SUMMARY In this paper, we address the problem of recovering the camera radial distortion coefficients from one image. The approach that we propose uses a special kind of snakes called radial distortion snakes. Radial distortion snakes behave like conventional deformable contours, except that their behavior are globally connected via a consistent model of image radial distortion. Experiments show that radial distortion snakes are more robust and accurate than conventional snakes and manual point selection.

key words: Calibration, radial distortion, snakes.

1. Introduction

Most cameras having wide fields of view suffer from non-linear distortion due to simplified lens construction and lens imperfection. Applications that require 3-D modeling of large scenes (e.g., [1], [8], [15]) or image compositing over a large scene area (e.g., [10], [14]) typically use cameras with wide fields of view. Images taken with cameras having wide fields of view are usually distorted with the characteristic barreling effect. This barreling effect is usually removed by calibrating the camera and subsequently correcting the input image using the extracted distortion parameters.

In general, there are two forms of non-linear camera distortion, namely radial distortion and decentering distortion. For most cameras, distortion is caused primarily by radial distortion, which produces the characteristic barreling effect. In this paper, we address the problem of recovering the camera radial distortion coefficients and the principal point from one image. We present a new technique that uses what we call radial distortion snakes. Unlike conventional snakes, the behavior of radial distortion snakes are globally connected via a consistent model of image radial distortion.

In order to remove camera distortions through software manipulation, image features need to be first detected and located, either manually or automatically. The locations of these features are then used to fit the distortion model for parameter recovery and image correction. In many cases, camera parameter recovery is separate from the feature detection process. While this does not present a problem if the feature detection is correct, directly linking the feature detection to parameter estimation results in more robust recovery, especially in the presence of noise. The spirit of this work is very much linked to this philosophy.

1.1 Prior work

There has been significant work done on camera calibration, but many of them require a specific calibration pattern with known exact dimensions. There is a class of work done on calibrating the camera using the scene image or images themselves, and possibly taking advantage of special structures such as straight lines, parallel straight lines, perpendicular lines, and so on. One relevant work that belong to this class is that of Becker and Bove [1]. They use the minimum vanishing point dispersion constraint to estimate both radial and decentering lens distortion. The user has to manually group parallel lines together.

Brown [3] uses a number of parallel plumb lines to compute the radial distortion parameters using an iterative gradient-descent technique involving the first order Taylor’s expansion of the line fit function. The extraction of points on the plumb lines is very manual intensive. Swaminathan and Nayar [13] use a user-guided self-calibration approach. The distortion parameters are computed from user-picked points along projections of straight lines in the image.

Stein [12] uses point correspondences between multiple views to extract radial distortion coefficients. He uses epipolar and trilinear constraints and searches for the amount of radial distortion that minimizes the errors in these constraints.

Photogrammetry methods usually rely on using known calibration points or structures [2], [3], [5], [17]. Tsai [17] uses corners of regularly spaced boxes of known dimensions for full camera calibration. In a more flexible arrangement, Faig [5] requires that the points used only be coplanar, and that there are identified horizontal and vertical points among these points. Wei and Ma [18], on the other hand, use projective invariants to recover radial distortion coefficients.

The idea of active deformable contours, or snakes, was first described in [9]. Since then, there has been numerous papers on the applications and refinement of
snakes. Snakes has been applied to, for example, from tracking human facial features [19] and cells [4] to reconstruction of objects [16]. In all these cases, the snakes, some of which may be parameterized, work independently of each other. In this respect, their behaviour is local. In our proposed method, radial distortion snakes are globally parameterized, and they deform in a globally consistent manner. Radial distortion snakes react to strong edges, but they use a common set of parameters to dictate their reaction to these strong edges (thus exhibiting global consistency).

2. Finding the radial distortion parameters

We begin this section with a brief description of the lens distortion equation.

2.1 The radial distortion equation

The modeling of lens distortion can be found in [11]. In essence, there are two kinds of lens distortion, namely radial and tangential (or decentering) distortion. Each kind of distortion is represented by an infinite series, but generally, a small number is adequate.

We assume that the tangential distortion can be neglected and the principal point is the center of the image. The radial distortion equations are then

\[ x_u = x_d + x_d \sum_{l=1}^{\infty} \kappa_l R_d^l \]  
\[ y_u = y_d + y_d \sum_{l=1}^{\infty} \kappa_l R_d^l \]  

where \( \kappa \)'s are the radial distortion parameters, \((x_u, y_u)\) is the theoretical undistorted image point location, \((x_d, y_d)\) is the measured distorted image point location, and \(R_d = x_d^2 + y_d^2\).

In our approach, the user draws lines on the image that correspond to projections of straight 3-D lines. These drawn lines need not be exact. We then use snakes to search for the best-fit lines to extract radial distortion parameters. A direct approach would be to use normal snakes.

2.2 Using conventional snakes

For a conventional snake, an objective function that is typically minimized (for snake \( j \)) is of the type

\[ C_{s,j} = (1 - \lambda) C_{\text{edge},j} + \lambda C_{\text{smooth},j} \]

\[ = \sum_i \left[ (1 - \lambda) \nabla I(p_{ij})^2 + \lambda \rho(p_{ij})^2 \right] \]

where \( C_{\text{edge}} \) is the edge strength cost, \( C_{\text{smooth}} \) is the smoothness cost, \( p_{ij} \) is the \( i \)th point in the \( j \)th snake, \( \nabla I() \) is the intensity gradient, \( \rho() \) is the local curvature, and \( \lambda \) a constant. Here, each snake is independent.

In our implementation, the motion of the snakes is based on two factors: motion smoothness (due to external forces) and spatial smoothness (due to internal forces). Given the original configuration of a snake, a point on the snake \( p_j \) moves by the amount \( \delta p_j \) at each step given by

\[ \delta p_j = (1 - \lambda) \sum_{k \in N_j} \mu_{jk} \delta p_{\text{edge},k} + \lambda \sum_{k \in N_j} \mu'_{jk} (p_k - p_j) \]

where \( N_j \) and \( N'_j \) are the neighborhood of pixel at \( p_j \), including \( p_j \). \( \delta p_{\text{edge},i} \) is the computed motion of the \( i \)th point towards the nearest detectable edge, with its magnitude being inversely proportional to its local intensity gradient. \( \mu_{jk} \) and \( \mu'_{jk} \) are the respective neighborhood weights. In our implementation, \( \lambda = 0.5 \) and \( N_j = N'_j \), the radius of the neighborhood being 5 and the weights \( \mu_{jk} = \mu'_{jk} \) being \( \{1, 2, 4, 8, 16, 32, 16, 8, 4, 2, 1\} \).

Once the snakes have settled, the camera radial distortion parameters can then be recovered using a least-squares formulation.

2.3 Using radial distortion snakes

Using conventional snakes have the problem of getting stuck on wrong local minima. This problem can be reduced by imposing more structure on the snake—namely, the shape of the snake has to be consistent with the expected distortion of straight lines due to global radial distortion. For this reason, we call such snakes radial distortion snakes.

For radial distortion snakes, the cost to be minimized can be stated as

\[ C = \eta C_{\text{kappa}} + (1 - \eta) \sum_j C_{s,j} \]

where \( C_{s,j} \) is defined in (3) and \( C_{\text{kappa}} \) is the error in fitting the radial distortion model. \( \eta \) is a time-varying variable in our case.

The complexity of the objective function \( C \) can be reduced if we consider the fact that the effect of radial distortion is rotationally invariant about the principal point, ignoring asymmetric distortions due to tangential distortion and non-unit aspect ratio.

Our implementation to minimize \( C \) has the following steps:

1. For each snake, find the best fit line,

2. Rotate each snake about the principal point so that the rotated best fit line is horizontal. Let the angle of this rotation be \( \alpha_i \) for the \( i \)th snake.

3. Estimate best fit set of radial distortion parameters \( \kappa_1, ..., \kappa_L \) from the rotated set of lines (described shortly).

4. Find the expected rotated distorted point \( p'_j = (x_j', y_j') \), whose undistorted version lies on a horizontal line, i.e.,
where $\kappa_1, \ldots, \kappa_L$ are the radial distortion parameters. We also artificially incorporate image noise to study the effects of noise on the snakes. Real images with significant radial distortion are used to illustrate their use in practice. For all the experiments described in this section, we recover $\kappa_1$ and $\kappa_2$ only, i.e., we set $L = 2$. This is generally sufficient for low to moderately distorted images in practice. The time taken for images of resolution 512 × 480 take 10-15 seconds on a 400 MHz PC to extract just these radial distortion parameters. However, computing the principal point as well took between 5-7 minutes. Note that this timing excludes the time to draw the initial snakes. For comparison, manually selected points and lines are used “as is” in computing the radial distortion parameters. In addition, the manually drawn lines are used as initial configurations for both the conventional and radial distortion snakes. These lines are drawn much less carefully than the points to illustrate the robustness of the radial snakes.

3.1 Metric for accuracy

In evaluating the accuracy of the recovered radial distortion parameters, it does not make much sense to compare directly the values of $\kappa_i$’s for cases where $L > 1$. This is because significantly different sets of $\kappa_i$’s may give rise to visually similar corrected images. In addition, coming up with an error metric using only the actual and estimated sets of $\kappa_i$’s is not trivial. In such a situation, it is more reasonable to compute an image distance-based metric that is a function of the error in removing the distortion from the image using the actual and estimated sets of radial distortion parameters. A reasonable error measure is the RMS difference $E_{RMS}$ between the predicted and the estimated corrected coordinates (based on actual and estimated radial distortion parameters respectively). The error measure $E_{RMS}$ is given by

$$E_{RMS} = \sqrt{\frac{1}{HW} \sum_{r=-\frac{H}{2}}^{\frac{H}{2}} \sum_{c=-\frac{W}{2}}^{\frac{W}{2}} d_{rc}}$$

where

$$d_{rc} = \left( x_{rc}^{act} - x_{rc}^{est} \right)^2 + \left( y_{rc}^{act} - y_{rc}^{est} \right)^2$$

In the case where the principal point is known to be coincident with the image center,

$$E_{RMS} = \sqrt{\frac{1}{HW} \sum_{r=-\frac{H}{2}}^{\frac{H}{2}} \sum_{c=-\frac{W}{2}}^{\frac{W}{2}} [c^2 + r^2] \Delta_{rc}}$$

$$E_{RMS} = \sqrt{\frac{1}{HW} \sum_{r=-\frac{H}{2}}^{\frac{H}{2}} \sum_{c=-\frac{W}{2}}^{\frac{W}{2}} R_{d,rc} \sum_{l=1}^{L} \delta \kappa_l R_{d,rc}}$$

3. Results

In this section, we present results from both synthetic and real images. The synthetic images are used to verify the accuracy of the recovered radial distortion parameters. We also artificially incorporate image noise to study the effects of noise on the snakes. Real images with significant radial distortion are used to illustrate their use in practice. For all the experiments described in this section, we recover $\kappa_1$ and $\kappa_2$ only, i.e., we set $L = 2$. This is generally sufficient for low to moderately distorted images in practice. The time taken for images of resolution 512 × 480 take 10-15 seconds on a 400 MHz PC to extract just these radial distortion parameters. However, computing the principal point as well took between 5-7 minutes. Note that this timing excludes the time to draw the initial snakes. For comparison, manually selected points and lines are used “as is” in computing the radial distortion parameters. In addition, the manually drawn lines are used as initial configurations for both the conventional and radial distortion snakes. These lines are drawn much less carefully than the points to illustrate the robustness of the radial snakes.

3.1 Metric for accuracy

In evaluating the accuracy of the recovered radial distortion parameters, it does not make much sense to compare directly the values of $\kappa_i$’s for cases where $L > 1$. This is because significantly different sets of $\kappa_i$’s may give rise to visually similar corrected images. In addition, coming up with an error metric using only the actual and estimated sets of $\kappa_i$’s is not trivial. In such a situation, it is more reasonable to compute an image distance-based metric that is a function of the error in removing the distortion from the image using the actual and estimated sets of radial distortion parameters. A reasonable error measure is the RMS difference $E_{RMS}$ between the predicted and the estimated corrected coordinates (based on actual and estimated radial distortion parameters respectively). The error measure $E_{RMS}$ is given by

$$E_{RMS} = \sqrt{\frac{1}{HW} \sum_{r=-\frac{H}{2}}^{\frac{H}{2}} \sum_{c=-\frac{W}{2}}^{\frac{W}{2}} d_{rc}}$$

where

$$d_{rc} = \left( x_{rc}^{act} - x_{rc}^{est} \right)^2 + \left( y_{rc}^{act} - y_{rc}^{est} \right)^2$$

In the case where the principal point is known to be coincident with the image center,

$$E_{RMS} = \sqrt{\frac{1}{HW} \sum_{r=-\frac{H}{2}}^{\frac{H}{2}} \sum_{c=-\frac{W}{2}}^{\frac{W}{2}} [c^2 + r^2] \Delta_{rc}}$$

$$E_{RMS} = \sqrt{\frac{1}{HW} \sum_{r=-\frac{H}{2}}^{\frac{H}{2}} \sum_{c=-\frac{W}{2}}^{\frac{W}{2}} R_{d,rc} \sum_{l=1}^{L} \delta \kappa_l R_{d,rc}}$$

3. Results

In this section, we present results from both synthetic and real images. The synthetic images are used to verify
to the actual radial distortion parameters corresponding to image noise to see how it affects both the conventional radial distortion parameters. In particular, we vary the containing straight lines and distort them with known values respectively, and subscripts $u$ and $d$ to denote the corrected and distorted values respectively.

### 3.2 Experiments using synthetic images

In our first set of experiments, we use synthetic images containing straight lines and distort them with known radial distortion parameters. In addition, we vary the image noise to see how it affects both the conventional and radial distortion snake algorithms. In particular, the actual radial distortion parameters corresponding to $\kappa_1 = 10^{-6}$ and $\kappa_2 = 10^{-10}$ are applied to images with a resolution of 480 $\times$ 512. To reduce the complexity of analysis, we fix the principal point to be the image center. The gaussian image noise (specified by the standard deviation in intensity level) is varied from 0 to 100 intensity levels. The maximum intensity level is 255.

Fig. 1 shows the results for the case with no image noise using all three methods: manually picked points (b), conventional snakes (e), and radial distortion snakes (f). The original distorted image is shown in (a). The resulting corrected images shown in (c) and (g) look very similar in this case. Fig. 2 shows results for the same test image with an image noise of 100 intensity levels. Here the conventional snakes had converged to an incorrect solution due to the high level of intensity noise (see Fig. 2(d)). The radial distortion snakes, however, pooled their information to yield a more reasonable solution (see Fig. 2(f)).

We also ran a series of experiments to find out the general effect of image intensity noise on the performance of both conventional and radial distortion snakes. The manually picked points and lines are used as references and are kept constant regardless of image noise. The results of these experiments are shown in Fig.s 3 and 4. The graphs in Fig. 3 show the variation of the two extracted radial distortion parameters as a function of image noise, while Fig. 4 displays the change in RMS pixel error with image noise. It can be observed from these two figures that using the RMS distortion error is a more intuitive metric to determine the degree of accuracy in recovering radial distortion parameters. Note that factors such as discretization errors (in localizing edges, for example) and limited convergence may cause results to be inexact. These factors may be responsible for the discrepancy for the no-noise case in Fig. 4.

For low image noise levels, both snake algorithms exhibited reasonable robustness to image noise. However, the radial distortion snake algorithm is even more stable despite the presence of high image noise, in comparison to the conventional snake algorithm. Note that both snake algorithms have essentially the same implementation and uses the same edge gradient estimation and same maximum number of 200 steps or iterations. The difference with the radial distortion snake is that all of the radial distortion snakes are *globally connected* via a common set of estimated radial distortion parameters. As a frame of reference, the radial distortion parameters that are estimated based directly from the user drawn lines ($\kappa_1 = 3.22 \times 10^{-8}$ and $\kappa_2 = 1.14 \times 10^{-10}$) yielded an RMS pixel location error of 4.15.

### 3.3 Experiments using real images

A second series of experiments was conducted using real images. The first real image that we used is of an office. Fig.s 5, 6, and 7 show the results of using manually picked points, conventional snakes, and radial snakes, respectively. The radial distortion para-
Fig. 1 Synthetic image with no image noise: (a) Original image, (b) Manually selected points (299 points), (c) Corrected image from (b), (d) Manually drawn lines, (e) After applying conventional snake algorithm, (f) Corrected image from (e), (g) After applying radial distortion snake algorithm, (h) Corrected image from (g) (very similar to (f)).

Fig. 3 Graphs showing the effect of gaussian image noise (standard deviation in intensity level) on recovered radial distortion parameters: (Left) $\kappa_1$ (true value is $10^{-6}$), and (Right) $\kappa_2$ (true value is $10^{-10}$). Note that for the manual method, the points were placed on the image with no noise. Also, for the other two methods, the snakes were initialized on the image with no noise.

Fig. 4 Graph showing the effect of gaussian image noise (standard deviation in intensity level) on RMS distortion removal error $E_{RMS}$.

In Fig. 8, a different set of initial snake configuration was used on the same office image. This figure illustrates a situation where the radial distortion snakes appeared to have converged to a better local minimum than that of conventional snakes for the same snake initialization. This example shows that the radial distortion snakes are more tolerant to errors in snake initialization by the user.

Another example involving a real image is shown in Fig. 9, where we show the results of applying both conventional and radial distortion snakes. This scene is a more difficult one due to the relative sparseness of structurally straight and long lines within the camera viewing space. As a result, the recovery of the radial distortion parameters are more sensitive to errors in the extracted curves. This is evidenced by Fig. 9(d) for the conventional snake algorithm, where slight errors have resulted in significantly erroneous estimates in the radial distortion parameters. On the other hand,
Fig. 5  Office scene (manual): (a) Original distorted image, (b) Selected points, (c) Corrected image.

Fig. 6  Office scene (using conventional snakes): (a) Initial snake configuration, (b) Final snake configuration, (c) Corrected image. Note the relatively uneven snake to the left of image in (b).

Fig. 7  Office scene (using radial distortion snakes): (a) Initial snake configuration, (b) Final snake configuration, (c) Corrected image. Notice the smoother snakes compared to Fig. 6.

Fig. 8  Office scene (comparing the two snake implementations): (a) Initial snake configuration, and final snake configuration for (b) Conventional snakes, (c) Radial distortion snakes.
the radial distortion snake algorithm resulted in a visually improved corrected image. Our algorithm works for highly distorted images as well, as Fig. 10 shows.

![Figure 9](image1.png)

**Fig. 9** Another example: (a) Original image, (b) Initial snake configuration, and final snake configuration for (c) Conventional snakes, (e) Radial distortion snakes. The respective corrected images are (d) and (f). Note that the snakes are shown in black here.

Radial distortion snakes appear to have the effect of widening the range of convergence compared to conventional snakes (as exemplified by Fig. 8). In other words, a larger range of possible initial configurations is available for radial distortion snakes for correct convergence than for conventional snakes. It is reasonable to conjecture that radial distortion snakes have fewer false local minima, although this number will increase with the number of radial distortion parameters to be estimated. We base our conjecture on the experiments conducted.

An example of radial distortion snakes converging to a wrong local minima is shown in Fig. 11. Here we again show the results of applying both conventional and radial distortion snakes. As can be seen, due to the rather bad placing of the initial snakes (specifically the two most vertical ones), the snakes in both implementations converged to straddle different parallel edges, causing incorrect estimated radial distortion parameters. The weak edges corresponding to the edge of the whiteboard also contribute to the failure.

![Figure 10](image2.png)

**Fig. 10** An example with very significant distortion: (a) Original image with initial snake configuration, (b) Original image with final snake configuration, and (c) Corrected image. The snakes are shown in black.

![Figure 11](image3.png)

**Fig. 11** Snake failure example: (a) Original image, (b) Initial snake configuration, and final snake configuration for (c) Conventional snakes, (d) Radial distortion snakes. The snakes are shown in black.

4. Discussion

The direct method of manually picking discrete direct line points is the simplest to implement and understand, but it is the most burdensome to the user. In our implementation, the user has to be relatively careful in choosing the points. This is because the user-designated input locations are used directly in the radial distortion parameter estimation step. Note that in our implementation, the initial location of each point indicated is rounded to the position of the nearest pixel. One can magnify sections of the image in order to be able to place each point more accurately, but this is even more manual intensive. The alternative is to add the intermediate process of automatic local edge searching and location refinement, but this may pose a problem in an image of a complicated scene with many local edges.

It is clear from experiments that using the radial distortion snakes is better than using conventional snakes. We have demonstrated that the radial distortion snakes find best adaptation according to best
global fit to radial distortion parameters. They appear to be less prone to being trapped in bad local minima in comparison to conventional snakes. At every step, the radial distortion snakes act together to give an optimal estimate of the global radial distortion parameters and deform in a consistent manner in approaching edges in the image.

In comparison to the radial distortion snake, each conventional snake is locally adaptive and works independently of all the other snakes in the same image. They are not specialized, nor are they designed to be optimal to the task (in our case, the recovery of radial distortion parameters). This is clearly another demonstration of the benefit of incorporating global task knowledge directly in the early stages of the problem-solving algorithm. The concept of the radial distortion snake is very much in the same spirit as that of task-oriented vision [7].

5. Summary and future work

Images that are taken with a camera that has a reasonably wide field of view are usually distorted with the characteristic barrelling effect. This barrelling effect is primarily due to radial distortion, which in turn is usually caused by lens imperfections. Removing this defect is important in many imaging applications, from digital photography to parts inspection to 3-D modeling from images.

We have described radial distortion snakes as a mechanism to recover radial distortion parameters from a single image. Radial distortion snakes deform in concert based on a common radial distortion model. One direction for future work is to extend this work to estimate the principal point and tangential (or decentering) distortion parameters as well. Another area is to fully automate the process of determining radial distortion by edge detection and linking, followed by hypothesis and testing. A robust estimator may be used to reject outliers (e.g., RANSAC-like algorithm [6]).

References


Sing Bing Kang He received his Ph.D. in Robotics from Carnegie Mellon University, Pittsburgh, USA in 1994. He is currently a researcher at Microsoft Corporation where he is working on environment modeling from images. He had won an Outstanding Paper award at CVPR in 1991 for his work on Complex EGI, and the Best IEEE Robotics and Automation Transaction Paper award in 1997 for his human-to-robot hand mapping work.