OPTICAL REVIEW Vol. 21, No. 2 (2014) 162–173

Real-Time Distortion Correction of Fish-Eye Lens Based on Bayer Image Signal

Shiming LAI*, Zhihui XIONG, Lidong CHEN, Xin TAN, and Maojun ZHANG

College of Information System and Management, National University of Defense Technology, Changsha, Hunan 410073, China

(Received September 9, 2013; Accepted December 28, 2013)

The imaging system based on a fish-eye lens generally has to correct the distortion of fish-eye images. The distortion correction based on the Bayer image signal is valuable, such as reducing the computation burden of image signal processing chips and providing a new imaging system structure of fish-eye lens. In this paper, a distortion correction method of fish-eye lens based on the Bayer image signal is proposed. Firstly, a distortion correction method that focuses on vertical straight lines and processing delay is proposed. Secondly, according to the correlation among color channels of the Bayer image, a novel Hermite interpolation method appropriate for Bayer image signal is proposed. Finally, a prototype system of fish-eye-lens-based imaging is established and the real-time field-programmable gate array (FPGA) implementation of the proposed method is demonstrated. The experiment demonstrates that the proposed distortion correction is not only characteristic of real-time processing and the smaller computation amount, but also applicable to embedded hardware. © 2014 The Japan Society of Applied Physics

Keywords: distortion correction, fish-eye lens, image signal processing, imaging system

1. Introduction

Since a fish-eye lens can view scenes within a wide angle (reaching or even exceeding 180°), they have been widely applied in the area of video surveillance. However, fish-eye images are inappropriate for viewing owing to their serious distortion, which may represent straight lines in the scenes as curves. Therefore, it is necessary to correct the fish-eye images into "visual" normal images by eliminating or reducing distortion, which is generally known as distortion correction of the fish-eye lens.

Although many studies on the distortion correction of the fish-eye lens have been reported, almost no research has taken the characteristics of the imaging system into account. General distortion correction methods suppose that the input and output are color images in RGB format or grey images. However, in imaging systems, the original signals collected and output by a complementary metal oxide semiconductor (CMOS) or charge-coupled device (CCD) imaging sensor are generally are Bayer image signals. In Bayer images, every pixel has only one value of green, red, and blue color channels. Among all pixels, 50% are green, while red and blue account for 25% each. Figure 1 shows the Bayer format image. To get the final RGB images, a series of processing (e.g., white balance, demosaicing, color correction, and gamma correction) must be conducted for the original Bayer image signal, which is generally known as image signal processing (ISP). In addition to the imaging sensor, the image signal processing chip is also an important component of the imaging system. A distortion correction based on the Bayer image signal that can be carried out before the ISP procedure has some advantages. For example, it can be carried out using independent hardware or be integrated into a CMOS sensor to output corrected Bayer images to ISP

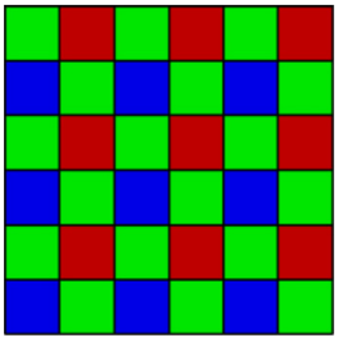


Fig. 1. (Color online) Bayer format image.

chips, thus reducing the computation burden of ISP chips and improving the flexibility of imaging system design.[1] However, the distortion correction based on the Bayer image signal faces some challenges. For example, it has a high demand of the processing delay of the image signal. Furthermore, interpolation of the Bayer image, which is required in distortion correction, cannot be well accomplished by using traditional image interpolation methods.

We aim to study how to conduct distortion correction of the fish-eye lens on a Bayer image signal. Since the imaging sensor outputs Bayer image signals successively, memory buffering and delayed processing of Bayer image signals are necessary in correction. Generally, directly applying the traditional distortion correction methods, which neglect the image processing delay, into Bayer image signal processing will bring a one-frame delay, thus influencing the real-time performance of the imaging system. We focus our attention on the image processing delay in our distortion correction method. Furthermore, the vertical distortion of vertical objects (e.g., buildings, pedestrians, etc.) in the fish-eye images severely impacts viewing, which is a key factor that

*E-mail address: shiming413@gmail.com

determines the correction method. Based on the processing delay and vertical straight lines, a new distortion correction method is proposed, which can process images without delay and correct vertical straight lines in the scenes. The image distortion correction requires image interpolation (that is, estimating the pixel value of the pixel with decimal coordinates). The traditional image interpolation algorithm cannot achieve a good interpolation effect when applied directly into Bayer image interpolation since it is only applicable to grey images or RGB images.[2] Based on the known strong correlation among color channels, the Hermite interpolation of a Bayer image is proposed, which is experimentally shown to be more accurate without increasing computation complexity compared with the traditional interpolation methods (e.g., cubic interpolation). To test the proposed algorithm, an experimental fish-eye imaging system is established. Different from the traditional fish-eye imaging system, a piece of a FPGA module is added between the CMOS sensor and the ISP chip. The distortion correction of the Bayer image signal is conducted on the FPGA module, thus enabling the ISP chip to receive the corrected Bayer image signal directly. The proposed algorithm is implemented on an FPGA board and the processing delay as well as the resource consumption are analyzed.

Section 2 gives a brief introduction to distortion correction of the fish-eye lens. Section 3 gives our correction method and the Hermite interpolation of the Bayer image. In Sect. 4, the experimental analysis of the Bayer image interpolation method is given firstly, and then the experimental fish-eye imaging system, FPGA implementation details, and analysis of correction delay and resource consumption are presented. Section 5 concludes the paper.

2. Brief Introduction to Distortion Correction of Fish-Eye Lens

Lens distortion correction is a basic research problem in the imaging field. Researchers mainly focus on the establishing and solving of a camera model, which represents the mapping between a 3D spherical surface and the 2D camera plane.[3–6] The distorted images with a normal viewing angle can be mapped on a 3D spherical surface through the camera model and then projected using the perspective projection to obtain orthoscopic images. This indicates that the camera model and perspective projection determine the correction method. However, perspective projection is inapplicable for the distortion correction of wide-angle images. If an image with a wide viewing angle is corrected by using the perspective projection, the image edge will be stretched severely. Theoretically, the perspective projection of a 180° fish-eye image will be an infinite image. In conclusion, although the camera model could represent the mapping relation between the fish-eye image and the 3D spherical surface, it is not able to provide a projection method to show the fish-eye image.

A complete 180° fish-eye image corresponds to a hemispherical surface. If longitude and latitude are taken as the coordinate system of the spherical surface, every pixel of the fish-eye image has a corresponding coordinate according to the fish-eye camera model. Therefore, the distortion correction of a fish-eye image can be viewed as the projection of the hemispherical surface onto a flat image plane. In fact, representing spherical maps onto a flat plane was an early challenge that cartographers faced. Scientists have proposed hundreds of projection methods,[7,8] including Mercator projection and Gauss–Kruger projection. The projection methods are differ from each other because of different demands or constraints. For example, Mercator projection characteristically represents loxodromes as straight segments, which conserve the angles with the meridians, so it is widely applied in drawing nautical and aviation charts. It is intuitive to represent fish-eye images by using cartographic projections. For instance, the open source software of panorama tools[9] applies different projection methods to map panoramic images, including fish-eye images and stitched panoramic images. There are also projection methods specially designed for representing panoramic images, such as Pannini.[10] However, none of these projection methods can eliminate all the distortions in fish-eye images of various different kinds of scenes. Carroll et al.,[11] Kopf et al.,[12] and Wei et al.[13] employed self-adaptive projection and calculated different projections according to different scene contents to adapt to different scenes and requirements and reduce the distortion of important image contents. The method of Carroll et al.[11] calculates projections with minimum distortion according to the image knowledge as horizontal straight lines, vertical straight lines and other straight lines marked by users on the image and the automatic detected face region. Such a self-adaptive projection method can correct most fish-eye images to obtain images appropriate for observation by the human eye. However, its application is significantly restricted owing to the requirement of manual marking of straight lines, which is impractical for a video capture system. Furthermore, corrected images output by self-adaptive projection have irregular shapes and often require abundant tailoring to obtain standard rectangular images, which will decrease image content. Memory buffering and processing delay have to be considered when correcting fish-eye image signals. No associated research has been reported yet according to our literature review.

Distortion correction will involve the calculation of pixels with decimal coordinates. Image interpolation is required for computing those pixels' values. Classical image interpolation includes bilinear interpolation and bicubic interpolation,[2,14] both of which are based on grey or color images. Direct application of them on a Bayer image will only provide a poor interpolation result. Note that the demosaicing technique in the imaging field is sometimes called Bayer image interpolation,[15] which focuses on the interpolation of the two absent colors on the Bayer image. This is different from our interpolation of Bayer images. To avoid confusion, the Bayer image interpolation referred to in this paper dose not mean demosaicing of the Bayer image. No research on Bayer image interpolation has been reported yet according to our literature review.

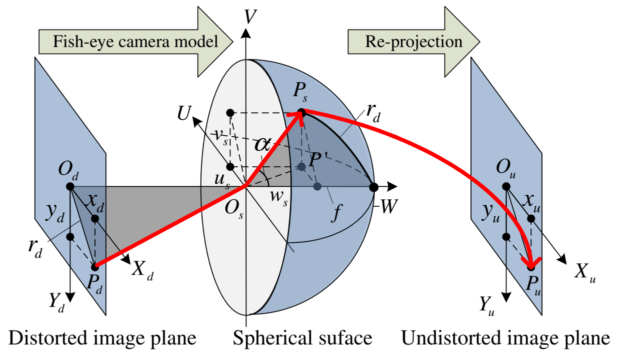


Fig. 2. (Color online) Distortion correction of fish-eye image includes two steps: firstly, project the fish-eye image onto a spherical surface by using a fish-eye camera model; secondly, re-project the spherical surface onto a 2D plane through a certain projection method to obtain the corrected image.

3. Distortion Correction based on Bayer Image Signal

3.1 Projection method

In Fig. 2, we suppose that the camera locates in the center O_s of a 3D sphere; then, the fish-eye lens with a 180° imaging angle can view all scenes within the range of a hemisphere. The longitude (θ) and latitude (φ) are taken as the spherical coordinate system. Considering the most common shooting direction $\overrightarrow{O_sW}$ (horizontal front), we can suppose both the longitude and latitude of the shooting direction as 0°. Let $P_s = (u_s, v_s, w_s)$ be a point on the spherical surface, $P_d = (x_d, y_d)$ is the projection of P_s on the fish-eye image plane (i.e., distorted image plane), and $P_u = (x_u, y_u)$ is the corresponding point on the undistorted image plane. Thus, a distortion correction method is equivalent to a conversion from (x_d, y_d) to (x_u, y_u).

The camera model of a fish-eye image in this paper is the most common equidistance model

$$\alpha = \frac{r_d}{f}, \tag{1}$$

where α is the angle between the incident ray O_sP_s and the principal axis O_sW, $r_d = \sqrt{x_d^2 + y_d^2}$ is the distance between the image point P_d and the principal point O_d (i.e., image center), and f is the equivalent focal length of the fish-eye lens. To better illustrate the problem, we let the radius of the sphere also be f, as shown in Fig. 2.

The coordinates of point P_s can be computed from (x_d, y_d) by using

$$\begin{cases} u_s = f \sin\alpha \cdot x_d/r_d, \\ v_s = f \sin\alpha \cdot y_d/r_d, \\ w_s = f \cos\alpha. \end{cases} \tag{2}$$

Let P' be the projection of P_s on the plane UO_sW. The spherical coordinates of P_s are

$$\begin{cases} \theta = \angle WO_sP' = \tan^{-1}(u_s/w_s), \\ \varphi = \angle P'O_sP_s = \sin^{-1}(v_s/f). \end{cases} \tag{3}$$

Thus, by combining Eqs. (1)–(3), the coordinates of the fish-eye image (x_d, y_d) can be converted into spherical coordinates through the following formula:

$$\begin{cases} \theta = \tan^{-1}\left[x_d \tan\left(\sqrt{x_d^2+y_d^2}/f\right)\Big/\sqrt{x_d^2+y_d^2}\right], \\ \varphi = \sin^{-1}\left[y_d \sin\left(\sqrt{x_d^2+y_d^2}/f\right)\Big/\sqrt{x_d^2+y_d^2}\right]. \end{cases} \tag{4}$$

Now, the problem becomes how to project P_s to P_u (i.e., the *re-projection* step in Fig. 2). The ideal projection method is the perspective projection, which enables all straight lines in the scene to remain as straight lines in the projected image. Since the corresponding projected image of a fish-eye image with a 180° viewing angle is theoretically an infinite image, perspective projection is inapplicable owing to its limited processing viewing angle. In fact, no practical projection method can ensure that all straight lines in a fish-eye image can be projected as straight lines. In a practical application, it is necessary to make a choice according to the specific requirements to obtain a satisfactory projection method. As a result, our projection method is developed from factors of vertical straight lines and processing delay.

3.1.1 Vertical straight lines

During horizontal shooting, the distortion of vertical objects (e.g., buildings and pedestrians) in the scene influences the visual effect significantly. To maintain vertical objects, the image's coordinates have to satisfy

$$x_u = f\theta. \tag{5}$$

Combining Eq. (5) with Eq. (1), then,

$$x_u = f \tan^{-1}\left[x_d \tan\left(\sqrt{x_d^2+y_d^2}/f\right)\Big/\sqrt{x_d^2+y_d^2}\right]. \tag{6}$$

3.1.2 Delay

The delay of the correction algorithm refers to the time difference between the output signal and the input signal, which is mainly determined by the distance between the output image coordinate and the input image coordinate. The distance between coordinates is mainly measured by the vertical coordinate since image signals are arranged in rows. To reduce the image signal processing delay, the corrected vertical coordinate shall be the same as that of the source image (fish-eye image):

$$y_u = y_d. \tag{7}$$

A new projection method is determined according to Eqs. (6) and (7). Figure 3 represents the longitude and latitude map of the corresponding fish-eye model and the proposed projection. The longitude and latitude ranges are $\theta \in (-\pi, \pi)$, $\varphi \in (-\pi/2, \pi/2)$, which represent a hemisphere. The intervals between both longitudes and latitudes are 10°.

Figure 4(a) is a fish-eye image and Figs. 4(b)–4(e) are the corrected images by the proposed new projection method and other existing correction methods. It can be seen from Fig. 4(e) that the proposed algorithm could correct the vertical straight lines. Also, our corrected image has no black edging, which means our algorithm can be applied directly without tailing and losing information content.

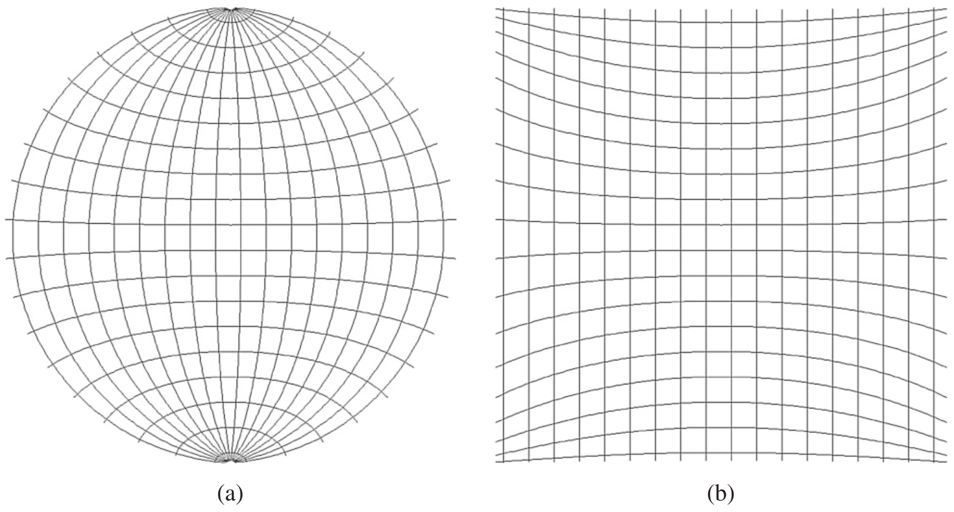


Fig. 3. Longitude and latitude maps of (a) fish-eye model and (b) proposed projection.

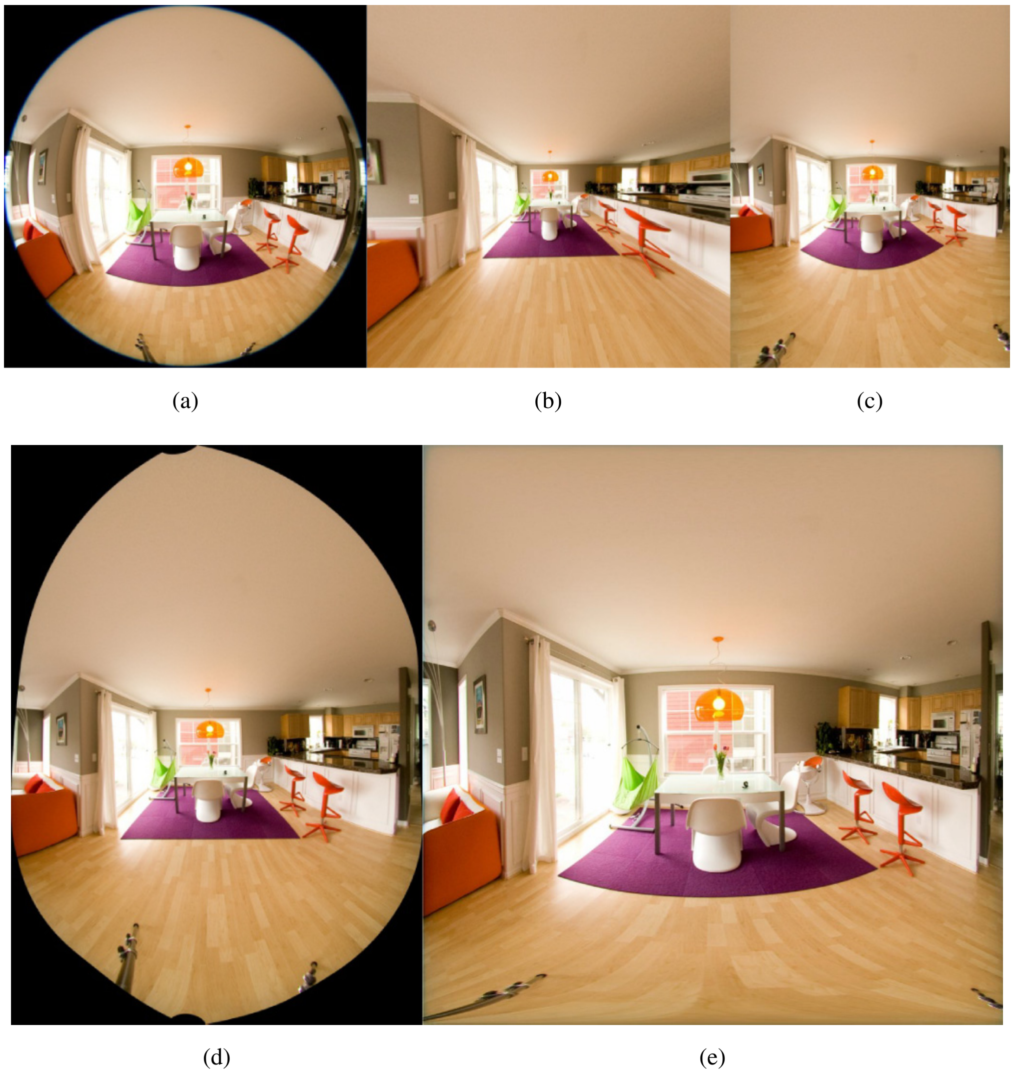
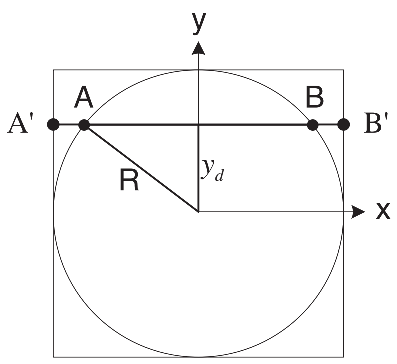
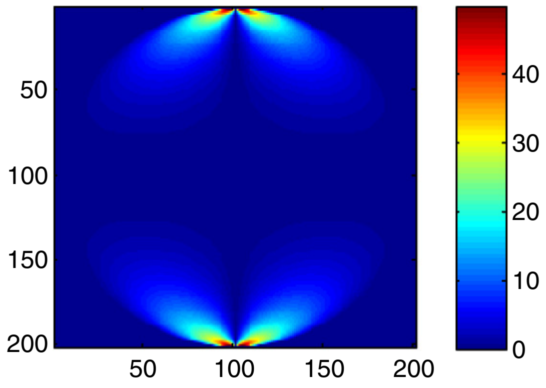


Fig. 4. (Color online) Fish-eye image correction effect of new projection method and other existing correction methods: (a) input fish-eye image, (b) perspective projection, (c) Mercator projection, (d) the method of Carroll et al., and (e) proposed projection method.

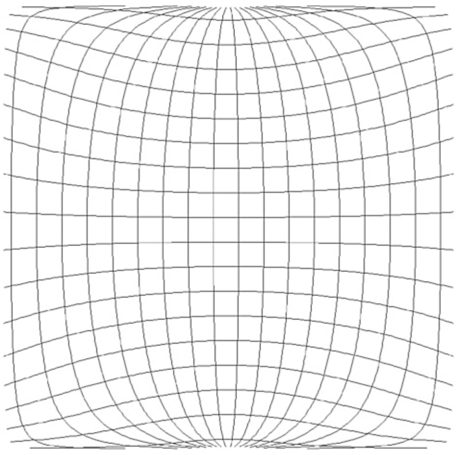


(a)

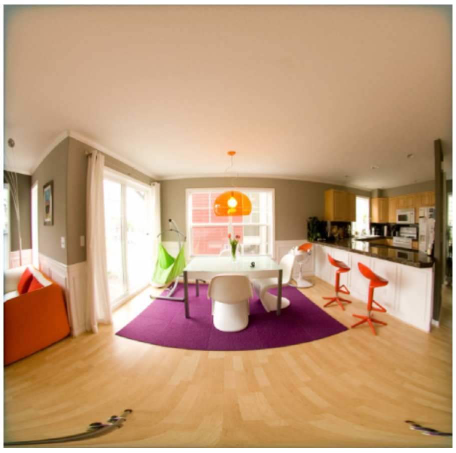


(b)

Fig. 5. (Color online) Simplified correction method: (a) correction method and (b) approximation error.



(a)



(b)

Fig. 6. (Color online) Simplified method: (a) longitude and latitude map and (b) correction effect.

3.2 Approximate correction method

The distortion correction calculation expressed by Eqs. (6) and (7) is relatively complicated, and involves calculation of the trigonometric function. These complicated calculations will consume some hardware resources when applied in an embedded system. For this reason, we apply a simplified approximate formula:

$$\begin{cases} x_u = kx_d \\ y_u = y_d \end{cases}, \tag{8}$$

where k is the stretching factor. We can assume that the size of the whole fish-eye image with a 180° viewing angle is 2R*2R. We use the following equation to compute k

$$k = \frac{R}{\sqrt{R^2 - y_d^2}}. \tag{9}$$

The distortion correction method described by Eqs. (8) and (9) is proximate to the projection model previously proposed. This can be interpreted by using Fig. 5(a). Firstly, the y-axis remains the same according to the requirement of no delay, that is, $y_u = y_d$. Secondly, all pixels in the same row stretch by using the stretching factor k, that is, $x_u = kx_d$. Be sure that the whole image is only filled with effective pixels after the stretching. In other words, all pixels on the straight line AB are stretched on the straight line $A'B'$. To analyze the approximation error, we compute an error image according to $\mathrm{err}(R+1+x_d, R+1+y_d) = |x_u^{(1)} - x_u^{(2)}|, x_d \in [-R, R], y_d \in [-R, R]$, where $x_u^{(1)}$ is computed by using Eq. (6) and $x_u^{(2)}$ is computed by using by Eq. (8). Figure 5(b) shows the error image when R is 100, and we can see that the approximation error is small in the middle part of the image. The average error is 2.38. If we only use the image with an aspect ratio of 16 : 9 in the middle part, the average error will be 0.80.

The correction effect and longitude and latitude of the simplified algorithm are shown in Fig. 6. Comparing Fig. 6 with Figs. 3(b) and 4(e), the simplified longitude and latitude show evident differences on the top and bottom parts, but a small difference in the middle part. The simplified algorithm can guarantee the correction effect of most image contents. The experimental image demonstrates that the corrected image basically can maintain vertical objects, as shown in Fig. 6(b). The simplified correction formula is very simple, only requiring one multiplication for every pixel and calculating one stretching factor for every row of pixels.

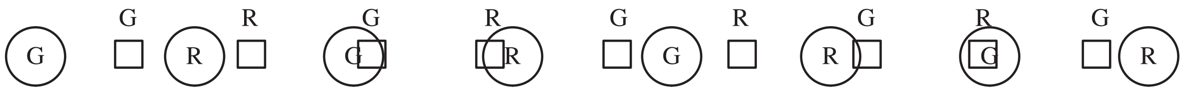


Fig. 7. One-dimensional Bayer image interpolation.

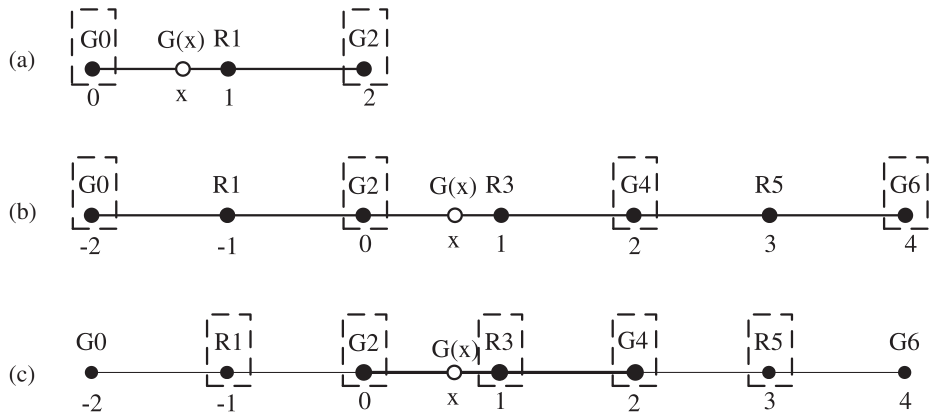


Fig. 8. Traditional interpolation methods and our Hermite interpolation method: (a) nearest neighbor interpolation and linear interpolation, (b) cubic interpolation, and (c) Hermite interpolation. $G(x)$ is the pixel values of interpolating G channel. The nearest neighbor interpolation and linear interpolation use the neighboring two G pixels for interpolation, while cubic interpolation uses four neighboring G pixels for interpolation. Hermite interpolation uses five neighboring pixels for interpolation, which involves both G and R channel information.

3.3 Hermite interpolation of Bayer image

The distortion correction involves the calculation of decimal pixel values, which is known as image interpolation. During horizontal distortion correction, two-dimensional Bayer image interpolation can be simplified into one-dimensional interpolation. Every row of the Bayer image contains pixel values of two channels. Take the G and R pixel row for example. G pixels and R pixels occurred alternatively in one pixel row. In Fig. 7, rings represent the pixels on the source image and boxes are the pixels on the corrected image. The corrected pixel values are also G pixels and R pixels alternatively. Therefore, one-dimensional Bayer image interpolation means to calculate the corrected G or R pixel value when given G, R, G, ..., R pixel values of the source image.

There are many image interpolations, mainly including the nearest neighbor interpolation, linear interpolation, and cubic interpolation.[14] Since general image interpolations are used to process two-dimensional images, the linear interpolation and cubic interpolation are commonly known as bilinear interpolation and bicubic interpolation. It is easy to deduce their interpolation formulas for one-dimensional Bayer images. To better introduce our Hermite interpolation, the nearest neighbor interpolation, linear interpolation, and cubic interpolation for a one-dimensional Bayer image are described.

(1) Nearest neighbor interpolation. In Fig. 8(a), suppose the pixel value $G(x)$ at x needs interpolation and $G0$, $R1$, and $G2$ are the pixel values of Positions 0, 1, and 2, respectively; then, $G(x)$ is determined by the nearest G pixel values. The interpolation formula is:

$$G(x) = \begin{cases} G0, & 0 \le x < 1, \\ G2, & 1 \le x < 2. \end{cases} \tag{10}$$

(2) Linear interpolation. In Fig. 8(a), linear interpolation, similar to the nearest neighbor interpolation, also involves the neighboring two G pixel values. Suppose $G(x)$ is the linear equation of x and given two equations ($G(0) = G0, G(2) = G2$) of two end points; then, it can get the linear interpolation equation of $G(x)$:

$$G(x) = \left(1 - \frac{x}{2}\right) \cdot G0 + \frac{x}{2} \cdot G2. \tag{11}$$

(3) Cubic interpolation. The cubic interpolation supposes the pixel value is the cubic function of the coordinate and involves four neighboring pixel values. Keys proposed the cubic interpolation equation.[14] Similarly, the cubic interpolation for a one-dimensional Bayer image supposes $G(x)$ is the cubic equation of x; then, according to Fig. 8(b), the cubic interpolation of $G(x)$ involves $G0$, $G2$, $G4$, and $G6$ (dotted boxes). Based on Keys' cubic interpolation equation, the cubic interpolation equation for a one-dimensional Bayer image can be inferred as

$$\begin{aligned} G(x) = {} & \left(-\frac{x^3}{16} + \frac{x^2}{4} - \frac{x}{4}\right) \cdot G0 + \left(\frac{3x^3}{16} - \frac{5x^2}{8} + 1\right) \\ & \times G2 + \left(-\frac{3x^3}{16} + \frac{x^2}{2} + \frac{x}{4}\right) \cdot G4 \\ & + \left(\frac{x^3}{16} - \frac{x^2}{8}\right) \cdot G6. \end{aligned} \tag{12}$$

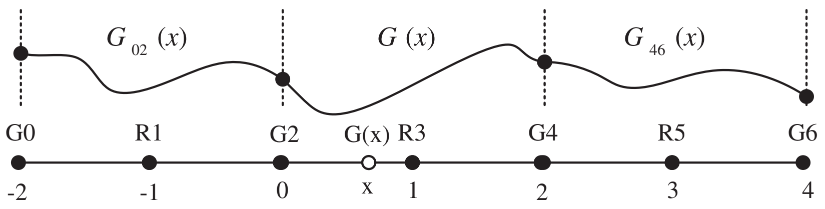


Fig. 9. Hermite interpolation of Bayer image (three intervals).

The distance between pixels of the same channel is widened owing to the cross arrangement of different color channels in the Bayer image. As a result, the pixel interval of Bayer image interpolation takes 2 unit lengths, while the pixel interval of traditional image interpolation is 1 unit length. An increased pixel interval will reduce the interpolation precision. The above-mentioned interpolations neglect the neighboring R pixels during the interpolation of G pixels. In fact, different color channels are strongly correlated with each other. Proper utilization of such correlation can contribute to higher interpolation precision. Therefore, the Hermite interpolation, which uses the correlation among different color channels for Bayer image interpolation, is proposed.

The Hermite interpolation algorithm divides the interpolation interval into several sub-intervals, which are used to solve the cubic polynomial curve of fitting variables. Different sub-intervals have different cubic polynomial curves. Although the fitting function is a piecewise function, it requires smooth connection between two neighboring sub-intervals. In other words, the cubic polynomial curve shall have the same function value and functional derivative value at the connection between two neighboring sub-intervals. According to Fig. 9, $G0, R1, G2, \dots, G6$ is a section of a Bayer image, which is divided into three intervals, namely, $[-2, 0]$, $[0, 2]$, and $[2, 4]$. The corresponding cubic polynomial curves of these three G channel intervals are $G_{02}(x)$, $G(x)$, and $G_{46}(x)$, respectively.

In view of the smooth connection of Hermite interpolation, the curves shall satisfy:

$$\begin{cases} G(0) = G_{02}(0), \\ G(2) = G_{46}(2), \\ G'(0) = G'_{02}(0), \\ G'(2) = G'_{46}(2). \end{cases} \tag{13}$$

According to the above two conditions, it can be known that the cubic polynomial curve equation (with four unknown parameters) can be solved if the function value and functional derivative value of two end points of the interval are known. In one-dimensional Bayer image interpolation, the function value of two interval end points of the G channel is known, whereas their derivative values are unknown and have to be calculated through other approaches.

The three color channels of the RGB image are closely correlated with each other. Generally speaking, if one pixel has a larger R pixel value, it also has larger G and B pixel values. Furthermore, the correlation of gradient image (convolution result of image and gradient operator) among these three color channels is more evident. Generally, different color channels have relatively similar high-frequency information. In natural images, the gradient values of these three color channels of most pixels can be viewed as equal. Such priori knowledge implies that the gradient value of one color channel can be estimated by the gradient information of other color channels, which is widely applied in demosaicing. On this basis, the derivative value of the R channel can be used to substitute the derivative value of the G channel approximately, thus providing a derivative equation for solving the Hermite equation.

Take the $G(x)$ in Fig. 9 for example. According to the priori hypothesis, we can obatin the following four equations:

$$\begin{cases} G(0) = G2, \\ G(2) = G4, \\ G'(0) = \dfrac{R3 - R1}{2}, \\ G'(2) = \dfrac{R5 - R3}{2}. \end{cases} \tag{14}$$

Suppose the equation of the $G(x)$ curve is:

$$G(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3. \tag{15}$$

Then, the known four equations are

$$\begin{cases} G(0) = a_0 = G2, \\ G(2) = a_0 + 2a_1 + 4a_2 + 8a_3 = G4, \\ G'(0) = a_1 = \dfrac{R3 - R1}{2}, \\ G'(2) = a_1 + 4a_2 + 12a_3 = \dfrac{R5 - R3}{2}. \end{cases} \tag{16}$$

Based on Eq. (16), we can obtain,

$$\begin{cases} a_0 = G2, \\ a_1 = \dfrac{R3 - R1}{2}, \\ a_2 = \dfrac{3(G4 - G2)}{4} + \dfrac{2R1 - R3 - R5}{4}, \\ a_3 = \dfrac{G2 - G4}{4} + \dfrac{R5 - R1}{8}. \end{cases} \tag{17}$$

Now, the Hermite interpolation formula of the one-dimensional Bayer image is known. $G(x)$ is solved from Eqs. (15) and (17). During the interpolation of $G(x)$, four parameters (a_0, a_1, a_2, a_3) shall be calculated first by substituting five pixel values ($R1$, $G2$, $R3$, $G4$, and $R5$) into Eq. (17). Then, the demanded $G(x)$ shall be calculated by substituting x coordinates into Eq. (15). In addition, the

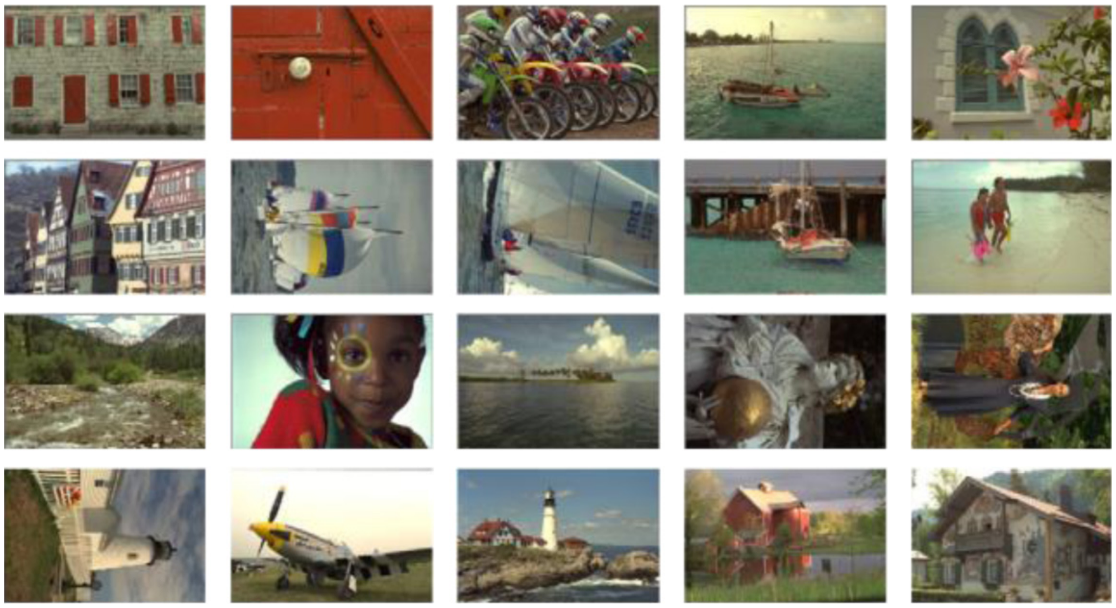


Fig. 10. (Color online) Test images from Kodak dataset.

solution of $G(x)$ also can be expressed by Eq. (18) which has a different computation mode. It calculates the weights of five interpolations (f_{R1},f_{G2},f_{R3},f_{G4},f_{R5}) according to x coordinates, based on which it calculates the weighted average of five pixel values ($R1$, $G2$, $R3$, $G4$, and $R5$) to obtain the needed $G(x)$. These two algorithms can be selected according to practical situations.

$$\begin{aligned} G(x) &= \left(-\frac{x^3}{8}+\frac{x^2}{2}-\frac{x}{2}\right)\cdot R1+\left(\frac{x^3}{4}-\frac{3x^2}{4}+1\right) \\ &\quad\times G2+\left(-\frac{x^2}{4}+\frac{x}{2}\right)\cdot R3+\left(-\frac{x^3}{4}+\frac{3x^2}{4}\right) \\ &\quad\times G4+\left(\frac{x^3}{8}-\frac{x^2}{4}\right)\cdot R5 \qquad (18) \\ &\triangleq f_{R1}(x)\cdot R1+f_{G2}(x)\cdot G2+f_{R3}(x)\cdot R3+f_{G4}(x) \\ &\quad\times G4+f_{R5}(x)\cdot R5. \end{aligned}$$

The Hermite interpolation formula based on Bayer priori information involves the nearest two G pixels and three R pixels. Hermite interpolation has several advantages. On one hand, traditional cubic interpolation completely neglects the neighboring R channel information, while Hermite interpolation makes use of the neighboring R pixels. Theoretically, rational utilization of pixel information of other channels can improve the interpolation precision to a large extent. In view of this principle, most demosaicing algorithms tried various means and achieved better interpolation effect. Therefore, Hermite interpolation shall have a higher interpolation precision compared than cubic interpolation. On the other hand, although both Hermite interpolation and cubic interpolation belong to cubic polynomial interpolation, Hermite interpolation involves five neighboring pixels, smaller than the interpolation range of cubic interpolation (eight neighboring pixels). This brings the Hermite interpolation advantages (e.g., small calculation amount).

4. Experimental Methods

In this section, firstly, we experimentally analyze Bayer image Hermite interpolation. Secondly, the proposed distortion correction method is verified on an FPGA board.

4.1 Experimental analysis of Bayer image interpolation

Among traditional image interpolations, the nearest neighbor interpolation is hardly used owing to its severe aliasing artifact. Cubic interpolation obtains higher accuracy than linear interpolation. Therefore, bicubic interpolation is commonly applied in image interpolation. To test the effect of Hermite interpolation on Bayer images, experiments of three interpolations (linear interpolation, cubic interpolation, and Hermite interpolation) are carried out under the same conditions for Bayer images. Taking the linear interpolation and cubic interpolation for reference, the proposed Hermite interpolation on a Bayer image is evaluated from the visual effect and peak signal-to-noise ratio (PSNR) of the image. The experiments use the Kodak dataset.[16] The dataset has 24 images (Fig. 10), in which many image details enable us to evaluate the effect of Hermite interpolation intuitively. For this reason, this dataset is the favorite dataset of researchers of demosaicing in evaluating the interpolation effect of various algorithms. Although the studied interpolation is different from that of demosaicing, their requirements and evaluation on the interpolation are similar.

To test the Bayer image interpolation in distortion correction, the images from the Kodak dataset are added with distortion and mosaic to obtain the distorted Bayer images. Based on these distorted Bayer images, three interpolations are applied for Bayer image interpolation in distortion correction, thus providing the corrected Bayer images. To evaluate and compare the interpolation effect of corrected Bayer images, on one hand, the corrected Bayer images are demosaiced[17] to obtain RGB images for the sake

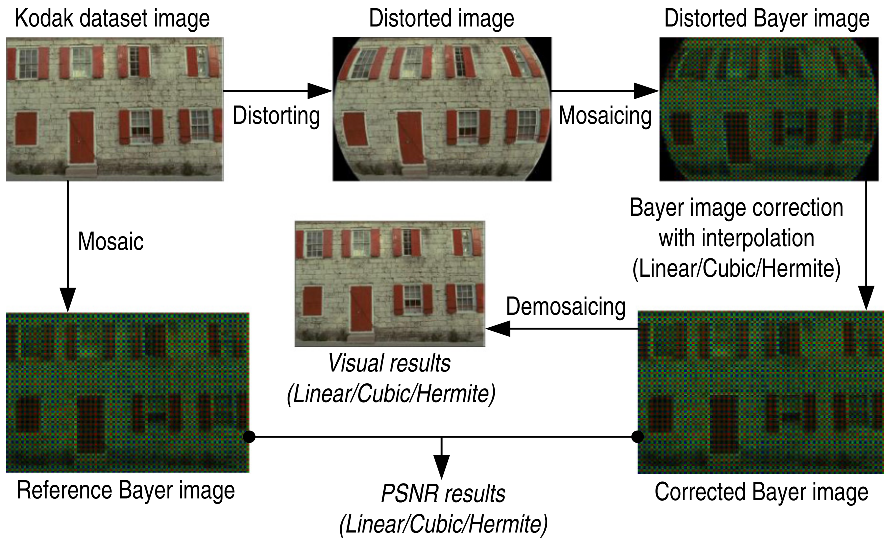

Kodak dataset image
Distorting
Distorted image
Mosaicing
Distorted Bayer image
Mosaic
Bayer image correction with interpolation (Linear/Cubic/Hermite)
Demosaicing
Visual results (Linear/Cubic/Hermite)
Reference Bayer image
PSNR results (Linear/Cubic/Hermite)
Corrected Bayer image

Fig. 11. (Color online) Comparison experiment of Bayer image interpolation effect.

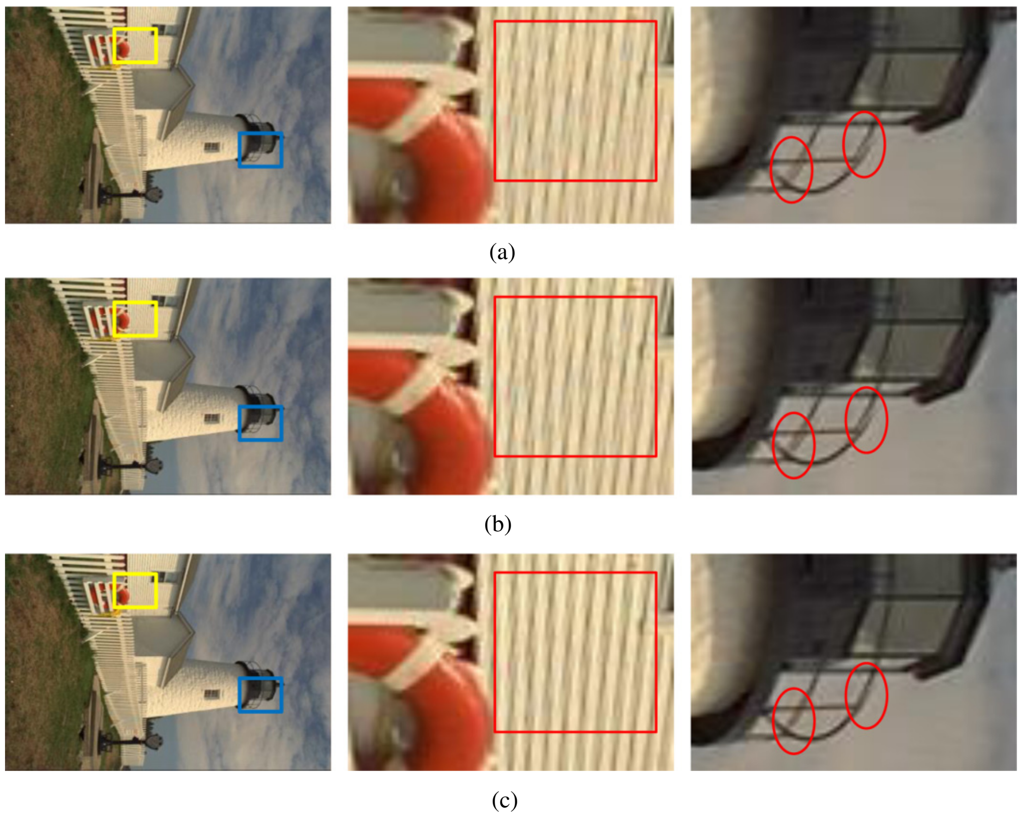


Fig. 12. (Color online) Visual results of interpolation effect comparison: (a) linear interpolation, (b) cubic interpolation, and (c) Hermite interpolation.

of intuitive evaluation and comparison of the interpolation effect from the perspective of the image effect. On the other hand, the original images from the Kodak dataset are added with mosaic to obtain Bayer images, which are used as a reference to calculate the PSNR of the corrected Bayer images, thus providing the quantitative comparison result of these three interpolations effects. Distortion correction applies the correction method described by Eqs. (8) and (9). A distortion adding equation can be obtained through the reciprocal transformation of the distortion correction equation. The basic experimental process is shown in Fig. 11.

Figure 12 represents the experimental results of an image. The three images of the first left column are the results of three interpolation methods and the images of the right two

Table 1. Testing PSNR (dB) of three interpolations. For all 24 testing images, the proposed Hermite interpolation achieves a higher PSNR compared with linear interpolation and cubic interpolation.

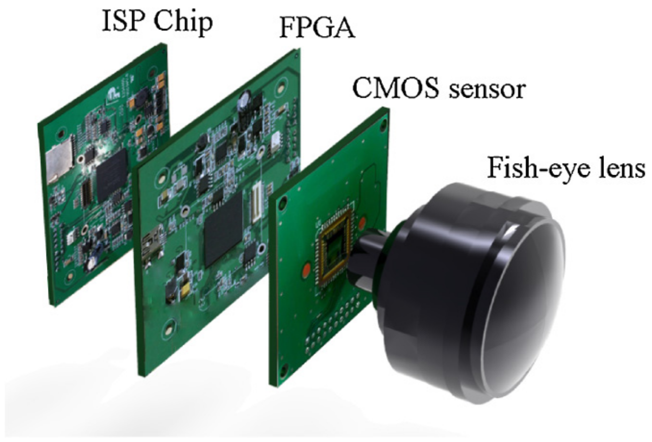
Method	Linear	Cubic	Hermite
1	28.3397	28.6264	31.1430
2	34.6072	34.7776	36.8320
3	37.2506	37.9944	40.4508
4	34.8381	35.3573	37.6632
5	27.7612	28.2699	30.9813
6	32.0905	32.4734	34.8706
7	35.2795	36.0861	38.6636
8	24.4214	24.6689	27.0478
9	33.9174	34.3457	36.7068
10	33.5049	34.1129	36.6462
11	32.3943	32.8305	35.2668
12	35.3424	35.8074	38.1037
13	26.4063	26.6969	29.1076
14	31.8545	32.3717	34.8103
15	31.5073	31.8176	33.9694
16	37.8926	38.5464	41.0116
17	32.9059	33.3530	35.8067
18	29.5375	29.8183	32.2079
19	29.9253	30.2310	32.3098
20	33.0676	33.5107	35.7215
21	31.6363	32.0334	34.5214
22	31.9587	32.3180	34.6598
23	36.8203	37.7203	40.1037
24	27.9189	28.2316	30.5792
Avg	32.1324	32.5833	34.9660

columns are the highlighted areas of the first left column images. Viewed from the images of the first column, the three interpolations accomplish correct correction interpolation and obtain correct images. The images of the second column are the amplified results of the yellow frame in the images of the first column, which reveal that linear interpolation and cubic interpolation produce evident color spots (within the red frame in the images), with the former producing the most color spots. Relatively speaking, Hermite interpolation produces smaller color spots. The images of the third column are the amplified results of the blue frame in the images of the first column, which reveal that linear interpolation and cubic interpolation destroy the image edges (red rings), while Hermite interpolation keeps the image edges almost completely. Hermite interpolation has obvious visual superiorities.

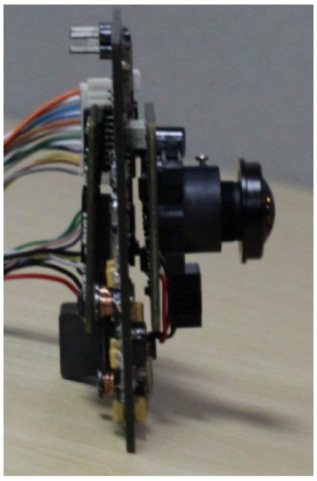
According to the experimental process in Fig. 11, the PSNR is calculated for the quantitative evaluation of the interpolation effect. The design formula of PSNR is

$$\mathrm{PSNR} = 10\log_{10}\frac{255^2}{\mathrm{MSE}}, \qquad (19)$$

where MSE is the mean square error of the image. The PSNR test result is listed in Table 1. Cubic interpolation has a higher PSNR than linear interpolation. The PSNR of Hermite interpolation is 2–3 dB higher than that of the previous two interpolations, indicating obvious superiorities.



(a)



(b)

Fig. 13. (Color online) The experimental system: (a) hardware structure and (b) experimental system.

4.2 *FPGA experimental verification*

An experimental system is designed to verify the proposed algorithm. The hardware structure of the system is shown in Fig. 13. Figure 13(a) describes the hardware structure, including the fish-eye lens, CMOS sensor, FPGA, and ISP chip. Figure 13(b) is the experimental system (side perspective). The image sensor is Micron's MT9P031, which is a 1/2.5″ CMOS image sensor with an active-pixel array of 2592H × 1944V (5 Mp). In our experiments, a windowing function is used to obtain a 1920H × 1080V × 30fps (1080p30fps) video, where the pixel number is 2 Mega (2 Mp). The fish-eye lens is Sunex's DSL215 miniature lens for a 1/2.5″ format imager, whose effective focal length is 1.55 mm and approximate horizontal fov is 186°. The FPGA is a Xilinx Spartan-6 LX25, a low-cost FPGA offering a good solution for cost-sensitive embedded applications. The ISP chip is Texas Instruments's TMS320DM368 digital media processor, which supports Bayer image signal input.

This system without the FPGA module is a common fish-eye imaging system. The fish-eye lens can capture lights within the 180° scene. The CMOS sensor converts the

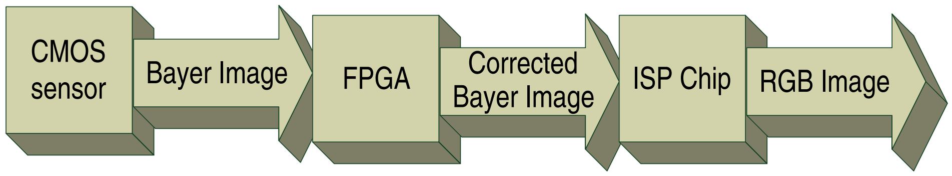


Fig. 14. (Color online) Image processing procedure.

Table 2. Hardware resource consumption of the algorithm.

Resources	Slices (3758 total)	Slices Reg (30064 total)	LUTs (15032 total)	BRAM/FIFO (52 total)	DSP48A1 (38 total)
Quantity (Percentage)	355 (9.4%)	723 (2.4%)	772 (5.1%)	2 (3.8%)	6 (15.8%)

optical image into the electronic signal to output the Bayer image signal. The ISP chip receives the Bayer image signal output by the CMOS sensor and then accomplish the ISP procedure to obtain and output (display, store, or transfer) standard RGB images (video). To verify the distortion correction of the Bayer image, an FPGA module is added behind the CMOS sensor in the imaging system, which makes the Bayer image signal output by the CMOS pass through the FPGA module before being delivered to the ISP chip. FPGA accomplishes the previously mentioned correction of fish-eye Bayer images. The image processing procedure is shown in Fig. 14, which demonstrates that the ISP chip receives the corrected Bayer images.

Since distortion correction only involves the x-axis, the correction of one row of pixels can be accomplished within the row of pixels. Because the fish-eye image is bilaterally symmetrical, the correction of the left half image only involves the left half image and the correction of the right half image only involves the right half image. As a result, we only have to use a Block Ram for one row of pixels in the FPGA to cache the successive Bayer image signal. The image correction starts after a half row of pixels have been cached. The corrected Bayer images are output to the ISP chip directly rather than being stored in the Block Ram. On one hand, the correction algorithm only requires a Block Ram for one row of pixels instead of abundant RAM (image cache). On the other hand, the correction algorithm will only bring a half row of pixels delay (less than 0.02 ms for an 1080P 30FPS video).

To save FPGA computing resources, the algorithm makes some fine turning during the correction on FPGA. When calculating the distortion correction coordinates using Eq. (8), the calculated stretching factors k of all rows are stored in the memory, which can be read when needed. This saves calculation of the stretching factor k (involving the square, square root, and other operations). As a result, the coordinate calculation of every pixel only requires one multiplication, consuming less computing resources.

Equation (18) is applied during the Hermite interpolation on one-dimensional Bayer images. Take the interpolation of the G channel at the GR pixel row as an example. After the interpolation coordinate x_d is calculated through Eq. (8), the five neighboring pixel values ($R1$, $G2$, $R3$, $G4$, and $R5$) can be located and their corresponding interpolation coordinates x can be calculated. It can be found from Eq. (18) that the weights ($f_{R1}(x), f_{G2}(x), f_{R3}(x), f_{G4}(x)$, and $f_{R5}(x)$) have to be calculated according to the x coordinate, which involves square and cubic operations. We discretized the domain of x ($x \in [0, 2]$) into 64 positions with equidistance. The five weights of these 64 positions were calculated in advance and stored in the memory. Two neighboring weights will be selected for linear interpolation to obatin the final weightes. Thus, the calculation of the weights is simplified into lookup and interpolation. Next, we conduct weighted average to the previous located five pixels according to the weights, thus finishing the Hermite interpolation and obtaining the final pixel value $G(x)$.

Table 2 presents the FPGA synthesis results for the distortion correction algorithm, including the number of slices, slices reg, look-up tables (LUTs), Block Ram (BRAM), and DSP48A1. The hardware resource utilization is reported for the Xilinx Spartan-6 LX25 device. The total resources of the device and percentages of the algorithm used are presented as well. The percentages show that our algorithm consumes very few resources. For example, on average, our algorithm consumes less than one-tenth of the total resources. This shows that our algorithm can be carried out using very low cost independent hardware or integrated into a CMOS sensor with few additional hardware resources.

5. Conclusions

In this paper, we propose a distortion correction of the fish-eye lens based on a Bayer image signal, which can be used in a real-time imaging system of the fish-eye lens and reduce distortions brought by the fish-eye lens. The experiment based on FPGA verifies this algorithm. In fact, the experimental system based on FPGA also implies a prototype system of fish-eye imaging product. For CMOS developers, such distortion correction can be integrated into the CMOS chip owing to its simple calculation and prompt data processing, thus saving the FPGA chip. This distortion correction method mainly focuses on the correction of vertical objects. However, the correction of non-vertical straight lines is also of great significance. Therefore, further study on a new distortion correction method with wider applicability is still necessary. Since this distortion correction only changes the horizontal axis, the interpolation only involves the one-dimensional image interpolation. However, with the variation of distortion correction, two-dimensional Bayer image interpolation will be challenged in future. Therefore, how to develop two-dimensional Hermite interpolation from one-dimensional Hermite interpolation also deserves detailed investigation.

Acknowledgment

This work was supported in part by the National Natural Science Foundation (NSFC) of China under Grant Nos. 61175006 and 61175015.

References

1) B. Tang and J. E. Crenshaw: U.S. Patent 20110141321 A1 (2009).
2) P. Thévenaz, T. Blu, and M. Unser: in *Handbook of Medical Imaging, Processing and Analysis*, ed. I. N. Bankman (Academic Press, San Diego, CA, 2000) p. 393.
3) A. Basu and S. Licardie: Pattern Recognition Lett. **16** (1995) 433.
4) F. Devernay and O. Faugeras: Mach. Vision Appl. **13** (2001) 14.
5) J. Kannala and S. S. Brandt: IEEE Trans. Pattern Anal. Mach. Intell. **28** (2006) 1335.
6) J. Wang, F. Shi, J. Zhang, and Y. Liu: Pattern Recognition **41** (2008) 607.
7) J. P. Snyder: *Flattening the Earth: Two Thousand Years of Map Projections* (University of Chicago Press, Chicago, IL, 1997) p. 1.
8) J. P. Snyder: *Map Projections—A Working Manual* (USGPO, NW Washington, D.C., 1987) p. 1.
9) Web [http://panotools.sourceforge.net/].
10) T. K. Sharpless, B. Postle, and D. M. German: Proc. 6th Int. Conf. Computational Aesthetics in Graphics, Visualization and Imaging, 2010, p. 9.
11) R. Carroll, M. Agrawal, and A. Agarwala: ACM Trans. Graphics **28** (2009) 43.
12) J. Kopf, D. Lischinski, O. Deussen, D. Cohen-Or, and M. Cohen: Comput. Graphics Forum, 2009, p. 1083.
13) J. Wei, C. F. Li, S. M. Hu, R. R. Martin, and C. L. Tai: IEEE Trans. Visualization Comput. Graphics **18** (2012) 1771.
14) R. Keys: IEEE Trans. Acoust. Speech Signal Process. **29** (1981) 1153.
15) D. Menon and G. Calvagno: Signal Process. Image Commun. **26** (2011) 518.
16) Web [http://www.cipr.rpi.edu/resource/stills/kodak.html].
17) Y. M. Lu, M. Karzand, and M. Vetterli: IEEE Trans. Image Process. **19** (2010) 2085.