January 2013

VLSI Implementation of Barrel Distortion Correction in Endoscopic Images based on Least Squares Estimation

R. Sathya Vignesh
r.sathya@gmail.com

S. Saranya Devi
s.saranyadevi@gmail.com

Follow this and additional works at: <https://www.interscience.in/ijpjt>

Part of the [Medical Pharmacology Commons](#)

Recommended Citation

Vignesh, R. Sathya and Devi, S. Saranya (2013) "VLSI Implementation of Barrel Distortion Correction in Endoscopic Images based on Least Squares Estimation," *International Journal of Pharmacology and Pharmaceutical Technology*: Vol. 1 : Iss. 1 , Article 8.

DOI: 10.47893/IJPPT.2013.1004

Available at: <https://www.interscience.in/ijpjt/vol1/iss1/8>

This Article is brought to you for free and open access by the Interscience Journals at Interscience Research Network. It has been accepted for inclusion in International Journal of Pharmacology and Pharmaceutical Technology by an authorized editor of Interscience Research Network. For more information, please contact sritampatnaik@gmail.com.

VLSI Implementation of Barrel Distortion Correction in Endoscopic Images based on Least Squares Estimation

R. Sathya Vignesh & S. Saranyadevi

Abstract - An efficient VLSI Implementation of Barrel Distortion Correction (BDC) in Endoscopic Images based on Least Squares Estimation is presented in this paper. Computational complexity is reduced by employing Odd order polynomial, as an approximation to Back-mapping expansion polynomial. This polynomial can be solved in monomial form, by Estrin's algorithm. In Estrin's algorithm, a high order expression can be factorized in to sub-expression, which can be evaluated in parallel. In our simulation, on comparison with some existing distortion correction techniques, 75% of hardware cost and 70% of memory requirement is reduced by using TSMC 0.18 μ m technology.

Keywords - Barrel Distortion Correction (BDC), Least Squares Estimation, Computational complexity, Odd order polynomial, Back-mapping expansion polynomial, Estrin's algorithm.

I. INTRODUCTION

Endoscopy is an invaluable tool in medical applications, such as pulmonary medicine, urology, orthopedic surgery, gynecology and surgical procedures. It is based on Minimal Invasive Therapy (MIT), which is becoming popular because of the use of natural or artificial orifices of the body for surgery and permits little or no injury to healthy organs and tissues [6]. In Endoscopy, the camera with wide angle lens (fish-eye lens) is employed to capture larger portion of the interior structure in a single image. These lenses enhance the imaging capability, but the images suffer from barrel distortion. Barrel Distortion causes the non-linear changes in the image, where the image regions at the corner of the image are compressed more than the interior region. Hence the outer regions are much smaller than their actual size. This inhomogeneous image compression causes intolerable errors in the images obtained during feature extraction.

Several researchers have presented various techniques and algorithms to correct barrel distortion, based on mathematical models [1] – [6]. Most of the software solutions were also presented. However these solutions do not provide high speed solutions, which is the most essential requirement in medical imaging. Some VLSI architectures [1], [2] and [4] of Barrel Distortion Correction (BDC) have also been presented. In [1] Chen et al have proposed VLSI architecture for real-time barrel distortion correction in Video-

Endoscopic images based on Hornor's algorithm. In [7], Asari had presented an efficient VLSI architecture to correct the barrel distorted images by mapping the algorithmic steps on to a linear array and a pipelined architecture is presented in [4]. As the correcting method for each distorted pixel requires more instruction cycles, the software solutions are tough to meet the constraint of real-time video-endoscopic applications. A pipelined Co-ordinate Rotation Digital Computer (CORDIC) is employed in this paper to develop the effectiveness of co-ordinate transformation. By eliminating the computation of angle θ and by employing the Odd-order back-mapping polynomial, our design achieves high performance with reduced hardware cost. This design can be either implemented with a dedicated chip or be integrated with other image processing components or as a single chip for wide angle camera applications.

By careful analysis, it is found that most computing resources are employed on co-ordinate transformation, back-mapping and linear interpolation procedures. As in [1], the computing resource is reduced by Odd-order polynomial approximation for back-mapping. To reduce the hardware cost, we employ algebraic transformation to solve polynomial equation. Therefore Estrin's algorithm [10] is employed, which improves the latency of back-mapping, when compared with Hornor's algorithm.

The rest of the paper is organized as follows. Proposed distortion correction technique is discussed briefly in section II, section III describes the proposed VLSI architecture in detail. In section IV comparisons and experimental results are presented. The conclusion is provided in section V.

II. PROPOSED DISTORTION CORRECTION TECHNIQUE

In this section, the algorithm to correct barrel distortion based on least-square estimation method is presented [1] – [4]. Let the Distorted and Corrected Image Space be represented by DIS and CIS respectively. Two steps in Barrel Distortion Correction are: 1) Back-mapping of all pixels on CIS on to DIS, and 2) Calculating the intensity of every pixel in CIS by linear interpolation [2]. The block diagram of proposed BDC is shown in Fig 1.

A. Cartesian to Polar Coordinate Transformation

The first step in the proposed BDC is the forward mapping of all pixels in Distorted Image Space (DIS) on to Corrected/Expanded Image Space (CIS). Let (u_c', v_c') and (u_c, v_c) represent the distortion center in DIS and Correction center in CIS respectively. In DIS, the distance ρ' from the distortion center (u_c', v_c') to any image pixel location (u', v') and the angle θ' between the pixel and distortion center are given by,

$$\rho' = \sqrt{(u' - u_c')^2 + (v' - v_c')^2} \quad (1a)$$

$$\theta' = \arctan\left(\frac{v' - v_c'}{u' - u_c'}\right) \quad (1b)$$

Fig. 1 : Block diagram of proposed BDC.

Let the same pixel (u', v') can be transformed to a new location (u, v) in CIS. The corresponding value of

the distance ρ and the angle θ from corrected center (u_c, v_c) to (u, v) are given by,

$$\rho = \sqrt{(u - u_c)^2 + (v - v_c)^2} \quad (2a)$$

$$\theta = \arctan\left(\frac{v - v_c}{u - u_c}\right) \quad (2b)$$

The primary objective of this mathematical model is to establish relationship between the distances ρ and ρ' . It is defined by an expansion polynomial of degree N as given by,

$$\rho = \sum_{n=1}^N (a_n) \rho'^n \quad (3a)$$

$$\theta = \theta' \quad (3b)$$

where a_n is the expansion co-efficient, that can be obtained by using least-squares estimation technique [6]. Here, the distortion has been assumed to be purely radial, which implies that, there will be no change in the argument values in CIS and DIS, i.e., $\theta = \theta'$. The co-ordinates of the new location (u, v) of the pixel in CIS given by,

$$u = u_c + \rho \cos \theta' \quad (4a)$$

$$v = v_c + \rho \sin \theta' \quad (4b)$$

B. Back-Mapping Procedure

As presented in [6], the back-mapping procedure can be described by a back-mapping expansion of degree N is given by,

$$\rho' = \sum_{n=1}^N (b_n) \rho^n \quad (5a)$$

$$\theta = \theta' \quad (5b)$$

where b_n is the back-mapping coefficient which can be obtained by the least-squares estimation method.

C. Polar to Cartesian Coordinate Transformation

The third step in proposed BDC is to perform polar to Cartesian coordinate conversion. Given a set of pixel location in DIS, their corresponding location in CIS can be obtained from expansion process, as described in [4]. The new pixel location (u', v') corresponding to ρ' and θ' is given by,

$$u' = u_c' + \rho' \cos \theta' \quad (6a)$$

$$v' = v_c' + \rho' \sin \theta' \quad (6b)$$

D. Polynomial Approximating Analysis And Simplified Back-Mapping Procedure

As mentioned in [1], the back-mapping expansion polynomial can be approximated to Odd-order or Even-order polynomial. But, Odd-order polynomial achieves on an average of 97.13% approximation, which is better

than Even-order polynomial by 58.53%. Thus the approximated Odd-order back-mapping polynomial is given by,

$$\rho' = c_0\rho + c_1\rho^3 + c_2\rho^5 + c_3\rho^7 + \dots \quad (7)$$

where $c_0, c_1, c_2, c_3 \dots$ are back-mapping coefficients of the odd-order back-mapping polynomial. From (2a), ρ value is the square root of the sum of $(u-u_c)^2$ and $(v-v_c)^2$. In [4], the Coordinate Rotation Digital Computer (CORDIC) module is implemented with enormous hardware to compute the value of ρ .

In order to achieve the low cost VLSI implementation, two processes are performed to reduce the computing resource of back-mapping. The first is to eliminate the calculation of θ [2]. As mentioned in (3b) and (5b), θ' and θ are the same. Thus, the $\cos\theta'$ and $\sin\theta'$ can be calculated as,

$$\cos\theta' = \cos\theta = \frac{u - u_c}{\rho} \quad (8a)$$

$$\sin\theta' = \sin\theta = \frac{v - v_c}{\rho} \quad (8b)$$

From [1], using eqn .8, the back-mapping and polar to Cartesian coordinate steps can be simplified by eliminating θ as given by,

$$u' = u_c' + \rho' \frac{u - u_c}{\rho} \\ = u_c' + (c_0 + c_1\rho^2 + c_2\rho^4 + c_3\rho^6 + \dots)(u - u_c) \quad (9a)$$

$$v' = v_c' + \rho' \frac{v - v_c}{\rho} \\ = v_c' + (c_0 + c_1\rho^2 + c_2\rho^4 + c_3\rho^6 + \dots)(v - v_c) \quad (9b)$$

From eqn .9, we find that, no odd power of ρ exists and hence the square-root calculation for ρ can be removed. Thus, the number of steps in BDC procedure can be greatly reduced. Further both u' and v' are calculated using ρ^2 rather than with ρ given by,

$$\rho^2 = (u - u_c)^2 + (v - v_c)^2 \quad (10)$$

E. Estrin's Algorithm

Estrin's algorithm comes from parallel computing research. It works by dividing the polynomial into an implied tree of multiply-adds, allowing each level of the tree to be evaluated in parallel. The main idea of Estrin's method is to separate the polynomial into terms of the form $(Bx+A)$, evaluating them and then using the results as the coefficients in the next level of $(Bx + A)$. However, unlike Horner's form, it does not reduce the number of arithmetic operations required to evaluate a polynomial. Estrin's method stems from the observation that regular patterns emerge when polynomials are

factorized. For example, a 3rd order expression can be factorized into Sub-expressions of $(Bx + A)$ as follows.

$$= x^3 + x^2 + x \quad (11) \\ = (x + 1)^2 + (bx+a)$$

Extending this to higher order polynomials, an Estrin's method table can be built up:

$$\begin{aligned} 0() &= \\ 1 &= (x + 1) \\ 2 &= x^2 + (x + 1) \\ 3 &= (x + 1)^2 + (x + 1) \\ 4 &= x^4 + (x + 1)^2 + (x + 1) \\ 5 &= (x + 1)^4 + (x + 1)^2 + (x + 1) \\ 6 &= x^2 + (x + 1)^4 + (x + 1)^2 + (x + 1) \end{aligned}$$

Fig 2: Estrin's Method table

For example, the evaluation of a 7th order polynomial $(hx^7 + gx^6 + fx^5 + ex^4 + dx^3 + cx^2 + bx + a)$ looks like this:

Fig 3: Evaluation of a 7th order polynomial using Estrin's method

The terms in each row of Figure 3 are evaluated in parallel; all the terms on a row must be fully evaluated before the next row can be started. Estrin's method uses an implicit binary evaluation tree implied by splitting $f(x)$ as:

$$\left[\sum_{0 \leq i < h} c_h + i^{x^i} \right] x^h + \left[\sum_{0 \leq i < h} c_i x^i \right] \quad (12)$$

where $h = \frac{(n+1)}{2}$ is a power of 2. An Nth degree polynomial, requires $\log_2(N) + 1$ rows of expressions to evaluate.

By Estrin's algorithm, the computing complexity of the evaluating polynomial can be efficiently decreased. The complexity of the back-mapping and polar to Cartesian coordinate procedures can also be reduced in similar manner. Eqns (9a) and (9b) can be rewritten as,

$$u' = u'_c + [(c_0 + c_1 \rho^2) + (c_2 + c_3 \rho^2) \rho^4] \times \frac{u - u_c}{\rho} \quad (13a)$$

$$v' = v'_c + [(c_0 + c_1 \rho^2) + (c_2 + c_3 \rho^2) \rho^4] \times \frac{v - v_c}{\rho} \quad (13b)$$

where $c_0, c_1, c_2, c_3, \dots, c_{n-1}, c_n$ are the combined mapping coefficients of the polynomial. Thus, the most complexity of computing the power values of ρ^2 ($\rho_4, \rho_6, \rho_8 \dots$) can be efficiently reduced by Estrin's algorithm.

F. Simplified Linear Interpolation

As presented in [1], the linear interpolation can be directly implemented by the linear interpolation equation as given by,

$$I(x, y) = I'(x, y) \times (1 - dx) \times (1 - dy) + I'(x + 1, y) \times dx \times (1 - dy) + I'(x, y + 1) \times (1 - dx) \times dy + I'(x + 1, y + 1) \times dx \times dy \quad (14)$$

where $I'(x, y)$, $I'(x + 1, y)$, $I'(x, y + 1)$ and $I'(x + 1, y + 1)$ are the intensities of the four neighboring pixels, at the location given by their indices (x, y) , $(x + 1, y)$, $(x, y + 1)$, and $(x + 1, y + 1)$. Now, the computing resource of this linear interpolation requires at least 8 multiply and 3 add operations. This implies that, eight multipliers and

three adders are necessary when implementing a VLSI circuit.

In order to reduce the computational complexity and the hardware cost of linear interpolation, the algebraic manipulation is employed. The modified Linear Interpolation equation is shown in equation (15). From equation (15d) it is clear that, this simplified linear interpolation requires only 3 multiplications and 6 addition operations [1].

$$I(x, y) = I'(x, y) \times (1 - dx) \times (1 - dy) + I'(x + 1, y) \times dx \times (1 - dy) + I'(x, y + 1) \times (1 - dx) \times dy + I'(x + 1, y + 1) \times dx \times dy \quad (15a)$$

$$= I'(x, y) \times [(1 - dx) \times (1 - dy) + dx \times (1 - dy) + (1 - dx) \times dy + dx \times dy] + I'(x + 1, y) \times dx \times (1 - dy) + I'(x, y + 1) \times (1 - dx) \times dy + I'(x + 1, y + 1) \times dx \times dy \quad (15b)$$

$$= I'(x, y) \times \{ (1 - dx) \times (1 - dy) + dx \times (1 - dy) + (1 - dx) \times dy + dx \times dy \} + I'(x + 1, y) \times dx \times (1 - dy) + I'(x, y + 1) \times (1 - dx) \times dy + I'(x + 1, y + 1) \times dx \times dy \quad (15c)$$

TABLE II
COMPARISON OF EQUATIONS AND COMPUTING RESOURCE WITH PREVIOUS TECHNOLOGIES

	Step1: Cartesian to Polar Coordinate	Step2: Back-Mapping	Step3: Polar to Cartesian Coordinate	Step4: Linear Interpolation	Computing Resource Per Pixel
[4]	$\rho' = \sqrt{(u' - u_c)^2 + (v' - v_c)^2}$ $\theta' = \arctan\left(\frac{v' - v_c}{u' - u_c}\right)$	$\rho' = \sum_{n=1}^N (b_n) \rho^n$	$u' = u_c' + \rho' \cos \theta'$ $v' = v_c' + \rho' \sin \theta'$	$I(u, v) = I'(x, y) \times (1 - dx) \times (1 - dy) + I'(x + 1, y) \times dx \times (1 - dy) + I'(x, y + 1) \times (1 - dx) \times dy + I'(x + 1, y + 1) \times dx \times dy$	1 square root 1 arctan 15 multiply 10 add
[2]	$\rho^2 = (u - u_c)^2 + (v - v_c)^2$ $u' = u'_c + (c_0 + c_1 \rho^2 + c_2 \rho^4 + c_3 \rho^6 + c_4 \rho^8)(u - u_c)$ $v' = v'_c + (c_0 + c_1 \rho^2 + c_2 \rho^4 + c_3 \rho^6 + c_4 \rho^8)(v - v_c)$			$I(u, v) = I'(x, y) \times (1 - dx) \times (1 - dy) + I'(x + 1, y) \times dx \times (1 - dy) + I'(x, y + 1) \times (1 - dx) \times dy + I'(x + 1, y + 1) \times dx \times dy$	19 multiply 14 add
[1]	$\rho^2 = (u - u_c)^2 + (v - v_c)^2$ $u' = u'_c + (c_{n-1} + c_n \rho^2 * f^{n-1}(\rho^2)) \times (u - u_c)$ $v' = v'_c + (c_{n-1} + c_n \rho^2 * f^{n-1}(\rho^2)) \times (v - v_c)$			$I(u, v) = dx \times \{ dy \times [I'(x + 1, y + 1) - I'(x, y + 1) - I'(x + 1, y) + I'(x, y)] + I'(x + 1, y) - I'(x, y) \} + d \times [I'(x, y + 1) - I'(x, y)] + I'(x, y)$	11 multiply 17 add
This Paper	$\rho^2 = (u - u_c)^2 + (v - v_c)^2$ $u' = u'_c + \{[(c_0 + c_1 \rho) + (c_2 + c_3 \rho) \rho^2] + c_4 \rho^4\} \times (u - u_c)$ $v' = v'_c + \{[(c_0 + c_1 \rho) + (c_2 + c_3 \rho) \rho^2] + c_4 \rho^4\} \times (v - v_c)$			$I(u, v) = dx \times \{ dy \times [I'(x + 1, y + 1) - I'(x, y + 1) - I'(x + 1, y) + I'(x, y)] + I'(x + 1, y) - I'(x, y) \} + d \times [I'(x, y + 1) - I'(x, y)] + I'(x, y)$	11 multiply 17 add

III. VLSI ARCHITECTURE

Fig. 1 shows the block diagram of the architecture of the BDC VLSI circuit. It consists of two main parts:

- 1] Combined Mapping Unit
- 2] Simplified Linear Interpolation Unit

The details of each part will be described as given below,

Fig. 1: Block diagram of the architecture for proposed distortion correction VLSI circuit

A. Combined Mapping Unit

For each pixel (u, v) in CIS the function of the combined mapping unit is to calculate the position of (u', v') in DIS. By employing the odd-order approximated polynomial and Horner's algorithm, the back-mapping results can be obtained by expanding [9] as

$$u' = u + c_0 + c_1 u + c_2 u^2 + c_3 u^3 + c_4 u^4 \quad (16a)$$

$$v' = v + c_0 + c_1 v + c_2 v^2 + c_3 v^3 + c_4 v^4 \quad (16b)$$

where c_0, c_1, c_2, c_3 and c_4 are combined mapping coefficients. Since the coefficients c_1, c_2, c_3 and c_4 are the coefficients of the terms ρ^2, ρ^4, ρ^6 and ρ^8 , the values of them are very much less than 1. Table III gives the set of the five combined mapping coefficients obtained in the current research [1].

TABLE III

COMBINED MAPPING COEFFICIENTS

c_0	0
c_1	$5.559280514717 \times 10^{-3}$
c_2	1.11027×10^{-10}
c_3	$-3.392835357017216 \times 10^{-17}$
c_4	$2.173691884067 \times 10^{-23}$

In order to condense the word length of the proposed hardware architecture, the major operations of (16a) and (16b) can be rewritten in terms of scaling operation as given by,

$$\begin{aligned}
 c_0 + \rho^2(c_1 + \rho^2(c_2 + \rho^2(c_3 + \rho^2 c_4))) &= c_0 + (\rho^2 \times 2^{-16}) \\
 \times \{ & (c_1 \times 2^{16}) + (\rho^2 \times 2^{-16})[(c_2 \times 2^{32}) + (\rho^2 \times 2^{-16})(c_3 \times 2^{48}) + (\rho^2 \times 2^{-16}) \times (c_4 \times 2^{64})] \} \\
 &= c_0 + (\rho^2 \times 2^{-16}) \times (c_1 \times 2^{16}) + (\rho^4 \times 2^{-32}) \times (c_2 \times 2^{32}) + (\rho^6 \times 2^{-48}) \times (c_3 \times 2^{48}) + (\rho^8 \times 2^{-64}) \times (c_4 \times 2^{64})
 \end{aligned} \quad (17)$$

By way of performing this manipulation, not only the values of the power terms ρ^2, ρ^4, ρ^6 and ρ^8 are scaled down but also the word length of the coefficients c_1, c_2, c_3 and c_4 are limited. The critical path of the combined mapping unit design can be shortened with the 20-stage pipelined architecture.

B. Simplified Linear Interpolation Unit

The objective of linear interpolation is to find the intensity value $I(u, v)$ of location (u', v') from the intensity values of four neighbouring pixels: $I(x, y)$, $I(x + 1, y)$, $I(x, y + 1)$, and $I(x + 1, y + 1)$. The function of linear interpolation is simplified by the algebraic manipulation as shown in (15). Fig. 3 shows the architecture of the simplified linear interpolation unit with the 11-stage pipelined architecture of the simplified linear interpolation unit [1]. In the figure, the multipliers are two-stage pipelined multipliers and the shadowed rectangles represent registers.

Fig. 4 shows the architecture of neighbouring pixels reading circuit and fractions obtaining circuit. The u' and v' are obtained from stage 20 of the combined mapping unit. The length of u' and v' is 24-bit, and the highest 16 of 24 bits are integer parts and the lowest 8 of 24 bits are fractional parts. Thus, the register value of x or y vector can be obtained directly by connecting wires with the highest 16-bit of the u' or v' register. The register value of dx or dy can be obtained from the lowest 8-bit of the u' or v' register in the same way. The size of each DIS RAM is the same as the size of the input image and the four intensity values of $I(x, y)$, $I(x + 1, y)$, $I(x, y + 1)$, and $I(x + 1, y + 1)$ can be read simultaneously from the four DIS RAMs.

C. Time Multiplexed Architectures

As described in Section III, the back-mapping procedure requires the most computing resource of the distortion correcting procedure. In Back-Mapping, the evaluation of polynomial can be described as an iteration function, which gives flexibility in implementing circuit with different architecture by time multiplexed technique [8],[9].

The fig. 5 shows three architectures of the Back-Mapping circuit. Since the back-mapping step can be described as an iteration function as shown in (16a) and (16b), the hardware architecture can be implemented as one-cycle, two-cycle, or four-cycle to complete the

back-mapping procedure. Fig. 5(a) shows the one-cycle architecture of back-mapping circuit, the same as the stages from 6 to 17 of the combined mapping unit in Fig. 2, which costs four pipelined multipliers and four adders. It can obtain one back-mapping result during each clock cycle with 12-stage pipelined architecture.

However, the hardware cost and memory requirement are quite a few. Fig. 5(b) and (c) shows the two-cycle and four-cycle architectures of backmapping circuit, respectively. The hardware costs of these architectures are less than that of one-cycle. Nevertheless, the execution time to get one back-mapping result needs two or four cycles. Fig. 6 shows the three architectures of the memory reading circuit for reading the identity values of four neighbouring pixels of location (u', v') . The Table IV gives the detailed comparisons between the hardware resources of previous architectures of distortion correction designs.

Fig. 2 : Architecture of the combined mapping unit.

Fig. 3 : Architecture of the simplified linear interpolation unit.

Fig. 4 : Architecture of neighboring pixels reading and fractions obtaining circuit.

Fig. 5 : Architectures of back-mapping circuit with (a) one-cycle, (b) twocycles, (c) four-cycle to complete a back-mapping procedure.

Fig. 6 : Architecture of memory reading circuit with (a) one-cycle, (b) two cycle, (c) four-cycle to complete reading four neighbouring pixel values.

TABLE IV
HARDWARE RESOURCES OF VARIOUS
DISTORTION CORRECTING ARCHITECTURES

	[1]	[2]	[3]	This Paper		
				One-Cycle	Two-Cycle	Three-Cycle
Square root	0	1	0	0	0	0
Arc-tangent	0	1	0	0	0	0
24 × 24 multiplier	1	7	7	4	2	1
16 × 24 multiplier	2	0	2	2	2	2
16 × 16 multiplier	2	0	2	2	2	2
8 × 8 multiplier	2	8	8	3	3	3
Adder	13	10	14	17	15	14

IV. EXPERIMENTAL RESULTS

The proposed algorithm was simulated using MATLAB R 2010a for barrel distorted Lena image and the result is given in fig.1. The same algorithm was implemented using SYNOPSIS Design Vision. The synthesis results shows that the proposed distortion correction circuit contains 13917 gates, which is same as [1], but the throughput is increased by 75%. The hardware emulation result for endoscopic images is shown in fig 2.

Fig 1. Correction of barrel distorted Lena image.

Fig.2. Hardware emulation image results of correcting barrel distortion in endoscopic images. (a) Distorted black-point images. (b) Corrected black-point image by this paper. (c) Distorted endoscopic image; (d) corrected Image. The white streak on the top-left of Image (c) appears too far away and the center of the bifurcation appears too close to the camera, due to the distortion. Image (d) is in proper perspective, since the distortion is corrected. (e) Gastrointestinal image captured by video-endoscope. (f) Corrected gastrointestinal image by this paper.

V. CONCLUSION

In this paper, efficient and low memory VLSI architecture of barrel distortion correcting circuit was presented for biomedical endoscope applications. The computational complexity of correcting functions is decreased by Estrin's algorithm and the algebraic manipulation of the linear interpolation. Further, the time multiplexed design provides different architectures for satisfying different applications. Comparing with other low-complexity architectures, this paper reduces at least 75% hardware cost and 70% memory requirement than other VLSI correcting designs.

REFERENCES

- [1] Shih-Lun Chen, Hong-Yi Huang, and Ching-Hsing Luo, "Time Multiplexed VLSI Architecture for Real-Time Barrel Distortion Correction in Video-Endoscopic images," *IEEE Trans. On Circuits and Systems*, vol. 21, no. 11, pp. 1612–621, Nov. 2011.
- [2] 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 Trans. Circuits Syst. II Express Briefs*, vol. 56, no. 1, pp. 51–55, Jan. 2009.
- [3] J. Kannala and S. S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 8, pp. 1335–1340, Aug. 2006.
- [4] H. T. Ngo and V. K. Asari, "A pipelined architecture for real-time correction of barrel distortion in wide-angle camera images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 3, pp. 436–444, Mar. 2005.
- [5] J. P. Helferty, C. Zhang, G. McLennan, and W. E. Higgins, "Videoendoscopic distortion correction and its application to virtual guidance of endoscopy," *IEEE Trans. Med. Imaging*, vol. 20, no. 7, pp. 605–617, Jul. 2001.
- [6] K. V. Asari, S. Kumar, and D. Radhakrishnan, "A new approach for nonlinear distortion correction in endoscopic images based on least squares estimation," *IEEE Trans. Med. Imaging*, vol. 18, no. 4, pp. 345–354, Apr. 1999.
- [7] V. K. Asari, "Design of an efficient VLSI architecture for non-linear spatial warping of wide-angle camera image," *J. Syst. Architect.*, vol. 50, pp. 743–755, Aug. 2004.
- [8] P. Tummeltshammer, C. Hoe, and M. Puschel, "Time-multiplexed multiple-constant multiplication," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 26, no. 9, pp. 1551–1563, Sep. 2007.
- [9] Y. S. Kwon and C. M. Kyung, "Performance-driven event-based synchronization for multi-FPGA simulation accelerator with event timemultiplexing bus," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 24, no. 9, pp. 1444–1456, Sep. 2005.
- [10] J. D. Bruguera, N. Guil, T. Lang, J. Villalba, and E. L. Zapata, "CORDIC based parallel/pipelined architecture for the Hough transform," *J. VLSI Signal Process.*, pp. 207–221, Jan. 2001.