We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Kalman Filtering Based Motion Estimation for Video Coding

Assist. Prof. Nai-Chung Yang[1], Prof. Chaur Heh Hsieh[2] and Prof. Chung Ming Kuo[1]
[1]*Dept. of information Engineering, I-Shou University, Dahsu Kaohsiung, 840,*
[2]*Dept. of Computer and Communication Engineering, Ming Chuan University, Gui-Shan, Taoyuan, 333, Taiwan, R.O.C.*

1. Introduction

Video compression is a very efficient method for storage and transmission of digital video signal. The applications include multimedia transmission, teleconferencing, videophone, high-definition television (HDTV), CD-ROM storages, etc. The hybrid coding techniques based on predictive and transform coding are the most popular and adopted by many video coding standards such as MPEG-1/2/4 [1] and H.261/H.263/H.264 [2, 3], owing to its high compression efficiency. In the hybrid coding system, the motion compensation, first proposed by Netravali and Robbins in 1997, plays a key role from the view point of coding efficiency and implementation cost [4-11]. A generic hybrid video coder is depicted in Figure 1.

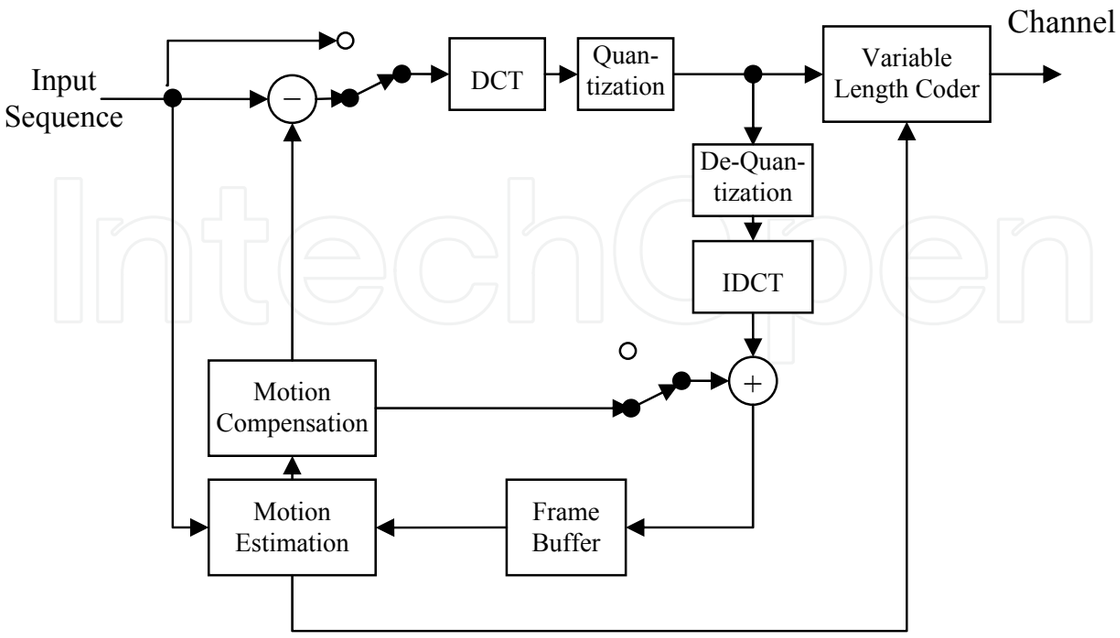


Fig. 1. A generic hybrid motion compensated DCT video coder.

Source: Kalman Filter: Recent Advances and Applications, Book edited by: Victor M. Moreno and Alberto Pigazo, ISBN 978-953-307-000-1, pp. 584, April 2009, I-Tech, Vienna, Austria

Open Access Database www.intechweb.org

The main idea of video compression to achieve compression is to remove spatial and temporal redundancies existing in video sequences. The temporal redundancy is usually removed by a motion compensated prediction scheme, whereas the spatial redundancy left in the prediction error is commonly reduced by a discrete cosine transform (DCT) coder. Motion compensated is a predictive technique in temporal direction, which compensates for the displacements of moving objects from the reference frame to the current frame. The displacement is obtained with the so-called motion vector estimation. Motion estimation obtains the motion compensated prediction by finding the motion vector (MV) between the reference frame and the current frame.

The most popular technique used for motion compensation (MC) is the block-matching algorithm (BMA) due to its simplicity and reasonable performance. In a typical BMA, the current frame of a video sequence is divided into non-overlapping square blocks of $N \times N$ pixels. For each reference block in the current frame, BMA searches for the best matched block within a search window of size $(2P+1) \times (2P+1)$ in the previous frame, where P stands for the maximum allowed displacement. Figure 2 depicts the basic principle of block matching.

In general, BMAs are affected by following factors: (i) search area, (ii) matching criterion, and (iii) searching scheme. The matching criterion is to measure the similarity between the block of the current frame and candidate block of the reference frame. Two typical matching criteria are mean square error (MSE) and mean absolute error (MAE), which are defined respectively as below:

$$MSE(u,v) = \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \left[f(x,y,k) - f(x+u, y+v, k-1) \right]^2 \text{ , for } u,v \in [-P,P]$$

$$MAE(u,v) = \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \left| f(x,y,k) - f(x+u, y+v, k-1) \right| \text{ , for } u,v \in [-P,P]$$

where $f(x,y,k)$ denotes the coordinate of the top left corner of the searching block of the current frame k, and (u,v) is the displacement of the matching block of frame $k-1$. The MAE is the most popular matching criterion due to its simplicity of hardware implementation.

The searching scheme is very important because it is significantly related to with the computational complexity and accuracy of motion estimation for general video applications. A straightforward way to obtain the motion vector is the full search algorithm (FSA), which searches all locations in the search window and selects the position with minimal matching error. However, its high computational complexity makes it often not suitable for real-time implementation. Therefore, many fast search algorithms have been developed to reduce the computational cost. In general, fast search algorithms reduce the computational burden by limiting the number of search locations or by sub-sampling the pixels of a block. However, they often converge to a local minimum, which leads to worse performance.

Most search algorithms estimate the motion vector (MV) for each block independently. In general moving scenes, it is very likely that a large homogeneous area in the picture frame will move in the same direction with similar velocities. Therefore, the displacements between neighboring blocks are highly correlated. Some schemes take advantage of this correlation to reduce the computational complexity [14-16].

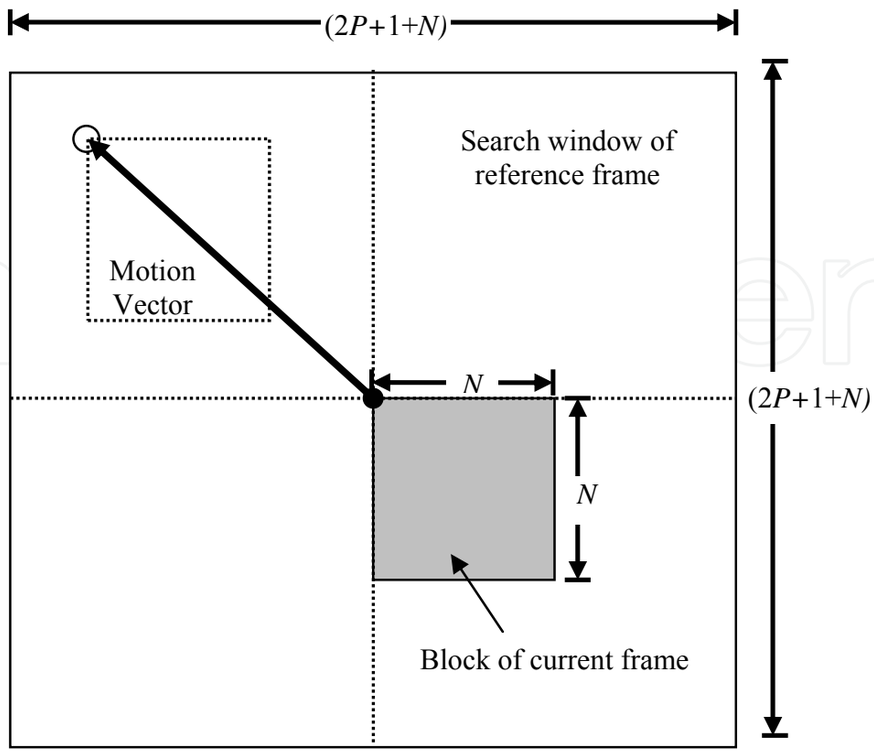


Fig. 2. Block matching algorithm.

There are two major problems for the existing fast search algorithms. One is that the estimation accuracy in terms of the energy or the entropy of the motion-compensated prediction error (MCPE) signal is worse than that of FSA. The other is that the true motion may not be obtained even with FSA, which is very important in some applications such as motion compensated interpolation and frame (field) rate conversion. Bierling [17] proposed a hierarchical search scheme to achieve a truer (smoother) motion vector field over FSA, but it results in a worse performance in terms of the energy of the MCPE signal.

The above two problems may arise from the following reasons [18]: (i) the basic assumptions, including pure translation, unchanged illumination in consecutive frames and noiseless environment, are not exactly correct; furthermore, another assumption that the occlusion of one object by another and uncovered background are neglected is also not exactly correct, (ii) the size of a moving object may not be equal to the prescribed block size, (iii) the fast search schemes often converge to a local optimum. In Section 2, we will introduce how to overcome these problems with a relatively low computational cost. We neither relax the above assumptions nor develop a globally optimal search scheme. Instead, we use the Kalman filter to compensate the incorrect and/or inaccurate estimates of motion. We first obtain a measurement of motion vector of a block by using a conventional fast search scheme. We then generate the predicted motion vector utilizing the motion correlation between spatial neighboring blocks. Based on the predicted and measured motion information, a Kalman filter is employed to obtain the optimal estimate of motion vector. In the new method, a local Kalman filter is developed, which is based on a novel

motion model that exploits both spatial and temporal motion correlations. The proposed local Kalman filter successfully addresses the difficulty of multi-dimensional state space representation, and thus it is simpler and more computationally efficient than the conventional 2-D Kalman filter such as reduced update Kalman filter (RUKF) [19]. In addition, we will also introduce an adaptive scheme to further improve estimate accuracy while without sending extra side information to the decoder.

In low- or very low- bit rate applications such as videoconference and videophone, the percentage of MV bit rate increases when overall rate budget decreases. Thus, the coding of MVs takes up a significant portion of the bandwidth [20]. Then in very low bit rate compression, the motion compensation must consider the assigned MV rate simultaneously. A joint rate and distortion (R-D) optimal motion estimation has been developed to achieve the trade-off between MV coding and residue coding [20-28]. In [25], a global optimum R-D motion estimation scheme is developed. The scheme achieves significant improvement of performance, but it employs Viterbi algorithm for optimization, which is very complicated and results in a significant time delay. In [26], a local optimum R-D motion estimation criterion was presented. It effectively reduces the complexity at the cost of performance degradation.

In Section 3, we will introduce two Kalman filter-based methods to improve the conventional R-D motion estimation, which are referred to as enhanced algorithm and embedded algorithm, respectively. In the enhanced algorithm, the Kalman filter is employed as a post processing of MV, which extends the integer-pixel accuracy of MV to fractional-pixel accuracy, thus enhancing the performance of motion compensation. Because the Kalman filter exists in both encoder and decoder, the method achieves higher compensation quality without increasing the bit rate for MV.

In the embedded algorithm, the Kalman filter is applied directly during the process of optimization of motion estimation. Since the R-D motion estimation consider compensation error (distortion) and bit rate simultaneously, when Kalman filter is applied the distortion will be reduced, and thus lowering the cost function. Therefore, the embedded algorithm can improve distortion and bit rate simultaneously. Specifically, this approach can be combined with existing advanced motion estimation algorithms such as overlapped block motion compensation (OBMC) [29,30], and those recommended in H.264 or MPEG-4 AVC [31, 32].

2. Motion estimation with Kalman filter

2.1 Review of Kalman filter

The Kalman filtering algorithm estimates the states of a system from noisy measurement [33-36]. There are two major features in Kalman filter. One is its mathematical formulation is described in terms of state-space representation, and the other is that its solution is computed recursively. It consists of two consecutive stages: prediction and updating. We summarize the Kaman filter algorithm as follows:

$$\text{Predicted equation: } \mathbf{v}(k) = \mathbf{\Phi}(k-1)\mathbf{v}(k-1) + \mathbf{\Gamma}(k)\mathbf{w}(k), \tag{1}$$

$$\text{Measurement equation: } \mathbf{z}(k) = \mathbf{H}(k)\mathbf{v}(k) + \mathbf{n}(k), \tag{2}$$

where $\mathbf{v}(k)$ and $\mathbf{z}(k)$ are state and measurement vector at time k, and H and are state transition, measurement and driving matrix, respectively. The model error $\mathbf{w}(k)$, with covariance matrix $\mathbf{Q}(k)$, and measurement error $\mathbf{n}(k)$, with covariance matrix $\mathbf{R}$(k), are often assumed to be Gaussian white noises; we assume that $\mathbf{w}$(k)~N(0,$\mathbf{Q}(k)$), $\mathbf{n}$(k)~N(0,$\mathbf{R}(k)$) and $E[\mathbf{w}(k)\mathbf{n}^T(l)]=0$ for all k and l. Let $E[\mathbf{v}(0)] = \hat{\mathbf{v}}(0)$, and $E[(\mathbf{v}(0) - \hat{\mathbf{v}}(0))(\mathbf{v}(0) - \hat{\mathbf{v}}(0))^T] = \mathbf{P}(0)$ be initial values. The prediction and updating are given as follows.

Prediction:

$$\text{State prediction: } \hat{\mathbf{v}}^-(k) = \mathbf{\Phi}(k-1)\hat{\mathbf{v}}^+(k-1) \tag{3}$$

Prediction-error covariance:

$$\mathbf{P}^-(k) = \mathbf{\Phi}(k-1)\mathbf{P}^+(k-1)\mathbf{\Phi}^T(k-1) + \mathbf{\Gamma}(k)\mathbf{Q}(k-1)\mathbf{\Gamma}^T(k) \tag{4}$$

Updating:

$$\text{State updating: } \hat{\mathbf{v}}^+(k) = \hat{\mathbf{v}}^-(k) + \mathbf{K}(k)[\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{v}}^-(k)] \tag{5}$$

$$\text{Updating-error covariance: } \mathbf{P}^+(k) = [\mathbf{I} - \mathbf{K}(k)\mathbf{H}(k)]\mathbf{P}^-(k) \tag{6}$$

$$\text{Kalman gain matrix: } \mathbf{K}(k) = \mathbf{P}^-(k)\mathbf{H}(k)[\mathbf{H}(k)\mathbf{P}^-(k)\mathbf{H}^T(k) + \mathbf{R}(k)]^{-1} \tag{7}$$

The $\mathbf{P}(k)$ is the error covariance matrix that is associated with the state estimate $\mathbf{v}$(k), and is defined as

$$\mathbf{P}(k) = E[(\mathbf{v}(k) - \hat{\mathbf{v}}(k))(\mathbf{v}(k) - \hat{\mathbf{v}}(k))^T]. \tag{8}$$

The superscripts "-" and "+" denote "before" and "after" measurement, respectively. The error covariance matrix $\mathbf{P}(k)$ provides a statistical measure of the uncertainty in $\mathbf{v}(k)$.

2.2 The overview of motion estimation with Kalman filter

In general, for moving scenes, the motion vectors among neighboring blocks are highly correlated. Therefore, the MV of the current block can be predicted from its neighboring blocks if an appropriate motion model is employed. Furthermore, any existing searching algorithms can be used to measure the MV. Using the predicted MV and the measured MV, a motion estimation method was developed, as depicted in Figure 3. The MV obtained with any conventional searching algorithm is defined as measurement, $\mathbf{z}(k)$. The measurement is then inputted to the Kalman filter and the updating estimate of MV could be obtained [37]. Because an identical Kalman filter will be used in the decoder, we can only send z(k), which is an integer, instead of $\hat{\mathbf{v}}(k)$, which is a real in general, to the receiver. By the same procedure, we can estimate $\hat{\mathbf{v}}(k)$ in the receiver, therefore we can achieve fractional-pixel accuracy with the bit rate of integer motion vector. In summary, there are two advantages for the new method: (i) it improves the performance of any conventional motion estimation due to the fractional-pixel accuracy; (ii) the transmitted bit rate for the motion vector is the

same as that of the input integer motion vector, therefore, the new method is compatible with the current video coding standards.

In the following, we will first introduce a motion model that exploits both spatial and temporal motion correlations, and then a local Kalman filter is developed accordingly. The local Kalman filter is simpler and more computationally efficient than the conventional 2-D Kalman filter such as RUKF. Therefore, it is more suitable for the real-time applications [48-49]. In addition, to further improve the motion estimate accuracy, we also introduce an adaptive scheme. The scheme can automatically adjust the uncertainty of prediction and measurement; however, it needs not to send side information to the decoder.

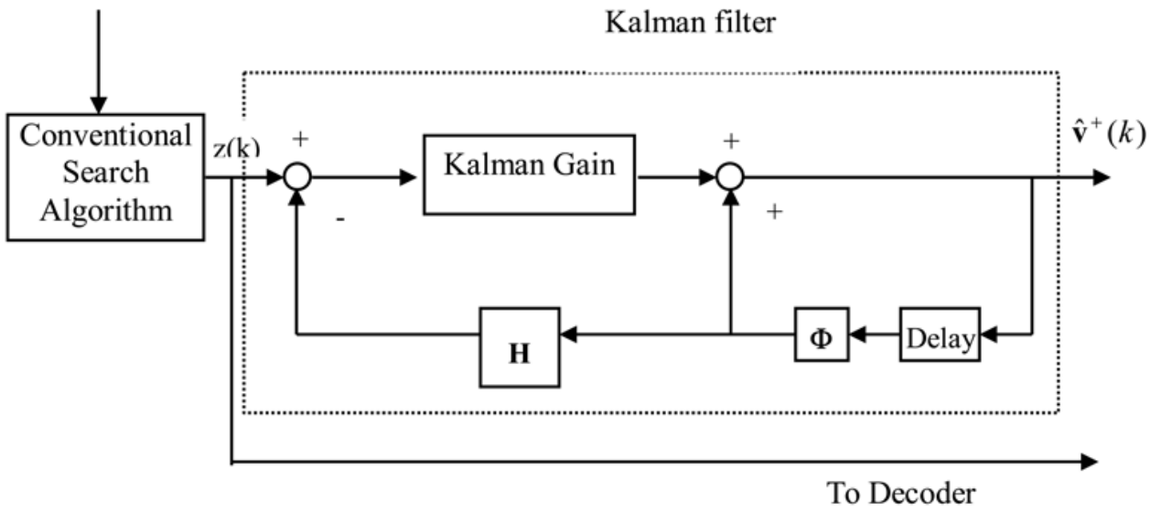


Fig. 3. Block diagram of motion estimation with Kalman filter

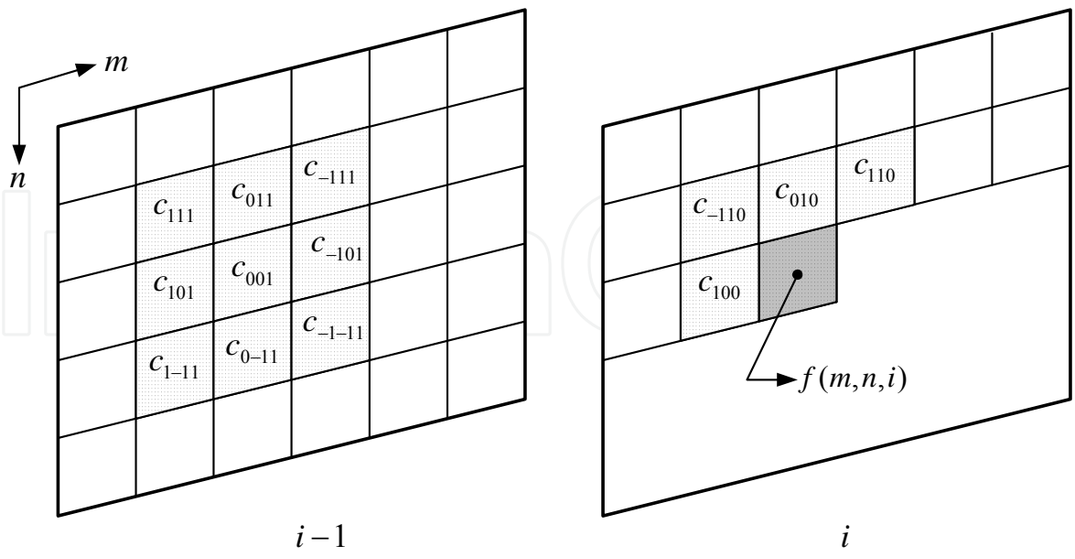
2.3 Motion estimation using Local Kalman Filter (LKF)

Let B(m,n,i) be the block at the location (m,n) in the ith frame, and $\mathbf{V}(m,n,i)=[v_x(m,n,i),v_y(m,n,i)]^T$ be the MV of B(m,n,i), where $v_x(m,n,i)$ and $v_y(m,n,i)$ denote the horizontal and vertical components, respectively. Assume that the MV is a random process, and the two components are independent. Then we can model these two components separately. In this work, we present a three-dimensional (3-D) AR model that exploits the relationship of motion vectors for only 3-D neighboring blocks that arrive at before the current block. We only choose the nearest neighboring blocks, in which the motion vectors are strongly correlated. We refer to this model as 3-D local model, which is expressed as

$$v_x(m,n,i)=\sum\sum\sum_{(k,l,p)\in S^{\oplus}} a_{klp}v_x(m-k,n-l,i-p)+w_x(m,n,i), \tag{9}$$

$$v_y(m,n,i)=\sum\sum\sum_{(k,l,p)\in S^{\oplus}} a_{klp}v_y(m-k,n-l,i-p)+w_y(m,n,i), \tag{10}$$

where $S^{\oplus}=\{l=0,k=1,p=0\}\cup\{l=1,|k|\le 1,p=0\}\cup\{|l|\le 1,|k|\le 1,p=1\}$. The support of the model mentioned above is depicted in Figure 4.



a) b)

Fig. 4. Causal AR models for motion vector associated with spatial and temporal neighborhood system.

2.3.1 State space representation of MV model

For the fully state propagation, we must represent the proposed models of Eqs. (9) and (10) in a state space. This will yield a 13-dimensional state vector. The high-dimension state vector will result in a huge computation for estimating the motion vector. To attack the computation problem, we decompose the AR model into two parts: filtering and prediction. The prediction part will not affect the state propagation; thus it is considered as a deterministic input. Consequently, the state space representation can be formulated as

$$\mathbf{v}(m,n,i) = \Phi\mathbf{v}(m-1,n,i) + \Lambda\mathbf{u}(m,n,i) + \Gamma\mathbf{w}(m,n,i)\,, \tag{11}$$

$$\mathbf{z}(m,n,i) = \mathrm{H}\mathbf{v}(m,n,i) + \mathbf{e}(m,n,i)\,. \tag{12}$$

In the above equations, $\mathbf{v}(m,n,i)$ represents the state vector at the location (m,n,i); $\mathrm{u}(m,n,i)$ denotes a deterministic input; and $\mathbf{\Phi}$, $\mathbf{\Lambda}$, $\mathbf{\Gamma}$ and $\mathbf{H}$ are the corresponding matrices. In our work, the deterministic input is defined as the prediction part of the model, which will be used to implement the local Kalman filter (LKF). Since the motion estimation processes the block one by one according to the order of raster scan, the state propagation should be performed in one-dimensional manner, as depicted in Eq. (11).

The main idea in LKF is the approximation of the MV $\mathbf{v}(m,n,i)$, which can not be represented in terms of $\mathbf{v}(m\text{-}1,n,i)$. We will demonstrate the state space representation in Eqs. (13) and (14) as follows.

$$\mathbf{v}(m,n,i) = \Phi\mathbf{v}(m-1,n,i) + \Lambda\mathbf{u}(m,n,i) + \Gamma\mathbf{w}(m,n,i) \tag{13}$$

where

$$\mathbf{v}(m,n,i)=\begin{bmatrix} v(m,n,i) \\ v(m-1,n,i) \\ v(m+2,n-1,i) \\ v(m+1,n-1,i) \\ v(m,n-1,i) \\ v(m+1,n,i-1) \end{bmatrix}, \mathbf{v}(m-1,n,i)=\begin{bmatrix} v(m-1,n,i) \\ v(m-2,n,i) \\ v(m-1,n-1,i) \\ v(m,n-1,i) \\ v(m-1,n,i) \\ v(m,n,i-1) \end{bmatrix}, \mathbf{u}(m,n,i)=\begin{bmatrix} v(m+1,n-1,i-1) \\ v(m,n-1,i-1) \\ v(m-1,n-1,i-1) \\ v(m+1,n,i-1) \\ v(m-1,n,i-1) \\ v(m+1,n+1,i-1) \\ v(m,n+1,i-1) \\ v(m-1,n+1,i-1) \end{bmatrix},$$

$$\mathbf{w}(m,n,i)=\begin{bmatrix} w(m,n,i) \\ w(m+1,n,i-1) \end{bmatrix},$$

$$\mathbf{\Phi}=\begin{bmatrix} c_{100} & 0 & c_{-110} & c_{010} & c_{110} & c_{001} \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \mathbf{\Lambda}=\begin{bmatrix} c_{-111} & c_{011} & c_{111} & c_{-101} & c_{101} & c_{-1-11} & c_{0-11} & c_{1-11} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and $\mathbf{\Gamma}=\begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$.

$$\mathbf{z}(m,n,i)=\mathbf{H}\mathbf{v}(m,n,i)+\mathbf{e}(m,n,i) \tag{14}$$

where $\mathbf{z}(m,n,i)=\begin{bmatrix} z(m,n,i) \\ z(m+2,n-1,i) \end{bmatrix}$, $\mathbf{H}=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$ and $\mathbf{e}(m,n,i)=\begin{bmatrix} e(m,n,i) \\ e(m+2,n-1,i) \end{bmatrix}$.

The motion vector $\mathbf{v}(m,n,i)$, which can not be represented in terms of $\mathbf{v}(m\text{-}1,n,i)$, consists of two components: $v(m\text{+}2,n\text{-}1,i)$ and $v(m\text{+}1,n,i\text{-}1)$. We use the most recent estimate with uncertainty to approximate them, i.e.,

$$v(m+2,n-1,i)=\hat{v}(m+2,n-1,i)+e(m+2,n-1,i), \tag{15}$$

$$v(m+1,n,i-1)=\hat{v}(m+1,n,i-1)+w(m+1,n,i-1), \tag{16}$$

The above equations indicate that the best available estimate is the most recent update of the MV, which is available at time (m,n,i). The current frame MV, $v(m\text{+}2,n\text{-}1,i)$, is incorporated into measurement, and the previous frame MV, $v(m\text{+}1,n,i\text{ -}1)$, is incorporated into deterministic input. In our work, the covariance of these two uncertainties is given a small

value for simplicity. Through the above process, the motion estimation with 3-D AR model can be realized by 1-D recursive manner.
Given these models, the Kalman filter is described in the following:
<1> Prediction

$$\text{State prediction: } \hat{\mathbf{v}}^{-}(m,n,i) = \Phi\hat{\mathbf{v}}^{+}(m-1,n,i) + \Lambda\mathbf{u}(m,n,i) \tag{17}$$

$$\text{Prediction-error covariance: } \mathbf{P}^{-}(m,n,i) = \Phi\mathbf{P}^{+}(m-1,n,i)\Phi^{T} + \Gamma\,\mathbf{Q}(m,n,i)\Gamma^{T} \tag{18}$$

<2> Updating:

$$\text{State updating: } \hat{\mathbf{v}}^{+}(m,n,i) = \hat{\mathbf{v}}^{-}(m,n,i) + \mathbf{K}(m,n,i)[\mathbf{z}(m,n,i) - \mathbf{H}\hat{\mathbf{v}}^{-}(m,n,i)] \tag{19}$$

$$\text{Updating-error covariance: } \mathbf{P}^{+}(m,n,i) = [\mathbf{I} - \mathbf{K}(m,n,i)\mathbf{H}]\mathbf{P}^{-}(m,n,i) \tag{20}$$

$$\text{Kalman gain matrix: } \mathbf{K}(m,n,i) = \mathbf{P}^{-}(m,n,i)\mathbf{H}^{T}[\mathbf{H}\mathbf{P}^{-}(m,n,i)\mathbf{H}^{T} + \mathbf{R}(m,n,i)]^{-1} \tag{21}$$

The $\mathbf{P}(m,n,i)$ is the error covariance matrix that is associated with the state estimate $\mathbf{v}(m,n,i)$, $\mathbf{R}(m,n,i)$ and $\mathbf{Q}(m,n,i)$ are the covariance of $\mathbf{e}(m,n,i)$ and $\mathbf{w}(m,n,i)$, respectively.
However, the local model can be simplified to consider only spatial or temporal support, and then the motion model and the corresponding state space representation are modified accordingly.

2.3.2 Spatial causal AR models for MV

Let $B(m,n,i)$ be the block at the location (m,n) in the ith frame, and $\mathbf{V}(m,n,i)=[v_x(m,n,i),v_y(m,n,i)]^T$ be the MV of $B(m,n,i)$, where $v_x(m,n,i)$ and $v_y(m,n,i)$ denote the horizontal and vertical components, respectively. Assume that the MV is a random process, and the two components are independent. A 2-D AR model exploits the motion information of only 2-D neighboring blocks that arrived before the current block. In the block matching, the calculation of matching criterion is performed block-by-block in a raster scan manner, i.e., from left to right and top to bottom. Thus we can define the 2-D AR model for a motion vector as

$$v_{i,x} = \sum_{(k,l)\in S^{+}} \sum a_{kl0} v_{i,x}(m-k,n-l,i) + w_{i,x}(m,n,i)\,, \tag{22}$$

$$v_{i,y} = \sum_{(k,l)\in S^{+}} \sum a_{kl0} v_{i,y}(m-k,n-l,i) + w_{i,y}(m,n,i)\,, \tag{23}$$

where $S^{+} = \{k \geq l,\ \forall l\} \cup \{k = 0,\ l \geq 1\}$ is the model support, and a_{kl0} are the model coefficients, which can be space varying or space invariant. For simplicity, we assume that the model is space invariant. Eq. (22) and (23) are also called the nonsymmetric half-plane (NSHP) model [19].
We only chose the nearest neighboring blocks in both horizontal and vertical direction because their motions are strongly correlated. We call this model as 2-D local motion model. In such case, Eq. (22) and (23) can be simplified as

$$v_{i,x}(m,n,i) = a_{100}v_{i,x}(m-1,n,i) + a_{-110}v_{i,x}(m+1,n-1,i) + a_{010}v_{i,x}(m,n-1,i) + a_{110}v_{i,x}(m-1,n-1,i) + w_{i,x}(m,n,i), \quad (24)$$

$$v_{i,y}(m,n,i) = a_{100}v_{i,y}(m-1,n,i) + a_{-110}v_{i,y}(m+1,n-1,i) + a_{010}v_{i,y}(m,n-1,i) + a_{110}v_{i,y}(m-1,n-1,i) + w_{i,y}(m,n,i). \quad (25)$$

The support of the model mentioned above is depicted in Figure 5.

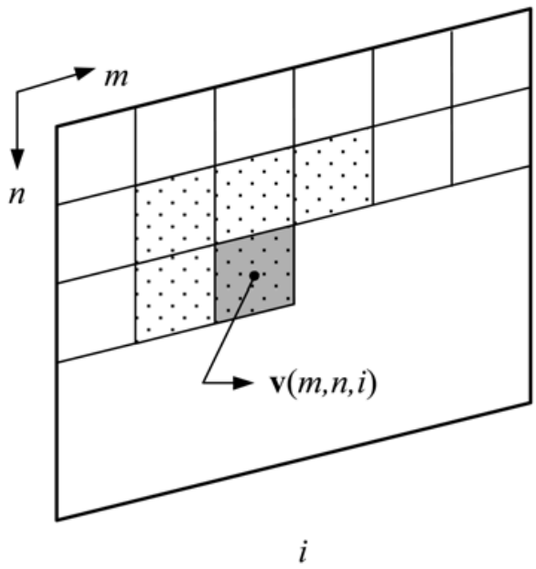


Fig. 5. Causal AR models for motion vector associated with spatial neighboring blocks.

2.3.3 State space representation of spatial local AR model

For the full state propagation, we must represent the proposed models, Eq. (11) and (12), in a state space. Since the Kalman filter is implemented by one-dimensional recursion, it is very difficult to transfer the two-dimensional AR model into one-dimensional state space representation [39,40]. To attack this problem, we introduce an extra deterministic input into the conventional state-space equations, and then we have the state-space representation as follows.

Predicted equation:

$$\mathbf{v}(m,n,i) = \Phi\mathbf{v}(m-1,n,i) + \Lambda\mathbf{u}(m,n,i) + \Gamma\mathbf{w}(m,n,i)\,, \quad (26)$$

where $\mathbf{v}(m,n,i)$ represents the state vector at the location (m,n,i); $\mathbf{u}(m,n,i)$ is the introduced deterministic input; and $\mathbf{\Phi}$, $\mathbf{\Lambda}$, $\mathbf{\Gamma}$ and $\mathbf{H}$ are the corresponding matrices. They are respectively defined as

$$\mathbf{v}(m,n,i)=\begin{bmatrix} v(m,n,i) \\ v(m-1,n,i) \\ v(m+2,n-1,i) \\ v(m+1,n-1,i) \\ v(m,n-1,i) \end{bmatrix}, \mathbf{v}(m-1,n,i)=\begin{bmatrix} v(m-1,n,i) \\ v(m-2,n,i) \\ v(m+1,n-1,i) \\ v(m,n-1,i) \\ v(m-1,n-1,i) \end{bmatrix},$$

$$\mathbf{u}(m,n,i)=\hat{v}(m+2,n-1,i), \quad \mathbf{w}(m,n)=\begin{bmatrix} w(m,n,i) \\ w(m+2,n-1,i) \end{bmatrix},$$

$$\mathbf{\Phi}=\begin{bmatrix} a_{100} & 0 & a_{-110} & a_{010} & a_{110} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \mathbf{\Lambda}=\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \text{and } \mathbf{\Gamma}=\begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Measurement equation:

$$\mathrm{z}(m,n,i)=\mathbf{H}\ \mathbf{v}(m,n,i)+e(m,n,i), \tag{27}$$

where $\mathbf{H}=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}$.

Because the element $v(m+2,n-1,i)$ of the motion vector $\mathbf{v}(m,n,i)$ can not be written in terms of its previous state, here we use the most recent estimate to approximate it, i.e.,

$$v(m+2,n-1,i)=\hat{v}(m+2,n-1,i)+w(m+2,n-1,i). \tag{28}$$

The above equations indicate that the best available estimate is the most recent update of the MV, which is available at time (m,n,i). Through the above process, the motion estimation based on 2-D AR model can be realized by 1-D recursive manner.

2.3.4 Temporal causal AR models for MV

Using the similar definition of the above spatial model, the AR models in the temporal direction are defined as

$$v_x(m,n,i)=\sum\sum\sum_{(k,l,p)\in S^{\oplus}} a_{klp}\, v_x(m-k,n-l,i-p)+w_x(m,n,i), \tag{29}$$

$$v_y(m,n,i)=\sum\sum\sum_{(k,l,p)\in S^{\oplus}} a_{klp}\, v_y(m-k,n-l,i-p)+w_y(m,n,i), \tag{30}$$

where $S^{\oplus}=\{|l|\le 1,\ |k|\le 1,\ p=1\}$. Like the spatial local model, only the adjacent neighboring blocks are considered as model support, as shown in Figure 6. In such case, the state-space representation of Eq. (29) and (30) are

Predicted equation:

$$v(m,n,i) = a_{001}v(m,n,i-1) + \mathbf{\Lambda}\begin{bmatrix} v(m+1,n-1,i-1) \\ v(m,n-1,i-1) \\ v(m-1,n-1,i-1) \\ v(m+1,n,i-1) \\ v(m-1,n,i-1) \\ v(m+1,n+1,i-1) \\ v(m,n+1,i-1) \\ v(m-1,n+1,i-1) \end{bmatrix} + w(m,n,i)\,, \tag{31}$$

where $\mathbf{\Lambda} = \begin{bmatrix} a_{-111} & a_{011} & a_{111} & a_{-101} & a_{101} & a_{-1-11} & a_{0-11} & a_{1-11} \end{bmatrix}$.

Measurement equation:

$$z(m,n,i) = v(m,n,i) + e(m,n,i)\,. \tag{32}$$

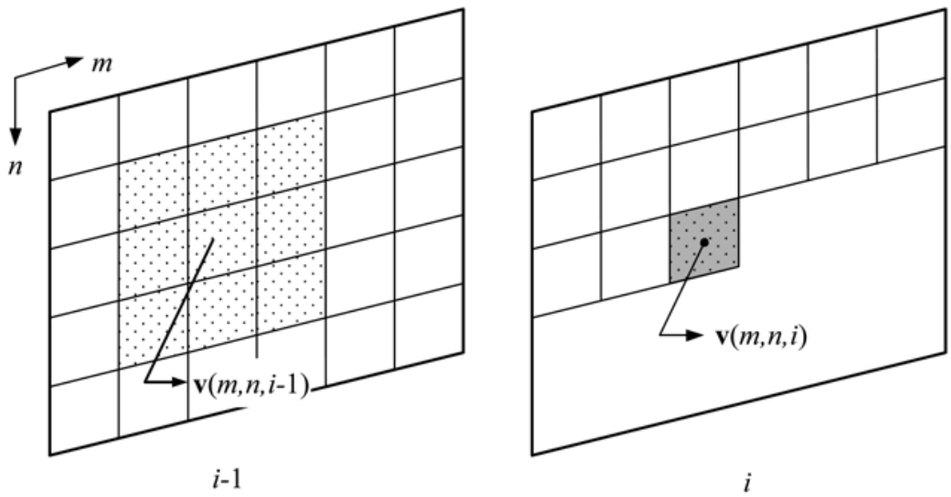


Fig. 6. Causal AR models for motion vector associated with temporal neighboring blocks.

Once the motion models and their state space representation are available, same procedure can then be obtained as in Section 2.3.1.

2.4 Adaptive Kalman filtering

In general, the motion correlation between the adjacent blocks cannot be modeled exactly. Similarly, the measurement of motion vector may have error due to incorrect, inaccurate and low precision estimation algorithms. Therefore, there exist uncertainties in both prediction and measurement processes. The uncertainties of prediction and measurement are represented by zero mean white Gaussian noise **w**(*m,n,i*) and **e**(*m,n,i*) with variance **q**(*m,n,i*) and **r**(*m,n,i*), respectively. In Kalman filtering algorithm, the Kalman gain depends on **q**(*m,n,i*) and **r**(*m,n,i*); therefore, the variances will determine the relative amount of updating using prediction or measurement information [41]. Due to the nonstationary nature of motion vector fields, the values of the variances **q**(*m,n,i*) and **r**(*m,n,i*) should be adjusted block by block to achieve better performance.

In [37], we introduced a distortion function, D_1 and D_2, to measure the uncertainty for both prediction and measurement, and use the distortion function as a reliability criterion. Based on such concept, we calculate the covariance $\mathbf{q}(m,n,i)$ and $\mathbf{r}(m,n,i)$ that are closely related to D_1 and D_2, and then obtain a time-varying Kalman gain. This results in more reliable estimate of MV. The idea behind the procedure is that we use the actual distortion of prediction and measurement to adjust the covariance instead of the conventional complex statistical on-line approaches [37]. Because the distortion measured is more trustworthy than any assumption, the developed scheme achieves very good performance as demonstrated in [37]. The major disadvantages of the scheme are: (i) it needs to send extra side-information, (ii) it increases the overall bit rate, and (iii) the bit stream may be not compatible with the current video coding standard. To overcome this problem, we will introduce an adaptive scheme, which is simpler and more effective than the previous schemes and it does not need to send extra side-information.

We first calculate the errors compensated by predicted MV and measured MV. And then investigate the relation between the difference of the two errors, Δd , and the difference of two motion vectors, ΔMV . Let D_1 and D_2 be the block distortion of motion compensation due to the measurement error and prediction error, respectively, which are defined as

$$\begin{aligned} D_1 &= \frac{1}{M \times N} \sum_{j=0}^{N-1} \sum_{l=0}^{N-1} \left| B\left(m+j, n+l, i\right) - \tilde{B}\left(m+j+z(m,n,i), n+l+z(m,n,i), i-1\right) \right| \\ &= MAD\left(z_x, z_y\right) \end{aligned}, \tag{33}$$

and

$$\begin{aligned} D_2 &= \frac{1}{M \times N} \sum_{j=0}^{N-1} \sum_{l=0}^{N-1} \left| B\left(m+j, n+l, i\right) - \tilde{B}\left(m+j+\hat{v}_x^-(m,n,i), n+l+\hat{v}_y^-(m,n,i), i-1\right) \right| \\ &= MAD\left(\hat{v}_x^-, \hat{v}_y^-\right) \end{aligned}, \tag{34}$$

where B_i and B_{i-1} are the current block and motion compensated block, respectively. The Δd is defined as

$$\Delta d = |\ D_1\text{-}D_2\ |\ . \tag{35}$$

When ΔMV increases, Δd first increased approximately exponentially, and then decreased exponentially; the increasing rate is larger than the decreasing rate. In general, the measurement is obtained by a real matching; thus a large ΔMV means that the prediction is far away from optimal location. This results in a large value of Δd . However, when ΔMV exceeds a certain value, the measurement may find incorrect position due to the restrictions of block matching, such as cover/uncover-background, complex motion types, etc. Hence Δd will decrease gradually according to the increase of ΔMV . Therefore, we can use two exponential functions to model the variance of prediction as:

$$q(m,n,i) = \begin{cases} 1 - a_1 \exp(-b_1 \left\| \mathbf{z}(m,n,i) - \hat{\mathbf{v}}^-(m,n,i) \right\|), & \left\| \mathbf{z} - \hat{\mathbf{v}}^- \right\| \le th \\ a_2 \exp(-b_2 (\left\| \mathbf{z}(m,n,i) - \hat{\mathbf{v}}^-(m,n,i) \right\| - th)), & \left\| \mathbf{z} - \hat{\mathbf{v}}^- \right\| > th \end{cases} \tag{36}$$

$$r(m,n,i) = 1\text{-}q(m,n,i), \tag{37}$$

where *th* is the turning point, which is a reliable index of measurement. If ΔMV is less than *th,* the measurement is reliable compared with prediction. However, when ΔMV is far away from *th*, the measurement is less reliable and the prediction should give more contribution. The parameters *a* and *b* affect the shape of the exponential function and are related with searching methods. The parameter values for full search are larger than those for fast search. Because the prediction can be calculated in the receiver, no extra-information needs to be sent. Therefore, this method is also suitable for real-time application.

2.5 Simulation results

Several image sequences including "Miss America", "Salesman", "Flower Garden" and "Susie" are evaluated to compare the performances of different motion estimation algorithms. The first two sequences are typical videoconference situations. In order to create larger motion displacement, each of the two sequences is reduced to 15 Hz with frame skipping. The last two sequences contain more complex motion such as rotation, and covered/uncovered background. They are converted from CCIR 601 format using the decimation filters recommended by the ISO/MPEG standard committee. The 30 successive frames of each sequence are used in simulation.

Four different algorithms are compared: (i) full search algorithm (FSA), (ii) new three-step algorithm (NTSS), (iii) NTSS combined with 3-D Kalman filter (3DLKF), and (iv) NTSS combined with adaptive Kalman filter (3DALKF). The size of the image block is 16×16. The search window is 15×15 pixels (i.e., S = 7) for "Miss America", "Salesman" and "Susie", 31×31(S = 15) for "Flower Garden". The threshold for the motion detection is 2 for each algorithm. The model parameters are obtained by off-line least-squared estimate. In our work, the parameters are given by c_{100}=7/C, c_{-110}=2/C, c_{010}=7/C, c_{110}=2/C, c_{001}=5/C, c_{-111}=0.25/C, c_{011}=0.5/C, c_{111}=0.25/C, c_{-101}=0.5/C, c_{101}=0.5/C, c_{-1-11}=0.25/C, c_{0-11}=0.5/C, and c_{1-11}=0.25/C. Where C is a normalization factor, and C=26 in our simulation. For non-adaptive algorithm, the covariance of $\mathbf{w}(m,n,i)$ and $\mathbf{e}(m,n,i)$ should be given *a priori*. In this work, the $q(m+2,n-1,i)$ and $r(m+1,n,i-1)$ are 0.095, $q(m,n,i)$ and $r(m,n,i)$ are 0.85 and 0.15, respectively. In the adaptive algorithm, $q(m,n,i)$ and $r(m,n,i)$ are adjusted automatically, the parameters *a*, *b* and *th* are set as a_1=0.55, a_2=1.10, b_1=0.985, b_2=0.009 and th=5.8 for "Flower Garden", a_1=1.10, a_2=0.98, b_1=0.735, b_2=0.008 and th=4.2 for others sequences. The value is obtained experimentally. The $q(m+2,n-1,i)$ and $r(m+1,n,i-1)$ are the same as the non-adaptive algorithm.

The motion-compensated prediction frame is obtained by displacing the previous reconstructed blocks using the estimated motion vectors. Since the estimated motion vector is a real value instead of an integer, the displaced pixels may not be on the sampling grid. Therefore, the well-known bilinear interpolation [17] is adopted to generate a motion compensated prediction frame.

Table 1 summarizes the comparison of the average PSNRs for various algorithms. It indicates that all our algorithms perform better than NTSS. The 3DLKF also obtains better performance than FSA on the average. It is noted that the 3DLKF needs few additional computations over NTSS. The 3DALKF give much better PSNR performance than FSA.

Figures 7-10 displays the comparison of PSNR of the test sequences obtained by various algorithms. It indicates that the proposed method improves the performance. The most

important point to note is that the adaptive algorithm can compensate poor measurement and thereby raise the PSNR significantly. In addition, the visual quality of the reconstructed image is also improved considerably. This can be seen from Figure 11, which shows the reconstructed images of frame 74 obtained by NTSS and 3DALKF, respectively. The NTSS algorithm yields the obvious distortion on some regions such as the left ear and the mouth, as shown in Figure 11 (a). The 3DLKF algorithm, as shown in Figure 11 (c), improves this significantly.

Image Sequence	Algorithm			
	NTSS	FSA	3DLKF	3DALKF
Miss America	38.2581	38.3956	38.6473	38.9077
Salesmen	34.6905	34.7827	34.9477	35.1080
Susie	37.8381	37.8742	38.2893	38.5298
Flower Garden	28.2516	28.4485	28.3836	28.5340
Average	34.7596	34.8753	35.067	35.2699

Table 1. Average PSNR for various algorithms

Figure 12 shows the motion vector fields of "Miss America" obtained by FSA, NTSS and 3DALKF, respectively. The motion vector fields obtained by 3DLKF algorithm are obviously smoother than those by the other algorithms. Although the hierarchical search algorithm presented in [17] can also achieve smooth motion vector fields, it obtains lower PSNR than FSA.

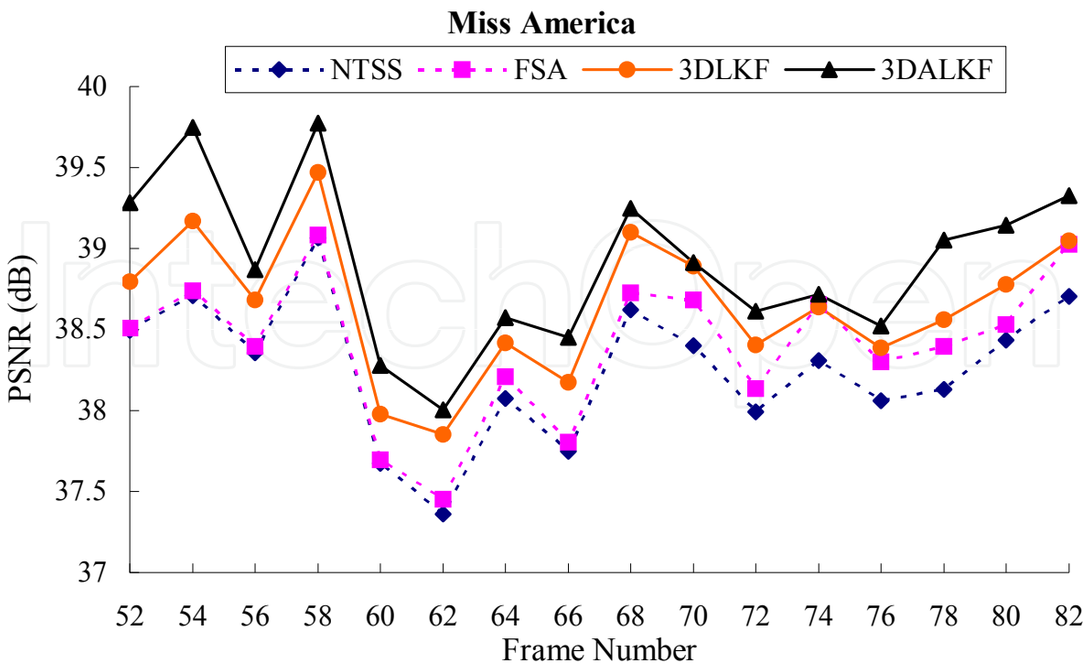


Fig. 7. The PSNR comparison for Miss America sequence at 15Hz

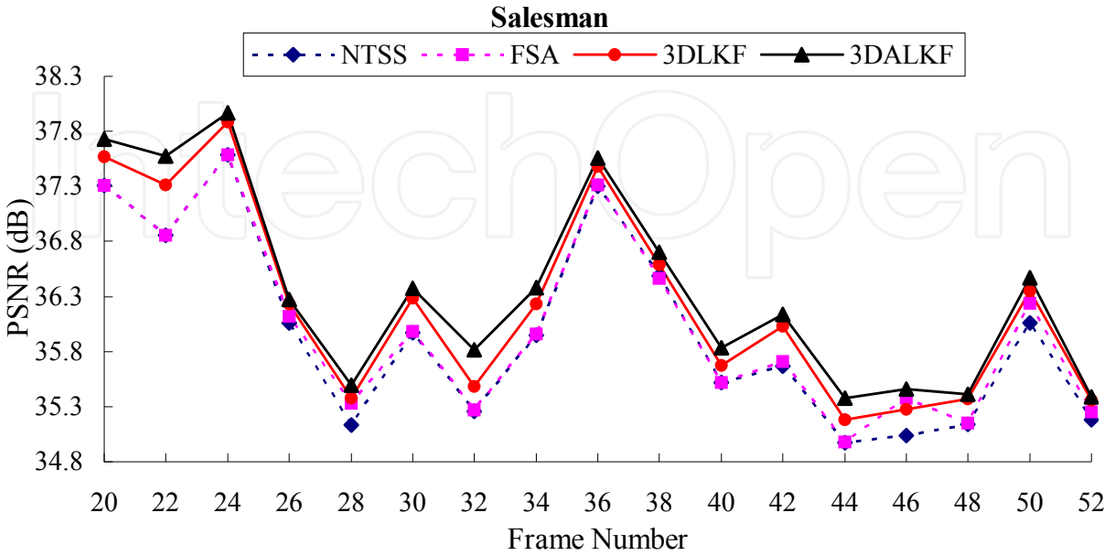


Fig. 8. The PSNR comparison for Salesman sequence at 15Hz

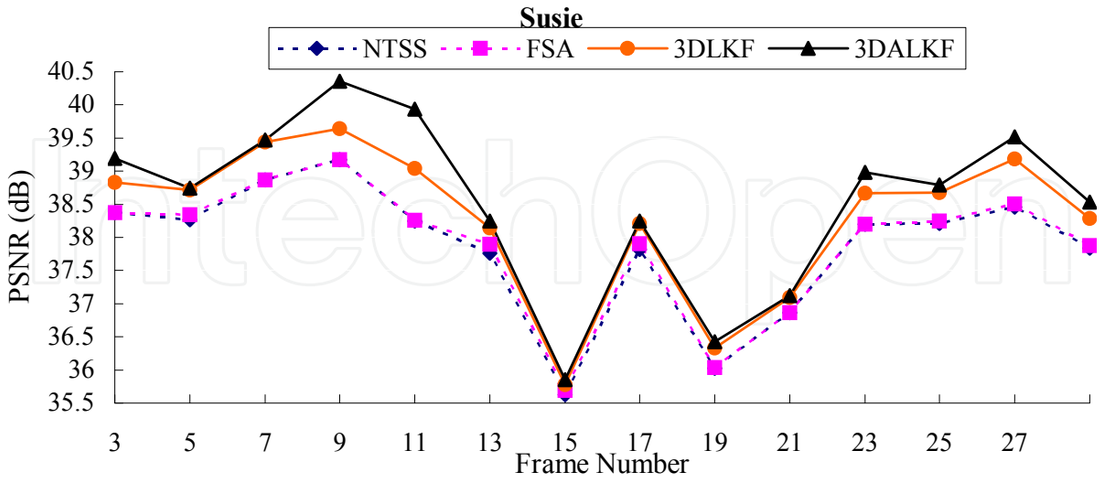


Fig. 9. The PSNR comparison for Susie sequence at 15Hz

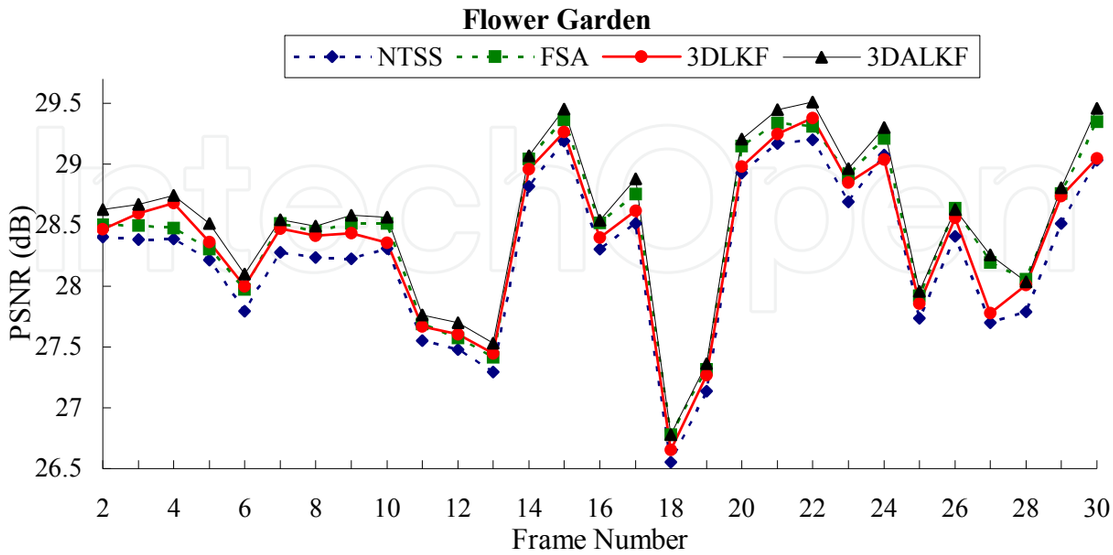


Fig. 10. The PSNR comparison for Flower Garden sequence at 30Hz



Fig. 11. The comparison of reconstructed image (a) Original image (b) NTSS and (c) 3DALKF.

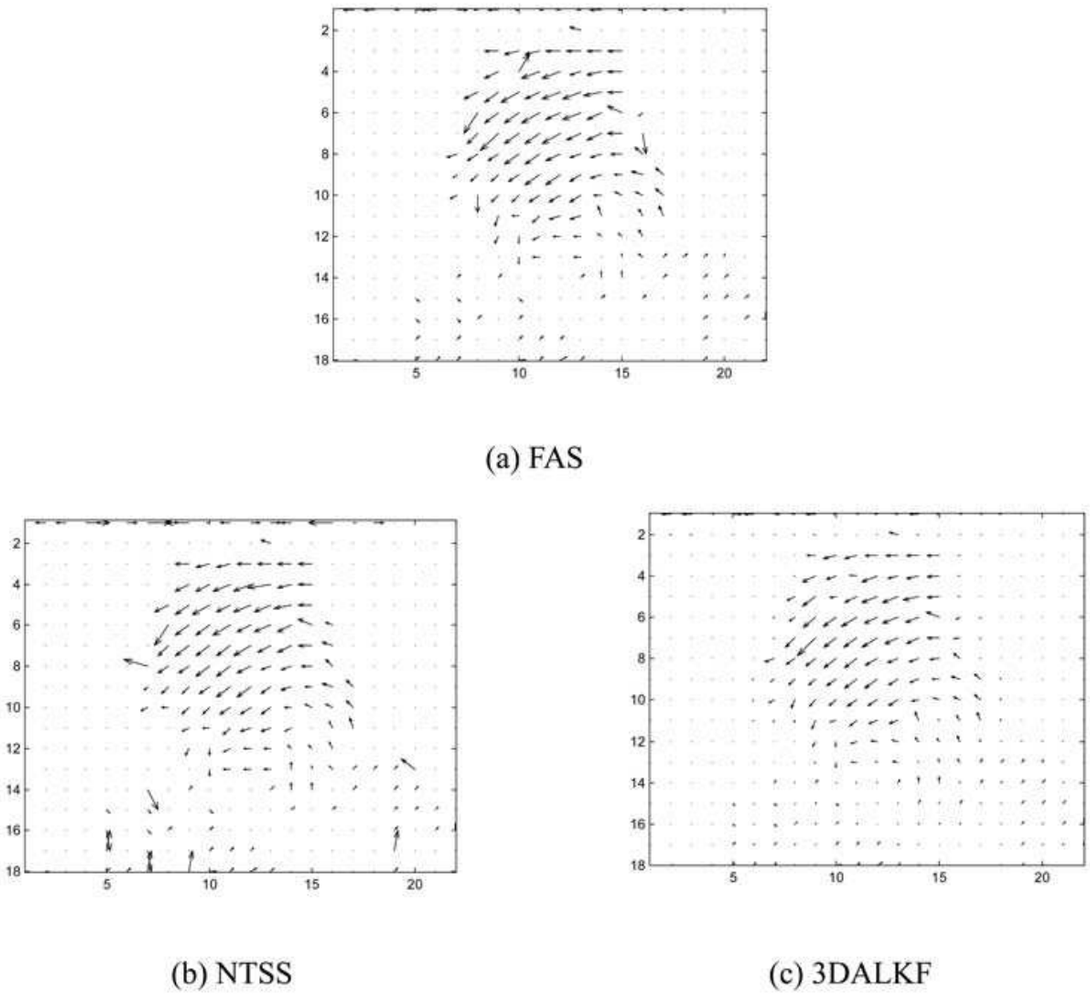


Fig. 12. The motion vector fields obtained by (a) FSA, (b) NTSS (c) 3DALKF

In the proposed methods, the 3DALKF is the most computationally expensive scheme. The computation time for FSA, NTSS, and 3DALKF are listed in Table 2. It is obvious that the computation time of the KF methods are only about 2/5 of the FSA, and is slightly more than that of NTSS. Thus it is suitable for real-time application.

Image Sequence	Computation time (second/frame)			
	NTSS	FSA	3DLKF	3DALKF
Miss America	1.0881	6.4162	2.4000	2.6950
Salesmen	0.5559	3.1694	1.5559	1.8147
Susie	0.9846	7.0262	2.0477	2.2954
Flower Garden	1.5017	44.3021	2.5297	2.8367
Average	1.0326	15.2285	2.1333	2.4105

Table 2. The computation time for various algorithms

3. Rate-constrained motion estimation with Kalman filter

In BMA, the motion compensated prediction difference blocks (called residue blocks) and the motion vectors are encoded and sent to the decoder. In high-quality applications, the bit rate for motion vectors, R_{mv}, is much less than that for residues, R_{res}; thus R_{mv} can be neglected in motion vector estimation. However, in low- or very low- bit rate applications such as videoconference and videophone, the percentage of motion vector bit rate is increased when overall rate budget decreases. Thus, the coding of motion vectors takes up a significant portion of the bandwidth [20]. Then in very low bit rate compression, the motion compensation must consider the assigned motion vector rate and residue rate simultaneously, which yields the so-called rate-constrained motion estimation. In this section, we will present two Kalman filtering based rate-constrained motion estimation algorithms.

3.1 Rate-distortion motion estimation

In conventional motion estimation, a major consideration is to reduce the motion compensated prediction error such that the coding rate for the prediction error can be reduced. This is true for high-rate applications because the bit rate for motion vector (R_{mv}) is only a very small part of the overall transmission rates. However, in low bit-rate or very low bit-rate situation, R_{mv} is a significant part of the available rate budget. For this reason, R_{mv} should be considered into the process of motion estimation. Therefore, the criterion of motion estimation must be modified accordingly.

In 1994, Bernd Girod addressed this problem first. He proposed a theoretical framework for rate-constrained motion estimation, and a new region based motion estimation scheme [22]. In motion compensated hybrid coding, the bit-rate can be divided into the displacement vector field, the prediction error, and additional side information. Very accurate motion compensation is not the key to achieve a better picture quality at low or very low bit-rates.

In 1998, Chen and Willson confirmed this point again [25], and analyzed this issue thoroughly. They explained a new estimation criterion in detail, and proposed a rate-constrained motion estimation for general video coding system. The performance of video compression depends on not only the motion compensation but also the rate budget, which include bit-rate for motion vector and bit-rate for prediction error. Therefore, the optimal solution can then be searched throughout the convex hull of all possible R-D pairs by minimizing the total Lagrangian cost function:

$$J_{\min}\left(\vec{v},q,\lambda\right)=\min_{D\in S^{k},q\in Q^{k}}\sum_{k=1}^{K}D_{k}\left(\vec{v}_{k},q_{k}\right)+\lambda\left[R_{k}^{mv}\left(\vec{v}\right)+R_{k}^{res}\left(\vec{v},q\right)\right], \tag{38}$$

where Q^{K} is the quantization parameter for K blocks, respectively. This approach, however, is computationally intensive, involving a joint optimization between motion estimation/ compensation and prediction residual coding schemes. From Eq. (38), we see that the DCT and quantization operations must be performed on an MV candidate basis in order to obtain $R_{k}^{res}\left(\vec{v},q\right)$ and $D_{k}\left(\vec{v}_{k},q_{k}\right)$. The significant computations make the scheme unacceptable for most practical applications, no matter what software or hardware implementation is adopted. Thus, they simplify Eq. (38) by only considering motion estimation error and bit-rate for MV.

Assume a frame is partitioned into K block sets. Let $\hat{v}_k \in U^K$ be the motion vector estimated for block k. Then the motion field of a frame is described by the K_1-tuple vector, $V = \left(\hat{v}_1, \hat{v}_2, ...\hat{v}_{K_1}\right) \in U^K$. The joint rate and distortion optimization can be interpreted as finding a motion vector field that minimizes the distortion under a given rate constraint, which can be formulated by the Lagrange multiplier method as

$$J_{\min}(v,\lambda) = \sum_{U^K} \min_{V \in U^K} \{\sum_{k=1}^{K} D_k(\hat{v}_k) + \lambda R_k(\hat{v}_k)\}, \tag{39}$$

where λ is the Lagrange multiplier; D_k and R_k are the motion-compensated distortion and the number of bits associated with motion vector of the block k, respectively. In most video coding standards, the motion vectors of blocks are differentially coded using Huffman code. Thus, the blocks are coded dependently. However, this simplification has two evident defects: (i) it is still too complex, and (ii) the performance is degraded.

In the same year, Coban and Merserau proposed different scheme on the RD-optimal problem [26]. They think that Eq. (39) is a principle for global optimal of R-D problem, but it is difficult in implementation. They supposed, if each block is coded independently, the solution Eq. (39) can be reduced to minimizing the Lagrangian cost function of each block, i.e.,

$$J_{\min}(v,\lambda) = \min_{V \in U^k} \{D_k(\hat{v}_k) + \lambda R_k(\hat{v}_k)\}. \tag{40}$$

In order to simplify the problem, although the MV's are coded differentially, the blocks will be treated as if they are being coded independently. This will lead to a locally optimal, globally sub-optimal solution. By this way, the framework of R-D optimal motion estimation is close to conventional motion estimation. Although it saves computation by ignoring the relation of blocks, it reduces the overall performance.

3.2 Enhanced R-D motion estimation using Kalman filter

The R-D motion estimation often yields smooth motion vector fields, as compared with conventional BMAs [25,26]. In other words, the resulting motion vectors are highly correlated. In this work, we try to fully exploit the correlation of motion vectors by using the Kalman filter. This is motivated by our previous works [37,40] that the Kalman filter is combined with the conventional BMA's to improve the estimate accuracy of motion vectors. The system consists of two cascaded stages: measurement of motion vector and Kalman filtering. We can employ a R-D fast search scheme [20,23-26] to obtain the measured motion vector. Then we model the motion vectors and generate the predicted motion vector utilizing the inter-block correlation. Based on the measured and predicted motion vectors, a Kalman filter is applied to obtain an optimal estimate of motion vector.

For the sake of simplicity in implementation, we employ the first-order AR (autoregressive) model to characterize the motion vector correlation. The motion vector of the block at location (m,n) of the i-th frame is denoted by $\mathrm{v}_i(m,n) = [v_{i,x}(m,n), v_{i,y}(m,n)]$, and its two components in horizontal and vertical directions are modeled as

$$v_{i,x}(m,n) = a_1 v_{i,x}(m,n-1) + w_{i,x}(m,n) \tag{41}$$

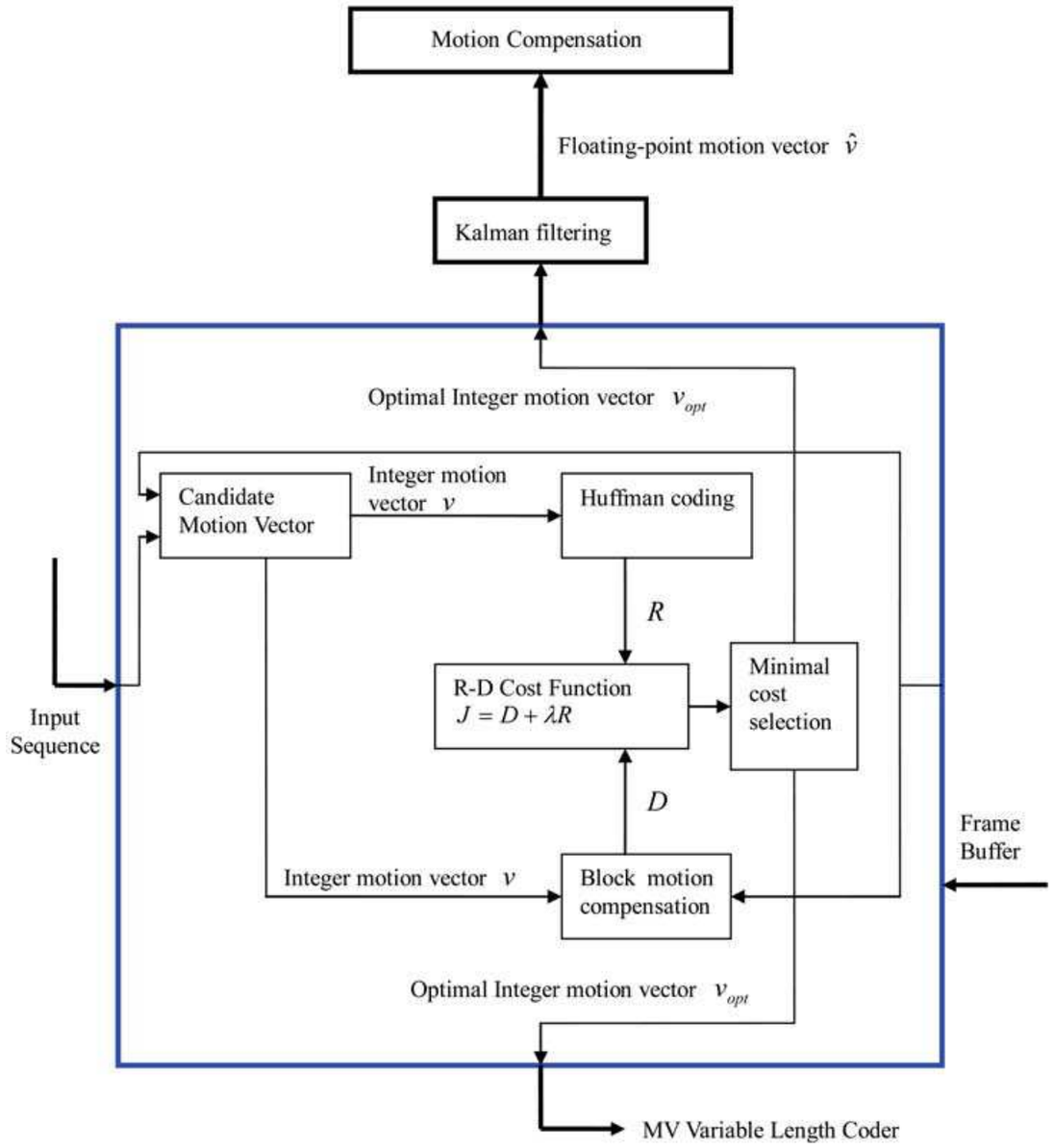


Fig. 13. The block diagram of the proposed enhanced R-D motion estimation algorithm

$$v_{i,y}(m,n) = b_1 v_{i,y}(m,n-1) + w_{i,y}(m,n)\,. \tag{42}$$

where $w_{i,x}(m,n)$ and $w_{i,y}(m,n)$ represent the model error components. In order to derive the state-space representation, the time indexes k and $k-1$ are used to represent the current block location (m,n), and the left-neighbor block location $(m-1,n)$, respectively. Consequently, the state-space representation of (41) and (42) are

$$\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} a_1 & 0 \\ 0 & b_1 \end{bmatrix} \begin{bmatrix} x_1(k-1) \\ x_2(k-1) \end{bmatrix} + \begin{bmatrix} w_{i,x}(k) \\ w_{i,y}(k) \end{bmatrix}, \tag{43}$$

or

$$\mathbf{V}_x(k) = \mathbf{\Phi}_x(k-1)\mathbf{V}_x(k-1) + \mathbf{\Gamma}(k)\mathbf{w}_x(k), \tag{44}$$

where we let $x_1(k) = v_{i,x}(k)$, $x_1(k-1) = v_{i,x}(k-1)$, $x_2(k) = v_{i,y}(k)$ and $x_2(k-1) = v_{i,y}(k-1)$. The error components, $w_{i,x}(k)$ and $w_{i,y}(k)$, are assumed to be Gaussian distribution with zero mean and the same variance $q(k)$.

The measurement equations for the horizontal and vertical directions are expressed by

$$\mathbf{z}(k) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} n_x(k) \\ n_y(k) \end{bmatrix}$$

$$= \mathbf{H}(k)\mathbf{V}_x(k) + \mathbf{n}_x(k), \tag{45}$$

where $n_x(k)$, $n_y(k)$ denote two measurement error components with the same variance $r(k)$.

In general, the model error $\mathbf{W}(k)$ and measurement error $\mathbf{n}(k)$ may have colored noises. We can model each colored noise by a low-order difference equation that is excited by white Gaussian noise, and augment the states associated colored noise models to the original state space representation. Finally, we apply the recursive filter to the augmented system. However, the procedure requires considerable computational complexity and is not suitable for our application. Moreover, the blocks are processed independently when the measurements are obtained by the R-D fast search algorithm [26]. Thus, we can assume that the measurement error is independent. For simplicity but without loss of generality, the prediction error and measurement error are assumed to be zero-mean Gaussian distribution with the same variances $q(k)$ and $r(k)$, respectively.

In the above equations, the measurement matrix $\mathbf{H}(k)$ is constant, and state transition matrix $\mathbf{\Phi}(k)$ can be estimated by the least square method. Since the motion field for low bit-rate applications is rather smooth, we assume that $q(k)$ and $r(k)$ are fixed values.

The algorithm is summarized as follows.

<Step 1> Measure motion vector

Measure the motion vector of a moving block, $\mathbf{z}(k) = [z_x(k) \quad z_y(k)]^T$, by any R-D search algorithms [20][23]-[26]. Encode the motion vector by H.263 Huffman table [7][9].

<Step 2> Kalman filtering

a. The predicted motion vector is obtained by

$$\hat{\mathbf{V}}^-(k) = \mathbf{\Phi}(k-1)\hat{\mathbf{V}}^+(k-1).$$

b. Calculate prediction-error covariance by

$$\mathbf{P}^-(k) = \mathbf{\Phi}(k-1)\mathbf{P}^+(k-1)\mathbf{\Phi}^T(k-1) + \mathbf{\Gamma}(k)\mathbf{Q}(k)\mathbf{\Gamma}^T(k).$$

c. Obtain Kalman gain by

$$\mathbf{K}(k) = \mathbf{P}^-(k)\mathbf{H}^T(k)[\mathbf{H}(k)\mathbf{P}^-(k)\mathbf{H}^T(k) + \mathbf{R}(k)]^{-1}.$$

d. The motion vector estimate is updated by

$$\hat{\mathbf{V}}^{+}(k) = \hat{\mathbf{V}}^{-}(k) + \mathbf{K}(k)[\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{V}}^{-}(k)] .$$

This is the final estimate output.

e. Calculate the filtering-error covariance by

$$\mathbf{P}^{+}(k) = [\mathrm{I} - \mathbf{K}(k)\mathbf{H}(k)]\mathbf{P}^{-}(k) .$$

<Step 3> Go to <step 1> for next block.

In the above algorithm, the optimal estimate $\hat{\mathbf{V}}(k)$ is usually real, which yields fractional-pixel accuracy estimate. The conventional BMA can also obtain the fractional-pixel motion vector by increasing resolution with interpolation and matching higher-resolution data on the new sampling grid. However, this not only increases computational complexity significantly, but also raises overhead bit rate for motion vector. On the contrast, the required computational overhead is much lower than that of the conventional BMA with fractional-pixel matching. In addition, using the same Kalman filter as in the encoder, the decoder can estimate the fractional part of motion vector by receiving integer motion vector. In summary, this method achieves fractional pixel performance with the same bit-rate for motion vector as an integer-search BMA, at the cost of a small increase of computational load at the decoder. Furthermore, because the Kalman filter is independent with motion estimation, it can be combined with any existing R-D motion estimation scheme with performance improvement.

3.3 Kalman filter embedded R-D motion estimation

The main feature of the above enhanced scheme is to obtain fractional pixel accuracy of motion vector with estimation instead of actual searching. Hence, no extra bit rate is needed for the fractional part of motion vector. However, because the enhanced algorithm does not involve the estimation process of motion vector, the obtained motion vector is not optimum from viewpoint of distortion. To address the problem, here we introduce a method for R-D motion estimation in which the Kalman filter is embedded. We refer to it as Kalman filter embedded R-D motion estimation and describe the details as follows.

The cost function of Kalman filter embedded R-D motion estimation can be formulated as

$$J_{\min}(v,\lambda) = \min_{V \in U^k} \{ Kalman\ [D_k(\hat{v}_k)] + \lambda R_k(\hat{v}_k) \} , \tag{46}$$

where the $Kalman[D_k(\hat{v}_k)]$ is a distortion of Kalman filter-based motion compensation. It is obtained by Kalman filtering the integer-point motion vector and the resulting floating-point motion vector is used to generate motion compensation prediction. In such case, the motion vector is represented in integer-point, but it can generate motion compensation with fractional pixel accuracy. Therefore, the assigned bit rate for motion vector is not affected by $Kalman[D_k(\hat{v}_k)]$, but the total cost function is reduced due to the accuracy increase in compensation. Figure 14 is the block diagram of the embedded algorithm. For simplicity, we select Eq. (46) as the criterion for motion estimation.

The Kalman filter embedded R-D motion estimation algorithm is summarized as follows.

<Step 1> Kalman filter-based motion estimation

a. Select a location in the search range and denote it as a candidate measurement of motion vector $[z_x, z_y]$.

b. Apply the Kalman filter to $[z_x, z_y]$ using the procedure of Step 2 in the previous section. Then we obtain an optimal estimate of motion vector $[\hat{v}_x, \hat{v}_y]$, which is with fractional accuracy. Calculate the distortion $Kalman[D_k(\hat{v}_k)]$ according to the $[\hat{v}_x, \hat{v}_y]$.

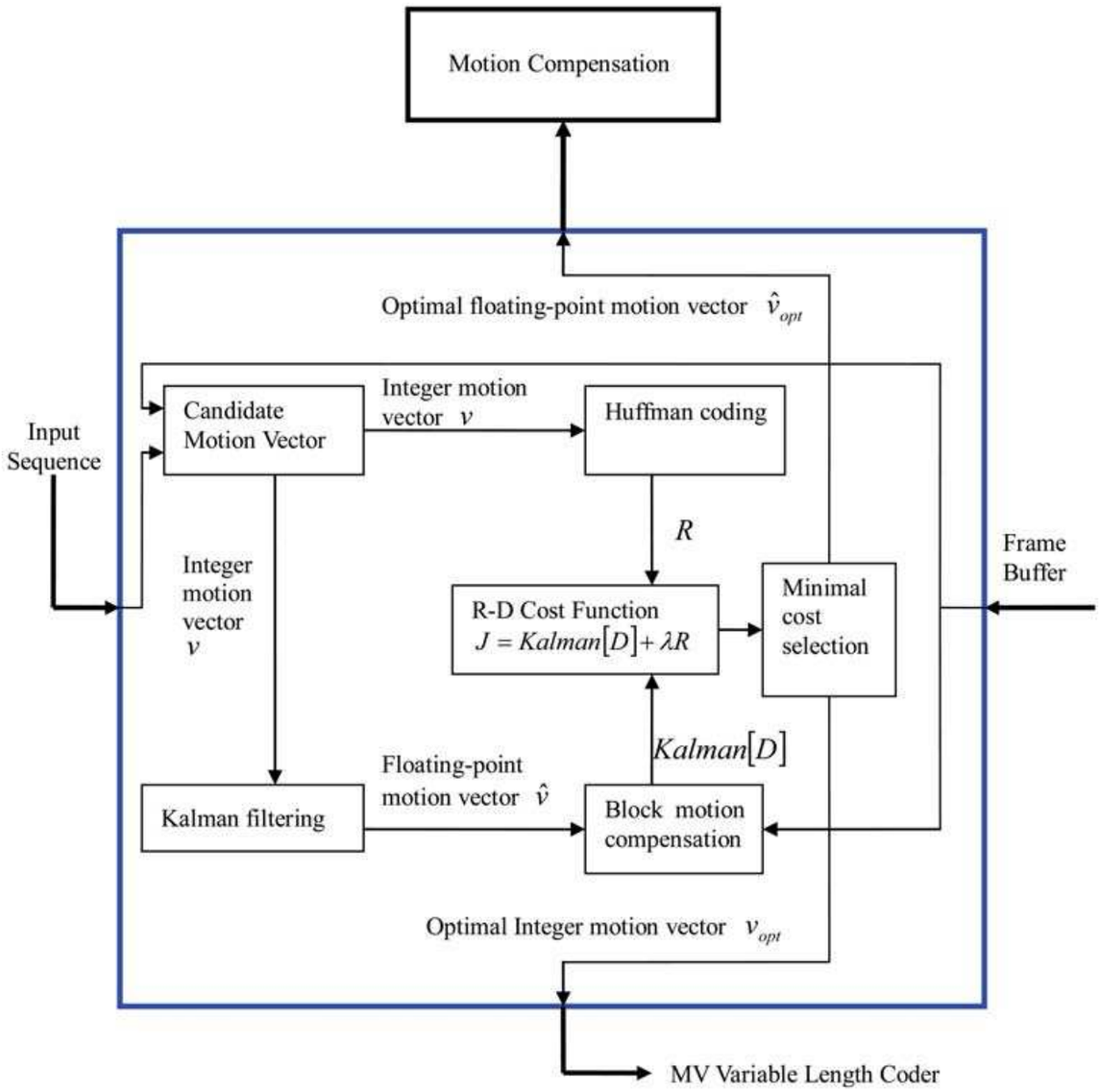


Fig. 14. The block diagram of the proposed embedded R-D motion estimation algorithm

<Step 2> Calculate the bit rate of the motion vector $[z_x, z_y]$ according to the H.263 Huffman table [7][9]. Notice that transmission motion vector is $[z_x, z_y]$, which is an integer; thus the required bit rate of motion vector is not affected by Kalman filter.
<Step 3> Using (46), we calculate the cost function. If the best match is found, go to <step 4>; otherwise, go back <Step 1> to select the next location for estimation.
<Step 4> Go to <step 1> for next block.

In the enhanced algorithm, the Kalman filter is not applied during the block searching. It is only used to enhance the performance when motion vector is obtained by R-D motion estimation. Therefore, the Kalman filter can be viewed as a post processing of motion estimation. However, in the embedded algorithm, the Kalman filter is applied for every block searching by employing the joint rate-distortion. Thus, it can be considered as a new R-D motion estimation approach. Since it includes the Kalman filter into the optimization process, the embedded method performs better than the enhanced version at the cost of computational complexity.

3.4 Simulation results

The performance of the RD-motion estimation with Kalman filter (RD-Kalman) was evaluated using a set of standard image sequences including Forman, Mother and Daughter, Carphone, Salesman and Claire. All sequences are with CIF (352×288) or QCIF (176×144) resolution and frame rate of 10 Hz. Since the RD-Kalman motion estimation has fractional pel accuracy, the results are compared with the conventional RD algorithm and MSE-optimal scheme with both integer and half-pixel accuracy. The block size 16×16 and search range 64×64 for CIF format and block size 16×16 (or 8×8) and search range 31×31 for QCIF format were chosen, respectively. The conventional RD and RD-Kalman adopted the same motion estimation strategy as that in [25]. Specifically, for the current block, the motion vectors of the left-neighbor block and up-neighbor block, and the motion vector obtained with MSE criterion, were selected as the predicted search center, and then a small search of 3×3 is performed.

For the KF-based motion estimation, the parameters are chosen experimentally as follows: the model coefficients $a_1 = b_1 = 1$, model error variance $q(k)$=0.8, measurement error variance $r(k)$=0.2, initial error covariance $\mathbf{P}(0)=\mathbf{I}$, and initial state $\hat{\mathbf{V}}(0) = 0$. It is evident that from [31], the estimated motion vectors are real values rather than integer. The displaced pixels may not be on the sampling grid. Therefore, the well-known bilinear interpolation is adopted to generate a motion compensated prediction frame. A Huffman codebook adopted from H.263 standard was used in the coding of 2-D differentially coded motion vectors. The various algorithms were compared in terms of rate and distortion performance. The common PSNR measure defined in the following was selected to evaluate distortion performance.

$$PSNR = 10 \cdot \log_{10} \frac{255^2}{MSE} \tag{47}$$

Moreover, rate performance was evaluated by the number of bits required to encode an image frame or a motion field.

The Lagrange multiplier λ, which controls the overall performance in the rate distortion sense, is a very important parameter. Generally, an iterative method is needed to determine the value of λ. However, it is very computational expensive. As pointed out in [26], for typical video coding applications λ is insensitive to different frames of a video sequence; thus a constant λ of 20 is adopted in the simulations.

The simulations were carried out by incorporating various motion estimation algorithms into an H.263 based MC-DCT video coding system. To be fair in the comparisons, we fixed the overall coding bit-rate at 4000 bits per frame for CIF-Claire and CIF-Salesman. For QCIF format, two block sizes are conducted for each sequence, which are assigned two different

bit-rates per frame, respectively. The bit rates preset are 2000 bits (8×8) and 1600 bits (16×16) for Forman, 1400 bits and 1000 bits for Mother & Daughter, and 2000 bits and 1300 bits for Carphone, respectively.

	CIF-Claire Sequence, fixed coding bit-rate at 4000 bits				
	Block size, 16 x 16				
	MC psnr	MC+Res psnr	MV rate	Res rate	Overall rate
Full Search	38.86	39.67	1620	2424	4048
Half Pixel	40.32	40.92	2089	1989	4078
Half Pixel(RD)	39.23	41.53	1349	2637	3986
Half Pixel(RD) + KF(en)	39.46	41.79	1349	2637	3986
Half Pixel(RD) + KF(em)	39.57	42.43	1133	2686	3849
RD Optimal	38.82	41.76	1326	2628	3954
RD + KF(en)	38.93	41.82	1326	2628	3954
RD + KF(em)	39.20	42.33	947	2784	3731

Table 3. Comparisons of compression performance, in terns of PSNR, Overall Bit Rate, and MV Bit Rate for various motion estimation algorithms using the CIF-Clair 100 frames under 15 frames/s

	CIF-Salesman Sequence, fixed coding bit-rate at 4000 bits				
	Block size, 16 x 16				
	MC psnr	MC+Res psnr	MV rate	Res rate	Overall rate
Full Search	37.42	40.56	1042	2802	3844
Half Pixel	38.94	41.28	1268	2770	4038
Half Pixel(RD)	37.88	40.82	1043	2773	3816
Half Pixel(RD) + KF(en)	38.23	41.13	1043	2773	3816
Half Pixel(RD) + KF(em)	38.78	41.67	977	2786	3763
RD Optimal	37.33	40.93	1028	2805	3833
RD + KF(en)	37.45	41.16	1028	2805	3833
RD + KF(em)	38.54	41.51	950	2878	3828

Table 4. Comparisons of compression performance, in terns of PSNR, Overall Bit Rate, and MV Bit Rate for various motion estimation algorithms using the CIF-Salesman 100 frames under 15 frames/s

The averaged results for 100 frames of CIF format sequences are summarized in Tables 3 and 4. The Kalman-based R-D motion estimation approach outperforms the MSE-optimal and conventional RD algorithms in terms of PSNR. Since the Kalman filter has fractional pel accuracy with the rates of integer motion vector, it achieves significant PSNR improvement, as expected. When the integer-based Kalman filter is compared to the motion estimation

methods in half pixel accuracy, it still achieves better PSNR, but not so significantly. It can be seen that the Kalman filter with half pixel accuracy performs better slightly than that with integer pixel accuracy. This may be due to the limitation of bilinear interpolation; i.e., the accuracy improvement is saturated when too many interpolations are performed. The performance may be further enhanced with the advanced interpolation filters [45,46].

	QCIF-Foreman Sequence, fixed coding bit-rate at 2000 bits				
	Block size, 8 x 8				
	MC psnr	MC+Res psnr	MV rate	Res rate	Overall rate
Full Search	32.37	33.49	1274	776	2050
Half Pixel	33.75	34.38	1438	626	2064
Half Pixel(RD)	33.44	34.57	1162	844	2006
Half Pixel(RD) + KF(en)	33.59	34.61	1162	844	2006
Half Pixel(RD) + KF(em)	33.71	34.93	1047	956	2003
RD Optimal	31.83	34.27	1096	903	1999
RD + KF(en)	31.92	34.33	1096	903	1999
RD + KF(em)	32.11	34.62	1022	971	1993

*MV: Motion Vector *MC：Motion Compensation. *Res：Prediction Residuals. (DFD)

Table 5. Comparisons of compression performance, in terns of PSNR, Overall Bit Rate, and MV Bit Rate for various motion estimation algorithms using the QCIF-Foreman 100 frames under 10 frames/s

	QCIF-Foreman Sequence, fixed coding bit-rate at 1600 bits				
	Block size, 16 x 16				
	MC psnr	MC+Res psnr	MV rate	Res rate	Overall rate
Full Search	30.75	32.79	635	970	1605
Half Pixel	31.83	33.55	728	893	1621
Half Pixel(RD)	31.69	33.72	655	948	1603
Half Pixel(RD) + KF(en)	31.76	33.79	655	948	1603
Half Pixel(RD) + KF(em)	31.89	33.97	604	993	1597
RD Optimal	30.62	33.13	583	1028	1611
RD + KF(en)	30.72	33.39	583	1028	1611
RD + KF(em)	31.04	33.67	553	1051	1604

Table 6. Comparisons of compression performance, in terns of PSNR, Overall Bit Rate, and MV Bit Rate for various motion estimation algorithms using the QCIF-Foreman 100 frames under 10 frames/s

At the same bite rate level and integer pixel accuracy, the enhanced algorithm achieved an average of 1.23 dB gain over MSE-optimal and 0.34 dB gain over the conventional RD. The embedded version achieved an average of 1.77 dB gain over MSE-optimal, and 0.88 dB gain over the conventional RD. Note that the new methods have lower bit rate. Tables 5-7 summarized the average results for QCIF format sequences. For both block sizes of 16×16 and 8×8, the Kalman filter-based R-D motion estimation approaches achieve significant PSNR improvement. Particularly, the embedded Kalman R-D algorithm achieves the best performance due to its ability in reduction of motion vector rate as well as the compensation distortion.

	QCIF-Mother & Daughter Sequence, fixed coding bit-rate at 1400 bits				
	Block size, 8 x 8				
	MC psnr	**MC+Res psnr**	**MV rate**	**Res rate**	**Overall rate**
Full Search	37.21	38.79	762	654	1216
Half Pixel	38.57	39.88	816	603	1419
Half Pixel(RD)	37.72	40.14	638	670	1308
Half Pixel(RD) + KF(en)	37.98	40.66	638	670	1308
Half Pixel(RD) + KF(em)	38.32	41.98	463	694	1157
RD Optimal	37.15	40.28	431	705	1136
RD + KF(en)	37.23	41.31	431	705	1136
RD + KF(em)	37.55	42.06	216	806	1022

Table 7. Comparisons of compression performance, in terns of PSNR, Overall Bit Rate, and MV Bit Rate for various motion estimation algorithms using the QCIF-Mother & Daughter 120 frames under 10 frames/s

Figures 15 and 16 compare the MSE-Optimal, conventional R-D, enhanced Kalman R-D and embedded Kalman R-D schemes with both integer and half pixel accuracy in terms of PSNR with approximately fixed bit rate for each sequence, respectively. Figures 17 to 19 compare these algorithms in terms of bit rate with approximately fixed PSNR for each sequence, respectively. The results indicate that the proposed schemes achieve better rate-distortion performance.

The motion vector fields generated by various algorithms are shown in Figures 20-21, respectively. The test sequences contain mainly small rotation and camera panning. This algorithm produces smoother motion fields because of the filtering effect of Kalman filter.

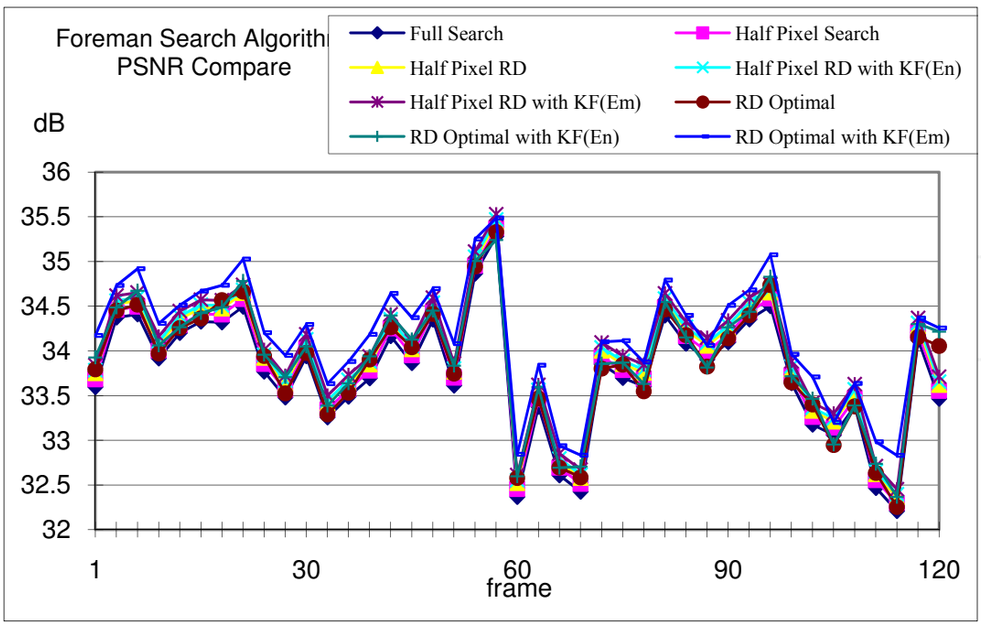


Fig. 15. Comparisons of PSNR performance using the QCIF-Froeman sequence, 120 frames at 10 frames/s, fixed coding bit-rate at 2000 bits. Block size = 8 x 8, search range = [-15, 16].

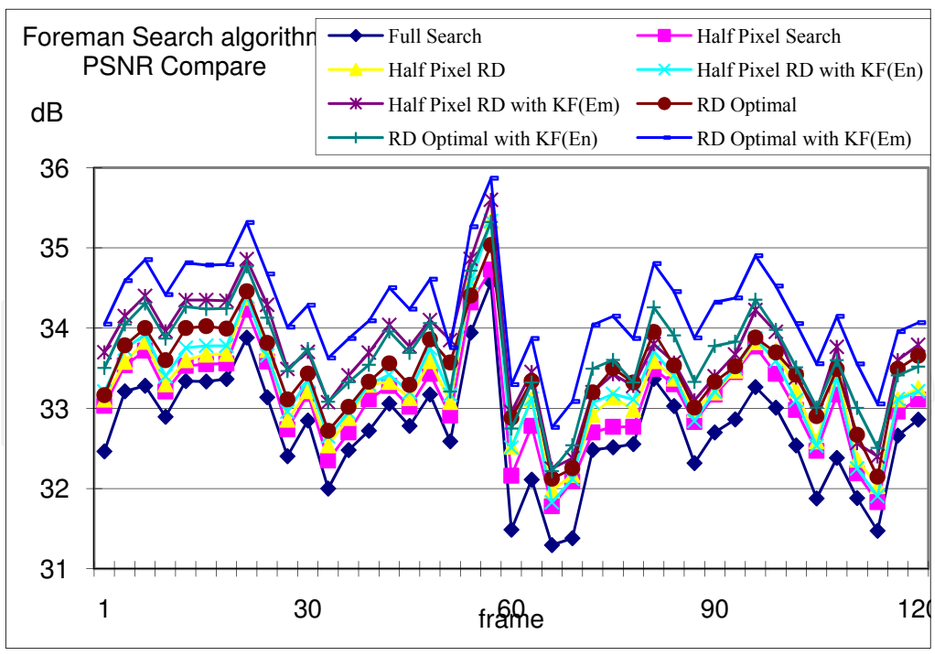


Fig. 16. Comparisons of PSNR performance using the QCIF-Froeman sequence, 120 frames at 10 frames/s, fixed coding bit-rate at 2000 bits. Block size = 16 x 16, search range = [-31, 32].

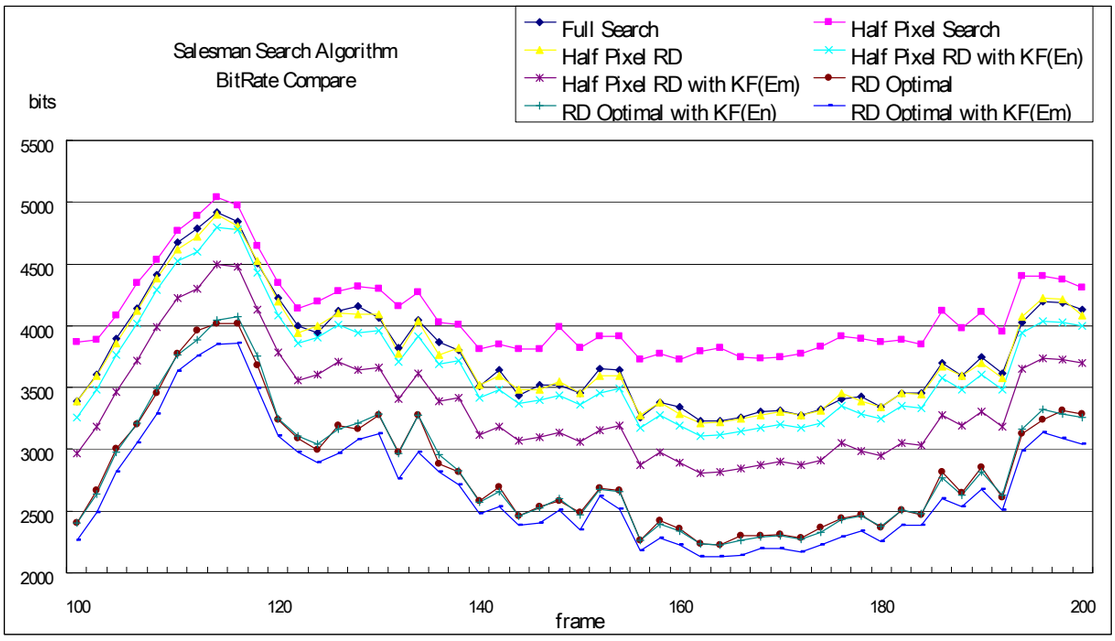


Fig. 17. Comparisons of bit-rate performance using the CIF-Salesman sequence, 120 frames at 10 frames/s, fixed average PSNR Full Search at 39.93 dB, Half Pixel Search at 40.01 dB, Half Pixel RD at 39.92 dB, Half Pixel RD with KF(En) at 39.95 dB, Half Pixel RD with KF(Em) at 40.05 dB, RD-Optimal at 39.88 dB, RD with KF(En) at 39.95 dB, and RD with KF(Em) at 39.98 dB.

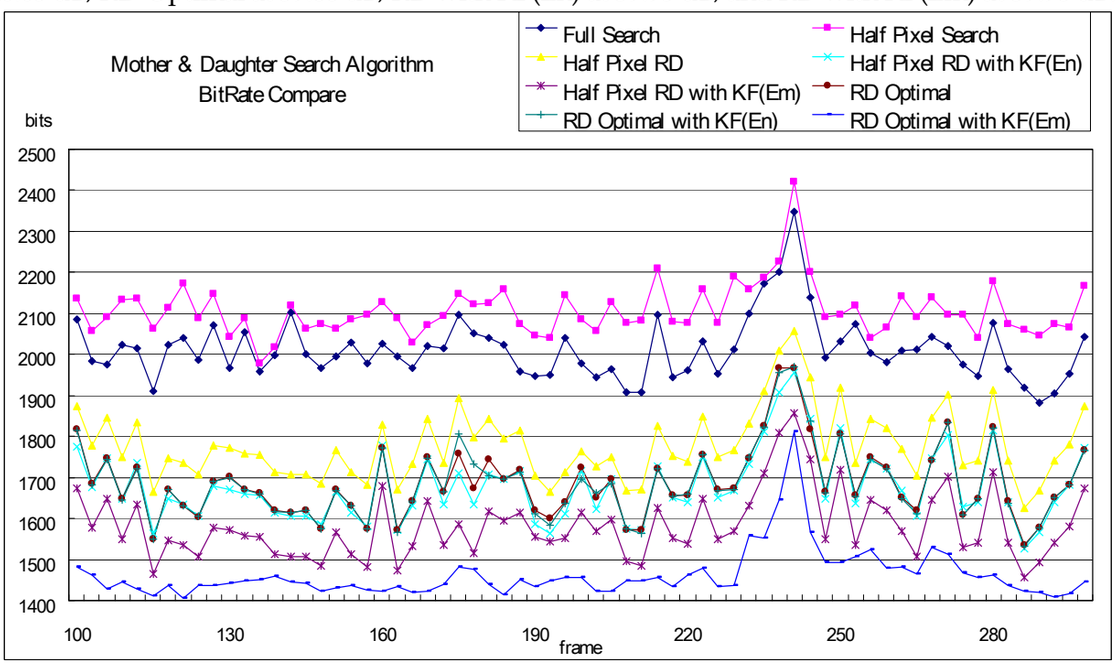


Fig. 18. Comparisons of bit-rate performance using the QCIF-Mother & Daughter sequence, 200 frames at 10 frames/s, fixed average PSNR Full Search at 38.80 dB, Half Pixel Search at 38.82 dB, Half Pixel RD at 38.75 dB, Half Pixel RD with KF(En) at 38.81 dB, Half Pixel RD with KF(Em) at 38.89 dB, RD-Optimal at 38.77 dB, RD with KF(En) at 38.83 dB, and RD with KF(Em) at 38.87 dB.

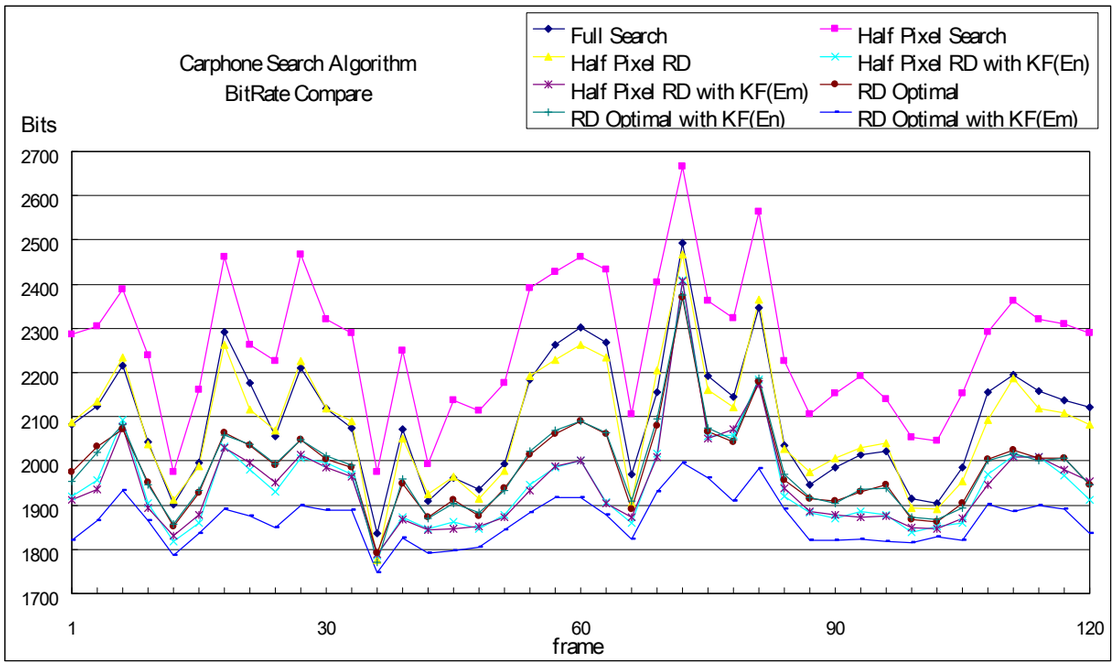


Fig. 19. Comparisons of bit-rate performance using the QCIF-Carphone sequence, 120 frames at 10 frames/s, fixed average PSNR Full Search at 37.23 dB, Half Pixel Search at 37.32 dB, Half Pixel RD at 37.20 dB, Half Pixel RD with KF(En) at 37.29 dB, Half Pixel RD with KF(Em) at 37.37 dB, RD-Optimal at 37.14 dB, RD with KF(En) at 37.28 dB, and RD with KF(Em) at 37.30 dB.

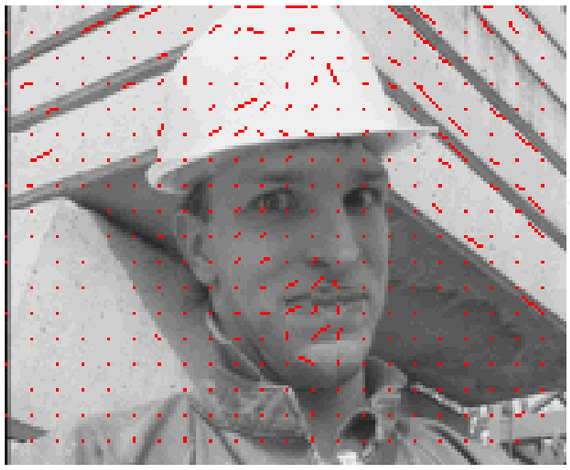


Fig. 20 (a). Motion field estimated by the conventional Half Pixel scheme on the QCIF-Foreman sequence frame 204. The PSNR quality is 34.56 dB and it requires 1230 bits to encode using the H.263 Huffman codebook.

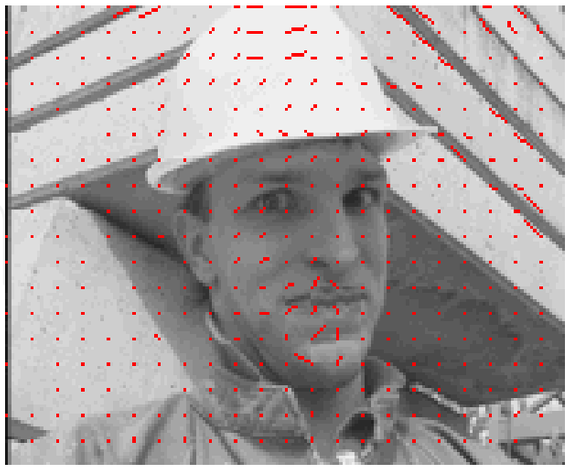


Fig. 20 (b). Motion field estimated by the Half Pixel with RD-Optimal on the QCIF-Foreman sequence frame 204. The PSNR quality is 34.15 dB and it requires 1158 bits to encode using the H.263 Huffman codebook.

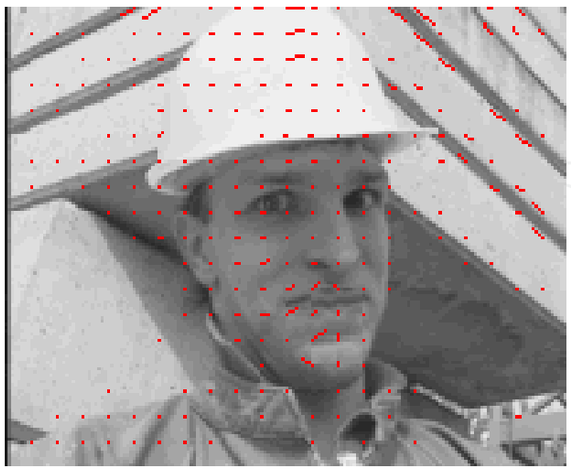


Fig. 20 (c). Motion field estimated by the Half Pixel RD with Enhanced Algorithm scheme on the QCIF-Foreman sequence frame 204. The PSNR quality is 34.27 dB and it requires 1158 bits to encode using the H.263 Huffman codebook.



Fig. 20 (d). Motion field estimated by the Half Pixel RD with Embedded Algorithm scheme on the QCIF-Foreman sequence frame 204. The PSNR quality is 34.66 dB and it requires 889 bits to encode using the H.263 Huffman codebook.

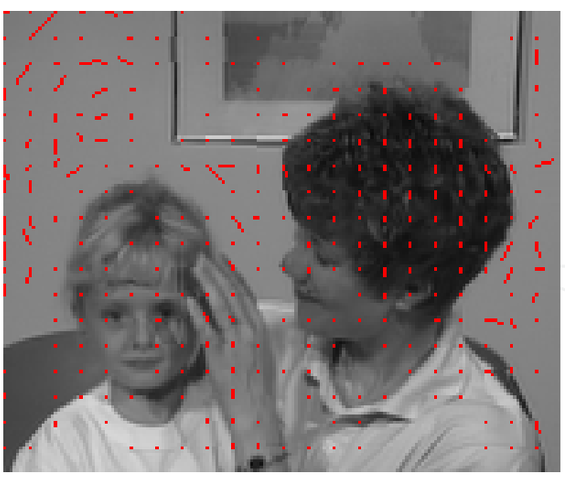


Fig. 21 (a). Motion field estimated by the conventional Half Pixel scheme on the QCIF-Mother & Daughter sequence frame 28. The PSNR quality is 34.83 dB and it requires 1476 bits to encode using the H.263 Huffman codebook.

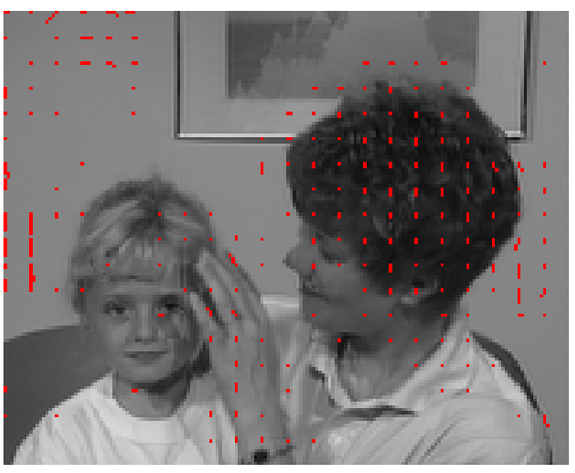


Fig. 21 (b). Motion field estimated by the Half Pixel with RD-Optimal scheme on the QCIF- Mother & Daughter sequence frame 28. The PSNR quality is 34.52 dB and it requires 1112 bits to encode using the H.263 Huffman codebook.

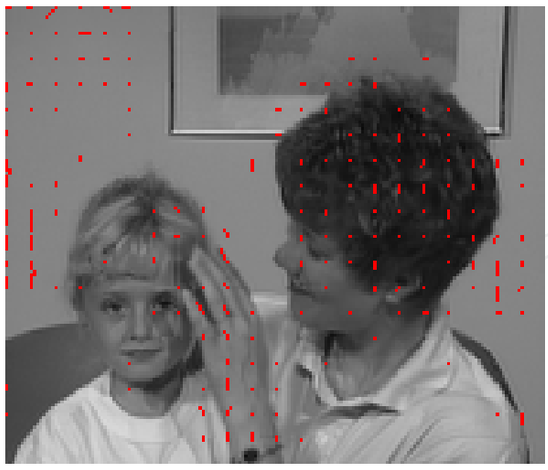


Fig. 21(c). Motion field estimated by the Half Pixel RD with Enhanced Algorithm scheme on the QCIF- Mother & Daughter sequence frame 28. The PSNR quality is 34.67 dB and it requires 1120 bits to encode using the H.263 Huffman codebook.

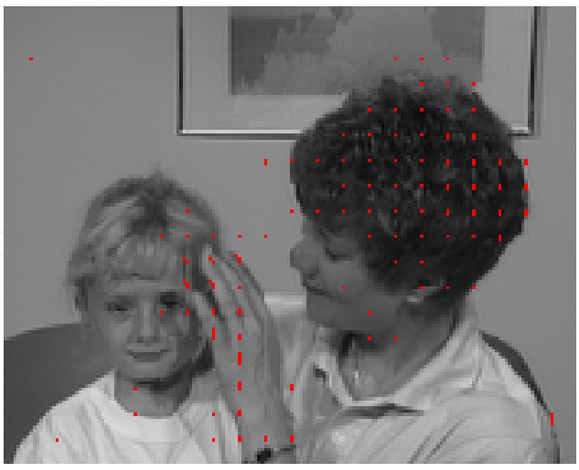


Fig. 21(d). Motion field estimated by the Half Pixel RD with Embedded Algorithm scheme on the QCIF- Mother & Daughter frame 28. The PSNR quality is 34.95 dB and it requires 868 bits to encode using the H.263 Huffman codebook.

In general, the Kalman filtering is computationally expensive. However, both the computational complexities of the enhanced and embedded algorithms are relatively small because the calculation of Kalman filtering can be significantly simplified. The extra computational load required for the algorithms is summarized in Table 8 [47]. It indicates the extra computation introduced by the proposed method is small.

Motion Estimation Algorithms	Computational Complexity		
	Additions	Multiplications	Bilinear interpolation
Enhanced algorithm (per block)	5	3	1 ($8N^2(\times)+6N^2(+)$)
Embedded algorithm (per search)	5	3	1 ($8N^2(\times)+6N^2(+)$)

Table 8. Extra computation required by Kalman filtering for each algorithm

4. Conclusions

In this chapter, we have introduced two types of motion estimation based on Kalman filter, without and with rate-constraint. The first type employs the predicted motion information and the measured motion information to obtain an optimal estimate of motion vector. The predicted motion is achieved through the use of AR models which characterize the motion correlation of neighboring blocks. The measurement motion is obtained by using any conventional block-matching fast search scheme. The results indicate that the method provides smoother motion vector fields than that of the full search scheme, and saves computational cost significantly.

For the rate-constraint case, we have introduced two efficient Kalman filter-based R-D motion estimation algorithms in which a simple 1-D Kalman filter is applied to improve the performance of conventional RD motion estimation. Since equivalent Kalman filters are used in both encoder and decoder, no extra information bit for motion vector is needed to send to the decoder. The algorithm achieves significantly PSNR gain with only a slight increase of complexity. The enhanced algorithm is a post processing, and can be easily combined with any conventional R-D motion estimation schemes. The embedded algorithm can more effectively exploit the correlation of block motion.

5. References

J. R. Jain and A. K. Jain. Displacement measurement and its application in interframe image coding. *IEEE Transactions on Communications*, 29: pages 1799-1808, 1981. [1]

S. Kappagantula and K. R. Rao. Motion compensated interframe image prediction. *IEEE Transactions on Communications*, 33(9): pages 1011-1015, 1985. [2]

M. Ghanbari. The cross-search algorithm for motion estimation. *IEEE Transcations on Communications*, 38(7): pages 950-953, 1990. [3]

L. W. Lee, J. F. Wang, J. Y. Lee, and J. D. Shie. Dynamic search-window adjustment and interlaced search for block-matching algorithm. *IEEE Trans. Circuits and Systems for Video Technology*. 3(1): pages 85-85, 1993. [4]

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*. 4(4): pages 438-442, 1994. [5]

L. M. Po and W. C. Ma. A novel four-step search algorithm for fast block motion estimation. *IEEE Transactions Circuits and Systems for Video Technology*. 6(3): pages 313-317, 1996. [6]

Video Coding for Low Bitrate Communication. ITU Telecom. Standardization sector of ITU, ITU-T Recommendation H.263, 1996. [7]

B. Zeng, R. Li and M.L. Liou. Optimization of fast block motion estimation algorithms. *IEEE Transactions Circuits and Systems for Video Technology*. 7(6): pages 833-844, 1997. [8]

Video Coding for Low Bitrate Communication. ITU Telecom. Standardization sector of ITU, Draft ITU-T Rec. H.263 Version 2, 1997. [9]

MPEG-4 Visual Fixed Draft International Standard. ISO/IEC 14 496-2, 1998. [10]

S. Zhu, K.K. Ma. A new diamond search algorithm for fast block-matching motion estimation. *IEEE Transactions Image Process*. 9(2): pages 287-290, 2000. [11]

MPEG-4VideoVerification ModelVersion 18.0. MPEGVideo Group, ISO/IEC JTC1/SC29/WG11 N3908, 2001. [12]

C. Zhu, X. Lin, L.P. Chau. An enhanced hexagonal search algorithm for block motion estimation. *In Proceedings of ISCAS* (2), pages 392-395, 2003. [13]

C. Hsieh, P.C. Lu, J.S. Shyu aud E. H. Lu. A motion estimation algorithm using inter-block correlation. *Electronic Letters,* 26(5): pages 276-277, Mar. 1990. [14]

S. Zafar, Y.Zhang and J. Baras. Predictive block-matching motion estimation for TV coding-part I: inter-block prediction. *IEEE Transactions Broadcasting*. 37(3): pages 97-101, 1991. [15]

Y. Zhang and S. Zafar. Predictive block-matching motion estimation for TV coding-part I: inter-frame prediction. *IEEE Transactions Broadcasting,* 37(3): pages 102- 105, 1991. [16]

M. Bierling. Displacement estimation by hierarchical block-matching. *SPIE, Visual Communications and Image Processing*, 1011: pages 942-951, 1988. [17]

A. N. Netravali and B. G. Haskell. *Digital Pictures: Representation and Compression.* New York: Plenum, 1988. [18]

J. W. Woods and C. H. Radewan. Kalman filtering in two dimensions. *IEEE Transactions Inform. Theory*, 23: pages 473-482, 1977. [19]

T. Wiegand, M. Lightstone, D. Mukherjee, T. G. Cambell, and S. K. Mitra. Rate- distortion optimized mode selection for very low bit rate video coding and the emerging H.263 standard. *IEEE Transactions on Circuits and Systems for Video Technology.* 6(2): pages 482–190, 1996. [20]

H. li, A. Lundmark, and R. Forchheimer. Image sequence coding at very low-bit bitrates: A review. *IEEE Transactions Image Process.*3(9): pages 589–609, 1994. [21]

B. Girod. Rate-constrained motion estimation. *Proc. SPIE Visual Commun. Image Process.* 2308: pages 1026–1034, 1994. [22]

F. Kossentini, Y.-W. Lee, M. J. T. Smith, and R. K.Ward. Predictive RD optimized motion estimation for very low bit rate video coding. *IEEE Journal on Selected Areas in Communitions*. 15(6): pages 1752–1763, 1997. [23]

D. T. Hoang, P. M. Long, and J. S. Vitter. Efficient cost measure for motion estimation at low bit rate. *IEEE Transactions on Circuits and Systems for Video Technology.* 8(5): pages 488–500, 1998. [24]

M. C. Chen and A. N.Willson. Rate-distortion optimal motion estimation algorithm for motion-compensated transform video coding. *IEEE Transactions on Circuits and Systems for Video Technology.* 8(3): pages 147–158, 1998. [25]

M. Z. Coban and R. M. Mersereau. A fast exhaustive search algorithm for rate-constrained motion estimation. *IEEE Transactions Image Process.*, 7(5): pages 769–773, 1998. [26]

J. C. H. Ju, Y. K. Chen, and S. Y. Kung. A fast rate-optimized motion estimation algorithm for low-bit rate video coding. *IEEE Transactions on Circuits and Systems for Video Technology*. 9(7): pages 994–1002, 1999. [27]

Y. Y. Sheila and S. Hemami. Generalized rate-distortion optimization for motion-compensated video coders. *IEEE Transactions on Circuits and Systems for Video Technology.* 10(6): pages 942–955, 2000. [28]

M. T. Orchard and G. J. Sullivan. Overlapped block motion compensation: An estimation-theoretic approach. *IEEE Transactions Image Process.* 3(5): pages 693–699, 1994. [29]

J. K. Su and R. M. Mersereau. Motion estimation methods for overlapped block motion compensation. *IEEE Transactions Image Process.* 9(6): pages 1509–1521, 2000. [30]

Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 ISO/IEC 14496-10AVC) Joint Video Team (JVT). Doc. JVT-G050, 2003. [31]

T. Wiegand, G. Sullivan, G. Bjøntegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology.*13(7): pages 560–576, Jul. 2003. [32]

C. K. Chui, and G. Chen. Kalman Filtering with Real-Time Applications. Springer Series in Information Sciences, 17: New York: Springer, 1987. [33]

S. Haykin. Adaptive Filter Theory, Prentice-Hall Inc., Englewood Cliffs, 1991. [34]

J. M. Mendel. Lessons in Digital Estimation Theory for signal processing, communications, and control. Prentice-Hall Inc., Englewood Cliffs, 1995. [35]

M. S. Grewal, and A. P. Andrews. Kalman Filtering Theory and Practice, Prentice-Hall Inc., Englewood Cliffs, 1993. [36]

C. M. Kuo, C. H. Hsieh, H. C. Lin and P. C. Lu. A new motion estimation method for video coding. *IEEE Transactions Broadcasting*. 42(2): pages 110-116, 1996. [37]

J. Kim and J. W. Woods. 3D Kalman Filter for Image Motion Estimation. *IEEE Transactions Image Processing*, 7: pages 42-52, 1998. [38]

D. Angwin and H. Kaufman. Image restoration using a reduced order model. *IEEE Transactions on Signal Processing*, 16: pages 21-28, 1989. [39]

C. M. Kuo, C. H. Hsieh, H. C. Lin and P. C. Lu. Motion Estimation Algorithm with Kalman Filetring. *IEE Electronic Letters*, 30(15): pages 1204-1206, 1994. [40]

R. K. Mehra. On the identification of variance and adaptive Kalman filtering. *IEEE Transactions Automat. Contr.* 15: 1970. [41]

R. A. HOWARD, .Systems Analysis of Semi-Markov Processes. *IEEE Transactions on Military Electronics*, 8(2): pages 114-124, 1964. [42]

M. T. Orchard and G. J. Sullivan. Overlapped block motion compensation: an estimation-theoretic approach. *IEEE Transactions Image Processing*, 3(5): pages 693-699, 1994. [43]

J. K. Su, R. M. Mersereau. Motion estimation methods for overlapped block motion compensation. *IEEE Transactions Image Processing*, 9(9): pages 1509-1521, 2000. [44]

T. Wedi, H.G. Musmann. Motion- and aliasing-compensated prediction for hybrid video coding. *IEEE Transactions of Circuits and Systems for Video Technology*, 13(7): pages 577 – 586, 2003.[45]

T. Wedi. Adaptive interpolation filter for motion compensated prediction. *In proceedings of the 2002 IEEE International Conference on Image Processing*. 2: pages 22-25, 2002.[46]

C. M. Kuo, S. C. Chung, P. Y. Shih. Kalman Filtering Based Rate-Constrained Motion Estimation for Very Low Bit Rate Video Coding. *IEEE Transactions Circuit and Systems for Video Technology,* 16(1): pages 3-18, 2006.[47]

C. M. Kuo, C. P. Chao and C. H. Hsieh. An Efficient Motion Estimation Algorithm for Video Coding Using Kalman Filter. *Real-Time Imaging*, 8, pp.253-264, 2002.[48]

C. M. Kuo, C. P. Chao and C. H. Hsieh. A New Motion estimation algorithm for Video Coding Using Adaptive Kalman Filter. *Real-Time Imaging*, 8, pp.387-398, 2002.[49]

IntechOpen



Kalman Filter Recent Advances and Applications

Edited by Victor M. Moreno and Alberto Pigazo

ISBN 978-953-307-000-1
Hard cover, 584 pages
Publisher InTech
Published online 01, April, 2009
Published in print edition April, 2009

The aim of this book is to provide an overview of recent developments in Kalman filter theory and their applications in engineering and scientific fields. The book is divided into 24 chapters and organized in five blocks corresponding to recent advances in Kalman filtering theory, applications in medical and biological sciences, tracking and positioning systems, electrical engineering and, finally, industrial processes and communication networks.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Nai-Chung Yang, Chaur Heh Hsieh and Chung Ming Kuo (2009). Kalman Filtering Based Motion Estimation for Video Coding, Kalman Filter Recent Advances and Applications, Victor M. Moreno and Alberto Pigazo (Ed.), ISBN: 978-953-307-000-1, InTech, Available from: http://www.intechopen.com/books/kalman_filter_recent_adavnces_and_applications/kalman_filtering_based_motion_estimation_for_video_coding

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2009 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen