An Efficient Neuron-Based Camera Distortion Correction System

Ching-Han Chen, Tun-Kai Yao, Zi-Hong Li, and Chia-Ming Kuo

Department of Computer Science & Information Engineering, National Central University
No.300, Jhongda Rd., Jhongli City, Taoyuan County 32001, Taiwan

pierre@csie.ncu.edu.tw
965402013@cc.ncu.edu.tw
zihongli0207@gmail.com

Abstract— **This study proposes an efficient system architecture for camera distortion correction, in which a neuron-based camera distortion corrector (NCDC) is applied to rapidly correct various camera and lens distortions and manufacturing flaws in economical cameras. Compared to traditional camera models for correcting camera and lens distortions, for which more than two types of models are used, the NCDC uses one neural model to correct geometric distortions and asymmetric manufacturing defects.**

The NCDC consists of a front-end that calculates back-map coordinates in distortion image space and a back-end that reads camera distortion image pixels from memory to interpolate a 24-bit color pixel corrected image. In a complex camera system with field-programmable gate array (FPGA), the back-end uses a burst mode and multi-port access architecture to fetch the image pixel from a high-latency memory, which increases the number of correction frames per second (CFPS) of the camera corrector. Moreover, a development board using high-speed memory was designed to increase the camera correction efficiency. The results show that the CFPS of an NCDC back-end with the proposed architecture is over 19.39x higher than professional solutions.

Keywords— **back propagation; camera distortion; distortion correction; FPGA; geometric distortion; neural network**

I. INTRODUCTION

Cameras are indispensable in most computer, communication, and consumer electronics products, especially photographic, vehicle electronics, surveillance, and numerous human interaction applications. Precision and the visible range are crucial for measurements in the medical field [1].

To capture an image with a greater visible range, the convex or concave lens of a wide-angle camera collects more light from captured objects. The collected light projects onto the fixed-size image-sensor array of the camera. However, the refracted angle of the light within the convex lens increases from the center to the edges, distorting the captured image at the rear of the convex lens. This barrel-type spatial distortion by the convex lens is traditionally corrected by a multi-piece lens group within a specific distortion range. This type of optical structure is large and costly to manufacture.

However, low-cost cameras have several types of manufacturing flaws in their lenses, holders, and sensor boards. These cameras consist of various convex and concave lenses and an image sensor. The center of all lenses and the image sensor must lie along the same optical axis. As shown in the structural diagram in

Fig. 1, two types of manufacturing defects can be observed between the lens holder and the sensor chip die in the camera module, in addition to the geometric and wide-angle distortions of the camera.

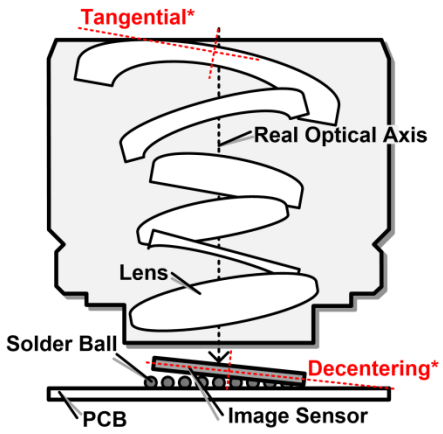


Fig. 1. Manufacturing flaws in camera sensor and lenses.

To correct for the low quality of cameras, Brown [2] presented a lens distortion model that includes the concept of decentering lens distortion. Weng [3] addressed the decentering and thin prism distortions in cameras, mathematically modeled these errors, and used a calibration board to obtain the distortion pattern used to calculate the parameters of the lens distortion model. These researchers used various software-implemented mathematical models to correct camera distortions.

For correcting wide-angle lens distortions in real time, Asari [4] presented a similar distortion polynomial model based on a nonlinear least squares curve fitting (NLSCF) scheme to obtain the coefficients of the distortion polynomial. In contrast to the wide-angle lens distortion model by Weng [3], the Asari polynomial model is easier to implement with very-large-scale integration (VLSI) architecture. Studies [5-9] recently proposed hardware accelerator designs based on the Asari method. However, low-cost cameras have additional distortions that are not corrected by the VLSI architecture; therefore, high-resolution videos captured with wide-angle

cameras cannot be corrected in real time when used for medical imaging [1] or other measuring purposes. Therefore, Chen [10] proposed an efficient method for correcting camera distortion based on the VLSI architecture, in which a neural camera distortion model is applied to correct various lens and manufacturing flaws of low-cost cameras rapidly. The neural corrector in [10] can produce over 300×10^6 back-map coordinates per second, which can process videos with resolutions of over 4096×2048 pixels.

The back-map coordinates in the distortion image space (DIS) are float coordinates that require four pixels on adjacent coordinates to interpolate a new pixel in the correction image space (CIS). In [4, 6-10], the researchers assumed that the interpolator can immediately fetch four pixels from memory to calculate the new pixel in CIS. However, if the correction system uses fast static random-access memory (SRAM) to store a camera-captured distortion image, the memory reader requires a clock cycle to fetch a pixel. The size of the captured image is over 2MB because the resolution of the camera is over 1920×1080 pixels. An SRAM chip is a costly device with small memory size. Therefore, the general system uses synchronous dynamic random-access memory (SDRAM) to store the distortion image. Because the back-map coordinates are not continuous between the previous and subsequent coordinates, the memory reader requires more clock cycles to fetch a pixel from SDRAM without a burst mode.

Chen [5] addressed the memory access problem in the VLSI architecture with SRAM, and proposed a time multiplexed method that shares part of the hardware circuit to reduce costs and throughput. The back-map generator in [5] produces a back-map coordinate for each of the four clock cycles. In the intermission time of the back-map generator, the interpolator can fetch four pixels from SRAM. However, the memory access action in an integrated system is more complex. As the interpolator fetches the pixels from memory, the camera control writes the next captured image to memory and the interpolated pixels are simultaneously written to memory.

This study proposes back-end architecture of an NCDC in an integrated system that consists of a camera subsystem, universal serial bus (USB) 2.0 subsystem, microprocessor unit (MPU) subsystem, and NCDC subsystem to efficiently use bandwidth and the space of the DDR SDRAM, which can use the generated back-map coordinates of the front-end NCDC to interpolate the pixels of the corrected image. Finally, the proposed system can rapidly correct camera geometric distortions and asymmetric manufacturing defects.

II. Camera Distortion Correction

A. Traditional Camera Distortion Models

With the real camera and lens distortions, the general perspective projection does not make the transition from world space to image space perfectly. To model the projection of the camera, Tsai [11] proposed several linear relationships between points in a 3D space. The projections in an image plane are established based on the pinhole camera model (PCM). In the Tsai camera model, the ideal wide-angle lens distortion was established.

Figure 2 shows a perspective projection model of a pinhole camera. O_c is the center of the image plane, and O_w is defined as the center of the world space. O_f is the center of the lens, which is also the center of the perspective projection model.

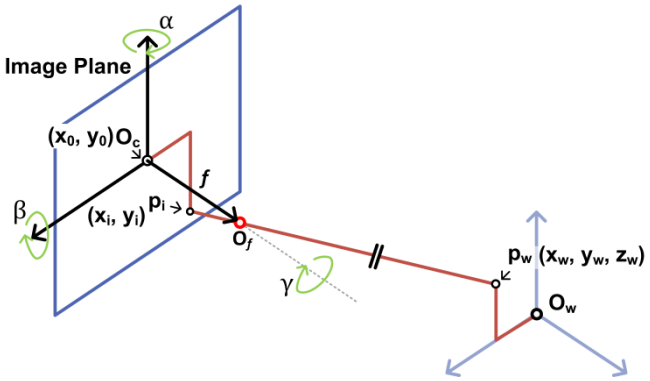


Fig. 2. Perspective projection of pinhole camera model.

p_w is an object point in the 3D world space, which is located at $[x_w;\ y_w;\ z_w]^T$, p_i is a projection point in 2D image-plane space, which is located at $[x_i;\ y_i]^T$. The mapping relationship of the world-space point p_w to its image point p_i with the PCM is provided by

$$\lambda p_i = C[R|t]p_w \tag{1}$$

where λ is the scaling factor of the projection size on the image-plane space. C is defined as an intrinsic parameter of the PCM presented in

$$\mathrm{C} = \begin{bmatrix} f\delta_x & s & u_0 \\ 0 & f\delta_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where $[x_c;\ y_c]^T$ are the coordinates of the camera lens center, f is defined as the focal distance, δ_x and δ_y are a fixed ratio between the width and the height of the pixel unit on the image sensor, respectively, and s is defined as the skew factor, which is zero for modern image sensors.

The matrix [R|t] is defined as an extrinsic parameter of the PCM, which is provided by

$$[\mathrm{R|t}] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \tag{3}$$

where t is a translation matrix, and R is a rotation matrix built on three Euler angles (α, β, γ), which is provided by

$$R = R_x(\alpha)R_y(\beta)R_z(\gamma) \tag{4}$$

The image plane rotates around the x-, y-, and z-axes with respective rotation angles α, β, and γ, which can be described by

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \quad (5)$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \quad (6)$$

$$R_z(\gamma) = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (7)$$

In (3), the translation matrix t is inserted in the back of the matrix R because the relationship mapping of (1) uses homogeneous coordinates to express the coordinates between the world space and the image space, which can be written as

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f\delta_x & s & u_0 \\ 0 & f\delta_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \quad (8)$$

In the PCM, other than the geometry distortion from the perspective projection model, nonlinear distortion is the most important form of lens aberration. Because a real camera uses a convex or concave lens to refract light from the object, the accuracy of the lens affects the angle of the refracted light in the lens. The surface at the periphery of the convex or concave lens has more curvature, which disturbs the linear translation result of (8).

To simplify the wide-angle lens correction method, Asari [4, 8, 9, 12] used a curve polynomial to model the radial lens distortion, as follows:

$$\rho' = \sum_{n=0}^{N} a_n \rho_d \quad (9)$$

where $\rho_d = \sqrt{\tilde{x}_d^2 + \tilde{y}_d^2}$, $\tilde{x}_d = x_d - x_0$ and $\tilde{y}_d = y_d - y_0$. $[x_d; \quad y_d]^T$ is a point in the DIS, a_n is the coefficient of the curve polynomial, and N is the order of the curve polynomial. The corresponding magnitude ρ_d is the distance from the lens center to a point in the DIS, and ρ' is the distance from the lens center to a point in the CIS.

The approach by Asari translates the lens distortion model from the Cartesian to the polar coordinates, and assumes that the lens distortion is a purely radial model, and that the point of the angle θ in the corrected space and the point of the angle θ' in the distortion space is the same. θ is given by

$$\theta = tan^{-1}\left(\frac{y_d - y_0}{x_d - x_0}\right) \quad (10)$$

Fig. 3 shows the polynomial back-map (PBM) process by Asari. Before the PBM, the corrected image space (x, y) of the Cartesian coordinate system uses (10) to translate to the polar coordinate system (ρ, θ). The translation requires the distortion center to obtain the magnitude and the angle in the polar coordinate system. If the distortion center (x_c, y_c) is inaccurate, PBM correction generates substantial errors. Therefore, the distortion center of the PBM is generated from the average of the estimated centers by using the described method.

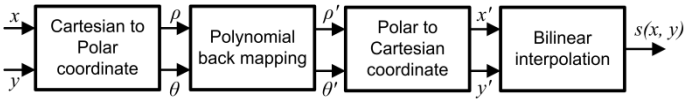


Fig. 3. Asari process for wide-angle lens distortion correction.

The point $[x'; \quad y']^T$ in the CIS is obtained from the polar coordinates $[\rho'; \quad \theta']^T$ in the CIS, which are provided by

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x_0 + \rho' \cos\theta' \\ y_0 + \rho' \sin\theta' \end{bmatrix} \quad (11)$$

Because the 2D coordinates translate to 1D coordinates in [4], the computation complexity is lower than in other research, which is useful for implementing the hardware accelerator for wide-angle correction [5, 6, 8, 9, 12].

However, the camera is not a perfectly manufactured product. In lower-quality cameras, Brown [2] proposed that the lens model is decomposed into two types of distortion: radial and tangential lens distortion. $[x'; \quad y']^T$ are points in the DIS, which are provided by

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x + x_r + x_t \\ y + y_r + y_t \end{bmatrix} \quad (12)$$

where $[x; \quad y]^T$ is a point in the CIS, x_r and y_r are the radial distortion vectors, and x_t and y_t are the tangential distortion vectors.

Radial lens distortion models more refraction angles from the lens center to the lens periphery for the convex or concave lens, which is described as

$$\begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} \tilde{x}_d(k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6 + \cdots) \\ \tilde{y}_d(k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6 + \cdots) \end{bmatrix} \quad (13)$$

where $\tilde{x}_d = x - x_0$, $\tilde{y}_d = y - y_0$, and $r_d = \sqrt{\tilde{x}_d^2 + \tilde{y}_d^2}$. $k_1, k_2 \ldots$ are the coefficients in the polynomial of the radial lens distortion model, which is used to change the curvature of the curve polynomial in the radial lens distortion model for various lenses.

In low-cost cameras, the image sensor is not parallel to the wide-angle lens, which is a manufacturing flaw. This causes the radial lens distortion to simultaneously follow tangential

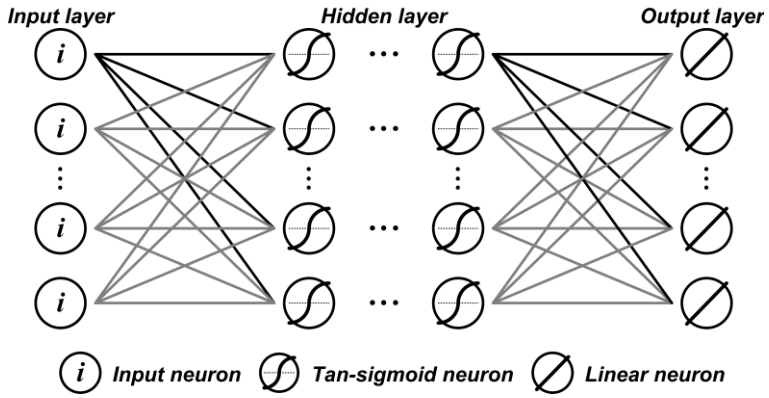


Fig. 4. Multi-layer feed-forward neural network.

distortion. To correct this distortion, Brown [2] modeled the decentering distortion, as expressed by

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} (p_1(r_d^2 + 2\tilde{x}_d^2) + 2p_2\tilde{x}_d\,\tilde{y}_d\,)(1 + p_3 r_d^2 + \cdots) \\ (2p_1\tilde{x}_d\,\tilde{y}_d\, + p_2(r_d^2 + 2\tilde{y}_d^2))(1 + p_3 r_d^2 + \cdots) \end{bmatrix} \quad (14)$$

where $p_1, p_2 \ldots$ are the coefficients in the polynomial of the tangential lens distortion model. For the wide-angle lens, (13) and (14) use higher-order terms to improve the correction accuracy.

However, the real low-price camera still has many other manufacturing flaws that are not modeled. Besides geometric and wide-angle distortion of the camera,

Fig. 1 shows the structure of a real camera module, indicating two types of manufacturing flaws between the lens holder and the chip die of the camera module.

The optical mechanism of a low-cost camera also has several manufacturing flaws. The convex lens of the wide-angle lens made by injection modeling is an asymmetrical convex lens, and a wide-angle lens, which has a more complex optical character, including a multi-piece lens group. Thus, a radial polynomial cannot model the complex optical character of asymmetrical barrel distortion. When the error from the manufacturing process accumulates, the optical center of the wide-angle lens diverges from the image sensor center. Correction of the error introduced by this manufacturing flaw requires more complex mathematical models to model the non-uniform distortion. A real distortion surface of a 120° low-price wide-angle lens in

Fig. 5 shows complex and non-uniform characteristics.

The images captured by a camera with a low-quality lens include horizontal, vertical, rotational, lens, and optical-mechanism errors. Manufacturing flaws cannot be predicted, and not every camera has the same flaws. However, if the correction uses complex mathematical models to correct all of the distortion in the camera, the computational time becomes extremely large, and several precise calibrations are required.

B. Neuron-Based Camera Distortion Corrector

Fig. 6 shows the distortion-correction process proposed by Chen [10]. The NCDC consists of off-line calibration and forward correction processes. In the off-line calibration process, the calibration dots are detected from the calibration image captured using a wide-angle camera. The training stage uses the detected calibration dots to train the neural network with back propagation. After the off-line process, the NCDC can independently produce the back-map coordinates; that is, the pixel location of CIS in the DIS. The back-map coordinates in the DIS are float coordinates that require four pixels on adjacent coordinates to interpolate the pixels in CIS.

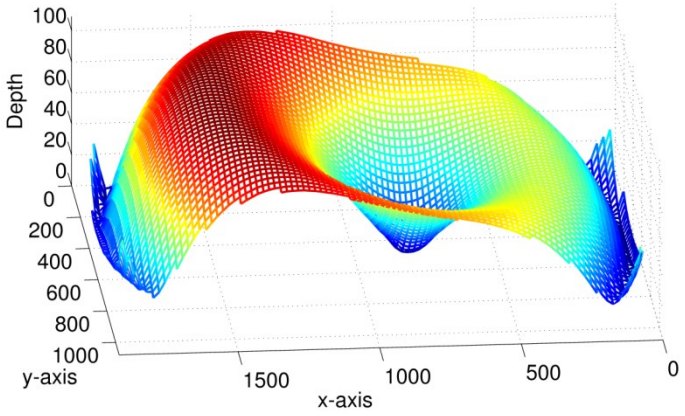


Fig. 5. Calculated distortion depth of the 120° wide-angle lens.

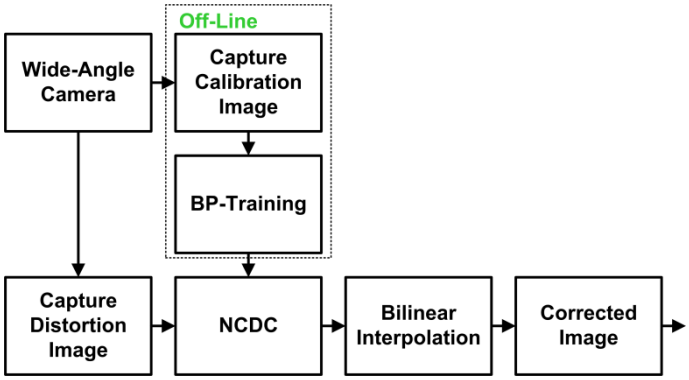


Fig. 6. NCDC process for wide-angle lens distortion correction.

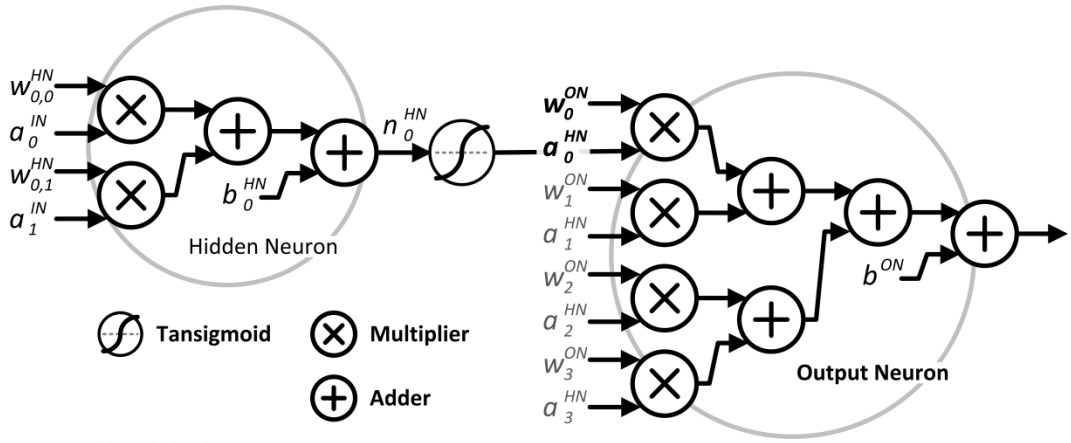


Fig. 8. Neural processor arithmetic hardware.

The backpropagation neural network (BPNN) consists of the feed-forward stage, which is the MFFNN, and the error backpropagation stage. Because most problems in the real world are nonlinear, the MFFNN uses a multilayered structure to enhance the adjustability of nonlinear output results. The MFFNN learns from the supervised learning method, which is an error backpropagation (EBP) [13] method for error-correction learning. The EBP method trains and modifies the weights and biases of the MFFNN.

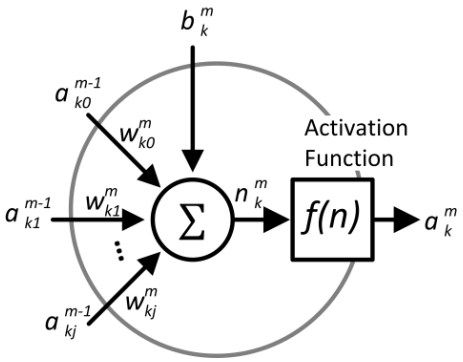


Fig. 7. A neuron of the BPNN.

Fig. 4 shows a general MFFNN, where layers are numbered from 0 to M. The neurons of the hidden layer are numbered from 0 to HN. The network training of the backpropagation method includes the feed-forward stage and the EBP stage.

In the feed-forward stage, the training patterns are sequentially inputted into the neurons from the input layer, and the input layer does not perform any operation. The calculated results propagate from the lower layer $(m-1)^{th}$ to the upper layer m^{th} via the neuron connection.

Fig. 7 shows a neuron, which consists of the multiply accumulator and the activation function.

Equation (15) is the multiply accumulator operation, which sums the weight of the upper neurons and the bias.

$$n_k^m = \sum_{j=0}^{HN_{m-1}} a_j^{m-1} w_{kj}^m + b_k^m \tag{15}$$

where $j < k$, and $m = 0\ to\ M$, w_{kj}^m is defined as the weight that corresponds to the connection from neuron j in the $(m-1)^{th}$ layer to k in the m^{th} layer, and b_k^m is defined as the bias of a neuron, which corresponds to neuron k in the m^{th} layer. Initialized values of the weights and the biases of the network are random numbers between 1 and -1. The neuron output a_k^m is activated from the weighted sum n_k^m in (16).

$$a_k^m = f(n_k^m) \tag{16}$$

where $k = 1\ to\ HN$. In
Fig. 4, the neurons of the hidden layer use the tan-sigmoid function (17) to activate the weighted sum, and γ is the scaling factor of the tan-sigmoid function that controls the curve rate.

$$f(n)_{tansig} = \frac{2}{1+e^{-\gamma n}} - 1 \tag{17}$$

The neurons of the output layer use a linear function (18) to activate the weighted sum of the hidden neurons, and the activated results of the output neurons are the output of the MLP neural network.

$$f(n)_{linear} = n \tag{18}$$

The EBP stage modifies the weights and biases of the neural network according to the error-correction rule. When m is M in (19), the expected value d_k minus the neuron output of the output layer obtained is the output error of the neural network. The network error propagates back from the output layer to the hidden layer.

$$\varepsilon_k^m = \begin{cases} d_k - a_k^m, & m = M \\ \sum_{j=1}^{HN_{m+1}} w_{kj}^{m+1} \delta_j^{m+1}, & m = 0 \sim (M-1) \end{cases} \tag{19}$$

The gradient δ_k^m uses (20) to calculate the weighted neuron error of the hidden layer.

$$\delta_k^m = \varepsilon_k^m f'(n_k^m) \tag{20}$$

The modification vector Δw_{kj}^m is calculated using (21), and the correction rate σ is defined as the learning rate, which is a scaling factor that limits the modification vector of the error correction in each iteration.

$$\Delta w_{kj}^m = \sigma \delta_k^m a_k^m \tag{21}$$

After calculating all the modification vectors Δw_{kj}^m, (22) updates the neuron weights of the neural network.

$$w_{kj}^m = \Delta w_{kj}^m + w_{kj}^m \tag{22}$$

The neuron bias modification method is similar to that of neuron weight modification. In the training period, all training patterns use the feed-forward operation to calculate the results, after which the backpropagation operation estimates the errors between the results and the target patterns. Based on these estimated errors, the weights and biases of the NN are modified by (22). The BPNN continues training with EBP until stopping conditions are met. After completing the EBP-based training, the BPNN stops the backpropagation and uses only feed-forward to back-map the camera distortion surface.

Fig. 8 shows the arithmetic hardware of a neural processor for the NCDC that consists of hidden neurons with two inputs and output neurons with four inputs. Because the input data of the correction NN are 2D coordinates (x, y), the hidden layer in the MFFNN consists of four neurons with two inputs. To operate the two-input neurons at a high frequency, the hidden neuron is designed using pipeline architecture. After the input value is multiplied by the weight, the pipe registers store these results. Next, the multiplied results and the bias stored in the next-stage pipe register are summed to generate the net signal. The MFFNN in this study used the register to store the descriptor of the NN that controls the weights and biases of the hidden neurons in the NN. In the memory mapping of the bus system, the bus interface operates the descriptor register to write a value to register or read a value by the microprocessor.

The arithmetic structure of the output neuron is similar to that of the two-input neuron. The pipe registers store the multiplied results of $a_{0\sim3}^{HN}$ and $w_{0\sim3}^{ON}$. The two multiplied results in each pipe register are summed by an adder, and these summed results are stored in the second-stage pipe register. The third-stage pipe register then receives the sum of the added results of the second stage in addition to the bias. Because a neuron connected to the next stage is an easy and simple way, the input value of the neuron is limited to a range between 1 and -1. Thus, the neurons can comprise any structure in the NN for serving diverse purposes.

When a wide-angle lens is switched to another camera, the NN can use the same neuron to correct the distortion. The input pixel coordinates of the new camera only require normalizing to a range between 1 and -1. The NN uses the new normalized data to train the weights and biases for the new camera. The normalized design of the neurons is a reusable structure, and for various applications the NN redesigns only the normalization arithmetic.

The arithmetic operation in the neuron uses a fixed point for calculations. The input and output widths of neurons in the NN for camera correction are 24 bits. The normalized value can use the complete 24-bit width of the multiplier and the adder in the neuron, which can calculate the output most precisely. When the weights and biases are controlled at a fixed-point value within the 24-bit width, the limited range can ensure that the multiplication and summing operations in the neuron do not overflow. Thus, the output range of the neuron is limited to a fixed range, which simplifies the effective operating range of the tan-sigmoid arithmetic.

C. Tan-Sigmoid Acceleration

Tan-sigmoid, which is the most critical function in the NN of VLSI implementation, consists of an exponential and a divider shown in (17). The exponential function of the tan-sigmoid is difficult to generate using the hardware arithmetic. In this study, the CORDIC was used to implement the hardware arithmetic of the exponential function. However, the CORDIC cannot directly generate the value of the exponential because it is operated in rotation mode to generate the results of $cosh(\theta)$ and $sinh(\theta)$. In (23), the exponential value of θ is the summed result of $cosh(\theta)$ and $sinh(\theta)$.

$$e^{\theta} = cosh(\theta) + sinh(\theta) \tag{23}$$

Equation (24) is the iterative function of the CORDIC that is based on a pseudo-rotation method. To generate the values of $cosh(\theta)$ and $sinh(\theta)$, the CORDIC is operated in rotation mode to calculate the following hyperbolic function:

$$\begin{cases} x_{k+1} = x_k - m\delta_k y_k 2^{-k} \\ y_{k+1} = y_k + \delta_k x_k 2^{-k} \\ z_{k+1} = z_k - \delta_k \varepsilon_k \end{cases} \tag{24}$$

where $m = -1$ for the hyperbolic function. In rotation mode, δ_k in (25) uses the sign of z_{k+1} to determine whether the next iteration is a positive or negative rotation.

$$\delta_k = \begin{cases} -1, & if\ z_k < 0 \\ 1, & otherwise. \end{cases} \tag{25}$$

where $k > 0$. In (26), ε_k is the rotating angle in the k stage.

$$\varepsilon_k = tanh^{-1}(2^{-k}) \tag{26}$$

where $k > 0$. After several iterative operations, z_{k+1} almost reaches 0. x_{k+1} and y_{k+1} converge on a hyperbolic term that can be written as

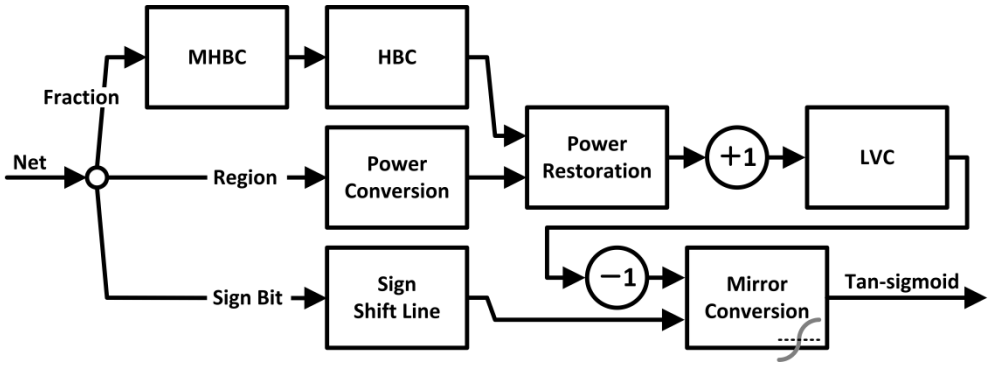


Fig. 9. The tan-sigmoid arithmetic structure for the MMBRC.

$$\begin{cases} x_{k+1} = K_h\,(x_0 cosh(z_0) + y_0 sinh(z_0) \\ y_{k+1} = K_h\,(y_0 cosh(z_0) + x_0\, sinh(z_0) \end{cases} \quad (27)$$

where $k > 0$. K_h, which is produced from each pseudo-rotation operation, is given by

$$K_h = \prod_{i=0}^{N} \sqrt{1 - 2^{-2i}} \quad (28)$$

where N is defined as the number of pseudo-rotation operations. K_h is a constant when N is greater than 25.

To expand the operating range of the HBC, most studies based on the CORDIC [14, 15] divide the operating range of the EXP into several regions. Asari [15] divided z_0 in (23) into integer and fraction regions, and preprocessed the integer results of the EXP stored in the LUT. The expanded $exp(-z_0)$ is calculated from

$$exp(-z_0) = exp(-z_{integer}) * exp(-z_{fraction}) \quad (29)$$

where $-z_{integer}$ is the negative integer part of $-z_0$, and $-z_{fraction}$ is the fraction part of $-z_0$ that is $0 \le -z_{fraction} < -1$. The HBC in [15] is used to generate $exp(-z_{fraction})$, and the preprocessing $exp(-z_{integer})$ multiplied by the result of the HBC is $exp(-z_0)$, which effectively expanded by more than ± 1.

Fig. 10 presents the mean squared error (MSE) results of three fixed points, which shows that when the width of the fixed point is more than 20 bits, the increased ratio of the calculation precise is less than the increased ratio from the 18-bit to 20-bit width. Based on this result, the $exp(-z_{fraction})$ of the tan-sigmoid in this study involved using a 20-bit fixed point for implementation.

In (17), after calculating the exponential, the result must be inversed. However, if a divider is used to inverse the exponential result, the hardware arithmetic comprises a large area to implement the divider. To use less area to implement the inverse arithmetic, in this study a linear CORDIC arithmetic was employed instead of the divider.

Fig. 11 shows the results of the tan-sigmoid calculated using a linear vector CORDIC (LVC) and a general divider (DIV), with the operation of the two methods at a fixed point domain.

Mix Modified HBC (MMHBC) [16] is used to generate more accurate and steady results than [15]. Fig. 13 presents the output characteristics of the tan-sigmoid obtained using the MMHBC and LVC from -15 to 15; these results are almost the same as the standard tan-sigmoid derived using a floating point.

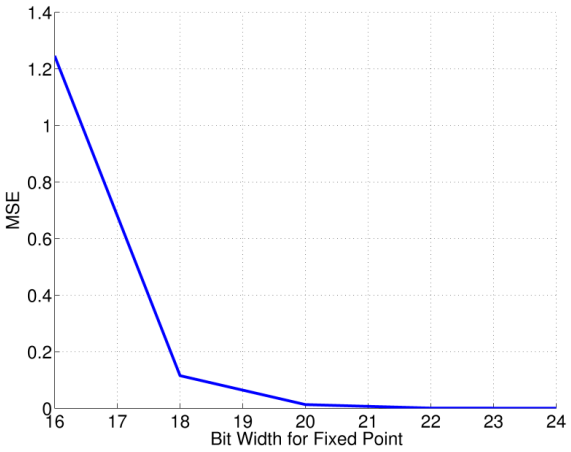


Fig. 10. MSE of the tan-sigmoid using three length fixed point.

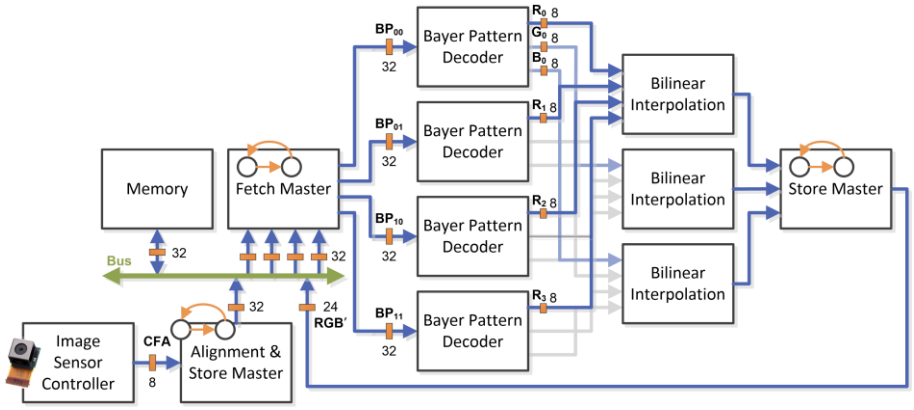


Fig. 12. Fast interpolation flow.

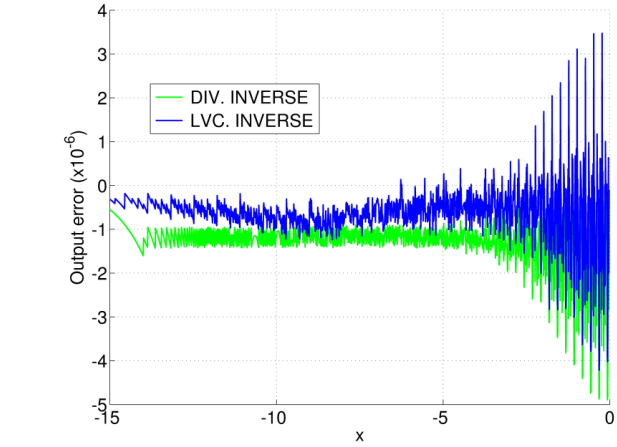


Fig. 11. Output error of the output error calculated using LVC and DIV.

Fig. 14 shows the output error of the tan-sigmoid calculated using the MMHBC and LVC from -15 to 15.

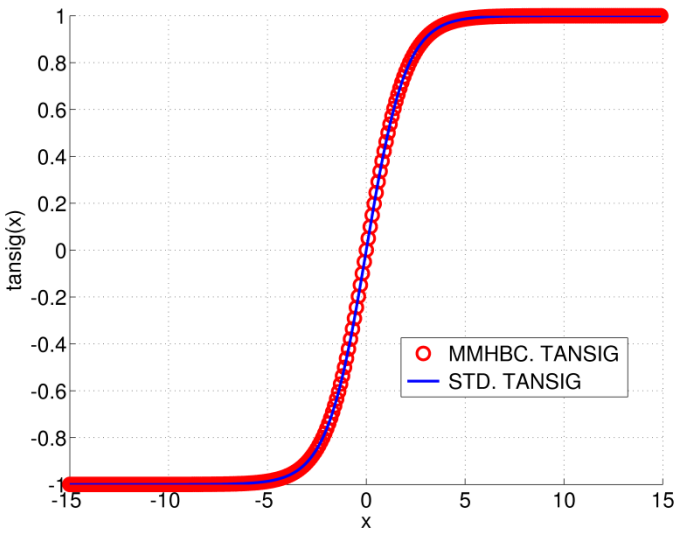


Fig. 13. Output characteristics of the tan-sigmoid obtained using the MMHBC and STD in full range.

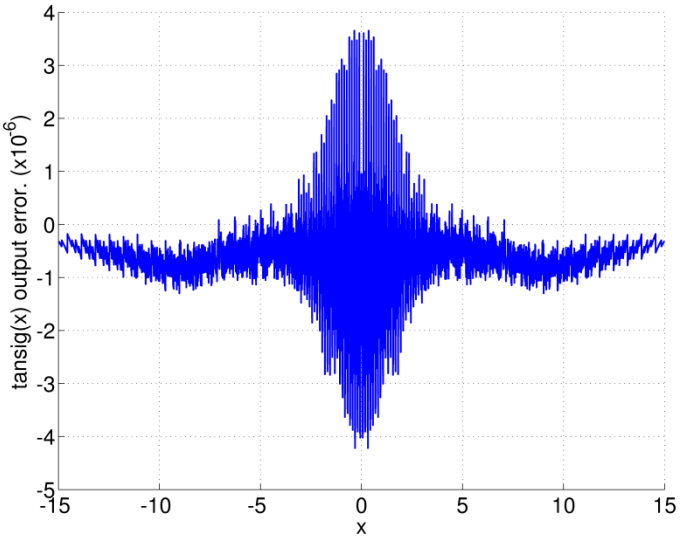


Fig. 14. Output error of the error across the entire range obtained using the MMHBC and LVC.

shows the arithmetic structure of the tan-sigmoid function. A bit-sequence separator is used to separate the input value into sign bit, region bit, and least fraction bit from the net of the neuron. Because the output of the tan-sigmoid is symmetric with respect to the zero axis, the sign bit of the net is stored in a shift buffer, which is used to determine if the output of the tan-sigmoid is positive or negative in the final pipeline stage. Hence, the tan-sigmoid can be calculated from e^z and e^{-z}.

Because the arithmetic of the tan-sigmoid uses a divider to invert the term $1 + e^{-z}$ in (17), the MHBC was used in this study only to calculate e^{-z}. The result is 1 because z is 0. Then, as z increases in negative the direction, the result of the EXP approaches 0.

Thus, the results of $1 + e^{-z}$ range from 2 to 1. The complexity of the divider arithmetic in (17) can be controlled because the divisor and dividend range are fixed, and the dividend is less than or equal to twice the divisor; this is important for using the LVC to implement the divider arithmetic in (17).

D. Bilinear Interpolation

When the pixel location of the DIS (x', y') is obtained, the image interpolation attempts to reconstruct a 2-D continuous signal, $s(x, y)$, from its discrete image samples, $s(x', y')$. The interpolating image space (x, y) of the correction is a real number, and the sampled image space (x', y') of the distortion is an integer. The image interpolation can be described formally as the convolution of the discrete image samples $s(x', y')$ with a 2-D reconstruction filter, h_{2D}, derived by

$$s(x, y) = \sum_{x'} \sum_{y'} s(x', y') * h_{2D}(x - x', y - y') \quad (30)$$

where $x, y \in N$ and $x', y' \in R$. To reduce the complexity of the interpolation [17], h_{2D} is converted into 1-D separable interpolation as

$$h_{2D}(dx, dy) = h(dx) * h(dy) \quad (31)$$

where h is symmetric. The difference dx is the interpolating x minus the sampled x', and the difference dy is similar to dx. The 1-D bilinear interpolation $h_{BL}(x)$ uses two adjacent pixels to obtain the interpolated pixel value, as follows:

$$h_{BL}(dx) = \begin{cases} 1 - |dx|, & 0 \le |dx| < 1 \\ 0, & \text{elsewhere.} \end{cases} \quad (32)$$

Fig. 15 shows the structure diagram of the bi-linear interpolator that was implemented based on (30). Because the arithmetic includes numerous multiplications and additions, the path is excessively long from input x, y to output s. The bilinear interpolator uses the pipe-line method to increase the operating frequency and throughput.

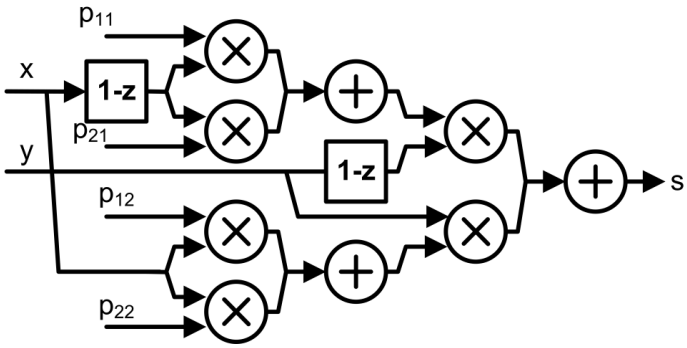


Fig. 15. Implementation of bilinear interpolator.

III. Camera Correction System

The memory access action in a real integrated system is more complex. This section presents the memory access architecture for the NCDC, which was integrated into a real system with a camera and implemented in two FPGA development boards using SDRAM and DDR3 memory.

A. Fast Integration Interpolation

Interpolation is the second performance bottleneck in camera correction methods. As indicated in Section II, the bilinear interpolation must read four pixels from memory to interpolate a new pixel in the middle of the four pixels, as shown in

Fig. 17.

The general image processing flow shown in

Fig. 16 restores the Bayer image to RGB color space before bi-linear interpolation. The RGB space image uses three times the memory space and time for storage. The fetch control shown in

Fig. 16 must read R, G, and B pixels, and the bi-linear interpolation must interpolate these pixels.

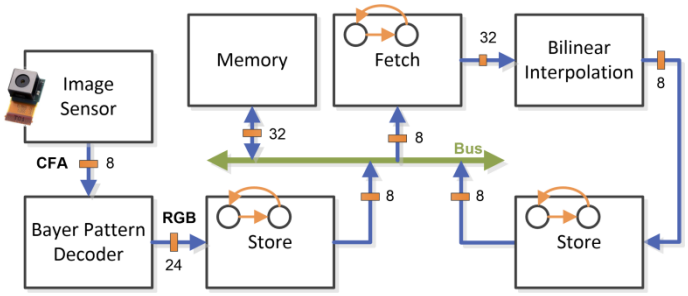


Fig. 16. Traditional image processing flow.

To reduce the image size and access time, the modified image processing flow does not restore the Bayer image of the camera. As shown in

Fig. 17, pixel-mapping can be directly interpolated from the pixels in the Bayer color space to the pixels in the CIS space.

The current and subsequent back-mapped coordinates is in the region surround the same four pixels, as shown in

Fig. 18. Based on the pixel-mapping, the fetch controller can reduce access for interpolating the subsequent CIS pixel.

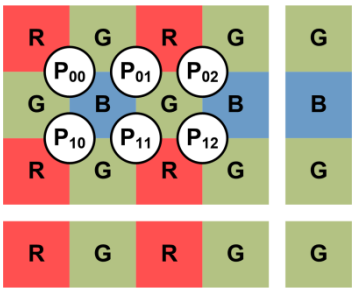
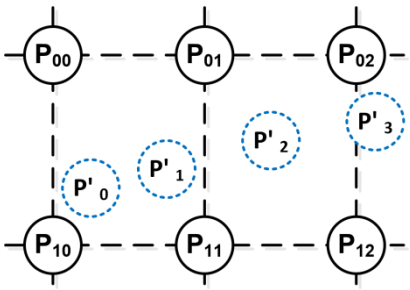


Fig. 17. Pixels mapping of the bilinear interpolation.



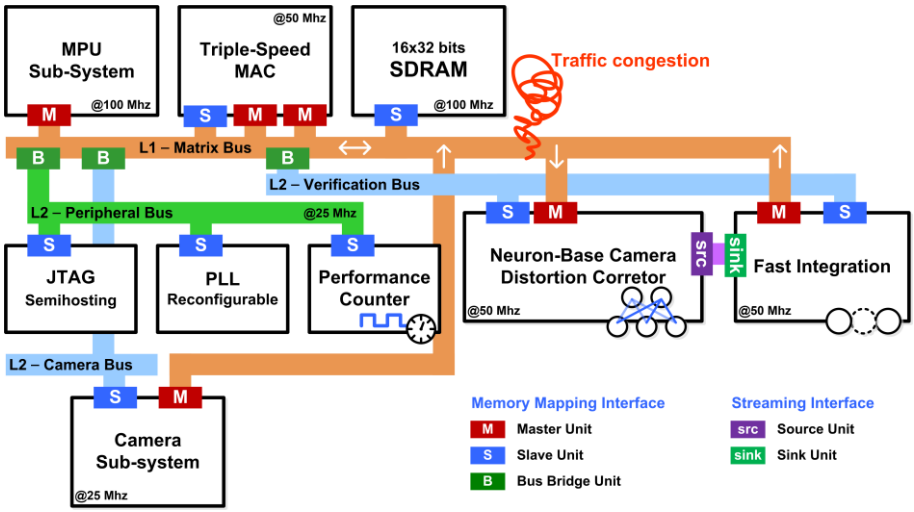


Fig. 19. Verification System of NCDC in DE2-115.

Fig. 18. Pixels mapping in DIS image.

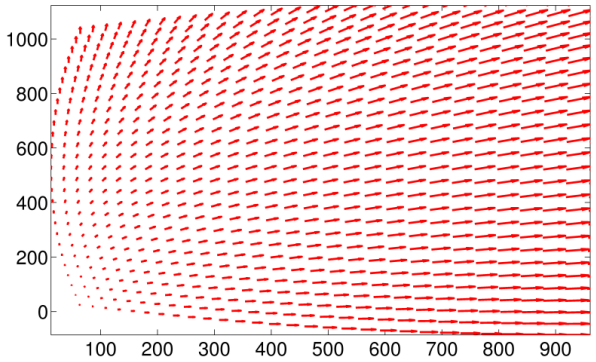


Fig. 20 shows the location of pixels from the DIS to CIS in the memory space; it is difficult for the distortion image pixels to pre-load DIS pixels from the memory space.

For the unusual memory access action, this study proposes fast interpolation arithmetic and flow. As shown in

Fig. 12, the raw image uses the scheduling control to align bytes to four bytes in one row, which can efficiently write image data to memory through the Avalon bus. The fetch control directly reads four bytes from memory to store the Bayer image, and the Bayer pattern decoder restores the 4 groups of RGB pixels. Finally, the bi-linear interpolation uses the RGB pixels to calculate the CIS pixel.

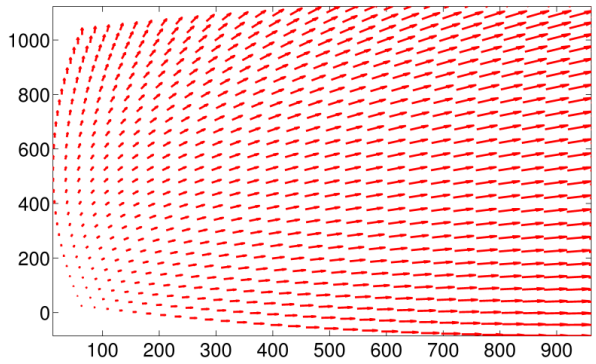


Fig. 20. Pixels location relation from DIS to CIS in memory space.

B. NCDC with a Terasic DE2-115 Development Board

To verify the camera distortion correction methods, this study proposes a verification system to analyze the real data stream between the neural processor and the wide-angle camera.

Fig. 19 shows the proposed verification system that was implemented on a Teraisc DE-115 development board with an Altera Cyclone IV FPGA device [18], which was produced using a 65nm complementary metal–oxide–semiconductor (COMS) process.

The verification system has four subsystems: (1) the MPU subsystem consists of an NIOS-II microprocessor, direct memory access (DMA), synchronous dynamic random access memory (SDRAM), static random access memory (SRAM), and flash memory controllers. The NIOS-II processor core is a reduced instruction set computing (RISC) architecture, which is used to configure and initiate devices in this verification system. The microprocessor manages all memory space in this system and commands the DMA to move large data from one location to another location.

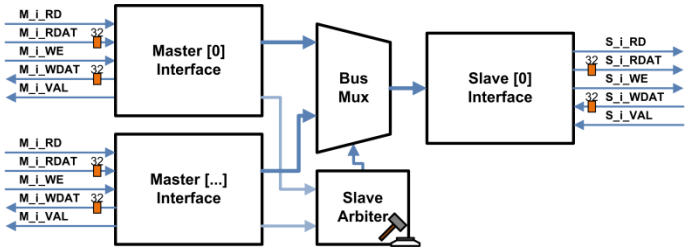


Fig. 21. Avalon bus architecture.

Fig. 21 shows the Avalon bus [19] diagram with slave arbitration architecture. In an advanced RISC machine (ARM) with advanced microcontroller bus architecture (AMBA) [20], the arbiter following the bus uses request signal of the masters to determine the master that can access the slave in the bus. In contrast to an ARM AMBA bus, the arbiter follows the slave, indicating that one layer of the Avalon bus is only one slave. As shown in

Fig. 19, each subsystem has one independent matrix bus that is a multi-layer bus and is operated at a respective different frequency of sub-system.

Fig. 22 shows the burst read protocol of the Avalon bus, which is used to continuously read data from the slave device. The master device reads all data from SDRAM or DDR and may use approximately 6-22 clock cycles. Because SDRAM or DDR memory generally pipe-lines the access command, the burst read protocol uses the characteristic of SDRAM or DDR memory to read data. Moreover, the integrated hardware arithmetic in the Avalon bus is operated at a differing clock domain. The burst read controller uses a pipe-line "READVALID" handshaking signal to detect valid data from the slave device. When starting a burst read action, the "READ" signal is asserted to "1," The "BURSTCOUNT" signal is simultaneously set to "0x04," indicating that the master device reads four words data in this burst read action. When data is ready to be fetched by the master, the slave device asserts the "READVALID" signal to "1". Based on this pipe-line access architecture, the master device can continuously originate a read action following the forward read action.

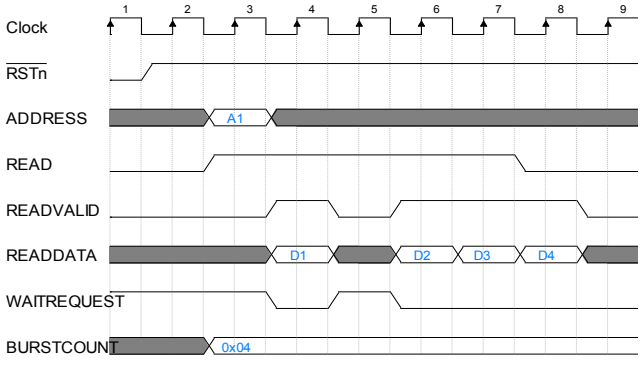


Fig. 22. Burst read protocol of Avalon bus.

Fig. 23 shows the burst write protocol of the Avalon bus, which is used to continuously write data to the slave device. The burst write protocol differs from the burst read protocol. Because the write protocol is a one phase action, the burst write does not require a handshaking signal to detect valid data from the slave device. When starting a burst write action, the master device asserts a "WRITE" signal to "1". The master device simultaneously sets 0x04 to the "BURSTCOUNT" port, indicating that the master device will continuously write four words to the slave device. Depending on whether the "WAITREQUEST" signal is asserted to "0", the master pushes the next word data to the "WRITEDATA" port of the Avalon bus.

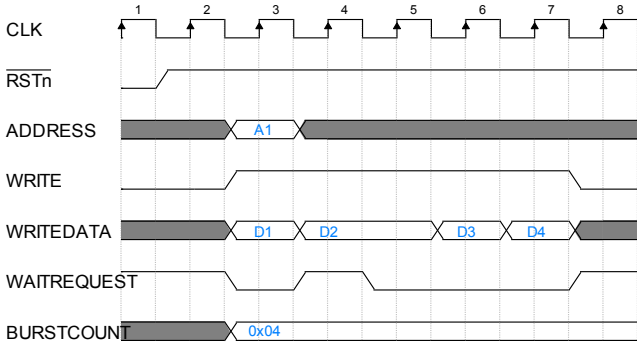


Fig. 23. Burst write protocol of Avalon bus.

(2) The camera subsystem shown in

Fig. 24 consists of a camera capture device, data alignment, I2C controller, control register, burst controller, and slave wrapper. This camera subsystem can be used in most CMOS sensors, which are designed for common purposes. This study used two differing CMOS sensors to capture the distortion image: (i) the Micron MT9D111 [21] is a 2-Mega-pixel sensor, and (ii) the Aptina MT9P014D00 [22] is a 5-Mega-pixel CMOS sensor. Most CMOS sensors use a Bayer color filter array to fetch light from an object. Generally, the captured image passes through a Bayer color interpolation to obtain the color image, and the size of the interpolated image increases to 3 times its original size. In a camera subsystem, the raw data uses the scheduling control to align bytes of a raw image to 4 bytes in one row, which can efficiently write image data to memory through the Avalon bus. The NIOS-II processor can change the operating mode of the CMOS sensor through the I^2C controller, which can access the control register of the CMOS sensor. The burst controller can use the burst mode of the Avalon bus to write an image to memory. When the burst controller writes data, the master monopolizes the slave in the Avalon bus, which can continuously write image data in each clock cycle. The control register of the camera subsystem is used to set its operating mode, and the memory address base is managed by the NIOS-II processor. The NIOS-II processor manages all memory space and allocates a fixed memory space to store the captured image. The address of the allocated memory space is written by the NIOS-II processor through the Avalon bus.

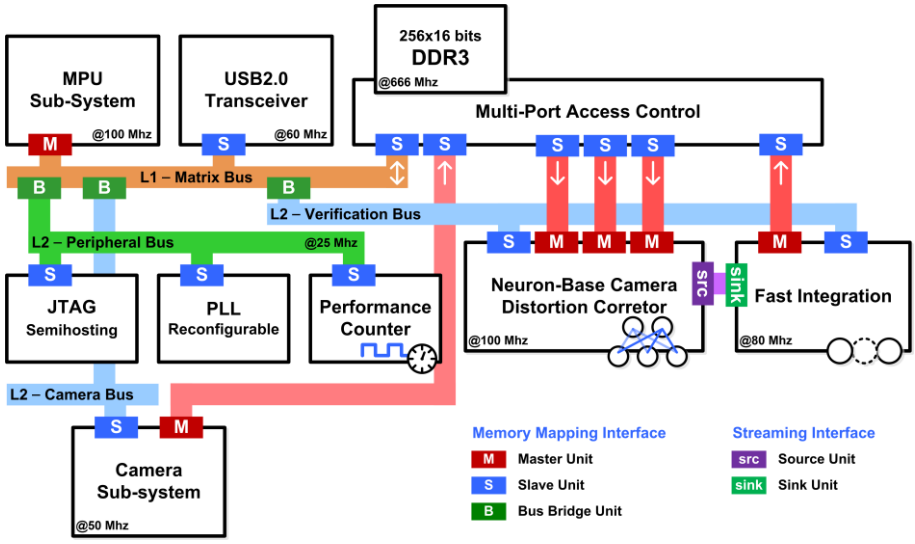


Fig. 26. Verification System of NCDC in FAKA2 development board.

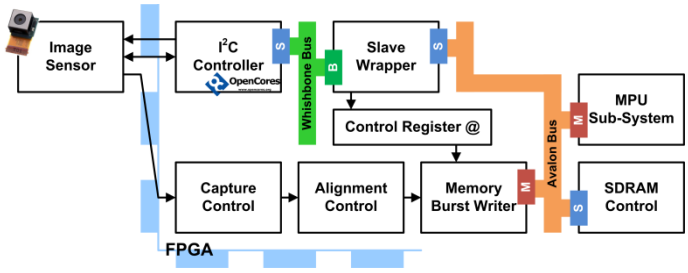


Fig. 24. Architecture of Camera Capture Controller in FPGA.

(3) The Giga-bit Ethernet subsystem shown in Fig. 25 consists of a medium access control (MAC) [23], description ram, read-DMA, and write-DMA. The MAC is used to access the data stream in the gateway of an Ethernet network through a physical (PHY) chip. Because the data is a stream from the MAC, the data stream must write to a memory space through the DMA before analysis. However, it is not efficient to use hardware arithmetic to analyze and calculate the Ethernet package. The half calculation of the Ethernet package is calculated by the NIOS-II processor. Moreover, the socket server service, which is operated by the NIOS-II processor, is used to respond to other clients, which are software programs in a personal computer (PC). When using the NIOS-II processor to compute several programs, the verification system ports a small operating system (OS) (i.e., uC/OS II) to manage multiple tasks.

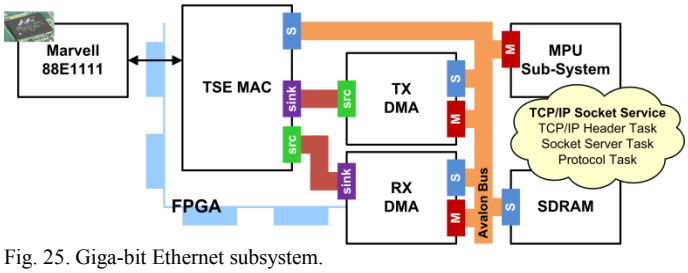


Fig. 25. Giga-bit Ethernet subsystem.

Fig. 27 shows a photograph of a real verification system on FPGA, which consists of a wide-angle lens, a 5M-pixel CMOS sensor, Altera Cyclone IV FPGA device, 64MB SDRAM, and 1 Gb Ethernet PHY.

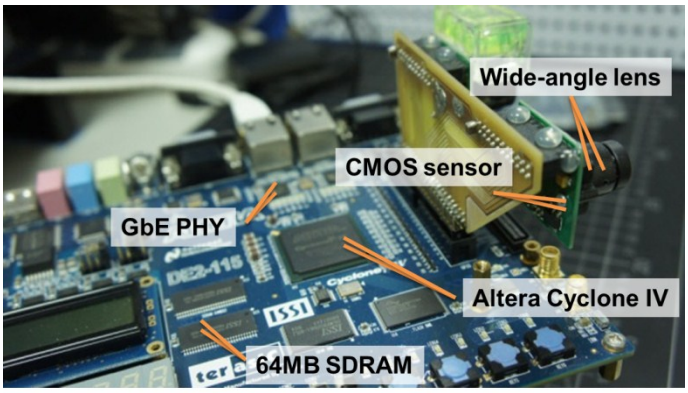


Fig. 27. Verification system on Terasic DE2-115 development board.

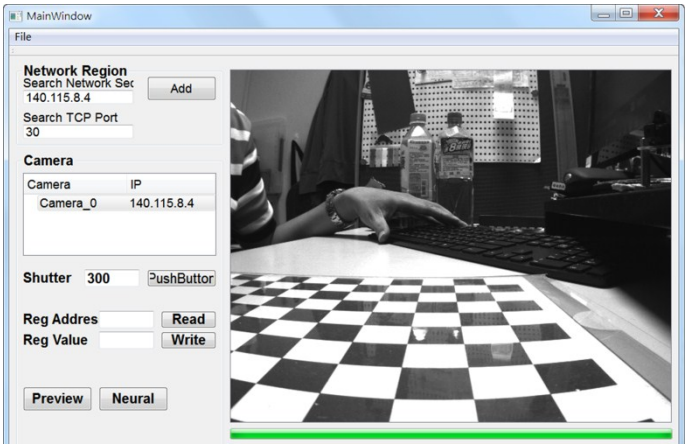


Fig. 28. GUI of client software for Ethernet and DE2-115.

Fig. 28 shows the graphical user interface (GUI) of client software on a PC, which is used to access the verification system on the FPGA development board through the Giga bit Ethernet. The software program of client uses the QT library [24] to build executable file for PC, and the software can re-build on Linux or other OS platforms. The interface can directly change the mode of the CMOS sensor, such as shutter, exposure time, image size, image formation, and capture speed. When using the Ethernet, the interface can access multiple verification systems simultaneously. Conversely, one verification system on the FPGA device can be operated by multiple users from anywhere simultaneously.

TABLE I shows the communication protocol between the client interface and the FPGA development board, which can control the I^2C controller in the verification system.

TABLE I
COMMUNICATION PROTOCOL.

Operating code	Address	Data	Function description
0x80	2 bytes	2 bytes	Read setting from CMOS sensor register.
0x81	2 bytes	2 bytes	Write setting to CMOS sensor register.
0x82	-	-	Reserve.
0x83	-	-	Receive an MRAW package.

TABLE II shows the formation of the MRAW packet. When the client interface sends the 0x83 operating code command (OPC) to the verification system, the NIOS-II processor controls the camera subsystem to capture a distortion image. The neural processor immediately corrects the distortion image and writes the corrected image to memory. The NIOS-II processor prepares the MRAW package and sends the package to the client through the Ethernet.

TABLE II
MRAW PACKAGE FORMATION.

Packet structure	Length	Description
Width	4 bytes	Image width.
Height	4 bytes	Image height.
Formation	1 bytes	Image formation.
Size	7 bytes	Image size.
Socket IP	4 bytes	Ethernet IP of verification system.
Socket Port	4 bytes	Ethernet port of verification system.
Capture time	4 bytes	Capture time from camera subsystem.
Image	Size bytes	Captured image.

Because several sub-systems share the SDRAM, as shown in

Fig. 19, most memory access commands wait until the current access command is completed. The interpolator in the back-end waits for the pixels to be fetched from the memory. Because the back-end cannot manage the back-map coordinates from the front-end, the clock gating controller stops the clock of the front-end until the buffer of the back-end is almost empty.

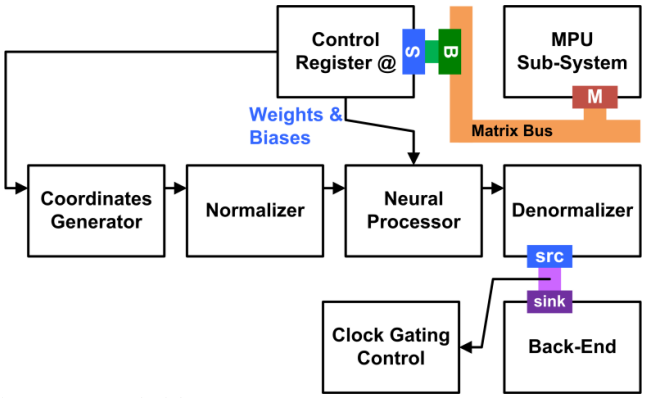


Fig. 29. Front-end of the NCDC.

Because the NCDC always waits for the SDRAM, the efficiency is low. To increase the efficiency of the system, a new development board was designed to improve the memory latency problem.

C. *NCDC with FAKA2-FPGA Development Board*

The specially designed FAKA2-FPGA development board shown in Fig. 31 consists of an Altera 28nm Cyclone V device and a 256MB DDR3 SDRAM.

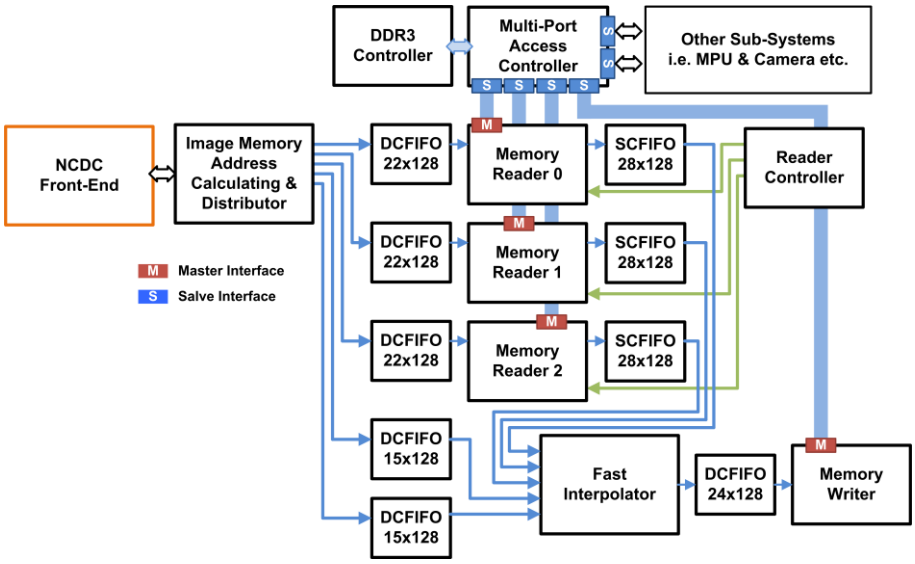


Fig. 30. Back-end of NCDC in FAKA2-FPGA.

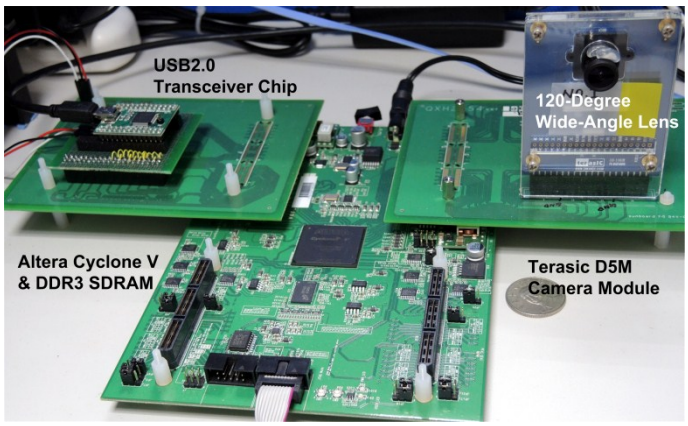


Fig. 31. FAKA2-FPGA Development board.

As shown in Fig. 31, the Terasic camera module and USB2.0 transceiver chip use a bridge board to connect with the FAKA2-FPGA.

Fig. 32 shows the USB2.0 controller in FPGA, which can transmit and receive data between the host of a PC and the correction system in FPGA.

Fig. 33 shows the GUI of client software on a PC, which is used to access the verification system on the FAKA2-FPGA through the USB2.0 transceiver chip.

Fig. 30 shows the multi-port access architecture for the NCDC, which uses 3 memory readers to send the burst read command to fetch the 3 pixels from the DDR3 SDRAM. Therefore, the controller can fetch 9 pixels required for the fast interpolator in one access command. For verification, the interpolated pixels use a memory writer to write memory after a fast interpolation operation. If the NCDC system is applied to other applications, the memory writer can be changed to a stream source interface that is similar to a camera interface.

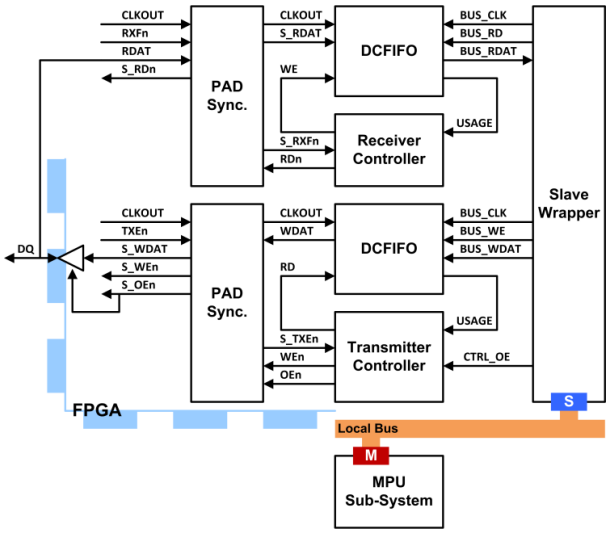


Fig. 32. Architecture of FTDI FT2232H USB2.0 Controller in FPGA.

Fig. 33. GUI of client software for USB2.0 and FAKA2-FPGA.

IV. Experiment

Using the NCDC with the DE2-115, the GUI shown in Fig. 28 captures the distortion image in Bayer color space from the camera with a 105° wide-angle lens.

Fig. 34 shows that the non-distortion color image has 1280x1024 resolution, which was corrected by the NCDC in the DE2-115.

Fig. 35 shows the image captured by the 1280x1024 resolution camera with a 105° wide-angle lens, and clearly shows barrel distortion.

To determine whether the corrected result is preferable to that of the PBM or the NCDC, this study used the Hough line detection [25] method to find the straight line in the corrected images.

Fig. 36 shows the line detection result of the PBM method, which missed some lines on the periphery of the corrected check-board image.

Fig. 37 shows the line detection result of the NCDC, which provided differing results to those in

Fig. 36 because it did not miss lines on the periphery of the corrected check-board image.

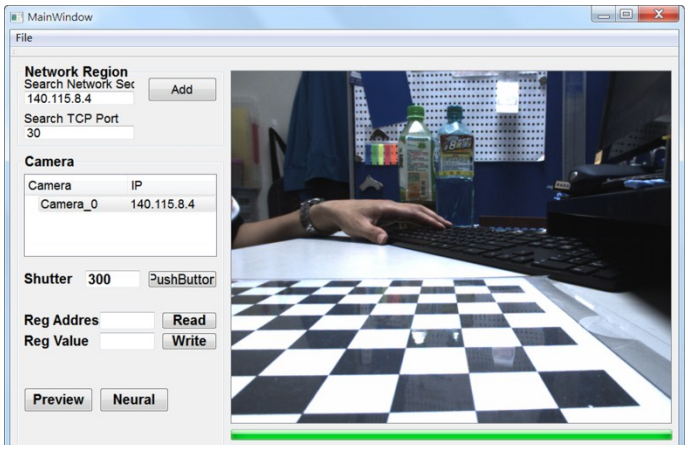


Fig. 34. Corrected Image by NCDC with DE2-115.

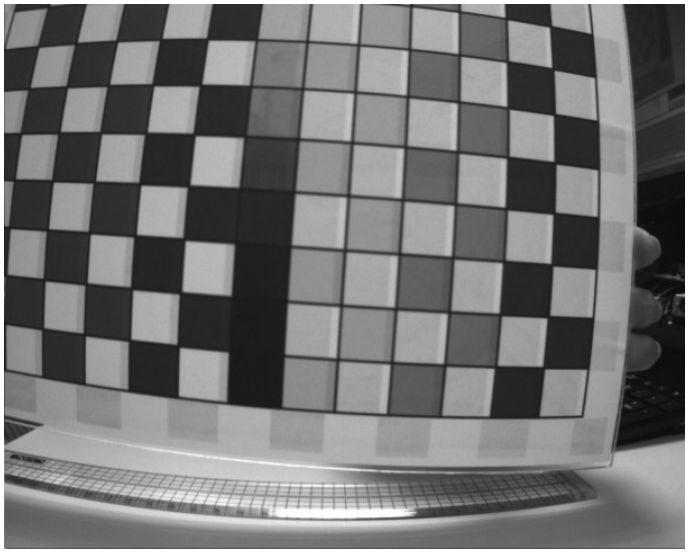


Fig. 35. Captured image from 105° wide-angle camera. (1280*1024)

These results show that the NCDC is more accurate in correcting the camera distortion caused by the camera, lens distortions, and various manufacturing flaws.

Fig. 38 shows the captured Bayer color image of the 1920x1080 resolution camera with a 120° wide-angle lens, and clearly shows barrel distortion.

Fig. 39 shows the corrected image of a 120° wide-angle camera using NCDC. Because the full-width range of the distortion image is shown in

Fig. 38, the corrected image in

Fig. 39 is smaller than that in

Fig. 38. The NCDC in FAKA2-FPGA simultaneously corrects the distortion of the camera and scales the image size to fit 1920 pixels for image width.

TABLE III shows an error comparison of the NCDC results obtained using four to six neurons in the hidden layer and the PBM with six to eight polynomial orders. Using floating-point software for the 120° wide-angle lens, the results show that the MSE reaches 0.037571 when four neurons are in the NCDC, and the maximal error is 1.40221260 pixels. When seven neurons are present, the maximal error in the whole image is under 0.32898513 pixels. However, in this camera sensor, the pixel width is 2.8 μm. If the neuron number is greater than four, the difference in the corrected image cannot be observed by the human eye. Therefore, the optimal neuron number was four for the 1920x1080 pixel camera with a 120° wide-angle lens. Although the chip area of the NCDC is 10 times larger than that used by Asari [8], the NCDC is 429 times more accurate than the six-order PBM based on the Asari method for 105° wide-angle lenses.

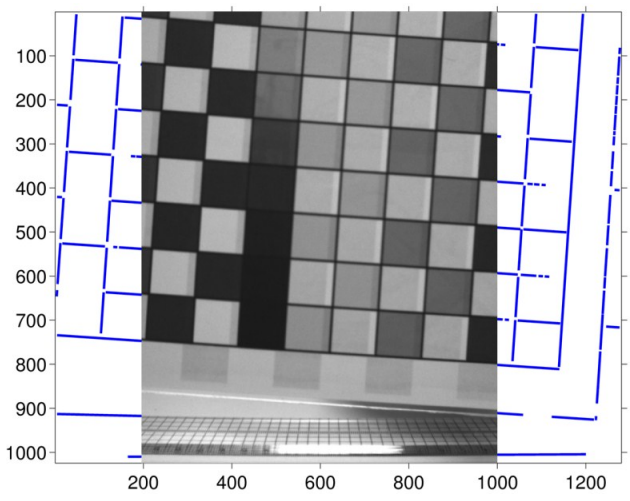


Fig. 36. Line detection of PBM corrected image.

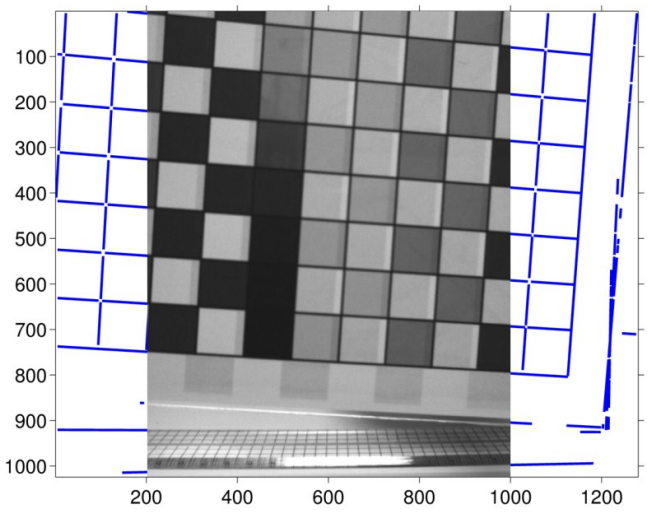


Fig. 37. Line detection of NCDC corrected image.

Because the hardware arithmetic uses a fixed point with the 120° wide-angle lens, some of the correction errors originate from the transformation errors of the floating-point coordinates. When the floating point is used, the operating range for the NCDC is approximately -1 to 1. However, to use the pipeline register width of the VLSI architecture efficiently, the range of the NCDC with a fixed point is -0.9999 to 0.9999. The results show that the MSE reaches 0.150788011, similar to the case with the NCDC hardware arithmetic with four neurons, and the maximal error is 1.170525455 pixels.

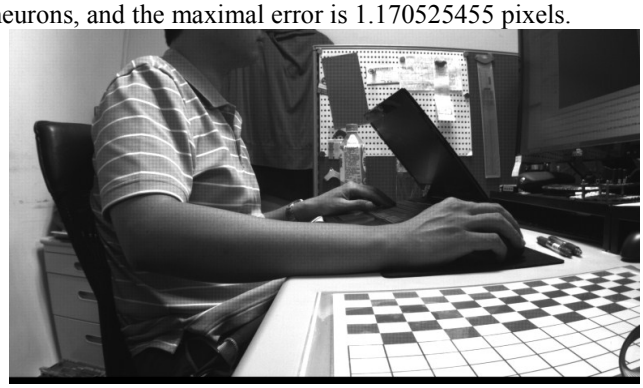


Fig. 38. Captured image from 120° wide-angle camera. (1920*1080)

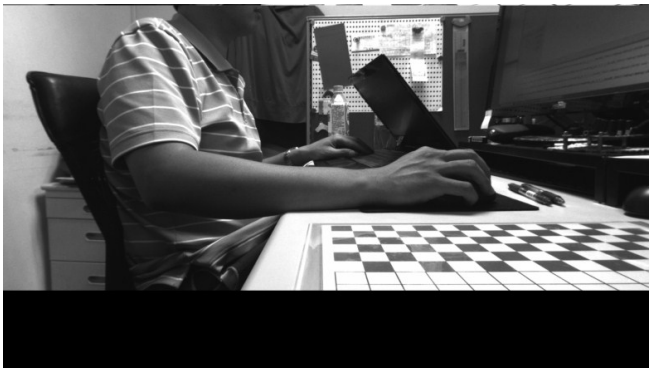


Fig. 39. Corrected image of 120° wide-angle camera using NCDC in FAKA-FPGA

TABLE III
COMPARISON OF CORRECTION PRECISION ERRORS FOR 105° AND 120° WIDE-ANGLE LENSES.

PBM	Order (N)	Lens	MSE	Avg. Error	Max. Error
	6	105°	87.9597	7.8584	21.1068
	7		87.9452	7.8571	20.9047
	8		87.9316	7.8566	20.8914
NCDC	**Neurons (HN)**				
	4	105°	0.205017	0.252331	4.621188
	5		0.021510	0.104972	0.573920
	6		0.005172	0.047855	0.357827
	4	120°	0.037571	0.122573	1.402213
	5		0.007127	0.056211	0.435714
	6		0.004582	0.044143	0.353196

Wide-angle camera distortion correction is a crucial function in medical and manufacturing equipment. However, the distortion correction of the camera is a complex computation that uses a graphic processing unit (GPU) to correct the distortions in medical applications [1]. In industry applications, LabView [26] (National Instruments Corporation) is commonly used to correct the camera for quality testing on production lines, which use traditional camera distortion models to correct camera distortions. LabView was run on an Intel i7 3.9Ghz PC with Dual Channel 1666 DDR3 SDRAM, which requires 0.7888 s to correct a gray 1920×1080 pixels image. The NCDC in DE2-115 with SDRAM requires 2.1346 s to correct a color 1280×1024 pixel image. The image of the NCDC in DE2-115 has 1.8963x more pixels than that of the image in LabView. Therefore, the performance of LabView is 1.4285x higher than the NCDC in DE2-115. As shown in TABLE IV, the NCDC in FAKA2-FPGA with single DDR3 SDRAM requires 0.1219 s to correct a color 1920×1080 pixels image. Therefore, the performance of the NCDC in FAKA-FPGA is 19.3929x higher than that of LabView.

TABLE IV
CORRECTION PERFORMANCE

Method	Memory	Resolution	Time (s)	FPS
LabView [26] in Intel i7	Dual Channel DDR3*	1920×1080 Gray	0.7880	1.27
NCDC in DE2-115	SDRAM	1280×1024 Color	2.1346	0.47
NCDC in FAKA2	DDR3	1920×1080 Color	0.1219	8.21

V. CONCLUSIONS

This study used efficient system architecture in FPGA for camera distortion correction to rapidly and accurately correct various lens and manufacturing flaws in low-cost cameras. For the NCDC with four neurons, the results show that the maximal corrected error in a whole image is less than 1.4 pixels, and the MSE approaches 0.0376 between the corrected and ideal results. The proposed novel neuron-based method is

more suitable than the traditional method to correct numerous asymmetric manufacturing defects in low-cost wide-angle cameras. Moreover, this study designed a FAKA2-FPGA development board for the NCDC. The proposed architecture of the NCDC with the FAKA2-FPGA development board has more than 19.3929x the performance than the popular professional solutions, indicating that the NCDC can be used for applications that do not require a high frame rate. For real time applications, such as medical endoscopy, the back-end of the NCDC can reduce the camera resolution to under 1024×768 pixels, or increase the operating frequency to increase the bandwidth usage rate of the DDR3 SDRAM.

REFERENCES

[1] R. Melo, J. P. Barreto, and G. Falcao, "A New Solution for Camera Calibration and Real-Time Image Distortion Correction in Medical Endoscopy–Initial Technical Evaluation," *IEEE Transactions on Biomedical Engineering,* vol. 59, pp. 634-644, Mar. 2012.

[2] D. C. Brown, "Close-range camera calibration," *Photogramm. Eng. Remote Sens.,* vol. 37, pp. 855-866, Jan. 1971.

[3] J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 14, pp. 965-980, Oct. 1992.

[4] K. V. Asari, S. Kumar, and D. Radhakrishnan, "A new approach for nonlinear distortion correction in endoscopic images based on least squares estimation," *IEEE Transactions on Medical Imaging,* vol. 18, pp. 345-354, Apr. 1999.

[5] S.-L. Chen, H.-Y. Huang, and C.-H. Luo, "Time Multiplexed VLSI Architecture for Real-Time Barrel Distortion Correction in Video-Endoscopic Images," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 21, pp. 1612-1621, Nov. 2011.

[6] P. Y. Chen, C. C. Huang, Y. H. Shiau, and Y. T. Chen, "A VLSI Implementation of Barrel Distortion Correction for Wide-Angle Camera Images," *IEEE Transactions on Circuits and Systems II-Express Briefs,* vol. 56, pp. 51-55, Jan. 2009.

[7] L. Qiang and N. M. Allinson, "FPGA Implementation of Pipelined Architecture for Optical Imaging Distortion Correction," in *IEEE Workshop on Signal Processing Systems Design and Implementation, SIPS '06.* , Banff, Canada, 2006, pp. 182-187.

[8] H. T. Ngo and V. K. Asari, "A pipelined architecture for real-time correction of barrel distortion in wide-angle camera images," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 15, pp. 436-444, Mar. 2005.

[9] K. V. Asari, "Design of an efficient VLSI architecture for non-linear spatial warping of wide-angle camera images," *Journal of Systems Architecture,* vol. 50, pp. 743-755, Aug. 2004.

[10] C.-H. Chen, T.-K. Yao, and C.-M. Kuo, "Wide-Angle Camera Distortion Correction Using Neural Back Mapping " in *IEEE International Symposium on Consumer Electronics, ISCE '13*, Hsinchu, Taiwan, 2013, pp. 171-172.

[11] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation,* vol. 3, pp. 323-344, Aug. 1987.

[12] H. T. Ngo and V. K. Asari, "Design of a High Performance Digital Architecture for Real-Time Correction of Radial Lens Distortion," in *Circuits and Systems, 2006. MWSCAS '06. 49th IEEE International Midwest Symposium on*, 2006, pp. 526-530.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *D. E. Rumelhart and J. L. McClelland, Eds., Parallel distributed processing: explorations in the microstructure of cognition, vol. 1*, ed Cambridge: MA: MIT Press, 1986, pp. 318-362.

[14] X. P. Zhu and Y. W. Chen, "Improved FPGA implementation of Probabilistic Neural Network for neural decoding," in *International Conference on Apperceiving Computing and Intelligence Analysis, ICACIA '10*, Chengdu, China, 2010, pp. 198-202.

[15] B. Gisutham, T. Srikanthan, and K. V. Asari, "A high speed flat CORDIC based neuron with multi-level activation function for robust pattern recognition," in *IEEE International Workshop on Computer Architectures for Machine Perception*, Padova, Italy 2000, pp. 87-94.

[16] C.-H. Chen, T.-K. Yao, and C.-M. Kuo, "An Efficient Neural Processor for Camera Distortion Correction," *IEEE Transactions on Circuits and Systems for Video Technology (reviewing),* 2013.

[17] H. Kim, Y. Cha, and S. Kim, "Curvature Interpolation Method for Image Zooming," *Image Processing, IEEE Transactions on,* vol. 20, pp. 1895-1903, 2011.

[18] A. Corporation. (Feb. 2013). *Cyclone IV FPGA Device Family Overview (12.1 ed.)*. Available: http://www.altera.com/literature/hb/cyclone-iv/cyiv-51001.pdf

[19] A. Corporation. (May 2011). *Avalon Interface Specifications (11.0 ed.)*. Available: http://www.altera.com/literature/manual/mnl_avalon_spec.pdf

[20] A. Holdings. (2008). *AMBA Open Specifications (2.0 ed.)*. Available: http://www.arm.com/zh/products/system-ip/amba/amba-open-specifications.php

[21] Aptina. (2012). *MT9D111D00STC Data Sheet*. Available: http://www.aptina.com/products/soc/mt9d111d00stc/

[22] Aptina. (2012). *MT9P014D00STC Data Sheet*. Available: http://www.aptina.com/products/image_sensors/mt9p014d00stc/

[23] A. Corporation. (Jan. 2013). *Triple-Speed Ethernet MegaCore Function User Guide (12.1 ed.)*. Available: http://www.altera.com/literature/ug/ug_ethernet.pdf

[24] Digia. (2011). *QT Project Wiki*. Available: http://qt-project.org/wiki/

[25] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Commun. ACM,* vol. 15, pp. 11-15, 1972.

[26] N. I. Corporation. *LabVIEW*. Available: http://www.ni.com/labview/