Fuzzy quantization based bit transform for low bit-resolution motion estimation

Chuan-Ming Song\textsuperscript{a,b,}\textsuperscript{*}, Yanwen Guo\textsuperscript{b}, Xiang-Hai Wang\textsuperscript{a,b}, Dan Liu\textsuperscript{a}

\textsuperscript{a} College of Computer and Information Technology, Liaoning Normal University, Dalian 116029, China
\textsuperscript{b} National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China

\textbf{Abstract}

This study proposes a novel fuzzy quantization based bit transform for low bit-resolution motion estimation. We formalize the procedure of bit resolution reduction by two successive steps, namely interval partitioning and interval mapping. The former is a many-to-one mapping which determines motion estimation performance, while the latter is a one-to-one mapping. To gain a reasonable interval partitioning, we propose a non-uniform quantization method to compute coarse thresholds. They are then refined by using a membership function to solve the mismatch of pixel values near threshold caused by camera noise, coding distortion, etc. Afterwards, we discuss that the sum of absolute difference (SAD) is one of the fast matching metrics suitable for low bit-resolution motion estimation in the sense of mean squared errors. A fuzzy quantization based low bit-resolution motion estimation algorithm is consequently proposed. Our algorithm not only can be directly employed in video codecs, but also be applied to other fast or complexity scalable motion estimation algorithms. Extensive experimental results show that the proposed algorithm can always achieve good motion estimation performances for video sequences with various characteristics. Compared with one-bit transform, multi-thresholding two-bit transform, and adaptive quantization based two-bit transform, our bit transform separately gains 0.98 dB, 0.42 dB, and 0.24 dB improvement in terms of average peak signal-to-noise ratio, with less computational cost as well.

\textcopyright 2013 Elsevier B.V. All rights reserved.

1. Introduction

Various video services, such as surveillance, video telephony/conferencing, mobile streaming, wireless LAN video in home network, and even beyond high definition video, are becoming available in more and more application scenarios with the rapid development of Internet, wireless communication, and pervasive computing technologies [1,2]. Mass videos are required to be reliably delivered to diverse clients over heterogeneous networks in real time. These clients may have varying capabilities in display resolution, computing power, network bandwidth, etc. An efficient video codec is essential for the video services in such a challenging scenario. Sullivan et al. [3] believed that most of the efficiency improvement of state-of-art video codecs, e.g., H.264/AVC, results from better temporal prediction and compensation. Motion estimation/compensation therefore plays a key role in coding system. Unfortunately, motion estimation is usually remarked as the most computationally intensive component, consuming up to 50\% [4], even to 60–80\% [5], of the computation evolved in the entire codec. Such a heavy computation load will inhibit practical video communication on portable and wireless devices with limited battery power. Hence an efficient motion estimation algorithm with low computational complexity is of great importance for real-time video services cross platforms.
Various fast algorithms have been proposed so far to lower motion estimation cost. They can be roughly classified into four categories.

The first category of algorithms searches for motion vector in a subset of possible candidate vectors, reducing the number of search locations per macroblock. Representative algorithms include three-step search [6], diamond search [7], cross-diamond-hexagon search [8], hexagon-based search [9], unsymmetrical-cross multi-hexagon-grid search [5], motion estimation using dynamic models [10], adaptive neighborhood elimination algorithm [11], adaptive motion search range prediction [12], and fast sub-pixel motion estimation [13].

The main concept of the second category is to compute matching errors using a fraction of pixels, reducing operation number per matching. Typical algorithms involve partial distortion search [14], multiresolution motion estimation [15–17], block matching with an adaptive pattern [18,19], and advanced spatial hierarchical motion estimation [20].

Low-complexity matching metrics are the core of the third category. Commonly used matching metrics include sum of absolute differences (SAD), pel difference classification (PDC), different pixel counts (DPC), bit exclusive-OR (BXOR) [21], minimized maximum error (MiniMax) [22], different pixel counts (DPC), bit exclusive-OR (BXOR) [23], and mean-predicted sum of absolute differences [24].

The fourth category of algorithms performs motion estimation on low bit-resolution pixels whose values are mapped to from eight bit-resolution pixels [25–61]. These algorithms contribute to simple hardware implementation, low power consumption, and memory bandwidth. A cost-effective and real-time video codec can thus be realized on a low-power device with limited capabilities.

In this study, we focus on the last category of algorithms, considering their potential in computational capability constraint devices. Our main contribution is a bit transform method which maps eight-bit values to those with lower bit-resolution. It is the crucial component of low bit-resolution motion estimation, determining the codec performance. We first formalize bit transform by interval partitioning and interval mapping. Through exploiting quantization theory, we then find an initial solution to interval partitioning. To resolve false partitioning caused by inter-frame noises, a membership function is employed to refine the initial thresholds.

Our approach has the following benefits:

1. Bit transform is bound to result in pixels' accuracy loss as well as performance degradation of motion estimation. Addressing bit transform by quantization enables us to better utilize existing theory. This facilitates reducing as much accuracy loss as possible so as to guarantee motion estimation efficiency.

2. Membership function achieves good robustness to coding distortions, camera noises, etc. It can effectively eliminate the mismatch due to hard thresholding, and improve prediction quality of complex videos with high spatial details or fast motion.

The remainder of this paper is organized as follows. Section 2 gives an overview of previous works. Sections 3 and 4 describe the fuzzy quantization approach. A fuzzy quantization based motion estimation is discussed in Section 5. We examine the proposed approach in Section 6 and conclude the paper in the last section.

2. Related works

A considerable amount of research has been done on bit transforms during the last decade. State-of-the-art transforms are mainly based on three approaches.

The first approach is filtering. Feng et al. [25] transformed an eight-bit pixel to its one-bit representation by comparing the pixel against the mean value. Mizuki et al. [26,27] obtained the binary representation by an edge detection procedure. Natarajan et al. [28,52,38] employed a 2-D multi-bandpass filter of size $17 \times 17$ to implement the one-bit transform. Ertürk et al. [29–31,33] proposed a novel filter kernel using shift operation instead of multiplication in [28]. Lee et al. [37,39,49–51] detected a zero-cross phase with mean as the DC bias and used the binary phase deviation as the bit transform result. Luo et al. [34] proposed a linear and symmetric filter to better register average spatial characteristics and to facilitate a more accurate binary representation. Although many improvements have been contributed, these algorithms always involve convolution operations and high computational overhead as a result.

Some studies exploit frequency-domain techniques for less computation. Wu et al. [32] presented a gradient-based thresholding in the discrete cosine transform (DCT) domain. As video coders always work in the DCT domain, this method incurs little overhead. Ertürk et al. [47] carried out the binarization in wavelet domain by only retaining filtered lowest frequency coefficients.

Another limitation of filtering-based bit transforms is that they are only adequate for one-bit motion estimation. In the case of higher bit-resolution, the filtering is required executing for multiple times.

The second approach is pixel truncation. Baek et al. [43] presented a criterion, named reduced bits mean absolute difference, which only uses the most $n$ significant bits to compute the matching errors when the bit-resolution is $n$. He et al. [44,57,60,59,58] truncated the least significant bits to realize the pixel resolution reduction. The number of truncated bits can be fixed or adaptively defined. Nevertheless, Patras et al. [62] stated that motion-compensated differences follow an independent Laplacian distribution. It indicates that the matching errors always occur on insignificant bits. If these bits are truncated, pixels in the area around the best-matched macroblock could be converted into the same low-resolution values. The resulting matching errors corresponding to all candidate vectors in this area will be a constant, disabled from distinguishing the optimal motion vector. Moshnyaga [61] addressed this issue by adaptively truncating the most significant bits according to the data variation.

Truncating the least significant bits tends to influence the motion estimation performance for sequences with low spatial detail or low amount of movement, and vice versa. Therefore, the least significant bit truncation and the most significant bit truncation should be skillfully
combined depending on the pixel distributions, instead of independently used.

The third approach is quantization. Lee et al. [40,41] utilized an adaptive quantization to implement a two-bit transform. Ertürk and Ertürk [4,42,53] proposed a multi-thresholding with mean value and approximated standard deviation. Furthermore, Kim et al. [54,55] made use of positive and negative second derivatives to improve [4,42,53]. Wang et al. [48] selected a low-pass filtered version of the current frame as a mid point, and then applied a uniform quantization with a pre-defined step size to accomplish the bit transform.

Both truncation and quantization approaches can be used for bit transform whose resolution is more than one bit. Their computational complexity is obviously lower than that of the filtering approach. Moreover, the truncation is equivalent to a uniform quantization in essence. The quantization thus provides an efficient strategy for bit transform.

However, current quantization based methods compute thresholds on macroblock basis. Since a search window compasses several macroblocks, the low bit-resolution reference tends to display discontinuities along two adjacent blocks if different thresholds are selected [32]. The motion estimation accuracy will be influenced by the blocking artifact. Even if a uniform threshold is adopted by the truncation-based approach, it cannot adapt to scene changes and would fail to provide a fine grain scratch for low bit-resolution motion estimation.

On the other hand, state-of-art bit transforms adopt a hard thresholding manner which regards pixel values lying on the opposite sides of the threshold as a mismatch even if their values are close. Because of camera noises, non-translational motion, quantization errors during coding, etc., there always exist differences between two best matched macroblocks [63]. Consequently, hard thresholding would take the best pair as a mismatch. Urban et al. [35,36] counted the pixel values away from threshold within a certain distance D as a match regardless of their values. But they neglected adjusting D to video characteristics. Large D will ignore the pixel differences that should have excluded invalid candidate vectors. On the contrary, small D cannot resist inter-frame noises and is not able to avoid an improper match.

In general, little attention has been payed to an adaptive uniform thresholding and its mechanism to deal with inter-frame noises, even though they will definitely contribute to performance improvement.

3. A bit transform based on quantization

Motion estimation with low bit-resolution is always inferior to that with full bit-resolution in terms of motion-compensated quality. The main reason is that the bit transform leads to a data loss. It is therefore crucial to reserve as much information of full resolution video as possible. In this section, we present a quantization based bit transform to address this issue.

Set $G = \{0, 1, 2, \ldots, 255\}$, $B = \{0, 1, 2, \ldots, 2^N - 1\}$. Then the bit transform mapping eight-bit values to those with N bit-resolution can be formalized by a map $f : G \rightarrow B$, $g \rightarrow b$.

Divide $f$ into two steps, namely interval partitioning $t : G \rightarrow G$, $g \rightarrow r$ and interval mapping $m : G \rightarrow B$, $r \rightarrow b$, and then we have $f(g) = m(t(g))$, i.e.,

$$
t(g) = \begin{cases}
1, & -1 < g \leq T_1 \\
2, & T_1 < g \leq T_2 \\
3, & T_2 < g \leq T_3 \\
\vdots & \\
2^m - 1, & T_{2^m-1} < g \leq 255
\end{cases}
$$

and

$$
m(r) = \begin{cases}
0, & r = r_1 \\
1, & r = r_2 \\
2, & r = r_3 \\
\vdots & \\
2^N - 1, & r = r_{2^N}
\end{cases}
$$

The interval partitioning is a many-to-one mapping, while the interval mapping is a one-to-one mapping. Hence the former decides the threshold selection as well as motion estimation performance. To preserve as much important information as possible in the low bit-resolution representation, we need to find an optimal mapping $m$ that minimizes

$$
D = E[(g - t(g))^2] = \sum_{g=0}^{255} p(g)(g - t(g))^2.
$$

in which $E[\cdot]$ and $p(\cdot)$ denote expectation operator and probability distribution function (PDF) of pixel values, respectively. However, finding $t$ is equivalent to simultaneously determining an optimal set of $r_i$ and $T_j(i \in \{1, 2, \ldots, 2^N\}, j \in \{1, 2, \ldots, 2^N - 1\})$. This is a nonlinear problem and cannot be solved with an easy.

Assume that $g$ is a discrete random variable whose PDF is $p$. As we know, optimum scalar quantization is to decide a set of thresholds and quanta that minimizes reconstruction errors. It has the same issue with interval partitioning. This motivates us to recourse quantization theory to solve Eqs. (1) and (3). According to the quantization theory, a uniform quantizer yields better approximation than a non-uniform one at high resolution [64], and vice versa. We thus adopt different strategies to accomplish the interval partitioning depending on the bit-resolution.

In the case that bit-resolution is higher than a threshold $R$, we use a set of uniform thresholds to define $T_j$ as

$$
T_j = 2^8 - Nj - 1.
$$

Set $T_0 = -1$, then we have

$$
r_i = [(T_i + T_{i-1})/2].
$$

When bit-resolution is lower than $R$, non-uniform thresholds are employed. Max [65] and Lloyd [66] ever proposed iterative trial-and-error procedures to adaptively compute non-uniform quantization steps. They also gave a list of parameters for signals following uniform, Gaussian, and Laplacian distributions. Unfortunately, hardly video signals obey any explicit and identical probability distribution. This forces us to run their methods frame-wise to assign optimum thresholds for each frame. Obviously,
this is impractical with respect to the computational complexity.

As we know, the uniform quantization is the optimum quantizer if a signal’s PDF is a uniform distribution. This conclusion enlightens us on an efficient solution to the interval partitioning. Provided that we can find a transform \( e \) from \( g \) to a random variable \( g' \in (0, 1, 2, ..., 255) \) satisfying \( g' \sim U(0, 255) \), we will easily obtain the optimum thresholds of \( g' \). Afterwards, the interval partitioning of \( g \) can be realized through inversely transforming the computed thresholds using \( e^{-1} \). Based on this idea, we present our method below.

**Algorithm 1 (Non-uniform interval partitioning).**

1. Calculate histogram \( p \) of the input frame.
2. Employ histogram equalization to map \( g \) to \( g' \), namely
   \[
   e(g) = \left\lfloor \frac{255 \times p(k)}{\sum_{k=0}^{255} p(k)} \right\rfloor.
   \]
3. Uniformly quantize \( g' \) to obtain threshold \( T_j = 2^k - k - 1 \), and \( \hat{T}_j \) by an inverse transform \( e^{-1} \). i.e., \( T_j = e^{-1}(\hat{T}_j) \).
4. Compute \( T_j \) by a hard thresholding manner. But we introduce a fuzzy logic into the bit transform.

After the non-uniform interval partitioning above, Eq. (5) is utilized again to compute \( r_j \).

Algorithm 1 is able to handle video frames obeying any PDF with low computational complexity. It should be noted that, however, Algorithm 1 only offers a suboptimal solution because histogram equalization cannot produce an ideal uniform distribution for discrete pixel values. Substitute \( r_j \) into Eq. (2), and combine it with Eq. (1). Then we map the current and reference frames to \( B \), on which the low bit-resolution motion estimation is executed. Now, our bit transform still adopts a hard thresholding manner. But observed that the hard thresholding always leads to performance degradation. We will analyze the reason in the next subsection. Afterwards, we will introduce a fuzzy logic into the bit transform.

**4. Threshold refinement using fuzzy logic**

Block-based motion estimation generally assumes that a pixel and its correspondence in a successive frame have the same value. Indeed, it is not the case. Due to camera capability, coding distortion, light changes, etc., the current macroblock is usually different from its best matched macroblock. Let \( M_c \) and \( M_r \) be the macroblocks separately in current and reference frames, respectively. \( M_c^* \) denotes the optimal prediction of the current macroblock. Then we may have \( M_c^*(x, y) > T_j \) and \( M_r(x, y) \leq T_j \), or \( M_c^*(x, y) \leq T_j \) and \( M_r(x, y) > T_j \). At this moment, \( M_c^*(x, y) \) and \( M_r(x, y) \) will be mapped to different intervals and produce matching errors. For low bit-resolution motion estimation, the matching error of a pixel is usually small. For example, the maximum matching error is just four in the case of two-bit motion estimation. This indicates that an inaccurate interval mapping brings about 25% matching deviation at least. Such a large deviation will inevitably lead to a false motion vector. In contrast, the maximum matching error of full bit-resolution motion estimation is 255, thus small matching error, e.g., one, takes up a negligible deviation. This is the reason why low bit-resolution motion estimation, especially one- or two-bit motion estimation, is sensitive to the threshold selection. To summarize, a bit transform is unfavorable for motion estimation when ignoring inter-frame noises and pixel changes.

**4.1. Membership function**

To find optimum motion vectors, it is necessary to estimate the probability that each pixel value within a certain distance from initial threshold belongs to its current interval. If the probability is low, modify the initial threshold such that each pixel value around it is positioned in the right interval. In our study, we use a membership function to describe the probability of a pixel value belonging to each interval.

**Definition.** \( \{T_1, T_2, ..., T_{2^N-1}\} \) gives an initial interval partitioning of \( G \), obtaining \( 2^N \) intervals namely \( Z_0, Z_1, ..., Z_{2^N-1} \). We define the probability that \( g \) is in \( Z_j \) as membership whose value depends on a membership function. The membership function is described as follows:

\[
M_1(g) = \int_{g}^{+\infty} p_1(s) \, ds, \tag{9}
\]

\[
M_2(g) = 1 - \int_{-\infty}^{g} p_{j-1}(s) \, ds - \int_{-\infty}^{g} p_j(s) \, ds, \tag{10}
\]

\[
M_{2^N}(g) = \int_{-\infty}^{g} p_{2^N-1}(s) \, ds, \tag{11}
\]

where \( M_j(g) \) denotes the membership function. \( p_j \) is the probability density function of \( T_j \), which relates to inter-frame noises and video signal itself. In this study, we assume that the inter-frame noises are caused by camera capability and coding distortion, and they are independent of each other and \( M_r(x, y) \). Since the light condition changes between successive frames could be ignored, we do not take it into consideration. Further, suppose that the noises due to camera capability and coding distortion follow normal distributions, namely \( N(0, \sigma_1^2) \) and \( N(0, \sigma_2^2) \), respectively, then we define \( p_j \) as

\[
p_j(s) = \frac{1}{\sqrt{2\pi(\sigma_1^2 + \sigma_2^2)}} \exp\left[-\frac{(s-T_j)^2}{2(\sigma_1^2 + \sigma_2^2)}\right]. \tag{12}
\]

Fig. 1 depicts the membership function in the case of \( N=4 \). Note that different probability density functions can also be accepted for different \( T_j \) according to the application scenarios.
4.2. Initial threshold refinement

Upon defining the membership function, the optimum interval partitioning corresponds to \( \{T_1, \ldots, T_{2^n-1}\} \) that maximizes

\[
\sum_{j=1}^{2^n} \sum_{g=T_{j-1}+1}^{T_j} M_j(g)p(g) \quad \text{s.t.} \quad |T_j - T_j| \leq \sqrt{\sigma_2^2 + \sigma_d^2} \tag{13}
\]

in which \( T_0 = -1 \) and \( T_{2^n} = 255 \). The last term in Eq. (13) constrains difference between \( |T_j| \) and \( |T_j| \). Otherwise, \( |T_j| \) may diverge from the suboptimal thresholds \( |T_j| \) solved in Section 3. As we know about the normal distribution, nearly 70% of the random variation amplitude is within one variance. Therefore, we seek \( T_j \) around \( T_j \) in a range of one variance. In the following, we discuss our solution to Eq. (13) under two conditions.

First, when \( p \) is the uniform distribution, \( p(\cdot) \) can be extracted as a common factor from Eq. (13). Now, that Eq. (13) is maximum is equivalent to maximizing

\[
\sum_{j=1}^{2^n} \sum_{g=T_{j-1}+1}^{T_j} M_j(g) \quad \text{s.t.} \quad |T_j - T_j| \leq \sqrt{\sigma_2^2 + \sigma_d^2}. \tag{14}
\]

From Fig. 1 we can find that the solution of \( T_j \) is the intersection of \( M_j \) and \( M_{j+1} \), i.e., \( T_j \). This tells us that uniform quantization provides the optimal interval partitioning for uniform distribution, needing not to be refined.

Second, \( p \) is non-uniform. Since most image pixels follow non-uniform distributions, video sequences are usually in this case. We could employ an optimization method to resolve Eq. (14) for each frame. This will consume considerable computational complexity. Here we present a compromise method for less computation.

Although the first condition can hardly be true for natural videos, its solution sheds a light on how to treat the second condition. The fundamental idea is to replace \( p(\cdot) \) of each interval using its average probability \( \overline{p} \) so that Eq. (13) is transformed to a similar form to Eq. (14). Because \( T_j \) is computed by means of histogram equalization, pixels’ accumulate probability distribution in each \( Z_i(i \in \{0, 1, \ldots, 2^N-1\}) \) is approximately the same. Let \( |Z_i| = T_{i+1} - T_i \) be the interval length. Then the smaller the \( |Z_i| \) is, the higher its \( \overline{p} \) will be. Furthermore, the interval length of a uniform distribution is 256/N, we thus use 256/(\( N \times |Z_i| \)) as a weight to roughly measure \( \overline{p} \) and simplify Eq. (13) to

\[
\sum_{j=1}^{2^n} \left\{ \frac{256}{(T_j - T_{j-1})N} \sum_{g=T_{j-1}+1}^{T_j} M_j(g) \right\}. \tag{15}
\]

From Eq. (15), \( T_j \) is closely related to weights of the corresponding intervals. Large difference between \( |Z_{j-1}| \) and \( |Z_j| \) (suppose \( |Z_{j-1}| > |Z_j| \)) indicates that the numbers of pixel values locating in \( Z_{j-1} \) and \( Z_j \) vary considerably. The probability is high that the pixel values within a certain distance from threshold \( T_j \) belong to \( Z_p \), since these pixel values are much probably partitioned into \( Z_{j-1} \) due to noises. Hence, the \( T_j \) should be moved toward \( T_{j-1} \) from \( T_j \). Based on this inference, we propose to adaptively refine \( T_j \) according to \( |Z_i| \). The procedure is shown in Algorithm 2.

Algorithm 2 (Initial threshold refinement).

1. Set \( j = 0 \).
2. If \( |Z| \leq 256i/N \), then let \( \Delta = \sqrt{\sigma_2^2 + \sigma_d^2} \times (256/N - |Z|)/(256/N) = \sqrt{\sigma_2^2 + \sigma_d^2} \times (256 - |Z|)/256, |Z| = |Z| + \Delta \). Otherwise, set \( |Z| = |Z| \), go to Step 4.
3. Let \( j = j + 1 \). If \( j = N \), go to Step 4. Otherwise, go to Step 2.
4. Let \( L = \sum_i |Z_i| \), refine the length and threshold of each interval by setting \( |Z_i| = 256 \times |Z_i|/L \), \( j \in \{0, 1, \ldots, 2^N-1\}, T_0 = -1, T_{2^n} = 255, T_j = T_{j-1} + |Z_{j-1}|, i \in \{1, 2, \ldots, 2^N-1\} \).

In the Step 2 of the above algorithm, \( \lambda \) is used to control computational complexity. The small the \( \lambda \) is, the less the complexity will be. Furthermore, \( (256/N - |Z|)/(256/N) = (256 - |Z|)/256 \) is to normalize \( \Delta \) so that the length variation of each adjusted interval is within one variance \( \sqrt{\sigma_2^2 + \sigma_d^2} \), as discussed in Eq. (13).

The remaining question is how to compute \( \sqrt{\sigma_2^2 + \sigma_d^2} \). Since we assume the inter-frame noises are independent of \( M_i(x, y) \), we have \( \sigma_d^2 = \sigma_2^2 + \sigma_2^2 + \sigma_2^2 \), where \( \sigma_d^2 \) and \( \sigma_2^2 \) denote separately the variances of pixel values in the current and reference frames, respectively. We thus calculate \( \sqrt{\sigma_2^2 + \sigma_2^2} \) by \( \sqrt{\sigma_2^2 + \sigma_d^2} \).

By substituting the computed \( |T_j| \) into Eqs. (1), (2) and (3), we obtain the low bit-resolution representation \( B \) of each frame. Before presenting our low complexity motion estimation approach on \( B \), we discuss the computational complexity of our bit transform.

4.3. Computational complexity of our bit transform

Without particularly pointed out, addition and subtraction operations are generally called “addition” hereafter, and multiplication and division called “multiplication”. The computational complexity of our proposed bit transform depends on two steps.

First, histogram equalization. Assume that the size of video frame is \( w \times h \) pixels. Counting the histogram \( p \) of the current frame requires \( w \times h \) additions. \( c \) can be
computed using 256 multiplications and 255 additions. Solving $e^{-1}$ needs 255 comparison operations. Note that, when $N \geq R$, the histogram equalization is not required, so are the above operations.

Second, threshold refinement. The mean value $\mu_c$ and $\mu_r$ of the current and reference frames are computed by total $2 \times (w \times h - 1)$ additions. In order to calculate the variance $\sigma_c^2$ and $\sigma_r^2$, $2 \times w \times h$ multiplications and $2 \times w \times h$ additions are required. Furthermore, one addition, one absolute conversion, and one square root operation are needed to obtain $\sqrt{\sigma_c^2 + \sigma_r^2}$. Afterwards, Step 2 of Algorithm 2 executes at most $2^{(N+1)}$ additions and $2 \times (2^N - 1) + 2$ multiplications. Step 4 is composed of $3 \times (2^N - 1)$ additions and $2 \times (2^N - 1)$ multiplications. In addition, quantizing each pixel needs at most $(2^N - 1)$ comparisons.

5. Low bit-resolution motion estimation

This section first talks over the matching metric suitable for low-bit-resolution motion estimation. A novel motion estimation approach is then developed.

5.1. Matching metric for low bit-resolution motion estimation

An efficient matching metric is one of the important components for motion estimation. Bahari et al. [59] found that DPC and BXOR metrics show good performance in the low-bit-resolution motion estimation. Among most low-bit-resolution motion estimation algorithms, the DPC metric is adopted. It can distinguish two situations only, i.e., match and mismatch. Nevertheless, the number of matching situations is more than two when $N > 1$. This implies that DPC cannot well exploit the information provided by high-bit-resolution pixels.

The BXOR metric was also researched [59,67]. This metric can be regarded as an operation on two $N$-dimensional vectors in a duality field $\{(0,1)\}$ where “⊕” and “⊗” denote “Exclusive-OR” and “AND” operations, respectively. Unfortunately, the output of BXOR does not completely conform to the real difference between the two pixel values. For example, if two pixels’ representations are separately “01” and “10”, the metric output is “11” (Fig. 2) which means that their difference is three. Obviously, the correct difference equals one. In such a case, the BXOR metric cannot distinguish between (01, 10) and (00, 11). Actually, (01, 10) is a better match than (00, 11). For three-bit matching, BXOR metric presents the same matter as shown in Fig. 2(b).

We believe that SAD is still the accurate and efficient matching metric for low-bit-resolution motion estimation. In our study, we adopt a bit-truncation SAD [23] defined as

$$h(u,v) = \sum_{x=0}^{S-1} \sum_{y=0}^{S-1} |I_c^{(q+N-1)}(x,y) - I_r^{(q+N-1)}(x+u,y+v)|$$

(16)

in which $I_c(\cdot, \cdot)$ and $I_r(\cdot, \cdot)$ are separately pixel values of the current and reference frames. $q$: $q+N-1$ indicates that operations are only executed on from the $q$-th to the $(q+N-1)$-th bit of each byte. On one hand, we could store multiple pixel values in one byte, available for parallel processing. On the other hand, we are able to create a table in memory to save the matching errors under all situations to realize metric (16) with low computational cost. Using $E_{r}^{q+N-1}(x,y)$ and $E_{c}^{q+N-1}(x+u,y+v)$ as an offset, we then employ the XLAT instruction of the assembly language to locate the matching error in the aforementioned table. This method requires less clock cycles than sequentially performing subtraction and absolute conversion operations.

5.2. Fuzzy quantization based low bit-resolution motion estimation

On the basis of the concept mentioned above, we present a low bit-resolution motion estimation approach based on fuzzy quantization. The current and reference frames are first processed by the bit transform proposed in Sections 3 and 4, respectively. Next, one of the conventional search strategies, e.g., full search, is applied for motion estimation using $h$ as the matching metric. This approach is summarized in Algorithm 3.

Algorithm 3 (Low bit-resolution motion estimation).

1. Separately compute the mean pixel values $\mu_c = E[I_c]$ and $\mu_r = E[I_r]$ of the current and reference frames, and their variance $\sigma_c^2 = E[I_c^2] - E^2[I_c]$ and $\sigma_r^2 = E[I_r^2] - E^2[I_r]$.
2. Carry out the histogram equalization (see Eq. (6)) on reference frame to obtain $g^r$.
3. Uniformly quantize $g^r$ and determine $\{T_j\}$. 

![Figure 2. Inaccurate matching evaluation by BXOR metric. (a) Two-bit BXOR metric. (b) Three-bit BXOR metric.](image-url)
4. Use \( e^{-1} \) (Eq. (7)) to solve initial thresholds \( T_j \).

5. Compute \( \sqrt{\sigma_c^2 + \sigma_d^2} \) by \( \sqrt{\sigma_c^2 + \sigma_d^2} = \sqrt{|\sigma_{cd}^2 - \sigma_{dc}^2|} \).

6. Employ Algorithm 2 to calculate \( T_j \) and perform the bit transform on current and reference frames. Note that the fuzzy quantization thresholds of the current and reference frames are identical.

7. Conduct block-based full search on low bit-resolution representations, using \( h \) to evaluate matching differences.

6. Experimental results and discussions

To verify the effectiveness of our bit transform in motion estimation, extensive experiments were conducted on the first 90 frames of several test sequences in various formats, which are listed in Table 1. The experimental platform is MPEG-4 verification software with version number "Microsoft-FDAM1-2.3-001123". The parameters are set as default configuration except the ones below: (1) GOP size is 16 with 1 I-frame and 15 P-frames. (2) Rate control algorithm is "TM5". (3) The quantization type is "H263". Motion estimation was carried out with integer-pel accuracy, using a block size of 16 \( \times \) 16 pixels and 8 \( \times \) 8 pixels in a search range of \( \pm 16 \) pixels. In addition, the parameters \( \lambda \) and \( R \) of our bit transform are respectively set to 0.625 and 4 which are determined through experiments. The quality of decoded video is evaluated by the Peak Signal-to-Noise Ratio (PSNR). We also compared the results against several representative methods, including
filtering-based one-bit full search [28] (1BT), adaptive quantization-based two-bit full search [23] (AQ-2BT), multithresholding-based two-bit full search [4] (MT-2BT), and eight-bit full search with SAD matching metric (8BT). Since state-of-the-art literatures research only one- or two-bit motion estimation, the experimental results of our algorithm (FQ-2BT) are not discussed in the case of three-and higher-bit representation in this study.

6.1. Subjective quality comparison

In the first experiment, we compared the subjective quality of decoded frames by our algorithm with those by representative algorithms at the bit-rate of 128 kbps (kilo bits per second). Figs. 3 and 4 illustrate the reconstructed 3rd frames of “Mobile & Calendar” sequence using various motion estimation algorithms with a block size of 16 $\times$ 16 and 8 $\times$ 8 pixels, respectively.

From Fig. 3, we can find that the reconstructed frames by 1BT and MT-2BT algorithms present the most serious ringing artifacts at edges. Several numbers on the calendar, e.g. 10, 17, and 24, are very blurred even missing. The result by AQ-2BT algorithm suffers less ringing artifact around number borders, but still has number “17” lost and one fake edge at the left side of the calendar. Although the PSNR of FQ-2BT is 0.14 dB lower than that of 8BT, FQ-2BT and 8BT algorithms contribute similar subjective quality which displays clearer numbers than those of 1BT, MT-2BT, and AQ-2BT.
In contrast, all algorithms achieve less ringing artifacts in Fig. 4 than those in Fig. 3. This indicates that motion estimation with a block size of 8 x 8 pixels gains better prediction for image details than that with 16 x 16 macroblocks. However, low bit-resolution pixels always present poorer details than eight-bit depth pixels, so that the motion estimation with small macroblocks tends to incur aperture effect and even ubiquitous matching. Hence, there is a slight drop in the PSNR of low bit-resolution motion estimation algorithms, except for our algorithm. The PSNR decrease of 1BT, MT-2BT, and AQ-2BT is separately 0.37 dB, 0.14 dB, and 0.08 dB. It demonstrates that our bit transform reaches the minimum information loss and better preserves the details and texture features of original video frames among all bit transform methods, obtaining high motion-compensation efficiency.

### 6.2. Objective quality comparison

In the second experiment, we compared our bit transform with previous methods in terms of PSNR. To better understand the two individual steps of our algorithm, namely the non-uniform interval partitioning and the threshold refinement, we also analyzed the performance of the bit transform only with non-uniform interval partitioning (NUQ-2BT). Table 2 shows the performance comparison results for a block size of 16 x 16 pixels, while Table 3 lists the results for a block size of 8 x 8 pixels. Figs. 5–8 illustrate the PSNR curves of decoded “Flower”, “Paris”, “Harbour”, and “Grandma” sequences by six motion estimation algorithms. As can be seen from Tables 2 and 3, the proposed bit transform and its motion estimation are superior to other bit transforms and low bit-resolution motion estimation methods including the NUQ-2BT in terms of average PSNR. Moreover, the performance of our algorithm is very stable on each frame demonstrated in Figs. 5–8.

Among four previous low bit-resolution methods, 1BT performs worst whose average PSNR is 1.88 dB lower than that of 8BT algorithm, since one-bit representation cannot provide rich enough pattern features for block matching. Two-bit depth pixels contain more detailed features of video scenes. The MT-2BT algorithm thus leads to a PSNR of up to 0.56 dB higher than 1BT, which is 1.32 dB lower than 8BT. Nevertheless, the bit transform of MT-2BT adopts varying thresholds for different macroblocks. This will inevitably produce discontinuities at the boundaries of neighboring macroblocks in one search window.

### Table 2

Average PSNR (dB) of various reconstructed sequences by each algorithm with a block size of 16 x 16 pixels.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Bit-rate (kbps)</th>
<th>8BT</th>
<th>1BT</th>
<th>MT-2BT</th>
<th>AQ-2BT</th>
<th>NUQ-2BT</th>
<th>FQ-2BT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flower</td>
<td>128</td>
<td>23.09</td>
<td>22.81</td>
<td>22.88</td>
<td>22.89</td>
<td>22.87</td>
<td>23.00</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>23.14</td>
<td>22.86</td>
<td>22.92</td>
<td>22.94</td>
<td>22.94</td>
<td>23.06</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>23.39</td>
<td>22.92</td>
<td>22.97</td>
<td>23.12</td>
<td>23.15</td>
<td>23.39</td>
</tr>
<tr>
<td>Football</td>
<td>128</td>
<td>28.60</td>
<td>26.92</td>
<td>27.85</td>
<td>27.75</td>
<td>27.77</td>
<td>27.96</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>29.03</td>
<td>26.99</td>
<td>27.92</td>
<td>27.88</td>
<td>27.87</td>
<td>28.05</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>30.19</td>
<td>27.13</td>
<td>28.52</td>
<td>28.81</td>
<td>28.63</td>
<td>28.95</td>
</tr>
<tr>
<td>Mobile</td>
<td>128</td>
<td>22.17</td>
<td>21.76</td>
<td>21.81</td>
<td>21.88</td>
<td>22.02</td>
<td>22.04</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>22.22</td>
<td>21.82</td>
<td>21.85</td>
<td>21.97</td>
<td>22.10</td>
<td>22.09</td>
</tr>
<tr>
<td>Paris</td>
<td>128</td>
<td>25.03</td>
<td>24.35</td>
<td>24.56</td>
<td>24.60</td>
<td>24.75</td>
<td>24.76</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>27.85</td>
<td>26.27</td>
<td>26.82</td>
<td>26.98</td>
<td>27.32</td>
<td>27.37</td>
</tr>
<tr>
<td>Stefan</td>
<td>128</td>
<td>24.14</td>
<td>23.59</td>
<td>23.83</td>
<td>23.76</td>
<td>23.87</td>
<td>23.87</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>24.36</td>
<td>23.67</td>
<td>23.92</td>
<td>23.86</td>
<td>23.95</td>
<td>23.98</td>
</tr>
<tr>
<td>Tennis</td>
<td>128</td>
<td>27.50</td>
<td>25.96</td>
<td>26.72</td>
<td>26.76</td>
<td>26.82</td>
<td>26.85</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>28.48</td>
<td>26.28</td>
<td>27.12</td>
<td>27.31</td>
<td>27.50</td>
<td>27.55</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>29.26</td>
<td>27.06</td>
<td>27.79</td>
<td>28.18</td>
<td>28.25</td>
<td>28.33</td>
</tr>
<tr>
<td>Tempete</td>
<td>128</td>
<td>24.32</td>
<td>23.82</td>
<td>23.92</td>
<td>23.96</td>
<td>24.03</td>
<td>24.09</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>25.69</td>
<td>24.83</td>
<td>24.92</td>
<td>25.10</td>
<td>25.27</td>
<td>25.36</td>
</tr>
<tr>
<td>City</td>
<td>1024</td>
<td>27.64</td>
<td>25.94</td>
<td>26.18</td>
<td>26.26</td>
<td>26.74</td>
<td>26.81</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>29.12</td>
<td>27.12</td>
<td>27.21</td>
<td>27.64</td>
<td>28.08</td>
<td>28.23</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>30.23</td>
<td>28.74</td>
<td>28.83</td>
<td>29.26</td>
<td>29.38</td>
<td>29.49</td>
</tr>
<tr>
<td>Crew</td>
<td>1024</td>
<td>30.83</td>
<td>28.87</td>
<td>29.51</td>
<td>29.46</td>
<td>29.60</td>
<td>29.69</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>32.34</td>
<td>29.63</td>
<td>30.48</td>
<td>30.65</td>
<td>31.04</td>
<td>31.08</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>33.61</td>
<td>31.29</td>
<td>32.11</td>
<td>32.25</td>
<td>32.44</td>
<td>32.47</td>
</tr>
<tr>
<td>Harbour</td>
<td>1024</td>
<td>26.10</td>
<td>25.22</td>
<td>25.36</td>
<td>25.56</td>
<td>25.67</td>
<td>25.69</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>27.17</td>
<td>25.48</td>
<td>25.65</td>
<td>25.86</td>
<td>26.36</td>
<td>26.46</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>26.79</td>
<td>26.79</td>
<td>27.02</td>
<td>27.62</td>
<td>27.67</td>
<td>27.80</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>28.53</td>
<td>26.29</td>
<td>26.67</td>
<td>26.98</td>
<td>27.57</td>
<td>27.62</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>32.28</td>
<td>29.48</td>
<td>30.55</td>
<td>31.11</td>
<td>31.33</td>
<td>31.40</td>
</tr>
<tr>
<td>Grandma</td>
<td>32</td>
<td>28.94</td>
<td>27.63</td>
<td>28.02</td>
<td>28.12</td>
<td>28.13</td>
<td>28.26</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>31.32</td>
<td>28.20</td>
<td>28.79</td>
<td>30.22</td>
<td>30.09</td>
<td>30.41</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>35.14</td>
<td>33.18</td>
<td>34.20</td>
<td>34.74</td>
<td>34.29</td>
<td>34.38</td>
</tr>
</tbody>
</table>
Furthermore, MT-2BT uses the variance of a video frame to complete bit transform. Because the encoder's scalar quantization always affects pixels' variance, the bit transform results tend to be inaccurate. Both the two factors of MT-2BT mislead its search for optimum matching block so as to degrade the encoder's coding efficiency. Through introducing an adaptive quantization strategy and a uniform threshold for blocks in one search window, AQ-2BT algorithm improves the bit transform quality and achieves 0.17 dB higher PSNR than that of MT-2BT. But AQ-2BT does not take scalar quantization noises into consideration either. Its hard thresholding technique brings about an

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Bit-rate (kbps)</th>
<th>8BT</th>
<th>1BT</th>
<th>MT-2BT</th>
<th>AQ-2BT</th>
<th>NUQ-2BT</th>
<th>FQ-2BT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flower</td>
<td>128</td>
<td>23.29</td>
<td>22.47</td>
<td>22.66</td>
<td>22.73</td>
<td>22.63</td>
<td>22.91</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>23.32</td>
<td>22.49</td>
<td>22.68</td>
<td>22.76</td>
<td>22.66</td>
<td>22.94</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>23.34</td>
<td>22.51</td>
<td>22.70</td>
<td>22.78</td>
<td>22.67</td>
<td>22.96</td>
</tr>
<tr>
<td>Football</td>
<td>128</td>
<td>29.91</td>
<td>27.06</td>
<td>28.19</td>
<td>28.35</td>
<td>28.07</td>
<td>28.41</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>30.03</td>
<td>27.14</td>
<td>28.28</td>
<td>28.45</td>
<td>28.12</td>
<td>28.50</td>
</tr>
<tr>
<td>Mobile</td>
<td>128</td>
<td>22.65</td>
<td>21.37</td>
<td>21.73</td>
<td>21.84</td>
<td>22.02</td>
<td>22.05</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>22.68</td>
<td>21.39</td>
<td>21.75</td>
<td>21.87</td>
<td>22.05</td>
<td>22.08</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>22.71</td>
<td>21.41</td>
<td>21.77</td>
<td>21.89</td>
<td>22.07</td>
<td>22.10</td>
</tr>
<tr>
<td>Paris</td>
<td>128</td>
<td>25.63</td>
<td>24.02</td>
<td>24.38</td>
<td>24.57</td>
<td>24.82</td>
<td>24.93</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>25.81</td>
<td>24.17</td>
<td>24.50</td>
<td>24.74</td>
<td>25.01</td>
<td>25.02</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>26.34</td>
<td>24.24</td>
<td>24.59</td>
<td>24.83</td>
<td>25.37</td>
<td>25.41</td>
</tr>
<tr>
<td>Stefan</td>
<td>128</td>
<td>24.55</td>
<td>23.06</td>
<td>23.67</td>
<td>23.69</td>
<td>23.79</td>
<td>23.80</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>24.62</td>
<td>23.12</td>
<td>23.72</td>
<td>23.75</td>
<td>23.87</td>
<td>23.89</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>24.67</td>
<td>23.15</td>
<td>23.76</td>
<td>23.80</td>
<td>23.91</td>
<td>23.93</td>
</tr>
<tr>
<td>Tennis</td>
<td>128</td>
<td>28.55</td>
<td>25.96</td>
<td>26.99</td>
<td>27.12</td>
<td>26.87</td>
<td>27.07</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>28.71</td>
<td>26.03</td>
<td>26.99</td>
<td>27.19</td>
<td>27.03</td>
<td>27.19</td>
</tr>
<tr>
<td>Tempete</td>
<td>128</td>
<td>24.95</td>
<td>23.43</td>
<td>23.72</td>
<td>23.88</td>
<td>24.00</td>
<td>24.06</td>
</tr>
<tr>
<td></td>
<td>384</td>
<td>25.02</td>
<td>23.48</td>
<td>23.68</td>
<td>23.94</td>
<td>24.05</td>
<td>24.10</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>25.08</td>
<td>23.51</td>
<td>23.79</td>
<td>23.98</td>
<td>24.10</td>
<td>24.15</td>
</tr>
<tr>
<td>City</td>
<td>1024</td>
<td>28.52</td>
<td>25.59</td>
<td>26.36</td>
<td>26.53</td>
<td>26.84</td>
<td>26.88</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>28.58</td>
<td>25.64</td>
<td>26.40</td>
<td>26.59</td>
<td>26.89</td>
<td>26.93</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>28.68</td>
<td>25.68</td>
<td>26.45</td>
<td>26.66</td>
<td>26.95</td>
<td>27.02</td>
</tr>
<tr>
<td>Crew</td>
<td>1024</td>
<td>32.48</td>
<td>28.76</td>
<td>29.86</td>
<td>28.42</td>
<td>30.00</td>
<td>30.01</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>32.56</td>
<td>28.81</td>
<td>29.91</td>
<td>30.36</td>
<td>30.05</td>
<td>30.07</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>32.63</td>
<td>28.86</td>
<td>29.99</td>
<td>30.43</td>
<td>30.10</td>
<td>30.15</td>
</tr>
<tr>
<td>Harbour</td>
<td>1024</td>
<td>27.25</td>
<td>24.86</td>
<td>25.38</td>
<td>24.99</td>
<td>25.86</td>
<td>25.88</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>27.30</td>
<td>24.90</td>
<td>25.42</td>
<td>25.79</td>
<td>25.91</td>
<td>25.92</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>27.39</td>
<td>24.95</td>
<td>25.49</td>
<td>25.86</td>
<td>25.97</td>
<td>26.00</td>
</tr>
<tr>
<td>Carphone</td>
<td>32</td>
<td>28.77</td>
<td>25.97</td>
<td>26.64</td>
<td>26.87</td>
<td>27.08</td>
<td>27.11</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>28.88</td>
<td>26.05</td>
<td>26.71</td>
<td>26.99</td>
<td>27.14</td>
<td>27.15</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>29.28</td>
<td>26.19</td>
<td>26.95</td>
<td>27.22</td>
<td>28.14</td>
<td>28.15</td>
</tr>
<tr>
<td>Grandma</td>
<td>32</td>
<td>29.73</td>
<td>26.70</td>
<td>27.87</td>
<td>28.10</td>
<td>28.13</td>
<td>28.23</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>29.95</td>
<td>26.81</td>
<td>28.03</td>
<td>28.30</td>
<td>28.33</td>
<td>28.41</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>31.89</td>
<td>27.12</td>
<td>28.48</td>
<td>29.03</td>
<td>29.49</td>
<td>30.37</td>
</tr>
</tbody>
</table>
inaccurate interval partitioning of pixel values, deteriorating the best-matched block and subsequent inter-frame prediction accuracy. Our non-uniform quantization based bit transform (NUQ-2BT) exploits optimum quantization theory and histogram equalization to reduce the information loss due to bit-resolution decrease. On one hand, it provides satisfying motion estimation performances for most of the test sequences, although it only produces a suboptimal solution. The NUQ-2BT achieves 0.87 dB, 0.31 dB, and 0.14 dB higher PSNR than those of 1BT, MT-2BT, and AQ-2BT in average. This illustrates that our non-uniform interval partitioning is an effective solution to the bit transform. On the other hand, we can find in Tables 2 and 3 that the NUQ-2BT is inefficient for complex videos with high spatial details or fast motion, e.g., “Flower” and “Football” sequences. The reason is that such videos tend to present higher inter-frame noises than those with low details or slow motion, especially after a lossy coding. In this case the NUQ-2BT method would be seriously affected, resulting in inaccurate motion estimation as discussed in Section 4. To address this issue, our FQ-2BT method uses a membership function working in a fuzzy manner to effectively alleviate the negative impact of encoder’s quantization noises. Additionally, it uses uniform threshold for the current and reference frames. This avoids both the discontinuity at macroblock’s boundary occurred in MT-2BT and repeatedly calculating quantization thresholds in AQ-2BT for each search window. Therefore, the proposed bit transform and motion estimation obtain superior coding performance to several representative methods, whose PSNR is 0.24 dB, 0.42 dB, 0.98 dB, and 0.10 dB higher than those of AQ-2BT, MT-2BT, 1BT, and NUQ-2BT, respectively.

6.3. Computational complexity comparison

The computational complexity is one of the key factors for evaluating motion estimation efficiency. In this subsection, we will analyze the computational complexity of each algorithm in terms of two stages, namely bit transform stage and motion estimation stage. Table 4 lists average operation numbers per pixel by various motion estimation algorithms for CIF format videos with a search range of ±16 pixels and a block size of 16 × 16 pixels. As can be found in Table 4, our algorithm performs less operations than 1BT, MT-2BT, and AQ-2BT at bit transform stage. While at the motion estimation stage, the proposed FQ-2BT algorithm still requires lower computational cost than three low bit-resolution methods, since the bit-truncation SAD metric h is able to be implemented by the XLAT instruction of assembly language. The complexity of each kind of operation depends on device’s hardware implementation and instruction set, which is beyond the scope of our study. We will not discuss the quantitative comparisons among several algorithms. In general, the computational cost of FQ-2BT is lower than 1BT, MT-2BT.
and AQ-2BT, and obviously cheaper than FS with eight-bit depth pixels.

7. Conclusion

This study addresses a novel bit transform for low bit-resolution motion estimation by exploiting the quantization theory and fuzzy membership function. We formulate the bit-depth downsampling of eight bit-depth pixels into optimum quantization in terms of mean squared error. Subsequently, we present an approximate solution using histogram equalization and uniform quantization, which is refined by a membership function with the variance of inter-frame noises as a variable. The membership function is able to reduce the interval partitioning errors due to camera capability and coding distortions, so as to improve bit transform accuracy and motion estimation efficiency. Extensive experimental results verify the effectiveness of our bit transform and its application in low bit-resolution motion estimation.

Note that we employ full search strategy in Algorithm 3 to eliminate the influences by different search strategies, thus making fair comparisons among several bit transform algorithms. In fact, the proposed bit transform method can be both combined with fast motion estimation and applied to complexity scalable motion estimation as an initial search.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (NSFC) under Grant nos. 41271422, 61073098, and 61373059, the Scientific Research Foundation for Ph.D. of Liaoning Province of China under Grant no. 20121076, and the Open Foundation of National Key Laboratory for Novel Software Technology of Nanjing University under Grant nos. KFKT2011B09 and KFKT2010B11, and the Jiangsu Key Laboratory's Open Foundation of Image Processing and Image Communication of Nanjing University of Posts & Telecommunications under Grant no. LBEK2011001, and the Foundation of Science and Technology Plan for Higher Education of Liaoning Province under Grant nos. L2011192 and L2013406, and the Dalian Foundation for Science and Technology under Grant no. 2012J21DW008.

References


