Speeding Up and Performance Evaluation of a Fully Automatic Radial Distortion Compensation Algorithm for Driving Assistance Cameras

Yuta KANUKI†a, Student Member and Naoya OHTA†, Member

SUMMARY Recently, cameras are equipped on cars in order to assist their drivers. These cameras often have a severe radial distortion because of their wide view angle, and sometimes it is necessary to compensate it in a fully automatic way in the field. We have proposed such a method, which uses the entropy of the histogram of oriented gradient (HOG) to evaluate the goodness of the compensation. Its performance was satisfactory, but the computational burden was too heavy to be executed by drive assistance devices. In this report, we discuss a method to speed up the algorithm, and obtain a new light algorithm feasible for such devices. We also show more comprehensive performance evaluation results than those in the previous reports.

key words: radial distortion, driving cameras, wide angle cameras, histogram of oriented gradient, entropy

1. Introduction

Cameras used for driving assistance devices usually have a wide view angle, and because of that, their images often contain large radial distortion. It is sometimes necessary to correct the images so that they do not contain radial distortion.

We need to know distortion parameters of the cameras in order to correct the image distortion. Many methods have been proposed for this purpose, but most of them require a special calibration process. For instance, the methods [1], [2] implemented as the function calibrateCamera in OpenCV [3] require a process of imaging a checkerboard pattern. However, for driving assistance devices sold separately from cameras, for instance, this additional process should be eliminated because relying on the users to do this process is problematic.

Methods estimating radial distortion parameters without any special processes have also been proposed. Among them, the methods using straightness of lines in images fit the current purpose because scenes seen from a car contain many line structures. The basic idea here is, when the parameters are correctly estimated, the restored image contains the most straight lines. One way to evaluate the amount of straight lines is to detect edges in the image and apply the Hough transform to detect lines [4], [5]. However, this strategy is effective only when the image contains strong edge of lines [7]. Therefore, we proposed a different strategy to use entropy of the histogram of oriented gradient [6], [7], to which we refer as the previous reports in this report. This strategy works reasonably well for ordinary driving scenes, but the algorithm described in those reports requires too much computing cost. In this report, we discuss ways to make the algorithm lighter and more efficient to shorten the processing time.

We made several modifications on the algorithm, but the key modifications are the following two. The first one is to limit the order of the distortion model to one while we used up to the second-order parameter in the previous reports. To use a simpler model seems somewhat straightforward modification, but to find a model that is simple but still with enough performance has great practical importance. The second modification is to compute the image gradient of the compensated image without image warping. This made the algorithm several times faster. In addition, we conducted an applicability test of the algorithm using six different types of cameras. We have used one type of camera in the previous reports, so we give a more comprehensive test in this report.

2. Proposed Method

We describe the proposed method with emphasis on the differences from the previous method reported in [6] and [7].

2.1 Radial Distortion Model

In general, the radial distortion is modeled by the following equations (cf. pp.189–193 of Ref. [8]).

\[
\begin{align*}
x' &= \left( 1 + \sum_{i=1}^{n} \kappa_i r^{2i} \right) x \\
y' &= \left( 1 + \sum_{i=1}^{n} \kappa_i r^{2i} \right) y
\end{align*}
\]

where \( r = \sqrt{x^2 + y^2} \), \( \kappa_i (i = 1, 2, \cdots, n) \) is the distortion parameters, \((x, y)\) is the coordinate of a point in the original (distorted) image, and \((x', y')\) is that in the corrected (undistorted) image. The coordinate origin is set at the principal point, and we assume it at the center of an image.
In previous reports, we set the order \( n \) to two, and so we had parameters \( k_1 \) and \( k_2 \). However, we have tested many cameras sold as car driving recorders in the market including ones stated in the experiment section in this report, and we conclude that only \( k_1 \) is enough to correct distortions of these cameras. Therefore, we adopt only \( k_1 \) as the model parameters here, as follows.

\[
x' = (1 + k_1 r^2) x \\
y' = (1 + k_1 r^2) y
\]  

As stated above, this conclusion that only \( k_1 \) is enough relies on our empirical results, which is based on visual evaluation of the corrected images. In order to support this qualitative observation, we conducted a quantitative experiment as follows. We imaged a checkerboard pattern using two drive recorder cameras, Thanko D6048HDL (camera A) and Yupiteru DRY-mini 1 (camera B), and corrected the images using the one and the two parameter models. The best parameters were estimated so that the corner points of the checkerboard pattern aligned straight. The original images are shown in the left of Fig. 1, the corrected images with two parameters in the middle, and with one parameter in the right. Average disparities (errors) of the corner points from the best fitted straight lines are shown in Table 1, together with the estimated parameters. The average disparity values become zero if the distortion model perfectly matches the camera’s distortion, and so, it can be used as an index of the goodness of distortion models. The values of two types of models in Table 1 are almost the same, which supports our conclusion.

![Camera A (Thanko D6048HDL)](image1)

![Camera B (Yupiteru DRY-mini 1)](image2)

**Fig. 1** Comparison of distortion models having one and two parameters. Original images (left), corrected images with \( k_1 \) and \( k_2 \) (middle), and with \( k_1 \) only.

**Table 1** Average disparities from straight lines and estimated parameter values. The upper and lower rows are for the two parameter model and the one parameter model, respectively.

| camera | ave. disparity | \( k_1 \) | \( k_2 \) |
|--------|---------------|--------|--------|
| A      | 1.676 pixels  | 8.57\times10^{-7} | 1.66\times10^{-11} |
|        | 1.766 pixels  | 1.92\times10^{-6} | —      |
| B      | 2.030 pixels  | 2.07\times10^{-6} | —      |
|        | 2.032 pixels  | 2.35\times10^{-6} | —      |

In the proposed method, we use the entropy of the histogram of oriented gradient of an image as the measure of the distortion, and minimize it to find the correct value of the parameter \( k_1 \). Detailed explanations of the measure are found in the previous reports [6] and [7], but we briefly restate the computation procedure here.

The magnitude \( m(x', y') \) and the angle \( \theta(x', y') \) of the gradient of the corrected image \( I' \) at the point \((x', y')\) are defined by the next equations.

\[
m(x', y') = \sqrt{\left(\frac{\partial I'}{\partial x'}\right)^2 + \left(\frac{\partial I'}{\partial y'}\right)^2} \tag{5}
\]

\[
\theta(x', y') = \text{mod}\left(\tan^{-1}\left(\frac{\partial I'}{\partial y'} / \frac{\partial I'}{\partial x'}\right), 180\right) \tag{6}
\]

Here, the angles are represented in degree unit. The opposite gradient directions \( \theta \) and \( \theta + 180 \) are regarded as identical by the modulo operation in Eq. (6). The histogram \( h(k) \) is computed by the following equations.

\[
h(k) = \frac{1}{S} \sum_{(x, y) \in I} \left\{ \delta \left( k, \left\lfloor \frac{\theta(x', y')}{180} \right\rfloor \right) m(x', y') W \right\} \tag{7}
\]

\[
S = \sum_{(x, y) \in I} m(x', y') W \tag{8}
\]

\[
W = \begin{cases} 
3 \left( \frac{\theta(x', y') - 45}{45} \right)^2 + 1 & (0 \leq \theta(x', y') < 90) \\
3 \left( \frac{\theta(x', y') - 135}{45} \right)^2 + 1 & (90 \leq \theta(x', y') < 180) 
\end{cases} \tag{9}
\]

where \( l \) is the number of bins, which is set to 360 here, \( \delta(\cdot, \cdot) \) is Kronecker’s delta, which takes one when the both integer arguments are equal, otherwise zero, and \( \lfloor \cdot \rfloor \) is the floor function. \( S \) is a normalizing factor, and \( W \) is a weight depending upon the gradient angle \( \theta(x', y') \). The entropy \( E \) of the histogram is defined as

\[
E = \sum_{k=0}^{l-1} (-h(k) \log_2 h(k)). \tag{10}
\]

### 2.3 Image Gradient Computation without Image Warping

In the previous papers, the image gradient were computed on the actually warped image, but the warping process is time consuming. Therefore, in this report, we compute the gradient directly on the original image \( I \) and convert it to that on the corrected image \( I' \), using the Jacobian matrix, as follows.

\[
\begin{pmatrix}
\frac{\partial I'}{\partial x'} \\
\frac{\partial I'}{\partial y'}
\end{pmatrix}
= \begin{pmatrix}
\frac{\partial x}{\partial x'} & \frac{\partial x}{\partial y'} \\
\frac{\partial y}{\partial x'} & \frac{\partial y}{\partial y'}
\end{pmatrix}^{-1}
\begin{pmatrix}
\frac{\partial I}{\partial x} \\
\frac{\partial I}{\partial y}
\end{pmatrix} \tag{11}
\]

where

\[
\frac{\partial x}{\partial x'} = 1 + k_1 (3x^2 + y^2) 
\]
### Table 2  Image sets and used driving recorders.

| set | manufacturer | model      | res.   | # of im. |
|-----|--------------|------------|--------|---------|
| A   | Thanko      | D6048HDL   | 640x480| 355     |
| B   | Yupiteru    | DRY-mini1  | 640x480| 403     |
| C   | Transcend   | DrivePro200| 640x360| 300     |
| D   | Yupiteru    | DRY-FH51   | 640x360| 271     |
| E   | Yupiteru    | DRY-FH210  | 640x360| 200     |
| F   | Yupiteru    | DRY-mini1  | 640x360| 552     |
| G   | KENWOOD     | KNA-DR300  | 640x360| 163     |

### Table 3  Estimation results and execution times.

| set | $\kappa_1$          | time   | prv.1 | prv.2 |
|-----|----------------------|--------|-------|-------|
| A   | $1.9 \times 10^{-6}$ | 0.474sec | 1.795sec | 209.5sec |
| B   | $2.7 \times 10^{-6}$ | 0.278sec | 1.775sec | 207.5sec |
| C   | $1.7 \times 10^{-7}$ | 0.189sec | 1.330sec | 161.5sec |
| D   | $1.2 \times 10^{-7}$ | 0.484sec | 1.326sec | 163.4sec |
| E   | $2.1 \times 10^{-7}$ | 0.486sec | 1.355sec | 165.2sec |
| F   | $2.1 \times 10^{-7}$ | 0.533sec | 1.293sec | 156.5sec |
| G   | $1.7 \times 10^{-8}$ | 0.441sec | 1.339sec | 160.5sec |

2.4 Algorithm for Parameter Estimation

The algorithm for $\kappa_1$ estimation is as follows.

1. Compute the image gradient on the original image, and only if its magnitude is greater than a threshold (50), proceed the following steps.
2. Convert the gradient to that on the corrected image $I'$ using Eq. (11).
3. Compute the histogram $h(k)$ by Eq. (7), and then the entropy $E$ by Eq. (10).
4. Search for $\kappa_1$ that gives the minimum value of the entropy $E$, changing the value of $\kappa_1$.

In the first step, if the magnitude of the gradient is small, the following steps are canceled. This manipulation contributes speeding up much, and is impossible if the image is first warped.

In the final step, the search range of $\kappa_1$ is from $1.0 \times 10^{-8}$ to $9.0 \times 10^{-3}$. We search nine values in each order, that is, try $1.0 \times 10^{-8}$, $2.0 \times 10^{-8}$, ..., $9.0 \times 10^{-8}$, then $1.0 \times 10^{-7}$, $2.0 \times 10^{-7}$, ..., $9.0 \times 10^{-7}$, and so on up to $9.0 \times 10^{-3}$. After finishing the search in this accuracy, we try again ten values around the minimizer with ten times smaller step. Therefore, the estimated value of $\kappa_1$ has two significant decimal digits.

We apply the above procedures for several hundred images sampled from an image sequence with one second intervals, and make a histogram of the estimated $\kappa_1$’s. The final estimation value of $\kappa_1$ is the mode of the histogram.

3. Experiments

We used seven image sets taken by six different types of
cameras as shown in Table 2. Each image set was made from a continuous driving scene by sampling at one second intervals as stated in Sect. 2.4. The total image number is shown in the rightmost column labeled “# of im.”. Image sets A and B were made from our own image sequences, but for the others, we used video clips found in YouTube.

The estimated values of $k_1$ are shown in Table 3. The correctness of the estimated $k_1$ values was verified by whether originally straight lines were recovered straight in the corrected image. For image sets A and B, an original image and its corrected image are shown in Fig. 2. For the other sets, we cannot show images because they are from YouTube and have the copyright problem, but we report here the straight lines were recovered to the same level as for sets A and B. The histograms of $k_1$ are shown in Fig. 3. Clear peaks are observed in all histograms. The $k_1$ values in the parentheses for image sets A and B are the values estimated by the method described in Sect. 2.1, and these values can be regarded as the ground truth. Similar values are also estimated from the driving scenes. From these observations, we concluded the $k_1$ values were successfully estimated by the proposed method.

Computation times by a PC having an Intel i7-3632QM 2.2 GHz CPU under single thread execution are also shown in Table 3. The third column labeled “time” is for the algorithm in this report, the fifth column (prv.2) is for the algorithm in the previous reports, and the fourth column (prv.1) is for times where the previous computation scheme was used, but the distortion model and the minimizing process described in Sect. 2.1 were adopted. The figures are the average time per frame. Comparing the fifth (prv.2) and fourth (prv.1) columns, the effect of adopting the simpler model is drastic. It makes the computation time more than one hundred times shorter. The effect of omitting an image warping can be evaluated by comparing the third (time) and fourth (prv.1) columns. The best speedup rate 7.0 is observed for image set C, the worst 2.4 is for F, and the average is 4.0. These results show that the algorithm modifications proposed in this report are very effective and practical.

4. Conclusion

In this report, we made modifications on the automatic radial distortion compensation procedure we had previously proposed and obtained both satisfactory estimation results and practical computation speed. We also conducted an applicability test using seven image sequences taken by six different types of cameras. The distortions were successfully corrected by the proposed method.

References

[1] Z. Zhang, “A flexible new technique for camera calibration,” IEEE Trans. Pattern Anal. Mach. Intell., vol.22, no.11, pp.1330–1334, 2000.
[2] D.C. Brown, “Close-range camera calibration,” Photogrammetric Engineering, vol.37, pp.855–866, 1971.
[3] http://opencv.org/
[4] R. Cucchiara, C. Grana, A. Prati, and R. Vezzani, “A Hough transform-based method for radial lens distortion correction,” Proc. 12th International Conference on Image Analysis and Processing, pp.182–187, 2003.
[5] E. Rosten and R. Loveland, “Camera distortion self-calibration using the plumb-line constraint and minimal Hough entropy,” Machine Vision and Applications, vol.22, no.1, pp.77–85, 2011.
[6] Y. Kanuki, N. Ohta, and A. Nagai, “Automatic compensation of radial distortion by minimizing entropy of histogram of oriented gradients,” Proc. 2nd IAPR Asian Conference on Pattern Recognition, pp.912–916, 2013.
[7] Y. Kanuki, N. Ohta, and A. Nagai, “Automatic calibration of radial distortion by entropy of histogram of oriented gradients,” IEICE Trans. Inf. & Syst. (Japanese Edition), vol.J97-D, no.9, pp.1510–1518, Sept. 2014.
[8] R. Hartley and A. Zisserman, Multiple view geometry in computer vision, Cambridge University Press, Cambridge, 2004.