第10卷第5期
1993年10月

控制理论与应用
CONTROL THEORY AND APPLICATIONS

Vol. 10, No. 5
Oct., 1993

脉动阵列执行 Kalman 滤波计算

慕德俊　戴冠中
(西北工业大学自动控制系·西安,710072)

摘要：本文从并行的观点出发,综述了用脉动(Systolic)阵列实现标准 Kalman 滤波、平方根滤波、平方根信息滤波和扩展 Kalman 滤波计算,文中还分析了各种阵列结构的大小,计算速度和处理器的利用率.

关键词：脉动阵列；Kalman 滤波；计算步

1 引　言

六十年代出现的 Kalman 滤波理论,已经成功地在信号处理、目标跟踪、自适应控制、导航系统等方面得到了广泛的应用,虽然它是一种有效的信号处理工具,但在滤波方程中,每个采样间隔矩阵计算的复杂性一般为 $O(n^3)$,使得滤波的实时性受到了限制. 随着 VLSI 集成度的大幅度提高,芯片功能的不断增强和价格的急剧下降,使得并行处理技术得到了应用[1]. 通过使用 $O(n^2)$ 个处理单元,可将滤波计算的复杂性降为 $O(n)$. Carnegie-Mellon 大学的 Kung 和他的助手们首先提出了脉动式阵列结构[2],这种结构是由一组相互连接的单元组成,按规则的几何形状排列,每个单元能完成若干简单的操作[3],单元的连接是固定的局部连接,信息以流水线方式在各单元之间流动,与外界通讯只发生在"边界"单元上,因而在实时计算中,往往比通用的计算机具有更高的效率[4],并特别适合于具有局部相关关系的线性递推运算象 Kalman 滤波计算. 本文通过多种阵列结构执行滤波计算来分析各个结构的特点.

2 实现滤波计算所需基本的脉动阵列结构

在滤波计算中需四种基本的矩阵计算,即回代、相乘、正交分解和 Faddeev 运算,许多文献中都对实现这些算法的阵列结构做过介绍,这里只简述如下.

2.1 矩阵相乘

若对于矩阵 A, B, C, D 有 $D=AB+C$,其递推关系可描述为 $A_{ij}=C_{ij}+\sum_{k=1}^{n}A_{ik}B_{kj}$. Bernard[5]给出了实现此算法的阵列结构. 如果 B 为三角矩阵或矢量,则仅需三角阵列或一维的线性阵列来实现此算法[4].

2.2 回代计算

对于 n 阶线性方程组 $Ax=B$,若对 A 进行三角分解使方程变为 $Rx=y$,则求解方程就变成一个回代计算问题,其中 R 为上三角阵,x, B 和 y 都为矢量. 递推关系描述为 $x_i=(y_i-\sum_{j=i+1}^{n}R_{ij}x_j)/R_{ii}$. 实现此算法的阵列结构有多种. Travassos[6]给出了其中的一种.

本文于1992年3月28日收到. 1992年10月30日收到修改稿.

2.3 正交三角分解

正交三角分解在数值上具有较好的稳定性，有多种分解算法，Chen[7]列出了 Givens 旋转法、修正的 Givens 旋转法[10]和修正的 Gram Schmidt 算法，并给出了相应的三角阵列结构，各单元的功能和算法的复杂性. 本文以 Givens 旋转法为基本的正交分解(QR)方法. 如果 R 为上三角阵，A 为满矩阵. 通过 QR 分解可完成 $Q\begin{bmatrix}R\\A\end{bmatrix}=\begin{bmatrix}R'\\0\end{bmatrix}$ 的变换，为了减少计算步，可预先将 R 阵存放在三角阵列中，然后将 A 阵输入阵列[8].

2.4 Faddeev 算法

设 A,B,C,D 为四个 n 阶方阵，A 为非奇异阵，有如下形式 $F=\begin{bmatrix}A & B\\-C & D\end{bmatrix}$ 构成一个 $2n\times 2n$ 阶方阵. 对 F 阵进行变换，使 TA 变为上三角阵，$-C$ 变为 0 阵，即 $\begin{bmatrix}A & B\\-C & D\end{bmatrix}\rightarrow\begin{bmatrix}TA & TB\\-C+WTA & D+WTB\end{bmatrix}\rightarrow\begin{bmatrix}TA & TB\\0 & D+CA^{-1}B\end{bmatrix}$，即可得到 $D+CA^{-1}B$. 若对 A,B,C,D 赋以不同的初值，或将 B,C,D 改为向量，则可进行更多的矢量、矩阵运算. 实现 Faddeev 算法采用梯形阵列结构[4].

3 Faddeev 算法实现滤波计算

设被估计系统是下列的线性离散时间系统[9]：

$$x(k+1)=A(k)x(k)+B(k)u(k)+W(k), \tag{1}$$

$$y(k)=C(k)x(k)+V(k). \tag{2}$$

其中 $x(k)$ 为 n 维状态矢量，$y(k)$ 为 m 维量测矢量，$A(k)$ 是 $n\times n$ 阶状态转移阵，$W(k)$ 是 n 维的系统噪声向量，$C(k)$ 是 $m\times n$ 阶输出阵，$V(k)$ 是 m 维的量测噪声向量. 系统噪声 $\{W(k)\}$ 与测量噪声 $\{V(k)\}$ 是互不相关的、零均值具有方差为 $Q(k)$，$R(k)$ 的白噪声序列，$u(k)$ 是 r 维的非随机控制向量，$B(k)$ 是 $n\times r$ 阶的一步控制阵. 则滤波方程为

$$\hat{x}(k+1|k)=A(k)\hat{x}(k)+B(k)u(k), \tag{3}$$

$$\hat{x}(k+1)=\hat{x}(k+1|k)+K(k+1)[y(k+1)-C(k+1)\hat{x}(k+1|k)], \tag{4}$$

$$P(k+1|k)=A(k)P(k)A^{\mathrm{T}}(k)+Q(k), \tag{5}$$

$$K(k+1)=P(k+1)C^{\mathrm{T}}(k+1)R^{-1}(k+1), \tag{6}$$

$$P(k+1)=[I-K(k+1)C(k+1)]P(k+1|k), \tag{7}$$

$$P^{-1}(k+1)=P^{-1}(k+1|k)+C^{\mathrm{T}}(k+1)R^{-1}(k+1)C(k+1). \tag{8}$$

应用 Faddeev 算法可分 9 步计算上述方程(3)～(7)，若将方程(7)改为(8)则可用 8 步执行一次滤波计算[10,11]，尤其是在无验前信息的情况下，$P^{-1}(k+1)$ 可初始化为零，这时方程(8)更为有效. 其 8 步的 Faddeev 计算如下表：

表 1 Faddeev 算法执行滤波计算

步	A	B	$-C$	D	$D+CA^{-1}B$
1	I	$\hat{x}(k)$	$-A(k)$	$B(k)u(k)$	$\hat{x}(k+1\|k)$
2	$P^{-1}(k)$	$A^{\mathrm{T}}(k)$	$-A(k)$	$Q(k)$	$P(k+1\|k)$

(续上表)

步	A	B	$-C$	D	$D+CA^{-1}B$
3	$R(k+1)$	I	$-C^{T}(k+1)$	0	$C^{T}(k+1)R^{-1}(k+1)$
4	$P(k+1\|k)$	I	$-I$	0	$P^{-1}(k+1\|k)$
5	I	$C(k+1)$	$-C^{T}(k+1)R^{-1}(k+1)$	$P^{-1}(k+1\|k)$	$P^{-1}(k+1)$
6	$P^{-1}(k+1)$	$C^{T}(k+1)R^{-1}(k+1)$	$-I$	0	$K(k+1)$
7	I	$\hat{x}(k+1\|k)$	$C(k+1)$	$y(k+1)$	$\Delta y(k+1)$
8	I	$\Delta y(k+1)$	$-K(k+1)$	$\hat{x}(k+1\|k)$	$\hat{x}(k+1)$

如果只用一个 Faddeev 梯形阵列完成上述 8 步计算[12]，则它的处理单元固定为$\frac{(3n+1)n}{2}$个，一般需要存贮器将处理器的输出结果 $D+CA^{-1}B$ 存贮，用作下次计算的输入，在有效地组织数据传送的情况下完成一次滤波计算需 $16n$ 个计算步(定义边界单元完成一次运算为一个计算步)，处理器的利用率约为 80%. 如果每步计算都单独使用一个 Faddeev 梯形阵列[13]，则每个阵列的大小可以调整，其框图见图 1. 图中的回路①与②的计算可同时进行. 通过有效的组织阵列间的数据传送，执行一次滤波计算需 $6n+m$ 个计算步，但处理器的利用率仅为 28%左右.

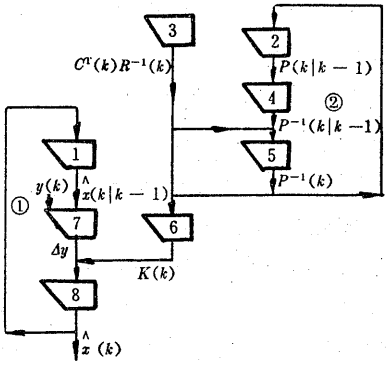

图 1　Feddeev 算法执行 Kalman 滤波

4　平方根滤波及阵列实现

以上的滤波计算中，由于使用了迭代方法进行矩阵求逆，随着计算量的增加和计算中的舍入误差，使 $P(k)$和 $P(k|k-1)$的值容易失去非负定和 $K(k)$的失真，造成滤波发散. 如果 $P(k)$和 $P(k|k-1)$用矩阵的平方根形式来代替，至少保证了它们在计算过程中的非负定性，因此滤波的稳定性得到了提高，另一方面这种平方根形式的滤波算法很容易映射到脉动阵列结构上，下面按照阵列结构从简单到复杂的顺序给予讨论.

4.1　平方根协方差滤波(Ⅰ)

Morf[14]提出的平方根协方差滤波算法，包含着对滤波和预测误差协方差阵的两个正交变换，对方程(1)，(2)有

$$\hat{x}(k+1|k)=A(k)\hat{x}(k)+B(k)u(k), \tag{9}$$

$$[A(k)P^{1/2}(k)\quad Q^{1/2}(k)]Q_1(k)=[P^{1/2}(k+1|k)\quad 0], \tag{10}$$

$$V_e(k+1)=C(k+1)P(k+1|k)C^{T}(k+1)+R(k+1), \tag{11}$$

$$\hat{x}(k+1)=\hat{x}(k+1|k)+P(k+1)C^{T}(k+1)V_e^{-1}(k+1)[y(k+1)-C(k+1)\hat{x}(k+1|k)], \tag{12}$$

$$Q_2(k)\begin{bmatrix} R^{T/2}(k+1) & 0 \\ P^{T/2}(k+1|k)C^{T}(k+1) & P^{T/2}(k+1|k)\end{bmatrix}$$

$$= \begin{bmatrix} V_*^{T/2}(k+1) & V_*^{-1/2}(k+1)C(k+1)P(k+1|k) \\ 0 & P^{T/2}(k+1) \end{bmatrix}. \tag{13}$$

其中 $P^{T/2}(k)$, $Q^{T/2}(k)$, $R^{T/2}(k)$为上三角阵, $Q_1(k)$和 $Q_2(k)$为两个正交矩阵. Andrew[15]和Jover[16]提出的阵列结构只能实现量测修正的计算, Sung[17]提出了用一组脉动阵列实现上述的滤波算法, 其方块图见图 2, 图中的大方框完成矩阵的相乘运算, 小方框执行向量的加减运算, 右边的梯形块完成 Faddeev 运算, 左边的两个三角块执行 QR 分解, 根据方程(13)中初始矩阵稀疏的特点, 将 $P^{T/2}(k+1|k)C^T(k+1)$, $P^{T/2}(k+1|k)$按照每行从下向上的顺序输入三角阵列, 使得三角化的过程中 $P^{T/2}(k+1|k)$始终保持上三角阵, 这样可减少计算步, 并能将三角阵列截成梯形阵列. 为了节省计算步, 还可将某些已知矩阵预先存入阵列中, 整个结构中所需的处理单元为 $(5n^2+6mn+2m^2+4m+n+4)/2$, 每次迭代计算所需的计算步为 $2m+3n$, 处理单元的利用率较低, 约为 13%～15%. 但此方法说明了一种复杂的算法可以用一组简单的阵列结构来实现, 并使数据流能快速平稳地流动, 充分利用算法的结构特点, 可将阵列简化, 将预测器与修正器解耦, 每次迭代需 $3n$ 个计算步[36].

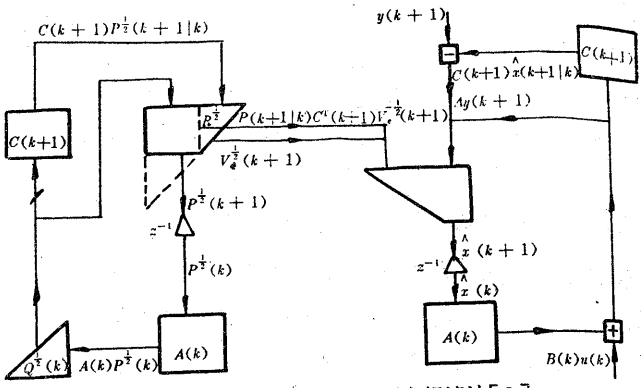

图 2 Systolic阵列实现平方根滤波[Ⅰ]

4.2 平方根协方差滤波(Ⅱ)[18,20]

对于方程(1), (2), Morf[14]还提出了一种平方根协方差滤波(SRCF)形式和一种平方根协方差预测(SRCP)形式. SRCF 方程如下:

$$Q_1(k)\begin{bmatrix} P^{T/2}(k)A^T(k)C^T(k+1) & P^{T/2}(k)A^T(k) \\ Q^{T/2}(k)C^T(k+1) & Q^{T/2}(k) \\ R^{T/2}(k+1) & 0 \end{bmatrix} = \begin{bmatrix} V_*^{T/2}(k) & V_*^{-1/2}(k)C(k+1)P(k+1|k) \\ 0 & P^{T/2}(k+1) \\ 0 & 0 \end{bmatrix}, \tag{14}$$

$$\hat{x}(k+1) = A(k)\hat{x}(k) + B(k)u(k) + P(k+1|k)C^T(k+1)V_*^{-1}(k)[y(k+1) - C(k+1)\{A(k)\hat{x}(k) + B(k)u(k)\}], \tag{15}$$

$$V_*^{1/2}(k)V_*^{T/2}(k) = C(k+1)A(k)P(k)A^T(k)C^T(k+1) + C(k+1)Q(k)C^T(k+1) + R(k+1),$$

$$P(k+1|k) = A(k)P(k)A^T(k) + Q(k).$$

SRCP 方程为:

$$Q_2(k)\begin{bmatrix} P^{T/2}(k|k-1)C^T(k) & P^{T/2}(k|k-1)A^T(k) \\ R^{T/2}(k) & 0 \\ 0 & Q^{T/2}(k) \end{bmatrix} = \begin{bmatrix} V_*^{T/2}(k) & V_*^{-1/2}(k)C(k)P(k|k-1)A^T(k) \\ 0 & P^{T/2}(k+1|k) \\ 0 & 0 \end{bmatrix}, \tag{16}$$

$$\hat{x}(k+1|k) = A(k)\hat{x}(k|k-1) + B(k)u(k) + A(k)P(k|k-1)C^{T}(k)V_{\cdot}^{-1}(k)[y(k) - C(k)\hat{x}(k|k-1)], \tag{17}$$

$$V_{\cdot}^{1/2}V_{\cdot}^{T/2} = C(k)P(k|k-1)C^{T}(k) + R(k).$$

对于一个具有两个量测的三阶系统，实现 SRCF 滤波计算的 Systolic 阵列结构[19]如图 3. 阵列在执行运算的过程中，可分四步进行，第一步对(14)式进行正交三角分解；第二步在阵列上面两行执行 Faddeev 运算来计算 $\hat{x}(k+1)$. 在 2.4 所述的 Faddeev 算法中，A,B,C,D 阵的分配如下：

$$\hat{x}^{T}(k+1) = \underbrace{[A(k)\hat{x}(k) + B(k)u(k)]^{T}}_{D} + \underbrace{[y(k+1) - C(k+1)\hat{x}(k+1|k)]^{T}}_{C(-\Delta y^{T})} \cdot \underbrace{[V_{\cdot}^{T/2}]^{-1}}_{A^{-1}} \underbrace{[P(k+1|k)C^{T}(k+1)V_{\cdot}^{-T/2}]^{T}}_{B}.$$

第三步在菱形阵列和下部的三角形阵列中进行乘法运算，其中下三角部分的输出作为下次迭代计算的输入；菱形部分的输出在第四步与其它矢量相加时也作为下次迭代计算的输入，这样就保证了数据流的快速平稳地流动. 由此方法构成的阵列处理器共需$[(n+m)(n+m+1)+2(n+1)]/2$个处理单元，每次迭代需 $3n+2m+1$ 个计算步，处理器的利用率约为 33%. 这种结构说明：一种算法可以通过几种不同的阵列分层实现，这样可减少全局通讯. 对于 SRCP 算法实现的阵列结构与图 3 相同，只是输入输出的数据流不同，执行一次循环计算需 $2n+2m+1$ 个计算步.

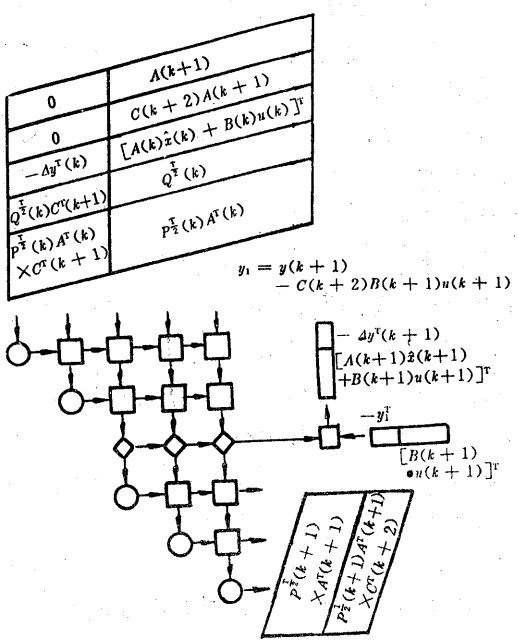

图 3　实现 SRCF 析阵列结构

4.3　平方根信息滤波(Ⅰ)[21,24]

Dyer 和 McReynolds[22]提出了一种信息滤波算法，它是以滤波和预测两种形式出现的，在此基础上 Paige[23,25]提出了平方根信息滤波算法，它更适合于在阵列结构上并行计算，平方根信息预测方程(SRIP)为

$$Q_1^{T}(k)\begin{bmatrix} P^{-1/2}(k|k-1) & 0 & P^{-1/2}(k|k-1)\hat{x}(k|k-1) \\ R^{-1/2}(k)C(k) & 0 & R^{-1/2}(k)y(k) \\ Q^{-1/2}(k)A(k) & -Q^{-1/2}(k) & -Q^{-1/2}(k)B(k)u(k) \end{bmatrix} = \begin{bmatrix} R_1^{T} & -R_1^{-1}A^{T}(k)Q^{-1}(k) & R_1^{-1}r \\ 0 & P^{-1/2}(k+1|k) & P^{-1/2}(k+1|k)\hat{x}(k+1|k) \\ 0 & 0 & * \end{bmatrix}, \tag{18}$$

这里 $P^{-1/2},R^{-1/2},Q^{-1/2}$ 为上三角阵，

$$R_1R_1^{T} = P^{-1}(k|k-1) + C^{T}(k)R^{-1}(k)C(k) + A^{T}(k)Q^{-1}(k)A(k),$$

$$r = P^{-1}(k|k-1)\hat{x}(k|k-1) + C^{T}(k)R^{-1}(k)Y(k) - A^{T}(k)Q^{-1}(k)B(k)u(k).$$

* 表示无关紧要的项.

平方根新息滤波方程(SRIF)为

$$Q_2^{\mathrm{T}}(k)\begin{bmatrix} P^{-\frac{1}{2}}(k) & 0 & P^{-\frac{1}{2}}(k)\hat{x}(k) \\ Q^{-\frac{1}{2}}(k)A(k) & -Q^{-\frac{1}{2}}(k) & -Q^{-\frac{1}{2}}(k)B(k)u(k) \\ 0 & R^{-\frac{1}{2}}(k+1)C(k+1) & R^{-\frac{1}{2}}(k+1)y(k+1) \end{bmatrix}$$

$$=\begin{bmatrix} R_1^{\mathrm{T}} & -R_1^{-1}A^{\mathrm{T}}(k)Q^{-1}(k) & R_1^{-1}r \\ 0 & P^{-\frac{1}{2}}(k+1) & P^{-\frac{1}{2}}(k+1)\hat{x}(k+1) \\ 0 & 0 & * \end{bmatrix}. \tag{19}$$

这里 $R_1R_1^{\mathrm{T}}=P^{-1}(k)+A^{\mathrm{T}}(k)Q^{-1}(k)A(k)$,

$r=P^{-1}(k)\hat{x}(k)-A^{\mathrm{T}}(k)Q^{-1}(k)B(k)u(k)$.

* 为无关紧要项.

实现 SRIP 的三角阵列结构见图 4,根据(18)式初始矩阵的特点,可预先将 $P^{-\frac{1}{2}}(k|k-1)$和 $P^{-\frac{1}{2}}(k|k-1)\hat{x}(k|k-1)$存入相应的阵列单元内,经过 $4n+m$ 个计算步的计算后,$P^{-\frac{1}{2}}(k+1|k)$和 $P^{-\frac{1}{2}}(k+1|k)\hat{x}(k+1|k)$在阵列中得到. 为了求出 $\hat{x}(k+1|k)$,可采用 2.2 所述的回代计算的方法. 图 4 的结构共需 $2n^2+3n+1$ 个处理单元,迭代周期为 $4n+m$ 个计算步,处理器的利用率为 17%～30%之间.

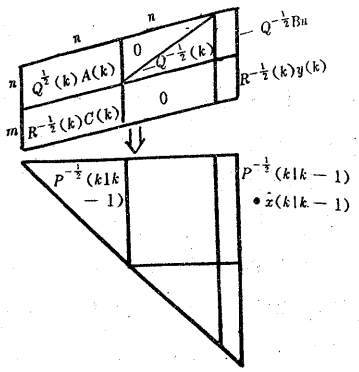

图 4 SRIP 的三角结构

从(18)式可以看出:初始矩阵为稀疏阵,中间列仅有一个上三角矩阵$-Q^{-\frac{1}{2}}(k)$,在 Givens 旋转时,若将 $Q^{-\frac{1}{2}}(k)A(k)$,$-Q^{-\frac{1}{2}}(k)$,$-Q^{-\frac{1}{2}}(k)B(k)u(k)$的每行按照由下到上的顺序输入阵列,则可将图 4 的三角阵列"截"成图 5 所示的梯形阵列,在图 5 中 $P^{-\frac{1}{2}}(k+1|k)$和 $P^{-\frac{1}{2}}(k+1|k)\hat{x}(k+1|k)$不需存在阵列的下部,而是直接进入了下次迭代的过程中,这就使得每次迭代的计算步减少到 $3n+m$,处理单元减少到$\frac{3n(n+1)}{2}$,处理器的利用率提高到 22%～42%(注:$P^{-\frac{1}{2}}(k|k-1)$,$P^{-\frac{1}{2}}(k|k-1)\hat{x}(k|k-1)$的初始阵预先存在阵列中).

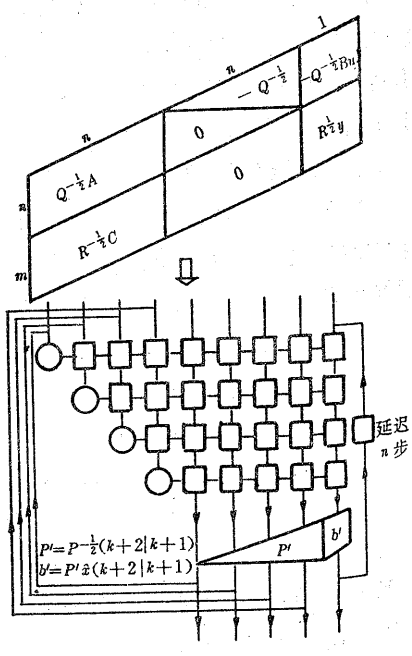

图 5 实现 SRIP 的梯形结构

在图 5 中,当初始矩阵的第一列被处理时,阵列的中间部分没有被使用,通过把阵列的方形部分"迭加"在三角部分上,可使所需的阵列结构缩减为方形(见图 6),图中用方框里带圆圈表示的对角单元有两种运算方式,最初作为三角分解的边界单元,当 $Q^{-\frac{1}{2}}A$ 的最后一行通过时,它们的存贮器置零,$-Q^{-\frac{1}{2}}$通过时,作为内部单元进行运算. 在 $Q^{-\frac{1}{2}}A$ 的旋转参数 $\cos\alpha$,$\sin\alpha$ 被计算后,从左向右通过 $n+1$ 个单元,$Q^{-\frac{1}{2}}A$ 阵和上三角阵 $Q^{-\frac{1}{2}}$之间的零矩阵保证了

旋转参数能正确地与$-Q^{-\frac{1}{2}}$矩阵进行相乘计算. 尽管每次迭代的计算步为$3n+m$,但使用的处理单元减为$n(n+1)$,处理器的利用率可达 33%～63%. 如果考虑到每次计算状态矢量所需的n个单元则整个结构使用了$n(n+2)$个单元. 由此可见使处理器利用率提高,必须考虑初始矩阵的特殊结构. 对于平方根信息滤波(SRIF)方程的阵列实现与图 6 相同,只是输入、输出的数据流不同.

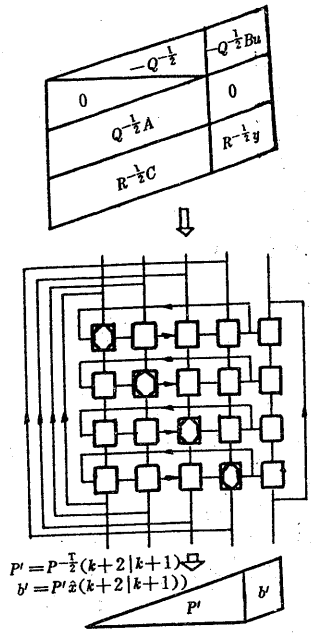

图 6　实现 SRIP 的方形结构

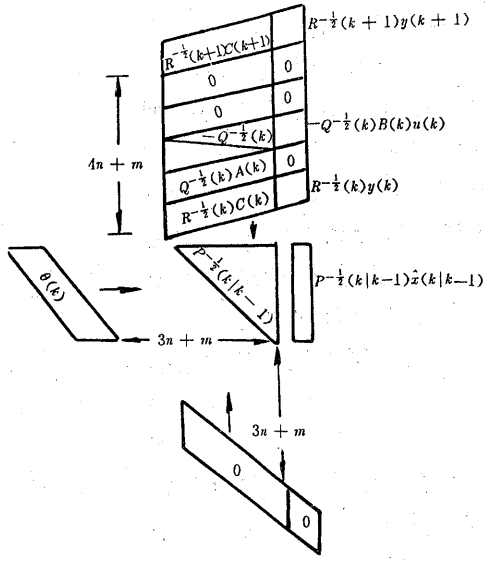

图 7　三角阵列实现平方根信息滤波

4.4　平方根信息滤波(Ⅱ)[26~28]

对于式(18),还有一种$n\times n$的三角阵列可实现其运算(见图 7). 在计算的过程中,首先通过$P^{-\frac{1}{2}}(k|k-1)$零化$R^{-\frac{1}{2}}(k)C(k)$,并将旋转参数θ存入缓冲器,$3n+m$个计算步后,$-Q^{-\frac{1}{2}}(k)$开始进入阵列,通过θ进行旋转计算,同时阵列的边界单元完成内部单元的功能,信息流转换成由下向上的方向. 同其它结构相比,这种结构使用的单元最少,为$\frac{n(n+3)}{2}$,处理器的利用率约在 68%以上,迭代周期为$4n+m$个计算步.

4.5　平方根信息滤波(Ⅲ)

Chen[29,30]提出了另一种结构(图 8)执行(19)式的正交分解(未考虑$u(k)$),这种结构可避免使用图 4 结构时,在每次迭代后对$P^{-\frac{1}{2}}(k+1|k)$和$P^{-\frac{1}{2}}(k+1|k)\hat{x}(k+1)$的提取,使它们仍保留在阵列下部的位置上,为此图 8 的结构在正交分解时

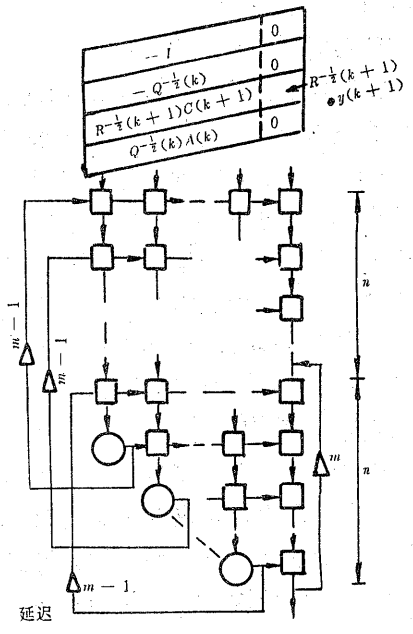

图 8　脉动阵列实现平方根信息滤波

采用了将(19)式的初始矩阵分条进行的方法，首先对 $Q^{-\frac{1}{2}}(k)A(k)$ 进行正交分解，将旋转参数反馈到上边的方形阵列并与 $-Q^{-\frac{1}{2}}(k)$ 旋转相乘，然后输入到下部的三角阵列进行正交分解. 在 $Q^{-\frac{1}{2}}(k)A(k)$ 分解完成后，将三角阵列清零，再对 $R^{-\frac{1}{2}}(k+1)C(k+1)$ 进行 QR 分解. 为了得到状态矢量 $\hat{x}(k+1)$，再对三角阵列输入 $-I$，0 阵进行 Gauss 消去计算. 本结构使用的处理单元为 $n(3n+5)/2$，迭代周期为 $3n+m$ 个计算步.

Gosling[31]等人又提出图 9 的结构是对图 8 结构的进一步改进，图 9 这种结构不需要在每次迭代中输入 $[-I,0]$ 阵来求状态矢量 $\hat{x}(k+1)$，而是通过右下角的三角阵列得到的 $P^{T/2}(k+1)$ 与☒形单元中 $P^{-\frac{1}{2}}(k+1)\hat{x}(k+1)$ 相乘来得到 $\hat{x}^T(k+1)$，这样每次迭代的周期减少为 $2n+m$ 个计算步，但这种结构多使用了 $n(n+1)/2$ 个处理单元.

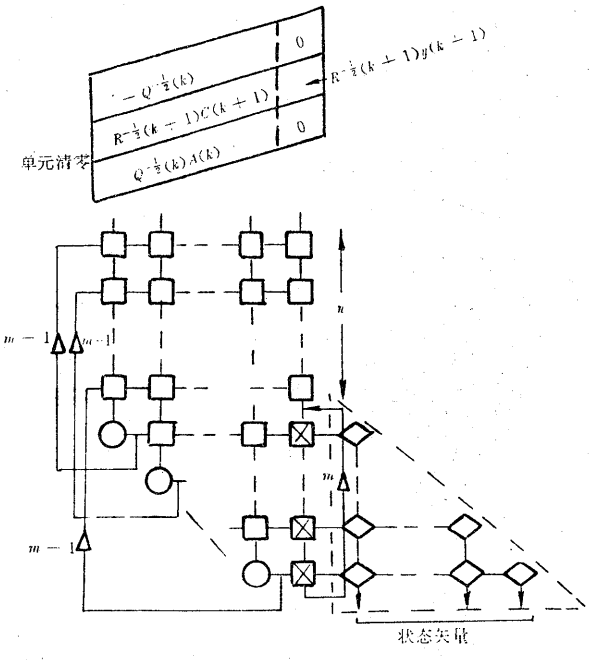

图 9 改进的结构实现平方根信息滤波

在上述实现平方根信息滤波的各种结构中，没有考虑 $Q^{-\frac{1}{2}}A$，$R^{-\frac{1}{2}}C$，$R^{-\frac{1}{2}}Y$，$-Q^{-\frac{1}{2}}$ 以及 $-Q^{-\frac{1}{2}}Bu$ 的产生，如果考虑这种情况还需要附加阵列来完成矩阵相乘以及求逆的 Cholesky 分解[27]. Gaston[31]列表比较了上述各种阵列结构的大小，处理器的利用率，以及迭代周期.

5 扩展的 Kalman 滤波

在工程实践中所遇到的具体问题，其数学模型往往是非线性的，对于这种模型，最常用的滤波算法是扩展的 Kalman 滤波算法(EKF). 为了实时计算，Baheti[32]提出了一种技术，将 EKF 算法影射到一组可编程的线性阵列上，并在 Warp 机上实现，为目标跟踪提供了应用. 为了进一步的并行计算，Chui[33]又提出了一种修正的扩展 Kalman 滤波算法(MEKF). 基于 QR 分解，奇异值分解以及 Faddeev 算法，Lu[34]推导出 MEKF 的平方根算法，并给出了并行结构. 对于 n 维状态矢量的估计，这种脉动阵列结构使用的处理单元为 $O(2n^2)$，迭代周期为 $O((\log n+17)n)$ 个计算步. 使用两个三角阵列结构可使处理器的利用率进一步提高，迭代周期缩短[35].

6 结论及展望

本文论述了近年来脉动阵列结构在执行标准 Kalman 滤波、平方根滤波、平方根信息滤波及平方根修正扩展 Kalman 滤波方面的进展，文中选用多种形式的结构，按照从简单到复杂的顺序分析了各种结构的特点以及结构的大小，处理器的利用率和迭代周期.

脉动阵列结构作为一种很有效的并行结构来执行滤波运算，已得到了许多实际的应用，但面向控制领域其它方面的并行算法的研究才刚刚开始[37]，相信随着 VLSI 的发展和

人们对新的并行算法的不断研究和对已有的串行算法的不断改进，这种脉动阵列结构及其它并行结构将在大规模的实时控制和信息处理领域得到广泛应用，给控制理论和应用带来新的变革.

参 考 文 献

[1] Mccanny, J. V.. Some Systolic Array Developments in the United Kingdom. IEEE Computer, 1987, 20(6):51—63

[2] Kung, H. T.. Why Systolic Arrays. IEEE Computer, 1982, 15(1):37—46

[3] 黄铠,F. A. 布里格斯. 并行计算机与并行处理. 北京:科学出版社,1990

[4] 陈国良,陈崚. VLSI 计算理论与并行算法. 合肥:中国科技大学出版社, 1991

[5] Bernard, S.. A Systolic Coprocessor for Matrix Computation Proceedings Working Conf. Parallel Processing. Pisa, Italy, 1988, 221—234

[6] Kung, S. Y.. VLSI Array Processors. Information and System Science Series, Prentice-Hall, 1985

[7] Chen, M. J.. On Realizations of Least-Squares Estimation and Kalman Filtering by Systolic Arrays. Proc. 1st Int Workshop Systolic Arrays, 1986, 161—170

[8] Gentleman, W. M.. Matrix Triangularizaiton by Systolic Arrays. SPIE Real-Time Signal Processing IV, 1981, 298:19—26

[9] 戴冠中,佟明安. 现代控制理论导论. 北京:国防工业出版社,1989, 225—253

[10] Yeh, H. G.. Kalman Filter and Systolic Processors. Proc ICASSP, 1986, 2139—2142

[11] Barua, S.. Systolic Implementation of Kalman Filter Using Programmable Real-Time Incoherent Matrix Multiplier, SPIE Real-Time Signal Processing. 1989, 1154:200—205

[12] Yeh, H. G.. Systolic Implementation on Kalman Filters. IEEE Trans. ASSP, 1988, 36(9):1514—1517

[13] Atherton, D. P.. Parallel Processing in Control. IEE Control Engineering Series. England, 1988, 215—232

[14] Morf, M.. Square Root Algorithms for Least Square Estimation. IEEE Trans. Automat. Contr., 1975, AC-20(4):487—497

[15] Andrews, A.. Parallel Processing of the Kalman Filter. Proc. Int. Conf. on Parallel Processing, 1981, 216—220

[16] Jover, J. M.. A Parallel Architecture for Kalman Filter Measurement Update and Parameter Estimation. Automatica, 1986, 22(1):43—57

[17] Sun, T. Y.. VLSI Implementation of Real-Time Kalman Filter. Proc. ICASSP, 1986, 2223—2226

[18] Irwin, G. W.. A systolic Architecture for Square Root Covariance Kalman Filtering. Systolic Array Processors, Prentice Hall, 1989, 561—571

[19] Gentleman, W. M.. Least Squares Computations by Givens Transformations without Square Roots. J inst Maths Applics, 1973, 13:329—336

[20] Maguire, L. P.. Transputer Implementation of Kalman Filters, IEE Pt. D, 1991, 138(4):355—362

[21] Gaston, F. M. F.. Systolic Approach to Square Root Information Kalman Filtering. Int. J. Control, 1989, 50(1):225—248

[22] Dyer, P.. Extension of Square Root Filtering to Include Process Noise J Optimation Theory and Its Applications, 1969, 3(6):444—458

[23] Paige, C. C.. Covariance Matrix Representations in Linear Filtering. Contemporary Mathematics, 1985, 309—321

[24] Gaston, F. M. F.. A Systolic Square Root Information Kalman Filter. Proc. Int. Conf. Systolic Array, 1988, 643—652

[25] Paig, C. C.. Least Squares Estimation of Discrete Linear Dynamic System Using Orthogonal Transformation. SIAM J Num. Anal, 1977, 14(2):180—193

[26] Lewis, P. S.. QR Algorithm and Array Architecture for Multichannel Adaptive Least Squares Lattice Filters. ICASSP,

1988, 2046—2048

[27] Kung, S. Y.. Systolic Array Designs for Kalman Filtering. IEEE Trans, SP, 1991, 30(1):171—182

[28] Kung, S. Y.. VLSI Array Processors. Prentice Hall, 1988

[29] Chen, M. J.. On Realization and Implementation of Kalman Filtering by Systolic Array. Proc. 21st Conf. Inform. Sci. Syst., 1987, 375—380

[30] Chen, M. J.. Systolic Kalman Filtering Based on QR Decomposition. SPIE, 826:25—32

[31] Gosling, E.. Direct Extraction of the State Vector from Systolic Implementations of the Square Root Kalman Filter. Systolic Array Processor, Prentice Hall, 1989, 42—51

[32] Baheti, R.. Mapping Extended Kalman Filters onto Linear Arrays. IEEE Trans. Automat., Contr., 1990, AC-35(2): 1310—1319

[33] Chui, C. K.. Modified Extended Kalman Filtering and a Real-Time Parallel Algorithm for System Parameter Identification. IEEE Trans. Automat. Contr., 1990, AC-35(1):100—104

[34] Lu, M.. A Parallel Square-Root Algorithm for Modified Extended Kalman Filter. IEEE Trans. on AES, 1992, 28(1): 153—162

[35]慕德俊,戴冠中. 修正的扩展 Kalman 滤波的 Systolic 算法及阵列结构. 自动化学报

[36]慕德俊,戴冠中. Systolic 阵列实现解耦 Kalman 滤波计算. 系统仿真学报,1992, 增刊:12—16

[37]胡保生,葛新科. 并行与分布控制研究的现状与未来. 控制理论与应用,1992,9(3):305—309

Systolic Implementations of Kalman Filters

MU Dejun and DAI Guanzhong

(Deparment of Automatic Control, Northwestern Polytechnical University · Xi'an, 710072, PRC)

Abstract: The paper reviews systolic implementations of standard, square-root, square-root information, square-root modified extended Kalman filtering from parallel views and analyses the number of processing cell, processor utilization and computing speed.

Key words: systolic array; Kalman filter; time-step

本文作者简介

慕德俊 1963年生. 1983年获军械工程学院导弹专业学士学位,1990年获西北工业大学惯导专业硕士学位,现为西北工业大学自动控制系博士生. 目前从事控制系统中的并行处理研究.

戴冠中 见本刊1993年第1期第12页.