第 6 卷第 4 期　　电路与系统学报　　Vol.6 No.4
2001 年 12 月　　JOURNAL OF CIRCUITS AND SYSTEMS　　December, 2001

文章编号:1007-0249(2001) 04-078-05

长序列信号快速相关及卷积的算法研究*

虞湘宾，　毕光国

(东南大学 无线电工程系, 江苏 南京 210096)

摘要：文章通过对快速傅立叶变换（FFT）的算法原理分析，根据线性相关和卷积的数学特征及物理含义，针对长序列信号，提出了一种基于 FFT 的长序列快速相关及卷积算法，用 C++进行了算法编程，在计算机上得到较好的实验效果，提高了运行速度.并结合算术傅立叶变换进行了改进。

关键词：快速傅立叶变换；快速相关；快速卷积；算术傅立叶变换

中图分类号：TN914.5　　文献标识码：A

1　引言

随着多媒体通信和计算机的发展，人们对于含有大量信息的图像数据、语音信号等多媒体信息的需求在日益提高，然而用以表示这些信息的数据量却很大，这反映在信号的长度比较长，如果直接进行信号处理，计算量将会较大，很不利于信号的实时处理与传输。一方面，可通过信号的压缩、编解码来达到对信号的实时处理；另一方面，可寻找快速算法来减少信号处理的计算量，提高运算效率。现在的不少书籍对快速卷积的算法研究较多，相对而言，相关性探讨的较少。而在现代通信及数字信号处理中，相关（也称线性相关）是一个十分重要的计算和分析方法。通常利用相关函数来分析随机信号的功率谱密度，它对确定信号的分析也有一定的作用。而且在随机信号的数字处理中，可以用相关函数来描述一个平稳随机信号的统计特性。现阶段比较流行的小波分析正在数字信号处理中得到了广泛的应用，其小波分解实质上就是信号与滤波器组的互相关，而小波重构是分解信号与镜象滤波器组的卷积。考虑到卷积和相关在信号处理中具有十分重要的作用，有必要对两者的快速运算作一些探讨和研究，特别是长序列数字信号处理的快速算法，以期达到实时性的目的。

2　基于傅立叶变换的快速相关及卷积

2.1 快速傅立叶变换(FFT)

快速傅立叶变换是在 1965 年由 Cooky 和 Tukey 提出的，是离散傅立叶变换（DFT）的一种快速算法。对于 N 点有限长序列 $x(n)$，其离散傅立叶变换为:

$$X(k)=DFT\{x(n)\}=\sum_{n=0}^{N-1}x(n)W_N^{nk}\,,\qquad k=0,1,..,N-1. \tag{1}$$

而反变换(IDFT)为:

$$x(n)=IDFT\{X(k)\}=\frac{1}{N}\sum_{k=0}^{N-1}X(k)W_N^{-nk}\,,\quad n=0,1,..,N-1. \tag{2}$$

从这两个式子可看出：计算每一个 $X(k)$ 或 $x(n)$ 有 N 次复乘，N-1 次复加，故完成整个运算总共需要 N^2 次复乘及 $N(N-1)$ 次复加。

FFT 算法的基本原理就是把一个 N（一般设 $N=2^L$，L 为整数）点 DFT 分解成两个 N/2 点 DFT，再把 N/2 点 DFT 分解成 N/4 点 DFT，再分解成 N/8 点 DFT，一直分解到两点的 DFT 为止。这种 N 为 2 的整数幂的 FFT 也称基-2 FFT，它可使原有 DFT 的计算量大为减少，通过实际的理论推导，采用 FFT 计算 N 点 DFT，计算量为 $(N/2)\log_2 N$ 次复乘，$N\log_2 N$ 次复加。鉴于一次复乘需要四次实乘和两次实加，所以相应的计算量为 $2N\log_2 N$ 次实乘和 $2N\log_2 N$ 次实加。N 点 IDFT 也可由 FFT 来完成，其计

* 收稿日期：2001-06-11　修订日期：2001-09-04
基金项目：国家自然科学基金资助项目（69772024）

算量与 N 点 DFT 相同。在 N 越大的情况下，FFT 算法较之直接 DFT 计算的优越性就越明显[1]。在后面的程序设计中，我们把 FFT 设定一个子程序 fft.c （用 C/C++语言均可设计），可根据设置的标志量同时完成 FFT 及其逆变换，以便进行快速相关、快速卷积计算时直接调用。

2.2 线性相关与线性卷积

相关主要是指在时域研究两个信号或信号自身之间的相互关系，广泛地应用于各种信号的处理和检测，如通信、雷达、声纳等领域，也应用于连续时间系统及离散时间系统等[1]。

设：两序列为 $x(n)$、$h(n)$，则 $x(n)$ 和 $h(n)$ 的线性卷积定义为：

$$y(n)=\sum_{m=-\infty}^{\infty}x(m)h(n-m)=\sum_{m=-\infty}^{\infty}x(n-m)h(m) \tag{3}$$

其线性相关定义为：

$$r_{xh}(n)=\sum_{m=-\infty}^{\infty}x(m)h^*(m-n)=\sum_{m=-\infty}^{\infty}x(n+m)h^*(m) \tag{4}$$

鉴于实际中的信号常为有限长实序列，上面的(3)、(4)两式相应的可变为：

$$y(n)=\sum_{m=0}^{N_1-1}x(m)h(n-m)=\sum_{m=0}^{N_2-1}x(n-m)h(m) \tag{5}$$

$$r_{xh}(n)=\sum_{m=0}^{N_2-1}h(m)x(n+m)=\sum_{m=0}^{N_1-1}h(m-n)x(m)$$

(6)

其中：N_1，N_2 分别为序列 $x(n),h(n)$ 的长度，$y(n),r_{xh}(n)$ 的长度为 N_1+N_2 -1。

2.3 基于 FFT 的线性相关的快速算法

设两实序列为 $x(n),h(n)$，其长度分别为 L，M，则由（6）式可得其相关值：$y(n)=\sum_{l=0}^{M-1}h(l)x(n+l)$

故 $y(n)$的长度为 L+M-1，为有限长序列。为了能运用 FFT 进行快速运算，且不产生混叠，则应选择周期 N 满足 N>=L+M-1，且 $N=2^r$ (r 为整数)，这样也可调用 FFT 子程序(fft.c)，便于计算。用补零的方法使 $x(n),h(n)$ 具有列长为 N。即：

$$x(n)=\begin{cases}x(n) & n=0,1,\ldots,L-1\\ 0 & n=L,\ldots,N-1\end{cases}\ ;\ h(n)=\begin{cases}h(n) & n=0,1,\ldots,M-1\\ 0 & n=M,\ldots,N-1\end{cases}$$

选择 $N\ge L+M-1$，且 $N=2^r$ （r 为整数）这样就可实现 FFT 及 IFFT，也便于直接调用第一节介绍的 fft.c，提高计算效率。具体的算法公式推导如下：

先进行周期延拓,即: $\tilde{x}(n)=x((n))_N$，$x(n)=\tilde{x}(n)R_N(n)$；$h(n)=\tilde{h}(n)R_N(n)$；$\tilde{h}(n)=h((n))_N$，其中: $0\le n\le N-1$，$R_N(n)=1$; 其他 n 时，$R_N(n)=0$。

$$\because\quad y(n)=\sum_{l=0}^{M-1}h(l)x(n+l)=\sum_{l-0}^{N-1}h(l)x(n+l)$$

$$\therefore\quad Y(k)=DFT\{y(n)\}=\sum_{n=0}^{N-1}y(n)W_N^{nk}=\sum_{n=0}^{N-1}\sum_{l=0}^{N-1}h(l)x(n+l)W_N^{nk}=\sum_{l=0}^{N-1}h(l)\sum_{n=0}^{N-1}x(n+l)W_N^{nk}$$

令 $n+l=m$，$\therefore\ Y(k)=\sum_{l=0}^{N-1}h(l)\sum_{m=l}^{N-1+l}x(m)W_N^{(m-l)k}=\sum_{l=0}^{N-1}h(l)W_N^{-lk}\sum_{m=l}^{N-1+l}x(m)W_N^{mk}$

又

$$\because\quad \sum_{m=l}^{N-1+l}x(m)W_N^{mk}=\sum_{m=l}^{N-1}x(m)W_N^{mk}+\sum_{m=N}^{N-1+l}x(m)W_N^{mk}\qquad (设：m=N+i)$$

$$=\sum_{i=0}^{l-1}x(N+i)W_N^{(N+i)k}+\sum_{m=l}^{N-1}x(m)W_N^{mk}=\sum_{i=0}^{l-1}x(i)W_N^{ik}+\sum_{m=l}^{N-1}x(m)W_N^{mk}$$

$$=\sum_{m=0}^{l-1}x(m)W_N^{mk}+\sum_{m=l}^{N-1}x(m)W_N^{mk}=\sum_{m=0}^{N-1}x(m)W_N^{mk}=X(k)$$

$\therefore\quad Y(k)=H^*(k)X(k)$ （其中利用 $x(n)$的周期延拓，$x(n)=x((n))_N R_N(n)$.）

故由此可得求相关值的算法步骤：

(1) 用 N 点 FFT 求出 $H(k)$，并求其共轭 $H^*(k)$；(2) 再用 N 点 FFT 算出 $x(n)$的 DFT，即 $X(k)$；

(3) 计算乘积 $Y(k)=H^{*}(k)\ X(k)$ ；　　　　(4) 对 $Y(k)$作 N 点 IFFT，得相关序列 $y(n)$。

以上所计算的都可利用 FFT 子程序来完成，整个运算量为：3 次 FFT 运算+N 次相乘。总共需要的相乘次数为：$m_F=3*((N/2)\log_2 N)+N$（复乘）$=N(4+6\log_2 N)$（实乘）。如果直接计算 $y(n)$，则需要的实乘次数为：$m_d=2LM$ 。两者相除可得比值：

$K_m=m_F/m_d=N(2+3\log_2 N)/(LM)$ 分两种情况给予讨论：

(1) $x(n)$与 $h(n)$的长度差不多。假设 $L=M$，则 $N=2L-1\approx 2L$ 。

$$\therefore \quad K_m=(2L)(2+3\log_2 2L)/(L*L)=(10+6\log_2 L)/L$$

由此可得计算量对照表 1。

从表 1 可看出，当 $L=M$ 时，超过 64 后，L 越长，快速相关的优势越明显，计算量越少。

表 1　两种方法的计算量比较　　　　实乘次数

长度 $L=M$	直接计算	快速相关	K_m	长度 $L=M$	直接计算	快速相关	K_m
8	128	448	3.5	64	8192	5888	0.72
16	512	1088	2.13	256	131072	29696	0.23
32	2048	2560	1.25	1024	2097152	143360	0.068

(2) 当 $x(n)$的长度很大时，即当 $L>>M$ 时，则 $N=L+M-1\approx L$ ，$K_m=(2+3\log_2 L)/M$ 。 所以，当 L 较大时，K_m 值会变大，从而使得快速相关的高效性不明显。因此有必要采取适当的方法加以改进，即下一节要讨论的分段法。同样，上述的快速相关可用 C/C++ 语言进行算法编程，在此命名为 correlation.c，以便下面的长序列快速相关设计程序时直接调用。至于快速卷积，可参考文献[2]。

3　长序列快速相关及卷积算法

3.1 算法原理

在实际的信号处理中，一般数字信号处理的单位冲激响应 $h(n)$较短，而数字信号 $x(n)$的长度较长。如果直接用快速相关或卷积计算，$h(n)$必须补很多个零值点，这样一来很不经济，二来快速性不明显（上一节已分析），因而有必要把 $x(n)$分成长度和 $h(n)$相仿的若干段，每段长为 $\underline{L}$，得 $x_i(n)$，i=0,1, ..N/$\boldsymbol{L}$-1。 N 是信号的长度， 将 $x_i(n)$ 与 $h(n)$进行相关或卷积，得到相应的输出 $y_i(n)$ ，然后根据递推公式，把 $y_i(n)$ 结合在一起，就得到总的输出 $y(n)$，而每段的相关或卷积都可采用相应的快速方法来处理。

表 2　快速相关的推导过程

$y(0)=h(0)x(0)+h(1)x(1)+h(2)x(2)+h(3)x(3);$	$y0(0)=h(0)x0(0)+h(1)x0(1)+h(2)x0(2)+h(3)x0(3);$	
$y(1)=h(0)x(1)+h(1)x(2)+h(2)x(3)+h(3)x(4);$	$y0(1)=h(0)x0(1)+h(1)x0(2)+h(2)x0(3)+h(3)x0(4);$	
$y(2)=h(0)x(2)+h(1)x(3)+h(2)x(4)+h(3)x(5);$	$y0(2)=h(0)x0(2)+h(1)x0(3)+h(2)x0(4);$	$y1(5)$
$y(3)=h(0)x(3)+h(1)x(4)+h(2)x(5)+h(3)x(6);$	$y0(3)=h(0)x0(3)+h(1)x0(4);$	$y1(6)$
$y(4)=h(0)x(4)+h(1)x(5)+h(2)x(6)+h(3)x(7);$	$y0(4)=h(0)x0(4);$	$y1(7)$
$y(5)=h(0)x(5)+h(1)x(6)+h(2)x(7)+h(3)x(8);$		$y1(0)$
$y(6)=h(0)x(6)+h(1)x(7)+h(2)x(8)+h(3)x(9);$		$y1(1)$
$y(7)=h(0)x(7)+h(1)x(8)+h(2)x(9);$		$y1(2)$
$y(8)=h(0)x(8)+h(1)x(9)$		$y1(3)$
$y(9)=h(0)x(9);$		$y1(4)$
$y(10)=h(3)x(0)=y(-3);$	$y0(5)=h(3)x0(0)=y(-3);$	
$y(11)=h(2)x(1)+h(1)x(0)=y(-2);$	$y0(6)=h(2)x0(0)+h(3)x0(1)=y(-2);$	
$y(12)=h(3)x(2)+h(2)x(1)+h(1)x(0)=y(-1);$	$y0(7)=h(1)x0(0)+h(2)x0(1)+h(3)x0(2)=y(-1);$	

注： $y1(0)=h(0)x1(0)+h(1)x1(1)+h(2)x1(2)+h(3)x1(3)=h(0)x(5)+h(1)x(6)+h(2)x(7)+h(3)x(8);$

$y1(1)=h(0)x1(1)+h(1)x1(2)+h(2)x1(3)+h(3)x1(4)=h(0)x(6)+h(1)x(7)+h(2)x(8)+h(3)x(9);$

$y1(2)=h(0)x1(2)+h(1)x1(3)+h(2)x1(4)=h(0)x(7)+h(1)x(8)+h(2)x(9);$

$y1(3)=h(0)x1(3)+h(1)x0(4)=h(0)x(8)+h(1)x(9);$

$y1(4)=h(0)x1(4)=h(0)x(9);$

$y1(5)=h(3)x1(0)=h(3)x(5);$

$y1(6)=h(2)x1(0)+h(3)x1(1)=h(2)x(5)+h(3)x(6);$

$y1(7)=h(1)x1(0)+h(2)x1(1)+h(3)x1(2)=h(1)x(5)+h(2)x(6)+h(3)x(7);$

3.2 算法推导过程

设 $x(n)$，$h(n)$的长度分别为 N,M，且 $N>>M$。其相关值

$$y(n)=\sum_{l=0}^{M-1}h(l)x(n+l)$$

线性相关后的总长度为 $N+M$-1，即从 -M+1～N-1。考虑到利用快速相关时已进行了周期延拓(周期为 $N+M$-1)。故:

$y(-M+1)=y(N+M-1-M+1)=y(N)$ …..$y(-1)=y(N+M-1-1)=y(N+M-2)$;因而只要计算 n 从 0～$N+M$-1 时 $y(n)$ 的值，就可反映 $h(n)$与 $x(n)$的线性相关值。把长序列 $x(n)$分成等长度 L 的 m 段，故 $N=m*L$，设 $p=m$-1。

为了便于分析问题，特举例加以说明推导过程。例如：$x(n)$是 N=10 的序列，$h(n)$是 M=4 的序列，选 L=5，则 l=2，p=1，N+L-1=13；设 N1>=L+M-1 且 2 为的幂次方（最小），则 N1=8。其直接相关后的表达式 $y(n)$以及各分段相关的表达式($y_0(n)$、$y_1(n)$)如表 2 所示。

其中:第一段，　$x(0)=x_0(0)$，　$x(1)=x_0(1)$，　$x(2)=x_0(2)$，　$x(3)=x_0(3)$，　$x(4)=x_0(4)$，

第二段，　$x(5)=x_1(0)$，　$x(6)=x1(1)$，　$x(7)=x1(2)$，　$x(8)=x1(3)$，　$x(9)=x1(4)$;

$y_0(n)$，　$y_1(n)$ $(0=<n<=L+L-1)$，分别表示分段后 $x_0(n)$，$x_1(n)$与 $h(n)$的相关值。

由表中可得: $y(0)=y_0(0)$，$y(1)=y_0(1)$，$y(2)=y_0(2)+y_1(5)$，$y(3)=y_0(3)+y_1(6)$，$y(4)=y_0(4)+y_1(7)$，$y(5)=y_1(0)$，$y(6)=y_1(1)$，$y(7)=y_1(2)$，$y(8)=y_1(3)$，$y(9)=y_1(4)$，　$y(10)=y_0(5)$，$y(11)=y_0(6)$，$y(12)=y_0(7)$。

经归纳总结得到递推公式:　　$y(-M+1)=y_0(-M+1)$，…..，$y(-1)=y_0(-1)$

$$y(kL)=y_k(0)，…..，y(kL+L-M)=y_k(L-M);\qquad 0\le k\le p=N/L-1$$

$$y(kL+1-M)=y_k(L)+y_{k-1}(L-M+1)，…..，y(kL-1)=y_k(L+M-2)+y_{k-1}(L-1);\quad 1\le k\le p$$

考虑到要利用 FFT，故分段后各段与 $h(n)$相关后的长度(即 N_1)要为 2 的幂次方，会出现多余的 0 值，如 L=5，M=2 时 $y_0(5)=y_0(6)=0$; L=5，M=3 时 $y_0(5)=0$ 等等。所以对上面的公式作了调整，使其具有一般性。即:　　$y(kL-M+1)=y_k(-M+1)+y_{k-1}(L-M+1)$，…..，$y(kL-1)=y_k(-1)+y_{k-1}(L-1)$;　$1\le k\le p$

也即：$y(kL-M+1)=y_k(N_1-M+1)+y_{k-1}(L-M+1)$，…..，$y(kL-1)=y_k(N_1-1)+y_{k-1}(L-1)$;　$1\le k\le p$

$y(kL)=y_k(0)$，…..，$y(kL+L-M)=y_k(L-M)$;　$0\le k\le p$

当 $k=p$ 时，　　$y(kL+L-M+1)=y_k(L-M+1)$，…..，　$y(kL+L-1)=y_k(L-1)$;

$y(kL+L)=y_0(N_1-M+1)$，…..，　$y(kL+L+M-2)=y_0(N_1-1)$

经过多个例子验证上述的公式是正确的。综上所述可编得长序列快速相关的算法程序。

Longcorrelation.c

```
{   complex d[ ], u[ ],c[ ],y[ ];    double h[ ], x[ ], a[ ][ ];    int i, j, k, p, M, N, L, N1; p=N/L-1;
    for(k=0;k<=p;k++)
      {   for(j=0;j<=M;j++)    { u[j].real=h[j]; u[j].imag=0;}
           for(j=0;j<L;j++)        { c[j].imag=0;c[j].real=*(x+j+k*L); }
          correlation(c,u,y,L,M,N1);    /*-调用快速相关子程序-----------*/
          for(j=0;j<N1;j++)    a[k][j]=y[j].real;     for(j=0;j<=L-M;j++)    d[k*L+j].real=a[k][j];           }
          for (k=1;k<=p;k++)    { for(j=-M+1;j<0;j++)    d[L*k+j].real=a[k][N1+j]+a[k-1][L+j];    }
          for(j=L-M+1;j<L;j++)      { d[L*p+j].real=a[p][j];      d[p*L+M-1+j].real=a[0][N1-L+j];    }
          for(j=0;j<N+M-1;j++)           printf("%3.1f\t",d[j].real);
}
```

通过以上的程序调试时，完全和直接的线性相关计算值相等;而且在 N 值较大的情况下，其计算时间远小于直接计算。具体可参见表 3。

表 3　两种方法的计算时间对照表　　　　　　s

长度	N=252 L=8	N=504 L=16	N=1008 L=16	N=2016 L=16	N=4032 L=16
直接算法	0.25	0.61	1.07	2.54	5.86
基于 FFT 算法	0.21	0.38	0.78	1.52	3.35

注：以上时间包括输出相关值的时间

例 1: 输入信号 $x(n)$={1，2，3，4，0，1，0，4，3，4，0，1，2，0，3}，滤波器响应 $h(n)$={2，3，1}，按照上述算法计算可得: $y_0(n)$={11.0，17.0，18.0，8.0，0.0，0.0，1.0，5.0}，　$y_1(n)$={ 6.0, 15.0, 21.0, 18.0, 8.0, 0.0, 1.0, 3.0}，$y_2(n)$= {5.0, 8.0, 7.0, 9.0, 6.0, 0.0, 0.0, 1.0}，相关输出值 $y(n)$= {11.0, 17.0, 18.0, 9.0, 3.0, 6.0, 15.0, 21.0, 18.0, 9.0, 5.0, 8.0, 7.0, 9.0, 6.0, 1.0, 5.0}.直接计算得: $y(n)$={11，17，18，9，3，6，15，21，18，9，5，8，7，9，6，1，5}.结果表明一致。

3.3　长序列快速卷积的算法

长序列快速卷积的算法原理可参考文献[2]，具体的推导过程与快速相关类似。这里只给出算法的递推公式和算法程序，公式如下：　　$y(0)=y_0(0)$,…..,$y(L-1)=y_0(L-1)$;

当 $1\le k\le p$ 时：　$y(kL)=y_k(0)+y_{k-1}(L)$,…..,$y(kL+M-2)=y_k(M-2)+y_{k-1}(L+M-2)$;

$y(kL+M\text{-}1)= y_k(M\text{-}1),\ldots, y(kL+L\text{-}1)= y_k(L\text{-}1);$

当 $k=p=N/L\text{-}1$ 时： $y(kL+L)=y_k(L),\ldots, y(kL+L+M\text{-}2)=y_k(L+M\text{-}2);$

根据上面的公式，可编得长序列快速卷积的算法程序，在此命名为 longcon.c。

Longcon.c

```
{   complex d[ ], u[ ],c[ ],y[ ];   double h[ ], x[ ], a[ ][ ];     int i, j, k, p, M, N, L, N1; p=N/L-1;
    for(k=0;k<=p;k++)
  {    for(j=0;j<M;j++)    { u[j].real=h[j]; u[j].imag=0;}
       for(j=0;j<L;j++)      { c[j].imag=0;c[j].real=*(x+j+k*L); }
      convolution(u,c,y,M,L,N1);   /*----调用快速卷积子程序---------*/
      for(j=0;j<N1;j++)      a[k][j]=y[j].real;          for(j=0;j<L;j++)    d[j].real=a[0][j];      }
      for (k=1;k<=p;k++)
         { for(j=0;j<M-1;j++)    d[L*k+j].real=a[k][j]+a[k-1][L+j];
           for(j=M-1;j<L;j++)    d[L*k+j].real=a[k][j];      }
  for (j=0;j<M-1;j++) {d[p*L+L+j] real=a[p][L+j]; for(j=0;j<N+M-1;j++)    printf("%3.1f\t",d[j].real);
}
```

以上程序调试完全和直接的线性卷积计算值相等;且在 N 值较大的情况下，计算时间远小于直接计算。

例 2：输入信号 $x(n)$={4，2，3，1，5，6，1，0，4，0，3，2，5，3，0，1，4，2}，滤波器冲激响应 $h(n)$={1，2，3}.可分成三段，N=18，L=6，M=3，$p=N/L\text{-}1=2$.按照上述算法程序计算可得: $y_0(n)$= {4.0, 10.0, 19.0, 13.0, 16.0, 19.0, 27.0, 18.0}，$y_1(n)$={1.0, 2.0, 7.0, 8.0, 15.0, 8.0, 13.0, 6.0}; $y_2(n)$= {5.0，13.0，21.0，10.0，6.0，13.0， 16.0，6.0}。进行分段叠加后得到卷积 $y(n)$= {4.0，10.0, 19.0, 13.0, 16.0, 19.0, 28.0, 20.0, 7.0, 8.0, 15.0, 8.0, 18.0, 19.0, 21.0, 10.0, 6.0, 13.0, 16.0, 6.0}。直接方法计算卷积可得: y(n)={4, 10, 19, 13, 16, 19, 28, 20, 7, 8, 15, 8, 18, 19, 21, 10, 6, 13, 16, 6}，结果表明一致。

鉴于两者有相似形，具体设计程序时，可把两者集合起来考虑。通过一个标志量 sign， if(sign=0) {longconvolution}; if(sign=1) {longcorrelation};共设一个子程序，以便计算时直接引用，提高运算效率。

4 结论和改进

表 4 FFT 和 AFT 计算效率的对照表

长度	521	911	971	1483	2417
FFT 效率	67.3%	68.03%	72.5%	71.23%	76.22%
AFT 效率	91.39%	91.78%	91.63%	91.81%	91.83%

本文采用快速傅立叶变换计算长序列的线性相关和卷积，达到了快速运算的目的。现正试用于图像的小波分解和重构，以期达到对图像的实时处理。由于 FFT 对长度为 2 的幂的信号进行计算时非常有效，但对于一般长度的信号，当 N 含较大素因子或为较大素数时，进行 FFT 需要补许多不必要的零，导致 FFT 子进程增多，算法程序更复杂，很不利于实际应用。1988 年，Tuffs 和 Sadasiv 提出的算术傅立叶变换(AFT)[3]，其乘法量仅为 0(N)，特别对 N 为素数时，其计算方法简单，效率明显高于 FFT（见表 4）。算法具有良好的并行性，且有较好的现成算法程序可用[5]，尤其适合 VLSI 设计，在数字图像处理等领域中得到了广泛应用[4]。从上面分析可看出，算术傅立叶变换和快速傅立叶变换具有互补的优势，如果能把两者有机的结合起来，应用到长序列的快速相关和卷积算法中，对任意长度的 N 都会使计算量显著下降，大大提高运行效率，更有利于信号的实时处理和传输。具体程序设计时，也可通过一个标志参数，当该参数为某个值（如为 0）时，进行长度为 2 的幂次方的信号的快速傅氏运算；当为另一个值（如为 1）时，调用 AFT 算法程序进行信号的算术傅氏运算，从而达到快速的目的。

参考文献：

[1] 程佩青. 数字信号处理教程[M]. 北京:清华大学出版社， 1998

[2] Sophocles J Orfanidis. Introduction To Signal Processing[M]. Beijing : tsinghua publish house， 1999

[3] Tufts D W and Sadasiv G. The arithmetic Fourier transform[J]. *IEEE ASSP Mag*， 1998-01, (1):13-11

[4] Reed I S, Tang Ming Shih, Truong T K, Hendon R. and Tufts D W. A VLSI architecture for simplified arithmetic fourier transform algorithm. *IEEE Trans. Signal Processing* ,1993-05, 40(5):1122-1132

[5] 张宪超, 武继刚， 蒋增荣， 陈国良.离散小波变换的算术傅里叶变换算法[J]. 电子学报， 2000， 28(5):105-107

作者简介： 虞湘宾，东南大学无线电工程系，博士研究生，研究方向：宽带多媒体通信及数字信号处理；毕光国，东南大学无线电工程系，教授，博士生指导老师。

Algorithms of Long Sequence Fast Correlation and Convolution

YU *X*iang-bin , Bi Guang-guo

(Department of Radio Engineering, Southeast University, Nanjing 210096, China)

Abstract: Based on the analysis of the principle of conventional Fast Fourier Transform (FFT) algorithm, considering the mathematical characteristics and the physical meaning of linear correlation and convolution, A fast correlation and convolution algorithm for long sequences FFT is proposed. This algorithm has been implemented using C++ programming language. Computer simulation results indicate that the proposed algorithm can accelerate the execution of FFT. Finally, this algorithm is improved in combination with arithmetic Fourier transform.

Key words: fast Fourier transform; fast correlation; fast convolution; arithmetic Fourier transform

---

(from page 77) （续 77 页）

Performance Simulation by Interpolation for BPSK All-digital Receivers

MENG Li-min[1,2], QIU Pei-liang[2]

(1. College of Information Engineering, Zhejiang University of Technology, Hangzhou 310032, China;

2. Institute of Information and Communication Engineering of Zhejiang Unive rsity, Hangzhou 310027, China)

Abstract: All-digital processing technology is indispensable for modern communication. The problem of timing adjustment in All-digital receivers is elucidated. The interpolation performance of BPSK all-digital receivers is investigated through simulation.

Key words: all-digital receivers; timing adjustment; interpolation

---

(*From page* 73) （续第 73 页）

New Method of Vision Based Vehicle Tracking and Traffic Parameter Estimation

JIANG Gang-yi[1,2], YU Mei[1], YE Xi-en[1], LIU Xiao [1]

(1. Institute of Circuits and Systems, Ningbo University, 315211, China；

2. National Lab.on Machine Perception, Peking University, Beijing 100871, China)

Abstract: Vision based traffic monitoring systems have several apparent advantages such as easily intervened and lower costs. An improved background subtraction method together with a new method that combines the improved background subtraction with edge detection is proposed for vehicle detection and shadow rejection, based on which real-time vehicle trucking, counting, classification and speed estimation are implemented. Experiment demonstrated that the vehicle detection rate is larger than 98%, during which the passive shadows resulted from roadside buildings grew considerably.