

University of Wollongong
Research Online

Faculty of Health and Behavioural Sciences -
Papers (Archive)

Faculty of Science, Medicine and Health

1-1-2008

**FPGA implementation of a predictive vector quantization image
compression algorithm for image sensor applications**

Yan Wang
Hong Kong University of Science and Technology

Amine Bermak
Hong Kong University of Science and Technology

Abdesselam Bouzerdoum
University of Wollongong, bouzer@uow.edu.au

Brian W. Ng
University of Adelaide

Follow this and additional works at: <https://ro.uow.edu.au/hbspapers>

 Part of the [Arts and Humanities Commons](#), [Life Sciences Commons](#), [Medicine and Health Sciences Commons](#), and the [Social and Behavioral Sciences Commons](#)

Recommended Citation

Wang, Yan; Bermak, Amine; Bouzerdoum, Abdesselam; and Ng, Brian W.: FPGA implementation of a predictive vector quantization image compression algorithm for image sensor applications 2008, 431-434.
<https://ro.uow.edu.au/hbspapers/2293>

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au

FPGA implementation of a predictive vector quantization image compression algorithm for image sensor applications

Abstract

This paper presents a hybrid image compression scheme based on a block based compression algorithm referred to as Vector Quantization (VQ) combined with the Differential Pulse Code Modulation (DPCM) technique. The proposed image compression technique called the PVQ scheme results in enhanced image quality as compared to the standalone VQ. The generated codebooks for the PVQ scheme are more robust for image coding than that of the VQ. This made our system a suitable candidate for developing on chip image sensor with integrated data compression processor. The proposed system was validated through FPGA implementation. The resulting implementation achieved good compression and image quality with moderate system complexity.

Keywords

FPGA, implementation, predictive, vector, quantization, image, compression, algorithm, for, image, sensor, applications

Disciplines

Arts and Humanities | Life Sciences | Medicine and Health Sciences | Social and Behavioral Sciences

Publication Details

Y. Wang, A. Bermak, A. Bouzerdoun & B. Ng, "FPGA implementation of a predictive vector quantization image compression algorithm for image sensor applications," in Proceedings of Fourth IEEE International Symposium on Electronic Design, Test and Applications (DELTA-2008), 2008, pp. 431-434.

FPGA Implementation of a Predictive Vector Quantization image compression algorithm for image sensor applications

Yan Wang^a, Amine Bermak^a, Abdesselam Bouzerdoum^b and Brian Ng^c

^a ECE Department, Hong Kong University of Science and Technology

^b University of Wollongong, Australia ^c University of Adelaide, Australia

Email: wongyin,eebermak@ust.hk

Abstract—This paper presents a hybrid image compression scheme based on a block based compression algorithm referred to as Vector Quantization (VQ) combined with the Differential Pulse Code Modulation (DPCM) technique. The proposed image compression technique called the PVQ scheme results in enhanced image quality as compared to the standalone VQ. The generated codebooks for the PVQ scheme are more robust for image coding than that of the VQ. This made our system a suitable candidate for developing on chip image sensor with integrated data compression processor. The proposed system was validated through FPGA implementation. The resulting implementation achieved good compression and image quality with moderate system complexity.

I. INTRODUCTION

Image compression plays an extremely important role in most of today's digital imaging applications such as cell phones, Digital Cameras, or even medical image acquisition devices, such as the camera-pill [1].

Intensive image processing and storage effort is required for delivering high quality images and videos through limited bandwidth. The burden of data transmission and storage for most applications is relieved by data compression which is achieved at the cost of extra computationally intensive processes [2]. Although a number of data compression algorithms have already been proposed in the literature, most of these algorithms often consume high power and occupy large silicon areas, when implemented on silicon. Image compression is still the most expensive process in terms of hardware implementation for digital imaging applications [3].

In this paper, vector quantization and predictive vector quantization algorithms are explored for developing hardware-friendly image compression system with integrated on-chip digital CMOS image sensor. Both algorithms are compared following different block based scanning path: Raster, Morton Z, and Hilbert scan [7]. In the predictive scheme, simple linear vector predictor is used to evaluate the practicability for hardware implementation. The LBG algorithm is used to train the codebook for different quantization processes using the *de facto* standard test images from the University of Southern California (USC) image database. Several images that are not included in the codebook training process are tested under different compression schemes. Furthermore, the circuit architecture for vector quantization will be proposed.

The remainder of this paper is organized as follows. Section II introduces the Vector quantization algorithm and the Predictive Vector Quantization Scheme. The codebook and the predictor design algorithm will be introduced in Section III. Section IV describes the MATLAB simulation results and FPGA validation for both VQ and PVQ schemes. Section V concludes this work.

II. VECTOR QUANTIZATION ALGORITHM

A natural extension of ordinary scalar quantization schemes is to quantize multiple components simultaneously, also known as Vector Quantization. Instead of performing quantization on a single source input signal, a vector quantizer codes clusters of signals from multiple input sources by their centers. In an image coding context, a block of k pixels will form a vector. Each image pixel is considered to be an element in this k -dimensional vector. The block size is usually equal to $\sqrt{k} \times \sqrt{k}$ for image coding, which is a square block. This square block is also called a tile. Pictorial illustration can be found in Figure 1(a), where the image shown is in a size of 8×8 and a block is in the size of 4×4 . Raster or Morton Z scanning pattern can be used within each block. The image is Vector Quantized using a block-based scanning strategy. Typical sizes for the block can be 2×2 , 4×4 , or 8×8 , ... etc.

The pixel values in the $\sqrt{k} \times \sqrt{k}$ matrix block are then rearranged into a $1 \times k$ vector following a certain scanning sequence. A vector of the input signal as a whole forms the input to the vector quantizer. At both the encoder and the decoder sides of the vector quantizer, there is a set of k -dimensional vectors, which is called *codebook* of the vector quantizer. The vectors in the codebook are referred to as *codeword* or *codevector*. Each codeword is selected to quantize the input vector and is assigned a binary index in order to represent it. This vector is fed as the input to the Vector Quantizer, where the stored codewords in the codebook have the same dimension as this vector. The graphical representation of the input image sequence vector and codewords are illustrated in Figure 1(b). The parameter k is the Block size, for typical sized block, for instance a 4×4 block, $k = 4 \times 4 = 16$. The value N is the Codebook size, which indicates the number of codewords that are stored in the codebook, the typical values are 16, 32, 64, 128, 256, 1024, etc.. The value m indicates the

number of vectors in an input image, which can be calculated as the ratio between the input image size and the block size. For example an image size of 64×64 and a block size of 4×4 would lead to a value m of $\frac{64 \times 64}{4 \times 4} = 256$.

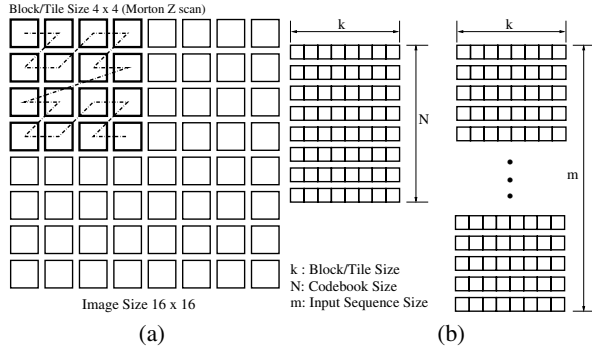


Fig. 1. (a). A typical image in a size of $8 \text{ pixels} \times 8 \text{ pixels}$. The image is Vector Quantized in a block by block manner. Within each block, Morton Z or Raster Scanning method can be used. (b). The tile is converted into a $1 \times k$ vector. The whole image sequence is then rearranged into an $m \times k$ array. N is the size of the Codebook, and m is the number of input image sequence.

At the encoder side, the input vector will be compared with each of the codewords in order to find the one which has the minimum distortion with an input vector. The index of this closest vector will be sent to the decoder. The distortion measure can be the Mean Absolute Error (MAE) or the Mean Square Error (MSE). The transmitted index value will be received and stored by the decoder, which has the same codebook as the encoder. The decoder then retrieves the codeword according to the index received to reconstruct the input vector. Therefore, the codeword serves as the quantized version of the input vector.

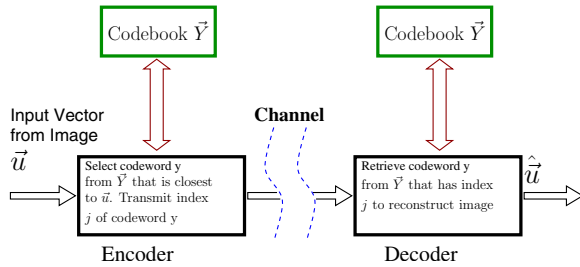


Fig. 2. The pictorial illustration of a typical Vector Quantization Scheme.

Figure 2 illustrates the details of a typical Vector Quantization Process. In the encoder side, the input vector \vec{u} from the input image sequence needs to be compared with all the codewords that are stored in the codebook \hat{Y} . The index j of the codeword, which has the minimum distortion/MSE will be sent through the channel to the decoder. On the decoder side, the received index j will be used to retrieve the codeword from the codebook, which is identical to the one on the encoder side.

The retrieved codeword \hat{u} is used as an approximation of the original input sequence \vec{u} .

A. Predictive Vector Quantization Scheme

The natural extension of the concept of Differential Pulse Code Modulation [4] can also be applied in the ordinary Vector Quantization algorithm, which is referred to as the Predictive Vector Quantization (PVQ). In PVQ, the input vector rearranged from a block of an image sequence is firstly subtracted from its prediction vector to form a difference/residual vector. The difference vector will then be compared with all the codewords in the codebook that are stored in the vector quantizer. The codeword which is closest to the difference vector will be picked-up and added to the prediction vector to reconstruct the current block. The reconstructed vector is then used to calculate the prediction vector of the next iteration. Each codeword has its own binary label or index. The index of the codeword which has the lowest distortion with the difference vector (closest) will be sent to the decoder through a channel. On the decoder side, the received index value is used to retrieve the codeword that can best represent the difference vector. By adding the difference vector with the prediction vector, the reconstructed image vector can be obtained. Both the encoder and the decoder have the same decoding, prediction and reconstruction mechanisms. The prediction vector is obtained based on the reconstructed vector. This is because the decoder has no other side information about the original input vector, and therefore should not be used for performing the prediction in order to eliminate cumulative errors. The illustration of a typical Predictive Vector Quantization Scheme is depicted in Figure 3.

B. Codebook Design Algorithm

The Vector Quantization and the Predictive Vector Quantization schemes both need a predefined codebook for performing the quantization. The codebook should be designed to best represent the input vector. The codebook design problem is very similar to that of the classifier design problem for pattern recognition as these two tasks share the same mathematical model. For a given input image, the codebook designed for vector quantization is based on the statistics of this image. Given the raw image data and the size of the codebook N , the VQ design objective is to find the centroid vectors of N clusters of input vectors. Linde, Buzo, and Gray [5] generalized a procedure for designing codebook of any dimension and any size. The details are as follows:

- 1) Start with an initial set of codebook $\{C_i\}_{i=1}^N$ and training vectors $\{X_i\}_{i=1}^m$, select error tolerance threshold ϵ .
- 2) Find the clustered/quantization regions

$$R_i = \{X_n : d(X_n, C_i) < d(X_n, C_j) \forall j \neq i\} \quad (1)$$

- 3) Compute the distortion

$$D[t] = \sum_{i=1}^N \left\{ \sum_{R_i} \|X - C_i\|^2 \right\} \quad (2)$$

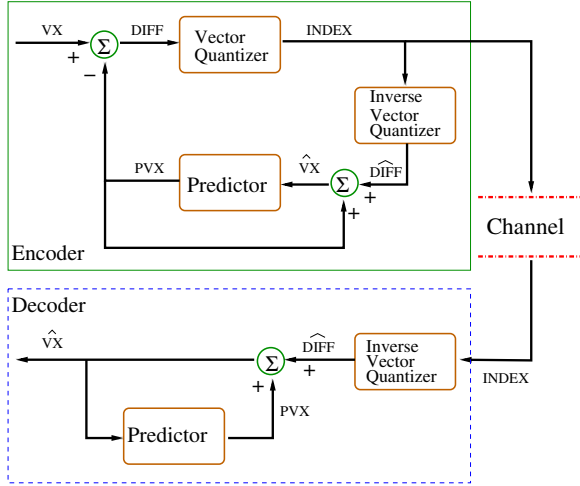


Fig. 3. Illustration of a typical Predictive Vector Quantization Scheme. In the encoder, the input vector \overrightarrow{VX} is subtracted from its prediction vector \overrightarrow{PVX} to form a difference vector \overrightarrow{DIFF} , which is then compared with all the codewords stored in the codebook of the Vector Quantizer. The Index of the codeword, which has the minimum distortion with the input vector is sent out through the channel to the decoder. This quantized vector \overrightarrow{DIFF} will then be added to the prediction vector to form the reconstructed input vector \overrightarrow{VX} . This reconstructed input vector is used to calculate the prediction vector \overrightarrow{PVX} for the next iteration. On the decoder side, the same decoding scheme is used.

- 4) If $D[t-1] - D[t] < \epsilon$, terminate the process; otherwise, repeat the process again starting from the step 2

In order to initialize the codebook, in the original paper [5], a method called splitting technique is introduced. Codebook is generated through applying the LGB algorithm iteratively. Another simple method proposed by Hilbert [6] is to randomly pick up the codewords from the input vector. The difference in terms of performance of these two methods is unnoticeable for our application.

C. Vector Prediction

There exist a large number of algorithms devoted for designing efficient and optimal vector predictors for PVQ, both linear [8] and nonlinear [9]. Both linear and nonlinear predictors require the knowledge of adjacent reconstructed blocks, upper and left, followed by a series of computations, to best estimate the pixel values for the current block. For most cases, image statistics are not known to the encoder, therefore predictors are mostly suboptimal. Most of the existing nonlinear predictors are too complicated to be hardware-friendly. Therefore, in order to be implemented in hardware, only relatively simple linear predictors are considered here. Figure 4 illustrates the pixels used in the vector prediction scheme for a 4×4 block.

In this prediction scheme, only pixels from the neighboring blocks will be used. x_i , $i = 1, 2, \dots, 16$ represent the pixels values in the current block. The reconstructed pixel values of the neighboring blocks that are of interest are denoted as a_1, b_1, c_1, d_1 in the upper block and a_2, b_2, c_2, d_2 in the left block. Simple linear predictor can be used here, for example:

$$\hat{x}_1 = (a_1 + a_2)/2, \hat{x}_2 = (b_1 + \hat{x}_1)/2, \hat{x}_3 = (c_1 + \hat{x}_2)/2, \\ \hat{x}_4 = (d_1 + \hat{x}_3)/2, \hat{x}_5 = (\hat{x}_1 + b_2)/2, \hat{x}_6 = (\hat{x}_5 + c_2)/2, \hat{x}_7 = (\hat{x}_9 + d_2)/2, \\ \hat{x}_8 = (\hat{x}_5 + \hat{x}_2)/2, \hat{x}_9 = (\hat{x}_{10} + \hat{x}_7)/2, \text{ etc. This linear predictor is used in our PVQ system.}$$

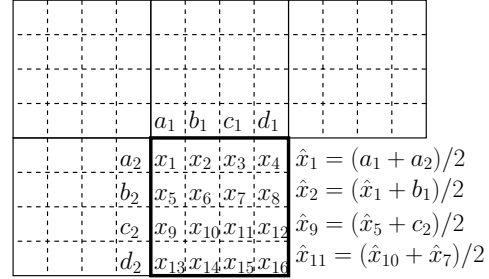


Fig. 4. Illustration of Vector Prediction Scheme.

III. EXPERIMENTAL RESULTS

The Vector Quantization and Predictive Vector Quantization schemes are designed based on the window-based scan patterns. VQ and PVQ schemes with Raster and Morton Z scan are built. For the PVQ system, the predictor described earlier is used to estimate the current block that are being encoded.

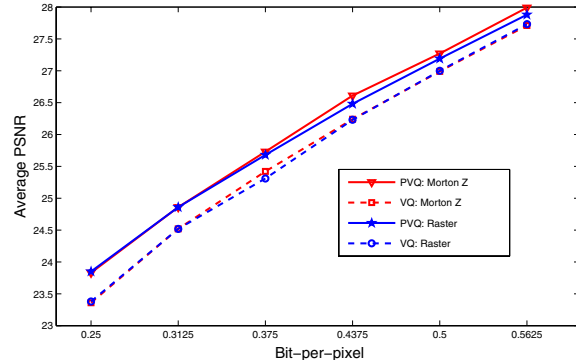


Fig. 5. Performance comparison between VQ and PVQ using both Raster and Morton Z scanning methods.

The performance of the VQ and the PVQ systems are compared in terms of image quality measured by PSNR and compression rate measured by Bit-per-Pixel (BPP). Codebooks are generated from images in the USC-SIPI image database. Images that are not selected for codebook generation are used to test the final codebook. Figure 5 presents the performance comparison between VQ and PVQ in both Raster and Morton Z scan order. The images chosen for this experiment are 256×256 Lena, Barbara, Baboon Cameramen, and Peppers images. The block size for quantization is 4×4 . In this figure, the average PSNR results are shown for comparison. It is clear that the PVQ schemes outperform the VQ schemes for both Raster and Morton Z scan. The PVQ scheme with Morton Z scan performs slightly better than with Raster scan. For both

TABLE I
FPGA DESIGN SUMMARY REPORT

VQ			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	1552	28672	5%
Number of 4 input LUTs	8822	28672	30%
Frequency (MHz)	25		
PVQ			
Logic Utilization	Used	Available	Utilization
Number of Slice Flip Flops	4374	28672	15%
Number of 4 input LUTs	15209	28672	53%
Frequency (MHz)	25		

cases the tested images are not used for the generation of codebooks. This indicates that by coding the errors between blocks, the codebook is less dependent on images, which potentially implies that a fixed or *universal* codebook can be used to code any image. This is mainly due to the fact that the codebook generated from residual blocks is more compact than that of the ordinary blocks so that although the simple predictor is far from optimal, the overall performance of the PVQ system is quite satisfactory and robust.



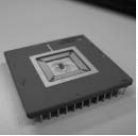



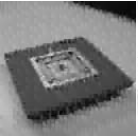



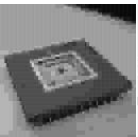

The VQ scheme with Morton Z scan and the PVQ system with Raster scan is implemented in the FPGA platform based on XILINX Virtex-II. Images to be tested are in the size of 128×128 . Raster Scan method is used for PVQ in order to reduce the required memory size for storing adjacent blocks. In the PVQ scheme, there are 7 pixel values that should be stored for each block. In total, $7 \times (128/4) \times 8 = 1792$ registers should be allocated for storing neighboring blocks. Table I summarizes the FPGA design report. The codebook size is fixed to 64, which translates into 0.375bpp. This size is chosen as it provides a good compromise between image quality and system complexity. The distortion measure used in the FPGA implementation is set to the Mean Absolute Error (MAE).

Four images (Baby, Sea, Chip, and Toy) are tested on the FPGA platforms and the PSNR results are listed in Table II. The reconstructed images are presented for comparison in terms of visual quality. The reconstructed images present some block-based distortion mainly due to the small image size as the smaller the image size is, the worse the image quality will be for typical Vector based quantization algorithm. However in general the PVQ outperforms the VQ system. The codebook used in the two systems are also trained from the USC image database.

IV. CONCLUSION

This paper presents a vector-based image compression algorithm, which consists of Vector Quantization scheme followed by DPCM/Predictive compression technique. MATLAB simulation results show improved performance achieved by PVQ over the ordinary VQ scheme. Even though the added DPCM block requires extra hardware resources, it was demonstrated that the codebook generated by coding the residual blocks makes the system more robust. The universal residual codebook can be versatile and applicable for different images. The

TABLE II
THE TESTING OF VQ AND PVQ SYSTEM IN FPGA PLATFORM

Original Image			
			
Predictive Vector Quantization with Raster Scan			
			
24.46dB	30.37dB	27.97dB	24.45dB
Vector Quantization with Morton Z Scan			
			
23.96dB	29.52dB	27.47dB	23.57dB

proposed compression scheme was implemented on an FPGA platform. The moderate complexity of the algorithm makes it a very suitable candidate for on-chip compact and low power image compression algorithm that can be integrated with a CMOS image sensor.

ACKNOWLEDGMENT

The work described in this paper was supported by a grant from the Research Grant Council of HK SAR, China (Project No 610405).

REFERENCES

- [1] G. Iddan, G. Meron, A. Glukhousky and P.Swain "Wireless capsule endoscopy", *Nature*, pp.405 - 407, 2000.
- [2] A. Olyaei, R. Genov, "Mixed-Signal Harr Wavelet Compression Image Architecture", *Midwest Symposium on Circuits and Systems(MWSCAS'05)*, Cincinnati, Ohio, 2005
- [3] Kawahito *et Al.*, "CMOS Image Sensor with Analog 2-D DCT-Based Compression Circuits", *IEEE Journal of Solid-State Circuits*, Vol. 32, No.12, pp.2029 - 2039, December 1997.
- [4] K. Sayood, "Introduction to Data Compression", 3rd Ed. *Morgan Kaufmann Publishers*, 2006
- [5] Y. Linde, A. Buzo, and R.M. Gray, "An Introduction for Vector Quantization Design", *IEEE Transactions on Communications*, COM-28:84-95, Jan, 1980
- [6] E.E. Hilbert, "Cluster Compression Algorithm – A Joint Clustering Data Compression Concept", *Technical Report, JPL Publication*, pp. 77-43, Washington, DC: NASA 1977
- [7] D. Hilbert, "Über die stetige Abbildung einer Linie auf ein Flächenstück", *Math. Ann.*, Vol. 38, pp. 459-460, 1891
- [8] J. E. Fowler Jr., M. R. Carbonara, S. C. Ahalt, "Image Coding Using Differential Vector Quantization", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, No. 5, Oct 1993
- [9] S. A. Rizvi, N. M. Nasrabadi, "Predictive Residual Vector Quantization", *IEEE Transactions on Image Processing*, vol. 4, No. 11, Nov 1995