Kalman Filtering and Systolic Processors

Hen-Geul Yeh

California State University, Long Beach, U.S.A.

Abstract

A new parallel computing structure of the systolic array type for implementing the Kalman filter for state-space estimation is presented. This corresponds to the adjustment of Kalman filter algorithms to the type of the Faddeev algorithm. The Faddeev algorithm provides a wide range of matrix computational capabilities and it maps easily into a concurrent systolic array architecture. The special arrangement for the data flow mapping from Kalman filter to Faddeev algorithm, and from Faddeev algorithm to architecture are presented.

Introduction

The Kalman filter has been successfully applied to many signal processing applications, including target tracking, adaptive controls, radar signal processing, on-board calibration of inertial systems, and in-flight estimation of aircraft controls and failure-proof systems [1-6]. The applicability of the Kalman filter to real-time processing problems is generally limited, however, by the filter's relative computational complexity [5, 6, 7]. In this paper, therefore, the Kalman filter is considered for parallel processing, especially in the systolic array processor which may be implemented as special chips by using VLSI technology.

First of all, the Kalman filter algorithm is modified to be the type of Faddeev algorithms [8, 9, 10]. The Faddeev algorithm is easily systematized for matrix calculations in a way that is highly desirable from an architectural point of view. On the other hand, the discrete Kalman filter is based on linear algebraic operations, i.e.; matrix-vector and matrix-matrix multiplications and additions. It is natural to arrange discrete Kalman filter algorithms into a form of Faddeev algorithms to maximize the capability of hardware implementation of systolic arrays. One of the more important features of this proposed algorithm is that it avoids the matrix inverse computation in the discrete Kalman filter algorithms and obtains the values of desired results directly at the end of the forward course of the computation, resulting in a considerable savings in added processing and storage. Secondly, since the Gaussian elimination procedure is applied through the computation, the numerical stability is obtained. Thirdly, the parallel, modular computer architecture which is consisted of systolic processor provides simultaneous high throughput and a capability for a wide variety of linear algebraic operations.

In the following sections, we describe, in order, the Kalman filter, the Faddeev algorithm, the proposed implementation, and the 2-Dimensional systolic array architecture.

Kalman Filter

Kalman filters have been shown to be the optimal linear estimator in the least square sense for estimating dynamic system states in linear systems. The Kalman filter updates state estimation based on prior estimates and observed measurements. It consists of the model of the dynamic process which performs the function of prediction and a feedback correction scheme. The measurements can be processed as they occur, and there is no need to store any measurement data. However, all the associate matrices which describe the system dynamic, measurement system, and noises are assumed to be known. Conventional discrete time-varying Kalman filtering process involves the propagation of state estimates and error covariance matrices from time sample to next time sample. Discussions and applications on Kalman filter can be found in many literature [1-6].

The following equations define a general dynamic system and a measurement system.

$$x(k+1) = \Phi(k)x(k) + w(k) \qquad (1)$$

$$z(k) = H(k)x(k) + v(k) \qquad (2)$$

$\Phi(k)$ is a nxn matrix called the state transition matrix which describes the plant. $x(k)$ is the state vector with n-dimension. $w(k)$ is a p-vector called the disturbance. $z(k)$ is a m-vector termed the measurement vector, $v(k)$ is also a m-vector called the measurement noise vector. $H(k)$ is a mxn matrix called the

* This work was supported by the Vice-President's Office for Faculty and Staff under the P.O.P. Program, California State University, Long Beach.

measurement matrix. The disturbance w(k) and noise v(k) are assumed to be zero mean Gaussian white noise sequences with covariance matrices Q(k) and R(k), respectively. Furthermore, sequences w(k) and v(k) are assumed to be statistically independent. The Kalman filter is described by the following equations under assumptions that matrices $\Phi(k)$, $\Gamma(k)$, $H(k)$, $R(k)$, and $Q(k)$ are known.

$$\hat{x}(k/k-1) = \Phi(k-1)\hat{x}(k-1/k-1) \tag{3}$$

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

$$P^{-1}(k/k) = P^{-1}(k/k-1) + H^T(k)R^{-1}(k)H(k) \tag{5}$$

$$K(k) = P(k/k)H^T(k)R^{-1}(k) \tag{6}$$

$$\Delta z(k) = z(k) - H(k)\hat{x}(k/k-1) \tag{7}$$

$$\hat{x}(k/k) = \hat{x}(k/k-1) + K(k)\Delta z(k) \tag{8}$$

$k = 1, 2, - - -$

Initial conditions are given as follows:

$$\hat{x}(0/0) = 0$$

$$P(0/0) = P(0) \quad \text{or} \quad P^{-1}(0/0) = P^{-1}(0)$$

Note that matrices P(k/k) and P(k+1/k) are commonly called error covariance matrices. Equations (3) and (4) are referred as time updates and equations (5), (6), and (8) are referred as measurement updates. These filter equations are computed in order as listed from equations (3) to (8).

Faddeev Algorithm

The Faddeev algorithm can be illustrated by the simple case of finding **CX+D**, given **AX=B**, where **A**, **B**, **C**, and **D** are known matrices, and **X** is an unknown matrix. If these matrices are written in the form

$$\begin{array}{c|c} A & B \\ \hline -C & D \end{array}$$

and one adds a suitable combination of the rows of **A** and B to **-C** and **D**, or

$$\begin{array}{c|c} A & B \\ \hline -C+WA & D+WB \end{array} \tag{9}$$

where **W** specifies the appropriate linear combination, such that the lower left hand side of (9) is zero, then **CX+D** will appear on the lower right hand side, e.g.;

$$\begin{array}{c|c} A & B \\ \hline 0 & CX+D \end{array}$$

This results because we have $\mathbf{W=CA^{-1}}$ so that $\mathbf{D+WB=D+CA^{-1}B}$, or since **AX=B**, **D+WB=D+CX**. As shown in Figure 1, numerous matrix operations such as multiplication, addition, and linear system solution are possible by selective entries in the four quadrants. In a sense, then, the Faddeev algorithm is programmable by simply positioning the data appropriately before calculations begin.

$$\begin{array}{c|c} A & I \\ \hline -I & 0 \end{array} \rightarrow A^{-1} \qquad \begin{array}{c|c} A & B \\ \hline -I & 0 \end{array} \rightarrow A^{-1}B$$

$$\begin{array}{c|c} I & B \\ \hline -C & 0 \end{array} \rightarrow CB \qquad \begin{array}{c|c} A & B \\ \hline -C & D \end{array} \rightarrow CA^{-1}B + D$$

$$\begin{array}{c|c} I & B \\ \hline -C & D \end{array} \rightarrow D + CB$$

Figure 1. Examples of variety of matrix operations possible with Faddeev algorithm.

The simplicity of the algorithm is due to the absence of a necessity to actually identify the multipliers of the rows of A and the elements of B; it is only necessary to "annul the last row." This can be done by orthogonal triangularization, a numerically stable procedure, combined with an equally stable Gaussian elimination procedure.

One of the more important features of this algorithm is that it avoids the usual backsubstitution or solution to the triangular linear system and obtains the values of the unknowns directly at the end of the forward course of the computation, resulting in a considerable savings in added processing and storage.

The Proposed Implementation

Kalman filter algorithms are adjusted to be the type of Faddeev algorithm in this section. Computations are cyclically propagated through the following ordered set of passes. Note that the new data could be shifted into the array from the top, row by row as the calculation proceeds, so that there would be no delay in starting the next matrix computation.

1st pass:

$$\begin{array}{r|l} I & \hat{x}(k-1/k-1) \\ \hline -\Phi(k-1) & 0 \end{array} \rightarrow \begin{array}{r|l} I & \hat{x}(k-1/k-1) \\ \hline 0 & \hat{x}(k/k-1) \end{array}$$

2nd pass:

$$\begin{array}{r|l} P^{-1}(k-1/k-1) & \Phi^T(k-1) \\ \hline -\Phi(k-1) & Q(k-1) \end{array} \rightarrow \begin{array}{r|l} P^{-1}(k-1/k-1) & \Phi^T(k-1) \\ \hline -\Phi(k-1) & P(k/k-1) \end{array}$$

41. 2. 2

3rd pass:

$$\begin{array}{c|c} R(k) & I \\ \hline -H^T(k) & 0 \end{array} \rightarrow \begin{array}{c|c} R(k) & I \\ \hline -H^T(k) & H^T(k)R^{-1}(k) \end{array}$$

4th pass:

$$\begin{array}{c|c} P(k/k-1) & I \\ \hline -I & 0 \end{array} \rightarrow \begin{array}{c|c} P(k/k-1) & I \\ \hline -I & P^{-1}(k/k-1) \end{array}$$

5th pass:

$$\begin{array}{c|c} I & H(k) \\ \hline -H^T(k)R^{-1}(k) & P^{-1}(k/k-1) \end{array} \rightarrow \begin{array}{c|c} I & H(k) \\ \hline -H^T(k)R^{-1}(k) & P^{-1}(k/k) \end{array}$$

6th pass:

$$\begin{array}{c|c} P^{-1}(k/k) & I \\ \hline -H^T(k)R^{-1}(k) & 0 \end{array} \rightarrow \begin{array}{c|c} P^{-1}(k/k) & H^T(k)R^{-1}(k) \\ \hline -I & K(k) \end{array}$$

7th pass:

$$\begin{array}{c|c} I & \hat{x}(k/k-1) \\ \hline H(k) & z(k) \end{array} \rightarrow \begin{array}{c|c} I & \hat{x}(k/k-1) \\ \hline H(k) & \Delta z(k) \end{array}$$

8th pass:

$$\begin{array}{c|c} I & \Delta z(k) \\ \hline -K(k) & \hat{x}(k/k-1) \end{array} \qquad \begin{array}{c|c} I & \Delta z(k) \\ \hline -K(k) & \hat{x}(k/k) \end{array}$$

Note that the results (lower right quadrant) of each pass in general, need to be stored and are used in later passes as new entries.

Architecture

A general architecture is proposed to process all eight passes as mentioned in the previous section. To fit the type of Faddeev algorithm, the number of columns of the matrix (lower left quadrant) should be less than or equal to that of the matrix (upper left quadrant). Fortunately, all eight passes meet this requirement so that the matrix (lower left quadrant) will be annulled row by row. This can be done by the ordinary Gaussian elimination. However, Gaussian elimination in general requires pivoting, and the pivoting stratagy is not suited to a systolic array since it may require global communication for pivot section. The neighbor pivoting technique introduces a zero to a row by subtracting a multiple of an adjacent row from it, interchanging the two rows when necessary to prevent the multiple from exceeding unity [11]. This process is shown in the modified Faddeev algorithm.

Modified Faddeev Algorithm:

$$\begin{array}{c|c} A & B \\ \hline -C & D \end{array} \rightarrow \begin{array}{c|c} T & MB \\ \hline -C + WT & D + WMB \end{array}$$

where T is upper triangular, and M is nonsingular (triangularization) matrix. Since $W = CT^{-1}$, the final result is $G = D + WMB$. In order to triangularize matrix A and to annul matrix C, it is necessry to have a two-step procedure. First, A is triangularized by neighbor pivoting process (simultaneously applied to B); second, C is annulled by Gaussian elimination using the diagonal elements of T as pivot elements. The square processor arrangement for both triangularization and annulling steps is shown in figure 2 [8, 9]. Note that if A is identity matrix (as shown in the passes 1, 5, 7, and 8), then the triangularization process can be omitted. However, to compute all 8 passes, the size of the processor is 2n cells (row) by 2n cells (column).

Summary

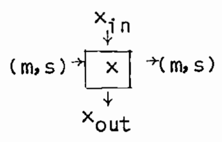
A novel scheme of implementing Kalman filtering algorithms on a 2-dimensional systolic array is presented. This systematized matrix calculation is based on the type of Faddeev algorithm. The calculation is performed with two steps; they are triangularization process, and annulling process. The 2-dimensional array of PEs, connected in a nearest neighbor mesh, which provides a high performance matrix computing capability and could be easily expandable for large size Kalman filtering operations.

Boundary Cell:

$$\begin{tikzcd} x_{in} \arrow[d] \\ \textcircled{x} \arrow[r] & (m, s) \end{tikzcd}$$

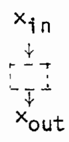
```
if $|x_{in}| > |x|$, then
    $s = 1$
    $m = -x/x_{in} (x_{in} \neq 0)$
    $m = 0$
    $x = x_{in}$
else
    $s = 0$
    $m = -x_{in}/x$
    $x = x$
```

Internal Cell

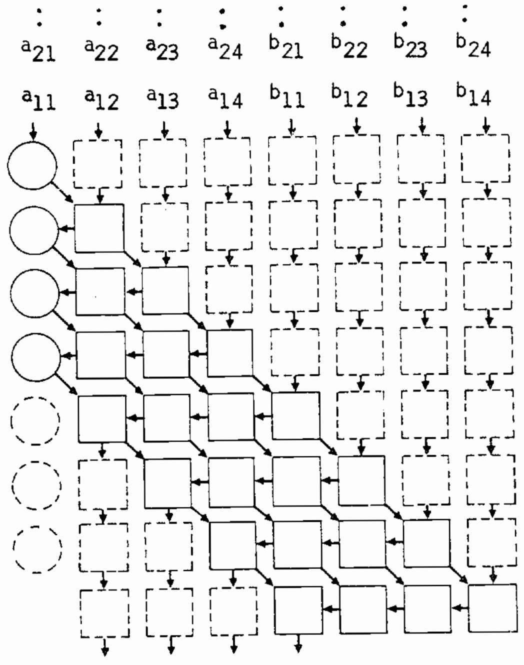


```
if s = 1, then
    $x_{out} = x + m \cdot x_{in}$
    $x = x_{in}$
else
    $x_{out} = x_{in} + m \bullet x$
    $x = x$
```

Delay Cell



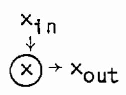
$x_{out} = x_{in}$



a_{21} a_{22} a_{23} a_{24} b_{21} b_{22} b_{23} b_{24}
a_{11} a_{12} a_{13} a_{14} b_{11} b_{12} b_{13} b_{14}

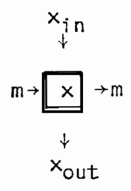
(a)

Boundary Cell



$x_{out} = x_{in}/x$

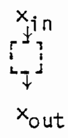
Internal Cell



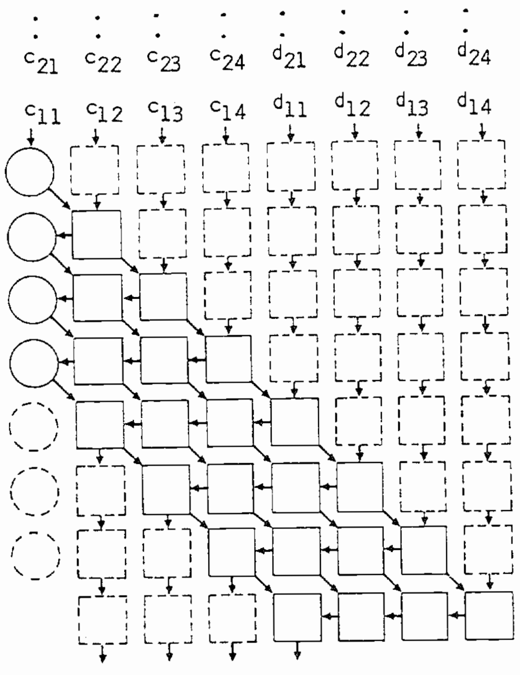
$X_{out} = x_{in} - mx$

x_{in}

Delay Cell



$x_{out} = x_{in}$



c_{21} c_{22} c_{23} c_{24} d_{21} d_{22} d_{23} d_{24}
c_{11} c_{12} c_{13} c_{14} d_{11} d_{12} d_{13} d_{14}

output matrix

(b)

Figure 2. (a) Data flow and PE arrangement for triangularization process, n=4.

(b) Data flow and PE arrangement for annulling process, n=4.

REFERENCES

1. A. Gebb, "Applied Optimal Estimation," The M.I.T. Press, 1974.
2. J. S. Meditch, "Stochastic Optimal Linear Estimation and Control," McGraw-Hill,
3. H. G. Yeh, "A Design Method For Failure-Proof Systems," Proc. of the 1983 American Control Conference.
4. H. G. Yeh, "Techniques for the Detection, Estimation, Distinction, and Compensation of Failures in Linear System," Ph.D. Dissertation, U. C. Irvine, 1982.
5. H. W. Sorenson, "Parameter Estimation," Marcel Dekker, 1980.
6. A. E. Bryson and Y. C. Ho, "Applied Optimal Control," Rev. ed., Halsted Press, Div. of John Wiley & Sons, 1975.
7. S. Y. Kung, H. J. Whitehouse, and T. Kailath, "VLSI and Modern Signal Processing," pp. 375-388, Prentice-Hall, 1985.
8. J. G. Nash and S. Hansen, "Modified Faddeev Algorithm for Matrix Manipulation," Proc. 1984 SPIE Conf.
9. J. G. Nash, "A Systolic/Cellular Computer Architecture for Linear Algebraic Operations," Proc. of the IEEE 1985 ICRA.
10. H. T. Kung, R. Sproull, and G. Steele, "VLSI Systems and Computations," Computer Science Press, 1981, pp. 367-378.
11. W. M. Gentleman and H. T. Kung, "Matrix Triangularization by Systolic Arrays," SPIE Vol. 298, Real-Time Signal Processing IV, 1981.