Automatic Radial Distortion Estimation from a Single Image

Faisal Bukhari · Matthew N. Dailey

Abstract Many computer vision algorithms rely on the assumptions of the pinhole camera model, but lens distortion with off-the-shelf cameras is usually significant enough to violate this assumption. Many methods for radial distortion estimation have been proposed, but they all have limitations. Robust automatic radial distortion estimation from a single natural image would be extremely useful for many applications, particularly those in human-made environments containing abundant lines. For example, it could be used in place of an extensive calibration procedure to get a mobile robot or quadrotor experiment up and running quickly in an indoor environment. We propose a new method for automatic radial distortion estimation based on the plumb-line approach. The method works from a single image and does not require a special calibration pattern. It is based on Fitzgibbon's division model, robust estimation of circular arcs, and robust estimation of distortion parameters. We perform an extensive empirical study of the method on synthetic images. We include a comparative statistical analysis of how different circle fitting methods contribute to accurate distortion parameter estimation. We finally provide qualitative results on a wide variety of challenging real images. The experiments demonstrate the method's ability to accurately identify distortion parameters and remove distortion from images.

Faisal Bukhari · Matthew N. Dailey
The authors are with the Computer Science and Information Management program, School of Engineering and Technology, Asian Institute of Technology, P.O. Box 4, Klong Luang, Pathumthani 12120, Thailand
Fax: +66 2 524 5721
E-mail: syed.faisal.bukhari@ait.ac.th, mdailey@ait.ac.th

1 Introduction

Most computer vision algorithms, particularly structure from motion algorithms, rely critically on the assumption of a linear pinhole camera model. However, most commercially available cameras introduce sufficiently severe optical distortion that the pinhole assumption is invalid, making distortion correction a must. Radial distortion is the most significant type of distortion in today's cameras [28,43,54]. It is most evident in images produced with low-cost, wide-angle lenses [23]. Such lenses are being widely deployed, for example, in automotive driver assistance applications [22,26]. But radial distortion is also significant enough in higher-quality cameras to introduce error into 3D reconstruction processes. Radial distortion bends straight lines into circular arcs [43,50], violating the main invariance preserved in the pinhole camera model, that straight lines in the world map to straight lines in the image plane [17,25]. Radial distortion may appear as barrel distortion, usually arising at short focal lengths, or pincushion distortion, usually arising at longer focal lengths. Besides radial distortion, another type of distortion is tangential distortion. We do not have experience with real cameras that introduce significant tangential distortion, so like most previous work [2,28,38,43,47,54], we ignore tangential distortion.

Methods for radial distortion estimation fall into three major categories: point correspondence [7,49,54],

multiple view autocalibration [4,21,24,36,42], and plumb-line [2,8,9,17,43,44,47,50].

Point correspondence based methods [7,49,54] are ideal for distortion estimation during pre-calibration of a camera with a fixed focal length. They identify image points with known 3D positions in multiple images using a known pattern such as a chessboard and then estimate the parameters of an undistortion function. The parameterized undistortion function can then be used to undistort specific images or point positions. These point correspondence methods are highly reliable and accurate; radial distortion estimation and removal is a solved problem for cameras that are pre-calibrated at a fixed focal length.

Manual camera calibration, however, is a tedious process that is not always possible, for example, when we want to process an existing image sequence acquired with an unknown camera, when we want to change the focal length dynamically during image sequence acquisition, or when we want to get a mobile robot experiment up and running quickly.

Multiple view auto-calibration is an active area of computer vision research that aims to extract camera parameters automatically from natural images. Auto-calibration methods use a sequence of arbitrary natural images without any special pattern or information about the scene. Although many auto-calibration methods assume a pinhole camera, others do attempt to simultaneously estimate radial distortion parameters and pinhole parameters [21,24,28,36,42].

Auto-calibration is a mature area of research, but the main limitation of this class of methods is that it requires multiple images under camera motion. For fixed cameras and for situations where immediate online estimation is desirable, multiple view methods are inappropriate.

In view of the limitations of the point correspondence and auto-calibration methods, robust automatic distortion estimation and and removal from a single natural image would be extremely useful for many applications, particular those in human-made environments containing abundant lines. For example, it could be used in place of an extensive calibration procedure to get a mobile robot or quadrotor experiment up and running quickly in an indoor environment. Plumb-line methods are the most promising for robust distortion estimation from a single image or a small number of images. Rather than using a known pattern or sequence of images under camera motion, they estimate distortion parameters directly from distorted straight lines in one or more images. Straight lines are frequent enough in most human-made environments to make distortion estimation from a single image possible [43,47,50].

The main limitations of this class of methods are that straight lines must be visible in the image and that images of actual curved lines may disrupt estimation. Some methods address these issues simply by utilizing human supervision to select the lines (see, e.g., [2,9,44]). But when human supervision is not used, plumb-line methods depend critically on the robustness and accuracy of the line detection algorithms. Some plumb-line approaches do not use all available lines for distortion estimation despite the fact that additional lines could minimize estimation error [43,47,50], or assume the distortion center as the center of the image [2,8,21,28,43], which is in contrast to some researchers' recommendations [24,46]. The Devernay and Faugeras [17] method is the only existing method that overcomes all of these limitations. However, it requires a complex process of polygonal approximation of the distorted lines. As we shall see, the distorted line detection process can be dramatically simplified by using an alternative distortion model.

We propose a new method based on the plumb-line approach that addresses the aforementioned limitations. The method works from a single image if the image contains a sufficient number of distorted straight lines. It does not require a calibration pattern or human intervention. We use Fitzgibbon's division model of radial distortion [21] with a single parameter. Our estimator is similar to that of Strand and Hayman [43] and Wang et al. [50] in that we estimate the parameters of the distortion model from the parameters of circular arcs identified in the distorted image, based on the fact that distorted straight lines can be modeled as circular under the division model [4,43,50]. Our contribution is to make the process fully automatic and robust to outliers using a two-step random sampling process. For the first step, we introduce a sampling algorithm to search the input image for subsequences of contour pixels that can be modeled as circular arcs. For the second step, we introduce a sampling algorithm that finds the distortion parameters consistent with the largest number of arcs. Based on these parameters, we undistort the input image. Some preliminary results from our method have previously appeared in a conference paper [11].

In this paper, to evaluate the new algorithm, we perform a comprehensive quantitative study of its performance on distorted synthetic images and provide extensive examples of its ability to remove distortion from a large, challenging set of real images taken from Web sites and previous papers on distortion estimation. We find that the algorithm performs very well, with excellent reconstruction of the original synthetic images even under severe barrel distortion and pincushion distortion. We also find that the method is able to

eliminate nearly all of the visible distortion in the real images, including those acquired with wide angle and fish-eye lenses. Finally, we perform a direct comparison of our method with that of Álvarez et al. [2], the only researchers who have provided a publicly accessible implementation of their method, on synthetic images. The Álvarez et al. method exploits user assistance in identifying points on straight lines, but we nevertheless find that our fully automatic method provides superior reconstruction of the original undistorted image. Our method is thus a practical solution to the important problem of radial distortion estimation and removal.

2 Mathematical Model

In this section, we outline the mathematical model of radial distortion assumed in the rest of the paper and show how to estimate the parameters of this model.

2.1 Distortion model

The most commonly used radial distortion model is the *even-order polynomial model*

$$x_u = x_d(1 + \lambda_1 r_d^2 + \lambda_2 r_d^4 + \lambda_3 r_d^6 + \cdots)$$
$$y_u = y_d(1 + \lambda_1 r_d^2 + \lambda_2 r_d^4 + \lambda_3 r_d^6 + \cdots), \tag{1}$$

where (x_u, y_u) and (x_d, y_d) are the corresponding coordinates of an undistorted point and a distorted point, respectively. r_d is the Euclidean distance of the distorted point to the distortion center. If the distortion center is the origin of the distorted image, we can simply write

$$r_d^2 = x_d^2 + y_d^2. \tag{2}$$

However, if (x_0, y_0) is the center of distortion (in the distorted image), we write

$$r_d^2 = (x_d - x_0)^2 + (y_d - y_0)^2 \tag{3}$$

and replace x_d and y_d in Equation (1) with $(x_d - x_0)$ and $(y_d - y_0)$, respectively. In the model, $x_0, y_0, \lambda_1, \lambda_2, \lambda_3, \cdots$ are the *distortion parameters*, which must be estimated from image measurements.

There have been objections to the even-order polynomial model. According to Wang et al. [50], the model performs well for small distortion, but for severe distortion, a prohibitively large number of non-zero distortion parameters are required.

Fitzgibbon [21] proposes an alternative model, the *division model*, as a more accurate approximation to

the typical camera's true undistortion function:

$$x_u = \frac{x_d}{1 + \lambda_1 r_d^2 + \lambda_2 r_d^4 + \cdots}$$
$$y_u = \frac{y_d}{1 + \lambda_1 r_d^2 + \lambda_2 r_d^4 + \cdots}.$$

The division model is preferred over the polynomial model because it requires fewer terms than the polynomial model in case of severe distortion [50]. It is also slightly easier to work with; inverting the single-parameter division model, for example, requires solution of a polynomial of degree two, whereas inverting the single-parameter polynomial model leads to a polynomial of degree three. In our work, we use the single-parameter division model (fixing $\lambda_2 = \cdots = 0$), because for most cameras, a single term is sufficient [17, 21, 50, 51]. When the center of distortion is not the origin, we can write the single-parameter division model in the form

$$x_u = x_0 + \frac{x_d - x_0}{1 + \lambda r_d^2}$$
$$y_u = y_0 + \frac{y_d - y_0}{1 + \lambda r_d^2} \tag{4}$$

with r_d^2 defined according to Equation (3). Strand and Hayman [43] find that for the typical case of relatively small barrel distortion (small negative values for λ), the single-parameter division model is highly correlated with the single-parameter polynomial model.

2.2 Distortion of a line under the single-parameter division model

Wang et al. [50] show that under the single-parameter division model, the distorted image of a straight line is a circular arc. However, they use the slope-y-intercept form of the equation of a line, which we avoid due to its inability to model vertical lines and its undesirable numerical properties. However, it is also easy to show that the general line

$$ax_u + by_u + c = 0 \tag{5}$$

is also imaged as a circular arc under the single parameter division model. To avoid the degenerate case $a = b = 0$, we impose the constraint that $a^2 + b^2 > 0$. (When convenient we will further assume the line parameters are normalized so that $a^2 + b^2 = 1$.) By substituting the image coordinates from Equation (4) into Equation (5), replacing r_d^2 by its definition from Equation (3), and simplifying, we obtain the circle

$$x_d^2 + y_d^2 + ex_d + fy_d + g = 0, \tag{6}$$

where

$$e = \frac{a}{c\lambda} - 2x_0$$
$$f = \frac{b}{c\lambda} - 2y_0 \qquad (7)$$
$$g = x_0^2 + y_0^2 - \frac{a}{c\lambda}x_0 - \frac{b}{c\lambda}y_0 + \frac{1}{\lambda} \; .$$

It is also possible to come to the conclusion that straight lines are imaged as circles using the parametric form of a straight line [43].

2.3 Inverse mapping

When undistorting an image, it is necessary to compute, for each pixel in the output undistorted image, the corresponding pixel position in the distorted image then perform interpolation to determine the actual pixel color or intensity in the output undistorted image (we use simple bilinear interpolation in all of the experiments reported on in this paper). However, while every distorted image point (x_d, y_d) is mapped to a unique undistorted image point (x_u, y_u) by Equation (4), the reverse is not true. To invert Equation (4) and find the value of x_d and y_d as a function of x_u and y_u, we first square and add the individual equations to obtain

$$(x_u - x_0)^2 + (y_u - y_0)^2 =$$
$$\frac{1}{(1 + \lambda r_d^2)^2}((x_d - x_0)^2 + (y_d - y_0)^2). \quad (8)$$

We then let r_u be the distance of (x_u, y_u) to the distortion center:

$$r_u^2 = (x_u - x_0)^2 + (y_u - y_0)^2. \qquad (9)$$

This lets us simplify Equation (8) to

$$r_d^2 - \frac{1}{\lambda r_u}r_d + \frac{1}{\lambda} = 0. \qquad (10)$$

For positive λ (pincushion distortion), given $0 < r_u^2 < \frac{1}{4\lambda}$, Equation (10) has two positive real roots. We use the smaller of the two. For negative λ (barrel distortion), given any $r_u^2 > 0$, there are always two real solutions, but one is negative. We use the positive solution. After solving for r_d in terms of r_u, the distorted image coordinates corresponding to (x_u, y_u) can be obtained as

$$x_d = x_0 + \left(\frac{r_d}{r_u}\right)(x_u - x_0)$$
$$y_d = y_0 + \left(\frac{r_d}{r_u}\right)(y_u - y_0). \qquad (11)$$

2.4 Estimating distortion parameters from circular arcs

Strand and Hayman [43] and Wang et al. [50] show that it is possible to estimate λ from the parameters of circular arcs identified in an image. Wang et al. [50] further show how both λ and the distortion center (if not assumed to be the center of the image) can be estimated from the parameters of three circular arcs identified in an image. We use their formulation. In equation (7), multiplying the equation for e by x_0, the equation for f by y_0, and adding the equations for ex_0, fy_0, and g, we obtain

$$x_0^2 + y_0^2 + ex_0 + fy_0 + g - \frac{1}{\lambda} = 0 \qquad (12)$$

For each of the three arcs $i \in \{1, 2, 3\}$, we use Equation (12) to obtain coefficients e_i, f_i, and g_i, then the distortion center can be estimated by solving the linear system

$$(e_1 - e_2)x_0 + (f_1 - f_2)y_0 + (g_1 - g_2) = 0$$
$$(e_1 - e_3)x_0 + (f_1 - f_3)y_0 + (g_1 - g_3) = 0, \qquad (13)$$

and an estimate of λ can be obtained from

$$\frac{1}{\lambda} = x_0^2 + y_0^2 + ex_0 + fy_0 + g \qquad (14)$$

using any of the three arcs' parameters in place of e, f, and g. See Wang et al. [50] for details.

3 Robust Radial Distortion Estimation

In this section, we provide a detailed algorithm for estimating the parameters of the mathematical model introduced in Section 2.

3.1 Identifying circular arcs

The first step in our method is to robustly identify as many non-overlapping circular arcs as possible in the distorted input image. Each arc is identified by a circle center, circle radius, and the contiguous sequence of pixels consistent with that circle.

To find arcs, we first extract edges and link adjacent edge pixels remaining contours. We discard any contour whose length is less than $l^{\min}$ pixels (we use $l^{\min} = 10$ pixels) and then we attempt to find long pixel subsequences within each contour that can be fit by circular arcs. Our method is based on random sampling and inspired by RANSAC [20], but, rather than finding a single model for all the data, we preserve all models (candidate circular arcs) not overlapping with

Input: Contours $C_1, C_2, \ldots$
Output: A is the output arc set
$A \leftarrow \emptyset$
foreach *contour C_i* **do**
    if $|C_i| \geq l^{min}$ **then**
        $N \leftarrow f(l^{\min}, |C_i|)$
        for $n \leftarrow 1$ **to** N **do**
            Sample three points $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$ from C_i.
            if $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$ *are not collinear* **then**
                Calculate x_c, y_c, r from $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$
                $A^{\mathrm{new}} \leftarrow$ arc for longest subsequence of C_i consistent with x_c, y_c, r
                if $|A^{new}| \geq l^{min}$ **then**
                    if A^{new} *does not overlap with any arc in A* **then**
                        $A \leftarrow A \cup \{A^{\mathrm{new}}\}$
                    end
                    else if A^{new} *is longer than every overlapping arc in A* **then**
                        Remove arcs overlapping with A^{new} from A
                        $A \leftarrow A \cup \{A^{\mathrm{new}}\}$
                    end
                end
            end
        end
    end
end

Algorithm 1: Robust arc identification.

other arcs in the same contour that have more support. The termination criterion is to stop once the probability that an arc of minimal length has not yet been found is small.

In Algorithm 1, we provide the details of the method. To determine the number of sampling iterations required, the algorithm uses a function $f(l, n)$, which gives the number of trials required to ensure that the probability of not sampling three of l inliers from a set of n points is small. This ensures that we sample a sufficient number of times to find, with high probability, all arcs with sufficient length in each contour.

3.2 Refining circular arc estimates

After the initial arc identification process is complete, each resulting arc, whose parameters have been calculated directly from the minimum sample of three points, is refined using the inlier pixel contour subsegment supporting that model. The gold standard objective function for circle fitting is

$$\Omega(x_c, y_c, r) = \sum_{i=1}^{N} d(x_i, y_i, x_c, y_c, r)^2, \qquad (15)$$

where (x_c, y_c) is the center of the circle, r is its radius, and $d(x, y, x_c, y_c, r)$ is the orthogonal distance of the

measured point (x, y) to the hypothetical circle. N is the number of pixels in a inlier contour. Since there is no closed-form solution for minimizing this objective function [1], we use an initial guess and the Levenberg-Marquardt (LM) iterative nonlinear least squares method to find a local minimum. To obtain the initial guess, we use a variety of methods as detailed in the next section.

3.2.1 Algebraic circle fitting methods

As the initial estimate of the circle's parameters, we use either the parameters calculated during the sampling procedure or one of three circle fitting methods, Pratt [35], Taubin [45], and Kukush-Markovsky-van-Huffel (KMvH) [29], based on algebraic error minimization. Both Pratt and Taubin use four parameters to specify a circle:

$$a(x^2 + y^2) + bx + cy + d = 0, \qquad (16)$$

with $a \neq 0$. The center of the circle is $(-\frac{b}{2a}, -\frac{c}{2a})$ and the radius is given by $r = \sqrt{(-\frac{b}{2a})^2 + (-\frac{c}{2a})^2 - \frac{d}{a}}$. The Pratt method minimizes the objective function

$$\Omega(a, b, c, d) = \sum_{i=1}^{N} a(x_i^2 + y_i^2) + bx_i + cy_i + d^2, \qquad (17)$$

subject to the constraint that $b^2 + c^2 - 4ad = 1$, to ensure that the parameterized equation represents an actual circle. The Taubin method minimizes the same objective function as the Pratt method, but imposes the constraint $4a^2\bar{z} + 4ab\bar{x} + 4ac\bar{y} + b^2 + c^2 = 1$, where $\bar{x}$ is the mean of the sample points' x coordinates, $\bar{y}$ is the mean of the sample points' y coordinates, and $\bar{z} = \frac{1}{N} \sum_{i=1}^{N} (x_i^2 + y_i^2)$. The additional constraint improves the convergence of the optimization [15].

We use one other algebraic method, the Kukush-Markovsky-van-Huffel (KMvH) consistent conic fitting method [29], which minimizes the objective function $\Omega(a, b, c, d) = \boldsymbol{A}^T \mathtt{M} \boldsymbol{A}$, where $\boldsymbol{A} = \begin{bmatrix} a & b & c & d \end{bmatrix}^T$ and $\mathtt{M}$ is an unbiased estimate of the data covariance matrix. The method guarantees convergence to the true parameters as the number of sampled points approaches infinity.

3.2.2 Geometric circle fitting methods

Researchers have proposed several circle-specific parameter refinement methods based on LM. In addition to generic LM, in this paper, we also experiment with Trust-Region-LM, Chernov and Lesort LM (Chernov-LM), and Reduced-LM.

In the Reduced-LM method [15], only two parameters are adjusted; i.e., x_c and y_c (the coordinates of the

center of the circle). The method minimizes the objective function

$$\Omega(x_c, y_c) = \sum_{i=1}^{N} \left[\bar{r} - \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \right]^2,$$

where $\bar{r} = \frac{1}{N} \sum_{i=1}^{N} \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}$.

The Chernov and Lesort LM method [16] guarantees convergence to a minimum of the objective function from any initial guess. They redefine the circle parameters with respect to Equation 16 as $A = -\frac{b}{2a}, B = -\frac{c}{2a}$, and $R^2 = \frac{b^2 + c^2 - 4ad}{4a^2}$. Their objective function is

$$\Omega(a, b, c, d) = 2 \frac{P_i}{1 + \sqrt{1 + 4aP_i}},$$

where $P_i = a(x_i^2 + y_i^2) + bx_i + c_i + d$.

The Trust-Region-LM method [32] guarantees proper initialization of LM's learning rate or control parameter λ and also provides an efficient rule for updating λ's value.

We coded all the geometric and algebraic circle fitting methods in C++ using OpenCV [6], with reference to Chernov's [14] MATLAB implementations.

3.3 Experimental design for circular arc estimation methods

We have found that different circle fitting methods provide very different performance in terms of radial distortion estimation. In the experimental evaluation (Section 4), we therefore perform a comprehensive study of the effect the circle fitting algorithm has on radial distortion correction. We use 10 different variations of the fitting methods described in the previous section. For a comparative summary of the different circle fitting algorithms, refer to Table 1, and for more detailed discussion of the methods, refer to Chernov [15].

In our experiments, the "Ransac" method means we simply accept the circular arc model computed from three sample points, without any refinement after calculating inliers. "Ransac-Taubin," "Ransac-Pratt," and "Ransac-KMvH" are the results of using the Taubin, Pratt, or KMvH methods to refine the arc models computed from three sample points. The names "Ransac-LM," "Ransac-TR-LM," "Ransac-Chernov-LM," and "Ransac-Reduced-LM" denote the application of general LM, Trust Region LM, Chernov's LM, or the Reduced LM methods previously described directly to the models computed from three sample points. Under the hypothesis that starting LM from the sample-based estimate might not work as well as an initial estimate closer to the optimum, we also performed two series of

experiments in which we first applied either the Taubin or the Pratt methods to the sample-based models then applied general LM to the Taubin or the Pratt estimates respectively. The results from these methods are shown as "Ransac-Taubin-LM" and "Ransac-Pratt-LM."

3.4 Estimating distortion parameters

Once we have obtained a set of circular arcs as candidate distorted straight lines, we use the estimator of Equations 13 and 14 and a RANSAC procedure to find the set of distortion parameters with maximal support. However, we use a modified notion of support; rather than counting the number of arcs fit by a particular model, we count the sum of the lengths (in pixels) of the arcs. Longer arcs provide much more accurate parameter estimates than shorter arcs. The weighted support strategy emphasizes models that fit as many long arcs as possible rather than models that fit many small arcs. In the sampling loop, we sample three arcs, calculate the model, then classify each arc as an inlier if, after undistortion, the arc's pixels fit a straight line.

The details are presented in Algorithm 2. In the sampling loop, we use adaptive calculation of the number of iterations required based on the highest number of pixels in inlier arcs seen so far. The termination criterion uses the same function $f(l, n)$ previously introduced to determine the number of trials required to ensure that the probability of never sampling three arcs from a consensus set of size (in pixels) at least the size of the current support set is small.

To judge whether an arc is an inlier for distortion parameters (λ, x_0, y_0), we first perform orthogonal regression to determine the line best fitting the pixels of the arc before and after undistortion using the candidate parameters. We use two criteria: the undistorted arc should be very close to the straight line in terms of RMSE, and the RMSE for the undistorted arc should be smaller than that of the original distorted arc. The second criterion is necessary to handle arcs that are already very close to straight lines in the original distorted image. We require that these arcs should not only be nearly straight after undistortion, but that they should also be more straight than they were before undistortion. This avoids situations in which incorrect distortion parameters make nearly straight contours less straight but still close enough to straight to be counted as inliers.

Once the required number of iterations have been performed, we obtain a final least squares estimate of the distortion parameters based on all inlier arcs and Equations 13 and 14. From each possible pair of inlier arcs, we form the linear system described by Equation

Table 1: Comparison of different circle fittings methods. LM = Levenberg Marquardt, KMvH = Kukush Markovsky van Huffel; IURAT = Invariant Under Rotations And Translations. $r_i = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}$, $\bar{r} = \frac{1}{N}\sum_{i=1}^{N} r_i$; $A = \pm\frac{1}{2R}$, $B = -2Aa$, $C = -2Ab$, $D = \frac{B^2 + C^2 - 1}{4A}$; $\bar{z} = \frac{1}{N}\sum_{i=1}^{N}\left(x_i^2 + y_i^2\right)$. (x_c, y_c) is the center and R is the radius of the circle. $\boldsymbol{A}$ is a vector representing the parameters of a circle.

Method	Initial guess	Outliers resistant	Geometric	Analytic	IURAT	Objective function	Comments
Ransac	Yes	Yes	No	No	Yes	None	Simple calculation of circle parameters from three points.
Generic-LM	Yes	No	Yes	No	Yes	$\Omega(x_c, y_c, R) = \sum_{i=1}^{N}(r_i - R)^2$	Properly initializing the learning rate is cumbersome [15].
Chernov-LM	Yes	No	Yes	No	No	$\Omega(a, b, c, d) = 2\frac{P_i}{1 + \sqrt{1 + 4aP_i}}$, where $P_i = a(x_i^2 + y_i^2) + bx_i + c_i + d$	Guaranteed convergence to a minimum of the objective function but cost per iteration is 4 times higher than any other geometric circle fitting method. Sensitive to the choice of coordinate system [16].
Reduced-LM	Yes	No	Yes	No	Yes	$\Omega(x_c, y_c) = \sum_{i=1}^{N}(\bar{r} - \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2})^2$, where $\bar{r} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}$	Objective function based only on the circle center. Faster than generic LM but more sensitive to bad initial guesses [15].
Trust-Region-LM	Yes	No	Yes	No	Yes	$\Omega(x_c, y_c, R) = \sum_{i=1}^{N}(r_i - R)^2$	Good rule for proper initialization and updating of the learning rate [32].
Pratt	No	No	No	Yes	Yes	$\Omega(a, b, c, d) = \sum_{i=1}^{n} a(x_i^2 + y_i^2) + bx_i + cy_i + d^2$, subject to the constraint $b^2 + c^2 - 4ad = 1$	Estimate converges to true parameters in case of small noise [35].
Taubin	No	No	No	Yes	Yes	$\Omega(a, b, c, d) = \sum_{i=1}^{n} a(x_i^2 + y_i^2) + bx_i + cy_i + d^2$, subject to the constraint $4a^2\bar{z} + 4ab\bar{x} + 4ac\bar{y} + b^2 + c^2 = 1$	Can be generalized to ellipses and other geometric curves [45]. Less biased than Pratt's method. Considered statistically more accurate than Pratt [1].
KMvH	No	No	No	Yes	Yes	$\Omega(a, b, c, d) = \boldsymbol{A}^T \mathtt{M} \boldsymbol{A}$, where $\boldsymbol{A} = \begin{bmatrix} a\ b\ c\ d \end{bmatrix}^T$ and $\mathtt{M}$ is an unbiased estimate of the data covariance matrix.	Guaranteed convergence to the true parameters as the number of sampled points approaches infinity [15, 29].

13 and find the least-squares solution for the distortion center (x_0, y_0). After estimating the distortion center, we estimate λ using Equation 14 and all inlier arcs.

3.5 Undistortion

The last step in our procedure is to undistort the input image. We use the optimal distortion parameters computed per the previous section and the inverse of the distortion model in Equation 11 with bilinear interpolation and appropriate translation and scale factors to produce the output undistorted image.

4 Quantitative Results on Synthetic Images

In this section, we describe a detailed quantitative study of the performance of our method on synthetic images. A sample of the synthetic images we use with results is shown in Fig. 1. We used the same original image (Fig. 1(a)) for all synthetic image experiments. In each experiment, we distort the original image using particular ground truth values for λ, x_0, and y_0 (Fig. 1(b)), extract Canny edges (Fig. 1(c)), link the edge pixels into contours (Fig. 1(d)), identify circular arcs among the contours (Fig. 1(e)), estimate the distortion parameters, and then use those parameters to undistort the image.

Input: Arc set A
Output: λ^*, x_0^*, y_0^* are the output distortion
          parameters
begin
    $(\lambda^*, x_0^*, y_0^*) \leftarrow (\emptyset, \emptyset, \emptyset)$
    if $|A| \geq 3$ **then**
        $n \leftarrow 0$
        $s^* \leftarrow 0$
        while *true* **do**
            $n \leftarrow n + 1$
            Sample three distinct arcs A_1, A_2, A_3 with
            probability proportional to lengths $|A_i|$.
            Estimate (λ, x_0, y_0) from A_1, A_2, A_3 per
            Equations 13 and 14
            $s \leftarrow$ support (in pixels) for (λ, x_0, y_0)
            if $s > s^*$ **then**
                $s^* \leftarrow s$
                $(\lambda^*, x_0^*, y_0^*) \leftarrow (\lambda, x_0, y_0)$
            end
            if $n \geq f(s, \sum_{i=1}^{|A|} |A_i|)$ **then**
                break
            end
        end
        Reestimate $(\lambda^*, x_0^*, y_0^*)$ using inlier arcs.
    end
end

Algorithm 2: Robust distortion parameter estimation.

To evaluate the quality of reconstruction of the original synthetic image, we use root mean-squared error (RMSE) and peak signal-to-noise ratio (PNSR), the most commonly used image quality metrics for cases in which the original undistorted image is available for comparison.

4.1 Synthetic images

We performed two series of experiments with synthetic images. For edge extraction, we modified OpenCV's Canny edge detector to automatically select a low gradient threshold and a high gradient threshold based on a cumulative histogram of magnitudes, similar to the Matlab implementation of Canny's method [31].

4.1.1 Experiment 1 (varying λ)

In a first series of runs, we varied λ while keeping the distortion center fixed at $(x_0, y_0) = (320, 240)$ but estimated all three parameters. For each level of λ, we compare 10 methods for circular arc estimation.

Fig. 2 shows the distorted image and a sample undistorted result for each level of λ using the "Ransac-LM" method. For the extreme case of $\lambda = 10^{-5}$, undistorted image points map to multiple valid distorted image points, so we only map the points for which $r_d^2 < \lambda$,
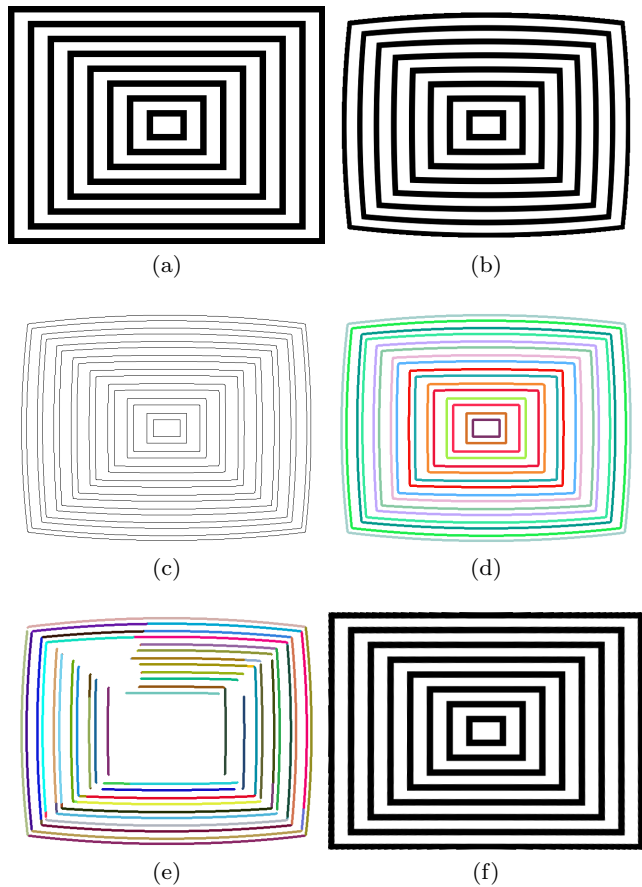

Fig. 1: Example experiment with synthetic image size 640×480. (a) Original image. (b) Distorted image with $\lambda = -10^{-6}$ and $(x_0, y_0) = (320, 240)$ (the image center). c) Canny edges (d) Extracted contours. (e) Estimated arcs. (f) Undistorted image using estimated values of $\lambda = -1.00419e^{-6}$, $x_0 = 319.352$, and $y_0 = 238.009$. Using true parameters RMSE = 3.27813 and PNSR = 37.8183 dB; Using estimated parameters RMSE = 3.86511 and PNSR = 36.3876 dB.

resulting in a circular valid region around the image center.

To precisely quantify the performance of each algorithm, for each level of λ and each circle fitting method, we performed 10 runs with different random seeds and collected three measurements in each case: the absolute relative estimation error for λ, i.e., $|(\lambda_{\text{est}} - \lambda_{\text{true}})/\lambda_{\text{true}}|$, RMSE, and PNSR. For the extreme case of $\lambda = 10^{-5}$, we only calculated RMSE and PNSR over the circular valid region shown in Fig. 2. The mean measurements with 95% confidence intervals are shown in Fig. 3(a–c).

The results in terms of relative estimation error for λ in Fig. 3(a) show quite clearly that our method is extremely accurate at estimating λ for moderate ($\lambda =$
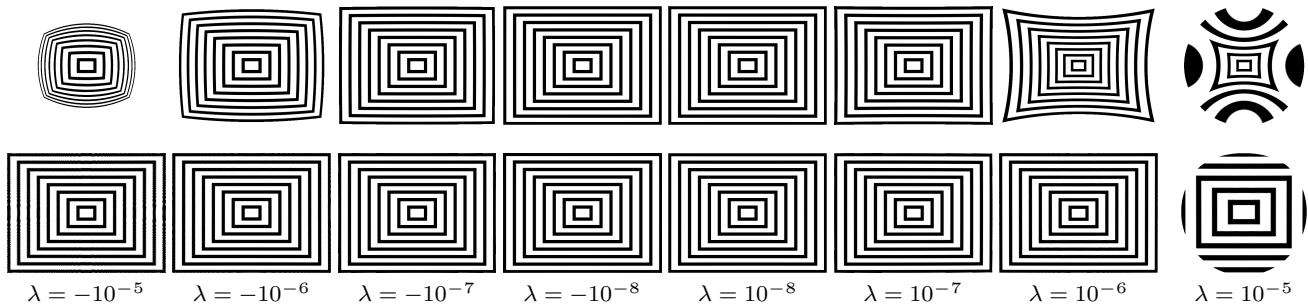
Fig. 2: Undistortion of synthetic images. Image size is 640×480 and distortion center is $(320, 240)$. First row: distorted images at different levels of λ. Second row: corresponding undistorted images using parameters estimated by the "Ransac-LM" circle fitting method.

$\pm 10^{-6}$) or extreme ($\lambda = \pm 10^{-5}$) distortion but quite inaccurate for very small levels of distortion. The inaccuracy with small distortion levels reflects two factors. First, since the ground truth value is extremely small in the first place, small deviations between estimated and ground truth parameter values give large relative errors. Second, when our algorithm fails to find a sufficient number of contours that can be modeled as circular arcs, it defaults to an estimate of $\lambda = 0$, which leads to a relative error of 1. Fortunately, as the RMSE and PNSR comparisons show, this relative inaccuracy for small λ does not affect our method's ability to reconstruct the original undistorted image.

Since the RMSE and PNSR results shown in Fig. 3(b–c) are difficult to interpret, we performed a series of statistical analyses. As the dependent variables, we used pixel intensity RMSE and PNSR. As the independent variables, we used the algorithm and the different levels of λ.

For both RMSE and PNSR, a two-way analysis of variance (ANOVA) revealed main effects for both predictor variables and an interaction. To understand the main effect of differing ability of each algorithm to reconstruct the original image, we performed a post-hoc analysis using the Tukey correction for all pairwise comparisons among the 10 different algorithms.

For both RMSE and PNSR, the best numerical results were obtained with Ransac-Pratt, but statistically, according to both measures, four algorithms, namely Ransac-Pratt, Ransac-Pratt-LM, Ransac-LM, Ransac-Reduced-LM, Ransac-Taubin, and Ransac-Taubin-LM are equivalent to and better than the remaining four algorithms (we use a familywise $\alpha = 0.05$ for all significance tests). Next are Ransac-KMvH, Ransac-Chernov-LM, and Ransac-TR-LM. These three algorithms are statistically equivalent. The last method, Ransac, is statistically worse than all the other algorithms.

4.1.2 Experiment 2 (varying distortion center)

In a second series of runs, we kept the distortion fixed at a moderate level of barrel distortion ($\lambda = -10^{-6}$) while varying the distortion center, then we estimated all three parameters of the distortion model. We used the same 10 circle fitting methods as in the first series of experiments manipulating λ. For each run, we measured the Euclidean distance between the estimated and ground truth distortion center as well as RMSE and PNSR. The results are shown in Fig. 3(d–f).

To further understand the results, we performed statistical analyses with all three dependent measures. As the independent variables, we used the algorithm and the different levels of distortion center. To simplify the presentation, we name the distortion center levels as $DC_1 = 0.0$ (distortion center $(320, 240)$), $DC_2 = 14.1$ (distortion center $(330, 250)$), $DC_3 = 28.3$ (distortion center $(340, 260)$), $DC_4 = 42.4$ (distortion center $(350, 270)$), $DC_5 = 56.6$ (distortion center $(360, 280)$), $DC_6 = 70.7$ (distortion center $(370, 290)$), $DC_7 = 84.9$ (distortion center $(380, 300)$), $DC_8 = 99.0$ (distortion center $(390, 310)$).

For all three dependent measures, two-way analyses of variance (ANOVAs) revealed main effects for both predictor variables and an interaction. We then performed post-hoc Tukey comparisons among the different algorithms and among the different distortion center distances.

For distortion center estimation error, Ransac-LM was numerically best but Ransac-LM, Ransac-Reduced-LM, Ransac-Pratt, Ransac-Taubin-LM, Ransac-KMvH, and Ransac-Pratt-LM were statistically equivalent and better than the other four algorithms. Distortion center estimation error was also affected by the distance between the distortion center from the image center. The results for DC_1 were significantly better than those for DC_2, and DC_7 was better than DC_8. However, DC_6
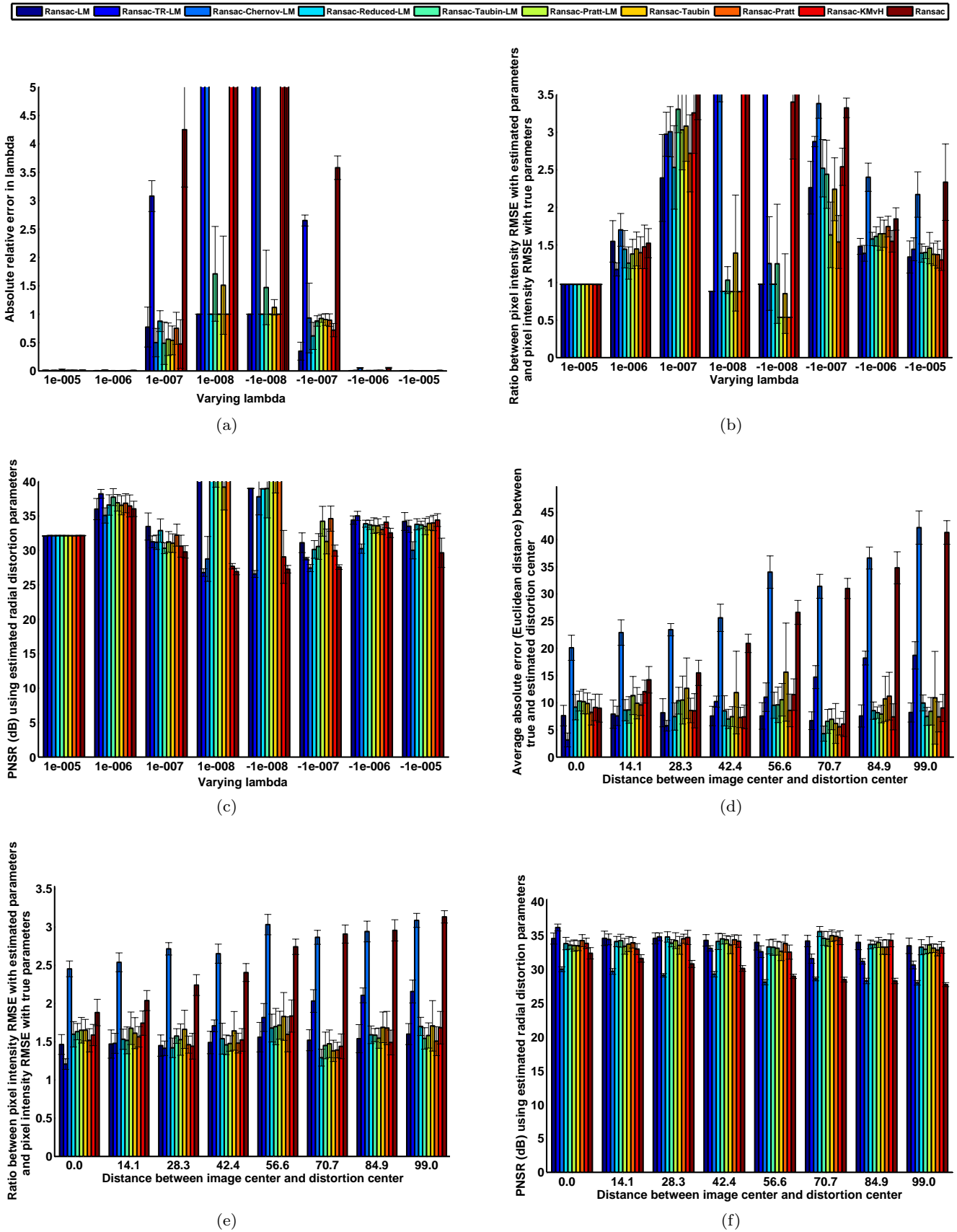
Fig. 3: Results of experiment 1 and experiment 2 using synthetic images. (a-c) represent results with varying λ, with distortion center fixed at the image center. (d-f) represent results with varying distance of the distortion center to the center of the image. λ is fixed at -10^{-6}. All graphs show averages over 10 runs with error bars showing 95% confidence intervals on the mean. (a) True versus estimated λ. (b) Average RMSE. (c) Average PNSR. (d) True versus estimated distance of the distortion center from the image center. (e) Average RMSE. (f) Average PNSR.

happened to be as easy as DC_4 and easier than either DC_5 or DC_7. Inspection of Fig. 3(d) indicates that the increasing trend is driven by just two or three algorithms.

For pixel intensity RMSE, Ransac-LM was numerically best but statistically equivalent to Ransac-Pratt. The next set was Ransac-Reduced-LM, Ransac-Taubin-LM, Ransac-Pratt-LM, Ransac-KMvH, and Ransac-Taubin. Statistically, these algorithms were equivalent to each other and better than the remaining three algorithms. There was also an effect of distortion center distance on RMSE; the results for DC_1, DC_2, DC_3, and DC_4 were statistically equivalent but the results for DC_6, DC_7, DC_4, and DC_8 were all significantly different with respectively increasing RMSE.

Finally, the ordering in terms of increasing PNSR was Ransac-LM, Ransac-Pratt, Ransac-Reduced-LM, Ransac-Taubin-LM, Ransac-Pratt-LM, Ransac-KMvH, Ransac-Taubin, Ransac-TR-LM, Ransac, Ransac-Chernov-LM. All differences were significant except among Ransac-Pratt-LM, Ransac-KM-vH, and Ransac-Taubin, which were statistically equivalent. Distortion center distance also had some effect, with DC_1, DC_2, and DC_3 yielding the best PNSR. These levels were significantly better than the others, for which the trend was DC_6, DC_4, DC_7, DC_5, DC_8, with significantly increasing PNSR, respectively.

4.1.3 Discussion of synthetic image experiments

Over the two series of runs, we observe that with moderate or extreme distortion, our method readily identifies the parameters of the distortion model and is successful at reconstructing the original undistorted image with high accuracy. For lower levels of distortion, the model parameters are more difficult to estimate accurately, but this inaccuracy does not affect the reconstruction results by much: Ransac-LM introduces at most only 2.5 times the reconstruction error of bilinear interpolation with the ground truth parameters.

Although some sensitivity to distortion center distance is observed, we can see readily from the data that this is only for some algorithms. The best algorithms such as Ransac-LM and Ransac-Pratt are clearly not affected by this factor. This is an improvement over other work [2, 8, 21, 28, 43].

Statistically, Ransac-LM and Ransac-Pratt are the winning algorithms. It is difficult to choose between the two since they both yield excellent performance. Ransac-Pratt is more computationally efficient. Ransac-LM provides significantly better PNSR over varying distortion center distances, and it has a somewhat lower maximum RMSE over varying levels of distortion. For these reasons, we have a slight preference for Ransac-LM.

Our method selects the distortion model consistent with the largest possible number of arcs found in the image. It could certainly be fooled by a synthetic image contain a group of real world curves that happen to be consistent with each other, leading to incorrect distortion parameter estimation. However such egregious cases are extremely unlikely to arise in practice. So long as there are several distorted straight lines, our algorithm will find them and estimate a distortion model to undistort those lines ignoring the curves or outliers.

5 Qualitative Results on Real Images

Next, we present a qualitative evaluation of the proposed method's ability to identify distortion parameters in several challenging real images from the Web and other papers on radial distortion estimation. The set contains images with severe barrel and pincushion distortion, showing the effects of fisheye lenses and wide angle lenses. Fig. 4 shows step-by-step results for one of the images, from Ociepka [33]. Note that some contours in Fig. 4(c) cannot be modeled as circular arcs. The algorithm discards contours due to 1) our criteria for circular arc selection is that a contiguous sequence of pixels must be consistent with a circle, 2) the radius of an identified circular arc should be less than the five times the image width, in order to discard long straight lines in the distorted image, and 3) we discard any contour whose length is less than 10 pixels, as small arcs tend to give suboptimal estimates of the distortion parameters. Fig. 5 summarizes the results for all of the images from the Web and previous papers on distortion estimation [3–5, 10, 12, 13, 18, 19, 27, 30, 34, 37–41, 44, 47, 48, 52, 53]. These results indicate the robustness and accuracy of our procedure. Many are difficult due to severe fisheye distortion and circular arcs that are not straight lines in the real world. Despite these challenges, our robust arc selection method is able to find consensus sets corresponding to distorted straight lines and is successful at removing most of the radial distortion from the images.

6 Comparison with Álvarez et al.'s [2] method

How do the results presented thus far compare to previous work? Since there is no standardized database with ground truth for radial distortion correction, it is difficult to say. However, Álvarez et al. [2] have deployed an excellent demo Web site for their method that allow users to submit an image for undistortion after manually selecting distorted lines from it. The method is also
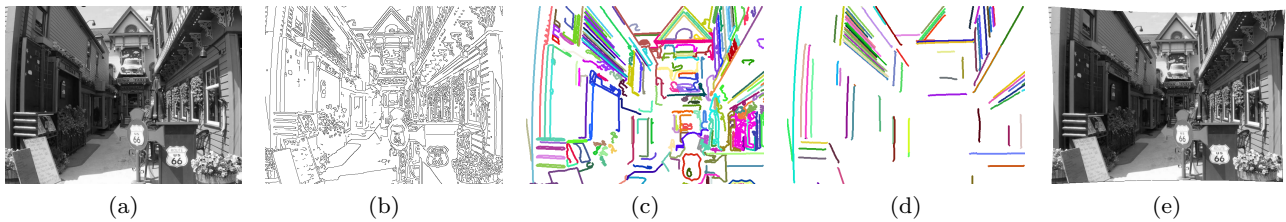
Fig. 4: Example results on real image. (a) Original image. (b) Detected edges. (c) Extracted contours. (d) Identified arcs. (e) Undistorted image using parameters estimated via the "Ransac-LM" circle fitting method. The distorted image is taken from Ociepka [33].

Table 2: Comparison with Alvarez et al. [2]. Image size is 640×480.

Method	Distortion center	RMSE (pixel intensity)	PNSR (dB)	Cpu time (s)	Total number of arcs
Álvarez et al. [2]	320×240	4.6274	34.8241	1.219[1]	81
	390×310	8.5204	29.5216	1.307[1]	73
Our method	320×240	3.86525	36.3872	6.0463[2]	81
	390×310	3.91738	36.2709	5.2171[2]	73

1. Includes network latency but does not include time to manually select contours.
2. Includes time to perform edge processing, automatically select circular contours, and estimate arcs.

plumb-line based, with source code available online. We selected two synthetic images as shown in Fig. 6 (a) and Fig. 6 (b) for comparison with their method. In the first image, the distortion center is $(320, 240)$ and $\lambda = -10^{-6}$; in the second image we moved the distortion center to $(390, 310)$ but kept λ fixed. For fair comparison, before submission, we selected the same number of lines while comparing our method with their method. To achieve this, we ran our algorithm to determined how many arcs are extracted by our method, then based on that we selected the same number of distorted lines, and as much as possible number of pixels for the Álvarez et al. method, and after obtaining the results, we manually scaled and translated the resulting images to be in the best possible alignment with the original undistorted image. Table 2 presents the results. When the image center is the distortion center, the Canny edge execution time is $0.6539s$, arc detection takes $1.5945s$, distortion parameter estimation time is $3.7980s$, and the total execution time is $6.0463s$. When the image center is not the distortion center, the Canny edge execution time is $0.6542s$, arc detection takes $1.6309s$, distortion parameter estimation time is $2.9320s$, and the total execution time is $5.2171s$. The Álvarez et al. method runs faster, even taking the Web service's network latency into account, but requires manual intervention to select distorted straight lines. It took us around 1 hour to select 81 contours using their Web demo. When the image center is the distortion center, the Álvarez et al. method performs well, with a moderate 19.718% increase in RMSE and 1.5 dB decrease in PNSR compared

to our method. When the distortion center is not the image center, our method degrades only slightly, but the Álvarez et al. method degrades substantially. This is expected because the Álvarez et al. method is not designed to handle distortion centers that are not the image center. But our method still performs slightly better than the Álvarez et al. method even when the image center *is* the distortion center, even though our algorithm is estimating rather than assuming the location of the image center and even though the method is fully automatic with no user intervention.

7 Conclusion

In this paper, we have introduced a new algorithm for automatic radial distortion estimation and removal using the plumb-line approach. The method works from a single image and does not require a special calibration pattern. It is based on Fitzgibbon's division model, robust estimation of circular arcs, and robust estimation of distortion parameters. As our method is based on circles, we also provide a detailed study of circle fitting methods and have found that two circle fitting methods, namely "Ransac-LM" and "Ransac-Pratt" perform better than the remaining 8 algorithms. "Ransac-Pratt" is a non-iterative circle fitting method and performs nearly as well as "Ransac-LM". Since "Ransac-Pratt" is computationally cheaper than "Ransac-LM," it may be recommended for applications in which runtime is important. Robust automatic radial distortion estima-

Fig. 5: Undistortion of real images. Column 1, column 3, and column 5 represent original images. Column 2, column 4, and column 6 represent undistorted images using parameters estimated via the "Ransac-LM" circle fitting method. The distorted images are taken from several sources [3–5, 10, 12, 13, 18, 19, 27, 30, 34, 37–41, 44, 47, 48, 52, 53].

tion from a single natural image would be extremely useful for many applications, particular those in human-made environments containing abundant lines. For example, it could be used to get a mobile robot or quadrotor experiment up and running quickly in an indoor environment. In a series of experiments on synthetic and challenging real images, we have demonstrated the method's ability to accurately identify distortion parameters and remove distortion from images. Data and source code based on OpenCV [6] is available online[1] for researchers interested in evaluating or extending our procedure.

References

1. Al-Sharadqah, A., Chernov., N.: Error analysis for circle fitting algorithms. The Electronic Journal of Statistics **3**, 886–911 (2009)
2. Alvarez, L., Gomez, L., Sendra, J.R.: An algebraic approach to lens distortion by line rectification. Journal of Mathematical Imaging and Vision **39**(1), 36–50 (2009)
3. Alvarez, L., Gomez, L., Sendra, J.R.: Algebraic lens distortion model estimation (2010). `http://www.ipol.im/pub/algo/ags_algebraic_lens_distortion_estimation/`
4. Barreto, J.P., Daniilidis, K.: Fundamental matrix for cameras with radial distortion. In: International Conference on Computer Vision (ICCV), pp. 625–632 (2005)
5. Bockaert, V.: Pincushion distortion (n.d.). `http://www.dpreview.com/learn/?/Glossary/Optical/Pincushion_Distortion_01.htm`
6. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
7. Braüer-Burchardt, C.: A simple new method for precise lens distortion correction of low cost camera systems. In: German Pattern Recognition Symposium, pp. 570–577 (2004)
8. Brauer-Burchardt, C., Voss, K.: A new algorithm to correct fish-eye- and strong wide-angle-lens-distortion from single images. In: IEEE International Conference on Image Processing, vol. 1, pp. 225 – 228 (2001)
9. Brown, D.C.: Close-range camera calibration. Photogrammetric Engineering **37**(8), 855–866 (1971)
10. Bucket, P.: Nikon 16mm fisheye photos (n.d.). `http://photobucket.com/images/Nikon+16mm+fisheye/`
11. Bukhari, F., Dailey, M.N.: Robust radial distortion from a single image. In: Proceedings of the 6th international conference on Advances in visual computing - Volume Part II, ISVC'10, pp. 11–20 (2010)
12. Chanel: How to fake the fisheye effect (n.d.). `http://chanelwood.com/how-to/photoshop/how-to-fake-the-fisheye-effect/`
13. Chen, P.Y., Huang, C.C., Shiau, Y.H., Chen, Y.T.: A vlsi implementation of barrel distortion correction for wide-angle camera images. Trans. Cir. Sys. **56**, 51–55 (2009)
14. Chernov, N.: Matlab code for circle fitting algorithms. `http://www.math.uab.edu/~chernov/cl/MATLABcircle.html` (1997)
15. Chernov, N.: Circular and Linear Regression: Fitting Circles and Lines by Least Squares. Chapman & Hall (2010)
16. Chernov, N., Lesort, C.: Least squares fitting of circles. J. Math. Imaging Vis. **23**, 239–252 (2005)
17. Devernay, F., Faugeras, O.: Straight lines have to be straight: Automatic calibration and removal of distortion from scenes of structured enviroments. Machine Vision and Applications **13**(1), 14–24 (2001)
18. Dyer, D.: Wide angle adapters for digital cameras (n.d.). `http://www.andromeda.com/people/ddyer/photo/wideangle.html`
19. El-Melegy, M.T., Farag, A.A.: Nonmetric lens distortion calibration: Closed-form solutions, robust estimation and model selection. Computer Vision, IEEE International Conference on **1** (2003)
20. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981). DOI http://doi.acm.org/10.1145/358669.358692
21. Fitzgibbon, A.W.: Simultaneous linear estimation of multiple view geometry and lens distortion. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp. 125–132 (2001)
22. Friel, M., Hughes, C., Denny, P., Jones, E., Glavin, M.: Automatic calibration of fish-eye cameras from automo-

---
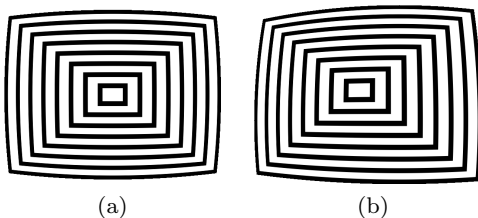
(a)   (b)

Fig. 6: Synthetic images used for comparison with Álvarez et al. [2]. (a) Distortion center is the image center $(320, 240)$. (b) Distortion center is at $(390, 310)$. Image size is 640×480 and $\lambda = -10^{-6}$.

tive video sequences. Intelligent Transport Systems, IET **4**(2), 136–148 (2010)

23. Gonzalez-Aguilera, D., Gomez-Lahoz, J., Rodriguez-Gonzalvez, P.: An automatic approach for radial lens distortion correction from a single image. IEEE Sensors Journal **11**(4) (2011)

24. Hartley, R., Kang, S.: Parameter-free radial distortion correction with center of distortion estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence **29**(8), 1309–1321 (2007)

25. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press (2004)

26. Hughes, C., Glavin, M., Jones, E., Denny, P.: Wide-angle camera technology for automotive applications: a review. Intelligent Transport Systems, IET **3**(1), 19–31 (2009)

27. Kbh3rd: camelback locomotive (2008). `http://en.wikipedia.org/wiki/File:B%2B0_173_camelback_locomotive.jpg`

28. Kukelova, Z., Pajdla, T.: A minimal solution to radial distortion autocalibration. IEEE Transactions on Pattern Analysis and Machine Intelligence **99**(1) (2011)

29. Kukush, A., Markovsky, I., Van Huffel, S.: Consistent estimation in an implicit quadratic measurement error model. Computational Statistics & Data Analysis **47**(1), 123–147 (2004)

30. Laksi, L.: Motorcycle taxi (n.d.). `http://leolaksi.wordpress.com/2009/12/28/more-photos-with-the-nikkor-16mm-f2-8-fisheye-lens/`

31. MATLAB: edge - find edges in grayscale image (2009). `http://www.mathworks.com/help/toolbox/images/ref/edge.html`

32. Moré, J.J.: The Levenberg-Marquardt algorithm: Implementation and theory. Lecture Notes in Mathematics pp. 105–116 (1978)

33. Ociepka, R.: Raynox dcr-720 with barrel distortion (2003). `http://www.pbase.com/ociepka/image/32663644`

34. Oleson, R.: Full-circle examples (n.d.). `http://rick_oleson.tripod.com/index-105.html`

35. Pratt, V.: Direct least-squares fitting of algebraic surfaces. SIGGRAPH Comput. Graph. **21**, 145–152 (1987)

36. Ramalingam, S., Sturm, P., Lodha, S.K.: Generic self-calibration of central cameras. Computer Vision and Image Understanding **114**(2), 210–219 (2010)

37. Rideout, S.: 10 ninjas steve's photostream (2002). `http://www.flickr.com/photos/steverideout/50185284/`

38. Rosten, E., Loveland, R.: Camera distortion self-calibration using the plumb-line constraint and minimal hough entropy. Mach. Vision Appl. **22**, 77–85 (2011)

39. Sarge: Misc (n.d.). `http://sarge.wheresthebeef.co.uk/Misc/350D_misc_100/IMG_2797.jpg`

40. Skewes, K.: Real estate photo tip: Wide angle lens correction (n.d.). `http://blogonlineed.com/2011/01/14/real-estate-photo-tip-wide-angle-lens-correction/`

41. Solheim, E.: How to remove distortion on a fisheye image (n.d.). `http://eirikso.com/2008/12/14/how-to-remove-distortion-on-a-fisheye-image/`

42. Stein, G.P.: Lens distortion calibration using point correspondences. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp. 602–608 (1996)

43. Strand, R., Hayman, E.: Correcting radial distortion by circle fitting. In: British Machine Vision Conference (BMVC) (2005)

44. Swaminathan, R., Nayar, S.: Non-Metric Calibration of Wide-Angle Lenses and Polycameras. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(10), 1172 – 1178 (2000)

45. Taubin, G.: Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **13**(11), 1115–1138 (1991)

46. Tavakoli, H.R., Pourreza, H.R.: Automated center of radial distortion estimation, using active targets. In: Asian Conference on Computer Vision (ACCV) (2010)

47. Thormählen, T., Broszio, H., Wassermann, I.: Robust line-based calibration of lens distortion from a single view. In: Computer Vision / Computer Graphics Collaboration for Model-based Imaging Rendering, Image Analysis and Graphical Special Effects, pp. 105–112 (2003)

48. Tomasi, C.: Sample image for CPS 296.1 homework assignment (2007). `http://www.cs.duke.edu/courses/spring06/cps296.1/homework/1/lab.gif`

49. Tsai, R.Y.: A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. Radiometry pp. 221–244 (1992)

50. Wang, A., Qiu, T., Shao, L.: A simple method of radial distortion correction with centre of distortion estimation. Journal of Mathematical Imaging and Vision **35**(3), 165–172 (2009)

51. Wang, J., Shi, F., Zhang, J., Liu, Y.: A new calibration model of camera lens distortion. Pattern Recogn. **41**, 607–615 (2008)

52. Whittaker, G.: Wide angles (n.d.). `https://picasaweb.google.com/gmw3027/WideAngles#5276761797451570738`

53. Yusuf: Correcting barrel distortion of wide and ultrawide lenses (n.d.). `http://www.photos-of-the-year.com/articles/barrel-distortion/`

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

Faisal Bukhari received the M.Sc. in Statistics from the Institute of Statistics, University of the Punjab (PU), Lahore, Pakistan. He received the M.Sc. in Computer Science from Punjab University College of Information Technology (PUCIT), Lahore, Pakistan. He also received the M.S. in Computer Science from the Asian Institute of Technology (AIT), Thailand. Currently, he is a Ph.D. candidate at AIT. His research interests lie in computer vision and machine learning.
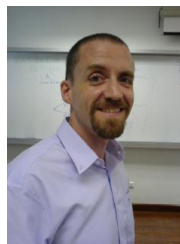
Matthew N. Dailey received the B.S. and M.S. in Computer Science from North Carolina State University and the Ph.D. in Computer Science and Cognitive Science from the University of California, San Diego. He spent two years as a Research Scientist with Vision Robotics Corporation of San Diego, CA USA and two years as a Lecturer

in the Computer Science and Information Technology
programs at Sirindhorn International Institute of Tech-
nology, Thammasat University, Thailand. In 2006, he
joined the Computer Science and Information Manage-
ment department at the Asian Institute of Technology,
Thailand. His research interests lie in machine learn-
ing, machine vision, robotics, systems security, and high
performance computing.