Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

ホモジニアス・ヘテロジニアスマルチコアによるDEM生成の高速化と性能評価

岩田健司[†1] 中村良介[†1] 田中良夫[†1]
増田知記[†1,†2] 町田亮介[†1,†2]
小島功[†1] 関口智嗣[†1]

マルチコアによる並列化手法とその効果は数多く報告されているが，応用場面によりそのアプローチは異なり，また精度を十分に担保する必要がある．この論文では，DEM(数値標高モデル) 生成を対象とし，異なるアーキテクチャのマルチコアプロセッサに対し高速化を行い，その性能を評価する．DEM 生成は主に，テンプレートマッチング，異常値内挿，メディアンフィルタといった並列化可能な処理で構成されている．並列化に加え，アルゴリズム面においても十分な精度を確保できた上で，ZNCC 再帰演算や census 変換法など高速な手法を導入した．ヘテロジニアスマルチコアである IBM PowerXCell 8i においては，SPE における DMA 転送の最適化などにより，約 350 倍に高速化された．ホモジニアスマルチコアである Intel Xeon においては，OpenMP による並列化や高速アルゴリズムの導入により，約 38 倍に高速化された．

Evaluation and Acceleration of DEM Generation on Homo- and Heterogeneous multi-core

Kenji Iwata,[†1] Ryosuke Nakamura,[†1]
Yoshio Tanaka,[†1] Tomoki Masuda,[†1,†2]
Ryosuke Machida,[†1,†2] Isao Kojima[†1]
and Satoshi Sekiguchi[†1]

Many approaches of parallelization on multi-core is reported, but the approach should differ by application and guaranteed accuracy enough. This paper described that the acceleration of DEM (Digital Elevation Model) generation on different architecture of multi-core is evaluated. The DEM generation composed of the template matching, interpolation and median filter. The recursive ZNCC calculation and the census transform, etc. was implemented for acceleration with guaranteed accuracy. The optimizaion on heterogeneous multi-core (IBM PowerXCell 8i) was sped up by about 350 times by optimizing the DMA transfer in SPE . The optimizaion on homogeneous multi-core (Intel Xeon) was sped up by about 38 times by using OpenMP and the high-speed algorithm.

1. はじめに

近年のマルチコアプロセッサの普及により，従来の数倍から数十倍の性能を手軽に得られる環境が整ってきており，大量のデータを処理する必要のある信号処理や画像処理などでの応用において，マルチコアでの劇的な高速化が期待される．しかしマルチコアには様々なアーキテクチャが存在しており，それぞれのアーキテクチャに対する並列化手法とその効果は，応用場面により全く異なるものである．たとえば，電波望遠鏡での信号の相関計算における様々なマルチコアアーキテクチャによる高速化が報告されている[1)] が，この結果がそのまま他の応用に適応できるわけではない．一方アーキテクチャによっては，倍精度演算やメモリアクセスなどに制限を受けることがあり，ただ単に高速化するだけではなく，応用面における精度を十分に担保する必要がある．この論文では DEM; Digital Elevation Model(数値標高モデル) の生成をケーススタディーとして，十分な精度を担保できるアルゴリズムを検討した上で，異なるアーキテクチャのマルチコアを用いた高速化手法とその効果を述べる．

衛星画像から生成される DEM は，地理情報システムや立体地図の作成，地形分析などの研究等に利用される．DEM には GTOPO30 や SRTM など，フリーで公開されているものがある[2)]．しかし GTOPO30 は一部で非常に精度が悪く，SRTM の空間分解能は 90m（アメリカ本土では 30m）である．ビルなどの人工物を判別[3)] や、詳細な地形解析にはより高精細な DEM が必要とされる。また地震等による地盤災害の計測などでは，直近のデータと過去のデータの比較（差分）を行う必要がある．これらのような場合には，ASTER[4)] や PRISM[5)] といった高空間分解能の衛星画像データから直接 DEM を生成しなくてはな

†1 産業技術総合研究所 情報技術研究部門
Information Technology Research Institute, National Institute of Advanced Industrial Science and Technology (AIST)
†2 株式会社フィックスターズ
Fixstars Corporation

Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

らない。

我々は GEO Grid (Global Earth Observation Grid; 地球観測グリッド)[6] の構築を行っており，今までに 150 万枚以上の ASTER 画像をインターネット上でアクセス可能とし，さまざまな計算を行う WEB サービスの提供を行っている．この一環として，高解像度の ASTER や PRISM の画像を対象とした高精度な DEM 生成サービスの提供を目指し，最新のマルチコア CPU を用いての高速化を行った．

マルチコアのアーキテクチャには，一般用途の同一コアを搭載したホモジニアスマルチコアと，マルチメディア処理などに特化したコアを混載したヘテロジニアスマルチコアがある．ホモジニアスマルチコアとして Intel x86 アーキテクチャの CPU である Xeon 5500 シリーズ (Nehalem コア)，ヘテロジニアスマルチコアとして，Cell アーキテクチャである IBM の PowerXCell 8i の両方で高速化を行ない，その高速化手法について共通する部分や異なる部分など，アーキテクチャの違いが高速化手法やその効果についてどう影響するのか，アーキテクチャによって高速化手法やその効果がどう異なるのかを示す．

また最適化にあたり，DEM の精度を維持できることを確認しつつ，アルゴリズム面での見直しも行った．DEM 生成は，テンプレートマッチング，異常値内挿，メディアンフィルタにおいて特に計算時間の大部分を占めている．テンプレートマッチングについては，明度の違いに対して頑健な ZNCC が業界標準として用いられているが，計算負荷が高い問題がある．ZNCC を簡略化した NCC については，再帰演算法により高速化できることが報告されている[9] が，ZNCC についても同様な高速化が可能であることを報告する．また，ZNCC よりもさらに頑健な census 変換法[8] についても，SIMD を利用して，高速な実装を行った．異常値内挿については，内部の計算精度を落とし，最終結果に影響が無いことを確認した．メディアンフィルタについては，SIMD に最適化されたソートアルゴリズムである AA-sort[11] を導入した．

この論文の構成は，2. では今回の最適化に使用したアーキテクチャについて，3. では DEM 生成の概要とプロファイリングについて，4. ではテンプレートマッチングの高速化について，5. で Cell における並列化，高速化手法とその効果について，6. で Xeon における並列化，高速化手法とその効果について，7. で全体の性能評価について述べ，8. で本論文をまとめる．

2. アーキテクチャの概要

今回の最適化に使用したプラットホームのスペックを，**表 1** に示す．各プロセッサについて以下に述べる．

表 1 評価に用いたプラットホームのスペック
Table 1 Specs of platform used in the experiments

	CPU	動作周波数	CPU 数	コアの構成
IBM BladeCenter QS22	PowerXCell 8i	3.2GHz	2	2 PPE + 16 SPE
HP Z800 Workstation	Xeon X5550	2.66GHz	2	8 Nehalem core

2.1 IBM PowerXCell 8i

IBM 社製の PowerXCell 8i は，メディア処理に適した Cell アーキテクチャのプロセッサであり，Cell/B.E. の倍精度浮動小数点演算性能を 5 倍に強化したプロセッサである．今回の DEM 生成においては倍精度演算が必要なため，Cell/B.E. より適していると言える．また，特殊なメモリを利用する Cell B.E. に対し，汎用の DDR2 DRAM を利用することからメモリ搭載の制限が少なく，より大規模計算に適している．

PowerXCell 8i は，通常の演算命令を実行する PPE を 1 基と，SIMD 演算を高速に実行する SPE を 8 基で構成されている，ヘテロジニアスマルチコアである．PPE はそれほど高速ではないため，高速化には SPE に処理を振り分けることになる．SPE を用いた並列化には，256KB のローカルストレージへの DMA 転送など，専用のプログラム設計が必要となり，Intel Xeon と比べ，作業を多く要する．

今回は，PowerXCell 8i を 2 基搭載した IBM BladeCenter QS22（以下 QS22 と呼ぶ）を評価に用いた．

2.2 Intel Xeon 5500 シリーズ (Nehalem)

Intel 社製の Xeon 5500 シリーズは Nehalem コアの Intel アーキテクチャCPU であり，同一性能の 4 コアを統合したホモジニアスマルチコアプロセッサである．並列化に関しては，すべてのコアで同一のプログラムが動作するため，開発は比較的容易である．SIMD として SSE4.2 までサポートしており，POPCNT など 4.2 節で述べる census 変換法に有利な命令などが実装されている．一般的な PC サーバーが利用できるため，調達が容易であるメリットがある．

今回は HP Z800 Workstation を（以下 Z800 と呼ぶ）評価に用いた．

3. DEM 生成の分析

3.1 DEM 生成の手順

DEM 生成の手順を**図 1** に示す．前処理では，衛星に異なる角度で装着された複数のセン

Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

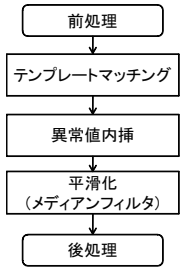


図 1 DEM 生成の手順
Fig. 1 Procedure of DEM generation

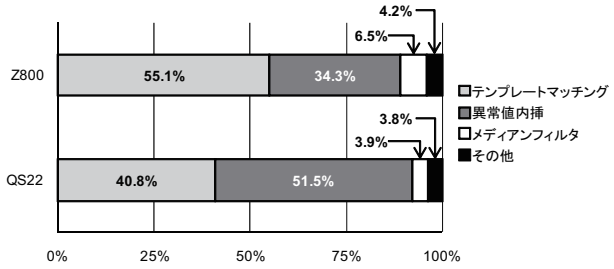


図 2 最適化前の処理時間の構成
Fig. 2 Computation time before optimization

サーによる同一エリアを観測した 2 枚の地形画像データを入力し，歪みの補正などを行う，

画像データの対応関係を調べ disparity map（視差分布）を算出するステレオマッチングが主処理である．ステレオマッチングは，同一地点を同定するためのテンプレートマッチング，欠損データを周囲のデータから補完する異常値内挿，ノイズを除去するためのメディアンフィルタによる平滑化処理から構成される．また，誤対応の抑止，およびマッチング範囲の絞り込みのため，coarse-to-fine 法を併用する．これは，はじめは解像度の低い coarse 画像を用いて大局的な対応付けを行い，徐々に解像度の高い fine 画像を用いることで，詳細な disparity map を生成するものである．よって，テンプレートマッチング，メディアンフィルタ，異常値内挿の各処理は，coarse-to-fine 法の各ステージごとに繰り返し実行される．今回は，10 ステージの coarse-to-fine 法を用いている．

後処理では，disparity map より衛星の位置情報から緯度経度や地球固定座標系への変換及び，視差を標高に変換することで，DEM 画像を生成し，出力する．

3.2 最適化前のプロファイル

C 言語で記述された同一の DEM 生成プログラムを Z800(Xeon)，QS22(Cell) の双方でコンパイルし，3962 × 4200 画素の ASTER 画像に対する各処理における計算時間の割合を計測した．計測結果を**図 2** に示す．このプログラムはシングルスレッドで動作しており，Cell においては PPE のみを用いている．

このように，テンプレートマッチングと異常値内挿に多くの時間を要している．Cell は Xeon に比べ，異常値内挿の占める割合が多いが，これは Xeon では 80bit 精度の演算を x87 命令で処理しているのに対し，Cell では 128bit 精度の演算をライブラリ呼び出しにより行っているためである．

テンプレートマッチング，異常値内挿，メディアンフィルタの各処理は，画像を分割して各コアに処理を割り振ることで，並列化が可能であり，すでに文献 7) において報告しており，マルチスレッド化によるプロセスレベルでの並列化，SIMD による命令レベルでの並列化により最適化を行った．また，より高精細なデータを用いるには高速化が不十分であることから，アルゴリズムの見直しにも着手した．

4. 高速なテンプレートマッチング法の提案

DEM 生成におけるテンプレートマッチングにおいては正規化相互相関 (ZNCC; Zero mean Normalized Cross Correlation) が一般的に利用されるが，積和演算を多用するため計算量が多い．そこで，予備実験により精度に問題のないことが確認された，census 変換法[8] を導入する．また，新たに ZNCC の再帰演算法を提案する．

4.1 ZNCC

2 枚の画像を基準画像 I，探索画像 J，視差のベクトルを $\mathbf{p}$，マッチングウインドウ内の座標の集合を W とした場合，ZNCC は，

$$C_{ZNCC}(\mathbf{p},W)=\frac{\sum_{\mathbf{q}\in W}\left\{I(\mathbf{q})-\bar{I}(W)\right\}\left\{J(\mathbf{p}+\mathbf{q})-\bar{J}(\mathbf{p},W)|\right\}}{\sqrt{\sum_{\mathbf{q}\in W}\left\{I(\mathbf{q})-\bar{I}(W)\right\}^2}\sqrt{\sum_{\mathbf{q}\in W}\left\{J(\mathbf{p}+\mathbf{q})-\bar{J}(\mathbf{p},W)\right\}^2}}. \quad (1)$$

ここで，

$$\bar{I}(W)=\frac{1}{|W|}\sum_{\mathbf{q}\in W}I(\mathbf{q}), \quad (2)$$

Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

$$\bar{J}(\mathbf{p},W)=\frac{1}{|W|}\sum_{\mathbf{q}\in W}J(\mathbf{p}+\mathbf{q}). \tag{3}$$

ただし $|W|$ は集合 W の元の個数である．

4.2 census 変換法

census 変換法[8] は次のようになる．

$$\xi(\mathbf{p},\mathbf{p}';K)=\begin{cases}1 & K(\mathbf{p}')<K(\mathbf{p}),\\ 0 & othewise.\end{cases} \tag{4}$$

$$C_{census}(\mathbf{p},W)=\left|\{\xi(\mathbf{q},\mathbf{r};I)\neq\xi(\mathbf{p}+\mathbf{q},\mathbf{s};J)\mid\mathbf{q}\in W\}\right|. \tag{5}$$

ただし，$\mathbf{r},\mathbf{s}$ はそれぞれ I,J におけるマッチングウインドウの中心座標を示す．すなわち中心の画素値との大小を 0,1 のビット列で表し，そのハミング距離を基準として用いる．明度差やノイズに対して極めて頑健な方法であり，さらに大小比較とビット列の計算のみで実装できることから，積和演算を要する NCC や ZNCC と比べて高速であるとされている．

4.3 ZNCC の再帰演算法

ZNCC の平均を 0 としたものが，NCC(Normalized Cross Correlation) であり，

$$C_{NCC}(\mathbf{p},W)=\frac{\sum_{\mathbf{q}\in W}\{I(\mathbf{q})J(\mathbf{p}+\mathbf{q})\}}{\sqrt{\sum_{\mathbf{q}\in W}I^2(\mathbf{q})}\sqrt{\sum_{\mathbf{q}\in W}J^2(\mathbf{p}+\mathbf{q})}}, \tag{6}$$

となる．NCC によるステレオマッチングでは，再帰相関演算を導入することにより計算量を大幅に減らすことができることが報告されている[9]．式 6 を関数 N,Q_I,Q_J を用いて

$$C_{NCC}(\mathbf{p},W)=\frac{N(\mathbf{p},W)}{\sqrt{Q_I(W)}\sqrt{Q_J(\mathbf{p},W)}}, \tag{7}$$

と記述する．ただし，

$$N(\mathbf{p},W)=\sum_{\mathbf{q}\in W}\{I(\mathbf{q})J(\mathbf{p}+\mathbf{q})\}, \tag{8}$$

$$Q_I(W)=\sum_{\mathbf{q}\in W}I^2(\mathbf{q}), \tag{9}$$

$$Q_J(\mathbf{p},W)=\sum_{\mathbf{q}\in W}J^2(\mathbf{p}+\mathbf{q}). \tag{10}$$

である．ここで Q_I は視差によらない値であり，ステレオマッチングにおいては C_{NCC} の最大値を探索するものであるから，計算を省略できる．NCC の計算で最も負荷が高いのは，

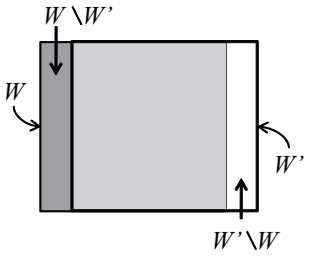


図 3 マッチングウインドウ
Fig. 3 Window of template matching

積和演算の N,Q_J の計算である．マッチングウインドウを W から移動した W' について考えると，

$$N(\mathbf{p},W')=N(\mathbf{p},W)-N(\mathbf{p},W\setminus W')+N(\mathbf{p},W'\setminus W) \tag{11}$$

$$Q_J(\mathbf{p},W')=Q_J(\mathbf{p},W)-Q_J(\mathbf{p},W\setminus W')\ +Q_J(\mathbf{p},W'\setminus W) \tag{12}$$

となる．ここで $W\setminus W'$ は，W から W' を引いた差集合である．$W,W',W\setminus W',W'\setminus W$ の関係を**図 3** に示す．すなわち W から，W' にずらした場合の N,Q_J の計算は，W における N,Q_J の値から，W' に含まれなくなった領域 $W\setminus W'$ の値を引き，W' に新たに加わった領域 $W'\setminus W$ の値を加えることで算出できる．このようにウインドウ W' の全点の積和を毎回計算する必要が無くなる．さらに図 3 では $W\setminus W',W'\setminus W$ は縦長の領域であるが，この領域に対して縦の方向にも同様に再帰的に算出できることから，積和演算回数を $1/|W|$ に減らすことができる．

ZNCC における再帰演算法は報告されていないが，NCC と同様に導入することが可能である．式 1 に式 2，式 3 を代入することで，平均の引き算を Σ の外に出すことが可能である．すなわち，式 1 を関数 M,D_I,D_J を用いて

$$C_{ZNCC}(\mathbf{p},W)=\frac{M(\mathbf{p},W)}{\sqrt{D_I(W)}\sqrt{D_J(\mathbf{p},W)}}, \tag{13}$$

と記述すると，M,D_I,D_J はそれぞれ，

$$\begin{aligned}M(\mathbf{p},W)&=\sum_{\mathbf{q}\in W}\left\{I(\mathbf{q})-\bar{I}(W)\right\}\left\{J(\mathbf{p}+\mathbf{q})-\bar{J}(\mathbf{p},W)\right\}\\&=\sum_{\mathbf{q}\in W}I(\mathbf{q})J(\mathbf{p}+\mathbf{q},W)-|W|\bar{I}(W)\bar{J}(\mathbf{p},W),\end{aligned} \tag{14}$$

Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

$$D_I(W) = \sum_{\mathbf{q}\in W} \left\{I(\mathbf{q}) - \bar{I}(W)\right\}^2 = \sum_{\mathbf{q}\in W} I^2(\mathbf{q}) - |W|\bar{I}^2(W), \quad (15)$$

$$D_J(\mathbf{p},W) = \sum_{\mathbf{q}\in W} \left\{J(\mathbf{p}+\mathbf{q}) - \bar{J}(\mathbf{p},W)\right\}^2 = \sum_{\mathbf{q}\in W} J^2(\mathbf{p}+\mathbf{q}) - |W|\bar{J}^2(\mathbf{p},W), \quad (16)$$

となる．これらを N, Q_I, Q_J を用いて記述することで以下のようになる．

$$M(\mathbf{p},W) = N(\mathbf{p},W) - |W|\bar{I}(W)\bar{J}(\mathbf{p},W), \quad (17)$$

$$D_I(W) = Q_I(W) - |W|\bar{I}^2(W), \quad (18)$$

$$D_J(\mathbf{p},W) = Q_J(\mathbf{p},W) - |W|\bar{J}^2(\mathbf{p},W). \quad (19)$$

D_I の計算も NCC の場合と同じく省略可能である．$\bar{I}, \bar{J}$ の計算においても式 11，12 と同様に再帰計算が可能であることから，ZNCC における M, D_J もすべて再帰演算が可能である．

5. Cell における並列化，高速化

Cell(IBM PowerXCell 8i) に対して行った最適化手法と，その効果を以下に述べる．なお，計算時間の計測には図 2 と同様の，3962 × 4200 画素の ASTER 画像を用いている．

5.1 異常値内挿の精度

異常値内挿は次のような手順で行う．

(1) ステレオマッチングの相関値がある閾値より低い部分や，傾斜がある閾値より大きい部分を異常領域としてマーキングする．
(2) 異常領域としてマーキングされた部分の連結領域を取り出す．
(3) 異常連結領域の周囲にある視差が正しく得られた部分から，最小二乗法により値を内挿する．

ここでの最小二乗法において，long double 精度 (128bit) の演算を行っていた．高い精度が必要な座標系変換部分と共通のプログラムを使用していたためであるが，異常値内挿においては，倍精度 (64bit) であっても得られる結果は全く変わらないことを確認した．PPE で動作速度を確認したところ，83.7 秒から，36.1 秒に約 2.3 倍に改善された．

文献 7) では，Cell/B.E. の SPE では倍精度演算が遅いため，単精度 (32bit) を用いていた．ただし，全体の 1.3%程度において平均で 1m 程度の誤差が発生する．PowerXCell 8i の SPE では倍精度の演算速度が改善されているため，今回は倍精度を用いることとする．

表 2 テンプレートマッチングの最適化による処理時間（秒）
Table 2 Computation time by optimization of template matching

最適化方法	処理時間	倍率
並列化なし (PPE のみ)	1721.29	1.00
ZNCC 計算の SPE での並列化	153.98	11.18
ZNCC 計算の SIMD 化	135.87	12.67
DMA 転送の最適化	34.49	49.91
ZNCC 再帰演算	30.18	57.03
その他前処理等を SPE 化	2.25	765.02

5.2 テンプレートマッチングの並列化，高速化

Cell における PPE は，それほど高速なプロセッサではない.，特に PPE のランダムアクセスは非常に遅いため、SPE に処理を振り分けることで，高速なローカルストレージ (LS) へのアクセスとなり，劇的に速度が向上する．

ただし OpenMP などの汎用的な並列化ライブラリを用いた場合には効率がよくないため，高い性能を達成するにためには個別に実装する必要がある．

ZNCC によるテンプレートマッチングの並列化，高速化においては，

(1) SPE への処理振り分けによる並列化．
(2) SIMD による命令レベルでの並列化．
(3) DMA 転送の最適化．
(4) ZNCC 再帰演算の導入．

といったアプローチを併用した．テンプレートマッチングのみに要した処理時間を**表 2** に示す．

まず，ZNCC の計算部分を，SPE に振り分け並列化を行うことで，大きく性能が向上した．さらに，SPE での処理を SIMD 化したが，それほど性能の向上は見られなかった．SPE では LS にデータを DMA 転送してから処理する必要があり，ここでは，ZNCC 計算の 1 つのウインドウ単位での振り分けを行っており，データの転送量が多く，バスがボトルネックとなっていると思われる．

Cell における高速化では，LS への DMA 転送を如何に効率的に行うかが肝要となる．文献 7) では，ラインごとに処理を分割し，振り分けを行っているが，今回は**図 4** のように画像を横幅 16 または 32 ピクセルの一定の幅で複数の縦長のブロックに分割し，空いている SPE へ処理を振り分けるように実装した．このような分割方法のメリットとして，**図 5** の

Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

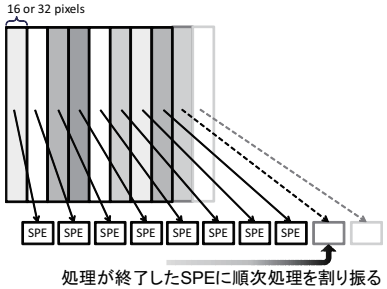


図 4 縦長のブロック分割による DMA 転送
Fig. 4 DMA transfer by portrait block splitting

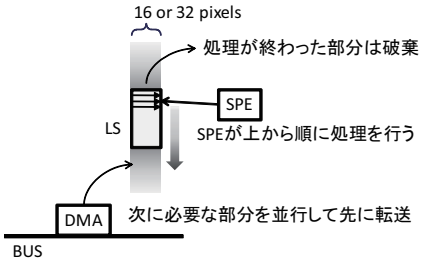


図 5 DMA 転送時間の隠蔽
Fig. 5 Latency hiding of DMA transfer

ように，SPE では各ブロックを上のラインから順次処理を行いながら，次のラインの処理に必要となるデータを並行して DMA 転送することで，DMA 転送時間の隠蔽を効率よく行うことができる．これにより，約 4 倍の高速化となった．これは上記の理由に加え，PPE からのタスクの発行回数を大幅に減少させられること，Cell で効率の悪い 1byte や 4byte 単位の転送から効率の良い 16byte 単位の転送に変更できることによるものも含まれる．また，縦長のブロックを用いているため，ウインドウ内の平均値 $\bar{I}, \bar{J}$ を SPE 内で再帰演算することができた．

さらに ZNCC 再帰演算を導入した．SPE での処理時間を**表 3** に示す．このように SPE 内部での ZNCC の計算時間は約 3.7 倍高速となった．最後に ZNCC 計算以外の前処理などを SPE で並列化し，全体的な最適化を行うことで 765 倍の高速化となった．

表 3 ZNCC 再帰演算における SPE の処理時間
Table 3 Computation time at SPE of recursive correlation calculation

処理内容	再帰演算なし (ミリ秒)	再帰演算あり (ミリ秒)
DMA 転送	41.33	100.99
ZNCC の計算	5309.36	1444.79

表 4 SPE における SIMD 化の効果
Table 4 Effectiveness of SIMD on SPE

処理	SIMD 化前 (秒)	SIMD 化後 (秒)	倍率
異常値内挿	12.06	1.42	8.51
メディアンフィルタ	7.38	1.51	4.90
処理全体	26.93	7.21	3.74
待ち時間を含む全体	39.60	11.78	3.36

なお，census 変換法によるテンプレートマッチングは ZNCC による最適化後に，SPE による ZNCC の計算部分を差し替えた．このようなビット列比較には，spu_cntb などの SIMD 命令を用いることで効率的な実装が可能であった．

5.3 その他の処理の並列化，高速化

異常値内挿やメディアンフィルタなどの処理も，テンプレートマッチングと同様に画像を縦長のブロックに分割し 8 個の SPE に処理を振り分け，SPE における各処理を SIMD 化し高速化した．SIMD 化の効果を**表 4** に示す．異常値内挿については，傾斜角度の計算や，最小二乗法の SIMD 化の効果が大きい．

メディアンフィルタは，注目値を中心とするある一定の大きさのウインドウ内の値をソートし，その中間値で置き換える．ウインドウのサイズは 3×3 から 9×9 程度を用いており，データがそれほど大きくない時に効率的な combsort[10] を用いていた．今回は，combsort を SIMD 向けに最適化した AA-sort[11] における In-core algorithm を導入して高速化を行った．このアルゴリズムは，SIMD を利用して比較処理と入れ替え処理を 4 並列に行う．ここにおいてパフォーマンス上の問題となる，アラインされていないメモリへのアクセスを完全に排除しているという特徴がある．このアルゴリズムにより，平滑化処理は約 5 倍に高速化された．

処理全体としては，3.74 倍に向上している．タスクの取得及び待ち合わせの時間を含めると 3.36 倍となった．

Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

表 5 OpenMP による並列化の効果
Table 5 Effectiveness of parallelization using OpenMP

処理	シングルスレッド (秒)	OpenMP による 8 スレッド (秒)	倍率
テンプレートマッチング	25.80	4.40	5.86
異常値内挿	18.40	3.13	5.88
メディアンフィルタ	10.03	1.36	7.39
その他	12.29	2.55	4.82
全体	66.53	11.44	5.81

6. Intel Xeon における並列化，高速化

Intel Xeon に対して行った最適化手法と，その効果を以下に述べる．ここでの計算時間の計測には図 2 と同し，3962 × 4200 画素の ASTER 画像を用いている．なお，異常値内挿の精度については Cell と同様に 64bit 精度を用いる．メディアンフィルタについても Cell と同じく AA-sort[11] を導入している．

6.1 OpenMP による並列化

OpenMP を用いて並列化を行った．テンプレートマッチングや異常値内挿処理では，画像のラインごとに処理を行う for ループが各処理の先頭部分に存在することから，この部分に OpenMP の制御命令を追加する形で実装した．

OpenMP を使用しない場合と，OpenMP により 8 スレッドに並列化した場合の比較を**表 5** に示す．OpenMP のような汎用ライブラリを用いるよりも，より詳細を精査しながら手動でスレッド化した方が，より高速になる可能性があるが，処理時間の多くを占めるステレオマッチング，異常値内挿，平滑化の各処理において，5.86 から 7.39 倍に高速化されており，8 コアを使用している現時点において，手動でスレッド化することによる改善の余地はあまり大きくないと判断し，OpenMP での並列化でとどめることとした．

6.2 テンプレートマッチングにおける高速化

テンプレートマッチングの高速化のため，census 変換法の SIMD による実装と，ZNCC 再帰演算を導入した．処理時間を**表 6** に示す．

まず census 変換法については，16bit ごとに大小を示す 0,1 を保持し，8 要素を並列に処理する方法では，SIMD レジスタへのデータの並び替えにオーバーヘッドがかかり，高速化の効果は小さかった．そこで，PMOVMSKB 命令により大小を示す 0,1 をビット列化しておき，ハミング距離を POPCNT 命令で求める方法に変更したところ，大きな改善効果があった．

表 6 Xeon におけるテンプレートマッチングの処理時間
Table 6 Computation time of template matching on Xeon

最適化方法	処理時間 (秒)
census SIMD なし	18.4
census SIMD 8 並列	15.3
census SIMD ビット列化	5.4
ZNCC 再帰演算なし	40.1
ZNCC 再帰演算あり	4.4

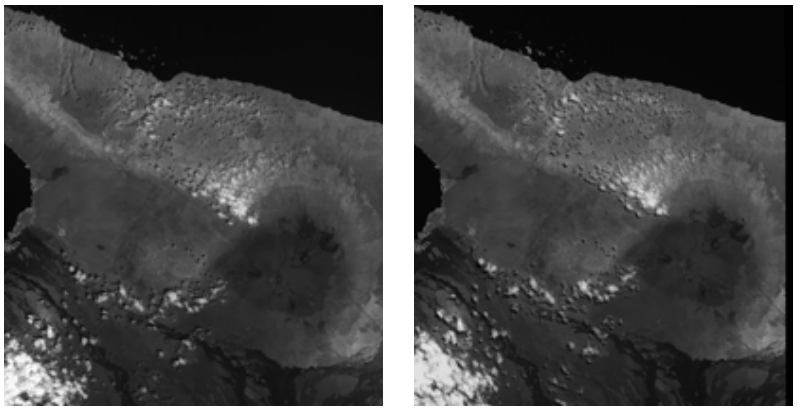


図 6 ASTER 画像
Fig. 6 ASTER images

ZNCC については，再帰演算を導入することで，大幅に高速化された．ZNCC 再帰演算は SIMD 化されていないが，SIMD 化した census 変換法よりも高速であった．

7. 全体の性能評価

性能評価には，2000 × 2000 と 3962 × 4200（**図 6**）の衛星画像を用いた．結果を**表 7** に示す．処理時間は画像の大きさにほぼ比例している．

Cell を搭載した QS22 においては PPE が比較的低速であることから，約 180〜380 倍の高速化を実現した．Intel Xeon を搭載した Z800 においては，単一のコアの性能が高いため性能向上率自体は Cell ほどではないが，それでも 26 倍から 37 倍の高速化を実現した．マルチコアによる並列化だけでなく，アルゴリズム面での改善効果も大きい．

Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

表 7 アルゴリズム，アーキテクチャ別の処理時間
Table 7 Computation time according to algorithms and architectures

アルゴリズム	マシン	2000 × 2000 最適化前 (秒)	2000 × 2000 最適化後 (秒)	2000 × 2000 倍率	3962 × 4200 最適化前 (秒)	3962 × 4200 最適化後 (秒)	3962 × 4200 倍率
census	QS22	846.41	3.41	248.21	4181.37	11.03	379.09
	Z800	80.11	3.03	26.41	397.19	14.05	28.28
ZNCC	QS22	844.93	3.60	234.70	4222.84	11.93	353.97
	Z800	87.27	2.70	32.30	429.25	11.44	37.51

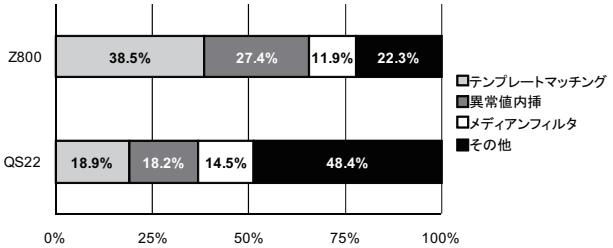
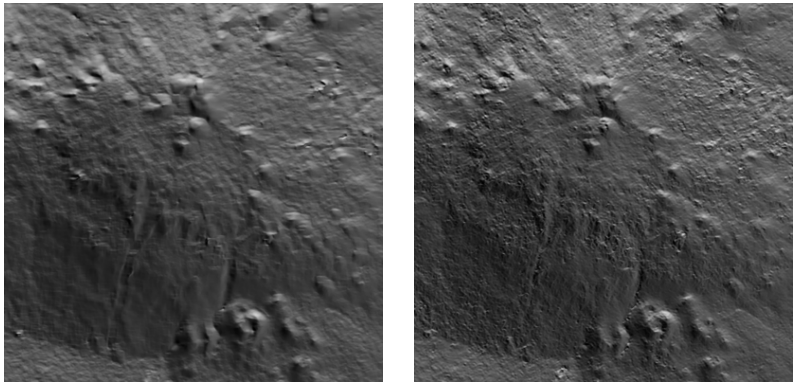


図 7 最適化後の処理時間の構成
Fig. 7 Computation time with optimization

ZNCC アルゴリズムにおいて，多くのコア数 (16 SPE) を擁するの QS22 と，半分のコア数である Z800 がほぼ互角の性能であった．QS22 と Z800 による ZNCC アルゴリズムによる最適化後の処理時間の構成を**図 7** に示す．QS22 では「その他」に含まれるファイル I/O 部分に時間を要していた．

テンプレートマッチングのアルゴリズムである ZNCC と census 変換法を比較すると，QS22 ではほぼ互角，Z800 では ZNCC の方が高速であった．ZNCC は計算負荷が高いため，census 変換法が開発されたという経緯がある[8] が，最新のマルチコアに ZNCC 再帰演算を導入し最適化した場合では，同等もしくは逆転する結果となった．最終的に生成された DEM について，図 6 における山頂付近の Hill-shade 画像を**図 8** に示す．このように，どちらの方法を用いても良好な DEM が生成されている．ただし (b)census の方が (a)ZNCC と比べ，精細感の高い画像が得られており，census 変換法の頑健性が見て取れる．どちらも高速化が可能であることから，業界標準である ZNCC か，より精細な DEM を得られる census か，利用者側で選べるような構成を取ることが可能である．



(a) ZNCC (b) census

図 8 生成された DEM の Hill-shade 画像
Fig. 8 Hill-shade images of DEM genarated

8. ま と め

DEM の生成を高速化するため，アルゴリズム及びアーキテクチャの両面から検討を行い，ヘテロジニアスマルチコアである Cell (IBM PowerXCell 8i) とホモジニアスマルチコアである Intel Xeon の双方に実際に実装し，その性能を検証した．計算時間の多くを占めるテンプレートマッチング，異常値内挿，メディアンフィルタといった各処理は並列性が高く，マルチコアを用いた高速化に向いていた．

ヘテロジニアスマルチコアである Cell においては，各処理を順次最適化するアプローチを取った．まずテンプレートマッチングから，処理を SPE へ割り振り，SIMD 化，高速なアルゴリズムの導入，DMA 転送の最適化を行った．以降，異常値内挿など他の処理を最適化した．このようにストリーム処理に適した SPE を用いる Cell での最適化は，個別の処理ごとにそれぞれ異なる並列化のプログラミングが必要である．

ホモジニアスマルチコアである Xeon では，OpenMP を用いて最初にすべての処理の並列化を行った．今回のケースでは並列性が高いためその効果が大きく，Cell と比べで並列化に関する最適化作業の手間は大幅に少なくて済んだ．その後，個別の処理について高速なアルゴリズムの導入や SIMD 化による最適化を行った．

今後は GEO Grid[6] の一環として，高速な DEM 生成を WEB サービスとして提供する

Vol.2009-ARC-186 No.3
Vol.2009-HPC-123 No.3
2009/11/30

システム構築を行う．GPGPU については，今回は倍精度演算が必要であることから対象としなかったが，この問題も解決しつつあるため今後は最適化作業を進める予定である．また，より高空間分解能の衛星画像の利用を想定したクラスタリングの検討も行う．

謝辞 census 変換法による高速化について助言頂いた日本原子力研究開発機構の武宮氏に感謝の意を表する．

参 考 文 献

1) Rob V. van Nieuwpoort and John W. Romein: Using Many-Core Hardware to Correlate Radio Astronomy Signals, Proceedings of the ACM International Conference on Supercomputing (ICS'09), pp. 440-449, June 8-12, (2009).
2) http://eros.usgs.gov/products/elevation/
3) 高木崇，川島英之，天笠俊之，北川博之：イベント検知に基づく衛星画像と Web コンテンツの統合，DEIM2009 (2009).
4) Y. Yamaguchi, B. A. Kahle, M. Pniel, H. Tsu, and T. Kawakami: " Overview of Advanced Spaceborne Thermal Emission and Reflection REadiometer (ASTER), " IEEE Trans. Geosci. Remote Sens., vol. 36, no. 4, pp. 1062-1071, (1998).
5) 度會英教，川西登音夫，大澤右二，松本暁洋，田殿武雄：パンクロマチック立体視センサ (PRISM)，信学技報 SANE2006-64, pp.1-6 (2006). (2006)
6) S. Sekiguchi, Y. Tanaka, I. Kojima, N. Yamamoto, S. Yokoyama, Y. Tanimura, R. Nakamura, K. Iwao and S. Tsuchida: Design Principles and IT Overview of the GEO Grid, IEEE Systems Journal, Vol.2, No.3, pp.374-389 (2008).
7) 安田絹子，江口剛，上田孝，近藤伸宏，福田悦生，原誠一，中村良介，田中良夫，関口智嗣：Cell/B.E. プロセッサによるステレオマッチングソフトウェアの高速化 (最適化・高速化)，情報処理学会研究報告 HPCS，vol. 59，pp.7-12 (2007).
8) R. Zabih and J. Woodfill: Non-parametric Local Transforms for Computing Visual Correspondence, Proc. ECCV'94, pp.151-158 (1994).
9) O. Faugeras, B. Hotz, H. Mathieu, T. Vieville, Z. Zhang, P. Fua, E. Thwron, L. Moll, G. Berry, J. Vuillemin, P. Bertin and C. Proy : Real time correlation-based stereo: Algorithm, implementations and applications, INRIA Technical Report, RR-2013 (1993).
10) S. Lacey and R. Box: A Fast, Easy Sort. Byte Magazine, April, pp. 315-320 (1991).
11) H. Inoue, T. Moriyama, H. Komatsu, and T. Nakatani: AA-Sort: A New Parallel Sorting Algorithm for Multi-Core SIMD Processors, PACT2007, pp.189-198 (2007).