透過型走査電子顕微鏡の実時間収差補正システムの性能予測

指導教官：市川周一　　学籍番号：023719 高橋翔

1　背景

電子顕微鏡は電子線で試料を照射し，電子レンズで結像させる装置である．電子レンズには球面収差が残存するため，像にボケが生じる（図 1）．高分解能の電子顕微鏡を作成するには球面収差の補正が必須である．

動的ホローコーン照明による無収差結像法 [1][2] は，球面収差を除去した位相像・振幅像を観察する手法である（図 3）．図 3 の信号処理部では，2 次元画像に複素 FFT をかけ，空間周波数を 8 の字状に抽出（図 2）した後，逆 FFT を行う．

本研究の目的は，信号処理部のチューニングを行い，収差補正システムの性能を見積もることである．

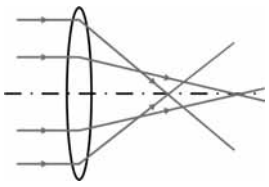

図 1: 球面収差

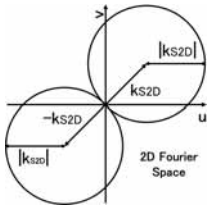

図 2: 8 の字フィルタ [1]

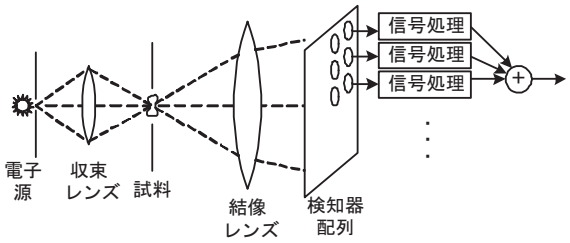

図 3: システム概略図 [1][2]

2　FFT

まず 2 次元複素 FFT のライブラリを調査し，4 種について実機上で性能を測定した（表 1）．Pentium4 2.4 GHz，RAM 512 MB，RedHat Linux 9.0 の計算機を使用し，コンパイラには icc7.1 を使用している．以下，データサイズを 1000×1000 に固定し，最も高速な FFTW3 [3] を採用して，以下のチューニングを行う．

・単精度で計算
・SSE/SSE2 等の SIMD 命令を使用
・プラン PATIENT を使用 [3]（プランとは，使用するアルゴリズムの選択，領域確保，領域初期化を行う処理である．デフォルトはプラン MEASURE を使用する）
・プラン作成時に，同時に計算する処理数 N を 2 以上に設定

上記手法を併用し 1000×1000 の処理時間を測定した（表 2）．N=8，単精度，プラン PATIENT，SSE を併用した場合，実行時間は 43 ms に短縮された．

表 1: ライブラリ比較（単位 ms）（倍精度）

	Size			
Library	256×256	500×500	512×512	1000×1000
FFTW 2.1.5	3.1×10^1	4.3×10^1	1.2×10^2	1.9×10^2
FFTW 3.0.1	1.1×10^1	3.3×10^1	7.0×10^1	1.4×10^2
MKL 6.1	1.1×10^1	5.0×10^1	5.0×10^1	2.2×10^2
ACML 1.0r2	2.1×10^1	8.8×10^1	9.3×10^1	4.0×10^2

表 2: FFTW3 のチューニング（単位 ms）（単精度）

	PATIENT		MEASURE	
N	SSE 有	SSE 無	SSE 有	SSE 無
1	7.0×10^1	1.4×10^2	9.0×10^1	1.4×10^2
4	5.4×10^1	1.2×10^2	6.7×10^1	1.3×10^2
8	4.3×10^1	1.1×10^2	6.3×10^1	1.3×10^2

3　8 の字フィルタ

文献 [1] に記載されているプログラム stemmda2.f から 8 の字フィルタ部分（図 2）を C 言語に移植した（プログラム P1）．P1 の実行時間は 2 章と同じ環境で 1.13×10^3 ms であった．P1 に icc 付属の Intel Math Library 等の単精度用ライブラリを導入し，アルゴリズムを改良したプログラム（P2）を作成した．P2 の実行時間は 3.0×10^1 ms に短縮された．

4　信号処理シミュレーション

本手法のシミュレーションプログラム stemmda2.f [1] を C 言語に移植した（stemmda2.c）．stemmda2.c は球面収差を含むサンプル画像（解像度 256×256，検知器 17×17 個）を作成し，収差補正処理をシミュレートするものである．stemmda2.c に 2 章，3 章で示した改良を加えたプログラム custom.c も作成し，比較する．

逆 FFT（IFFT）は線形演算であるため，stemmda2.c，custom.c の処理手順を図 4 から図 5 に変更できる．それぞれの処理時間を測定した結果を，表 3 に示す．単精度で図 5 の手法を採用した場合が一番高速で，このとき custom.c は stemmda2.c の 2.0 倍高速であった．

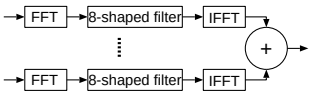

図 4: 逆 FFT 後に加算

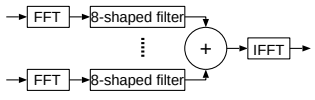

図 5: 加算後に逆 FFT

表 3: 比較結果

	単精度 [sec.]		倍精度 [sec.]	
	IFFT → Σ	Σ → IFFT	IFFT → Σ	Σ → IFFT
stemmda2.c	9.1	4.4	19	8.1
custom.c	4.3	2.2	13	6.2
速度比	2.1	2.0	1.5	1.3

5　考察

表 2 より，単精度 1000×1000 の複素 FFT/IFFT（N=8）は 43 ms で処理できる．また，1000×1000 の 8 の字フィルタは 30 ms で処理できる（3 章）．よって図 3 の信号処理部は合計 116 ms で実行できる．目標時間（30 ms）で処理するには 3.9 倍の高速化が必要である．近年の CPU の速度向上は 1 年で 1.58 倍である [4] から 1000×1000 の信号処理を実時間処理できるのは $\frac{\log 3.9}{\log 1.58}$ ∼ 3 年後と予想される．

次に 4 章の結果から，解像度 256×256 のシステムの性能を予測する．信号処理は図 5 の手法（単精度）とする．全ノードの出力を加算する時間を T_Σ，IFFT の時間を T_{IFFT}（表 1 より 11 ms）とすれば，n 台のノードを使用した場合の処理時間 T は，

$$T = \left\lceil \frac{289}{n} \right\rceil \frac{2.2}{289} + T_\Sigma + T_{IFFT} \quad (sec.).$$

16 ノード使用した場合 $T = 0.16 + T_\Sigma$ $(sec.)$ となる．T_Σ が十分小さなシステムであれば，最大 6 コマ/秒で観察が可能である．

参考文献

[1] 生田 孝: “画像処理による収束光学系収差補正機構を持つ透過型走査光学顕微鏡の試作,” 科研費報告書 基盤 (B)(2) 11555021 (2001).

[2] 志水 隆一, 生田 孝: “走査型顕微鏡装置,” 特許 3035612 (2002).

[3] FFTW Home Page, `http://www.fftw.org/`.

[4] Hennessy, Patterson: “Computer Architecture: A Quantitative Approach,” Morgan Kaufman (1996).