VLSI Implementation of the Kalman Filter for Image Restoration

Jin Yun Zhang
Department of Electrical Engineering,
University of Ottawa,
Ottawa, Ontario, K1N 6N5

Heng Da Cheng
School of Computer Science,
Technical University of Nova Scotia,
Halifax, Nova Scotia, B3J 2X4

Abstract

In this paper, parallel VLSI implementation of the Kalman filter is developed for digital image restoration. The original image model and the degradation model are both represented by Roesser's 2-D SISO state-space models, a simple composite dynamic structure based on a cascade technique is obtained. From this composite model, the Kalman filtering equations are established by defining a proper state vector. To speed up the recursive estimation procedure, the computational independence between the samples along the diagonal lines in a first-quadrant system is examined and the diagonal filtering scheme is used. Finally, a dedicated VLSI architecture for high speed applications is proposed, it results in estimation of one pixel in each multiplication/addition time period.

1 Introduction

The Kalman filter has been one of the most widely applied techniques. There are many successful applications, such as adaptive controls, signal processing, target tracking and communications. Recently, considerable effort has been devoted to the application of the 1-D recursive Kalman filter to restore the noisy and blurred 2-D images. By using a state space model, the unknown state vector is estimated recursively for each new observation. It has been shown that the Kalman filter is an optimal estimator for image restoration.

There are several problems in extending the standard 1-D recursive state filtering techniques to the 2-D case, such as: how to establish a suitable 2-D recursive model by defining a proper state vector; how to reduce the dimensionality of the resulting state vectors by reasonable approximation; how to speed up the Kalman filtering procedure by processing signals in parallel.

In recent years, image modeling and reduction of the orders of the Kalman filter for image restoration have been received considerable attention. Several different filtering schemes, such as line-by-line filtering, vector filtering, strip filtering and block filtering, have been proposed [1-9]. Woods and Radewan [4] have proposed two 2-D Kalman processors for the images degraded by noise. One, called the strip processor, updates a line at a time; the other, called the reduced update Kalman filter (RUKF), is a scalar processor. The RUKF scheme was shown to offer significant reduction in the total computation load. Angwin and Kaufman [6] proposed the reduced order model Kalman filter (ROMKF), which is based on a low-order state-space model of an image. The low dimension of this system results in decreased computation times. In [7], Suresh and Shenoi proposed the Kalman strip filtering with modeling the blur by a 2-D state-space structure. Wu [8] employed three-dimensional state-space models to develop another strip filtering model for the degraded image with a nonsymmetric half-plane support. Later, Azimi-Sadjadi and Wong [9] presented a two-dimensional block Kalman filtering. This 2-D block state-space model takes into account the correlations of the image data in successive neighboring blocks and reduces the edge effects. However, the optimal Kalman filter for strip observations as well as the one for the block observations are characterized by complexities and large computational requirements.

The processing of a Kalman filter requires matrix/vector operations such as multiplication, addition, subtraction, and inversion. Among these, matrix inversion is the most difficulty to implement. Especially in the 2-D case, the dimensionalities of the

matrices are much larger than the 1-D case. Fortunately, with the rapid development of VLSI integrated circuits, it is feasible to implement Kalman filters by parallel array architectures. Recently, VLSI implementations of the Kalman filters have been given by several authors [10-12]. However, it is still a problem to realize the Kalman filter for image restoration by simple means and at real-time speed.

In this paper, a diagonal 2-D Kalman filtering scheme which models both the image and the linear spatial invariant (LSI) blur by 2-D state-space structures is proposed. We start with a brief review of digital image restoration. Next, for simplifying the filtering procedure, 2-D state-space structures for autoregressive (AR) image generation model and LSI blur model with the quarter-plane region of support are introduced. Then, these two state-space structures are cascaded to form a composite state-space dynamic model and the Kalman filter equations are established. Finally, to achieve a real-time processing speed, an efficient VLSI implementation with the diagonal scheme is presented.

2 Image Restoration

In many practical imaging situations, blur is a distortion which affects the image in a deterministic manner, while noise is a stochastic phenomenon which corrupts the image. Therefore, the observed image pixels for a blurred and noisy image can be modeled as the output of a linear filter, which is described by

$$y(m,n) = \Sigma_{i,j\in R_1} h_{ij}(m,n) f(m-i,n-j) + v(m,n) \quad (1)$$

where $h_{ij}(m,n)$ represents the space-varying degradation function or point spread function (PSF) with support denoted by R_1; $f(m,n)$ represents the uncorrupted image of size $M \times N$; $y(m,n)$ is the observed image; and $v(m,n)$ represents the observation noise. The noise which is a stochastic phenomenon, in most practical situation may be considered to be white Gaussian. For LSI blur, (1) can be written as:

$$y(m,n) = \Sigma_{i,j\in R_1} h_{ij} f(m-i,n-j) + v(m,n) \quad (2)$$

The original image can be modeled as an autoregressive Gauss-Markov process driven by uncorrelated Gaussian white noise, described by

$$f(m,n) = \Sigma_{k,l\in R_2} c_{kl} f(m-k,n-l) + u(m,n) \quad (3)$$

In this equation, $f(m,n)$ represents the original image; c_{kl} represents space-invariant model coefficients; $u(m,n)$ is the noise process accounting for the error in the model; and R_2 represents the support region. The block diagram representation of the input-output model of the degraded images is given in Fig.1.

The image restoration problem is to estimate $f(m,n)$ from the observed image $y(m,n)$ according to some optimality criterion. There are two cases, one is given the PSF of the LSI blur and some statistical knowledge of the noise as well as image coefficients obtained from some available set of similar images. To use a recursive spatial domain estimator, i.e. the Kalman filter, we incorporate the blurred model and image model into a state dynamic model. The Kalman equations are then established and the system matrices are derived according to the PSF and the image coefficients. In other case, image parameters are not available. It is necessary to consider the implementation of an adaptive 2-D estimator to identify and estimate the images. This paper is concerned with the former case.

3 State-Space Modeling

3.1 Image Generation Model

In order to use Kalman filter to restore the image, we first have to know the correlations between pixels of the original image. When the previous pixels are estimated, the current pixel can be predicted based on the correlation. Then, using the current observation, we correct the prediction to get the optimal estimation.

The image generation model has been proposed by several authors. Basically, the image is modeled as an autoregressive Gaussian-Markov process driven by uncorrelated Gaussian white noise [4,5]. In [7,8], Suresh and Azimi-Sadjadi modeled images as a causal quarter-plane vector AR process. However, the correlation between pixels which lie on the same column in the strip is not incorporated. Here, the Roesser's 2-D state-space model [13] will be used to represent the image AR model with a quarter-plane region of support.

Consider an image process that starts from the upper-left-hand corner of the image and then proceeds horizontally line-by-line. The current output pixel of the image process only depends on the past pixels. Assume the image pixels are produced recursively by a first- quadrant or "causal" system.

Then, we have Roesser's state equations for equation (3)

$$\begin{bmatrix} R_u(m+1,n) \\ S_u(m,n+1) \end{bmatrix} = \begin{bmatrix} A_u^{11} & A_u^{12} \\ A_u^{21} & A_u^{22} \end{bmatrix} \begin{bmatrix} R_u(m,n) \\ S_u(m,n) \end{bmatrix} + \begin{bmatrix} B_u^1 \\ B_u^2 \end{bmatrix} u(m,n); \quad (4)$$

$$f(m,n) = \begin{bmatrix} C_u^1 & C_u^2 \end{bmatrix} \begin{bmatrix} R_u(m,n) \\ S_u(m,n) \end{bmatrix} + D_u u(m,n). \quad (5)$$

where $m = 0, 1, ..., M-1, n = 0, 1, ..., N-1$. $u(m,n)$ and $f(m,n)$ are the driven noise and the output image, respectively. $R_u(m,n)$ is the horizontal state-space vector component of dimension $r_u \times 1$, and $S_u(m,n)$ is the vertical component of dimension $s_u \times 1$. $\mathbf{A}_u, \mathbf{B}_u, \mathbf{C}_u$, and $\mathbf{D}_u$ are $(r_u + s_u) \times (r_u + s_u), (r_u + s_u) \times 1, 1 \times (r_u + s_u)$, and 1×1 matrices. r_u and s_u are the number of dependent pixels respectively in the horizontal and in the vertical directions related to the current pixel. Notice that the current state vector is calculated by the following equation:

$$\begin{bmatrix} R_u(m,n) \\ S_u(m,n) \end{bmatrix} = \mathbf{A}^{10} \begin{bmatrix} R_u(m-1,n) \\ S_u(m-1,n) \end{bmatrix} + \mathbf{A}^{01} \begin{bmatrix} R_u(m,n-1) \\ S_u(m,n-1) \end{bmatrix} + \mathbf{B}^{11} \begin{bmatrix} u(m-1,n) \\ u(m,n-1) \end{bmatrix} \quad (6)$$

where

$$\mathbf{A}^{10} = \begin{bmatrix} A_u^{11} & A_u^{12} \\ 0 & 0 \end{bmatrix}, \quad (7)$$

$$\mathbf{A}^{01} = \begin{bmatrix} 0 & 0 \\ A_u^{21} & A_u^{22} \end{bmatrix}, \quad (8)$$

$$\mathbf{B}^{11} = \begin{bmatrix} B_u^1 & 0 \\ 0 & B_u^2 \end{bmatrix}. \quad (9)$$

When the image coefficients in equation (3) are found, the system matrices in (4) and (5) can be easily derived by using a signal flow graph mapping method [14]. For example, if

$$f(m,n) = c_{10} f(m-1,n) + c_{01} f(m,n-1) + c_{11} f(m-1,n-1) + u(m,n), \quad (10)$$

and the state variables are chosen as the outputs of the delay operators, as shown in the signal flow graph Fig.2, the matrices $\mathbf{A}_u, \mathbf{B}_u, \mathbf{C}_u$, and $\mathbf{D}_u$ for Roesser's model are as follows:

$$\mathbf{A}_u = \begin{bmatrix} A_u^{11} & A_u^{12} \\ A_u^{21} & A_u^{22} \end{bmatrix} = \begin{bmatrix} c_{10} & 1 \\ c_{11} & c_{01} \end{bmatrix}; \quad (11)$$

$$\mathbf{B}_u = \begin{bmatrix} B_u^1 \\ B_u^2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \quad (12)$$

$$\mathbf{C}_u = \begin{bmatrix} C_u^1 & C_u^2 \end{bmatrix} = \begin{bmatrix} c_{10} & 1 \end{bmatrix}; \quad (13)$$

$$\mathbf{D}_u = D_u = 1. \quad (14)$$

3.2 LSI Blur Model

Our aim now is to obtain a model of blurs, the deterministic degradation. 2-D state-space models for linear image processing have been proposed by several authors. Here, we will use Roesser's 2-D state-space model again since it is more general one. Let $f(m,n)$ be the scalar input, the pixel element generated by the image process mentioned above. $y(m,n)$ represents the blurred pixel at the same location. A vector $R_f(m,n)$ is defined as the horizontal state, which is assumed to convey information in the horizontal direction. Similarly, a vector $S_f(m,n)$ can be chosen as the vertical state to convey information in the vertical direction. The vectors $R_f(m,n)$ and $S_f(m,n)$ together are called as the local state since they are propagated locally and can only determine the output pixels next to them. For a infinite impulse response (IIR) system or a finite impulse response (FIR) system, the state vectors can be chosen as the outputs of the delay operators. In general Roesser's local state-space equations are:

$$\begin{bmatrix} R_f(m+1,n) \\ S_f(m,n+1) \end{bmatrix} = \begin{bmatrix} A_f^{11} & A_f^{12} \\ A_f^{21} & A_f^{22} \end{bmatrix} \begin{bmatrix} R_f(m,n) \\ S_f(m,n) \end{bmatrix} + \begin{bmatrix} B_f^1 \\ B_f^2 \end{bmatrix} f(m,n); \quad (15)$$

$$y'(m,n) = \begin{bmatrix} C_f^1 & C_f^2 \end{bmatrix} \begin{bmatrix} R_f(m,n) \\ S_f(m,n) \end{bmatrix} + D_f f(m,n). \quad (16)$$

where, $\mathbf{A}_f, \mathbf{B}_f, \mathbf{C}_f$, and $\mathbf{D}_f$ are system matrices of appropriate dimensions.

For a known linear spatial invariant blur, h_{ij}, a local state-space realization can be obtained by inspecting the relationship between the input, the output and the state vectors of the signal flow graph. Then, given values for the boundary conditions (such as all zero) and the input $f(m,n)$, the equations produce an output state vector and an output pixel value, and the image is processed recursively.

4 Kalman Filter Formulation

To implement the Kalman filter, we must incorporate the image generation model and the LSI blur

model into a composite state dynamic model, i.e. a state update equation and a state-dependent output equation. In other words, we have to determine the elements of the state vector for the whole system such that the following equations can be obtained:

$$\mathbf{x}(m+1,n) = \mathbf{A}\mathbf{x}(m,n) + \mathbf{G}\mathbf{w}(m,n) + \mathbf{H}\mathbf{u}(m,n)) \quad (17)$$

$$y(m,n) = \mathbf{C}\mathbf{x}(m,n) + v(m,n) \quad (18)$$

where $\mathbf{x}(m,n)$ represents the state vector; $\mathbf{w}(m,n)$ denotes a deterministic input vector; $\mathbf{u}(m,n)$ is a noise vector; and $\mathbf{A}, \mathbf{C}, \mathbf{G}$ and $\mathbf{H}$ are system matrices. The terms $v(m,n)$ and $y(m,n)$ have been defined previously.

As shown in Fig.3, the image generation model and the blur model are cascaded to form the whole system, and a white noise $v(m,n)$ is added as the observation noise. Let

$$R(m,n) = \begin{bmatrix} R_u(m,n) \\ R_f(m,n) \end{bmatrix} \quad (19)$$

$$S(m,n) = \begin{bmatrix} S_u(m,n) \\ S_f(m,n) \end{bmatrix} \quad (20)$$

thus, we have

$$\begin{bmatrix} R(m+1,n) \\ S(m,n+1) \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} R(m,n) \\ S(m,n) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u(m,n) \quad (21)$$

$$y(m,n) = \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} R(m,n) \\ S(m,n) \end{bmatrix} + Du(m,n) + v(m,n) \quad (22)$$

where,

$$A_{11} = \begin{bmatrix} A_u^{11} & 0 \\ B_f^1 C_u^1 & A_f^{11} \end{bmatrix}, A_{12} = \begin{bmatrix} A_u^{12} & 0 \\ B_f^1 C_u^2 & A_f^{12} \end{bmatrix}$$

$$A_{21} = \begin{bmatrix} A_u^{21} & 0 \\ B_f^2 C_u^1 & A_f^{21} \end{bmatrix}, A_{22} = \begin{bmatrix} A_u^{22} & 0 \\ B_f^2 C_u^2 & A_f^{22} \end{bmatrix}$$

$$B_1 = \begin{bmatrix} B_u^1 \\ B_f^1 D_u \end{bmatrix}, B_2 = \begin{bmatrix} B_u^2 \\ B_f^2 D_u \end{bmatrix}$$

$$C_1 = \begin{bmatrix} D_f C_u^1 & C_f^1 \end{bmatrix}, C_2 = \begin{bmatrix} D_f C_u^2 & C_f^2 \end{bmatrix}$$

$$D = D_f D_u$$

By comparing equations (17,18) and equations (21,22), we find that there are two problems to be solved. First, the input random process $u(m,n)$ appears in the observation equation of (22) but not in (18). Thus, the equations of the conventional Kalman filter gain and the error covariance matrix can not be used directly. Second, since the local state propagates horizontally and vertically, the state at $(m+1,n)$th position depends on the states of positions of (m,n) and $(m+1,n-1)$. Therefore, we modify the state vectors of the system by including $u(m,n)$ and the state vectors for (m,n) and $(m+1,n-1)$ in it. Let $k = (m,n)$, and define:

$$R(k) = \begin{bmatrix} R(m,n) \\ R(m+1,n-1) \end{bmatrix} \quad (23)$$

$$S(k) = \begin{bmatrix} S(m,n) \\ S(m+1,n-1) \end{bmatrix} \quad (24)$$

$$U(k) = \begin{bmatrix} u(m,n) \\ u(m+1,n-1) \end{bmatrix} \quad (25)$$

$$R(k+1) = \begin{bmatrix} R(m+1,n) \\ R(m+2,n-1) \end{bmatrix} \quad (26)$$

$$S(k+1) = \begin{bmatrix} S(m+1,n) \\ S(m+2,n-1) \end{bmatrix} \quad (27)$$

$$U(k+1) = \begin{bmatrix} u(m+1,n) \\ u(m+2,n-1) \end{bmatrix} \quad (28)$$

$$u_2(k) = \begin{bmatrix} u(m+1,n) \\ u(m+2,n-1) \end{bmatrix} \quad (29)$$

The state vectors are then defined as

$$\mathbf{x}(k) = \begin{bmatrix} R(k) \\ S(k) \\ U(k) \end{bmatrix} \quad (30)$$

$$\mathbf{x}(k+1) = \begin{bmatrix} R(k+1) \\ S(k+1) \\ U(k+1) \end{bmatrix} \quad (31)$$

Now, the problem left is that the state vector $\mathbf{x}(k+1)$ can not be represented in terms of $\mathbf{x}(k)$ since $R(m+2,n-1)$ and $S(m+2,n-1)$ are not related to $\mathbf{x}(k-1)$. We approximate these values by their most recent estimates with the uncertainty represented in a noise term. Let

$$\mathbf{x}_2(k) = \begin{bmatrix} R(m+2,n-1) \\ S(m+2,n-1) \end{bmatrix} \quad (32)$$

$$\mathbf{w}(k) = \begin{bmatrix} \hat{R}(m+2,n-1) \\ \hat{S}(m+2,n-1) \end{bmatrix} \quad (33)$$

Then we have:

$$\mathbf{x}_2(k) = \mathbf{w}(k) + \mathbf{w}_2(k) \quad (34)$$

where $\mathbf{w}(k)$ indicates the best available estimate of the state vector, $\mathbf{x}_2(k)$, and $\mathbf{w}_2(k)$ is the noise included to account for uncertainty in this approximation. The equations (21) and (22) are changed to:

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{G}\mathbf{x}_2(k) + \mathbf{F}u_2(k) \quad (35)$$

$$y(k) = \mathbf{C}\mathbf{x}(k) + v(k) \quad (36)$$

where

$$\mathbf{A} = \begin{bmatrix} A_{11} & 0 & A_{12} & 0 & B_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & A_{21} & 0 & A_{22} & 0 & B_2 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (37)$$

$$\mathbf{G} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}^T \quad (38)$$

$$\mathbf{F} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T \quad (39)$$

$$\mathbf{C} = \begin{bmatrix} C_1 & 0 & C_2 & 0 & D & 0 \end{bmatrix} \quad (40)$$

where ^{T} denotes the transpose of the matrix. Finally, by substituting $\mathbf{w}(k)$ and $\mathbf{w}_2(k)$ for $\mathbf{x}_2(k)$, define $u(k) = \begin{bmatrix} \mathbf{w}_2(k) & \mathbf{u}_2(k) \end{bmatrix}^T$, and $\mathbf{H} = \begin{bmatrix} \mathbf{G} & \mathbf{F} \end{bmatrix}$, the desired equations as (17) and (18) are obtained:

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{G}\mathbf{w}(k) + \mathbf{H}\mathbf{u}(k) \quad (41)$$

$$y(k) = \mathbf{C}\mathbf{x}(k) + v(k) \quad (42)$$

Therefore, we can use the conventional Kalman filtering formula to estimate the state $\mathbf{x}(k)$.

$$\hat{\mathbf{x}}(k|k-1) = \mathbf{A}\hat{\mathbf{x}}(k-1|k-1) + \mathbf{G}\mathbf{w}(k) \quad (43)$$

$$\mathbf{P}(k|k-1) = \mathbf{A}\mathbf{P}(k-1|k-1)\mathbf{A}^T + \mathbf{H}\mathbf{Q}_u(k-1)\mathbf{H}^T \quad (44)$$

$$K(k) = \mathbf{P}(k|k-1)\mathbf{C}^T\{\mathbf{C}\mathbf{P}(k|k-1)\mathbf{C}^T + Q_v(k)\}^{-1} \quad (45)$$

$$\mathbf{P}(k|k) = \{\mathbf{I} - K(k)\mathbf{C}\}\mathbf{P}(k|k-1) \quad (46)$$

$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + K(k) \times \{y(k) - \mathbf{C}\hat{\mathbf{x}}(k|k-1)\} \quad (47)$$

$$\hat{f}(k) = \begin{bmatrix} C_u^1 & C_u^2 \end{bmatrix} \begin{bmatrix} \hat{R}_u(k) \\ \hat{S}_u(k) \end{bmatrix} \quad (48)$$

where $\mathbf{P}(k)$ is the error covariance matrix and $K(k)$ is the Kalman filter gain. $\mathbf{Q}_u(k) = E\{\mathbf{u}(k)\mathbf{u}^T(k)\}, Q_v(k) = E\{v(k)v^T(k)\}$, and $\hat{f}(k)$ is the optimal estimate of the image.

5 Parallel VLSI Implementation

In this section, we propose a VLSI Kalman filter architecture for image restoration. As shown above, the Kalman filter is computationally intensive since in each estimation, many matrix operations are performed. In our scheme, the filter is a scalar processor. The dimension of $\{\mathbf{C}\mathbf{P}(k-1|k-1)\mathbf{C}^T + \mathbf{Q}_v\}^{-1}$ is 1×1. As a result, the matrix inversion is avoided. However, matrix-matrix multiplications and matrix-vector multiplications still prevent the potential application of this Kalman filter to high speed processing.

In order to accelerate the Kalman filter processing speed, highly parallel VLSI systolic structures have been proposed by several authors to perform the Kalman filtering [10-12]. Here, based on the analysis of our image model, we will give a very fast implementation architecture.

First, the matrices of the Kalman filter equations are highly sparse and in effect, not all components of the state vector $\mathbf{x}(k)$ have to be updated. Only the state components for pixel at (m, n) need to be calculated, while the state components for pixel at $(m+1, n-1)$ are already generated. Therefore, the computations for equations (43) and (47) can be reduced. Due to the same reason, the implementations of (44), (45) and (46) can also be simplified.

Second, for the first-quadrant image model, the pixels along the diagonal lines are computationally independent. They can be processed concurrently. Thus, to achieve high speed and high utilization of the systolic array structures, the diagonal scanning method is used [15,16]. That is, image is divided into horizontal strips. The width of the strip, W, depends on the number of delays between the input and the output of the implementation. Let us consider the process of one diagonal segment within one strip, which starts from the nth line. As shown in Fig.4, when the state vector of pixel $(m-1, n)$, $\mathbf{x}(m-1, n) = \begin{bmatrix} R(m-1, n) & S(m-1, n) & u(m-1, n) \end{bmatrix}^T$, enters the array processor, the prediction of the state component, $\hat{R}(m, n)$ can be calculated. At the same time, $\hat{S}(m, n)$ is generated based on the initialized or previous estimated values of the state vectors of pixel $(m, n-1)$, $\mathbf{x}(m, n-1)$. Next, $\mathbf{x}(m-2, n+1)$ is ready to enter the array for producing $\hat{R}(m-1, n+1)$. And, at same time, the state vector $\mathbf{x}(m-1, n)$ can be used to calculate the vertical component $\hat{S}(m-1, n+1)$, and so on. Therefore, for each pixel, the horizontal and the vertical state components can be computed simultaneously. And for the pixels along the diagonal lines, they can be processed concurrently or sequentially without waiting time.

Third, the Kalman gain evaluation is determined by the model parameters and the statistics of the random process, $\mathbf{Q}_u$ and Q_v. Hence, $K(k)$ can be calculated before estimation is carried out since it

does not depend on the measurements [17]. Furthermore, since the Kalman gain converges to a steady-state solution after a certain number of iteration, it can be evaluated off-line until it converges and then can be used in the real-time state estimation.

Although, in (45), gains are updated recursively as the estimation proceeds. To implement (44), (45) and (46), it is time consumable since they can not be performed in parallel. Fortunately, when the diagonal estimation scheme is used, the gain generated at kth iteration will not be used by the next iteration. It will be used by the processing of the pixel at the same line of the next segment. That is, the gain $K(k)$ is related to $K(k+W)$ not to $K(k+1)$. Therefore, the Kalman gains, $K(k), K(k+1), \ldots K(k+W-1)$ can be calculated in parallel based on the results of the previous diagonal segment.

The implementation of this Kalman filtering consists of two parts: the state vector update part and the error covariance and Kalman filter gain calculation part.

The state update part is shown in Fig.5, which performs (43),(47) and (48). In this design, we assume that vertical and horizontal state vectors for the boundary are initialized to zeroes. A simple systolic array is used to generate the predicted estimate of the state vectors. Next, a one-dimensional array is used to obtain the measured error between the observation value, $y(m,n)$, and the predicted value. Then another 1-D array is followed to correct the estimated states to the optimal values.

To begin the operations, the state vector of pixel $(m-1,n)$, $\mathbf{x}(m-1,n) = [\ R(m-1,n)\ \ S(m-1,n)\ \ u(m-1,n)\ ]^T$, enters the array, the prediction of the state component, $\hat{R}(m,n)$ can be calculated. At the same time, $\hat{S}(m,n)$ is generated by the initialized or previous estimated values of the state vectors of pixel $(m,n-1)$, $\mathbf{x}(m,n-1)$. Then, $\mathbf{x}(m-2,n+1)$ enters the array for producing $\hat{R}(m-1,n+1)$. And, the state vector $\mathbf{x}(m-1,n)$ propagates to the right for calculating $\hat{S}(m-1,n+1)$, and so on. When the boundary of the strip is reached, the generated state vectors have to be stored in buffers for processing the next strip. The estimated state vectors of the last strip stored in the buffers are popped to the array for processing the next segment. It takes one cycle, which is equal to W clock times, to process one diagonal segment. It is clear that every cycle one initial state vector for the first line of the strip needs to be retrieved from the storage, one generated state vector for the last line has to be stored. Notice that in this design at every clock one pixel is estimated and the clock period depends on the time in which the $K(k)$ can be generated.

The implementation of the Kalman filter gain calculation part is given in Fig.6. It is a direct realization of (44), (45) and (46). We use the rectangular systolic arrays and one dimensional systolic arrays to implement the matrix-matrix multiplications, matrix-vector multiplications, and divisions. Note that the latency for this implementation is not small. But, the inputs for one segment can be pipelined into the array, the outputs come out from the array at every clock time. When the delay of the state update part is equal to the delay of the gain calculation part, one pixel can be estimated at every clock time. It results in real-time processing.

In the proposed Kalman filter array processors, besides the delay and the multiplex, there is one division processing element. All the others are the basic processing elements, that is, they perform the multiplication and accumulation operations. After the initialization, the processor utilization efficiency is 100%.

6 Conclusions

This paper gives a real-time implementation of 2-D Kalman filtering for image restoration. The Roesser's 2-D local state space model is used to represent the image process and the blur process. As a result, a simple procedure to establish the Kalman filter equations is obtained. This scalar filtering algorithm provides a computationally feasible procedure for the restoration of large images. To realizing the Kalman filtering procedure, a VLSI systolic array structures is presented. For higher speed and higher utilization of this processor, a diagonal scanning method is suggested. The proposed filter scheme can be extended to the causal image model and the causal blur with nonsymmetric half-plane support.

7 References

1. N.E. Nasi and T. Assefi, "Bayesian recursive image estimation", *IEEE Trans. Comput.*, vol.C-21, pp.734-738, July 1972.

2. A. O. Aboutalib and L. M. Silverman, "Restoration of motion degraded images", *IEEE Trans. Circuits and Systems*, vol.CAS-22, pp.278-286, Mar. 1975.

3. A.O. Aboutalib, M. S. Murphy, and L. M. Silverman, "Digital restoration of images degraded by general motion blur", *IEEE Trans. Automat. Contr.*, vol.AC-22, pp.294-302, June 1977.

4. J. W. Woods and C. H. Rademan, "Kalman filtering in two dimensions", *IEEE Trans. Information Theory*, vol.IT-23, pp.473-482, July 1977.

5. J. W. Woods and V. K. Ingle, "Kalman filtering in two dimensions: Further results", *IEEE Trans. Acoust., Speech, Signal Processing*, vol.ASSP-30, pp.188-197, April 1981.

6. D. L. Angwin and H. Kaufman, "Image restoration using reduced order models", *Signal Processing* (North-Holland), vol.16, pp.21-28, 1989.

7. B. R. Suresh and B. A. Shenoi, "New results in two-dimensional Kalman filtering with applications to image restoration", *IEEE Trans. Circuits and Systems*, vol.CAS-28, pp.307-319, April 1981.

8. Z. Wu, "Multidimensional state-space model Kalman filtering with application to image restoration", *IEEE Trans. Acoustics, Speech, Signal Processing*, vol.ASSP-33, pp.1576-1592, December 1985.

9. M. R. Azimi-Sadjadi and P. W. Wong, "Two-dimensional block Kalman filtering for image restoration", *IEEE Trans. Acoustics, Speech, Signal Processing*, vol.ASSP-35, pp.1736-1749, December 1987.

10. T. Y. Sung and Y. H. Hu, "Parallel VLSI implementation of the Kalman filter", *IEEE Trans. Aerospace, Electronics Systems*, vol.AES-23, no.2, pp.215-224, March 1987.

11. S. Y. Kung, "VLSI array processors", Prentice Hall, 1988.

12. H. G. Yeh, "Systolic Implementation on Kalman filters", *IEEE Trans. Acoustics, Speech, Signal Processing*, vol.ASSP-36, no.9, pp.1514-1517, September 1988.

13. R. P. Roesser, "A discrete state-space model for linear image processing", *IEEE Trans. Automatic Control*, vol.AT-20, no.1, pp.1-10, Feb. 1975.

14. D. E. Dudgeon and R. M. Mersereau, "Multidimensional digital signal processing", *Prentice-Hall, Inc.* , Englewood Cliffs, New Jersey 1984.

15. T. Aboulnasr and W. Steenaart, "Real-time systolic array processor for 2-D spatial filtering", *IEEE Trans. Circuits and Systems*, vol.CAS-35, no.4, pp.451-455, April 1988.

16. J. Y. Zhang and W. Steenaart, "VLSI implementation of high speed two-dimensional state-space recursive filtering", *Proceedings of International Symposium on Circuits and Systems*, pp.1099-1102, May 1989.

17. S. M. Bozic, "Digital Kalman filtering", Edward Arnold (Publishers) Ltd., 1979.

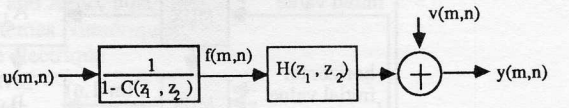

Fig. 1 Block diagram representation of the image generation model and the degradation model.

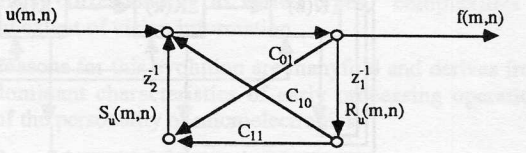

Fig.2 The flow graph of AR image model for first-quadrant support region with first order.

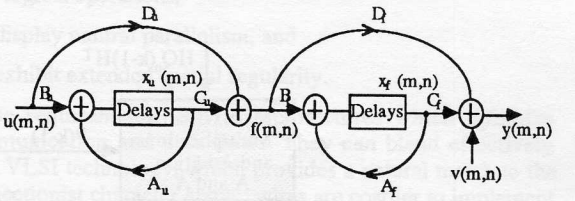

Fig. 3 The cascade of two 2-D state space models. The first one is for image generation and the second one is for the LSI blur model.

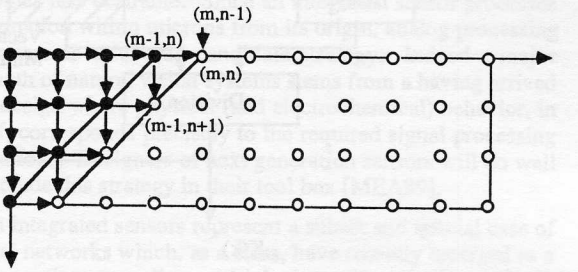

Fig. 4 The diagonal estimate process within each horizontal strip.(W = 4)

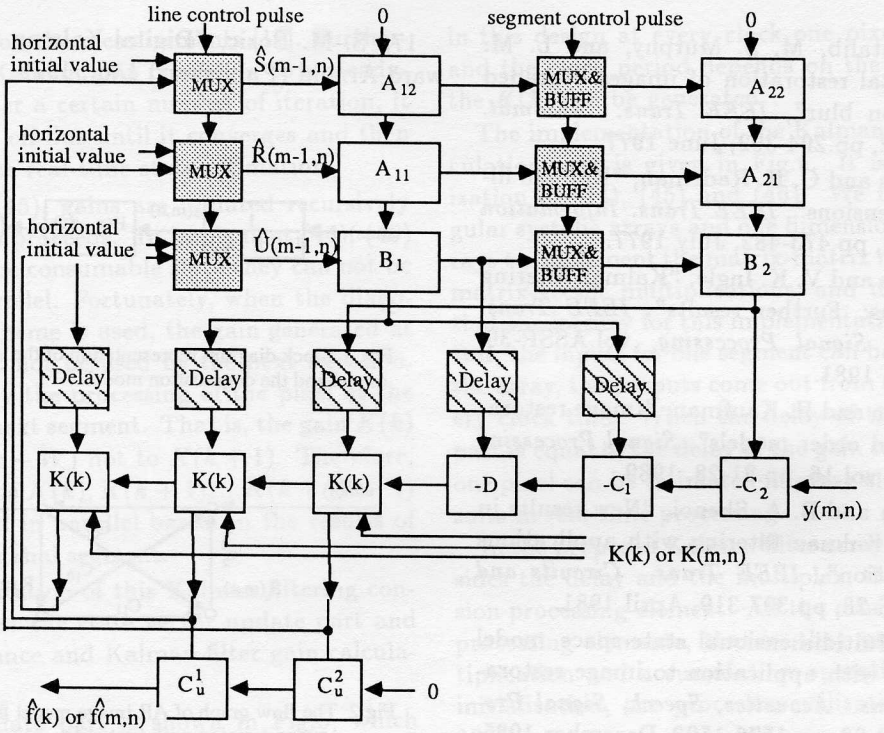

Fig. 5 The architecture for state update part of the Kalman filtering.

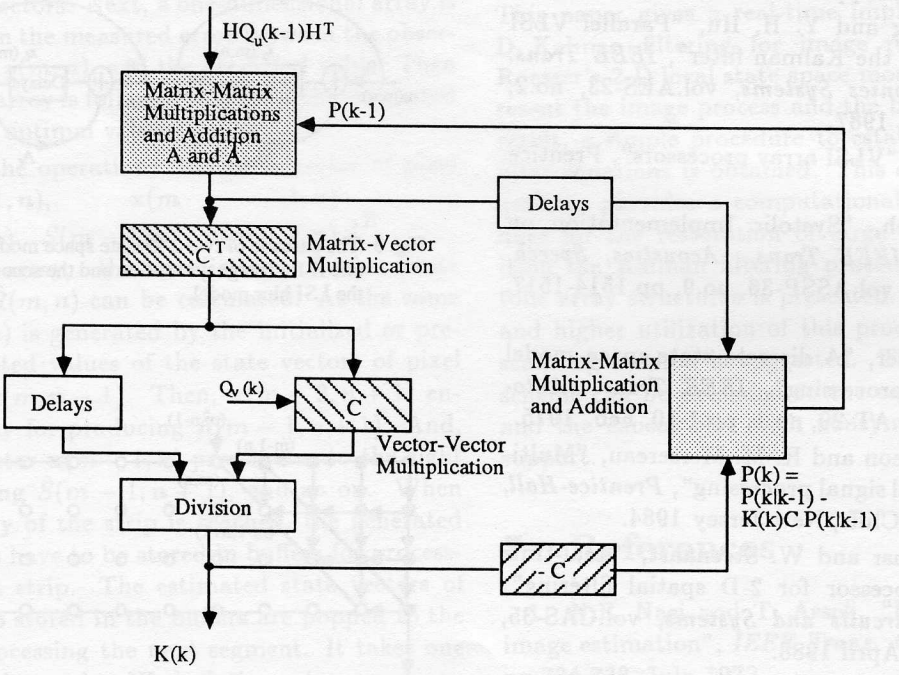

Fig. 6 The implementation of the error covariance and Kalman gain calcluation. All the processing blocks are systolic arrays.