

# CT 画像再構成におけるフィルタ処理の高速化

## A Fast Algorithm of High-pass Filter for CT Reconstruction

村上 純一†  
Junichi Murakami

森川 良孝‡  
Yoshitaka Morikawa

### 1. まえがき

X 線 CT の代表的な再構成アルゴリズムとして「畳込み逆投影 (Convolution-Backprojection:CB) 法」が知られている。CB 法の再構成アルゴリズムでは、最初に投影信号に対して補正関数との畳込みを用いた高域強調処理を施し、次に、個々の投影方向に沿って逆投影する。逆投影の総和が再構成画像となる。CB 法では、前半部の高域強調処理に比較して後半部における逆投影に要する演算量が大きい。そのため、高域強調処理時間は問題にならない。これに対して、逆投影処理を木構造フィルタバンクにより行う高速アルゴリズム (以下、TFB 法) [1]では高域強調処理の演算量低減も重要である。

従来、高域強調処理には高速 Fourier 変換 (FFT) が用いられてきたが、差分操作と Hilbert 変換[2]により演算量の低減を図った高域強調フィルタを提案する。Hilbert 変換を効率のよい全域通過フィルタを用いて実装し、境界条件から導出される初期値を用いて本来無限長のフィルタ処理を有限長化した。その結果、 $256 \times 256$  画素の場合 FFT 法と比較して 1/5 の時間短縮となった。

また、後半部の逆投影処理においても境界の計算を工夫することにより木構造フィルタバンクの処理時間を短縮し、マルチプロセッサ構成に適した浮動小数点演算 DSP においてほぼ実時間観測が可能な処理速度 (33ms) を達成した。

### 2. 高域強調フィルタ

#### 2.1 全域通過フィルタによる設計

X 線 CT の再構成アルゴリズムでは、周波数応答が  $|\omega|$  で高域強調する Ram-Lak フィルタ、高周波領域で利得がやや低下する Shepp-Logan フィルタ (以下、SL フィルタ) 等が用いられる[3]。前者のフィルタは周波数応答が  $j\omega$  である微分作用素と  $j\text{sgn}(\omega)$  である Hilbert 作用素との縦続接続により実現できる。微分作用素の最簡な実現は差分フィルタで与えられることから、Hilbert 作用素は全域通過フィルタを用いて設計した。

次式で示す全域通過フィルタ  $A(z)$  はフィルタ係数  $a$  を適切に設計した場合、全体的には線形位相進み特性  $\omega/2$  をもち、 $\omega = \pm\pi$  で  $\pi$  の位相ジャンプがあるようにすることができる[4]。

$$A(z) = \frac{1+az}{1+az^{-1}} \quad (1)$$

この特性を利用すれば、次式で示すフィルタ  $F(z)$  は、線形位相で遮断周波数  $\pi/2$  の低域通過フィルタとなる。

$$F(z) = \frac{1}{2} \left[ 1 + \frac{1}{2} \{ z^{-1} A(z^2) + z A(z^2) \} \right] \quad (2)$$

フィルタ  $F(z)$  の振幅を 2 倍し、1 だけシフトすれば次の離散型 Hilbert フィルタ  $H(z)$  を得る。

$$\begin{aligned} H(z) &= 2F(-jz) - 1 \\ &= \frac{jz^{-1}}{2} \{ A(-z^2) - z^2 A(-z^2) \} \end{aligned} \quad (3)$$

また、微分作用素の最簡な実現として次の差分フィルタ  $D(z)$  を用いる。

$$D(z) = \frac{z}{2j\pi} (1 - z^{-2}) \quad (4)$$

これらより、高域強調フィルタ  $V(z)$  は

$$\begin{aligned} V(z) &= H(z^{1/2}) D(z^{1/2}) \\ &= \frac{1-z^{-1}}{4\pi} \{ A(-z) - z A(-z^{-1}) \} \end{aligned} \quad (5)$$

として得られる。これは、並列にフィルタ  $A(-z^{\pm 1})$  を施した結果の 1 標本分の差分に 1 標本分の差分を施すことにより得られることを示している。しかし、再構成画像の品質を維持するためには、次式に示す高次の全域通過フィルタを用いる必要があり、次数  $K$  の選択が重要である。

$$A_K(z) = \prod_{k=1}^K \frac{1 + a_k z^{-(-1)^k}}{1 + a_k z^{(-1)^k}} \quad (6)$$

ただし、 $0 < a_1 < a_2 < \dots < a_K < 1$  である。

$256 \times 256$  画素の再構成画像による予備実験では、 $K=4$  で表 1 に示すパラメータの結果が良好であった。この場合のフィルタ  $V(z)$ 、及び、理想的な SL フィルタの周波数特性を  $0 \leq \omega < \pi$  の範囲で図 1 に示す。同図より、設計した高域強調フィルタ  $V(z)$  は原点付近で SL フィルタをよく近似していることがわかる。

#### 2.2 計算機への実装

$K=4$  のとき式(6)の  $k$  を奇偶に分けて展開すれば、 $n$  が増加する右方向フィルタと  $n$  が減少する左方向フィルタの縦続接続として表現できる。これらのフィルタは、それぞれ次式のように繰り返し計算することにより実現できる。

表 1 全域通過フィルタ  $A(z)$  のパラメータ

$a_1$	$a_2$	$a_3$	$a_4$
0.49	0.86	0.95	0.99

† 詫間電波工業高等専門学校

‡ 岡山大学大学院自然科学研究科

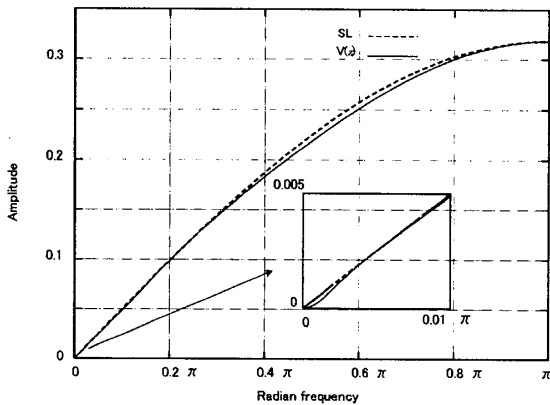


図1 高域強調フィルタ  $V(z)$  の周波数特性

$$v[n] = u[n] + r_1(u[n+1] - v[n-1]) + r_2(u[n+2] - v[n-2]) \quad n: \text{増加} \quad (7)$$

$$w[n] = v[n] + l_1(v[n-1] - w[n+1]) + l_2(v[n-2] - w[n+2]) \quad n: \text{減少} \quad (8)$$

ただし、 $u[n]$ ,  $v[n]$  は右方向フィルタの、 $v[n]$ ,  $w[n]$  は左方向フィルタのそれぞれ入出力信号であり、 $r_1 = a_1 + a_3$ ,  $r_2 = a_1 a_3$ ,  $l_1 = a_2 + a_4$ ,  $l_2 = a_2 a_4$  である。

式(7)の  $v[n]$  は初期休止条件により逐次  $v[n]$  を求めることができるが、式(8)の  $w[n]$  は式(7)の結果を使用するからこの解を  $n \rightarrow \infty$  まで求めなければならないという難点がある。しかし、入力信号  $u[n]$  は有限長であり、 $n \geq N$  では  $u[n] = 0$  であることを用いれば、式(7)は  $v[n] + r_1 v[n-1] + r_2 v[n-2] = 0$  となり、この解は  $v[N-1]$  と  $v[N-2]$  により完全に記述できる。したがって、この結果に式(8)の左方向フィルタを施す場合の表現が求められ、初期値として次式を得ることができる。

$$\begin{bmatrix} w[N] \\ w[N+1] \end{bmatrix} = \frac{1}{a_1 - a_3} \begin{bmatrix} c_1 - c_3 & a_3 c_1 - a_1 c_3 \\ a_1 c_1 - a_3 c_3 & a_1 a_3 (c_1 - c_3) \end{bmatrix} \cdot \begin{bmatrix} v[N-1] \\ v[N-2] \end{bmatrix} \quad (9)$$

ただし、

$$c_i = \frac{(a_2 - a_i)(a_4 - a_i)}{(1 - a_2 a_i)(1 - a_4 a_i)} \quad i = 1, 3 \quad (10)$$

式(9)を初期値として  $n$  を減少させながら逐次式(8)を適用すれば所望の範囲で  $A(z)$  の出力が計算できる。

### 2.3 演算量の比較

1 投影信号の長さを  $N$  として、CB 法と提案法の高域強調処理に必要な演算量を比較する。まず、CB 法では、補正関数との畳込みを FFT を用いた周波数領域での乗算で実現する。周波数空間での乗算は FFT の演算量と比較して十分小さいので無視することとするが、周波数の折返しを防ぐために信号の前後に  $N/2$  ずつ 0 を挿入し、長さを  $2N$  としなければならないから、逆変換を含めると乗算回数は  $8N \log_2 2N$ 、加算回数は  $8N \log_2 2N$  となる。

表2 高域強調処理の演算量の比較

手法	乗算	加算
従来法 (CB)	$8N \log_2 2N$	$8N \log_2 2N$
提案法 (TFB)	$9N$	$18N$

次に、提案法では 4 次の全域通過フィルタの 1 点に必要な演算は乗算 4 回、加算 8 回である。フィルタ  $A_4(-z^{\pm 1})$  を施すためこの 2 倍の演算を要し、差分演算を考慮すると高域強調フィルタに要する総演算量は乗算  $9N$  回、加算  $18N$  回である。表 2 に演算量の比較結果を示す。同表によれば  $N=256$  の場合、提案法は従来法に比べて乗算に関して  $1/8$  に演算量を低減できる。

### 3. 木構造フィルタバンク (TFB) 法

X 線 CT の投影信号は積分操作により得られるから、積分操作を投影方向の狭帯域低域通過フィルタリングと考えダウンサンプリングを加えるならば、フィルタバンクの分析過程となる。したがって再構成はフィルタバンクの合成過程の設計問題となる。

TFB 法では、高域強調フィルタを施した投影信号をダウンサンプリングして実空間に配置し、アップサンプリングにより生じた虚像を除去するフィルタリング操作により逆投影処理を実現する[1]。このとき使用する投影角に応じた傾き  $k$  の平行四辺形フィルタ  $P_k$  は、

$$P_k(z_0, z_1) = \frac{1}{2} \{1 + z_0^{-k} z_1 U(z_0) U(z_0^{2k-1} z_1^2)\} \quad (11)$$

ただし、

$$U(z) = \frac{1}{2} \left\{ \frac{1+az}{1+az^{-1}} + \frac{a+z}{1+az} \right\}, \quad a = 0.5 \quad (12)$$

である。平行四辺形フィルタ  $P_k$  の入力信号のスペクトルは隣接する 2 個のフィルタの通過域にある。そこで傾斜角に依存する 2 個の斜め方向フィルタ  $z_0^{-k} U(z_0^{2k-1} z_1^2)$  を個別に施した結果を加えた後、傾斜角に依存しない横方向フィルタ  $U(z_0)$  を施せば、図 2 に示すように、木構造状の合成フィルタを構成できる。同図において、まず隣接する投影角に対応する 2 つの入力信号を加え合わせて偶数行に収め、次にそれぞれの入力信号に傾きの異なる斜め方向のフィルタを施して足し合わせ、その結果に横方向のフィルタを施した結果を奇数行に収める、という操作を行う。

斜め方向フィルタ  $z_0^{-k} U(z_0^{2k-1} z_1^2)$  の見通しの良い実装法としては、2 次元配列からフィルタを施す方向の信号を取り出して 1 次元配列に納め、その 1 次元配列に  $U(z)$  を施し  $k$  だけ遅延させた信号を元の 2 次元配列に返すことが考えられる。しかし、この方法では信号配列の 2 次元、1 次元の相互の変換や、1 次元信号の境界判別が複雑な条件文になり処理時間を要するという欠点がある。そこで、処理を 2 次元配列で行うこととし、フィルタ演算を行う領域をあらかじめ求めておくことにより、繰返し処理の時間短縮を図る。

次に、図 2 中最下段のパスの実装について述べる。まず、 $k+1$  遅延処理( $z_0^{-k+1}$ )は図 3 に示すように 2 次元配列上で直接

行う。同図において破線部は原信号である。列方向に  $k+1$  だけ遅延された信号を得るには、実線部を取り出せばよいが、フィルタ処理後点線部に戻すために信号の左側に空白部分ができる。この部分は信号が矢印方向 ( $z_0^{2k+1}z_1$  方向) に減衰係数  $\rho$  で減衰していると仮定して、フィルタ方向に最近傍の値を  $\rho$  倍して代入する。最下行の場合はフィルタ方向に該当する点がないため、すぐ右の値を代入する。このデータに対してフィルタ処理  $U(z_0^{2k+1}z_1)$  を行う。  $U(z)$  の処理は入力信号に 2 個の全域通過フィルタを施した結果の平均である。

図 4 にフィルタ処理の方向を示す。図中灰色の点では初期値、及び、最終値の計算を行い、白丸の点では 2 次元配列のまま斜め方向にフィルタの繰返し計算の 1 つを行う。黒丸の点では何も行わない。最後に、それぞれの結果の平均を求めることでフィルタ出力が得られる。

この操作を斜めフィルタ方向でなくそれぞれの列方向に行うことにより端の条件処理がより単純になり、高速なフィルタ処理が可能となった。

#### 4. DSP への実装

本研究では、CISC 型プロセッサに比べてマルチプロセッサを構成し易い DSP を用いて再構成アルゴリズムの実装を行った。実装には、固定小数点演算専用 DSP である TMS320C6416 (以下 C6416, Texas Instruments 社), 及び、浮動小数点演算 DSP である ADSP-TS201 (以下 TS201, Analog Devices 社) を用いた。

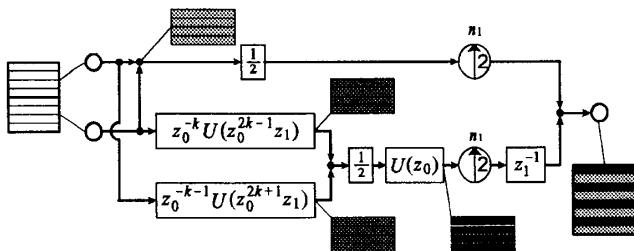


図 2 木構造フィルタバンクの処理ユニット

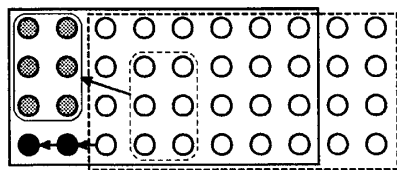


図 3  $k+1$  遅延処理 ( $k=1$ )

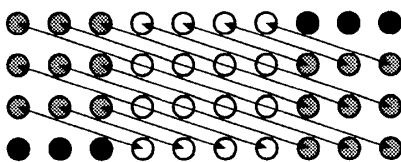


図 4 フィルタ  $U(z_0^{2k+1} z_1)$  の実装

#### 4.1 固定小数点演算 DSP への実装

固定小数点演算 DSP である C6416 は、6 個の ALU と 2 個の乗算器により最大 8 個の高速な並列演算が可能であり、1MB の内部 RAM は L2 キャッシュとしても利用できるため、画像をそのまま高速な内部キャッシュに納めることができることから画像処理に適した DSP である。しかし、固定小数点演算専用であるため、浮動小数点演算に比べて丸め誤差による精度の悪化に注意を要する。C6416 のレジスタ長は 32 ビットであるが、乗算器は 16 ビット  $\times$  16 ビット動作であり、バイトアクセスが可能なることから、32 ビットの中に 2 個の 16 ビットデータをペアで使用する 16 ビットパック化データ形式で使用するのが効率的である。C6416 用 C 言語コンパイラでは、最適化オプションによりアセンブリ言語に匹敵する高度な並列化が実現でき、2 組の演算ユニットがバランスよく使用される。

固定小数点演算では小数点の位置は明示的に示されず、コードの中で取り扱わねばならない。そのための表記として小数の桁数  $n$  を  $Qn$  により表す  $Q$  フォーマットが用いられる。 $Q$  フォーマットによる固定小数点演算では、加算は  $Q$  フォーマットをそろえて行い、乗算は結果を 2 倍のビット数で求め、 $Qn$  フォーマットの場合は  $n$  ビット右シフトする。

再構成アルゴリズムでは、主として  $Q12$  フォーマットを用い、乗算時の丸め誤差による精度の悪化を防ぐため、乗算時にはビットシフトを必要なだけ行わず、上位にビットを残すようにし、不足したビットシフト量については、再構成画像を出力する際に行うこととした。これにより、画質を損なうことなく、固定小数点演算による再構成が可能となった。

#### 4.2 浮動小数点演算 DSP への実装

TS201 は、C6416 と動作クロック周波数が同程度であるため非常に高速な演算が可能であり、内部には 1 ブロック 4M ビットから構成される RAM を 6 ブロックもつため画像処理にも適している。更に、TS201 はマルチプロセッサ構成に適したアーキテクチャをもっている。

しかし、TS201 では、C 言語コンパイラの最適化オプションのみでは 2 組の演算ユニットを並列にバランスよく用いることはできず、また、同一レジスタの連続アクセスではストール (待ち時間) が入ることから、C6416 に比べて約 2 倍の処理時間がかかった。2 組の演算ユニットを用いた並列化には、アセンブリ言語によるプログラミングが不可欠である。ただし、すべてを最初からアセンブリ言語により作成することは多大な時間を要するため C 言語コンパイラのアセンブラ出力を使用するのが効率的である。

### 5. シミュレーション

本節では、提案法及び従来法のシミュレーション結果を示す。対象は Shepp & Logan の Head Phantom  $256 \times 256$  画素である。ただし、濃度値は、骨: 2.0, 組織: 1.02, 腫瘍: 1.12 とした。

アルゴリズムの評価は、シミュレーションによる処理時間と、次式で示す信号対雑音比 (SNR) による再構成画像の画質により行う。

$$SNR = 10 \log_{10} \left( \frac{256 \times 256}{\text{平均 2 乗誤差}} \right) \quad (13)$$

本実験では、3.で述べた減衰係数  $\rho$  は、簡略化のため  $\rho = 1$  に選んだ。図5に再構成シミュレーション結果を示し、図6に断面の一部を示す。また、表3に処理時間結果をCPUと共に示す。同表において、Pentium4はPCであり、コンパイラはIntel C++ 8.0を用い、演算は単精度実数型で行った。DSP (C6416)は固定小数点演算であるので、投影データの作成、及び、提案法で必要となるダウンサンプリング処理はPCで行い、その処理時間を加えた。DSP (TS201)は、単精度実数型で演算を行った。

図1に示す高域強調フィルタは、全域ではほぼ同じ特性が得られているが、周波数0付近の特性は提案法では高次系となっている。このため低域のゲインを補償する必要があるが、シミュレーション結果より画素値を一様に1.07倍することで十分補正できた。この値は、表1のフィルタ定数では周波数0付近のSLフィルタとの最大誤差が約0.025%であることによるものである。信号対雑音比(SNR)による再構成画像の画質比較では、従来法が35.6dB、提案法が35.5dBであった。

FFTによる高域強調フィルタと提案するフィルタの処理時間を比較すると、提案法の処理時間は1/5であり、DSPによる実験結果はPCと同等である。しかし、DSPでは、FFTを高度に並列化することが可能であるため、表3には示していないがTS201のアセンブリ言語での比較では、提案法の高域強調フィルタの処理時間はFFTの2/3であった。なお、TS201のC言語では、約10msの処理時間であった。

逆投影処理では、提案する木構造法は従来法であるCB法に比較して約1/13の処理時間であった。PCでは、動作クロック数3.2GHzのCPUでも大きな改善は認められず、むしろメモリアクセス速度の影響が大きいようである。

DSPの実装結果は、いずれも実時間観測に必要な再構成時間(33ms)をほぼ達成した。なお、TS201のC言語では約2倍の処理時間であった。16ビット固定小数点演算では十分な精度が得られないため再構成画像の劣化が考えられたが、それぞれの楕円内部では濃淡度はほぼ一定であり再構成画像は従来法と遜色なく、十分実用化できるレベルである。

## 6. むすび

X線CTの代表的な再構成アルゴリズムとして知られる「畳込み逆投影(CB)法」前半部の畳込み処理を差分操作とHilbert変換による高域強調フィルタを用いて行った。特に、Hilbert変換を効率のよい全域通過フィルタを用いて実装し、演算量を低減した。後半部の逆投影についても、斜め方向フィルタの計算を改良した木構造フィルタバンクにより行い処理時間の短縮を図った。

この提案アルゴリズムを固定小数点演算DSPであるTMS320C6416、及び、浮動小数点演算DSPであるADSP-TS201Sに実装した。固定小数点演算DSPへの実装では、16ビットの丸め誤差による精度の悪化を考慮してビットシフトをフィルタ処理部と画像出力部に配分したことにより、従来法と比較して遜色のない再構成画像が得られた。また、浮動小数点演算DSPへの実装では、一般に、固定小数点演算よりも処理速度が遅い問題があるが、最適化や並列化手法を用いて処理速度の向上を図り、いずれのDSPにおいてもほぼ実時間観測可能な処理速度を達成した。

今後の課題は、マルチプロセッサ構成のDSPを用いて、提案アルゴリズムを3次元再構成に適用することである。

表3 処理時間 (ms)

実験	CPU	高域強調	逆投影	合計
従来法 (CB)	Pentium4 (2.6GHz)	10	200	210
提案法 (TFB)	Pentium4 (2.6GHz)	2	16	18
	DSP (C6416, 500MHz)	2	36	38
	DSP (TS201, 500MHz)	2	31	33

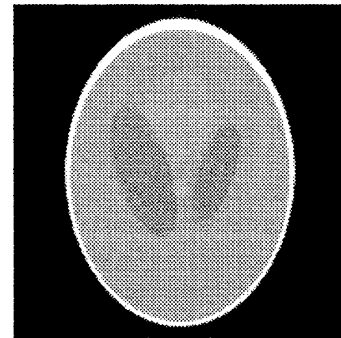


図5 再構成シミュレーション結果

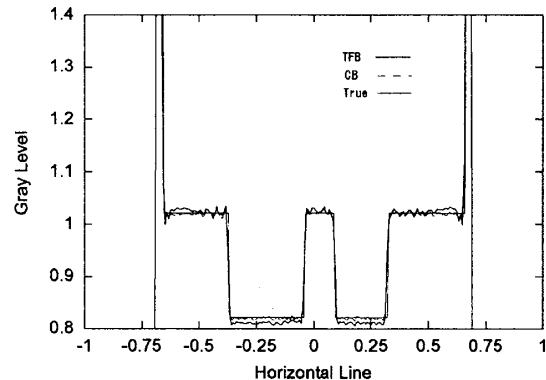


図6 断面の比較 (単精度実数  $y=-0.047$ )

謝辞 本研究の一部は文部科学省科学研究費補助金基盤研究(C)課題番号16560367の助成によるものである。

## 文献

- [1] 村上純一, 溝脇一成, 森川良孝, “木構造フィルタバンクを用いたCTの再構成アルゴリズム,” 信学論 (D-II), vol.J84-D-II, no.3, pp.580-589, March 2001.
- [2] A.K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.
- [3] L.A. Shepp and B.F. Logan, “The Fourier reconstruction of a head section,” IEEE Trans. Nucl. Sci., vol.NS-21, no.3, pp.21-43, 1974.
- [4] P.P. Vaidyanathan, *Multirate Systems and Filter Banks*, Prentice Hall, 1993.