angles; it was chosen not to use any other world reference, so it is impossible to estimate the yaw angle. The aim is to obtain a real time calibration system: this version of the algorithm

is not working in real time yet but its development was studied in such a way that the transition to real time will be straightforward.

Methods for recovering intrinsic and extrinsic camera parameters are usually classified into two categories.

The first category includes *self-calibration* methods, which achieve calibration using multiple views of a static scene from a moving camera, eg. those described in [2] and [3]. The second approach exploits a calibration object. These methods can be further classified according to the information about the calibration object. Some methods assume to know the 3D position of a sufficient number of points (or equivalently the position of a known shape). To this category belongs the approach proposed by Tsai in [4], the DLT method firstly introduced by Abdel-Aziz et al. in [5] and the algorithm described in [6]. Other methods need only generic 3D geometry information, eg. coplanar points, as shown in [7]; a third category requires only the shape and the size of the calibration object. The proposed algorithm belongs to the last category and uses a ball as the calibration object, as also shown in [8]: the work by Agrawal needs a perfect contour recognition of a sphere filling a large part of image; this constraint is not required by the work described in this paper.

The next section presents the algorithm details, section 3 shows some results obtained on both synthetic and real images, and finally in section 4 conclusions and future developments are presented.

2 Algorithm

In this section a ball geometric invariant is deduced and applied in the calibration process to iteratively estimate pitch and roll angles of a camera orientation.

Considering the rotation matrix with respect to pitch (θ) and roll (ρ), with no care about yaw (γ) angle

$$R = \begin{bmatrix} 0 & -c\rho & -s\rho \\ -s\theta & s\rho c\theta & -c\rho c\theta \\ c\theta & s\rho s\theta & -c\rho s\theta \end{bmatrix} \qquad (1)$$

where $c\rho$, $c\theta$, $s\rho$ and $s\theta$ stands for $\cos\rho$, $\cos\theta$, $\sin\rho$ and $\sin\theta$ respectively, and the translation vector $T = [0\ 0\ h]^T$, where h is the camera height from the ground, a mapping from 3D to 2D is performed.

By applying the standard mathematical pin-hole model and an inverse perspective mapping (IPM) it is possible to establish a bijective mapping between ground plane points and image pixels.

The calibration process produces $\widetilde{R}$. IPM points (2D) are related to the 3D points by the following equation:
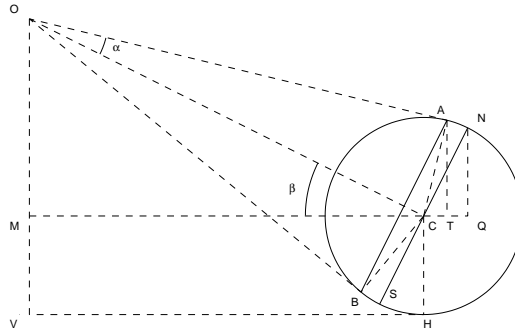
$$k \cdot \widetilde{R}^T \cdot R \begin{bmatrix} X \\ Y \\ Z - h \end{bmatrix} = \begin{bmatrix} X_{ipm} \\ Y_{ipm} \\ 0 - h \end{bmatrix} \qquad (2)$$

When $\widetilde{R} = R$ then a perfect calibration is achieved and $\widetilde{R}^T \cdot R$ is an identity matrix; hence, we have

$$X_{ipm} = \frac{h \cdot X}{h - Z} \tag{3}$$

$$Y_{ipm} = \frac{h \cdot Y}{h - Z} \tag{4}$$

Since camera intrinsic parameters are supposed to be known they are omitted in equation 2.

Fig. 1. Ball section view from an observer O

Considering spherical objects, just like the one depicted in figure 1, the following notations are assumed

$$C = [x \ y \ r]^T \tag{5}$$

$$d = \sqrt{x^2 + y^2} = \overline{HV} \tag{6}$$

$$D = \sqrt{d^2 + (h - r)^2} = \overline{OC} \tag{7}$$

where r is the ball radius.

Points A and B in figure 1 represent the points of interest: line $\overline{AB}$ delimits the ball area visible by an observer O. In order to simplify mathematical relationships handled in the following steps of the algorithm, line $\overline{AB}$ is approximated by the diameter of the observed spherical object, that is with line $\overline{NS}$. In appendix A the correct expression of $\overline{AB}$ and the error caused by the approximation are estimated. A proportion between triangle $O\overset{\Delta}{C}M$ and $C\overset{\Delta}{N}Q$ together with the symmetry of N and S with respect to the ball center (C) allow to express segments $\overline{CQ}$ and $\overline{QN}$ as

$$\overline{CQ} = \frac{(h - r) \cdot r}{D} = \frac{r(h - r)}{\sqrt{x^2 + y^2 + (h - r)^2}} \tag{8}$$

$$\overline{NQ} = \frac{d \cdot r}{D} = \frac{r\sqrt{x^2 + y^2}}{\sqrt{x^2 + y^2 + (h - r)^2}} \tag{9}$$

When the road plane matches the assumption of a flat area, the IPM transformation recovers the texture of the ground plane. When objects are present on the ground plane they appear stretched in the IPM image. The spherical shape of the ball when remapped via the IPM transformation is reshaped into a pseudo-ellipse as shown in figure 2.

(a)   (b)
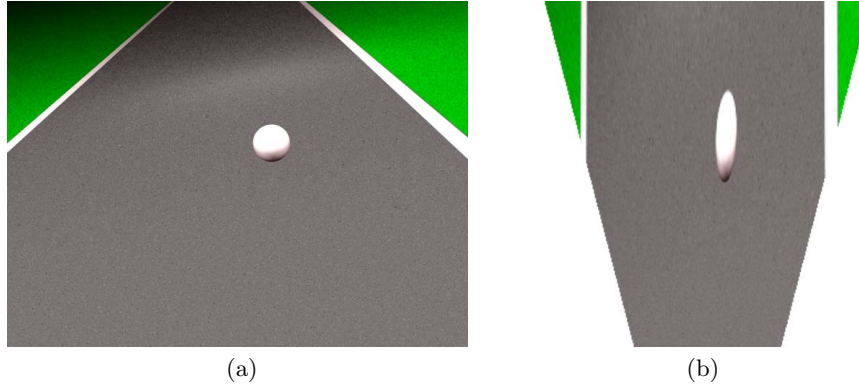
Fig. 2. IPM reshaping: (a) original image of a ball in 3D world (b) the ball after removal of the perspective effect.

Now, applying equations 3 and 4, segments $\overline{CQ}$ and $\overline{QN}$ are used to compute the major axis of the pseudo-ellipse resulting from the IPM transformation.

$$\overline{NS} = \frac{2\ r\ h\ D^3}{[(h-r)^2 - r^2]D^2 + r^2\ (h-r)^2} \tag{10}$$
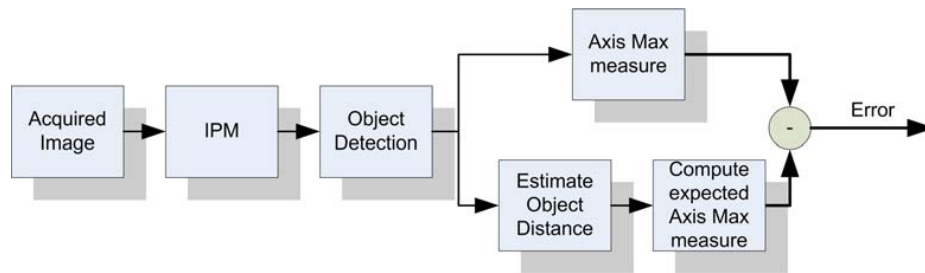
Fig. 3. Estimation error computing

It can be demonstrated that in perfect calibration scenarios $\overline{NS}$ is a geometric invariant with respect to θ and ρ.

Equation 10 is not enough to compute the calibration parameters since it was deduced under the assumption of knowing the correct ρ and θ angles and distance D. Anyway it could be used to check the achievement of the correct calibration condition. In fact, as depicted in the block diagram of figure 3, a formal calibration check can be obtained by comparing the length of the major axis of the ellipse with the value given by equation 10. Distance evaluation is performed by using the coordinates of the ball center in the IPM image into equations 3-4, with $Z = r$, while ball detection may be obtained using the method described in [9] and approximating the ball shape in the IPM image with an ellipse.

A critical phase of the algorithm is the update of estimated calibration parameters $\widetilde{\theta}$ and $\widetilde{\rho}$, used in the IPM process. This is done using a steepest-descent algorithm along quadratic error gradient.

The isolines of the error function (see figure 4) are approximatively parallel lines even in the neighborhood of *calibration point* corresponding to the correct pitch and roll; so if it is simple to identify the line intersecting the calibration point (identified as *calibration line* in figure 4) by a descent gradient algorithm, it is more complex to move along this line to identify the calibration point, then the convergence toward the global minimum of the error function is hard. Matrices in equation 2, under a first order approximation, become

$$\widetilde{R}^T R \simeq \begin{bmatrix} 1 & 0 & \Delta\theta \\ 0 & 1 & -\Delta\rho \\ -\Delta\theta & \Delta\rho & 1 \end{bmatrix} \tag{11}$$

where $\Delta\theta = \widetilde{\theta} - \theta$ and $\Delta\rho = \widetilde{\rho} - \rho$.

Isolines are identified by constant $x_{ipm}^2 + y_{ipm}^2 = d_{const}^2$; computing x_{ipm} and y_{ipm} by the equation 2 using the approximation 11, and excluding second order terms, the isoline equation becomes equal to

$$2x_0 z_0 H \Delta\theta = 2y_0 z_0 H \Delta\rho - (h^2(x_0^2 + y_0^2) - d^2 z_0^2) \tag{12}$$

where $H = h^2 + d^2$ and $(x_0, y_0, z_0)^T$ is the real world position of the ball center. Notice that equation 12 represents a line in ρ-θ plane with slope y_0/x_0. Following the orthogonal vector $(y_0, -x_0)^T$ the line passing through the calibration point can be identified.

By iteratively executing the steps described in algorithm 3, using images of ball in different positions, many lines with different slope are found. All lines intersect in a single point in ρ-θ plane: this is the calibration point. In order to find the intersection point the ρ-θ plane is partitioned in accumulator cells; the calibration point is located in the cell accumulating the maximum value. An outline of the algorithm is shown below:

3   Results

The algorithm has been tested both with synthetic and real images (see figure 5).
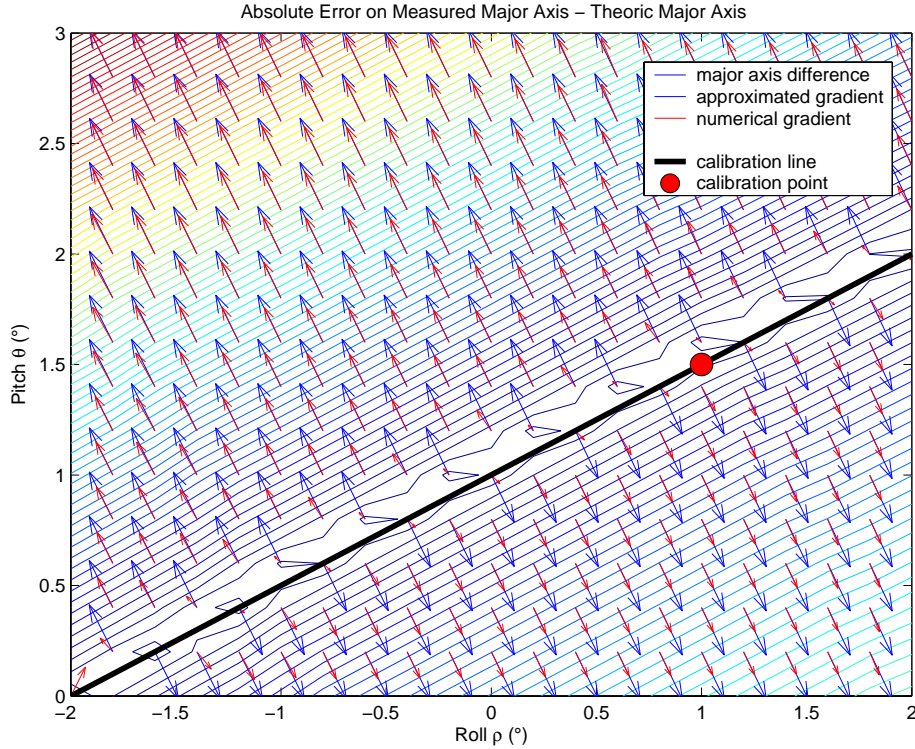
Fig. 4. Isolines of calibration error with gradients. The calibration point is indicated with a red circle.


Algorithm 1 Calibration algorithm

Require: $\widetilde{\rho}_0$, $\widetilde{\theta}_0$, ball image
 1: compute error and estimate ball position (x_c, y_c) like in figure 3, using $\widetilde{\rho}_k$ and $\widetilde{\theta}_k$;
 2: update $(\widetilde{\rho}_{k+1}, \widetilde{\theta}_{k+1})$ incrementing of a step $\eta \times error$ in direction of $(-y_c, x_c) \cdot sign(error)$;
 3: stop when error sign changes and trace line on cell accumulator plane; increment k otherwise.

The mathematical correctness was tested with synthetic images. In this scenario a set of rolling ball images are created assuming to use a camera with known intrinsic and extrinsic parameters; then the images are processed by the algorithm and a comparison between output results and expected parameters values is done. Statistical data reported in figure 6 confirm that the underlying idea is correct and that the algorithm can achieve a fine grade of precision: due to the algorithm structure the maximum error is the estimation of the ρ angle. Mean error on θ prediction is about $0.57°$ while on ρ is about $1.26°$ using square accumulator cells of $0.5°$ per side.
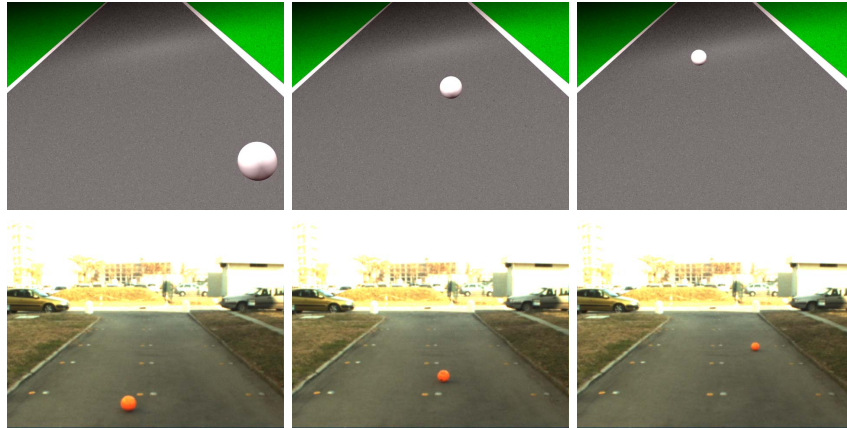
Fig. 5. Top row: images of a ball on a flat surface; Bottom row: images of a real sequence acquired on the usual calibration grid that was used as a reference to assess the precision of this method.
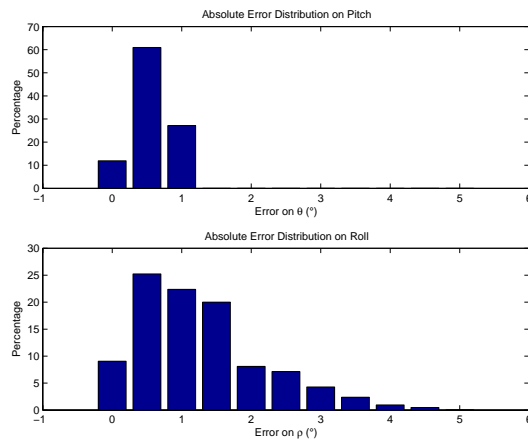

Fig. 6. Statistic of calibration error with artificial images

Tests on sequences of real images acquired by a monocular vision system show high sensitivity of calibration variables prediction on the accuracy of experimental setup parameters. As reported in figure 7 a small error in measuring camera height from the ground plane can determine a significant error, especially on real image analysis.

The analysis of the error computed on real images determines a value of h for which the error is zero. This value of h represents the correct height estimation. This shows that an error of about 2% in measuring an algorithm input variable can cause significantly different prediction error.
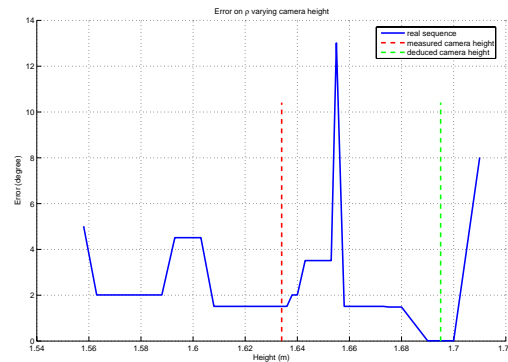
Fig. 7. Error on roll prediction at different camera height

4 Conclusions

The proposed algorithm has been proved to work correctly: its precision (computed on synthetic images) is about 0.5° for the pitch angle and 1.2° for the roll one as shown in section 3. It is experimentally demonstrated that 10 images are enough to obtain a satisfactory calibration in standard conditions; a greater number of images may not provide more precise results. The required number of images depends anyway on the ball position, and therefore on the slope of calibration lines obtained in different processing.

Unfortunately the result is sensibly influenced by errors in measurement of input parameters that are considered as known (e.g. camera height, ball sizes, camera intrinsic parameters).

Moreover uncorrect ball shape detection may cause low precision results too: in unstructured environments, such as when no clear contrast between the calibration object and the background is present, precision may be reduced.

Other weaknesses can be attributed to the stop condition of the iterative process: due to the presence of local minima the algorithm could identify a wrong calibration line. Moreover the assumption that the correct calibration point stands in the accumulator cell with maximum value stored is not always true: other heuristics, that use accumulator cells or other methods, are required to overcome limitations on the computation of the lines intersection.

The current implementation is not real time: for each processed image the algorithm needs to compute a number of IPM transformations, that are computationally slow; the choice of the images to be used is currently made by the user and the user can also help the ball detection by identifying at least three points on the ball edge.

This method can be very easily extended to stereo vision system: once the pitch and roll angles are computed for both cameras, the relative yaw between the two cameras can be computed using only two pairs of synchronized images.

A Appendix: Approximation of Pseudo-ellipse Major axis

The *inverse perspective mapping* of line $\overline{NS}$ provides a good approximation of the real major axis $\overline{AB}$ (see figure 1). In this section the exact formula of $\overline{AB}$ is deduced and compared with our approximation. The position of end points A and B with respect to the ball center is easily achieved, when the measure of segments $\overline{CT}$ and $\overline{AT}$ is known (T is the projection of A on line MC). The value of angle $C\hat{A}T$ is equal to the difference of angles β and α (figure 1):

$$\alpha = \arcsin \frac{r}{D}, \quad \beta = \arcsin \frac{h-r}{D} \tag{13}$$

Thus the relative position of point A (and its symmetric B) is achieved by projecting radius $\overline{AC}$ on segments
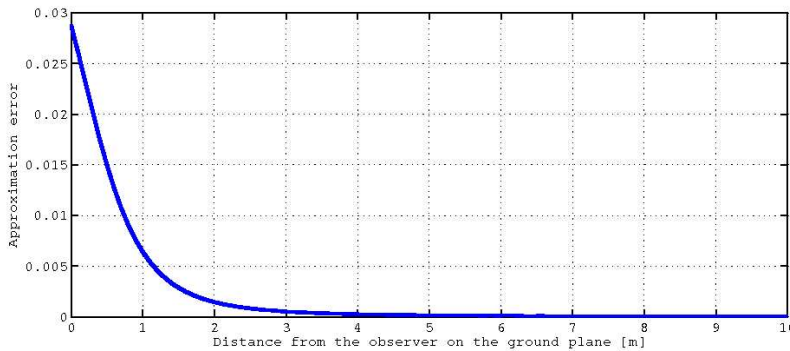
$$\overline{CT} = \frac{r\ \left(\sqrt{D^2 - r^2}\ (h-r) - d\ r\right)}{D^2} \tag{14}$$

$$\overline{AT} = \frac{r\ \left(r\ (h-r) + d\ \sqrt{D^2 - r^2}\right)}{D^2} \tag{15}$$

Under the assumption of flat area, IPM transformation given by equations 3 and 4 maps points A and B expressed in 3D world coordinates into pseudo-ellipse extreme points. Finally the resulting length of major axis of pseudo-ellipse is computed:

$$\overline{AB} = \frac{2\ h\ r\ D^4\ v\ \left(t^2 - d^2\right)}{t^2\ \left(D^4 - r^4\right) - d^2\ r^2\ v^2 - 2\ r^3\ t\ d\ v} \tag{16}$$

where $t = (h - r)$ and $v = \sqrt{D^2 - r^2}$.

Fig. 8. Relative error due to the approximation of pseudo-ellipse axis $\overline{AB}$ with $\overline{NS}$ with respect to the distance camera pinhole-ball center d (camera height $h = 1.5\ m$, ball radius $r = 0.3\ m$).

Note that the equation 16 is more complex and less suitable for algebraic manipulation than the approximation 10 used in this paper. Figure 8 shows a comparison of the two estimations of pseudo-ellipse axis with respect to the ground distance of the ball center from the camera pinhole. The approximation error grows as the distance D between the camera and the ball decrease; however, even in the worst case, the error committed using $\overline{NS}$ instead of $\overline{AB}$ is negligible: the maximum relative error is equal to 3 %.

References

1. DARPA: http://www.grandchallenge.org/ (2005)
2. Hartley, R.I.: An algorithm for self calibration from several views. Proc. IEEE Conference on Computer Vision and Patter Recognition **1**(7) (June 1994) 908–912
3. Hartley, R.I.: Euclidean Reconstruction from Uncalibrated Views. In: Applications of Invariance in Computer Vision. (1993) 237–256
4. Tsai, R.Y.: 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 **RA-3** (aug 1987) 323–344
5. Abdel-Aziz, Y., Karara, H.: Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. Proceedings of the Symposium on Close-Range Photogrammetry (1971) 1–18
6. Bouguet, J., Perona, P.: Camera Calibration from Points and Lines in Dual-Space Geometry. Proc. 5th European Conf. on Computer Vision 2–6
7. Zhang, Z.: A Flexible New Technique for Camera Calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 1330–1334
8. Agrawal, M., Davis, L.S.: Complete camera calibration using spheres: A dual-space approach. In: Procs. IEEE Intl. Conf. on Robotics and Automation. Volume 2., Albuquerque,NM (oct 2003) 782–789
9. Fitzgibbon, A., Pilu, M., Fisher, R.B.: Direct Least Square Fitting of Ellipses. IEEE Transactions on Pattern Analysis and Machine Intelligence **21** (may 1999) 476–480