DCT-Based Motion Estimation

Ut-Va Koc and *K. J. Ray Liu*

Electrical Engineering Department and Institute for Systems Research
University of Maryland at College Park
College Park, Maryland 20742
koc@eng.umd.edu and kjrliu@eng.umd.edu

ABSTRACT

A new motion estimation approach, the DCT-Based Motion Estimation Scheme (DXT-ME) utilizing the sinusoidal orthogonal principles to estimate displacements of moving objects in the transform domain, based upon the concept of *pseudo phases*, is presented in this paper. The computational complexity of this method is only $O(N^2)$ for an $N \times N$ block in comparison to the $O(N^4)$ complexity of Full Search Block Matching Approach (BMA-ME). In addition, the DXT-ME algorithm has solely highly parallel local operations and this property makes parallel implementation feasible. Furthermore, incorporation of DXT-ME with a video coder using DCT can combine the DCT and motion estimation algorithm to achieve further saving in overall system complexity and increase the system throughput. Unlike the pel-recursive algorithm, this scheme is robust for even very noisy images. Due to its feature matching property, we can employ simple preprocessing on images of complicated scenery to extract the features of moving objects for DXT-ME to further improve its performance. Finally simulation on a number of video sequences is presented to compare DXT-ME with BMA-ME.

Keywords: Motion estimation, video coding, video compression

This work is supported in part by the ONR grant N00014-93-1-0566, NSF NYI Award MIP9457397, and MIPS/MicroStar.

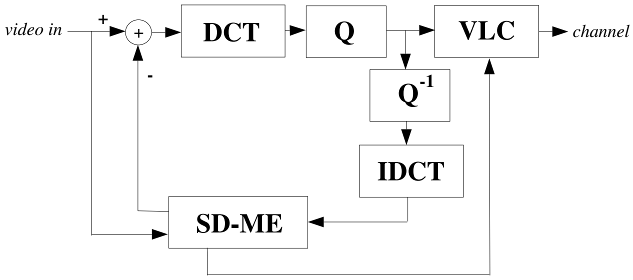
I. Introduction

In recent years, great interests have been found in motion estimation due to its various promising applications [1] in high definition television (HDTV), multimedia, video telephony, target detection and tracking , and computer vision, et al. Extensive research has been done over many years in developing new algorithms [1], [2] and designing cost-effective and massively parallel hardware architectures [3], [4], [5], [6] suitable for current VLSI technology.

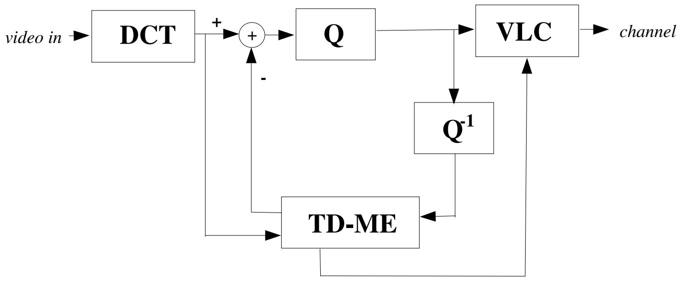
The most commonly used motion estimation scheme in video coding is the Full Search Block Matching Algorithm (BMA-ME) which searches for the best candidate block among all the blocks in a search area of larger size in terms of either the mean-square error [7] or the mean of the absolute frame difference [8]. The computational complexity of this approach is very high, i.e. $O(N^4)$ for a $N \times N$ block. Even so, BMA-ME has been successfully implemented on VLSI chips [3], [4], [5]. To reduce the number of computations, a number of suboptimal fast block matching algorithms have been proposed [7], [8], [9], [10], [11], [12] . However, these algorithms require three or more sequential steps to find suboptimal estimates. Recently a correlation-based approach (CLT-ME) [13] using Complex Lapped Transform (CLT) to avoid the block effect was proposed but it still requires searching over a larger search area and thus results in a very high computational burden. Moreover, motion estimation using the CLT-ME is accurate on moving sharp edges but not on blur edges.

In addition to block-based approaches, pel-based estimation methods such as Pel-Recursive Algorithm (PRA-ME) [14], [15] and Optical Flow Approach (OFA-ME) [16] , are very vulnerable to noise by virtue of their involving only local operations and may suffer from the instability problem. For multiframe motion detection, 3D-FFT has been successfully used to estimate motion in several consecutive frames [17], [18], based on the phenomenon that the spatial and temporal frequencies of a moving object lie on a plane of spatiotemporal space [19]. However, this requires processing of several frames rather than two, and the fast Fourier transform operates on complex numbers and is not used in most video standards.

In most international video coding standards such as CCITT H.261 [20] , MPEG [21] as well as the proposed HDTV standard, Discrete Cosine Transform (DCT) and block-based motion estimation are the essential elements to achieve spatial and temporal compression, respectively. Most implementations of a standard-compliant coder adopt the structure of Coder III (originally named in [22]) as shown in Fig. 1(a). The DCT is located inside the loop of temporal prediction, which also includes an Inverse DCT (IDCT) and a spatial-domain motion estimator (SD-ME) which is usually the BMA-ME. The IDCT is needed solely for transforming the DCT coefficients back to the spatial domain in which the SD-ME



(a) Coder III



(b) Coder II

Fig. 1. Coder structures: (a) Coder III is the motion-compensated DCT hybrid coder used in MPEG or H.261 standards with motion estimation done in the spatial domain. (b) Coder II is the motion-compensated DCT hybrid coder with motion estimation performed in the transform domain.

estimates motion vectors and performs motion compensated prediction. This is an undesirable coder architecture for the following reasons. In addition to the additional complexity added to the overall architecture, the DCT and IDCT must be put inside the feedback loop which has long been recognized as the major bottleneck of the entire digital video system for high-end real-time applications. The throughput of the coder is limited by the processing speed of the feedback loop, which is roughly the total time for the data stream to go through each component in the loop. Therefore the DCT (or IDCT) must be designed to operate at least twice as fast as the incoming data stream. A compromise is to remove the loop and perform open-loop motion estimation based upon original images instead of recontructed images in sacrifice of the performance of the coder [23].

An alternative solution without degradation of the performance is to develop a motion estimation algorithm which can work in the DCT transform domain as remarked in [22]. In this way, the DCT can be moved out of the loop as depicted in Fig. 1(b) and thus the operating speed of this DCT can be reduced to the data rate of the incoming stream. Moreover, the IDCT is removed from the feedback loop which now has only two simple components Q and Q^{-1} (the quantizers) in addition to the transform-domain motion estimator (TD-ME). This not only reduces the complexity of the coder but also resolve the bottleneck problem without any tradeoff of the performance. Furthermore, as pointed out in [22], different components can be jointly optimized if they operate in the same transform domain.

In this paper, we present a novel algorithm for motion estimation, called the DCT-Based Motion Estimation (DXT-ME), to estimate motion in the discrete cosine transform or discrete sine transform (DCT/DST or DXT for short) domain. DXT-ME is based on the principle of orthogonality of sinusoidal functions. This new algorithm has certain merits over conventional methods. It has very low computational complexity (on the order of N^2 compared to N^4 for BMA-ME) and is robust even in a noisy environment. This algorithm takes DCT coefficients of images as input to estimate motions and therefore can be incorporated efficiently with the DCT-based coders used for most current video compression standards. As explained before, this combination of both the DCT and motion estimation into a single component reduces the coder complexity and at the same time increases the system throughput. Finally, due to the fact that the computation of pseudo phases is inherently highly local operation, a highly parallel pipelined architecture for this algorithm is possible.

In the next section, the principles behind this motion estimation scheme are presented. In Section III, the algorithm of DXT-ME is then considered. For video sequences of complicated scenery, some preprocessing is necessary to further improve the performance of this estimator. This preprocessing step is discussed in Section IV along with a simple extension of DXT-ME similar to the decision rule used in MPEG standards. Simulation results are given and discussed in Section V. Finally, we conclude the paper in Section VI.

II. Sinusoidal Orthogonal Principles

As well known, Fourier transform (FT) of a signal, $x(t)$ is related to FT of its shifted (or delayed if t represents time) version, $x(t-\tau)$, by this equation:

$$\mathcal{F}\{x(t-\tau)\} = e^{-j\omega\tau}\mathcal{F}\{x(t)\}, \tag{1}$$

where $\mathcal{F}\{\cdot\}$ denotes Fourier transform. The phase of Fourier transform of the shifted signal contains the information about the amount of the shift τ, which can easily be extracted. However, Discrete Cosine Transform (DCT) or its counterpart, Discrete Sine Transform (DST), do not have any phase components as usually found in discrete Fourier transform (DFT), but DCT (or DST) coefficients of a shifted signal do also carry this shift information. To facilitate explanation of the idea behind DXT-ME, let us first consider the case of one-dimensional discrete signals. Suppose that the signal $\{x_1(n);\ n \in \{0,\dots,N-1\}\}$ is right shifted by an amount m (in our convention, a right shift means that $m > 0$) to generate another signal $\{x_2(n);\ n \in \{0,\dots,N-1\}\}$. The values of $x_1(n)$ are all zeros outside the support region $\mathcal{S}(x_1)$. Therefore,

$$x_2(n) = \begin{cases} x_1(n-m), & \text{for } n-m \in \mathcal{S}(x_1), \\ 0, & \text{elsewhere.} \end{cases} \tag{2}$$

(2) implies that both signals have resemblance to each other except that the signal is shifted. It can be shown that, for $k = 1, \ldots, N-1$,

$$X_2^C(k) = Z_1^C(k)\cos[\frac{k\pi}{N}(m+\frac{1}{2})] - Z_1^S(k)\sin[\frac{k\pi}{N}(m+\frac{1}{2})], \tag{3}$$

$$X_2^S(k) = Z_1^S(k)\cos[\frac{k\pi}{N}(m+\frac{1}{2})] + Z_1^C(k)\sin[\frac{k\pi}{N}(m+\frac{1}{2})]. \tag{4}$$

Here X_2^S and X_2^C are DST (DST-II) and DCT (DCT-II) of the second kind of $x_2(n)$, respectively, whereas Z_1^S and Z_1^C are DST (DST-I) and DCT (DCT-I) of the first kind of $x_1(n)$, respectively, as defined as follows [24] :

$$X_2^C(k) = \frac{2}{N}C(k)\sum_{n=0}^{N-1} x_2(n)\cos[\frac{k\pi}{N}(n+0.5)];\ k \in \{0, \ldots, N-1\}, \tag{5}$$

$$X_2^S(k) = \frac{2}{N}C(k)\sum_{n=0}^{N-1} x_2(n)\sin[\frac{k\pi}{N}(n+0.5)];\ k \in \{1, \ldots, N\}, \tag{6}$$

$$Z_1^C(k) = \frac{2}{N}C(k)\sum_{n=0}^{N-1} x_1(n)\cos[\frac{k\pi}{N}(n)];\ k \in \{0, \ldots, N\}, \tag{7}$$

$$Z_1^S(k) = \frac{2}{N}C(k)\sum_{n=0}^{N-1} x_1(n)\sin[\frac{k\pi}{N}(n)];\ k \in \{1, \ldots, N-1\}, \tag{8}$$

$$\text{where } C(k) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{for } k = 0 \text{ or } N, \\ 1, & \text{otherwise,} \end{cases}$$

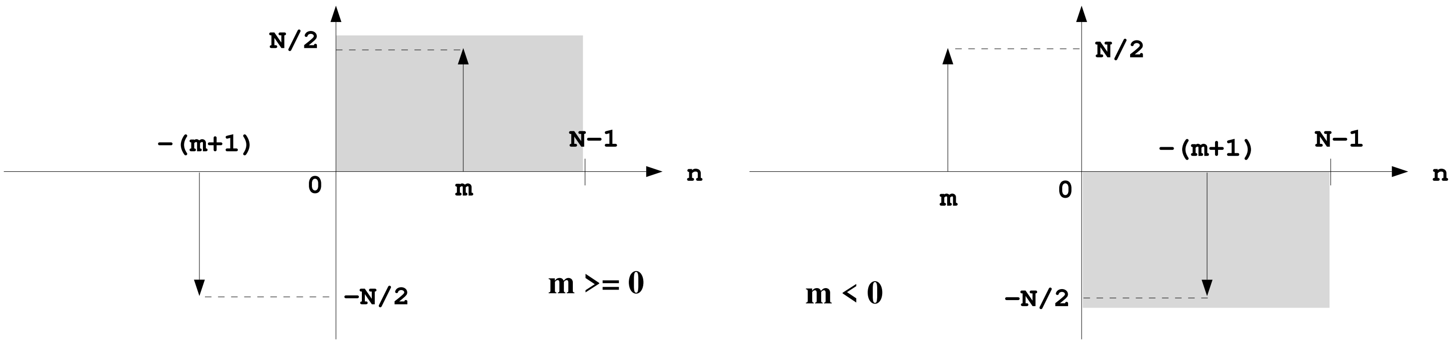
The displacement, m, is embedded solely in the terms $g_m^s(k) = \sin[\frac{k\pi}{N}(m+\frac{1}{2})]$ and $g_m^c(k) = \cos[\frac{k\pi}{N}(m+\frac{1}{2})]$, which are called *pseudo phases* analogous to phases in Fourier transform of shifted signals. To find out m, we first solve (3) and (4) for the pseudo phases and then use the sinusoidal orthogonal principles as follows:

$$\frac{2}{N}\sum_{k=1}^{N} C^2(k)\sin[\frac{k\pi}{N}(m+\frac{1}{2})]\sin[\frac{k\pi}{N}(n+\frac{1}{2})] = \delta(m-n) - \delta(m+n+1), \tag{9}$$

$$\frac{2}{N}\sum_{k=0}^{N-1} C^2(k)\cos[\frac{k\pi}{N}(m+\frac{1}{2})]\cos[\frac{k\pi}{N}(n+\frac{1}{2})] = \delta(m-n) + \delta(m+n+1), \tag{10}$$

Here $\delta(n)$ is the discrete impulse function.

Indeed, if we replace $\sin[\frac{k\pi}{N}(m+\frac{1}{2})]$ and $\cos[\frac{k\pi}{N}(m+\frac{1}{2})]$ by the computed sine and cosine pseudo phase components, $\hat{g}_m^s(k)$ and $\hat{g}_m^c(k)$, respectively in (9) and (10), both equations simply become IDST-II and



(a) How to detect right shift (b) How to detect left shift

Fig. 2. How the direction of motion is determined based on the sign of the peak value after application of the sinusoidal orthogonal principle for the DST-II kernel to pseudo phases

IDCT-II operations on $\hat{g}_m^s(k)$ and $\hat{g}_m^c(k)$:

$$IDSTII(\hat{g}_m^s) = \frac{2}{N}\sum_{k=1}^{N} C^2(k)\hat{g}_m^s(k)\sin[\frac{k\pi}{N}(n+\frac{1}{2})], \quad (11)$$

$$IDCTII(\hat{g}_m^c) = \frac{2}{N}\sum_{k=0}^{N-1} C^2(k)\hat{g}_m^c(k)\cos[\frac{k\pi}{N}(n+\frac{1}{2})]. \quad (12)$$

The notation $\hat{g}$ is used to distinguish the computed pseudo phase from the one in a noiseless situation (i.e. $\sin[\frac{k\pi}{N}(m+\frac{1}{2})]$ or $\cos[\frac{k\pi}{N}(m+\frac{1}{2})]$). A closer look at the right-hand side of (9) tells us that $\delta(m-n)$ and $\delta(m+n+1)$ have opposite signs. This property will help us detect the directions of a motion. If we perform an IDST-II operation on the pseudo phases found, then the observable window of the index space in the inverse DST domain will be limited to $\{0,\ldots,N-1\}$. As illustrated in Fig. 2, for a right-shift motion, one spike (generated by the positive δ function) is pointing upwards at the location $n=m$ in the gray region (i.e. the observable index space), while the other δ pointing downwards at $n=-(m+1)$ outside the gray region. In contrary, for a left-shift motion, the negative spike at $n=-(m+1)>0$ falls in the gray region but the positive δ function at $n=m$ stays out of the observable index space. It can easily be seen that a positive peak value in the gray region implies a right shift and a negative one means a left shift. This enables us to determine from the sign of the peak value the direction of the movement of a signal.

The concept of pseudo phases plus the application of the sinusoidal orthogonal principles leads to a new approach to estimate the translational motion as depicted in Fig. 3 (a):

1. Compute the DCT-I and DST-I coefficients of $x_1(n)$ and the DCT-II and DST-II coefficients of $x_2(n)$.

2. Compute the pseudo phase $\hat{g}_m^s(k)$ for $k = 1, \ldots, N$ by solving this equation:

$$\hat{g}_m^s(k) = \begin{cases} \frac{Z_1^C(k)\cdot X_2^S(k) - Z_1^S(k)\cdot X_2^C(k)}{[Z_1^C(k)]^2 + [Z_1^S(k)]^2}, & \text{for } k \neq N, \\ \frac{1}{\sqrt{2}}, & \text{for } k = N. \end{cases} \tag{13}$$

3. Feed the computed pseudo phase, $\{\hat{g}_m^s(k);\ k = 1, \ldots, N\}$, into an IDST-II decoder to produce an output $\{d(n);\ n = 0, \ldots, N-1\}$, and search for the peak value. Then the estimated displacement $\hat{m}$ can be found by

$$\hat{m} = \begin{cases} i_p, & \text{if } d(i_p) > 0, \\ -(i_p + 1), & \text{if } d(i_p) < 0, \end{cases} \tag{14}$$

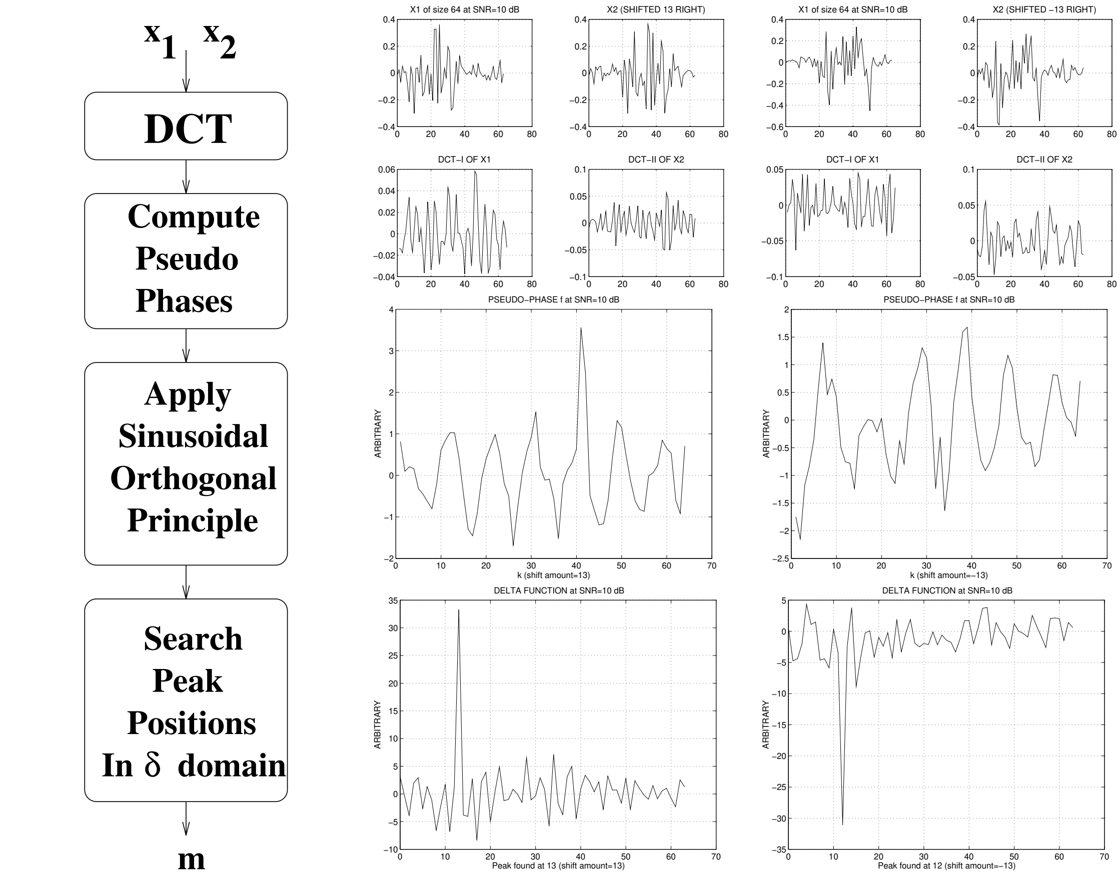
where $i_p = arg\max_n |d(n)|$ is the index at which the peak value is located.

In Step 1, the DCT and DST can be generated simultaneously with only $3N$ multipliers [25], [26], [27], and the computation of DCT-I can be easily obtained from DCT-II with minimal overhead as will be shown later. In Step 2, if noise is absent and there is only purely translational motion, $\hat{g}_m(k)$ will be equal to $\sin \frac{k\pi}{N}(m + 0.5)$. The output $d(n)$ will then be an impulse function in the observation window. This procedure is illustrated by two examples in Fig. 3(b) and (c) with a randomly generated signal as input at $SNR = 20$ dB. From these two examples, it is obvious that the motion estimate is accurate even though strong noise is present.

III. DCT-Based Motion Estimation Algorithm (DXT-ME)

The concept of how to extract shift values from the pseudo phases of one dimensional signals, as explained in Section II, can be extended to provide the basis of a new DCT-based motion estimation scheme for two dimensional images, which we call DXT-ME Motion Estimation Scheme. Before going into details of this new algorithm, let us confine the problem of motion estimation to the scenario in which an object moves translationally by m_1 in X direction and n_1 in Y direction as viewed on the camera plane and within the scope of a camera in a noiseless environment as shown in Fig. 4. Then we can extract the motion vector out of the two consecutive frames of the images of that moving object by making use of the sinusoidal orthogonal principles (9) and (10).

The algorithm of this DCT-based motion estimation scheme is depicted in Fig. 5. The previous frame x_{t-1} and the current frame x_t are fed into 2D-DCT-II and 2D-DCT-I coders respectively. An 2D-DCT-II coder computes four coefficients, DCCTII, DCSTII, DSCTII, and DSSTII, each of which is defined



(a) 1-D Motion Estimation (b) Shift Right (c) Shift Left

Fig. 3. Illustration of the concept of DCT-based motion estimation in one dimensional case

as a two-dimensional separable function formed by 1D-DCT/DST-II kernels:

$$X_t^{cc}(k,l) = \frac{4}{N^2}C(k)C(l)\sum_{m,n=0}^{N-1} x_t(m,n)\cos[\frac{k\pi}{N}(m+0.5)]\cos[\frac{l\pi}{N}(n+0.5)], \quad k,l \in \{0,\ldots,N-1\}, \tag{15}$$

$$X_t^{cs}(k,l) = \frac{4}{N^2}C(k)C(l)\sum_{m,n=0}^{N-1} x_t(m,n)\cos[\frac{k\pi}{N}(m+0.5)]\sin[\frac{l\pi}{N}(n+0.5)], \quad k \in \{0,\ldots,N-1\}, l \in \{1,\ldots,N\}, \tag{16}$$

$$X_t^{sc}(k,l) = \frac{4}{N^2}C(k)C(l)\sum_{m,n=0}^{N-1} x_t(m,n)\sin[\frac{k\pi}{N}(m+0.5)]\cos[\frac{l\pi}{N}(n+0.5)], \quad k \in \{1,\ldots,N\}, l \in \{0,\ldots,N-1\}, \tag{17}$$

$$X_t^{ss}(k,l) = \frac{4}{N^2}C(k)C(l)\sum_{m,n=0}^{N-1} x_t(m,n)\sin[\frac{k\pi}{N}(m+0.5)]\sin[\frac{l\pi}{N}(n+0.5)], \quad k,l \in \{1,\ldots,N\}, \tag{18}$$

or symbolically,

$$X_t^{cc} = DCCTII(x_t),\ X_t^{cs} = DCSTII(x_t),\ X_t^{sc} = DSCTII(x_t),\ X_t^{ss} = DSSTII(x_t).$$

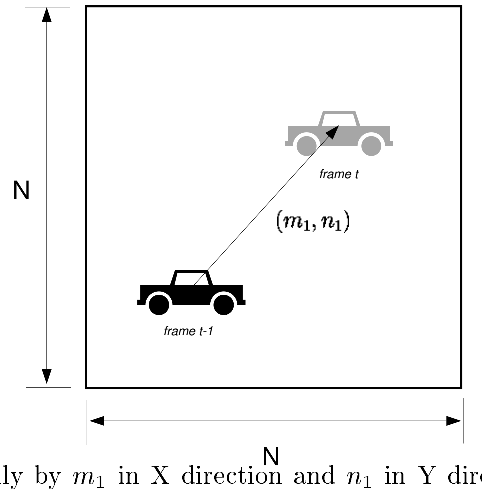
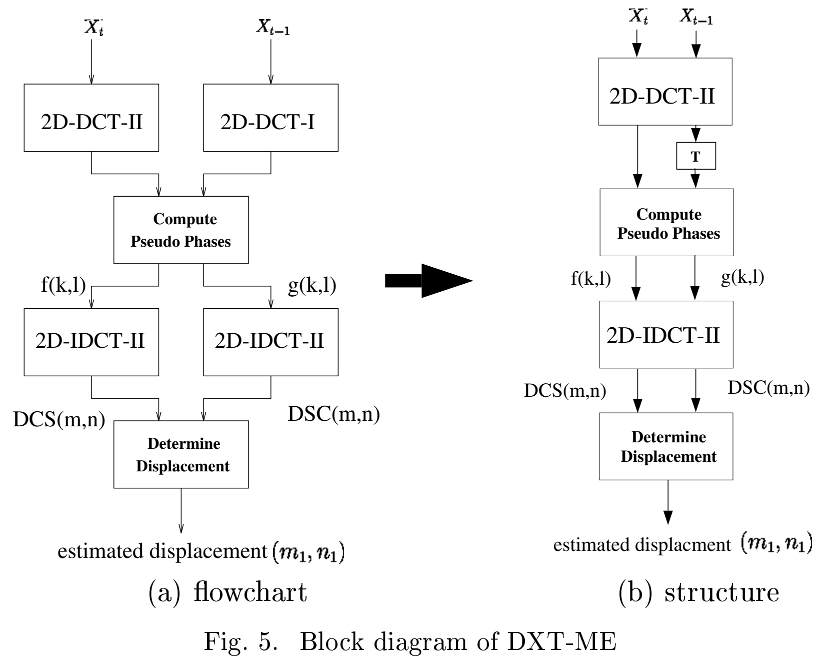


Fig. 4. An object moves translationally by m_1 in X direction and n_1 in Y direction as viewed on the camera plane.

In the same fashion, the two dimensional DCT coefficients of the first kind (2D-DCT-I) are calculated



(a) flowchart (b) structure

Fig. 5. Block diagram of DXT-ME

based on 1D-DCT/DST-I kernels:

$$Z_{t-1}^{cc}(k,l) = \frac{4}{N^2}C(k)C(l)\sum_{m,n=0}^{N-1} x_{t-1}(m,n)\cos[\frac{k\pi}{N}(m)]\cos[\frac{l\pi}{N}(n)], \quad k,l \in \{0,\ldots,N\}, \tag{19}$$

$$Z_{t-1}^{cs}(k,l) = \frac{4}{N^2}C(k)C(l)\sum_{m,n=0}^{N-1} x_{t-1}(m,n)\cos[\frac{k\pi}{N}(m)]\sin[\frac{l\pi}{N}(n)], \quad k \in \{0,\ldots,N\}, l \in \{1,\ldots,N-1\}, \tag{20}$$

$$Z_{t-1}^{sc}(k,l) = \frac{4}{N^2}C(k)C(l)\sum_{m,n=0}^{N-1} x_{t-1}(m,n)\sin[\frac{k\pi}{N}(m)]\cos[\frac{l\pi}{N}(n)], \quad k \in \{1,\ldots,N-1\}, l \in \{0,\ldots,N\}, \tag{21}$$

$$Z_{t-1}^{ss}(k,l) = \frac{4}{N^2}C(k)C(l)\sum_{m,n=0}^{N-1} x_{t-1}(m,n)\sin[\frac{k\pi}{N}(m)]\sin[\frac{l\pi}{N}(n)], \quad k,l \in \{1,\ldots,N-1\}, \tag{22}$$

or symbolically,

$$Z_{t-1}^{cc} = DCCTI(x_{t-1}),\ Z_{t-1}^{cs} = DCSTI(x_{t-1}),\ Z_{t-1}^{sc} = DSCTI(x_{t-1}),\ Z_{t-1}^{ss} = DSSTI(x_{t-1}).$$

Similar to one dimensional case, assuming that only translational motion is allowed, one can derive a set of equations to relate DCT coefficients of $x_{t-1}(m,n)$ with those of $x_t(m,n)$ in the same way as in (3) and (4).

$$\mathbf{Z}_{t-1}(k,l)\cdot\vec{\theta}(k,l) = \vec{\mathbf{x}}_t(k,l),\ \text{for } k,l \in \mathcal{N}, \tag{23}$$

where $\mathcal{N} = \{1,\ldots,N-1\}$,

$$\mathbf{Z}_{t-1}(k,l) = \begin{bmatrix} Z_{t-1}^{cc}(k,l) & -Z_{t-1}^{cs}(k,l) & -Z_{t-1}^{sc}(k,l) & Z_{t-1}^{ss}(k,l) \\ Z_{t-1}^{cs}(k,l) & Z_{t-1}^{cc}(k,l) & -Z_{t-1}^{ss}(k,l) & -Z_{t-1}^{sc}(k,l) \\ Z_{t-1}^{sc}(k,l) & -Z_{t-1}^{ss}(k,l) & Z_{t-1}^{cc}(k,l) & -Z_{t-1}^{cs}(k,l) \\ Z_{t-1}^{ss}(k,l) & Z_{t-1}^{sc}(k,l) & Z_{t-1}^{cs}(k,l) & Z_{t-1}^{cc}(k,l) \end{bmatrix}, \tag{24}$$

$$\vec{\theta}(k,l) = \begin{bmatrix} g_{m_1n_1}^{CC}(k,l) \\ g_{m_1n_1}^{CS}(k,l) \\ g_{m_1n_1}^{SC}(k,l) \\ g_{m_1n_1}^{SS}(k,l) \end{bmatrix} = \begin{bmatrix} \cos\frac{k\pi}{N}(m_1+0.5)\cos\frac{l\pi}{N}(n_1+0.5) \\ \cos\frac{k\pi}{N}(m_1+0.5)\sin\frac{l\pi}{N}(n_1+0.5) \\ \sin\frac{k\pi}{N}(m_1+0.5)\cos\frac{l\pi}{N}(n_1+0.5) \\ \sin\frac{k\pi}{N}(m_1+0.5)\sin\frac{l\pi}{N}(n_1+0.5) \end{bmatrix}, \tag{25}$$

$$\vec{\mathbf{x}}_t(k,l) = \begin{bmatrix} X_t^{cc}(k,l) & X_t^{cs}(k,l) & X_t^{sc}(k,l) & X_t^{ss}(k,l) \end{bmatrix}^T. \tag{26}$$

Here $\mathbf{Z}_{t-1}(k,l) \in R^{4\times4}$ is the *system matrix* of DXT-ME at (k,l). At the boundaries of each block in the transform domain, the DCT coefficients of $x_{t-1}(m,n)$ and $x_t(m,n)$ have one dimensional relationship as given below:

$$\begin{bmatrix} Z_{t-1}^{cc}(k,l) & -Z_{t-1}^{cs}(k,l) \\ Z_{t-1}^{cs}(k,l) & Z_{t-1}^{cc}(k,l) \end{bmatrix}\begin{bmatrix} \cos\frac{l\pi}{N}(n_1+0.5) \\ \sin\frac{l\pi}{N}(n_1+0.5) \end{bmatrix} = \begin{bmatrix} X_t^{cc}(k,l) \\ X_t^{cs}(k,l) \end{bmatrix}, \quad k=0,\ l\in\mathcal{N}, \tag{27}$$

$$\begin{bmatrix} Z_{t-1}^{cc}(k,l) & -Z_{t-1}^{sc}(k,l) \\ Z_{t-1}^{sc}(k,l) & Z_{t-1}^{cc}(k,l) \end{bmatrix}\begin{bmatrix} \cos\frac{k\pi}{N}(m_1+0.5) \\ \sin\frac{k\pi}{N}(m_1+0.5) \end{bmatrix} = \begin{bmatrix} X_t^{cc}(k,l) \\ X_t^{sc}(k,l) \end{bmatrix}, \quad l=0,\ k\in\mathcal{N}, \tag{28}$$

$$\begin{bmatrix} Z_{t-1}^{cc}(k,l) & -Z_{t-1}^{cs}(k,l) \\ Z_{t-1}^{cs}(k,l) & Z_{t-1}^{cc}(k,l) \end{bmatrix} \begin{bmatrix} \cos\frac{l\pi}{N}(n_1+0.5) \\ \sin\frac{l\pi}{N}(n_1+0.5) \end{bmatrix} = (-1)^{m_1} \begin{bmatrix} X_t^{sc}(k,l) \\ X_t^{ss}(k,l) \end{bmatrix}, \quad k=N,\, l\in\mathcal{N}, \tag{29}$$

$$\begin{bmatrix} Z_{t-1}^{cc}(k,l) & -Z_{t-1}^{sc}(k,l) \\ Z_{t-1}^{sc}(k,l) & Z_{t-1}^{cc}(k,l) \end{bmatrix} \begin{bmatrix} \cos\frac{k\pi}{N}(m_1+0.5) \\ \sin\frac{k\pi}{N}(m_1+0.5) \end{bmatrix} = (-1)^{n_1} \begin{bmatrix} X_t^{cs}(k,l) \\ X_t^{ss}(k,l) \end{bmatrix}, \quad l=N,\, k\in\mathcal{N}, \tag{30}$$

$$(-1)^{n_1} Z_{t-1}^{cc}(k,l) = X_t^{cs}(k,l), \quad k=0,\, l=N, \tag{31}$$

$$(-1)^{m_1} Z_{t-1}^{cc}(k,l) = X_t^{sc}(k,l), \quad k=N,\, l=0. \tag{32}$$

In a two dimensional space, an object may move in four possible directions: northeast (NE: $m_1 > 0$, $n_1 > 0$), northwest (NW: $m_1 < 0$, $n_1 > 0$), southeast (SE: $m_1 > 0$, $n_1 < 0$), and southwest (NW: $m_1 < 0$, $n_1 < 0$). As explained in Section II, the orthogonal equation for the DST-II kernel in (9) can be applied to the pseudo phase $\hat{g}_m^s(k)$ to determine the sign of m (i.e. the direction of the shift). In order to detect the signs of both m_1 and n_1 (or equivalently the direction of motion), it becomes obvious from the observation in the one dimensional case that it is necessary to compute the pseudo phases $\hat{g}_{m_1n_1}^{SC}(\cdot,\cdot)$ and $\hat{g}_{m_1n_1}^{CS}(\cdot,\cdot)$ so that the signs of m_1 and n_1 can be determined from $\hat{g}_{m_1n_1}^{SC}(\cdot,\cdot)$ and $\hat{g}_{m_1n_1}^{CS}(\cdot,\cdot)$, respectively. By taking the block boundary equations (27)–(32) into consideration, we define two pseudo phase functions as follows:

$$f_{m_1n_1}(k,l) = \begin{cases} \hat{g}_{m_1n_1}^{CS}(k,l), & \text{for } k,l\in\{1,\ldots,N-1\}, \\ \frac{1}{\sqrt{2}}\frac{Z_{t-1}^{cc}(k,l)X_t^{cs}(k,l)-Z_{t-1}^{cs}(k,l)X_t^{cc}(k,l)}{(Z_{t-1}^{cc}(k,l))^2+(Z_{t-1}^{cs}(k,l))^2}, & \text{for } k=0, l\in\{1,\ldots,N-1\}, \\ \frac{1}{\sqrt{2}}\frac{Z_{t-1}^{cc}(k,l)X_t^{cs}(k,l)+Z_{t-1}^{sc}(k,l)X_t^{ss}(k,l)}{(Z_{t-1}^{cc}(k,l))^2+(Z_{t-1}^{sc}(k,l))^2}, & \text{for } l=N, k\in\{1,\ldots,N-1\}, \\ \frac{1}{2}\frac{X_t^{cs}(k,l)}{Z_{t-1}^{cc}(k,l)}, & \text{for } k=0, l=N, \text{ and } Z_{t-1}^{cc}(k,l)\neq 0, \\ \frac{1}{2}, & \text{for } k=0, l=N, \text{ and } Z_{t-1}^{cc}(k,l)=0; \end{cases} \tag{33}$$

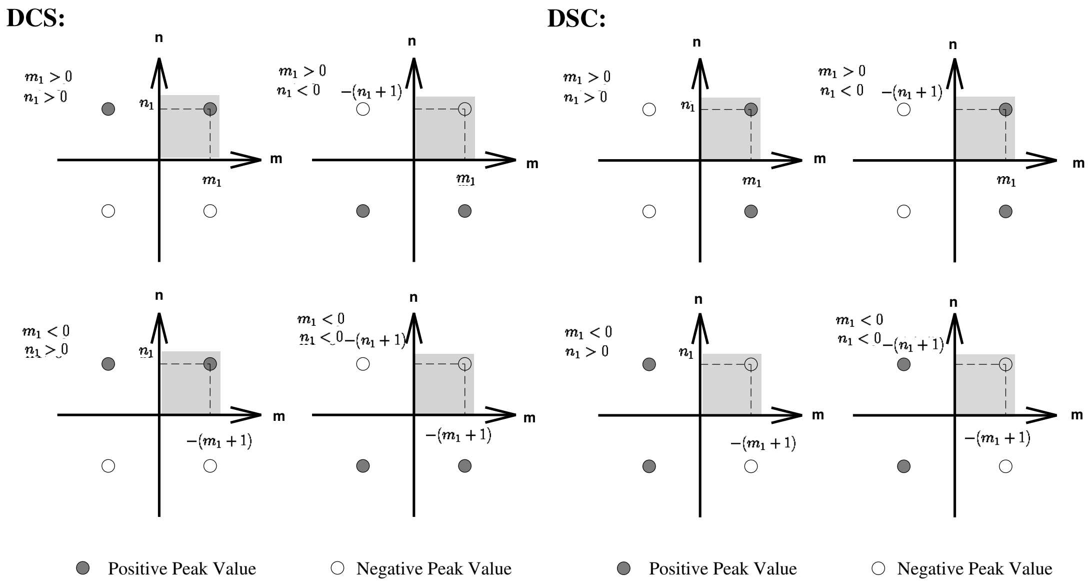
$$g_{m_1n_1}(k,l) = \begin{cases} \hat{g}_{m_1n_1}^{SC}(k,l), & \text{for } k,l\in\{1,\ldots,N-1\}, \\ \frac{1}{\sqrt{2}}\frac{Z_{t-1}^{cc}(k,l)X_t^{sc}(k,l)-Z_{t-1}^{sc}(k,l)X_t^{cc}(k,l)}{(Z_{t-1}^{cc}(k,l))^2+(Z_{t-1}^{sc}(k,l))^2}, & \text{for } l=0, k\in\{1,\ldots,N-1\}, \\ \frac{1}{\sqrt{2}}\frac{Z_{t-1}^{cc}(k,l)X_t^{sc}(k,l)+Z_{t-1}^{cs}(k,l)X_t^{ss}(k,l)}{(Z_{t-1}^{cc}(k,l))^2+(Z_{t-1}^{cs}(k,l))^2}, & \text{for } k=N, l\in\{1,\ldots,N-1\}, \\ \frac{1}{2}\frac{X_t^{sc}(k,l)}{Z_{t-1}^{cc}(k,l)}, & \text{for } k=N, l=0, \text{ and } Z_{t-1}^{cc}(k,l)\neq 0, \\ \frac{1}{2}, & \text{for } k=N, l=0, \text{ and } Z_{t-1}^{cc}(k,l)=0. \end{cases} \tag{34}$$

These two pseudo phase functions pass through 2D-IDCT-II coders ($IDCSTII$ and $IDSCTII$) to generate two functions, $DCS(\cdot,\cdot)$ and $DSC(\cdot,\cdot)$ in view of the orthogonal property of DCT-II and DST-II in (9) and (10):

$$DCS(m,n) \;=\; IDCSTII(f_{m_1n_1})$$

$$
\begin{aligned}
&= \frac{4}{N^2}\sum_{k=0}^{N-1}\sum_{l=1}^{N} C(k)C(l)f_{m_1n_1}(k,l)\cos\frac{k\pi}{N}(m+\frac{1}{2})\sin\frac{l\pi}{N}(n+\frac{1}{2}) \\
&= [\delta(m-m_1)+\delta(m+m_1+1)]\cdot[\delta(n-n_1)-\delta(n+n_1+1)], \qquad (35)
\end{aligned}
$$

$$
\begin{aligned}
DSC(m,n) &= IDSCTII(g_{m_1n_1}) \\
&= \frac{4}{N^2}\sum_{k=1}^{N}\sum_{l=0}^{N-1} C(k)C(l)g_{m_1n_1}(k,l)\sin\frac{k\pi}{N}(m+\frac{1}{2})\cos\frac{l\pi}{N}(n+\frac{1}{2}) \\
&= [\delta(m-m_1)-\delta(m+m_1+1)]\cdot[\delta(n-n_1)+\delta(n+n_1+1)]. \qquad (36)
\end{aligned}
$$



(a) DCS (b) DSC

Fig. 6. How the direction of motion is determined based on the sign of the peak value

By the same argument as in one dimensional case, the 2D-IDCT-II coders limit the observable index space $\{(i,j) : i,j = 0,\ldots,N-1\}$ of DCS and DSC to the first quadrant of the entire index space shown as gray regions in Fig. 6 which depicts (35) and (36). Similar to one dimensional case, if m_1 is positive, the observable peak value of $DSC(m,n)$ will be positive regardless of the sign of n_1 since $DSC(m,n) = \delta(m-m_1)\cdot[\delta(n-n_1)+\delta(n+n_1+1)]$ in the observable index space. Likewise, if m_1 is negative, the observable peak value of $DSC(m,n)$ will be negative because $DSC(m,n) = \delta(m+m_1+1)\cdot[\delta(n-n_1)+\delta(n+n_1+1)]$ in the gray region. As a result, the sign of the observable peak value of DSC determines the sign of m_1. The same reasoning may apply to DCS in the determination of the sign of n_1. The estimated displacement, $\hat{d} = (\hat{m}_1,\hat{n}_1)$, can thus be found by locating the peaks of DCS and DSC over $\{0,\ldots,N-1\}^2$ or over an index range of interest, usually, $\Phi = \{0,\ldots,N/2\}^2$ for slow motion. How the peak signs determine the direction of movement is summarized in Table I. Once

Sign of DSC Peak	Sign of DCS Peak	Peak Index	Direction of Motion
$+$	$+$	(m_1, n_1)	northeast
$+$	$-$	$(m_1, -(n_1+1))$	southeast
$-$	$+$	$(-(m_1+1), n_1)$	northwest
$-$	$-$	$(-(m_1+1), -(n_1+1))$	southwest

TABLE I
DETERMINATION OF DIRECTION OF MOVEMENT (m_1, n_1) FROM THE SIGNS OF DSC AND DCS

the direction is found, $\hat{d}$ can be estimated accordingly:

$$\hat{m}_1 = \begin{cases} i_{DSC} = i_{DCS}, & \text{if } DSC(i_{DSC}, j_{DSC}) > 0, \\ -(i_{DSC}+1) = -(i_{DCS}+1), & \text{if } DSC(i_{DSC}, j_{DSC}) < 0, \end{cases} \tag{37}$$

$$\hat{n}_1 = \begin{cases} j_{DCS} = j_{DSC}, & \text{if } DCS(i_{DCS}, j_{DCS}) > 0, \\ -(j_{DCS}+1) = -(j_{DSC}+1), & \text{if } DCS(i_{DCS}, j_{DCS}) < 0, \end{cases} \tag{38}$$

where

$$(i_{DCS}, j_{DCS}) = arg \max_{m,n\in\Phi} |DCS(m,n)|, \tag{39}$$

$$(i_{DSC}, j_{DSC}) = arg \max_{m,n\in\Phi} |DSC(m,n)|. \tag{40}$$

Normally, these two peak indices are consistent but in noisy circumstances, they may not agree. In this case, an arbitration rule must be made to pick the best index (i_D, j_D) in terms of minimum nonpeak-to-peak ratio (NPR):

$$(i_D, j_D) = \begin{cases} (i_{DSC}, j_{DSC}) & \text{if } NPR(DSC) < NPR(DCS), \\ (i_{DCS}, j_{DCS}) & \text{if } NPR(DSC) > NPR(DCS). \end{cases} \tag{41}$$

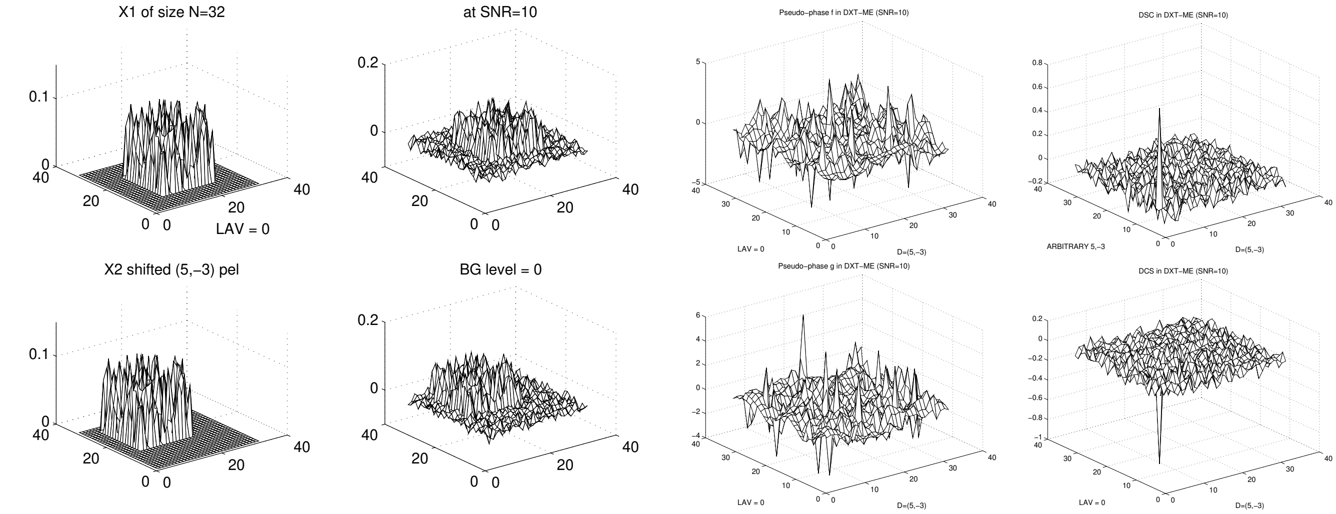
This index (i_D, j_D) will then be used to determine $\hat{d}$ by (37) and (38). Here NPR is defined as the ratio of the average of all absolute non-peak values to the absolute peak value. Thus $0 \le NPR \le 1$, and for a pure impulse function, $NPR = 0$. Such an approach to choose the best index among the two indices is found empirically to improve the noise immunity of this estimator.

In situations where slow motion is preferred, it is better to search the peak value in a zigzag way as widely used in DCT-based hybrid video coding [20][21] . Starting from the index $(0,0)$, zigzagly scan all the DCS (or DSC) values and mark the point as the new peak index if the value at that point (i,j) is larger than the current peak value by more than a preset threshold θ:

$$(i_{DCS}, j_{DCS}) = (i,j) \text{ if } DCS(i,j) > DCS(i_{DCS}, j_{DCS}) + \theta, \tag{42}$$

$$(i_{DSC}, j_{DSC}) = (i, j) \text{ if } DSC(i, j) > DSC(i_{DSC}, j_{DSC}) + \theta. \tag{43}$$

In this way, large spurious spikes at the higher index points will not affect the performance of the estimator and thus improve its noise immunity further.



(a) original inputs x_1 and x_2 (b) noise added (c) f and g (d) DSC and DCS

Fig. 7. DXT-ME performed on the images of an object moving in the direction (5, -3) with additive white Gaussian noise at SNR = 10 dB in a completely dark environment

Fig. 7 demonstrates this DXT-ME algorithm. Images of a rectangularly-shaped moving object with arbitrary texture are generated as in Fig. 7(a) and corrupted by additive white Gaussian noise at SNR = 10 dB as in Fig. 7(b). The resulted pseudo phase functions f and g, as well as DCS and DSC, are depicted in Fig. 7 (c) and (d) correspondingly. Large peaks can be seen clearly in Fig. 7(d) on rough surfaces caused by noise in spite of noisy input images. The positions of these peaks give us an accurate motion estimate $(5, -3)$.

A. Stability of DXT-ME Motion Estimator

The stability of this motion estimator depends upon the property of the determinant of the system matrix $|\mathbf{Z}_{t-1}(k,l)|$ in (23). A zero or near-zero value of $|\mathbf{Z}_{t-1}(k,l)|$ may jeopardize the performance. However, it can be shown analytically that this determinant will rarely be zero. Some algebraic manupilations on the determinant of $\mathbf{Z}_{t-1}(k,l)$ give us this close form of $|\mathbf{Z}_{t-1}(k,l)|$:

$$|\mathbf{Z}_{t-1}(k,l)| = \begin{vmatrix} Z^{cc}_{t-1}(k,l) & -Z^{cs}_{t-1}(k,l) & -Z^{sc}_{t-1}(k,l) & Z^{ss}_{t-1}(k,l) \\ Z^{cs}_{t-1}(k,l) & Z^{cc}_{t-1}(k,l) & -Z^{ss}_{t-1}(k,l) & -Z^{sc}_{t-1}(k,l) \\ Z^{sc}_{t-1}(k,l) & -Z^{ss}_{t-1}(k,l) & Z^{cc}_{t-1}(k,l) & -Z^{cs}_{t-1}(k,l) \\ Z^{ss}_{t-1}(k,l) & Z^{sc}_{t-1}(k,l) & Z^{cs}_{t-1}(k,l) & Z^{cc}_{t-1}(k,l) \end{vmatrix}$$

$$
\begin{aligned}
&= (Z^{cs}_{t-1}(k,l)^2 - Z^{sc}_{t-1}(k,l)^2)^2 + (Z^{cc}_{t-1}(k,l)^2 - Z^{ss}_{t-1}(k,l)^2)^2 \\
&\quad +2(Z^{cc}_{t-1}(k,l)Z^{cs}_{t-1}(k,l) + Z^{sc}_{t-1}(k,l)Z^{ss}_{t-1}(k,l))^2 \\
&\quad +2(Z^{cc}_{t-1}(k,l)Z^{sc}_{t-1}(k,l) + Z^{cs}_{t-1}(k,l)Z^{ss}_{t-1}(k,l))^2 \qquad (44)
\end{aligned}
$$

in which $|\mathbf{Z}_{t-1}(k,l)| = 0$ implies that

$$
Z^{cs}_{t-1}(k,l) = \pm Z^{sc}_{t-1}(k,l), \qquad Z^{cc}_{t-1}(k,l) = \pm Z^{ss}_{t-1}(k,l),
$$

$$
Z^{cc}_{t-1}(k,l)Z^{cs}_{t-1}(k,l) = -Z^{sc}_{t-1}(k,l)Z^{ss}_{t-1}(k,l), \quad Z^{cc}_{t-1}(k,l)Z^{sc}_{t-1}(k,l) = -Z^{cs}_{t-1}(k,l)Z^{ss}_{t-1}(k,l).
$$

However, the last two equalities allow only two possible conditions:

$$
\textit{either} \quad Z^{cs}_{t-1}(k,l) = Z^{sc}_{t-1}(k,l), \quad Z^{cc}_{t-1}(k,l) = -Z^{ss}_{t-1}(k,l), \qquad (45)
$$

$$
\textit{or} \quad Z^{cs}_{t-1}(k,l) = -Z^{sc}_{t-1}(k,l), \quad Z^{cc}_{t-1}(k,l) = Z^{ss}_{t-1}(k,l). \qquad (46)
$$

Alternatively, in their explicit compact forms,

$$
\sum_{m,n=0}^{N-1} x_{t-1}(m,n)\sin[\frac{\pi}{N}(km \mp ln)] = 0, \qquad (47)
$$

$$
\sum_{m,n=0}^{N-1} x_{t-1}(m,n)\cos[\frac{\pi}{N}(km \mp ln)] = 0. \qquad (48)
$$

Here the minus signs in (47) and (48) correspond to the first condition in (45) and the plus signs correspond to (46). Satisfying either condition requires $x_{t-1}(m,n) \equiv 0$. Therefore it is very unlikely that this determinant is zero and as such, the DXT-ME is a stable estimator. Even so, if $\mathbf{Z}_{t-1}(k,l) = 0$ really happens or $\mathbf{Z}_{t-1}(k,l)$ is less than a threshold, then we can let $f(k,l) = g(k,l) = 1$, which is equivalent to the situation when $x_{t-1}(m,n) \equiv 0$. In this way, the catastrophical effect of computational precision of a certain implementation on the stability of DXT-ME will be kept to minimum or even eliminated.

B. Motion Estimation In A Uniformly Bright Background

What if an object is moving in a uniformly bright background instead of a completely dark environment? It can be shown analytically and empirically that uniformly bright background introduces only very small spikes which does not affect the accuracy of the estimate. Suppose that $\{x_{t-1}(m,n)\}$ and $\{x_t(m,n)\}$ are pixel values of 2 consecutive frames of an object displaced by (m_1, n_1) on a uniformly bright background. Then let $y_t(m,n)$ and $y_{t-1}(m,n)$ be the pixel value of $x_t(m,n)$ and $x_{t-1}(m,n)$ subtracted by the background pixel value c ($c > 0$) respectively:

$$
y_t(m,n) = x_t(m,n) - c, \qquad (49)
$$

$$y_{t-1}(m,n) = x_{t-1}(m,n) - c. \tag{50}$$

In this way, $\{x_{t-1}(m,n)\}$ and $\{x_t(m,n)\}$ can be considered as the images of an object moving in a dark environment. Denote $\mathbf{Z}_{t-1}(k,l)$ as the system matrix of the input image x_{t-1} and $\mathbf{U}_{t-1}(k,l)$ as that of y_{t-1} for $k,l \in \mathcal{N}$. Also let $\vec{\mathbf{x}}_t(k,l)$ be the vector of the 2D-DCT-II coefficients of x_t and $\vec{\mathbf{y}}_t(k,l)$ be the vector for y_t. Applying the DXT-ME algorithm to both situations, we have, for $k,l \in \mathcal{N}$,

$$\mathbf{Z}_{t-1}(k,l) \cdot \vec{\theta}_{m_1 n_1}(k,l) = \vec{\mathbf{x}}_t(k,l), \tag{51}$$

$$\mathbf{U}_{t-1}(k,l) \cdot \vec{\phi}_{m_1 n_1}(k,l) = \vec{\mathbf{y}}_t(k,l). \tag{52}$$

Here $\vec{\phi}_{m_1 n_1}(k,l)$ is the vector of the computed pseudo phases for the case of dark background and thus

$$\vec{\phi}_{m_1 n_1}(k,l) = [g^{CC}_{m_1 n_1}(k,l),\ g^{CS}_{m_1 n_1}(k,l),\ g^{SC}_{m_1 n_1}(k,l),\ g^{SS}_{m_1 n_1}(k,l)]^T$$

but $\vec{\theta}_{m_1 n_1}(k,l)$ is for uniformly bright background and

$$\vec{\theta}_{m_1 n_1}(k,l) = [\hat{g}^{CC}_{m_1 n_1}(k,l),\ \hat{g}^{CS}_{m_1 n_1}(k,l),\ \hat{g}^{SC}_{m_1 n_1}(k,l),\ \hat{g}^{SS}_{m_1 n_1}(k,l)]^T \neq \vec{\phi}_{m_1 n_1}(k,l).$$

Starting from the definition of each element in $\mathbf{Z}_{t-1}(k,l)$ and $\vec{\mathbf{x}}_t(k,l)$, we obtain

$$\mathbf{Z}_{t-1}(k,l) = \mathbf{U}_{t-1}(k,l) + c \cdot \mathbf{D}(k,l), \tag{53}$$

$$\vec{\mathbf{x}}_t(k,l) = \vec{\mathbf{y}}_t(k,l) + c \cdot \vec{\mathbf{c}}(k,l), \tag{54}$$

where $\mathbf{D}(k,l)$ is the system matrix with $\{d(m,n) = 1,\ \forall m,n = \{0,\ldots,N-1\}\}$ as input and $\vec{\mathbf{c}}(k,l)$ is the vector of the 2D-DCT-II coefficients of $d(m,n)$. Substituting (53) and (54) into (52), we get

$$\mathbf{Z}_{t-1}(k,l) \cdot \vec{\theta}_{m_1 n_1}(k,l) = \mathbf{Z}_{t-1}(k,l) \cdot \vec{\phi}_{m_1 n_1}(k,l) + c \cdot [\vec{\mathbf{c}}(k,l) - \mathbf{D}(k,l) \cdot \vec{\phi}_{m_1 n_1}(k,l)]. \tag{55}$$

Since $\vec{\mathbf{c}}(k,l) = \mathbf{D}(k,l) \cdot \vec{\phi}_{00}(k,l)$, (55) becomes

$$\vec{\theta}_{m_1 n_1}(k,l) = \vec{\phi}_{m_1 n_1}(k,l) + c\mathbf{Z}^{-1}_{t-1}(k,l)\mathbf{D}(k,l)[\vec{\phi}_{00}(k,l) - \vec{\phi}_{m_1 n_1}(k,l)], \tag{56}$$

provided that $|\mathbf{Z}_{t-1}(k,l)| \neq 0$. Similar results can also be found at block boundaries. Referring to (24), we know that $\mathbf{D}(k,l)$ is composed of $D^{cc}(k,l)$, $D^{cs}(k,l)$, $D^{sc}(k,l)$, and $D^{ss}(k,l)$, each of which is a separable function made up by

$$D^c(k) \equiv \frac{2}{N}C(k)\sum_{m=0}^{N-1}\cos[\frac{k\pi}{N}m] = \frac{2}{N}C(k)\{0.5[1-(-1)^k] + N \cdot \delta(k)\},$$

$$D^s(k) \equiv \frac{2}{N}C(k)\sum_{m=0}^{N-1}\sin[\frac{k\pi}{N}m] = \begin{cases} \frac{2}{N}C(k)\frac{[1-(-1)^k]}{2\tan\frac{k\pi}{2N}}, & \text{for } k \neq 0, \\ 0, & \text{for } k = 0. \end{cases}$$

From the above equations, we can see that $D^c(k) = D^s(k) = 0$ if k is even, and for odd $k > 0$, $D^c(k) = \frac{2}{N}$ while $D^s(k) = \frac{2}{N \tan \frac{k\pi}{2N}}$. Hence, $D^{cc}(k,l) = D^{cs}(k,l) = D^{sc}(k,l) = D^{ss}(k,l) = 0$ if either k or l is even. As a result, $\vec{\theta}_{m_1 n_1}(k,l) = \vec{\phi}_{m_1 n_1}(k,l)$ if either k or l is even. For odd indices k and l, it is possible to find a constant s and a matrix $\mathbf{N}(k,l) \in R^{4\times 4}$ such that $\mathbf{U}_{t-1}(k,l) = s[\mathbf{D}(k,l) - \mathbf{N}(k,l)]$ and $|\mathbf{N}(k,l)\mathbf{D}^{-1}(k,l)| < 1$ for $|\mathbf{D}(k,l)| \neq 0$. Therefore, for $|\frac{s}{s+c}\mathbf{N}(k,l)\mathbf{D}^{-1}(k,l)| < 1$,

$$
\begin{aligned}
c\mathbf{Z}_{t-1}^{-1}(k,l)\mathbf{D}(k,l) &= \frac{c}{s+c}[\mathbf{I} - \frac{s}{s+c}\mathbf{N}(k,l)\mathbf{D}^{-1}(k,l)]^{-1} &\quad (57)\\
&= \frac{c}{s+c}\{\mathbf{I} + \frac{s}{s+c}\mathbf{N}(k,l)\mathbf{D}^{-1}(k,l) + [\frac{s}{s+c}\mathbf{N}(k,l)\mathbf{D}^{-1}(k,l)]^2 + \ldots\}. &\quad (58)
\end{aligned}
$$

If we lump all the high-order terms of $\frac{s}{s+c}\mathbf{N}(k,l)\mathbf{D}^{-1}(k,l)$ in one term $\mathbf{H}(k,l)$, then

$$\vec{\theta}_{m_1 n_1}(k,l) = \vec{\phi}_{m_1 n_1}(k,l) + [\frac{c}{s+c} + \mathbf{H}(k,l)][\vec{\phi}_{00}(k,l) - \vec{\phi}_{m_1 n_1}(k,l)]. \quad (59)$$

Usually, $0 \leq c, s \leq 255$ for the maximum gray level equal to 255. Typically $s = 1$. For moderately large c, $\mathbf{H}(k,l)$ is very small. Define the subsampled version of the pseudo-phase function $\vec{\phi}_{ab}(k,l)$ as

$$\vec{\lambda}_{ab}(k,l) \equiv \begin{cases} \vec{\phi}_{ab}(k,l), & \text{if both } k \text{ and } l \text{ are odd,} \\ 0, & \text{otherwise .} \end{cases} \quad (60)$$

Then

$$\vec{\theta}_{m_1 n_1}(k,l) = \vec{\phi}_{m_1 n_1}(k,l) + [\frac{c}{s+c} + \mathbf{H}(k,l)]\{\vec{\lambda}_{00} - \vec{\lambda}_{m_1 n_1}\}. \quad (61)$$

Recall that a 2D-IDCT-II operation on $\vec{\phi}_{m_1 n_1}(k,l)$ or $\vec{\phi}_{00}(k,l)$ produces $\vec{\delta}_{m_1 n_1}$ or $\vec{\delta}_{00}$, respectively, where

$$\vec{\delta}_{ab}(m,n) = \begin{bmatrix} (\delta(m-a) + \delta(m+a+1))(\delta(n-b) + \delta(n+b+1)) \\ (\delta(m-a) + \delta(m+a+1))(\delta(n-b) - \delta(n+b+1)) \\ (\delta(m-a) - \delta(m+a+1))(\delta(n-b) + \delta(n+b+1)) \\ (\delta(m-a) - \delta(m+a+1))(\delta(n-b) - \delta(n+b+1)) \end{bmatrix}.$$

Therefore,

$$\vec{\mathbf{d}}(m,n) \equiv \text{ 2D-DCT-II}\{\vec{\theta}_{m_1 n_1}\} = \vec{\delta}_{m_1 n_1}(m,n) + \frac{c}{s+c} \text{ 2D-DCT-II}\{\vec{\lambda}_{00} - \vec{\lambda}_{m_1 n_1}\} + \vec{\mathbf{n}}(m,n), \quad (62)$$

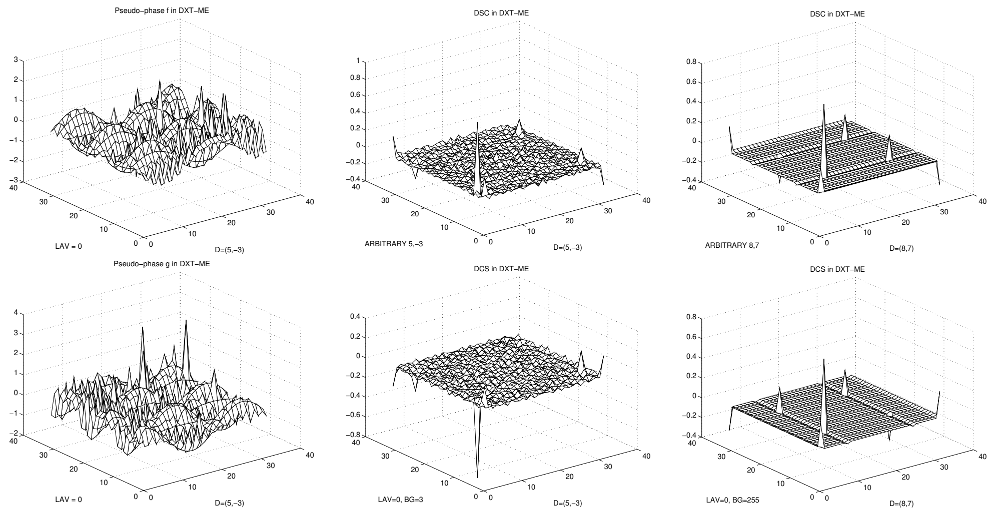
where $\vec{\mathbf{n}}$ is the noise term contributed from 2D-DCT-II$\{\mathbf{H}(k,l)[\vec{\lambda}_{00} - \vec{\lambda}_{m_1 n_1}]\}$. Because $\vec{\lambda}_{ab}$ is equivalent to downsampling $\vec{\phi}_{ab}$ in a 2D index space and it is known that downsampling produces in the transform domain mirror images of magnitude only one-fourth of the original and of sign depending on the transform function, we obtain

$$
\begin{aligned}
\vec{\mathbf{E}}_{m_1 n_1}(m,n) \equiv \text{ 2D-DCT-II}\{\vec{\lambda}_{m_1 n_1}\} = \frac{1}{4}[&\vec{\delta}_{m_1 n_1}(m,n) + \text{diag}(\vec{\zeta}_1) \cdot \vec{\delta}_{(N-1-m_1)n_1}(m,n) \quad (63)\\
&+\text{diag}(\vec{\zeta}_2) \cdot \vec{\delta}_{m_1(N-1-n_1)}(m,n) + \text{diag}(\vec{\zeta}_3) \cdot \vec{\delta}_{(N-1-m_1)(N-1-n_1)}(m,n)],
\end{aligned}
$$

where diag$(\cdot)$ is the diagonal matrix of a vector and $\vec{\zeta}_i$ $(i = 1, 2, 3)$ is a vector consisting of ± 1. A similar expression can also be established for 2D-DCT-II$\{\vec{\lambda}_{00}\}$. In conclusion,

$$\vec{\mathbf{d}}(m, n) = \vec{\delta}_{m_1 n_1}(m, n) + \frac{c}{4(s+c)}[\vec{\mathbf{E}}_{00}(m, n) - \vec{\mathbf{E}}_{m_1 n_1}(m, n)] + \vec{\mathbf{n}}(m, n). \tag{64}$$

The above equation predicts the presence of a very small noise term $\vec{\mathbf{n}}$ and several small spikes, $\vec{\mathbf{E}}_{00}$ and $\vec{\mathbf{E}}_{m_1 n_1}$, of magnitude moderated by $\frac{c}{4(s+c)}$ which are much smaller than the displacement peak, as displayed in Fig. 8 (b) and (c) where $\vec{\mathbf{n}}$ for the case of $c = 3$ in (b) is observable but very small and can be regarded as noise whereas $\vec{\mathbf{n}}$ is practically absent as in (c) when $c = 255$.



(a) f and g (b) DSC and DCS (c) another DSC and DCS

Fig. 8. (a)(b) An object is moving in the direction (5, -3) in a uniformly bright background ($c = 3$). (c) Another object is moving northeast (8,7) for background pixel values $= c = 255$.

C. Computational Issues and Complexity

The block diagram in Fig. 5(a) shows that a separate 2D-DCT-I is needed in addition to the standard DCT (2D-DCT-II). This is undesirable from the complexity viewpoint. However, this problem can be circumvented by considering the point-to-point relationship between 2D-DCT-I and 2D-DCT-II coefficients in the frequency domain for $k, l \in \mathcal{N}$:

$$\begin{bmatrix} Z^{cc}_{t-1}(k,l) \\ Z^{cs}_{t-1}(k,l) \\ Z^{sc}_{t-1}(k,l) \\ Z^{ss}_{t-1}(k,l) \end{bmatrix} = \begin{bmatrix} \cos\frac{k\pi}{2N}\cos\frac{l\pi}{2N} & \cos\frac{k\pi}{2N}\sin\frac{l\pi}{2N} & \sin\frac{k\pi}{2N}\cos\frac{l\pi}{2N} & \sin\frac{k\pi}{2N}\sin\frac{l\pi}{2N} \\ -\cos\frac{k\pi}{2N}\sin\frac{l\pi}{2N} & \cos\frac{k\pi}{2N}\cos\frac{l\pi}{2N} & -\sin\frac{k\pi}{2N}\sin\frac{l\pi}{2N} & \sin\frac{k\pi}{2N}\cos\frac{l\pi}{2N} \\ -\sin\frac{k\pi}{2N}\cos\frac{l\pi}{2N} & -\sin\frac{k\pi}{2N}\sin\frac{l\pi}{2N} & \cos\frac{k\pi}{2N}\cos\frac{l\pi}{2N} & \cos\frac{k\pi}{2N}\sin\frac{l\pi}{2N} \\ \sin\frac{k\pi}{2N}\sin\frac{l\pi}{2N} & -\sin\frac{k\pi}{2N}\cos\frac{l\pi}{2N} & -\cos\frac{k\pi}{2N}\sin\frac{l\pi}{2N} & \cos\frac{k\pi}{2N}\cos\frac{l\pi}{2N} \end{bmatrix} \begin{bmatrix} X^{cc}_{t-1}(k,l) \\ X^{cs}_{t-1}(k,l) \\ X^{sc}_{t-1}(k,l) \\ X^{ss}_{t-1}(k,l) \end{bmatrix} \tag{65}$$

where X^{cc}_{t-1}, X^{cs}_{t-1}, X^{sc}_{t-1}, and X^{ss}_{t-1} are the 2D-DCT-II coefficients of the previous frame. Similar relation also exists for the coefficients at block boundaries. This observation results in the simple structure in

Fig. 5(b), where Block T is a coefficient transformation unit realizing (65).

Stage	Component	Computational Complexity
1	2D-DCT-II	$O_{dct} = O(N)$
	Coeff. Transformation Unit (T)	$O(N^2)$
2	Pseudo Phase Computation	$O(N^2)$
3	2D-IDCT-II	$O_{dct} = O(N)$
4	Peak Searching	$O(N^2)$
	Estimation	$O(1)$

TABLE II
COMPUTATIONAL COMPLEXITY OF EACH STAGE IN DXT-ME

If the DCT has computational complexity O_{dct}, the overall complexity of DXT-ME is $O(N^2) + O_{dct}$ with the complexity of each component summarized in Table II. The computational complexity of the pseudo phase computation component is only $O(N^2)$ for an $N \times N$ block and so is the unit to determine the displacement. For the computation of the pseudo phase functions $f(\cdot,\cdot)$ in (33) and $g(\cdot,\cdot)$ in (34), DSCT, DCST and DSST coefficients (regarded as DST coefficients) must be calculated in addition to DCCT coefficients (i.e. the usual 2D DCT). However all these coefficients can be generated with little overhead in the course of computing 2D DCT coefficients. As a matter of fact, a parallel and fully-pipelined 2D DCT lattice structure has been developed [25], [26], [27] to generate 2D DCT coefficients at a cost of $O(N)$ operations. This DCT coder computes DCT and DST coefficients dually due to its internal lattice architecture. These internally generated DST coefficients can be output to the DXT-ME module for pseudo phase computation. This same lattice structure can also be modified as a 2D IDCT which also has $O(N)$ complexity. To sum up, the computational complexity of this DXT-ME is only $O(N^2)$, much lower than the $O(N^4)$ complexity of BMA-ME.

A closer look at (33), (34) and (65) reveals that the operations of pseudo phase computation and coefficient transformation are performed independently at each point (k, l) in the transform domain and therefore are inherently highly parallel operations. Since most of the operations in the DXT-ME algorithm involve mainly pseudo phase computation and coefficient transformation in addition to DCT and Inverse DCT operations which have been studied extensively, the DXT-ME algorithm can easily be implemented on highly parallel array processors or dedicated circuits. This is very different from BMA-ME which requires shifting of pixels and summation of differences of pixel values and hence discourages parallel implementation.

IV. Preprocessing Steps and Overlapping Approach

For complicated video sequences in which objects may move across the border of blocks in non-uniform background, preprocessing can be employed to enhance the features of motion objects and avoid violation of the assumption made for DXT-ME before feeding the images into the DXT-ME motion estimator. Intuitively speaking, the DXT-ME algorithm tries to match the features of any object on two consecutive frames so that any translation motion can be estimated regardless of the shape and texture of the object as long as these two frames contain significant energy level of the object features. Due to this feature matching property of the DXT-ME algorithm, effective preprocessing will improve the performance of motion estimation if preprocessing can enhance the object features in the original sequence. In order to keep the computational complexity of the overall motion estimator low, the chosen preprocessing function must be simple but effective in the sense that unwanted features will not affect the accuracy of estimation. Our study found that both edge extraction and frame differentiation are simple and efective schemes for extraction of motion information.

Edges of an object can represent the object itself in motion estimation as its features [28] and contain the information of motion without violating the assumption for DXT-ME. The other advantage of edge extraction is that any change in the illumination condition does not alter the edge information and in turn makes no false motion estimates by the DXT-ME algorithm. Since we only intend to extract the main features of moving objects while keeping the overall complexity low, we employ a very simple edge detection by convolving horizontal and vertical Sobel operators of size 3×3

$$H_s = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \ V_s = H_s^T \tag{66}$$

with the image to obtain horizontal and vertical gradients respectively and then combine both gradients by taking the square root of the sum of the squares of both gradients [29] . Edge detection provides us the features of moving objects but also the features of the background (stationary objects) which is undesirable. However, if the features of the background have smaller energy than those of moving objects within every block containing moving objects, then the background features will not affect the performance of DXT-ME. The computational complexity of this preprocessing step is only $O(N^2)$ and thus the overall computational complexity is still $O(N^2)$.

Frame differentiation generates an image of the difference of two consecutive frames. This frame differentiated image contains no background objects but the difference of moving objects between two frames. The DXT-ME estimator operates directly on this frame differentiated sequence to predict motion in the original sequence. The estimate will be good if the moving objects are moving constantly in one direction in three consecutive frames. For 30 frames per second, the standard NTSC frame rate, objects can usually be viewed as moving at a constant speed in three consecutive frames. Obviously, this step also has only $O(N^2)$ computational complexity.

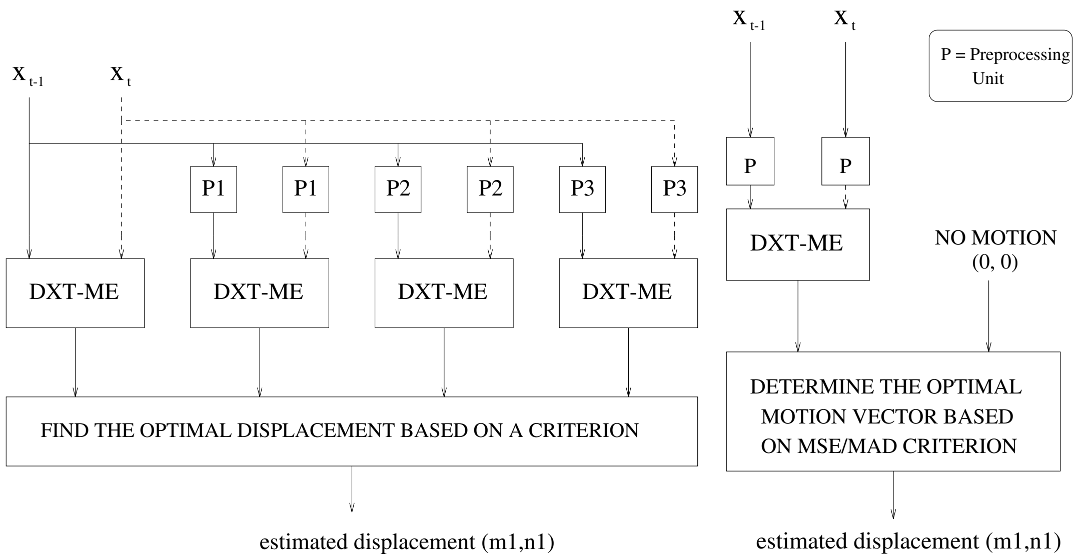


Fig. 9. Block Diagrams of Extended DXT-ME Estimator (E-DXT-ME) and Simplified Extended DXT-ME (SE-DXT-ME)

Alternatively, instead of using only one preprocessing function, we can employ several simple difference operators in the preprocessing step to extract features of images as shown in Fig. 9(a), in which four DXT-ME estimators generate four candidate estimates of which one can be chosen as the final estimated displacement based upon either the mean squared error per pixel (MSE) [7] or the mean of absolute differences per pixel (MAD) criteria [8].

Preferably, a simple decision rule similar to the one used in the MPEG-1 standard [21] , as depicted in Fig. 9(b), is used to choose among the DXT-ME estimate and no motion. This simplified extended DXT-ME motion estimator works very well as will be shown in the next section.

In Section III, we mention that peaks of DSC and DCS are searched over a fixed index range of interest $\Phi = \{0, \ldots, N/2\}^2$. However, if we follow the partitioning approach used in BMA-ME, then we may dynamically adjust Φ. At first, partition the whole current frame into $bs \times bs$ nonoverlapping reference blocks shown as the shaded area in Fig. 10(a). Each reference block is associated with a larger search area (of size sa) in the previous frame (the dotted region in the same figure) in the same way as for BMA-ME. From the position of a reference block and its associated search area, a search range

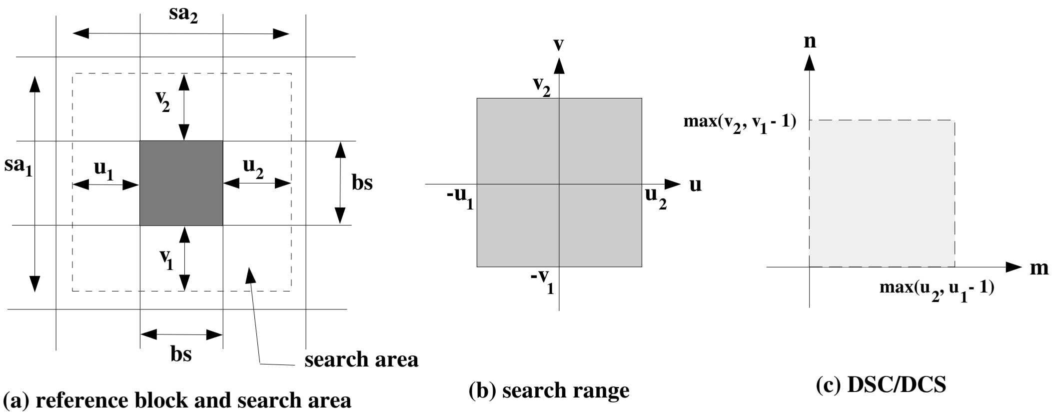


Fig. 10. Overlapping approach

$\mathcal{D} = \{(u,v) : -u_1 \leq u \leq u_2,\ -v_1 \leq v \leq v_2\}$ can then be determined as in Fig. 10(b). Differing from BMA-ME, DXT-ME requires that the reference block size and the search area size must be equal. Thus, instead of using the reference block, we use the block of the same size and position in the current frame as the search area of the previous frame. The peak values of DSC and DCS are searched in a zigzag way as described in Section III over this index range $\Phi = \{0, \ldots, \max(u_2, u_1 - 1)\} \times \{0, \ldots, \max(v_2, v_1 - 1)\}$. In addition to the requirement that the new peak value must be larger than the current peak value by a preset threshold, it is necessary to examine if the motion estimate determined by the new peak index lies in the search region $\mathcal{D}$. Since search areas overlap on one another, the SE-DXT-ME architecture utilizing this approach is called Overlapping SE-DXT-ME.

V. Simulation Results

Simulations have been performed on a number of video sequences with different characteristics. The performance of the DXT-ME scheme is evaluated in terms of MSE (mean squared error per pel) defined as $MSE = \frac{\sum_{m,n}[\hat{x}(m,n) - x(m,n)]^2}{N^2}$, and compared with that of the full search block matching method (BMA-ME), which minimizes the MSE function over the whole search area:

$$\hat{d} = (\hat{u}, \hat{v}) = arg \min_{u,v} \frac{\sum_{m,n}[x_2(m,n) - x_1(m-u, n-v)]^2}{N^2}.$$

The MSE values of two consecutive frames without motion compensation (DIF) are also computed for comparison. The MSE value for DXT-ME is expected to be upper bounded by the MSE value without motion compensation.

To test the performance of DXT-ME on noisy images, an image of a small car (SCAR_1) is manually shifted to produce the second frame (SCAR_2) with a known displacement and additive Gaussian noise is added to attain a desired signal-to-ratio (SNR) level. Since the object (small car) moves within the

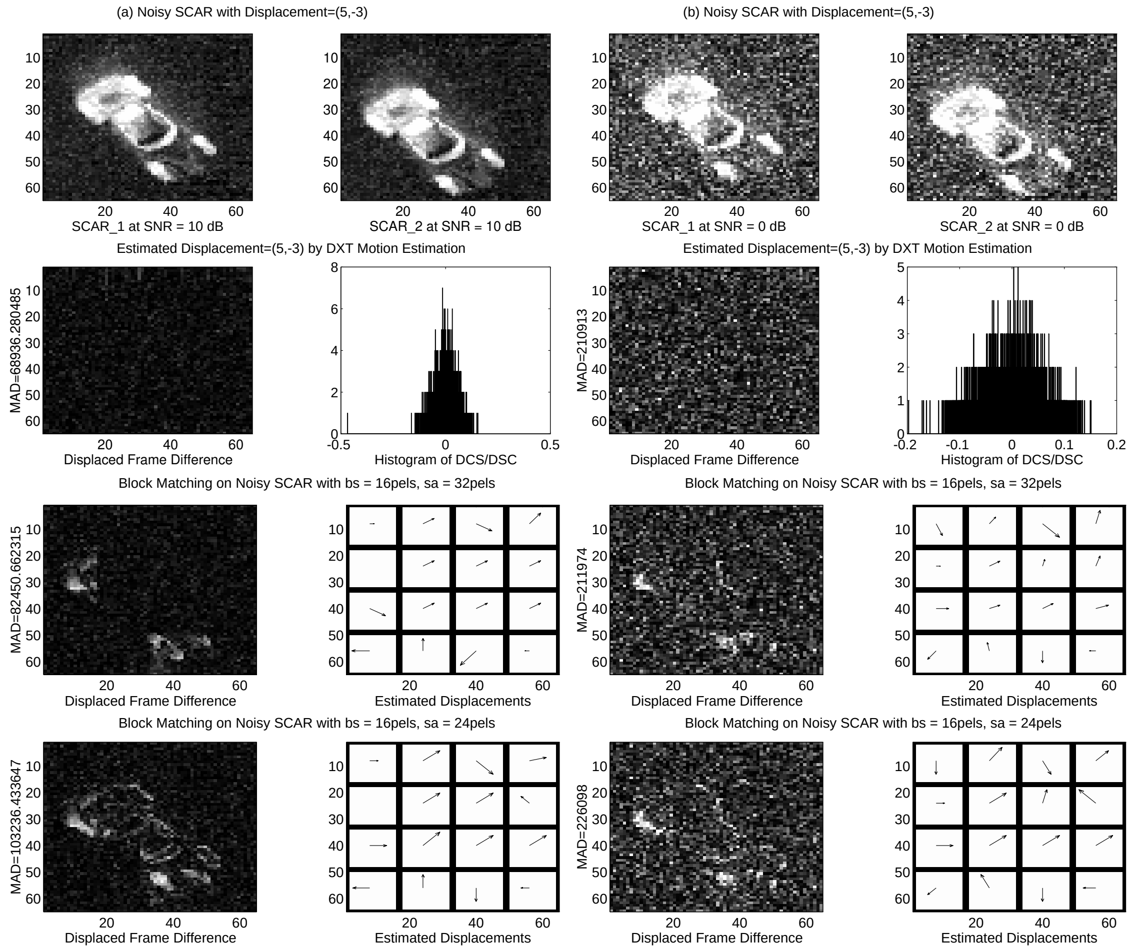


Fig. 11. Comparison of DXT-ME of size 64 by 64 pels with Full-Search Block Matching Method (BMA-ME) of block size (bs = 16 pels) but different search areas (sa = 32 or 24 pels) on a noisy small car (SCAR) with (a) SNR = 10 dB, (b) SNR = 0 dB.

boundary of the frame in a completely darken background, no preprocessing is required. As can be seen in Fig. 11, DXT-ME is performed on the whole image of block size 64×64 and estimates the motion correctly at SNR level even down to 0 dB, whereas BMA-ME produces some wrong motion estimates for boundary blocks and blocks of low signal energy. The values of MAD also indicate better overall performance of DXT-ME over BMA-ME for this sequence. Furthermore, DXT-ME can perform on the whole frame while BMA-ME needs division of the frame into sub-blocks due to the requirement of larger search areas than reference blocks. This is one of the reasons that BMA-ME does not work so well as DXT-ME because smaller block size makes BMA-ME more susceptible to noise and operation of DXT-ME on the whole frame instead of on smaller blocks lends itself to better noise immunity. Even though the Kalman filtering approach [30] can also estimate velocity accurately for a sequence of noisy images, it requires iterative complicated computations while DXT-ME can estimate motion based upon two consecutive frames in one step, requiring low-complexity computations.

(a) Original (b) Edge extracted (c) Frame differentiated

Fig. 12. Frame 57 in the sequence "Flower Garden" (FG)

The first sequence is the "Flower Garden" (FG) sequence where the camera is moving before a big tree and a flower garden in front of a house. Each frame has 720×486 pixels. Simple preprocessing is applied to this sequence: edge extraction or frame differentiation. Simulation of SE-DXT-ME of block size 8, 16, 32, as well as 64 was performed on 100 frames of this sequence. The results were compared with those of BMA-ME of block size 8, 16, 32 with the search area being twice as large as the block size as shown in Fig. 13.

As can be seen in Fig. 12(b), the edge extracted frames contain significant features of moving objects in the original frames so that DXT-ME can estimate the movement of the objects based upon the information provided by the edge extracted frames. Because the camera is moving at a constant speed in one direction, the moving objects occupy almost the whole scene. Therefore, the background features do not interfere with the operation of DXT-ME. The frame differentiated images of the "Flower Garden" sequence , one of which is shown in Fig. 12(c), have the residual energy strong enough for DXT-ME to estimate the motion directly on this frame differentiated sequence due to the constant movement of the camera.

Observable in Fig. 13, a large reference block hampers the performance of BMA-ME indicated by high MSE values whereas increasing block size can boost up the performance of DXT-ME with smaller MSE values. The reason is that a block of larger size for DXT-ME contains more features of objects, which enables DXT-ME to find a better estimate due to its feature matching property, and also a block of larger size means a larger search area because the size of a search area is the same as the block size for DXT-ME. As a matter of fact, BMA-ME is supposed to perform better than DXT-ME if both methods use the same block size because BMA-ME requires a larger search area and thus BMA-ME has more information available before processing than DXT-ME. Therefore, it is not fair to compare BMA-ME with DXT-ME for the same block size. Instead, it is more reasonable to compare BMA-ME

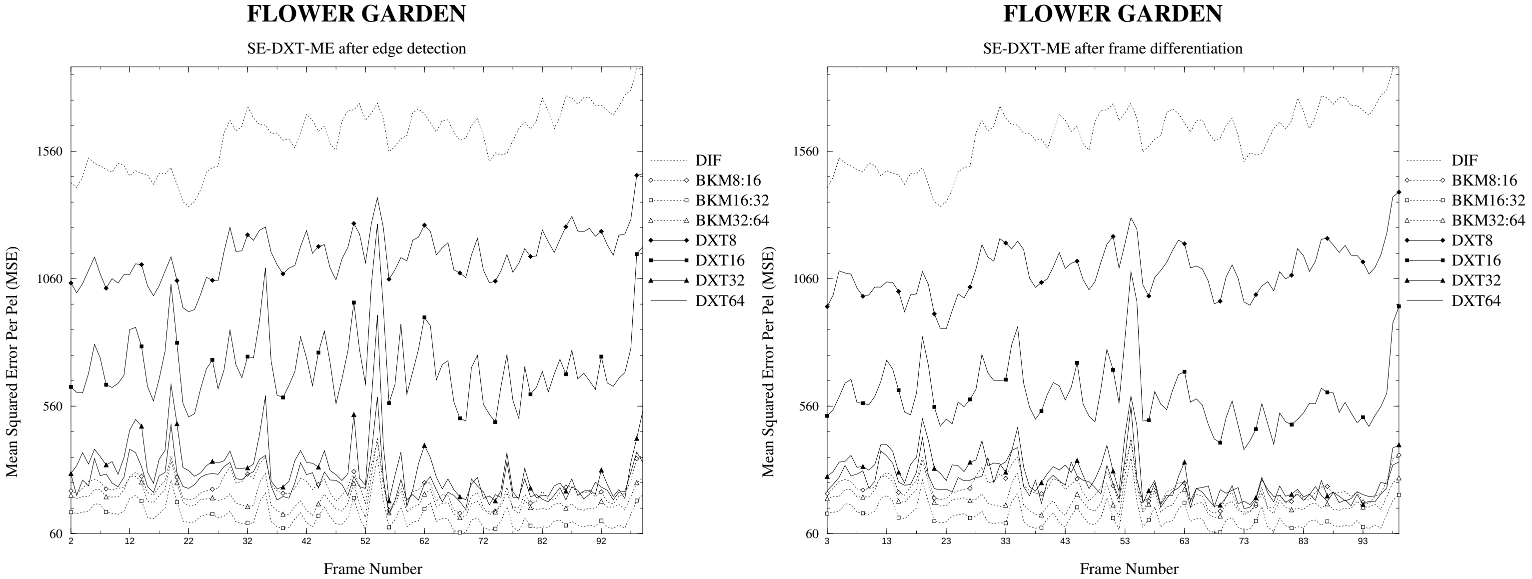
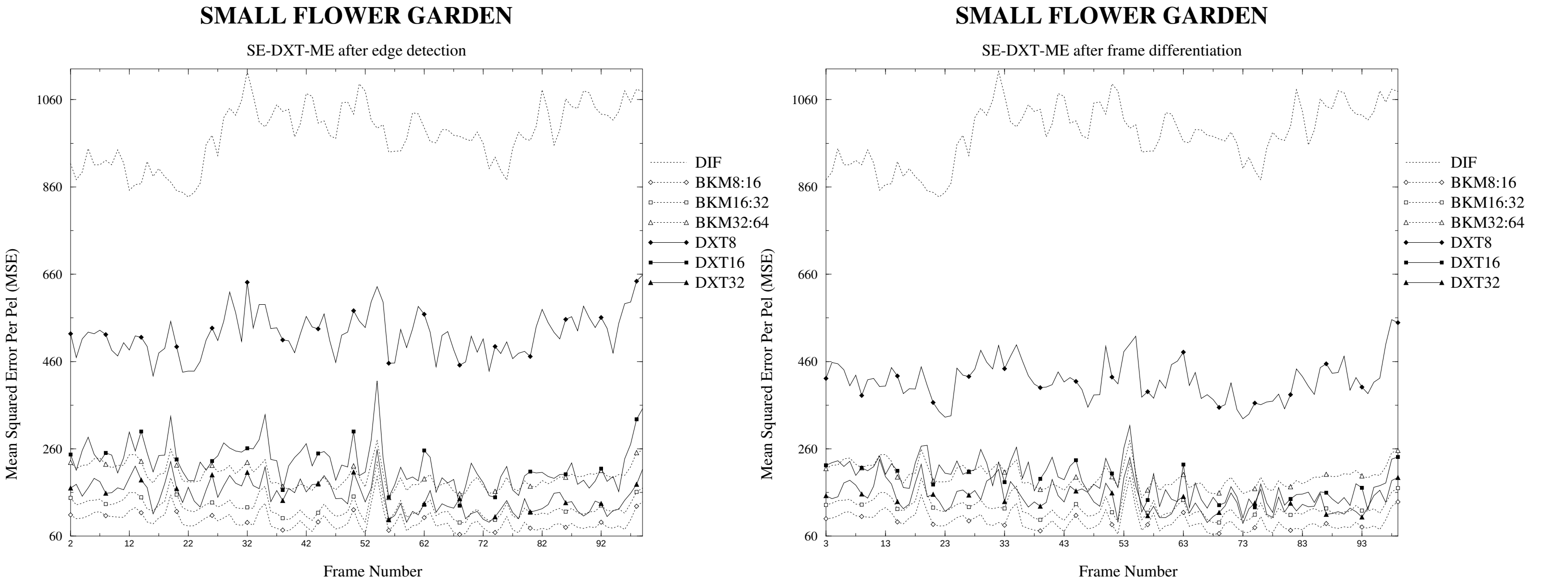


Fig. 13. Comparison of SE-DXT-ME with BMA-ME on "Flower Garden"

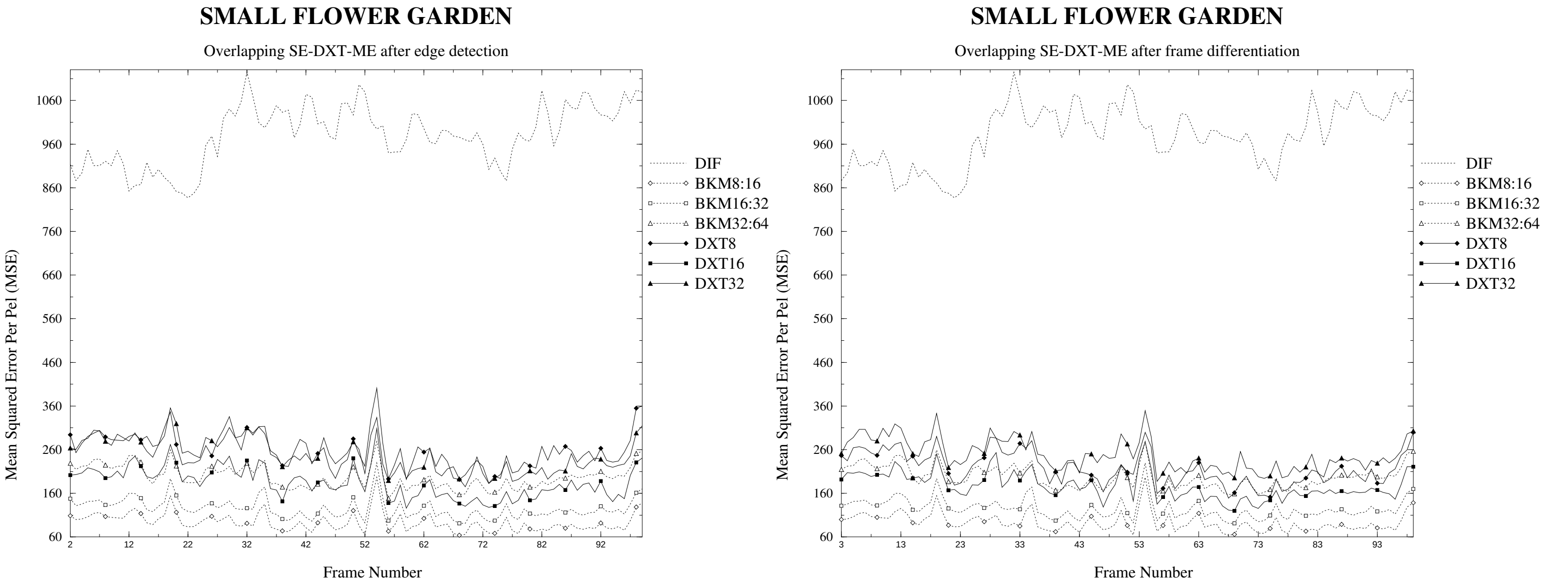
with DXT-ME of the block size equal to the size of the search area of BMA-ME. As shown in Fig. 13, the MSE values for DXT-ME of block size 32 (DXT32) preprocessed by either edge extraction or frame differentiation are comparable with or even better than those for BMA-ME of block size 32 and search size 64.

If the sequence is shrinked to half size (352×224 pixels) and forms the small "Flower Garden" sequence, then each block will capture more features than a block of the same size in the original "Flower Garden" sequence. The simulation results are plotted in Fig. 14 for SE-DXT-ME and Overlapping SE-DXT-ME discussed in Section IV. As expected, the MSE values for SE-DXT-ME of block size 16 now become as small as those for BMA-ME of block size 32 as shown in Fig. 14(a). Some points of the curve for SE-DXT-ME of block size 32 (DXT32) with either preprocessing step are below the points of BMA-ME of search size 32 (BKM16:32). If the overlapping approach is adopted in determining the search region, Fig. 14(b) suggests that both Overlapping SE-DXT-ME and BMA-ME have comparable performance for either preprocessing method. The fact that smaller frame size of the same contents improves the performance of DXT-ME in terms of smaller MSE values recommends the hierachical motion estimation technique combined with DXT-ME (SE-DXT-ME or Overlapping SE-DXT-ME).

Another simulation is done on the "Infrared Car" sequence which has the frame size 192×224 and one obvious moving object, the car moving along the curved road towards the camera fixed on the ground. In Fig. 15(b), the features of both the car and the background are captured in the edge extracted frames. Even though the background features are not desirable, the simulation for SE-DXT-ME of various block sizes shows that the estimates of SE-DXT-ME produce low MSE values compared to the



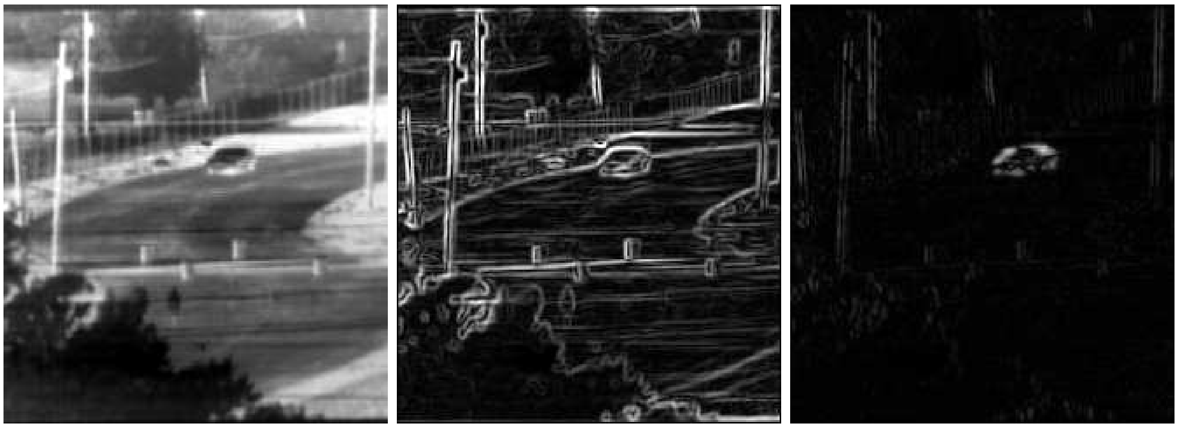
(a) Comparison of SE-DXT-ME with BMA-ME on small "Flower Garden"



(b) Comparison of Overlapping SE-DXT-ME with BMA-ME on small "Flower Garden"

Fig. 14. Simulation on small "Flower Garden"

result of BMA-ME, especially in certain frames such as the 6th to 13th frames shown in Fig. 16(a). This can be explained by taking a closer look at the edge extracted frame in Fig. 15(b) where the background features, such as the trees and poles, are far away from the car, especially in the 6th to 13th frames. For the first few frames, the features of the roadside behind the car mix with the features of the car and affect the performance of DXT-ME but then the car moves away from the roadside towards the camera so that the car features are isolated from the background features and so DXT-ME can estimate motion more accurately. As to the frame differentiated images as shown in Fig. 15(c), the residual energy of the moving car is completely separated from the rest of the scene in most of the preprocessed frames and,



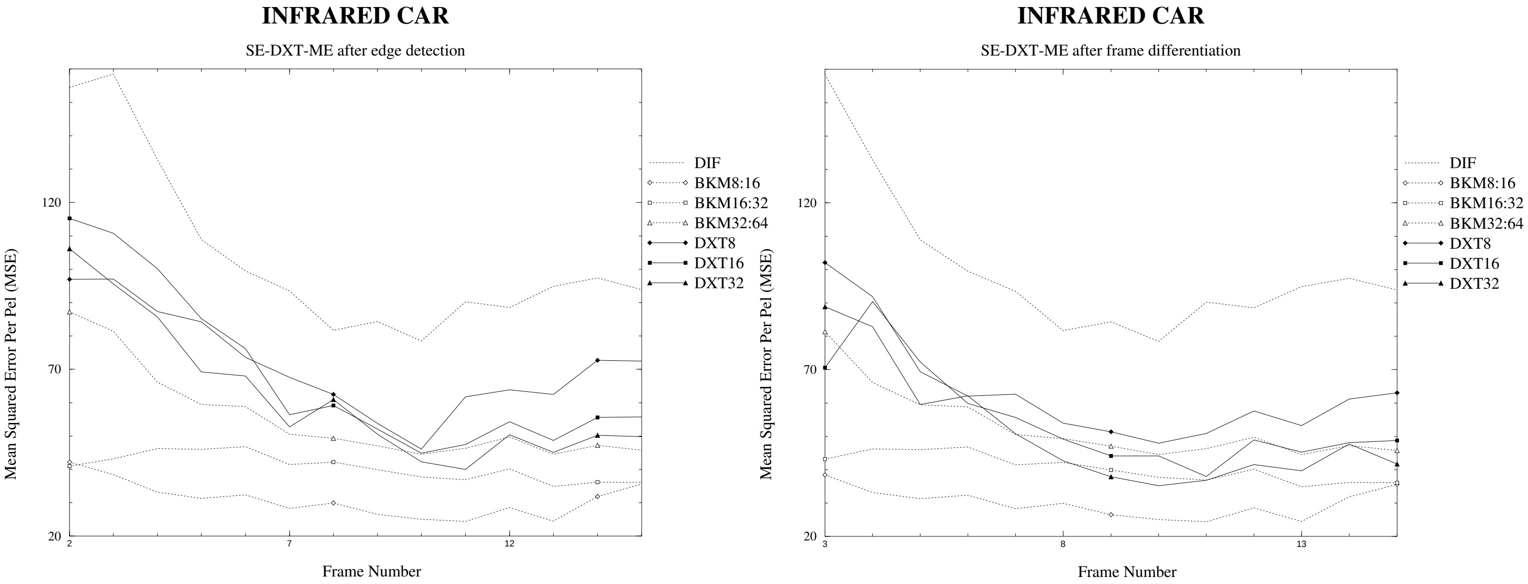
(a) Original (b) Edge extracted (c) Frame differentiated

Fig. 15. Sequence "Infrared Car" (CAR)

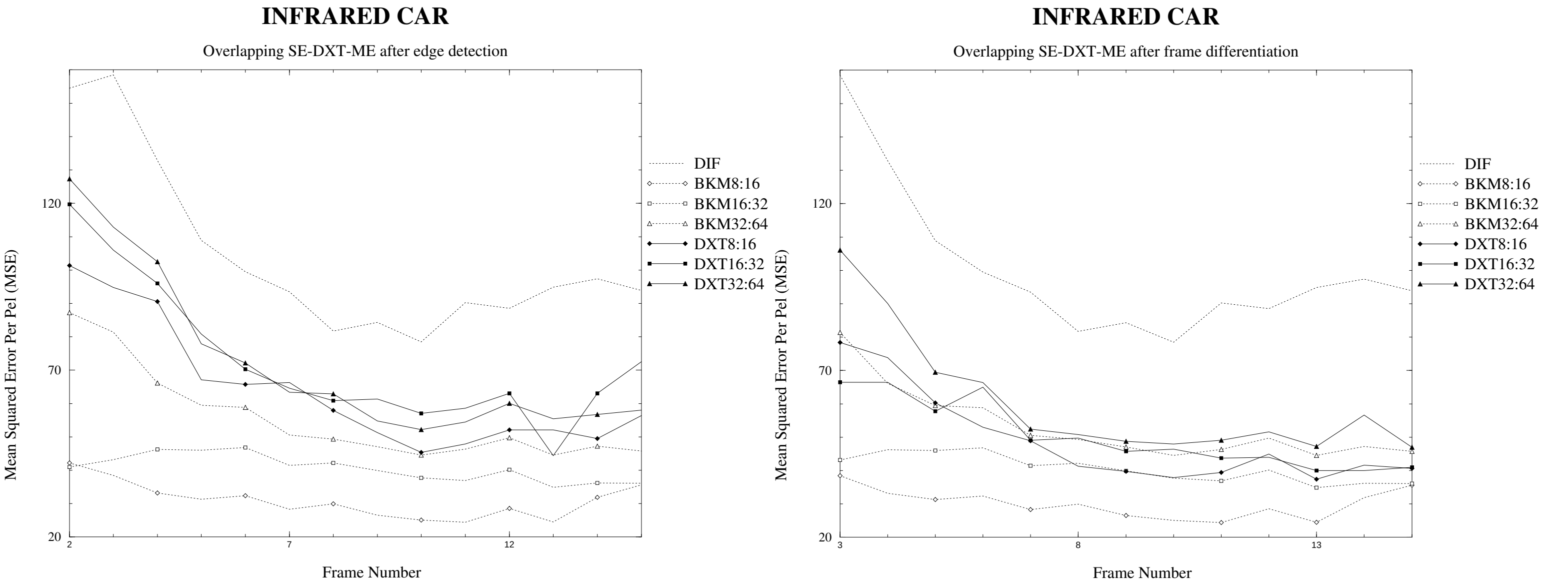
therefore, lower MSE values are obtained with this preprocessing function than with edge extraction. As can be seen in Fig. 16(a), the MSE value of SE-DXT-ME of block size 32 at Frame 9 is smaller than that of BMA-ME of search area 32. In other words, SE-DXT-ME is better than BMA-ME in this particular case. However, the results of Overlapping SE-DXT-ME in Fig. 16(b) show little improvement over SE-DXT-ME with fixed $\Phi = \{0, \ldots, N/2\}^2$, in contrary to the large gain in the performance of Overlapping SE-DXT-ME over SE-DXT-ME on the small "Flower Garden" sequence.

Simulation is also performed on the "Miss America" sequence, in which a lady is talking to the camera. Each frame has 352×288 pixels. This sequence has only little translational motion of the head and shoulders but mainly the mouth and eyes open and close. This makes the task of motion estimation difficult for this sequence but the DXT-ME algorithm can still perform reasonably well compared to the BMA-ME method, as can be found in Fig. 17. Especially for the Overlapping SE-DXT-ME algorithm with block size 32, the MSE values are very close to those of BMA-ME of different block size as shown in Fig. 17(b).

The last sequence is the small "Table Tennis" which has the frame size 352×224 pixels. In this sequence, the first twenty three frames contain solely a bouncing ball with rough texture of the wall behind. Thus, the edge extracted frames capture a lot of background features mixed with the ball features. This influences negatively the estimation of DXT-ME. However, the frame differentiated images have no such background features and as a result Overlapping SE-DXT-ME of block size 32 (DXT32:64) with frame differences as input performs much better than Overlapping SE-DXT-ME after edge extraction (see Fig. 18(b)) even though the ball is not moving in a constant speed and its residual energy after frame subtraction is weak. After the 23^{rd} frame, the camera is zooming out quickly, making



(a) Comparison of SE-DXT-ME with BMA-ME



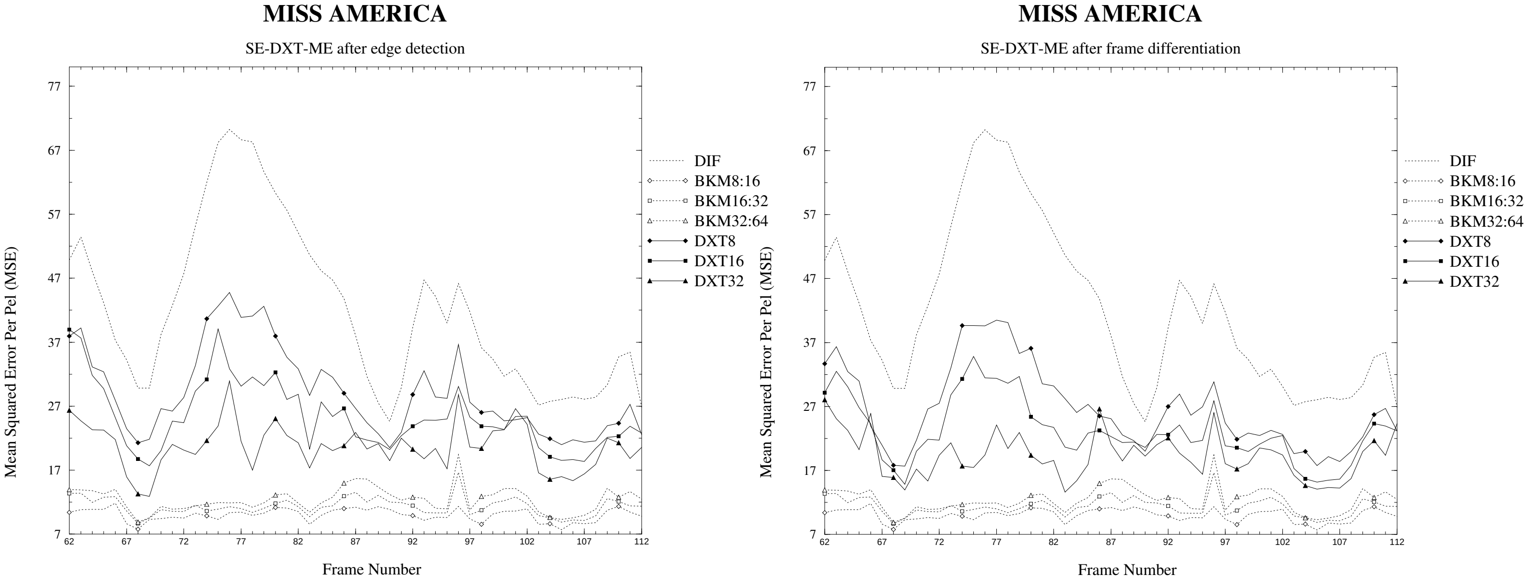
(b) Comparison of Overlapping SE-DXT-ME with BMA-ME

Fig. 16. Comparison of DXT-ME with BMA-ME on "Infrared Car"

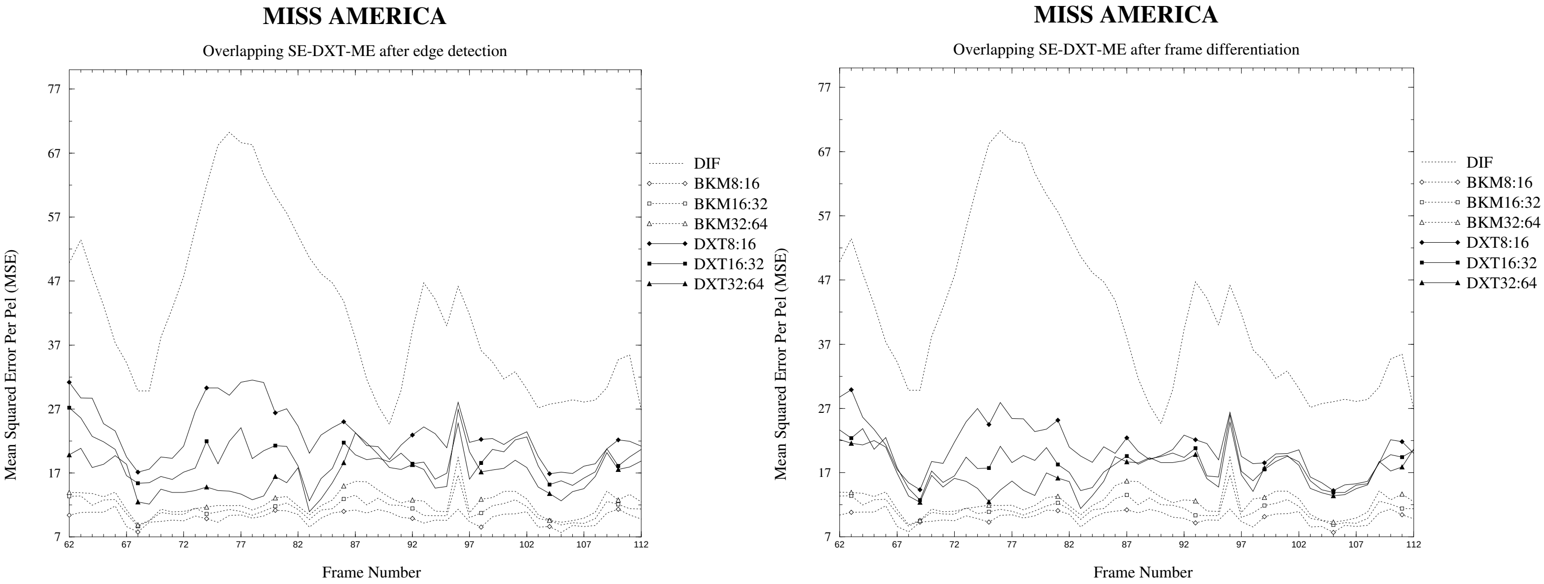
no method predict well. Then the zooming action slows down. In this situation, the MSE values of SE-DXT-ME go down suddenly to as low as those of BMA-ME in Fig. 18(a).

VI. Conclusion

In this paper, we presented the new motion estimation algorithm DXT-ME that computes the DCT pseudo-phases of images and employs the sinusoidal orthogonal principles to estimate motions in the transform domain. In this way, it can be incorporated into codecs of various image compression protocols like MPEG, CCITT H261, etc. and enables us to utilizes the advancement of DCT codecs which is



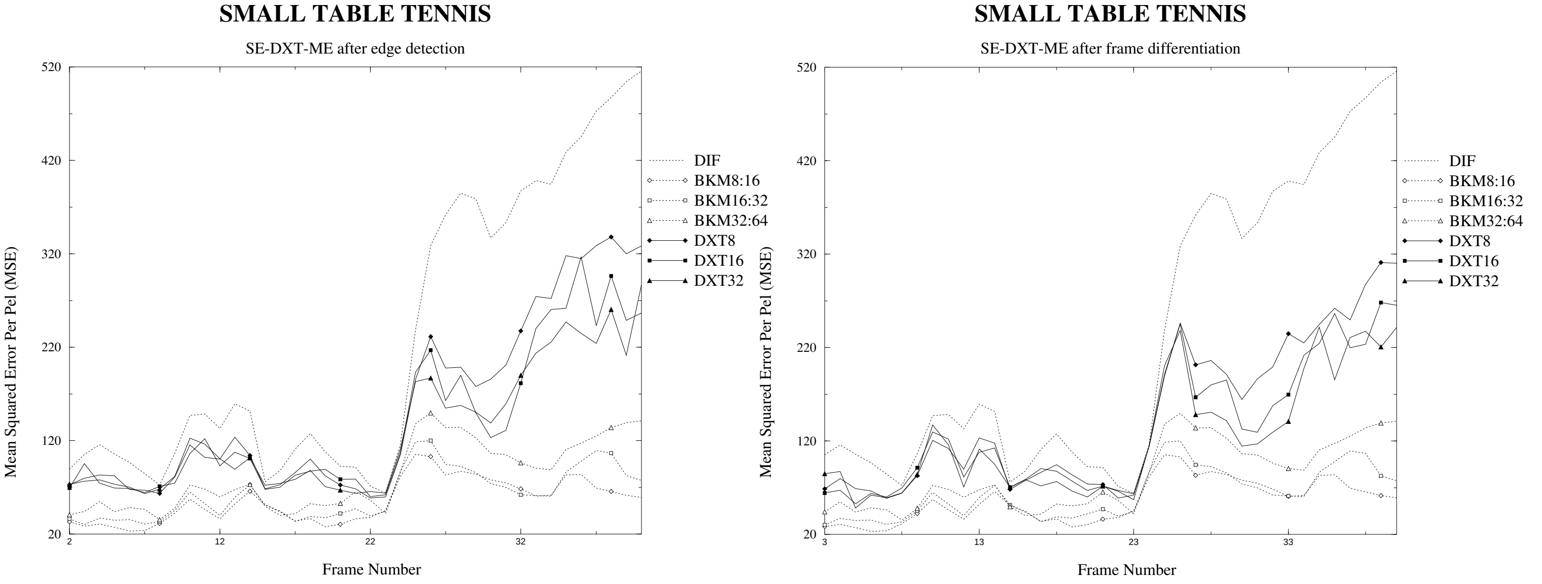
(a) Comparison of SE-DXT-ME with BMA-ME



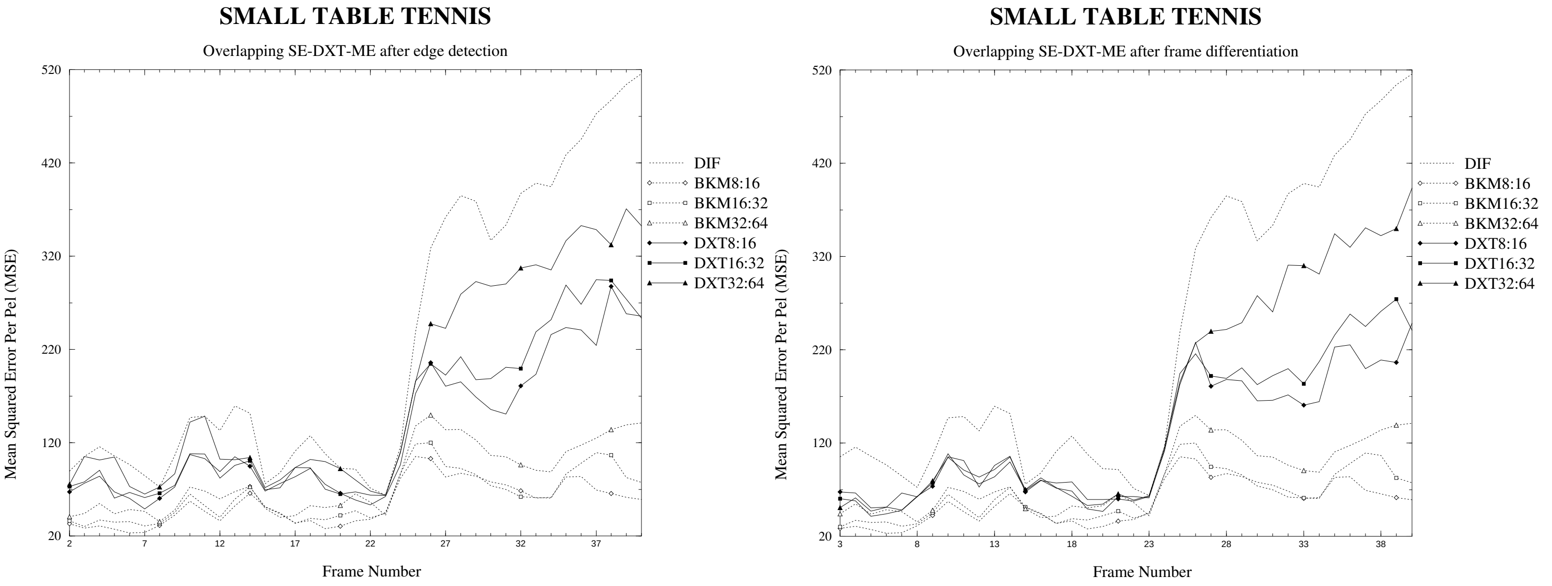
(b) Comparison of Overlapping SE-DXT-ME with BMA-ME

Fig. 17. Comparison of DXT-ME with BMA-ME on "Miss America"

under extensive research. In addition, motion estimation in the DCT domain enables us to remove the IDCT component in the loop and moves the DCT component out of the loop as explained in Section I. This not only reduces the coder complexity but at the same time increases the system throughput [22]. Furthermore, it requires much less computational complexity $O(N^2)$ as compared to $O(N^4)$ for BMA-ME and has very good noise immunity. Due to its feature matching property, it is possible to use simple preprocessing to enhance the features of moving objects in order to further improve the performance of DXT-ME, which is demonstrated to be comparably as good as that of BMA-ME in terms of MSE values according to the simulation results on a number of video sequences with different visual characteristics.



(a) Comparison of SE-DXT-ME with BMA-ME



(b) Comparison of Overlapping SE-DXT-ME with BMA-ME

Fig. 18. Comparison of DXT-ME with BMA-ME on small "Table Tennis"

Finally, this DXT-ME algorithm has inherently highly parallel operations and as a result it can easily be implemented on highly parallel array processors or dedicated circuits.

References

[1] J. K. Aggarwal and N. Nandhakumar, "On the computation of motion from sequences of images – a review", *Proceedings of the IEEE*, vol. 76, no. 8, pp. 917–935, August 1988.

[2] H. G. Musmann, P. Pirsch, and H.-J. Grallert, "Advances in picture coding", *Proceedings of the IEEE*, vol. 73, no. 4, pp. 523–548, April 1993.

[3] K. M. Yang, M. T. Sun, and L. Wu, "A family of VLSI designs for the motion compensation block-matching algorithm", *IEEE Trans. Circuits and Systems*, vol. 36, no. 10, pp. 1317–1325, October 1989.

[4] T. Komarek and P. Pirsch, "Array architectures for block-matching algorithms", *IEEE Trans. Circuits and Systems*,

vol. 36, no. 10, pp. 1301–1308, October 1989.
[5] L. D. Vos and M. Stegherr, "Parametrizable VLSI architectures for the full-search block-matching algorithm", *IEEE Trans. Circuits and Systems*, vol. 36, no. 10, pp. 1309–1316, October 1989.
[6] R. C. Kim and S. U. Lee, "A VLSI architecture for a pel recursive motion estimation algorithm", *IEEE Trans. Circuits and Systems*, vol. 36, no. 10, pp. 1291–1300, October 1989.
[7] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding", *IEEE Trans. Communications*, vol. COM-29, pp. 1799–1806, December 1981.
[8] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing", in *Proc. Nat. Telecom. Conf.*, New Orleans, LA, December 1981, pp. G5.3.1–G.5.3.5.
[9] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation", *IEEE Trans. Communications*, vol. COM-33, no. 8, pp. 888–896, August 1985.
[10] M. Ghanbari, "The cross-search algorithm for motion estimation", *IEEE Trans. Communications*, vol. 38, no. 7, pp. 950–953, July 1990.
[11] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 3, no. 2, pp. 148–157, April 1993.
[12] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438–442, August 1994.
[13] R. W. Young and N. G. Kingsbury, "Frequency-domain motion estimation using a complex lapped transform", *IEEE Trans. Image Processing*, vol. 2, no. 1, pp. 2–17, January 1993.
[14] A. N. Netravali and J. D. Robbins, "Motion compensated television coding – part 1", *Bell Syst. Tech. J.*, vol. 58, pp. 631–670, March 1979.
[15] J. D. Robbins and A. N. Netravali, "Recursive motion compensation: A review", in *Image Sequence Processing and Dynamic Scene Analysis*, T. S. Huang, Ed., pp. 76–103. Springer-Verlag, Berlin, Germany, 1983.
[16] A. Singh, *Optic Flow Computation – A Unified Perspective*, IEEE Computer Society Press, 1991.
[17] B. Porat and B. Friedlander, "A frequency domain algorithm for multiframe detection and estimation of dim targets", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 4, pp. 398–401, April 1990.
[18] A. Kojima, N. Sakurai, and J. Kishigami, "Motion detection using 3D-FFT spectrum", in *ICASSP-93*, Minnesota, April 1993, IEEE, vol. V, pp. V213–V216.
[19] D. Heeger, "A model for extraction of image flow", in *Proc. First Int'l Conf. Computer Vision*, London, 1987, pp. 181–190.
[20] CCITT Recommendation H.261, *Video Codec for Audiovisual Services at p × 64 kbit/s*, CCITT, August 1990.
[21] CCITT Recommendation MPEG-1, *Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s*, ISO/IEC 11172, Geneve Switzerland, 1993.
[22] H. Li, A. Lundmark, and R. Forchheimer, "Image sequence coding at very low bitrates: A review", *IEEE Trans. Image Processing*, vol. 3, no. 5, pp. 589–608, September 1994.
[23] J. S. McVeigh and S.-W. Wu, "Comparative study of partial closed-loop versus open-loop motion estimation for coding of HDTV", in *Proc. IEEE Workshop on Visual Signal Processing and Communications*, New Brunswick, September 1994, pp. 63–68.
[24] P. Yip and K. R. Rao, "On the shift property of DCT's and DST's", *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. ASSP-35, no. 3, pp. 404–406, March 1987.
[25] C. T. Chiu and K. J. R. Liu, "Real-time parallel and fully pipelined two-dimensional DCT lattice structures with applications to HDTV systems", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 2, no. 1, pp. 25–37, March 1992.
[26] K. J. R. Liu and C. T. Chiu, "Unified parallel lattice structures for time-recursive Discrete Cosine/Sine/Hartley transforms", *IEEE Trans. Signal Processing*, vol. 41, no. 3, pp. 1357–1377, March 1993.
[27] K. J. R. Liu, C. T. Chiu, R. K. Kologotla, and J. F. JaJa, "Optimal unified architectures for the real-time computation of time-recursive Discrete Sinusoidal Transforms", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 4, no. 2, pp. 168–180, April 1994.
[28] A. Zakhor and F. Lari, "Edge-based 3-d camera motion estimation with application to video coding", *IEEE Trans. Image Processing*, vol. 2, no. 4, pp. 481–498, October 1993.
[29] A. K. Jain, *Fundamental of Digital Image Processing*, Prentice-Hall, 1989.
[30] J. B. Burl, "A reduced order extended kalman filter for sequential images containing a moving object", *IEEE Trans. Image Processing*, vol. 2, no. 3, pp. 285–295, July 1993.