Tema

Tendências em Matemática Aplicada e Computacional, **15**, N. 1 (2014), 119-130
© 2014 Sociedade Brasileira de Matemática Aplicada e Computacional
www.scielo.br/tema
doi: 10.5540/tema.2014.015.01.0119

A Faster Pixel-Decimation Method for Block Motion Estimation in H.264/AVC

R.C. LINS[1], D.B. HENRIQUES[1], E.A. LIMA[2*] and S.B. MELO[1]

Received on August 19, 2012 / Accepted on January 29, 2014

ABSTRACT. This work explores the highly advantageous cost/benefit relation presented by low discrepancy sequences as a pixel-decimation technique to improve the block estimation performance in the H.264/AVC. The proposed method is able to efficiently estimate motion vectors for block matching algorithms present in the H.264 by using latticed decimation sampled according to the Van Der Corput-Halton sequences. This paper further explores motion estimation within the H.264, validating it with real-case-video-encoding scenarios. The results have shown that this technique incorporated to the H.264 is generally more efficient than other decimation techniques used in similar conditions.

Keywords: motion estimation, low-discrepancy sequences, video coding.

1 INTRODUCTION

Motion estimation (ME) is an important part of data compression methods adopted by all existing video coding standards, such as H.263, MPEG-4, etc. In the H.264, the ME mechanism is designed with many new features, such as a variable block size, multiple reference frames and a quarter pixel accurate estimation. The new features obviously improve the coding efficiency. However, the gain of its high performance is at the price of heavy computational complexity [12]. Experimental results have shown that the ME process usually takes about 60% of the computational load for the case of the single reference frame, and 80% for the case of the multiple reference frames. Therefore, the reduction of the ME computational complexity while still maintaining as much as possible the same coding efficiency has become an important issue in the H.264 community.

Even though the Full Search algorithm (FS) [10] does provide the most accurate result, its overly time consuming process makes it unsuitable for a real time video application. Fast search algorithms significantly improve the encoding speed with negligible loss in picture quality. There

*Corresponding author: Emerson Alexandre de Oliveira Lima
[1]Centro de Informática, CIn, UFPE – Universidade Federal de Pernambuco, 50740-530 Recife, PE, Brazil.
E-mails: rcl@cin.ufpe.br; dbbh@cin.ufpe.br; sbm@cin.ufpe.br
[2]UPE – Universidade de Pernambuco, 50100-010 Recife, PE, Brazil. E-mail: eal@poli.br

are many fast search techniques that have been proposed such as the Three-Step Search (TSS) [4], the New Three-Step Search (NTSS) [6], the Four-Step Search (4SS) [9], the Cross-Diamond Search (CDS) [2] and the Block-Based Gradient Descent Search (BBGDS) [7], all attempting to employ square-shaped patterns of different sizes to search for the best-matching block within the search window. However, these algorithms tend to be trapped in local minima when the motion does not appropriately match with the predefined pattern.

To overcome this problem, fast hybrid algorithms have been introduced such as Enhanced Predictive Zonal Search (EPZS) and Unsymmetrical-Cross Multi-hexagon-grid Search (UMHS) [1]. These algorithms combine several fixed techniques to balance the quality and the encoding speed of the video compression, and has been adopted in the H.264/MPEG-4 AVC Reference Software.

Another approach to reduce computational effort in the ME is to use a pixel-decimation technique within its similarity measure. Any pixel-decimation technique can be combined with a motion estimation approach by replacing the similarity measure with the proposed technique, which is the focus of this paper. We propose in this work the incorporation in the H.264 of a pixel-decimation pattern where the selection of the pixels is based on *low discrepancy sequences*, more specifically the Van Der Corput Halton (VDH) low discrepancy sequence, for it presents better approximations, according to the similarity measure used in the block-matching [5].

2 MOTION ESTIMATION

Techniques for motion estimation identify blocks in the current frame that best match blocks in a previous (reference) frame. This is done in order to reduce redundancy by expressing the current frame simply as a composition of motion vectors describing the movement of blocks in relation to the reference frame. The purpose of motion estimation is to find the *optimal motion vector* representing the displacement between the current block in the reference frame and its best matching block in the adjacent frame.

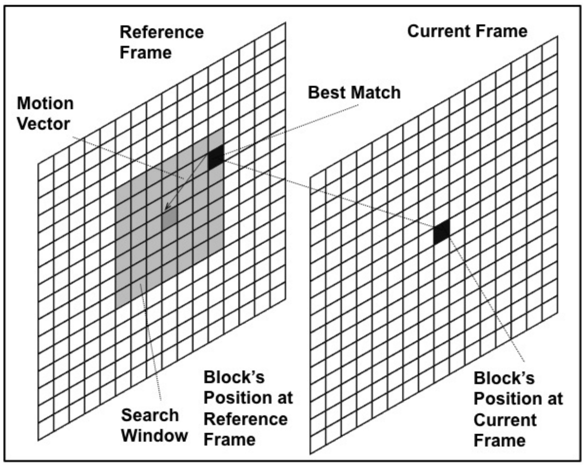

Reference Frame
Current Frame
Motion Vector
Best Match
Block's Position at Reference Frame
Search Window
Block's Position at Current Frame

Figure 1: Elements present in motion estimation.

Given a block of size $M \times N$ and a search window of size $(2dm + 1) \times (2dm + 1)$, where dm is the maximum search range, in pixels, for the vertical and horizontal coordinates of the block, the motion vector is the displacement between such a block and a similar block in the next frame which is the best match in terms of its luminance contents. To find the best match between two blocks one can use the mean absolute deviation (MAD), defined by:

$$MAD(i, j) = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} |s_k(m, n) - s_{k-1}(m + i, n + j)|, \tag{1}$$

where $s_k(m, n)$ represents the luminance value of the pixel in the reference block, $s_{k-1}(m + i, n + j)$ is the luminance value of the pixel in the block from the previous frame, (m, n) gives the local coordinates of the upper left corner of the reference block in the current frame and i and j are the coordinates offsets in the search window.

The goal is to find a motion vector (u, v) associated to the minimum $MAD(i, j)$, where i and $j \in [-dm, dm]$. In FS, the equation (1) is calculated for all $(2dm + 1)^2$ positions of the candidate blocks within the search window, and the block with the lowest MAD (minimum distortion) is selected as the prediction, i.e., the position of this block within the search window corresponds to the motion vector. Some algorithms use SAD (sum of average distortion) instead of MAD to determine the best match for the motion vector, where MAD is the normalized form of SAD ($MAD = \frac{1}{MN} SAD$).

3 OVERVIEW OF H.264/AVC MOTION ESTIMATION

Similar to the previous video coding standards, H.264 is also a block-based motion-compensated hybrid video coder. However, it possesses many specifics that considerably affect the coding efficiency and complexity of the ME process. In order to achieve a higher coding efficiency, the H.264 adopts multiple block sizes for the ME. As specified in H.264, an Motion Block (MB) can be partitioned into 7 different block sizes or modes for inter-frame prediction. The block can be of sizes 16×16, 16×8, 8×16, or 8×8. The 8×8 mode can further lead to another 4 small block sizes, namely, 8×8, 8×4, 4×8, and 4×4, as illustrated in Figure 2 [10]. Moreover, H.264 supports multiple reference frames. In other words, more than one previously decoded frame can be employed as reference frames to code the current frame.

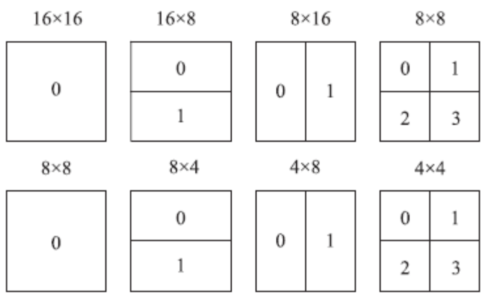


Figure 2: The various H.264 block sizes with their associated block indexing shown inside each block.

In the H.264/MPEG-4 AVC Reference Software the Lagrangian rate distortion optimization (RDO) function shown in Equation 2 is exploited as its block size decision criterion

$$J(\mathbf{m}, \lambda_{motion}) = SAD(s, c(\mathbf{m})) + \lambda_{motion} \times R(\mathbf{m} - \mathbf{p}), \tag{2}$$

where $\mathbf{m} = (m_x, m_y)^T$ is the current motion vector (MV), $\mathbf{p} = (p_x, p_y)^T$ is the predicted MV, and λ_{motion} is the Lagrangian multiplier. SAD is used for the distortion measurement, and $R(\mathbf{m}-\mathbf{p})$ represents the number of bits required to code the difference between the current MV $\mathbf{m}$ and the predicted MV $\mathbf{p}$. All block sizes are investigated based on this criterion and the one with minimum cost is considered to be the optimal one. Consequently, the computational complexity resulted from such an exhaustive search is extremely high, and a fast ME algorithm to effectively reduce the computational complexity is therefore desirable.

3.1 Fast Motion Estimation Algorithms in H.264/AVC

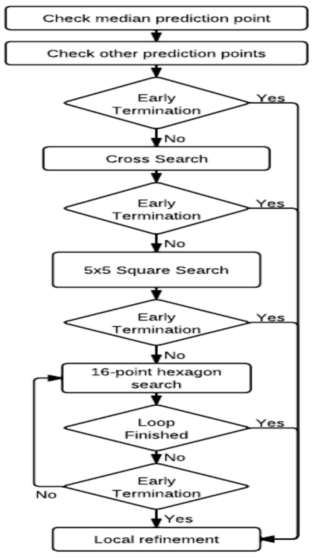
The H.264 video coder comprises two major fast search algorithms to perform motion estimation: the Unsymmetrical-cross Multi-Hexagon grid Search algorithm (UMHS) and the Enhanced Predictive Zonal Search (EPZS).

UMHS replaces the single framework with many kinds of combination frameworks and effectively enhances the prediction. Although the UMHS algorithm is composed of many steps, it significantly reduces the computational complexity by effectively reducing the total amount of compared pixels. It is a hierarchical search strategy and includes four main patterns which can be viewed in Figure 3(a). Figure 3(b) ilustrates the search process of UMHS algorithm. The search begins with the unsymmetrical cross search.

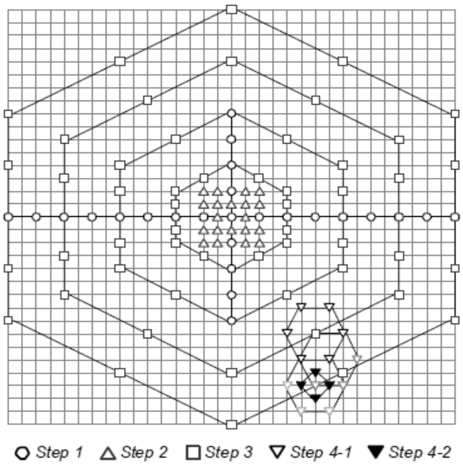
The EPZS on the other hand is considered by the author [11] as an improvement of both the Predictive Motion Vector Field Adaptive Search Technique (PMVFAST) as well as the Advanced Predictive Diamond Zonal Search (APDZS). The first step consists of finding an optimal predictor among several candidates (see Figure 4). Some possible optimal predictors are the spatial or temporal mean motion vector, the initial motion vector, the differentially increased/decreased motion vector based on two previous frames and others. After the optimal predictor has been selected, the algorithm then defines an improved adaptive threshold to be used as an early termination parameter while diminishing the possibility of a quality loss due to this optimization (early termination). The last step is a simplified search pattern where the algorithm checks the optimal predictor's neighborhood for the best match for the motion vector.

4 LOW-DISCREPANCY SEQUENCES

Monte Carlo methods form a class of algorithms that use pseudo-random numbers in tasks that usually involve numerical processing of a large amount of data [5]. These methods have been applied to a great variety of numerical problems such as in the iterated integral estimation, and usually present great advantages over the traditional approaches of partition intervals such as Simpson's and the like [3].



(a) Diagram of the UMHS algorithm;



(b) The search process of UMHS algorithm.

Figure 3: The UMHS algorithm.

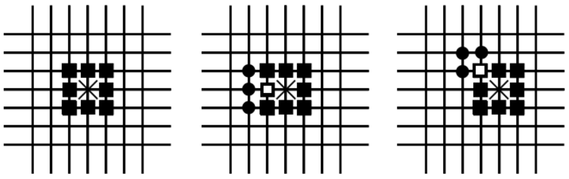


Figure 4: The EPZS algorithm.

Among the pseudo-random sequences, the so called low discrepancy sequences have been the object of major research, given its apparent superiority with respect to space uniform cover and the convergence performance in the integral estimates.

We call *discrepancy* the measure of uniformity of a given sequence[1]. More formally:

Definition 1 (Discrepancy). *Let* $\omega = \{x_1, x_2, \cdots, x_n, x_{n+1}, \cdots\}$ *an infinite sequence of real numbers in the interval* $[0, 1]$ *and let* $I \subseteq [0, 1]$ *one sub-interval. We define* $A(I; n)$ *the amount of points of the subsequence* $x_1, x_2, \cdots, x_n$ *belonging to I, i.e,* $A(I; n) = |I \cap \{x_1, x_2, \cdots, x_n\}|$. *In a uniform sequence, the amount of points* $A(I; n)$ *is proportional to* I*'s measure, i.e,*

$$\lim_{n\to\infty} \frac{A([\alpha, \beta); n)}{n} = \beta - \alpha.$$

The measure of the deviation

$$D_n = D_n(\omega) = \sup_{0\leq\alpha<\beta\leq1} \left| \frac{A([\alpha, \beta); n)}{n} - (\beta - \alpha) \right|$$

[1]The definitions and notation that follow are the same as in [5].

is called Discrepancy *of the first n points of the sequence ω.*

An associate measure of a somewhat simpler computation is the so called *star-discrepancy* expressed by:

$$D_n^* = D_n^*(\omega) = \sup_{0<\beta\le 1} \left| \frac{A([0,\beta);n)}{n} - \beta \right|.$$

A well established result [5] relates both measures through the inequality

$$D_n^* \le D_n \le 2D_n^*.$$

Considering sequences in the interval [0, 1], if the discrepancy of the sequence is zero, then it has a completely uniform cover and if it is near 1 then the sequence will be poorly uniform, leaving empty chunks in the interval. Some sequences may be regarded as a low discrepancy, i.e., near zero discrepancy, when considered as a whole, however they may present a high discrepancy behavior during their early stages of construction. For instance, consider a total of 256 points for a given low discrepancy sequence, but it is possible that, if we take an intermediate phase of its construction, encompassing 64 points for example, this sequence may present a poorly uniform cover, which is a high discrepancy behavior. Other sequences possess the property of behaving like a low discrepancy sequence in all of its construction phases, such as Van Der Corput-Halton's. In this paper, we will utilize Van Der Corput-Halton's sequences [5], defined as follows:

Definition 2 (Van der Corput-Halton's Sequence (VDH)). *Let $b > 1$ be a positive integer. The sequence $VDH_b = \{x_1, x_2, \cdots\}$ for which the term x_i is defined by*

$$x_i = \sum_{j=0}^{s} \frac{a_j}{b^{j+1}},$$

where $\sum_{j=0}^{s} a_j b^j$ is the expansion of the number $i-1$ in base b, a_j are the expansion's coefficients and s is the number of terms. It is called Van Der Corput-Halton's Sequence in the base b.

4.1 Low-Discrepancy Sequences and Block-Matching

Let X be an $M \times N$ block in the current frame and Y be an $M \times N$ block in the previous frame's search window. Let $P = \{p_1, p_2, \cdots, p_k\}$ be the subset of $\{1, 2, \cdots, M\} \times \{1, 2, \cdots, N\}$, i.e, a subset of all possible coordinates of blocks X e Y. We define the mean absolute difference *induced* by P as:

$$MAD_P(X,Y) = \frac{1}{k} \sum_{l=1}^{k} |X(p_l) - Y(p_l)|. \tag{3}$$

Notice that if $P = \{1, 2, \cdots, M\} \times \{1, 2, \cdots, N\}$ then Equation (3) corresponds to Equation (1) for fixed positions (m, n) and (i, j). As will be seen briefly, the use of low discrepancy sequences to evaluate the MAD-related summations presented very encouraging results.

In order to employ low discrepancy sequences by making use of the concept of induced MAD, we need to transform the sequence VDH_b of real numbers in the interval [0, 1) into a sequence P of integers lying in the set[2] $\{1, 2, \cdots, m \times n\}$.

One approach consists of taking P as the ordering of $\{1, 2, \cdots, m \times n\}$ induced by the indexing of VDH_b, i.e., $p_i = j$ if and only if x_i is the j-th element in the ordered list of VDH_b's elements. For instance, the sequence $A = \{0.8, 0.2, 0.7, 0.4\}$ induces, through this mechanism, the sequence $B = \{4, 1, 3, 2\}$, meaning that in the ordered listing of A its original elements hold respectively the fourth, first, third and second positions. Through this approach, the repetitions are avoided but the convergence is preserved.

Pixel decimation is used to reduce the number of compared pixels while measuring the distortion for each block during the motion estimation search. Figure 5 shows an example of an 8×8 lattice on an 16×16 block for the VDH pattern. The VDH pattern presents an $N : 1$ subsampling lattice that can provide a computational cost improvement by a factor of N. This approach can also be referred to as VDH-based-pixel-decimation pattern.

Taking into consideration that motion estimation within H.264/AVC uses variable size blocks, VDH decimation in lattice patterns is consistent to the codec's structure. According to Section 3, the H.264 encoding standard works with a hierarchical flow where any given block may be broken into smaller blocks. Lattice decimation is able to use a predefined pattern for an 8×8 block when coding larger blocks (8×16 or 16×16 for example). This is also suitable for the RDO optimization used by the Reference Software for the same reasons stated above. This is possible due to the recursive mechanics of codecs. Each layer is formed by equal sized blocks. The blocks are then formed recursively by combining a lower layer in lattice patterns and can, in turn, be broken down into more blocks by advancing a layer. In this manner, a 16×16 block is formed by four 8×8 which are in turn formed by another four 4×4 blocks. As the block hierarchy is optimally chosen by the H.264/AVC Standard, the lattice-formed VDH decimation pattern is able to adjust to the adequate block size.

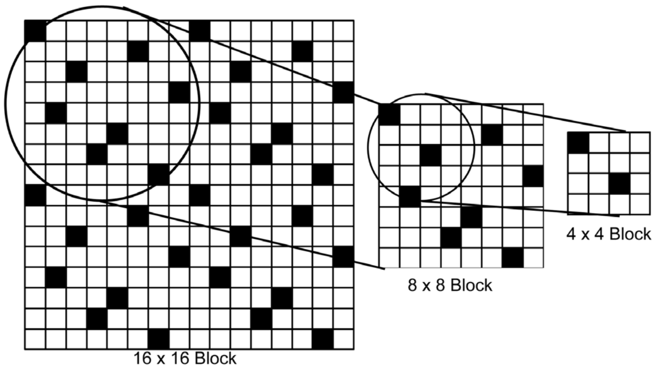


Figure 5: VDH subsampling pattern for an 8×8 lattice on a 16×16 block.

[2]In fact, the sequence P in the definition of induced metric is a subset of $\{1, 2, \cdots, m\} \times \{1, 2, \cdots, n\}$. By using the standard linearization of a matrix, it is enough to consider P as a subset of $\{1, 2, \cdots, m \times n\}$

For the particular case of the Full Search algorithm, consider a sequence $\{P_l\}_{l=1\cdots k} = \{(a_l, b_l)\}_{l=1\ldots k}$ where $a_1, a_2, \ldots, a_k$ is the VDH sequence in base 2 and $b_1, b_2, \ldots, b_k$ is the VDH sequence in base 3. Normalizing the sequences $a_1, a_2, \ldots, a_k$ for the interval $[1, M]$ and $b_1, b_2, \ldots, b_k$ for the interval $[1, N]$ we obtain a scan of the whole block by a sequence denoted by P_l.

We emphasize that no matter which approach is used for the transformation, or how complex the sequence generation is, only one sequence needs to be generated for each block size. This sequence can then be used to produce any estimation of block distortion measure (MAD) as a function of the number of sampled points in the sequence.

5 RESULTS AND DISCUSSION

The proposed pixel-decimation patterns were implemented using Open Source *Scilab* software without any encoding and on the H.264/MPEG-4 AVC Reference Software version 18.3 [8]. We performed simulations on the following input sequences: *Soccer*, *Salesman*, *Akiyo*, *Coastguard*, *Foreman*, *News* and *Claire*, which combined presents various movement types (i.e: fast movement, slow movement, etc.). The chosen video sequences are standard test sequences.

Figures 6 and 7 present the mean squared error (MSE) values between a frame estimated by the FS algorithm and another estimated by FS using low discrepancy sequence decimation (FS+VDH) for all the VDH-based subsampling patterns, ranging from 1% to 100% of the block's pixels. The MSE obtained with the FS+VDH method quickly converges to the pure FS method as the number of sampled pixels approaches 30% of total pixels in the block. The aforementioned result strongly favors a choice of $\frac{3}{10}$ of the total pixels in each block to be subsampled, representing Full Search's approximate MSE with only $\frac{3}{10}$ of the computational effort (worst-case scenario). This represents a 76-pixel MAD computation in a 16×16 block (256 pixels). And even though a smaller subsampling implies in an inferior approximation, the convergence property is mantained.

Table 1 presents the MSE mean values between the original and estimated frames by the FS, TSS, NTSS, 4SS, BBGDS, CDS and FS+VDH algorithms. Due to the limitation of the maximum search range for some of the chosen algorithms, we chose a search range of ±7 with 16×16-size blocks, and 90 frames. We selected a $\frac{1}{8}$ subsampling rate for the FS+VDH algorithm. For the *Akiyo* sequence, the FS+VDH method performed similarly to the FS, but with reduced computational effort regarding the calculation of the MAD. For all sequences, the proposed method (FS+VDH) outperformed all other fast search algorithms.

The Figure 8 shows the efficiency of the proposed method with respect to the (subjective) visual quality of the recovered frame for the Claire video sequence. The Figure 8(a) and Figure 8(b) present the best the worst qualities (MSE) for the recovered frames, respectively. We call to attention that even in the worst case, the estimated frame presents a satisfactory visual quality.

The coded search algorithms were tested using the framework provided by the H.264/AVC Reference Software. We once more stress that lattice decimation is based solely on the similarity

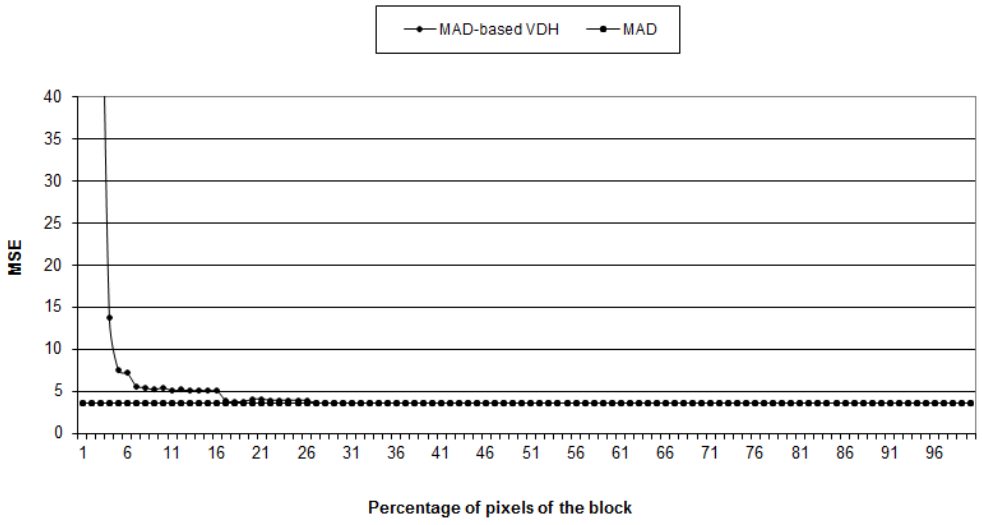


Figure 6: MSE value between the original and the estimated frame for the Claire video sequence's second frame.

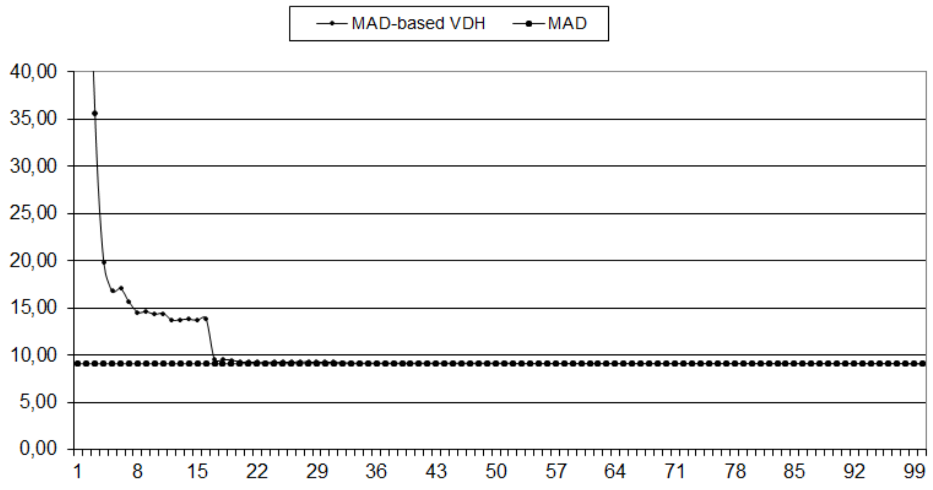


Figure 7: MSE value between the original and the estimated frame for the Salesman video sequence's second frame.

measure. The SAD coding within the software was altered so that VDH decimation could be configured along the simulation. We also included a counter as to measure the number of pixels compared by the SAD similarity measure. The Reference Software encompasses an early termination condition which we did not remove to maintain a practical study-case. The simulation data as well as the total number of comparisons made by the coding of each video sequence can be seen in Table 2. We encoded all sequences using the following test conditions: 150 frames, an IntraPeriod of 4, quantization parameter of 28 and 4 reference frames (NumberReference-

Table 1: MSE mean values.

Algorithm	Akiyo	Coastguard	Foreman	News
FS	5.47	104.53	43.03	18.92
TSS	5.67	118.32	49.62	19.51
NTSS	5.47	112.36	45.34	19.44
4SS	5.57	120.75	47.71	19.51
BBGDS	5.47	123.48	44.92	19.30
CDS	5.54	117.75	46.35	19.30
FS+VDH	5.47	104.88	43.18	18.94

(a) Frame 11 (b) Frame 97

Figure 8: Estimated frames in Claire's video sequence.

Frames). The search range varied according to the video format (32 for HD sequences and 16 for non-HD sequences). The other statistics such as peak noise-signal ratio (PSNR) were measured by the Reference Software's framework. In our experiments, the first column contains the used video sequence (Sequence), the second column displays the used method (Method), the third column shows the peak noise-signal ratio (PSNR), the fourth, the total number of pixel-comparison operations used by each method for that particular sequence (OP) and the last column has the ratio of operations performed by that method in comparison to the Full Search. The time was not added as a comparison parameter since the particularities of the used hardware rapidly evolve, making the actual number of operations a more concrete benchmark for future works.

Again the VDH lattice decimation pattern is shown to outperform the other traditional methods whilst maintaining a near-Full Search PSNR. By the results shown in Table 2 it is clear that adopting the proposed technique conveys a significant advantage to the computational complexity with little or no undermining of the motion estimation quality.

6 CONCLUSION

This paper proposed a faster pixel-decimation method for block motion estimation in H.264/AVC. To assert the advantage of the proposed technique, we simulated video coding using both fast search algorithms available in the Reference Software (UMHS and EPZS). In this

Table 2: H.264 Reference Software Simulation Results

Sequence	Method	PSNR	OP	Ratio over FS
Foreman CIF	FS	37.684	1.67E+11	1
	UMHS	37.650	1.24E+10	13.49
	EPZS	37.688	1.02E+10	16.32
	FS + VDH	37.657	6.76E+10	2.46
	UMHS + VDH	37.636	4.13E+09	40.41
	EPZS + VDH	37.659	2.69E+09	62
Foreman QCIF	FS	36.999	4.36E+10	1
	UMHS	36.997	3.52E+09	12.40
	EPZS	37.021	2.83E+09	15.38
	FS + VDH	36.992	1.68E+10	2.60
	UMHS + VDH	36.976	1.19E+09	36.64
	EPZS + VDH	37.003	7.36E+08	59.28
Soccer HD	FS	37.328	2.63E+12	1
	UMHS	37.300	5.86E+10	44.91
	EPZS	37.332	4.03E+10	65.36
	FS + VDH	37.331	1.03E+12	2.56
	UMHS + VDH	37.302	1.84E+10	143.04
	EPZS + VDH	37.332	1.13E+10	233.98

paper we described why the VDH sequence is a good candidate for a decimation sequence for the H.264/AVC. Our method displayed superior performance in the tested video sequences. The lattice-based approach is also able to adapt to the malleable structure of the RDO H.264/AVC coding, thus presenting a good candidate for a simple but effective means to decrease computational complexity associated with motion estimation.

RESUMO. Este trabalho explora o custo-benefício vantajoso apresentado por sequências de baixa discrepância como uma técnica de decimação de pixeis para melhorar a performance da estimação de blocos no h.264/avc. O método proposto é capaz de estimar vetores de movimento de forma eficiente para algoritmos de casamento de blocos presentes no codec h.264 utilizando decimação ladrilhada amostrada através da sequência van der corput-halton. Este artigo explora a estimação de movimento dentro do h.264, validando-a com casos práticos de codificação de vídeos. Os resultados mostram que esta técnica incorporada ao h.264 é geralmente mais eficiente que outras técnicas de decimação utilizadas em condições semelhantes.

Palavras-chave: estimação de movimento, sequências de baixa discrepância, codificação de vídeo.

REFERENCES

[1] Z.B. Chen, P. Zhou & Y. He. *Fast integer pel and fractional pel motion estimation for JVT*. JVT-F017, 6th Meeting, Awaji, JP, pp. 5–12, (2002).

[2] C.H. Cheung and L.M. Po. A novel cross-diamond search algorithm for fast block motion estimation. *IEEE Transactions on Circuits Systems for Video Technology*, **12**(12) (2002), 1168–1177.

[3] F. James. Monte carlo theory and practice. *Rep. Prog. Phys.*, **43** (1980), 1147–1188.

[4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima & T. Ishiguro. Motion-compensated interframe coding for video conferencing. In *Proceedings of NTC81*, pages C9.6.1–9.6.5, Los Angeles, November (1981).

[5] L. Kuipers & H. Neiderreiter. *Uniform Distribution of Sequences*. Addison Wesley Publishing Company, Inc., New York, (1992).

[6] R. Li, B. Zeng & L.M. Liou. A new three-step search algorithm for block motion estimation. *IEEE Transactions on Circuits Systems for Video Technology*, **4**(4) (1994), 438–442.

[7] L.K. Liu & E. Peig. A block-based gradient descent search algorithm for block motion estimation in video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, **6**(8) (1996), 419–423.

[8] Joint Video Team of ISO/IEC JTC1/SC29/WG11 and ITU-T SG16/Q.6 Doc. JVT-G050. *Draft ITU-T Recommendation H.264 and Draft ISO/IEC 14496-10 AVC*, Mar (2003).

[9] L. Po & W. Ma. A novel four-step search algorithm for fast block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, **6** (1996), 313–317, June.

[10] Iain E.G. Richardson. H.264 and mpeg-4 video compression. Video coding for next-generation multimedia. In *Wiley*, (2005).

[11] A.M. Tourapis. Enhanced predictive zonal search for single and multiple frame motion estimation. In *Visual Communications and Image Processing 2002 (VCIP-2002)*, (2002).

[12] T. Wiegand, G.J. Sullivan, G. Bjntegaard & A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, **13**(7) (2003), 560–576.