Adaptive-Quantization Digital Image Sensor for Low-Power Image Compression

Chen Shoushun, Amine Bermak, *Senior Member, IEEE*, Wang Yan, and Dominique Martinez

***Abstract*—The recent emergence of new applications in the area of wireless video sensor network and ultra-low-power biomedical applications (such as the wireless camera pill) have created new design challenges and frontiers requiring extensive research work. In such applications, it is often required to capture a large amount of data and process them in real time while the hardware is constrained to take very little physical space and to consume very little power. This is only possible using custom single-chip solutions integrating image sensor and hardware-friendly image compression algorithms. This paper proposes an adaptive quantization scheme based on boundary adaptation procedure followed by an online quadrant tree decomposition processing enabling low power and yet robust and compact image compression processor integrated together with a digital CMOS image sensor. The image sensor chip has been implemented using 0.35-μm CMOS technology and operates at 3.3 V. Simulation and experimental results show compression figures corresponding to 0.6–0.8 bit per pixel, while maintaining reasonable peak signal-to-noise ratio levels and very low operating power consumption. In addition, the proposed compression processor is expected to benefit significantly from higher resolution and Megapixels CMOS imaging technology.**

I. Introduction

THE *wide spread* of today's mobile and portable devices, wireless sensor network technologies as well as cutting-edge biomedical microsystems, such as the camera micro-pill [1], require imaging front-end that acquire the image, process, and transmit data using very low power. The last decade has witnessed a very rapid emergence of CMOS imaging technology as the technology of choice for portable digital imaging products [2], [3]. Standard CMOS fabrication process has enabled the concept of a camera-on-chip, resulting in reduced manufacturing costs, compactness, and low-power operation [2], [3]. Advanced deep-submicrometer technologies have enabled higher resolution and higher frame-rate image sensors featuring improved image and video quality but at the expense of increased output bandwidth. For portable wireless video sensors, this increased output data rate translates into higher transmission power dissipation, wider channel bandwidth, and increased memory size [4]. Image compression relaxes these requirements but, unfortunately, at the cost of additional complex processing. Indeed, image compression is the most expensive hardware in digital video camera [5]. A number of on-chip prototypes for image compression have been recently proposed, which in some cases even include image sensors [5]–[13]. Unfortunately, image compression remains a very challenging task which, even if implemented on-chip, would result in high power consumption and large silicon area. This would limit the prospect of implementing low-power image acquisition and image compression on a single chip.

In this paper, an adaptive quantization scheme based on a boundary adaptation procedure followed by an efficient online quadrant tree decomposition algorithm is proposed to achieve low-power and yet robust image compression integrated together with a digital CMOS image sensor. The complexity and power consumption of the compression processor is independent of the size of the imager, making it very attractive for high-resolution and high-frame-rate image sensors. In our proposed architecture, the image is first acquired using a time-domain CMOS digital pixel sensor array followed by an adaptive quantization scheme that permits to compress the data to a lower number of bits [typically 1–2 bits-per-pixel (BPP)]. Further compression is accomplished, while scanning out the pixel values, using the quadrant tree decomposition (QTD) algorithm. QTD compresses spatially redundant data in the binary image and allows to achieve $\leq$1 BPP, without any further degradation of the image quality as no comparison with a threshold is required. This is mainly true because of the binary nature of the image data at the output of the adaptive quantizer.

The remainder of this paper is organized as follows. Section II introduces the algorithmic considerations for both the adaptive quantization and the QTD compression algorithms. The compression performance expressed in terms of BPP and the image quality expressed in terms of PSNR are also reported in this section. Section III describes the VLSI architecture, while Section IV reports the experimental results obtained from the prototype chip. Section V concludes this study.

II. Algorithmic Considerations

In a video communication application, a pair of encoder and decoder is required. The image encoder converts, at each time step n, a sampled version u_n of the pixel value into a digital form J_n. The codeword is then transmitted over the channel C to the decoder, which reconstructs the pixel value $\hat{u}_n$, as close as possible to the original image source. The most efficient way to handle nonstationary signals, such as pixel intensity, is to continuously adapt the encoder/decoder pair. In backward adaptation, the transmitted codeword is used to adjust the encoder parameters. Backward adaptation is of primary interest because it does not require side information and hence no additional bit

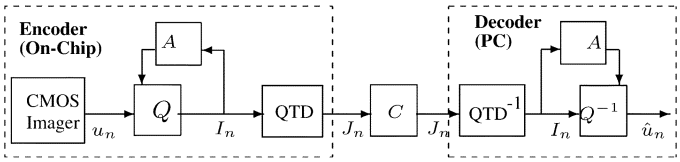


Fig. 1. Block diagram of the proposed adaptive quantizer Q/QTD compression. Q, A, QTD, C, QTD^{-1}, and Q^{-1} represent the quantizer, the adaptation block, the QTD block, the channel, the QTD inverse, and the quantizer inverse, respectively. The CMOS image sensor is first used to acquire the image data which are then quantized using an adaptive Q. The chip implements both the image capture and the encoder which consists of an adaptive Q/QTD compression block. Image reconstruction is done on the decoder side using inverse operation. The quantizer inverse Q^{-1} presents a symmetrical structure in order to track the encoder adaptive parameters.

is needed. Fig. 1 shows the proposed backward adaptive quantizer/compression system. The image is first acquired using a CMOS digital pixel sensor array. Once this is done, the content of the nondestructive on-pixel memories is scanned. The adaptive quantizer Q then generates the digital codeword I_n for the input pixel, at each time step n. A minimum of 1 bit is required for I_n and, hence, a maximum of $1/8$ compression ratio can be achieved in the first stage and for an image initially encoded with 8 bits per pixel. Following adaptive quantization, a QTD algorithm is performed online in order to compress further any spatial redundancy in the quantized array values. Interestingly, this is achieved without any further degradation of the quantized image as QTD is performed on the digital codeword, i.e., binary values. The CMOS imager, the adaptive quantizer Q, as well as the QTD processing are all performed on-chip, while the reconstruction procedure is performed off-line using a PC. The latter performs the QTD inverse (QTD^{-1}) and the decoding operation before displaying the reconstructed frame.

A. Backward Adaptive Quantization

The quantizer, illustrated by Q in Fig. 1, is specified by an ordered set of boundary points $x_0 < x_1 < \cdots < x_{i-1} < x_i < \cdots < x_{N-1} < x_N$ delimiting N disjoint quantization intervals $R_1, \ldots, R_i, \ldots, R_N$, with $R_i = [x_{i-1}, x_i]$. The size of the quantization interval i is noted by $\delta_i = (x_i - x_{i-1})$. The quantizer maps pixel intensity u_n sampled at time n into one of N quantization levels $y_i, i = 1 \ldots N$, such that

$$\hat{u}_n = \sum_{i=1}^{N} y_i \, \mathbb{1}_{R_i}(u_n)$$

with $\mathbb{1}_{R_i}(u_n) = 1$ if $u_n \in R_i$ and $= 0$ otherwise. The quantizer output I_n is defined by the N-bit binary vector $(\mathbb{1}_{R_1}, \ldots, \mathbb{1}_{R_N})$, although a more compact representation J_n is actually obtained by an additional processing stage for transmission (see Fig. 1 and Section II-B). The reconstruction levels y_i are taken as the midpoints of their corresponding quantization intervals: $y_i = (x_{i-1} + x_i)/2$. The boundary points delimiting the quantization intervals are therefore the only parameters to adapt.

The adaptation block is illustrated by A in Fig. 1. In our quantizer, the extreme boundary points x_0 and x_N are fixed by the quantization range, but the other boundary points from x_1 to

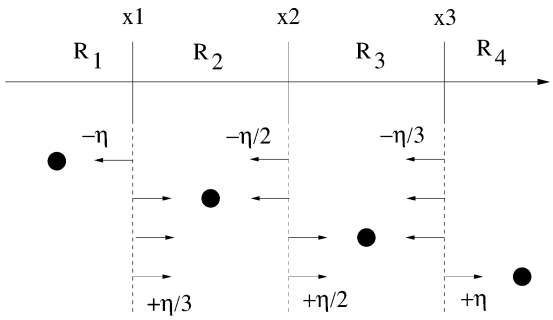


Fig. 2. Example of the backward adaptation FBAR_0 for a 2-bit quantizer. There are three adaptive boundary points x_1, x_2, and x_3 delimiting four quantization intervals from R_1 to R_4. The reconstructed values are taken as the midpoint of their corresponding intervals, $y_2 = (x_1 + x_2)/2$ and $y_3 = (x_2 + x_3)/2$, except for the extreme intervals for which $y_1 = x_1$ and $y_4 = x_3$. At each time step, the input pixel intensity falls within a given interval. Thus, there are four cases to be considered. For each one, the active interval is indicated by a black dot in the figure. Each boundary point is shifted in the direction of the active interval by a quantity η divided by the number of bins on this side. When the interval R_1 is active, the boundary points x_1, x_2, and x_3 decrease by $-\eta, -\eta/2$, and $-\eta/3$, respectively (see the first adaptation row in the figure). When R_2 is active, x_1 increases by $+\eta/3$ and x_2 and x_3 decrease by $-\eta/2$ and $-\eta/3$, respectively (see the second adaptation row in the figure). When R_3 is active, x_1 and x_2 increase by $+\eta/3$ and $+\eta/2$ and x_3 decreases by $-\eta/3$ (see the third adaptation row in the figure). When R_4 is active, x_1, x_2, and x_3 increase by $+\eta/3, +\eta/2$, and $+\eta$, respectively (see the last adaptation row in the figure).

x_{N-1} are parameters that change over time. Because the decoder has a structure similar to the encoder, the same adaptation rule is implemented at both sides of the channel. At each time step n, the transmitted codeword J_n is used to adjust the quantizing parameters (backward adaptation)

$$\Delta x_i = x_i(n) - x_i(n-1) \tag{1}$$

where $i = 1 \ldots N-1$. The backward adaptation rule, called FBAR_r for the Fast Boundary Adaptation Rule, is obtained by updating all boundary points at each time step

$$\Delta x_i = \frac{\eta}{N-i} \sum_{k=i+1}^{N} \delta_k^r \mathbb{1}_{R_k} - \frac{\eta}{i} \sum_{k=1}^{i} \delta_k^r \mathbb{1}_{R_k} \tag{2}$$

where η is the step size, a positive scalar.

It has been shown that FBAR_r given by (2) minimizes the rth power law distortion D_r [14], e.g., the mean absolute error when $r = 1$ or the mean square error when $r = 2$. At convergence, all of the N quantization intervals R_i will have the same distortion $D_r(i) = D_r/N$. This property guarantees an optimal high-resolution quantization [15]. Although the maximization of the signal-to-noise ratio (SNR) implies the minimization of the mean square error, FBAR_r with $r = 0$ was used in our implementation because of its simplicity. Indeed, FBAR_0 now reduces to

$$\Delta x_i = \frac{\eta}{N-i} \sum_{k=i+1}^{N} \mathbb{1}_{R_k} - \frac{\eta}{i} \sum_{k=1}^{i} \mathbb{1}_{R_k} \tag{3}$$

in which the size of the intervals is no longer taken into account.

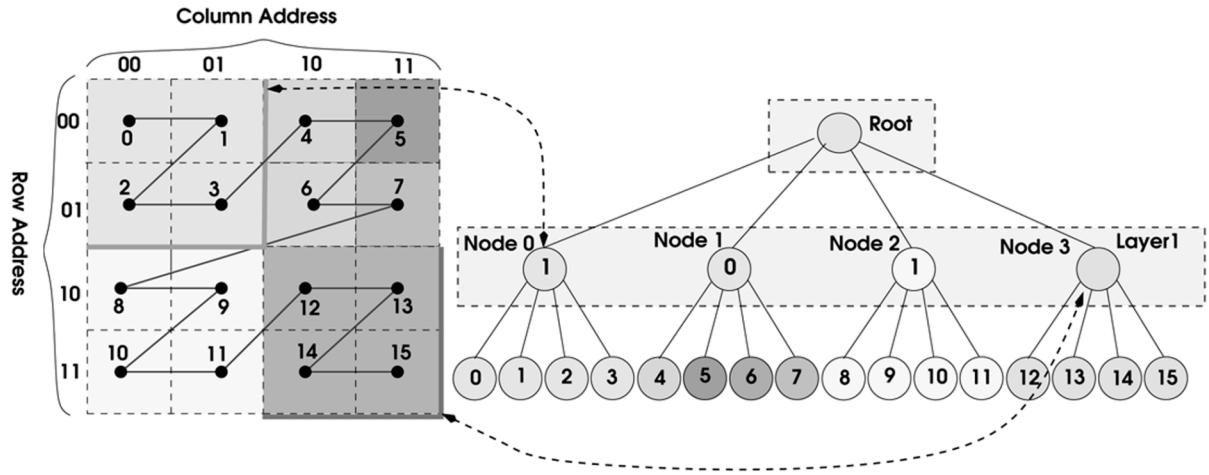


Fig. 3. Addressing strategy of the pixel array and its corresponding tree structure.

Equation (3) can be understood by noting that i and $N-i$ correspond to the number of quantization intervals at the left and the right sides of the boundary point x_i. Let us first assume that, at a given time step, the input pixel intensity falls within a given interval located at the left of x_i. We have $\sum_{k=1}^{i} \mathbb{1}_{R_k} = 1, \sum_{k=i+1}^{N} \mathbb{1}_{R_k} = 0$ and, given (3), $\Delta x_i = -\eta/i$. Thus, the boundary point x_i is left-shifted by the quantity $-\eta$ divided by the number of bins i located on its left side. When the input falls within a given interval located at the right, we have $\sum_{k=1}^{i} \mathbb{1}_{R_k} = 0, \sum_{k=i+1}^{N} \mathbb{1}_{R_k} = 1$, and $\Delta x_i = \eta/(N-i)$. The boundary point x_i is now right-shifted by the quantity η divided by the number of bins $N-i$ located on its right side. This is illustrated in Fig. 2 for a 2-bit quantizer. It can be shown that FBAR_0 given by (3) leads to an equiprobable quantization [14], i.e., at convergence the probability of being active is $1/N$ for all the intervals. For a 1-bit quantizer, (3) reduces to

$$\Delta x = \eta(\mathbb{1}_{R_2} - \mathbb{1}_{R_1})$$

where x is the unique boundary point, and R_1 and R_2 are the left and right quantization intervals, respectively. At each time step, x is thus increased or decreased by $\Delta x = \pm\eta$. At convergence, we have on average $\langle \mathbb{1}_{R_2} \rangle = \langle \mathbb{1}_{R_1} \rangle$ and $\langle \Delta x \rangle = 0$. The boundary point x oscillates around the median value of the input so that the probability of having either R_1 or R_2 active is $1/2$.

B. QTD Lossless Compression

The compression procedure based on the QTD [16] algorithm is performed by building a tree, which contains spatially redundant data in the quantized image. Multiple hierarchical layers of the tree, corresponding each to a square block within the pixel array, are constructed hierarchically in one iteration while the pixels are being scanned and quantized. Therefore, the proposed VLSI architecture enables one iteration adaptive quantization as well as the construction of the entire hierarchical structure of the quadrant tree during the scanning phase of the imager. Fig. 3 describes the addressing strategy of the array, which permits to systematically construct the corresponding QTD tree structure. For the sake of clarity, only a 4×4 pixel array is illustrated; however, the same concept is extended to any pixel array size. First, the array is divided into four quadrants. Each quadrant will be allocated a 2-bit binary code and associated with a leaf within the primary tree, as illustrated in Fig. 3.

Each quadrant is further divided into four subblocks and two more bits are used in order to encode the new subtree. The overall structure now presents a root and 16 leaves, each of them are encoded using a 4-bit address. The procedure is repeated in a hierarchical way until the image is segmented down to the pixel level. The leaves of the tree correspond to the pixels of the array image, and each hierarchical level within the tree corresponds to a quadrant grouping of the image array. For an $n \times n$ image array, $\log_2 n$ layers are required to construct the tree. One can note that the address of the leaves, i.e., the pixels, are sequentially addressed from left to right. An important and interesting feature in this addressing methodology is the mapping relationship between the pixels' address in the tree and that in the pixel array. It can be easily observed that the row and column addresses are the even and the odd bits of the pixel's tree address, respectively. For example, a leaf pixel in the tree with an address 'b010110 corresponds to a pixel with row address 'b001 and column address 'b110 in the array. A direct mapping between the address in the tree and the one in the pixel array is therefore obtained using Morton (Z) scan [17]. The tree construction procedure described earlier appears as a bottom-up approach; however, the procedure used is in fact performed in parallel. In our prototype chip, first the bottom layer is naturally constructed by a direct mapping between the 64×64 (4096) pixel array and the tree leaves (no storage is needed). The upper layer (layer 1) is constituted of 1024 $(4096/4)$ nodes, where in each node a flag bit is stored indicating whether the corresponding 2×2 pixels children can be compressed or not. A 1-bit register is therefore required in order to store the flag bits in 1024 flip-flops organized as a memory bank. The flag bits of the entire tree are obtained during the scanning mode of the array. After every four cycles, the four quantized pixel values scanned are compared. If the four quantized pixel values are different, then a flag "0" is written into the corresponding register, which means that the four pixels

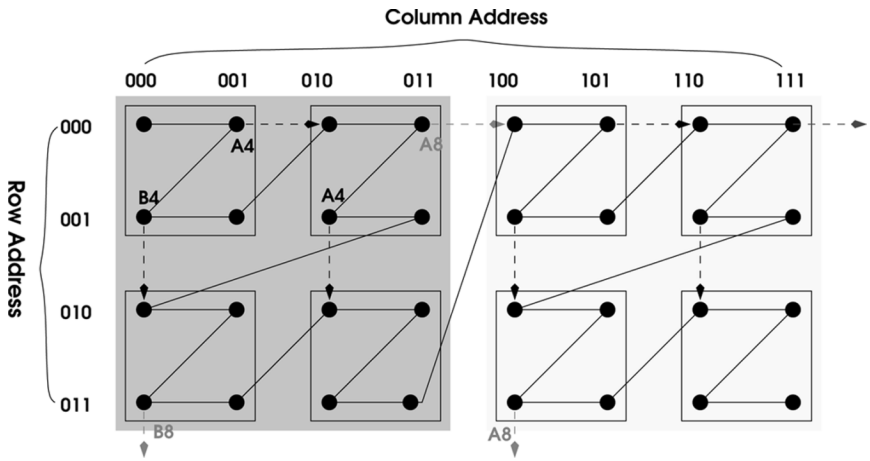


Fig. 4. Smooth boundary point propagation scheme for 4×4 and 8×8 pixel quadrants. Two registers $(A4, B4)$ are needed to store the boundary point for the 4×4 quadrant level and two other registers $(A8, B8)$ are needed to store those related to the 8×8 quadrant level.

cannot be represented by a single value (i.e., it is noncompressible). The second layer of the tree includes 256 nodes, where each node groups four elements of the first layer. At each node, a flag bit is once again used in order to indicate whether the four nodes (16 pixels) can be compressed or not. The same procedure is used to build up the flag vector of the second layer. This is done in parallel with the construction of the flag vector of the first layer by comparing the 16 quantized pixel values scanned out. Similarly, a flag "0" is loaded into the corresponding register, if the 16 pixels cannot be compressed and a flag value of "1" is loaded otherwise. The procedure is carried out in parallel and the root of the tree is reached one clock cycle after the last pixel is read-out. The tree is therefore built in one iteration and during the scanning procedure of the pixel array. The flag bit located at the root of the tree would indicate whether the entire image can be compressed into a single value, which will happen only if all the quantized pixel values are the same. QTD attempts to remove spatial redundancy by compressing similar pixel intensities belonging to the same block and representing them by a single value. It is very important to note that when pixels present similar intensity values, the adaptive 1-bit Q will converge and hence enters into an oscillation mode in which sequences of "$010101\ldots$" are generated. Thus, in fact, removing redundancy at the output of the 1-bit adaptive Q is equivalent to looking at oscillation rather than looking at similar quantized pixel values. This can be simply realized using a flip-flop and an XNOR gate, as will be described in Section II-C.

C. Smooth Boundary Point Propagation

The Morton (Z) scan strategy [17] is a quadrant or window-based read-out, which is compatible with the QTD algorithm. A direct mapping is obtained between the QTD tree structure and the pixel array using odd and even addresses as explained in the previous section and reported in [17]. In addition, the addressing requirement for a Morton (Z) scan can be very easily implemented in hardware. Indeed, Morton (Z) addressing for an $n \times n$ image array can be implemented in hardware using a single $2 \times \log_2 n$ bits counter while feeding the even and odd bits of the counter to the row and column address decoders, respectively. This results in significantly simplified addressing strategy. Unfortunately, the Morton (Z) scan presents a serious drawback when combined with the adaptive Q, presented earlier. The transition from one quadrant to the next involves jumping to a non-neighboring pixel resulting in spatial discontinuity affecting the performance of the adaptive Q. Due to the inherent hierarchical partition of the QTD algorithm, this transition gets larger and larger when scanning the array. As a consequence, one can expect sharp deviations in the pixel's values during transitions from one quadrant to another. This will introduce large errors in the adaptive Q at the edge of the quadrants. To address this problem, we propose a smooth boundary point propagation scheme, as shown in Fig. 4. One can note that, when the Morton (Z) scan transits from one quadrant to another, instead of taking the boundary point from the previously scanned pixel, the boundary point is taken from the physically nearest neighbor of the previous quadrant. Implementing such a scheme is not very complicated, as storing boundary points from a specific locations is repeatedly required. Only two registers are required for each tree layer in the case of a 1-bit quantizer while six registers are required for a 2-bit quantizer, as the number of boundary points involved is three times larger as compared with the 1-bit quantizer.

D. Simulation Results

The compression ratio expressed in terms of BPP as well as the quality of the compressed images expressed in PSNR were evaluated for still images and video datasets. We have evaluated the performance for 1-bit and 2-bit adaptive Q followed by QTD block. The adaptive rule FBAR_0 given by (3) is used because of its simplicity. However, performance of FBAR_0 is dependent on a particular choice for η. On the one hand, a large η is needed to track rapid fluctuations in consecutive pixel values. On the other hand, a small η is needed to avoid large amplitude oscillations at convergence. To circumvent this problem, we propose to make η adaptive using the following heuristic rule: if the active quantization interval does not change between two consecutive pixel readings, we consider that the current quantizing parameters are far from the optimum and η is then

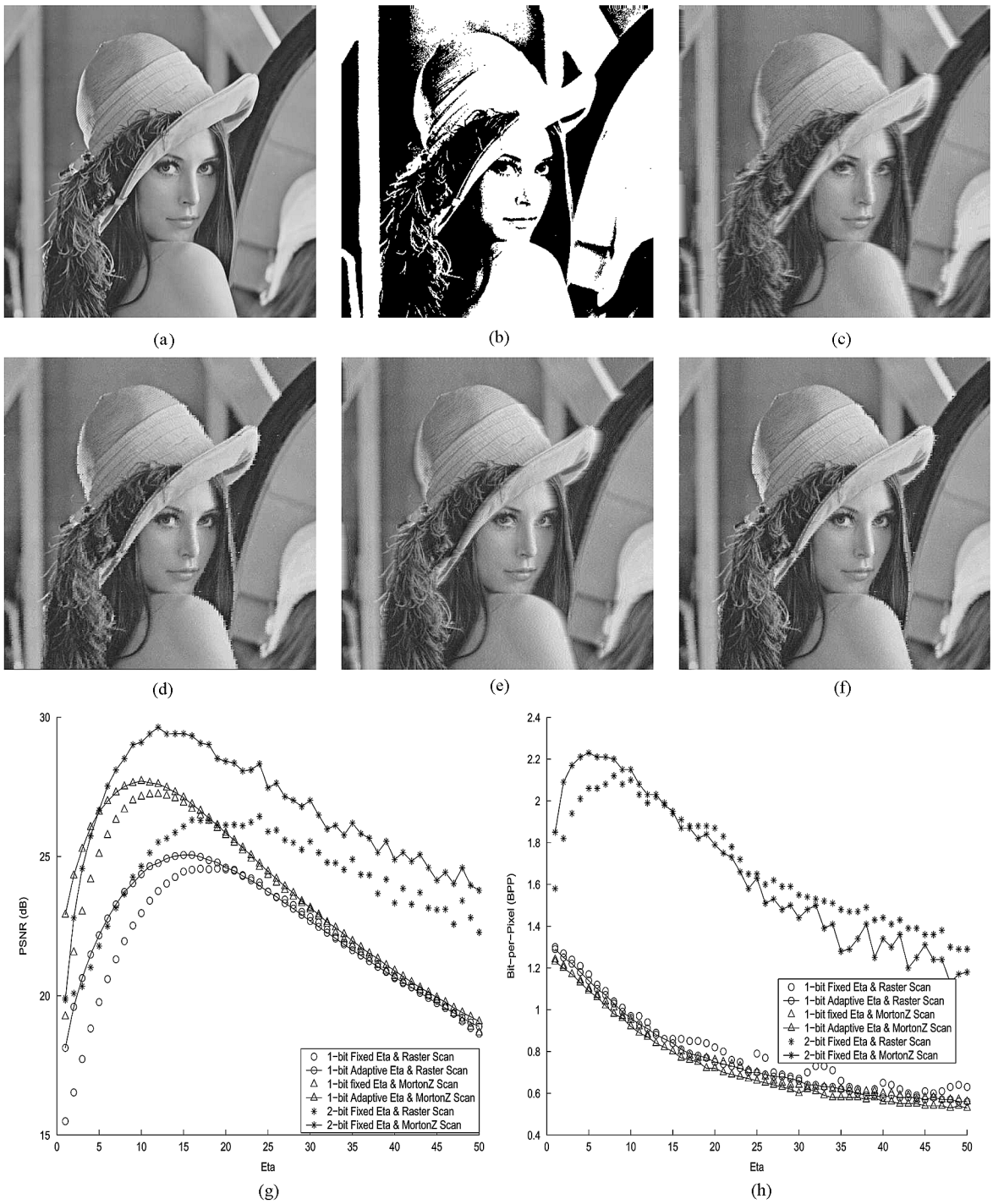


Fig. 5. Simulation results for the Lena image. (a) 256×256 original image. (b) Quantized image using a 1-bit fixed boundary point. Results for the 1-bit quantizer are using (c) fixed η raster scan, (d) 1-bit quantizer using adaptive η smooth boundary Morton (Z) scan, (e) 2-bit quantizer using raster scan, and (f) 2-bit quantizer using smooth boundary Morton (Z) scan, respectively. (g) PSNR and (h) BPP for all quantizers. The values for η used in the simulation results reported in (b)–(f) are 0, 20, 10.5, 24.5, and 13, respectively.

multiplied by $\Lambda > 1$ ($\Lambda = 1.125$ here). If the active quantization interval changes between two consecutive pixel readings, we consider that the current quantizing parameters are near the optimum, and thus η is reset to its initial value. The value of $\Lambda = 1.125$ was selected through extensive Matlab simulation. The optimum value was found to be in the range of 1.10–1.13 for all tested images. The value of 1.125 was selected because this coefficient can be simply implemented in hardware using 3-bit right shift and addition operations. We have compared the performance of FBAR_0 using fixed and adaptive η. In order to evaluate the effect of using the Morton (Z) scanning procedure as compared with a conventional raster scanning, we also compared the PSNR and the compression ratio using both scanning methodologies. Fig. 5 shows the simulation results obtained for the Lena image. The image quality is not affected by the QTD compression block as reconstruction will allow to reconstruct

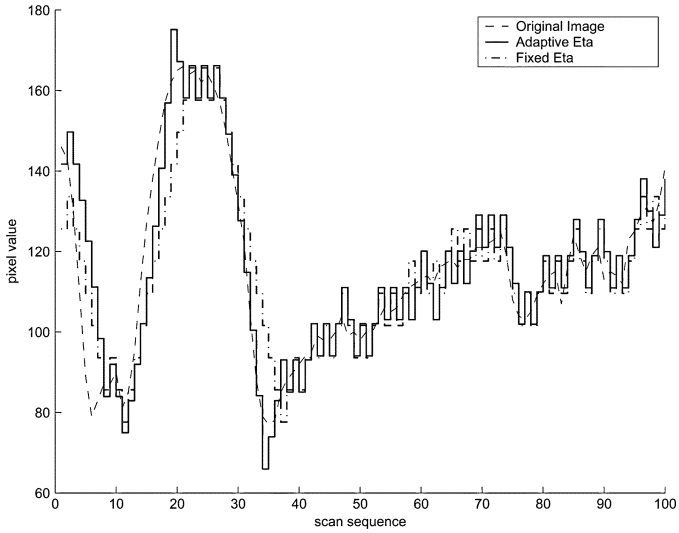


Fig. 6. Row image signal (dashed line) and the quantized values (the boundary point) in the case of fixed η (dotted line) and adaptive η (solid line). Adaptive η permits to converge faster, resulting in reduced mismatch as compared with the original signal.

exactly the same quantized image. However, the image quality as well as the compression ratios are highly constrained by the number of bits, the parameters of the adaptive Q, and the scanning methodology.

It is clear from the simulation that a 2-bit quantizer features improved PSNR as compared to a 1-bit quantizer but at the expense of lower compression performance (BPP). The adaptive η scheme was also compared with the fixed η scheme in terms of the achievable PSNR and compression ratio. It was found that the adaptive η not only provides an improved PSNR but also improved compression ratio. Improved PSNR is explained by the fact that the adaptive η uses larger adaptation steps for fast transient in the original image, while small steps are used in the case of stationary or slowly varying signals. This in turn increases the convergence speed and decreases the mismatch between the original image and its quantized counterpart. This is clearly illustrated in Fig. 6, which plots a row image signal (dashed line) and shows the quantized values (which correspond to the boundary point) in the case of fixed η (dotted line) and adaptive η (solid line). It is clear that adaptive η permits to track the input signal at a faster rate and hence reaches the oscillation stage faster in case of stationary or semistationary input signal. This will result in higher compression ratios because QTD will remove spatially redundant pixels which are actually oscillating pixels belonging to the same quadrant. This explains why both PSNR and BPP are improved simultaneously using an adaptive boundary adjustment rule.

One should also note from Fig. 5(g) and (h) that both PSNR and BPP are highly dependent upon the value of η. Using a large value for η will permit reaching oscillation faster, and, hence, improved compression ratios are obtained when using QTD. However, η cannot be increased indefinitely as it will result in a rapidly degraded PSNR performance. For most of the tested 8-bit images, the optimum η was found to be around 14–18. It is very interesting to note that better performance in terms of both PSNR and compression ratios are obtained when using our smooth boundary Morton (Z) scan methodology as compared with a raster scan, particularly in images which present spatially redundant information (images featuring large background for example). Indeed, Morton (Z) scan permits to hierarchically access square blocks of pixels presenting higher likelihood of similarity, as it is a block-based read-out strategy. Table I shows the simulation results for seven different images. The 2-bit quantizer using smooth boundary Morton (Z) scan achieves the best PSNR performance (average of 31.6 dB), while the 1-bit adaptive η using smooth boundary Morton (Z) scan achieves the lowest BPP (average of 0.62 BPP). In addition, this latter quantizer presents the highest ratio $R = (\mathrm{PSNR})/(\mathrm{BPP})$ which shows that the 1-bit adaptive η using smooth boundary Morton (Z) scan achieves the best tradeoff between image quality and compression ratio. Finally, it is important to note that our adaptive quantization scheme and the overall proposed compression scheme is a pixel-based approach. This means that adaptation is performed on-the-fly and convergence is attained rapidly. This feature is very important as it can affect the performance for video sequences. Fig. 7 reports the PSNR values as function of the frame number for both the Miss America and Claire sequences. Since the adaptation is pixel-based, maximum performance (or close to maximum) is attained in the first frame and the performance is not affected by changes from frame to frame, making the proposed scheme quite robust for video applications.

III. VLSI Architecture

A. Imager Architecture

Fig. 8(a) shows the block diagram of a single-chip CMOS image sensor with the adaptive Q and the QTD processor. The image array consists of 64×64 digital pixel sensors equipped with pixel-level nondestructive storage elements [18]. Each pixel is composed of a photosensitive device (reverse-biased photodiode P_d) with its internal capacitance C_d, a reset transistor, a comparator, and a feedback circuit [19]. The voltage at the sensing node of the photodiode (V_n) is first reset to V_C. After the reset phase, the light falling onto the photodiode discharges C_d, resulting in a decreasing voltage V_n across the photodiode node. The accumulated charge in the pixel is converted to a time stamp using a comparator and an SR latch [19]. The comparator constantly monitors the voltage drop across the photodiode V_n and compares it to a reference voltage V_{rf}. Once V_n reaches the reference voltage V_{rf}, the comparator triggers a pulse. The time taken from the start of the integration until the triggering of the comparator output is seen as a pulsewidth-modulated (PWM) signal. In order to convert this PWM signal into a digital code, the output pulse of the comparator is used as a write enable signal. A time stamp provided by a global timing circuit is therefore recorded into each pixel whenever the comparator is triggered [19]. In this PWM coding scheme, the illumination received by each pixel is coded using a single pulse. This represents a major advantage as switching activity is reduced to only a single transition in each frame for each pixel, thus allowing for lower power consumption and reduced switching noise [19]. The pixel array is operated in two separate phases. The first phase corresponds to the integration phase in which the illumination

TABLE I

PSNR (dB) AND BIT-PER-PIXEL (BPP) FOR SOME SAMPLE IMAGES USING 1-b Q WITH FIXED η RASTER SCAN (1-b η_0-R), 1-b Q WITH ADAPTIVE η RASTER SCAN (1-b η-R), 1-b Q USING FIXED η SMOOTH BOUNDARY MORTON (Z) SCAN (1-b η_0-MZ), 1-b Q USING ADAPTIVE η SMOOTH BOUNDARY MORTON (Z) SCAN (1-b η-MZ), 2-b Q USING RASTER SCAN (2-b η_0-R) AND 2-b Q USING SMOOTH BOUNDARY MORTON (Z) SCAN (2-b η_0-MZ). THE 2-BIT η_0-MZ Q ACHIEVES THE BEST PSNR PERFORMANCE WHILE THE 1-b η-MZ Q ACHIEVES THE LOWEST BPP. THE PSNR OF THE 1-b η-MZ Q PRESENTS THE HIGHEST PSNR WHEN COMPARED WITH THE OTHER THREE TYPES OF 1-b Q. $R = (\mathrm{PSNR})/(\mathrm{BPP})$ [dB/BPP] FOR THE AVERAGE FIGURES, WHICH IS A MEASURE OF THE TRADEOFF BETWEEN THE TWO CRITERIA (PSNR, BPP). THE 1-b η-MZ Q PRESENTS THE BEST TRADEOFF BETWEEN THE PSNR AND BPP FIGURES

Quantizer	Sample Images														Average		
	Lena		News		Cancer		Gut		Elaine		Plane		Bacteria				
	psnr	BPP	psnr	BPP	psnr	BPP	psnr	BPP	psnr	BPP	psnr	BPP	psnr	BPP	psnr	BPP	R
1-b η_0-R	22.9	0.88	22.9	0.58	26.2	0.76	29.0	0.70	26.0	0.87	28.8	0.50	28.7	0.73	26.4	0.71	37.18
1-b η-R	24.6	0.89	24.0	0.59	27.1	0.75	31.9	0.71	26.4	0.87	30.1	0.52	30.6	0.61	27.8	0.70	39.71
1-b η_0-MZ	27.1	0.72	25.9	0.56	28.5	0.69	32.2	0.58	28.5	0.73	30.1	0.49	32.6	0.73	29.2	0.64	45.62
1-b η-MZ	27.6	**0.71**	26.1	**0.56**	29.0	**0.68**	33.9	**0.57**	29.0	**0.71**	30.8	**0.50**	32.7	**0.61**	29.9	**0.62**	**48.22**
2-b η_0-R	25.4	1.93	25.2	1.28	29.5	1.77	31.0	1.63	27.5	1.95	32.5	1.05	32.9	1.57	29.0	1.59	18.24
2-b η_0-MZ	**29.9**	1.59	**28.4**	1.11	**31.0**	1.62	**34.0**	1.31	**30.2**	1.71	**33.8**	0.78	**33.9**	1.55	**31.6**	1.40	22.57

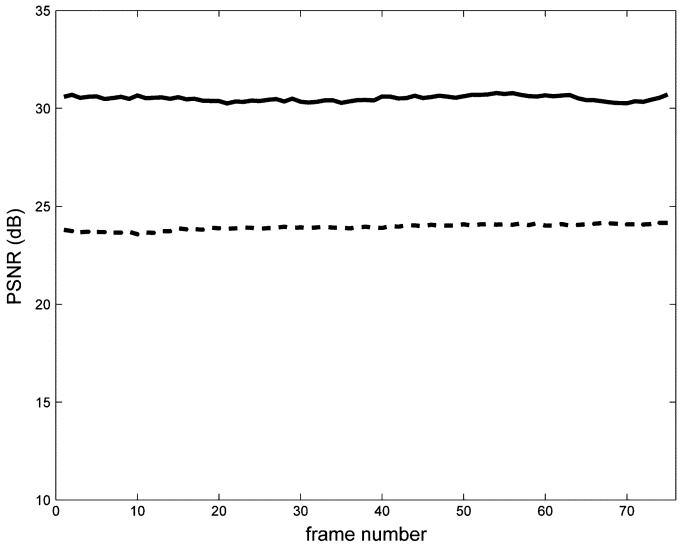


Fig. 7. Simulation results for video sequences. The solid curve is for the Miss America sequence and the dashed curve is for the Claire sequence. FBAR_0 with adaptive η and raster scanning was used for adapting the quantizer.

level is recorded and each pixel sets its own integration time by allowing the photodiode to discharge until the reference voltage is reached. This time-domain conversion results in a number of advantages such as higher dynamic range as well as improved SNR as no saturation occurs regardless of the illumination intensity within each pixel [19]. Fig. 8(b) shows the timing diagram of the sensor illustrating clearly the two operating modes (integration $(W/R = 1)$ and read-out $(W/R = 0)$ modes. During the integration phase, the global timer circuit is enabled and the pixels are operated in parallel. In fact, the pixel circuitry can be divided into an analog front-end circuit used to obtain the PWM signal and an SRAM storage element. Compared with our first-generation digital pixel sensor array [19] and in addition to the compression processor integrated on-chip, this pixel array includes a more advanced address encoding scheme enabling to implement the smooth boundary Morton (Z) scanning strategy in addition to the conventional raster scan one. The pixel layout was also improved using a more compact floor-planning strategy and an improved SRAM layout, resulting in 3% improvement in terms of fill factor as compared with the first-generation DPS [19]. Furthermore, a power-management control unit is added in order to reduce further the power consumption of the first generation imager. For instance, during the read-out phase of the SRAM memory, the pixel's analog front-end circuitry as well as the global timer are forced into a stand-by mode. Once the integration phase is performed, the pixel array can be viewed as a distributed static memory and the adaptive quantization as well as the QTD compression are performed in parallel during the read-out scanning phase. The circuit related to the adaptive quantization as well as QTD compression are detailed in the upcoming subsections.

B. Adaptive Quantization Building Block

A very interesting feature about the adaptive quantization scheme, proposed in this paper, is the fact that it can be very easily implemented using simple digital circuitry. Fig. 9 shows the diagram of the 1-bit adaptive Q (blocks within the solid line box) which includes a digital comparator $C1$, an 8-bit multiplexer MUX1, an 8-bit adder and one 8-bit register (BP Reg). As the pixel value is read from the array using a gray encoding, a gray-to-binary conversion is also required.

In a 1-bit Q, the pixel value needs to be compared with a boundary point BP which is initially set to the midrange. The boundary point is then adjusted by $\pm\eta$ depending upon the comparison result. Note that adding $\pm\eta$ depending on the comparison result is implemented in a very compact way using the output of the comparator I_n as a control input of a multiplexer, which selects between η or $\overline{\eta}$. Using the comparator output as a carry-in signal of the adder, we can obtain $-\eta = \overline{\eta} + 1$ in the case where $U_n < \mathrm{BP}$, i.e., $I_n = 0$. This results in a very compact implementation of the incrementer/decrementer circuit, as illustrated in Fig. 9. It is possible to implement an adaptive η using the circuit extension shown under the dashed line box of Fig. 9. A D-flip-flop and an XOR gate are added in order to detect if two consecutive comparison results are equal. If this is the case, the value of η is increased by a ratio set to 1.125 by selecting the right output of the multiplexer Mux2. Once the value of η is adapted, the boundary point is adjusted accordingly using the same circuit described earlier. The same circuit can be extended for a higher number of bits. For a 2-bit adaptive Q, the pixel value U_n needs to be compared with three boundary points values. The boundary points adjustment is performed in a way similar to the 1-bit adaptive Q but requires three adjustment circuits operating in parallel. While the n-bit adaptive Q requires

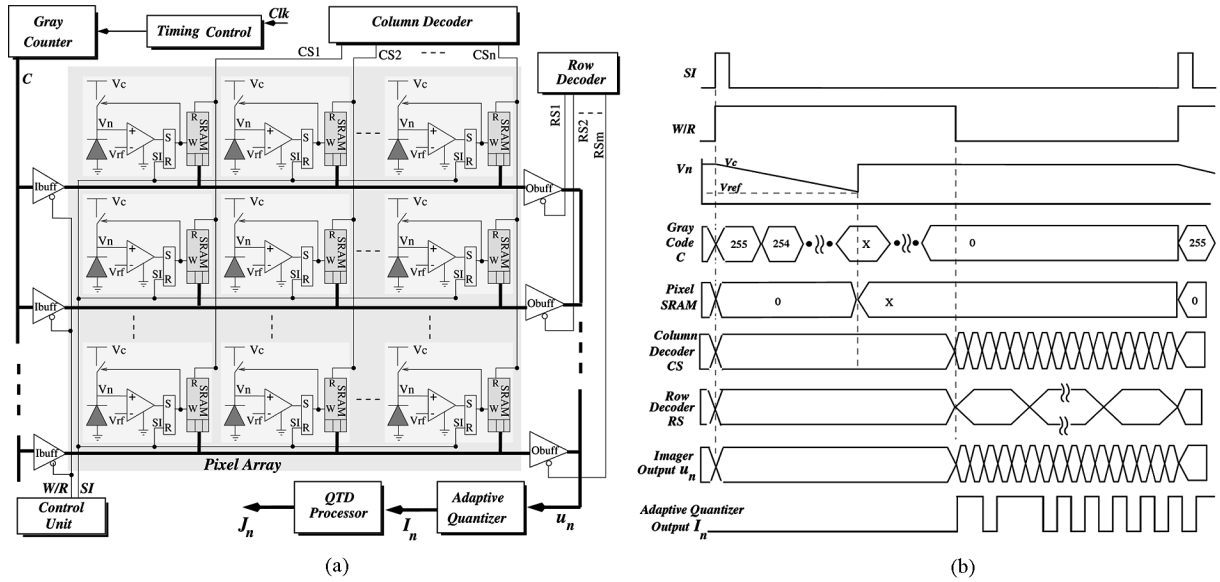


Fig. 8. (a) Block diagram of a single chip CMOS image sensor with the adaptive quantizer and the QTD processor. (b) Timing diagram showing the operation sequences and the two operation mode ($W/R = 1$ corresponds to integration and image acquisition mode while $W/R = 0$ corresponds to the read-out and image compression mode).

only very simple digital building blocks (FF, registers and comparators), it requires $2^n - 1$ comparators and BP registers.

C. QTD Building Block

Another very interesting property of our VLSI architecture is related to the fact that QTD compression procedure is performed while the array is being scanned. This is realized by constructing all of the layers of the tree structure in parallel while scanning the array. A single additional clock cycle is then required to trim the tree. This interesting feature is realized using the circuit described in Fig. 10, which shows the QTD circuit for a 1-bit Q. Note that higher number of bits are obtained using a digital comparator instead of single XNOR (equality) gate. The circuit operates in two modes, namely: 1) tree construction and 2) tree trimming modes. In the first mode, the goal is to obtain the flag bits for all layers of the tree while scanning the quantized pixel values. For the sake of simplicity, Fig. 10 shows the procedure for three layers while the circuit can be naturally extended for any tree structure.

The flip-flops shown at each layer are used to store the flag bits of the different nodes of the tree. The flag bits of the first and second layers are updated every 16 and four clock cycles, respectively. At each clock cycle, the current output value of the quantizer (I_n) is compared with the previously quantized value (I_{n-1}). If the two values are different, then the flag bit is reset. In fact, in our 1-bit quantizer, similar pixel values would correspond to an oscillation at the output of the quantizer and hence different pixel values would correspond to similar consecutive values at the output of the 1-bit quantizer. The flag bit is therefore reset if the previous and the current outputs of the quantizer (I_{n-1} and I_n) are the same. This is realized using one flip-flop and the XNOR gate shown in Fig. 10. For the first layer, the first flip-flop responsible for storing the flag corresponding to the first 4×4 pixels quadrant is enabled within the first 16 clock cycles using the 2-MSB address lines (ADDR4 and ADDR5).

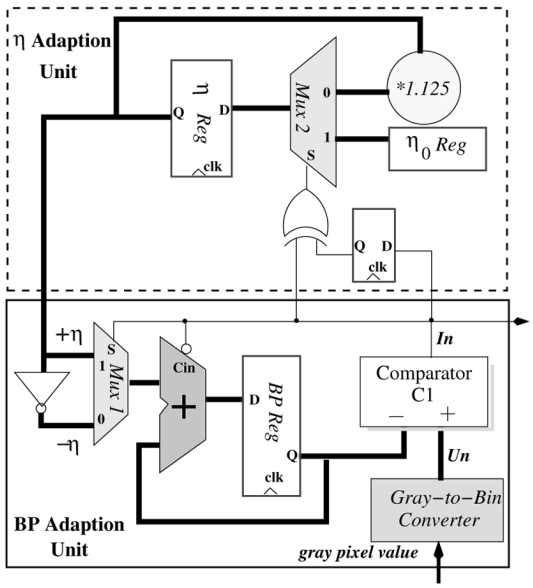


Fig. 9. 1-bit adaptive Q building block. The building blocks represented under the solid line box correspond to the fixed η adaptive Q, while the building blocks represented under the dashed line box are the circuit extension required to realize the adaptive η Q.

Subsequent flip-flops are enabled sequentially every 16 cycles using address decoder D1. Initially, all flip-flops are set to "1." If the previous and the current outputs of the quantizer (I_{n-1} and I_n) are the same for at least one cycle within the 16 clock cycles, the flag bit is reset. An OR gate is used in order to ensure the first element of each block will not affect the value of the flag bit. Indeed, for the first address value corresponding to the first element of the quadrant, the output of OR gates is low and hence the reset operation cannot be performed while scanning the first element of the quadrant. Similarly, the operation of building the flag bits for the second layer is carried out in parallel. Address

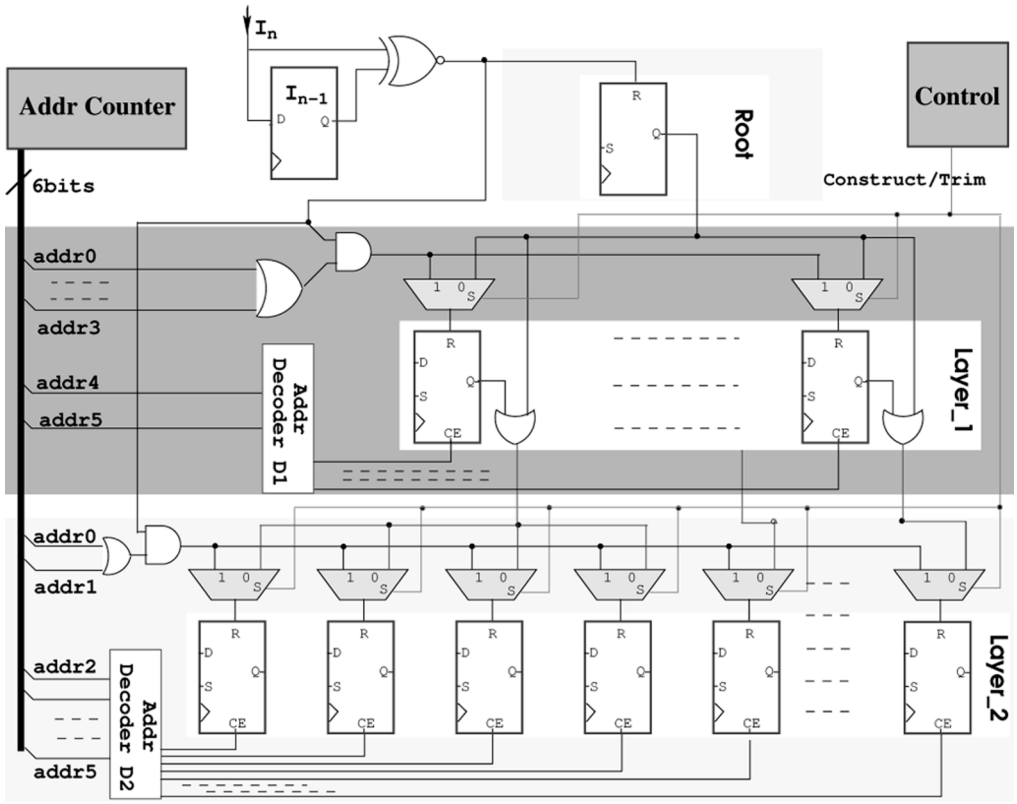
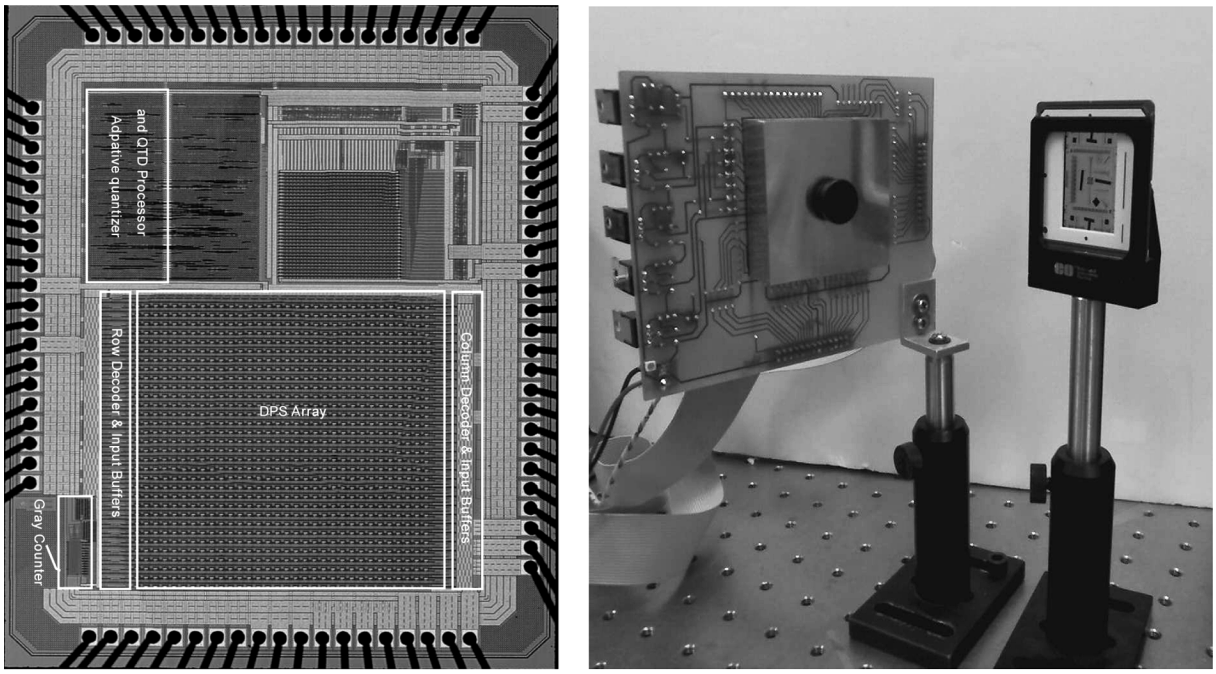


Fig. 10. QTD circuit showing three layers of a tree structure for a 1-bit Q. Note that the structure can be extended to any tree structure and it can also be extended to more than 1-bit quantizers using digital comparators instead of an XNOR gate.

decoder D2 is used to select one out of 16 flip-flops every four cycles using the 4 MSBs of the address line (ADDR2-ADDR5). Once all of the tree is constructed, the trimming mode starts and is performed in parallel for the entire tree structure using a single clock cycle. In this mode, the Const/Trim signal provided by the control circuit is set to "0," allowing to reset the flag bits of a given layer if and only if its parents present a flag bit value equal to "1." For example, if the root flag is equal to "1," all flip-flops belonging to lower layers are reset in one clock cycle using the flag bit of the root (Q of the DFF in Fig. 10). Once the tree is trimmed, first the flag bits of the tree are transmitted to the receiver end. The flag bits are used to control the read-out sequence such that compressed pixels are skipped while transmitting the compressed image data. This procedure does not require buffering the image data as the pixel array is accessed in order to retrieve noncompressed pixel values. The adaptive Q is enabled again during this phase and is performed only on noncompressed quadrants.

IV. Experimental Results

The single-chip image sensor and compression processor was implemented using 0.35-μm AMI CMOS digital process (1-poly five metal layers). Fig. 11(a) shows the layout with a total silicon area of 3.8×4.5 mm^2. It should be mentioned that the chip is a multiproject prototype, and we have highlighted in Fig. 11(a) the circuit parts related to this work. The pixel array was implemented using a full-custom approach while the digital processing parts related to adaptive Q and the QTD compression is done using automatic placement and routing tools. The digital processor which occupies an area of 1.8 mm^2 includes a large number of operating configurations such as: 1-bit and 2-bit quantizers with fixed and adaptive η, with and without QTD and using both raster and smooth boundary Morton (Z) scan. Each of these options can be applied in a modular way allowing to facilitate the test and debugging process of the prototype. The pixel array occupies around 75% of the total area dedicated to this project. One should note that if only a 1-bit or 2-bit Q followed by QTD processing is used without additional operating modes, this figure can be increased to more than 90% allowing to have most of the silicon area dedicated to the pixel array. In order to test the different building blocks of the imager, a modular test strategy was adopted. Test structures were added in order to test separately each block within the image sensor and the compression processor. An electrooptical experimental setup was developed in order to characterize the sensor array. The electrical part consists of a PCB on which the device under test (DUT) is mounted. Control signals required for the DUT are provided through National Instrument Data Acquisition board. Both the 8-bit digital output of the DPS array and the compressed image are captured and used to display the compressed and noncompressed images on the PC. All of the timing control required for both image capture and compression are generated on-chip, hence no control circuit is required, and the chip is fully operational without extra hardware. Fig. 11(b) shows the experimental PCB board including the lens mount and the connections to the data acquisition board. First, the functionality of all building blocks is tested before fully characterizing the



(a) (b)

Fig. 11. (a) Microphotograph of the prototype Chip. The prototype includes a DPS array and the compression processor. It should be noted that the design is a multiproject chip. (b) The experimental PCB board including the lens mounted on the DUT and the connections to the data acquisition board.

TABLE II
SUMMARY OF THE IMAGER PERFORMANCE

Technology	Alcatel 0.35μm, 5 metal single-poly, twin well, CMOS
Supply Voltage	3.3V
$V_C - V_{rf}$	0.5-0.8V
Pixel dynamic operating range	$>$100dB
FPN	0.8%
Pixel Area	45μm $\times$ 45μm
Pixel Array Area	$>$75%
Fill factor	18%

sensor and acquiring sample images. Table II summarizes the performance of the DPS array.

Sample 64×64 images were acquired from the prototype using different operating modes. For each mode, the compression ratio expressed in terms of BPP and the quality of the image was also measured using PSNR figures. Fig. 12 shows the acquired images with and without compression. From top to bottom, (a) represents the 8-bit captured image without compression while (b), (c), (d), and (e) represent the reconstructed compressed images using 1-bit adaptive Q with fixed η raster scan, 1-bit adaptive Q with adaptive η raster scan, 1-bit adaptive Q using adaptive η smooth boundary Morton (Z) scan and 2-bit quantizer using smooth boundary Morton (Z) scan, respectively. Visually, it is quite obvious that the 2-bit Q using smooth boundary point Morton (Z) scan presents the best image quality for all sample images. The 1-bit Q using adaptive η and smooth boundary Morton (Z) scan performs better in terms of image quality as compared to all 1-bit adaptive quantizers.

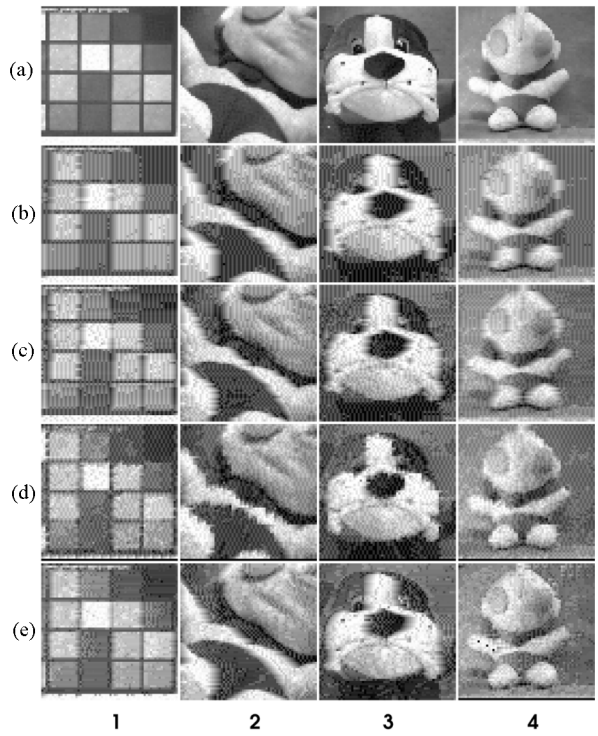


Fig. 12. Captured images under different processing modes. Row (A.) shows the 8-bit captured images without compression, (B.)–(E.) represent the reconstructed compressed images using 1-bit adaptive Q with fixed η raster scan, 1-bit adaptive Q with adaptive η raster scan, 1-bit adaptive Q using adaptive η smooth boundary Morton (Z) scan, and 2-bit Q using smooth boundary Morton (Z) scan, respectively.

The BPP and PSNR figures were also evaluated for the experimentally acquired images. Table III illustrates these figures for

TABLE III
BPP AND PSNR FIGURES FOR THE EXPERIMENTALLY CAPTURED IMAGES SHOWN IN FIG. 12. M STANDS FOR THE MODE OF OPERATION CORRESPONDING TO FIG. 12

m	Image Number							
	1		2		3		4	
B	0.8	20	0.9	20	0.9	19	0.9	22
C	0.9	20	0.9	20	0.9	19	0.9	22
D	0.8	20	0.9	21	0.8	20	0.8	23
E	1.7	22	1.9	22	1.8	22	1.8	24

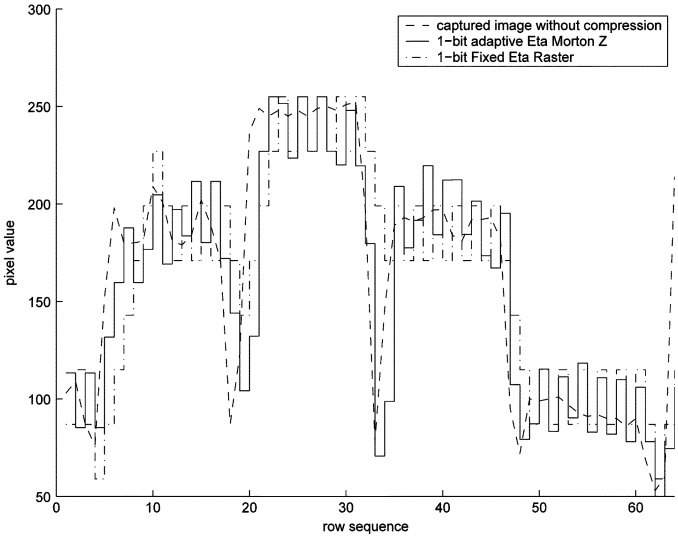


Fig. 13. Row signal of the captured image-3 without compression (dashed line) and the boundary point in the case of 1-bit adaptive Q with fixed η raster scan (dotted line) and 1-bit adaptive Q using adaptive η smooth boundary Morton (Z) scan (solid line). The last one permits to converge faster resulting in reduced mismatch as compared to the uncompressed image signal.

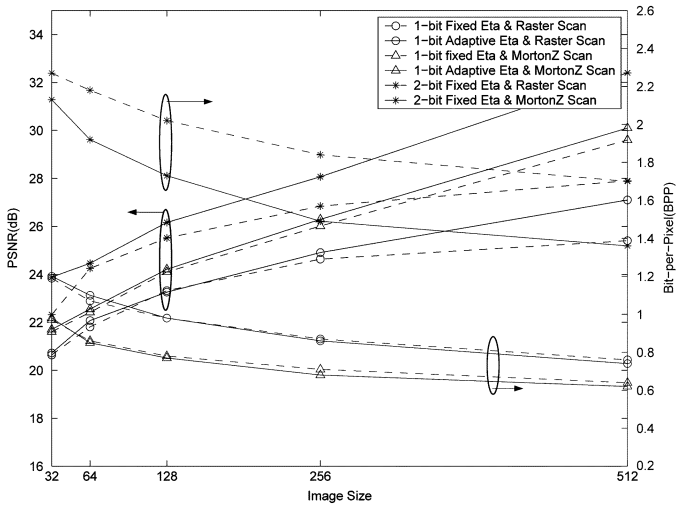


Fig. 14. Average PSNR and BPP figures as function of the image size for the dataset reported in Table I. It is clear that the proposed compression architecture is much more effective for large size images. For 1-bit adaptive Q, using 512×512 image size instead of 64×64 enables a 38% and a 25% improvement in terms of PSNR and BPP, respectively.

TABLE IV
HARDWARE COMPLEXITY AND POWER CONSUMPTION (3.3 V$/100$ MHz) FOR ALL BUILDING BLOCKS AND THE OVERALL PROCESSOR. THE POWER COULD ONLY BE MEASURED FOR THE OVERALL PROCESSOR WHILE, FOR THE BUILDING BLOCKS, ONLY ESTIMATED POWER IS REPORTED

Quantizer Type	Power (mW)		number of Transistors
	Estimated	Measured	
1-bit Quantizer	0.6	-	1.6k
2-bit Quantizer	2.2	-	5.4k
QTD Processor	2.4	-	46k
Overall Processor	5.2	6.3	53k

all four sample images. As expected, the 1-bit quantizer using adaptive η and smooth boundary Morton (Z) scan presents the best PSNR and BPP figures as compared to all 1-bit adaptive quantizers. Fig. 13 shows the experimental measurement of row data from image sample 3. It is experimentally confirmed that the adaptive η converges faster and reduces the mismatch with respect to uncompressed data.

For our sample images using the 1-bit Morton (Z) adaptive Q, the PSNR figures are in the range of 21 and the average BPP is equal to 0.8. The experimentally measured performance is clearly lower than that reported in the simulation results section. This is primarily due to the fact that the experimentally acquired images are of much smaller size (64×64) as compared to the one used for the simulation section (512×512). It usually takes some time for the adaptive quantizer to converge and, thus, better performance is expected to be obtained for images of a larger size because the additional pixels would allow to reach convergence more easily. In order to validate our argument, Fig. 14 shows the average PSNR and BPP figures for different sizes of the images reported in Table I. Regardless of the adaptive quantization schemes, it is evident that significant performance improvements are obtained when using large size images. For 1-bit adaptive Q, using 512×512 format instead of 64×64 enables a 38% and a 25% improvements in terms of PSNR and BPP, respectively. This represents a significant improvement and points out to an important finding suggesting that the compression architecture proposed in this paper is much more effective for large-size images.

The compression processor was tested separately using both 1-bit and 2-bit adaptive quantization. Results showed that the circuit operates correctly for a frequency of up to 100 MHz. However, it should be noticed that this operating speed would not be possible in real time as the bottleneck of the system will be dictated by the access time of the SRAM memory while operating in the compression mode. Table IV reports the hardware complexity and power consumption results for the 1-bit Q, 2-bit Q, QTD processor, and the overall compression processor. Unfortunately, the power could only be measured for the overall compression processor, while for the different building blocks, only estimated power is reported. The reconfigurable adaptive and fixed η 1-bit adaptive Q requires only 1.6 k transistors and consumes less than 1 mW of estimated power while achieving compression ratios corresponding to less than 1 BPP. It is also important to note that, while QTD building block requires the largest number of transistors (46 k mainly required for storing the flag bits), it still consumes little power (about 2 mW). This is explained by the hierarchical nature of the circuit with a maximum of $\log_2 n$ cells ($n = 64$ in our circuit) being updated during each iteration of the tree construction.

Table V reports further the performance comparison between our circuit and some recently reported on-chip compression processors [8]–[13] based on the discrete cosine transform (DCT)

TABLE V
PERFORMANCE COMPARISON OF OUR DESIGN WITH SOME ON-CHIP COMPRESSION PROCESSORS BASED ON DCT AND ICT REPORTED IN THE LITERATURE [8]–[13]. IN THE TABLE, DS AND NA STAND FOR DESIGN STRATEGY AND NOT AVAILABLE, RESPECTIVELY

Processors	1-D ICT [8]	2-D ICT[9]	2-D DCT[10]	2-D DCT[11]	2-D IDCT[12]	2-D DCT[13]	Ours
Technology	0.7μ CMOS	0.35μ CMOS	0.6μ CMOS	0.6μ CMOS	0.35μ CMOS	0.3μCMOS	0.35μ CMOS
DS	semi-custom	semi-custom	semi-custom	semi-custom	semi-custom	full-custom	standard cell
Circuit size	23.4mm^2	9.3mm^2	70mm^2	12.3mm^2	18.2mm^2	4mm^2	$\mathbf{1.8\text{mm}^2}$
Transistors	76k	70k	152k	78k	119k	120k	53k
Voltage	5V	3.3V	2.0V	3.3V	3.3V	0.9V	3.3V
Power	310mW	170mW	133mW	120mW	NA	10mW	**6.3mW**
Frequency	50Mhz	300Mhz	133Mhz	100Mhz	100Mhz	150Mhz	100Mhz

and a more simplified and hardware-friendly scheme, namely, the integer cosine transform (ICT). However, it should be noticed that the comparison of different compression hardware is often difficult and can be very tricky as the objective can be different and hence computational requirements, image quality and compression performance are different. Even though our objective is not to build a DCT or ICT processor, we believe that it is important to compare with DCT, which has become an international standard for sequential codecs as JPEG, MPEG, H.261, H.263, etc. . . [20], [21]. In addition, DCT was also recently reported as an important processing stage in wireless video sensor network and ultra low power biomedical applications, such as the wireless camera pill [22]–[24]. Table V shows that our compression processor occupies a silicon area much lower when compared with DCT and ICT processors realized in the same technology (ten times lower than that of [12] and five times lower than that of [9]). The normalized power[1] is also much lower when compared with other processors realized in similar technological processes (around 14 times lower than [13] and around nine times lower than [9]). It should be noted that our design is only a test-bed prototype including a large number of operating modes and configurations. Greater improvement in terms of power and silicon area can be further achieved by fixing the parameters of our architecture (number of bits of the adaptive Q, scan methodology) and by adopting a full-custom design strategy.

V. CONCLUSION AND DISCUSSION

This paper reports the theory, simulation, VLSI design, and experimental measurements of a single-chip CMOS image sensor and a compression processor. The compression scheme relies on a novel architecture combining boundary adaptation adaptive quantization and an efficient online QTD. The image is first acquired using a time-domain CMOS digital pixel sensor array followed by an adaptive quantization scheme that permits to compress the data to a lower number of bits (typically 1–2 BPP). Further compression is accomplished, while scanning out the pixel values, using QTD algorithm. QTD compresses spatially redundant data in the binary image without any further degradation of the image quality as no comparison with a threshold is required. The performance in terms of both image quality (PSNR) and compression ratio (BPP) were further improved using a novel smooth boundary Morton (Z) scan and a heuristic adaptive boundary rule, which were also implemented in VLSI. Results showed that a PSNR figure of 27–30 dB and a 0.6–0.8 BPP can be achieved while using a very compact 1-bit adaptive Q with Morton (Z) scan for an array resolution of 512×512 pixels. These performance were obtained for raw data without any postprocessing. It is important to note that PSNR figures can be improved at the receiver end using optimal filtering techniques. This is reasonably possible, as in wireless sensor networks the computational and power constraints are loosened on the receiver or the decoder end but not the emitter and the encoder, which need to be compact and very low power.

[1]Power is normalized with respect to the operating frequency and supply voltage.

TABLE VI
DESIGN DECISION TRADEOFFS. ESTIMATION IS BASED ON AN IMAGE SIZE OF 512×512

Quantizer	Low Power $\Leftarrow$		$\Rightarrow$ Quality image	
	1-bit Q/ QTD	2-bit Q/ QTD	3-bit Q/ QTD	≥ 4 Q/ QTD
Power (mW)	< 12	< 15	< 18	≥ 25
Quality PSNR (dB)	$28 - 30$	$31 - 33$	$34 - 36$	> 36
Comp. ratio (BPP)	0.6-0.8	1.3-1.7	2.4-2.8	> 3.4

The normalized power and the silicon area of our on-chip compression processor was compared with a number of image compression on-chip solutions. It is shown that, while our processor is very compact (less than 55 k transistors implementing all modes of operation), it also features a much lower power consumption when compared with other processors realized in similar technological processes (around 14 times lower than [13] and around nine times lower than [9]). Using our proposed compression scheme and depending on the requirements of the application at hand, it is possible to trade power for improved PSNR, and vice versa. Table VI discusses the estimated achievable tradeoff between power consumption, image quality, and compression ratio for a 512×512 pixel array. Furthermore, the proposed compression processor is expected to benefit significantly from higher resolution and Megapixels CMOS imaging technology.

REFERENCES

[1] G. Iddan, G. Meron, A. Glukhousky, and P. Swain, "Wireless capsule endoscopy," *Nature*, pp. 405–417, 2000.

[2] E. Fossum, "CMOS image sensors: Electronic camera-on-chip," *IEEE Trans. Electron Devices*, vol. 44, no. 10, pp. 1689–98, Oct. 1997.

[3] *CMOS Imagers: From Phototransduction to Image Processing*O. Yadid-Pecht and R. Etienne-Cummings, Eds. Boston, MA: Kluwer, 2004.

[4] A. Olyaei and R. Genov, "Mixed-signal Haar wavelet compression image architecture," in *Proc. Midwest Symp. Circuits Syst.*, Cincinnati, OH, 2005, vol. 2, pp. 1267–1270.

[5] Kawahito, "CMOS image sensor with analog 2-D DCT-based compression circuits," *IEEE J. Solid-State Circuits*, vol. 32, no. 12, pp. 2029–2039, Dec. 1997.

[6] Kawahito, "Low-power motion vector estimation using iterative search block-matching methods and a high-speed non-destructive CMOS image sensor," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1084–101092, Dec. 2002.

[7] Q. Luo and J. G. Harris, "A novel integration of on-sensor wavelet compression for a CMOS imager," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2002, vol. III, pp. 325–328.

[8] T. C. J. Pang, C. S. O. Choy, C. F. Chan, and W. K. Cham, "A self-timed ICT chip for image coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 6, pp. 856–860, Sep. 1999.

[9] G. A. Ruiz, J. A. Michell, and A. M. Buron, "Parallel-pipeline 8×8 forward 2-D ICT processor chip for image coding," *IEEE Trans. Signal Process.*, vol. 53, no. 2, pp. 714–723, Feb. 2005.

[10] L. G. Chen, J. Y. Jiu, H. C. Chang, Y. P. Lee, and C. W. Ku, "A low-power 8×8 direct 2D-DCT chip design," *J. VLSI Signal Process.*, vol. 26, pp. 319–332, 2000.

[11] T. H. Chen, "A cost-effective 8×8 2-D IDCT core processor with folded architecture," *IEEE Trans. Consumer Electron.*, vol. 45, no. 2, pp. 333–339, May 1999.

[12] J. S. Chiang, Y. F. Chiu, and T. H. Chang, "A high throughput 2-dimensional DCT/IDCT architecture for real-time image and video system," in *Proc. 8th IEEE Int. Conf. Electron., Circuits Syst.*, Piscataway, NJ, 2001, vol. 2, pp. 867–870.

[13] T. Kuroda, T. Fujita, S. Mita, T. Nagamatsu, S. Yoshioka, K. Suzuki, F. Sano, M. Norishima, M. Murota, M. Kako, M. Kinugawa, M. Kakumu, and T. Sakurai, "A 0.9-V, 150 MHz, 10-mW, 4 mm, 2-D discrete cosine transform core processor with variable threshold-voltage (VT) scheme," *IEEE J. Solid-State Circuits*, vol. 31, no. 11, pp. 1770–1779, Nov. 1996.

[14] D. Martinez and M. M. Van Hulle, "Generalized boundary adaptation rule for minimizing r-th power law distortion in the high resolution case," *Neural Netw.*, vol. 8, no. 6, pp. 891–900, 1995.

[15] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inf. Theory*, vol. IT-25, no. 4, pp. 373–380, Jul. 1979.

[16] E. Artyomov and O. Yadid-Pecht, "Adaptive multiple resolution CMOS active pixel sensor," in *Proc. Int. Symp. Circuits Syst.*, Vancouver, BC, Canada, 2004, vol. 4, pp. 836–839.

[17] E. Artyomov, Y. Rivenson, G. Levi, and O. Yadid-Pecht, "Morton (Z) scan based real-time variable resolution CMOS image sensor," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 7, pp. 947–952, Jul. 2005.

[18] B. Fowler, A. El Gamal, and D. X. D. Yang, "A CMOS area image sensor with pixel-level A/D conversion," in *ISSCC Dig. Tech. Papers*, San Fransico, CA, Feb. 1994, pp. 226–227.

[19] A. Kitchen, A. Bermak, and A. Bouzerdoum, "A digital pixel sensor array with programmable dynamic range," *IEEE Trans. Electron Devices*, vol. 52, no. 12, pp. 2591–2601, Dec. 2005.

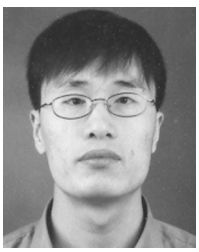
[20] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*, 2nd ed. Boston, MA: Kluwer, 1997.

[21] K. R. Rao and J. J. Hwang, *Techniques and Standards for Image Video and Audio Coding*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[22] L. R. Dung, M. C. Lin, T. Y. Chen, Y. Z. Song, S. J. Huang, and P. K. Weng, "Implementation of ultra-low-power image compression for capsule endoscope," in *Proc. VLSI Design/CAD Conf.*, 2005, pp. 1–9.

[23] X. Xie, G. Li, D. Li, C. Zhang, and Z. Wang, "A new near-lossless image compression algorithm suitable for hardware design in wireless endoscopy system," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2005, vol. 1, pp. 1125–1128.

[24] C. Watson, Effective topologies for the constraints set forth for biomedical sensor networks Norfolk State Univ., Dept., Comput. Sci., Jun. 2004, Tech. Rep. NSUCS-2004-006.



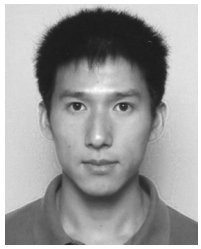
Chen Shoushun received the B.S. degree from the Department of Microelectronics, Peking University, Beijing, China, in 2000, and the M.E. degree from the Institute of Microelectronics, Chinese Academy of Sciences, Beijing, in 2003. His master's thesis dealt with the signal integrity problems in the design of the "Godson-1" CPU which was the first general purpose CPU designed in China. He is currently working toward the Ph.D. degree at the Hong Kong University of Science and Technology (HKUST), Kowloon Bay.

In Septmeber 2003, he joined HKUST, where his doctoral work focuses on the area of time-domain CMOS image sensors. His research interests are in low-power CMOS image sensors and on-chip image processing using time-domain encoding, low leakage, and asynchronous read-out techniques.

Amine Bermak (M'99–SM'04) received the M.Eng. and Ph.D. degrees in electronic engineering from Paul Sabatier University, Toulouse, France, in 1994 and 1998, respectively.

During his doctoral studies, he was part of the Microsystems and Microstructures Research Group, French National Research Center LAAS-CNRS, Toulouse, France, where he developed a 3-D VLSI chip for artificial neural-network classification and detection applications. He joined the Advanced Computer Architecture Research Group, York University, York, U.K., where he was a Postdoctoral Researcher involved with VLSI implementation of CMM neural networks for vision applications in a project funded by British Aerospace. In November 1998, he joined Edith Cowan University, Perth, Australia, first as a Research Fellow working on smart vision sensors, then as a Lecturer and a Senior Lecturer with the school of Engineering and Mathematics. He is currently an Assistant Professor with the Electrical and Electronic Engineering Department, Hong Kong University of Science and Technology (HKUST), Kowloon Bay, where he is also serving as the Associate Director of the Computer Engineering Program. His research interests are related to VLSI circuits and systems for signal, image processing, sensors and microsystems applications. He has published extensively on the above topics in various journals, book chapters, and refereed international conferences.

Dr. Bermak was the recipient of the Bechtel Foundation Engineering Teaching Excellence Award and the IEEE Chester Sall Award in 2004. In 2005, he received the Best Paper Award at the 5th IEEE International Workshop on System-On-Chip for Real-Time Applications. He is a member of the IEEE Circuits and Systems Society committee on sensory systems.



Wang Yan received the B.Eng. degree in mechatronic engineering from the Faculty of Science and Engineering, City University of Hong Kong, Hong Kong, in 2005. He is currently working toward the M.Phil. degree at the Hong Kong University of Science and Technology, Hong Kong.

In September 2005, he joined HKUST, where his studies focus on image compression for on-chip CMOS image sensors. His current research work is exploring new algorithms for hardware-friendly image compression and their field-programmable gate array implementation and validation.

Dominique Martinez received the Ph.D. degree in electrical and electronic engineering from the Paul Sabatier University, Toulouse, France, in 1992.

He was a Postdoctoral Fellow with the Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, and the VLSI Group, Harvard University, Cambridge, MA, in 1992 and 1994, respectively. From 1993 to 1999, he was with LAAS-CNRS, Toulouse, where his research interests were concerned with machine learning (artificial neural networks and support vector machines). In 2000, he joined LORIA-CNRS, Vandoeuvre-Les-Nancy, France, where his research interests currently focus on biologically plausible spiking neural networks for sensory processing, with particular application to artificial olfaction (neuromorphic electronic noses).