卒業論文

題目

Cell/B.E.における動画像動き探索処理の高速化

指導教員

近藤 利夫 教授，大野 和彦 講師

2009年

三重大学 工学部 情報工学科
計算機アーキテクチャ研究室

中村 佳史 (406843)

内容梗概

近年，次世代のスーパーハイビジョン試験放送が予定されるなど動画像の高精細化がデジタル化と共に進んでいる．それに伴う処理量の増大は著しく動画像のリアルタイムでのエンコードが容易ではなくなっている．エンコード処理の要である動き探索処理が膨大な演算量を必要とするからである．従って，現状ではスーパーハイビジョンの実時間符号化システムを専用機器を用いて構成すると多大なコストを要する．そこで本研究では，スーパーハイビジョンの実時間符号化の要求にも応えられる動き探索の高速処理を安価な並列処理システム上で実現することを目的とする．

並列処理システムの構成プロセッサには，動き探索処理に適した並列処理構成を持つ Cell Broadband Engine(Cell/B.E.) を用いる．このプロセッサを搭載する安価な Play Station 3(PS3) を並べるだけで要求性能を満たす並列処理システムが構成可能になるからである．

本研究では，動き探索の基本演算である差分絶対値和の計算で使用される参照画像データを再利用することで，ロードのオーバーヘッドを低減し，動き探索の高速化を実現する．

オープンソースの x264 ソフトウェアエンコーダに参照画像の画素値データを再利用する処理を実装し，性能評価を行った．その結果，参照画像データの再利用により，動き探索処理の速度が向上し，高速化されることが示された．

Abstract

Recent years, the high definition of video images has made rapid progress with the digital imaging, such as actually, as early as 2015, the trial of the next generation's Super Hi-Vision broadcasting is scheduled. The encode of high definition video images is not easy in realtime by increasing the amount of the processing. The motion search of the encode processing requires a huge amount of the operation. A Super Hi-Vision realtime encoder becomes expensive. Therefore, this research aims to achieve a high speed processing of the motion search by a cheap parallel processing system that can satisfy the demand of Super Hi-Vision realtime encoding.

The processor unit of the parallel processing system use Cell Broadband Engine(Cell/B.E.) that have the parallel processing structure that is appropriate for the motion search. Because it is used in a cheap Play Station 3(PS3).

In this research, sum of absolute difference is a basic arithmetic of the motion search. And, its calculation can reuse reference image data. The overhead of the load reduce by reuse in reference image data, and achieve speed up of the motion search.

It was implemented a process which reusing the pixel value data of reference image on the x264 software encoder of open source. Performance has been evaluated. As a result, the speed of the motion search was improved by the reuse of the reference image data.

目 次

図 目 次

表 目 次

1　はじめに

次世代のスーパーハイビジョン試験放送が平成 27 年から予定されるなど近年動画像の高精細化が進んでいる．それに伴い，動画像のエンコード処理が増大し，リアルタイムでの動画像処理が容易ではなくなっている．エンコード処理の要の動き探索処理で動画像が高精細化するにつれて，動き探索の点数がフレーム面積の 2 乗に比例して増大し，動き探索の処理量がさらに増大するからである．それでも，ハイビジョンまでのリアルタイム符号化処理はマルチコアプロセッサを複数搭載した PC で実現可能なところまで到達しているので，それらをさらに複数用いる並列処理が実現できれば，スーパーハイビジョンのような超高精細の動画像符号化も可能になる．

Cell Broadband Engine(Cell/B.E.) はマルチコアプロセッサであり，2 種類のプロセッサコアから構成されている．また，コア間の並列処理により，高い性能を発揮することができる．その Cell/B.E. の並列処理構成に適した処理の 1 つに動き探索がある．なぜなら，Cell/B.E. は，複数個のコアを持つので，動き探索を並列処理できるからである．また，最大 16 個の値を格納できる 128 ビットレジスタを持つので，動き探索での差分絶対値和の計算を高速に行うことができるからである．

動き探索処理で行われている差分絶対値和 (SAD) の計算では，参照画像データを使用している．オープンソースの x264 ソフトウェアエンコーダでは，この差分絶対値和の計算の度に参照画像データを読み込んでいるためロードのオーバーヘッドが速度向上の障害となっている．しかし，このロードオーバーヘッドは，一旦ロードした参照画像データを近傍点の探索の時にも利用 (再利用) することで低減できる．

動き探索法には，全探索，追跡型探索などがある．全探索では，探索範囲の探索点全てに対して差分絶対値和の計算を行うため，演算量が多くなる．追跡型探索では，探索中心とその近傍の決められた探索点 (探索パタン) に対する差分絶対値和を，探索中心が探索パタンの局小点となるまで繰り返す．ただし，探索中心は毎回局小点に移動する．全探索と追跡型探索の探索範囲は同じであるので，追跡型探索は全探索より演算量を低減できる．また，追跡型探索は，近年の主流となっており，演算量低減と検出精度向上に不可欠とも言える探索手法である．ダイヤモンド，ヘキサゴン，スクエアの 3 つは，追跡型探索を構成する基本の探索パタンである．従来，スクエアは高効率アルゴリズムの主要な探索パタンではなく，ダイヤモンド，ヘキサゴンより探索効率が落ちるとされ，利用価値の低い基本の探索パタンと見なされてきた．しかし，ダイヤモンド，

ヘキサゴンは，探索点間の再利用可能な共通部分の面積が小さく，参照画像データの探索点当たりのロード回数が多くなる．そこで，参照画像データの探索点当たりのロード回数が少なくなるスクエアサーチを使用する．参照画像のロードデータを再利用することによって，新たなデータのロード回数を減らし，動き探索処理を高速に行うことが可能となる．

この参照画像の画素値データの再利用によって，動き探索処理を高速化できれば，Cell/B.E. によるスーパーハイビジョン符号化の実現性が高まる．そこで，本研究ではこのアプローチに最も適した追跡形のスクエアサーチの高速化を追求する．

2 動き探索とその問題点

2.1 差分絶対値和 (SAD: Sum of Absolute Differences)

差分絶対値和とは，符号化対象画像と参照画像の違いを数値で表現したものである．差分絶対値和の値が小さい程，符号化対象画像と参照画像の違いが少なく，似ていることになる．差分絶対値和の値が大きい程，符号化対象画像と参照画像の違いが大きく，似ていないことになる．差分絶対値和の求め方を以下に示す．

1. 符号化対象画像と参照画像の各画素値の差分を計算する．

2. 各画素値の差分を絶対値にする．

3. 各画素値の絶対値の合計を計算する．

4. 差分絶対値和が求まる．

例を用いた差分絶対値の計算過程を図 2.1 に示す．

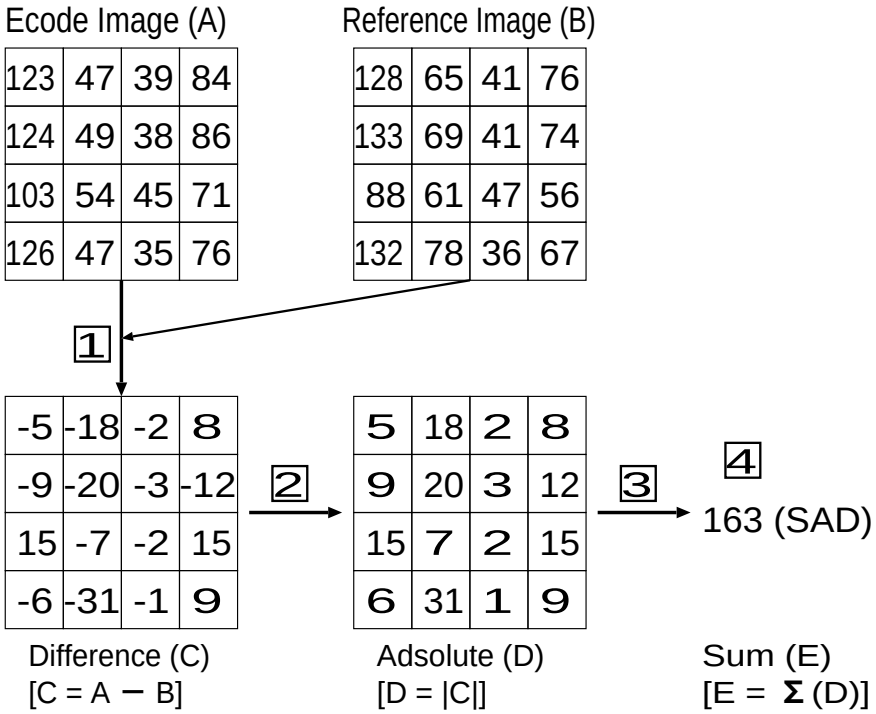

図 2.1: 差分絶対値和の計算過程

2.2 動き探索

動き検出処理では，参照画像と符号化対象画像の間で映っている物がどのように動いたかを検出して，動きベクトルを求める．この画像間の

動きベクトルと求める処理には，動き探索処理が用いられる．動き探索処理では，参照画像と符号化対象画像の差分絶対値和 (SAD) を利用して処理が行われる．

追跡形の動き探索処理の流れを図 2.2 に示す．

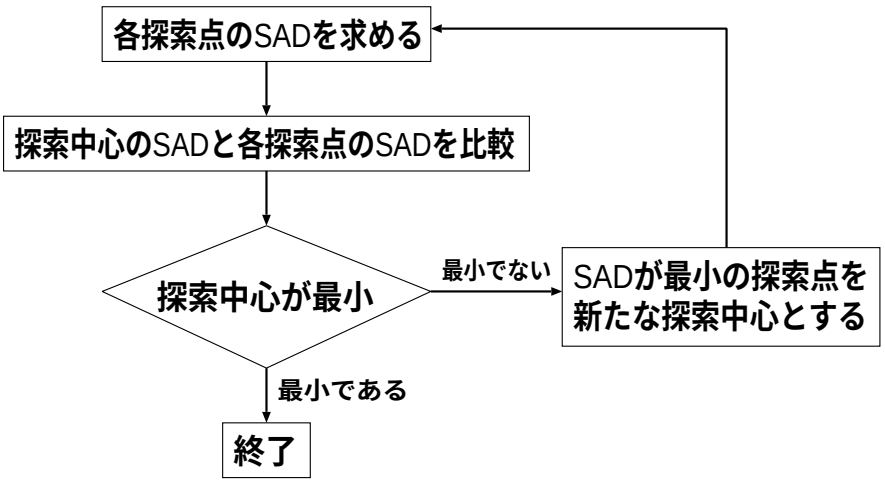

図 2.2: 追跡型の動き探索処理の流れ

スクエアサーチ，ダイヤモンドサーチ，ヘキサゴンサーチにおいて参照画像の画素値データの再利用を行う場合，各動き探索方法によって画素値データの探索点当たりのロード回数が異なる．探索点当たりのロード回数は，差分絶対値和の計算に使用する参照画像の画素値データ全てを探索点数で割ることによって求めることができる．探索点当たりのロード回数によりロードのオーバーヘッドが変わる．差分絶対値和の計算のために使用する参照画像データのロードが必要な動き探索処理では，探

索点当たりのロード回数によって処理速度に大きな影響を及ぼす．そのため，動き探索処理の速度を向上させるには，探索点当たりのロード回数を少なくする必要がある．

参照画像の画素値データの再利用では，2 回目以降の探索において差分絶対値和を計算するために必要となる参照画像の画素値データに前回の探索で使用したデータを使用するものである．前回の探索で使用した全てのデータを使用できるとは限らない．新たに必要となった参照画像の画素値データに関しては，読み込み処理を行わなければならない．しかし，再利用できる部分のデータに関しては，読み込み処理を行うことなく参照画像の画素値データを利用することができる．

各動き探索方法によって参照画像の画素値データを再利用できる部分が異なる．8x8 の場合について各動き探索での再利用できるデータの部分を示す．ダイヤモンドサーチについては，探索中心が右方向に移動した場合を図 2.3 に示す．ヘキサゴンサーチについては，探索中心が右下方向に移動した場合を図 2.4 に示す．スクエアサーチについては，探索中心が右下方向に移動した場合を図 2.5 に示す．最も色の濃いところが，データの再利用により使用することが可能な参照画像データである．

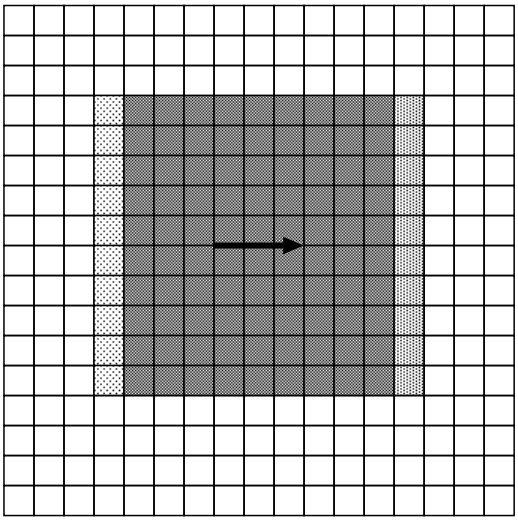

図 2.3: データ再利用可能部分 (ダイヤモンドサーチ)

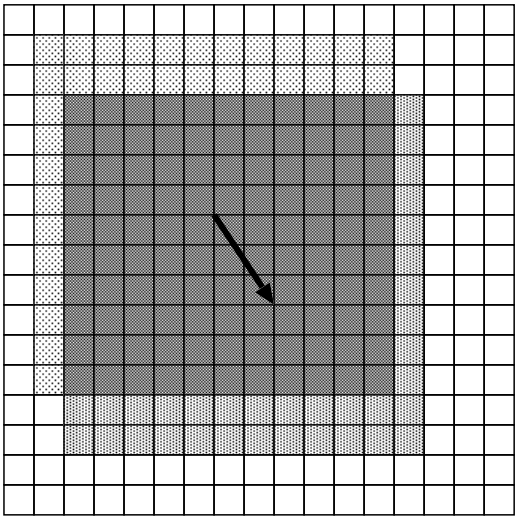

図 2.4: データ再利用可能部分 (ヘキサゴンサーチ)

2.3 問題点

動き探索処理で行われる差分絶対値和の演算の所要クロック数は，20x(ブロックの高さ)+43 サイクルである．ロードの所要クロック数は，19x(ブ

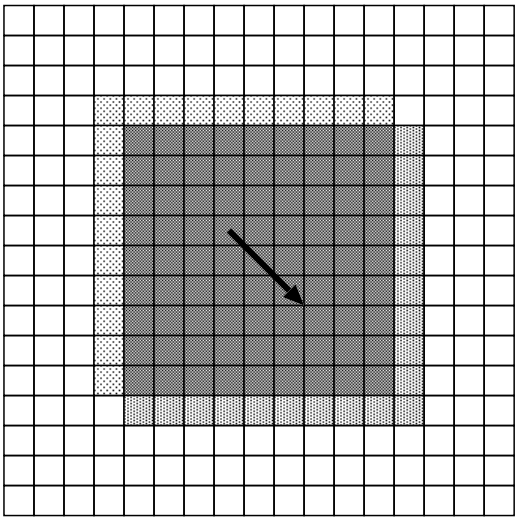

図 2.5: データ再利用可能部分 (スクエアサーチ)

ロックの高さ) である．これは，探索点 1 点に関する演算の所要クロック数とロードの所要クロック数である．各探索点でロードされる参照画像の画素値データには，同一データが存在する．しかし，探索点毎に参照画像の画素値データのロードを行うため，ロードのオーバーヘッドが大きくなる．

オープンソースの x264 ソフトウェアエンコーダの動き探索処理では，探索中心が移動する度に，毎回参照画像の画素値データを読み込んでから，差分絶対値和の計算を行っている．また，参照画像の画素値データには再利用の可能性があるにもかかわらず，データの再利用が行われていない．そのため，参照画像の画素値データを毎回読み込む分，ロード

のオーバーヘッドが大きくなっている．

3　Cell/B.E.とx86，GPGPUとの優位性比較

3.1　SIMD(Single Instruction Multiple Data)

SIMDとは，1つの命令で複数のデータに対して演算を同時に行う処理のことである．例えば，32ビットのデータ同士の演算が4回ある場合にSIMD演算を使用すると，1回の演算で全ての結果を求めることができる．通常の演算を図3.6に，SIMD演算を図3.7に示す．

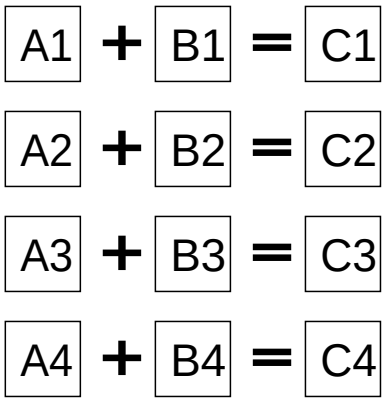

図 3.6: 通常の演算

Cell/B.E.，x86，GPUのいずれでもSIMDあるいはSIMDに似た演算機構が用いられており，これが処理の高速化を担っている．このSIMDは，動画像処理においての動き探索処理にも適しており，SIMDを活かすことが高速化を図る上で必須となる．

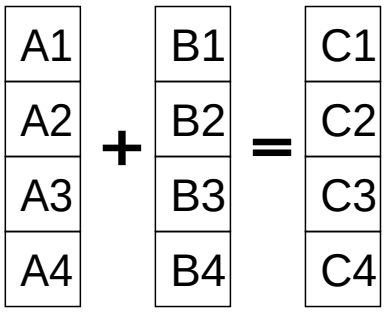

図 3.7: SIMD 演算

3.2 Cell Broadband Engine (Cell/B.E.)

Cell Broadband Engine(Cell/B.E.) は、非対称マルチコアプロセッサであり，制御系プロセッサコア (PPE) と演算系プロセッサコア (SPE) から構成される。また，Cell は高い演算性能を持っている．Cell/B.E. の構成を図 3.8 に示す.

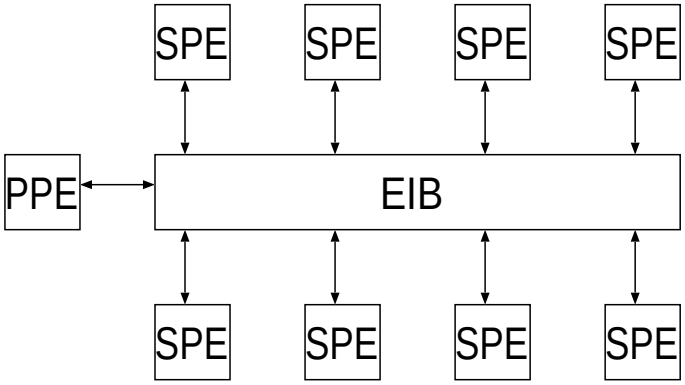

図 3.8: Cell Broadband Engine の構成

PPE(PowerPC Processor Element) は，64 ビット PowerPC アーキテク

チャと同等の性能を持つ汎用プロセッサであり，1 個搭載されている．PPE は、主に制御系の処理を行っており，オペレーティング・システムの実行や SPE を制御する役割などがある．

SPE(Synergistic Processor Element) は，SIMD 系アーキテクチャであり，SPE コアは 8 個搭載されている．SPE では，SIMD 演算を行うことが可能である．また，SPE は 128 ビットレジスタを 128 本持っており，単純な計算を繰り返すマルチメディア処理をより効率的に処理できる．さらに，SPE を複数個使用して並列処理を行うことで，より高い性能を発揮することができる．

EIB(Element Interconnect Bus) は，コア間をリング接続するネットワークである高速なバスで，PPE，SPE，メインメモリなどに接続されている．動き探索処理においても，EIB の接続構成を活かす SPE 間並列処理が必要である．

3.3 x86

x86 は，汎用プロセッサで，一般に最もよく使用されているプロセッサである．64 ビット化 (EM64T) に伴い，128 ビットレジスタである XMM レジスタが 16 本に拡張された．また，SSE4.1 では，参照画像の画素値

データの横方向の再利用が可能な mpsadbw 命令が追加された.

3.4 GPGPU

GPU(Graphics Processing Units) は，画像処理を専門に行う補助演算装置である．また，GPU は 512 ないしそれ以上の演算ユニットを持っており，可能な並列処理数が多いため高い並列処理能力がある.

GPGPU(General-Purpose computing on Graphics Processing Units) とは、GPU を画像処理以外の汎用の数値計算に使用することを目的とした技術のことである.

3.5 優位性比較

Cell/B.E. の SPE は，128 ビットレジスタを x86 プロセッサの 8 倍の 128 本持っている．このため，参照画像の画素値データを x86 プロセッサより多く持つことが可能である．x86 プロセッサの mpsadbw 命令を使用すると，横方向のデータの再利用が可能であるが，参照画像の画素値データが SAD の結果に変わってしまい，データの再利用を行うことができなくなってしまう.

H.264/AVC の動き探索処理を 1 次元の SIMD アレイで実行する場合，並列処理点数を大きく取れるスクエアサーチでも，4x3 点，8x3 点，16x3

点の 3 種で平均では高々24 並列程度である．GPU の 512 ないしそれ以上の並列処理を活かすことは困難である．

4 スクエアサーチとダイヤモンドサーチ，ヘキサゴンサーチの優位性比較

4.1 スクエアサーチ

スクエアサーチとは，探索中心と各探索点 (上下左右斜め方向の 8 点) との差分絶対値和の比較を行い，探索中心が最小となる位置を探索する方法である．

スクエアサーチの各探索点を図 4.9 に示す．太線で囲まれた所は探索中心であり，色の付いている所は探索点である．

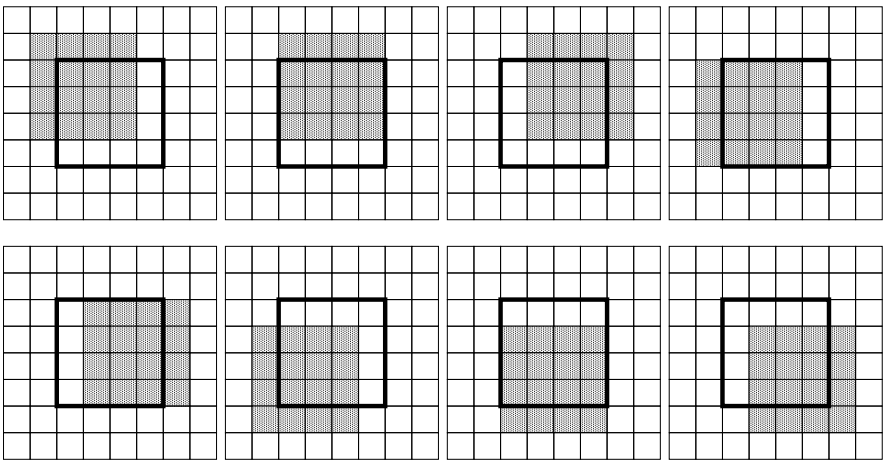

図 4.9: スクエアサーチの各探索点

スクエアサーチのアルゴリズムを以下に示す．

1. 探索中心と各探索点 (上下左右斜め方向の 8 点) との差分絶対値和の比較を行う.

2. 比較の結果，探索中心が最小となれば，探索は終了となる．探索中心が最小でないならば，3 へ進む.

3. 最小となった探索点を新たな探索中心として探索を行う．探索中心の移動により新たに増えた探索点とだけ差分絶対値和の比較を行う．移動方向が上下左右方向の場合は 3 点，斜め方向の場合は 5 点と比較を行う.

4. 2 へ戻る.

4.2 ダイヤモンドサーチ

ダイヤモンドサーチとは，探索中心と各探索点 (上下左右方向の 4 点) との差分絶対値和の比較を行い，探索中心が最小となる位置を探索する方法である.

ダイヤモンドサーチの各探索点を図 4.10 に示す．太線で囲まれた所は探索中心であり，色の付いている所は探索点である.

ダイヤモンドサーチのアルゴリズムを以下に示す.

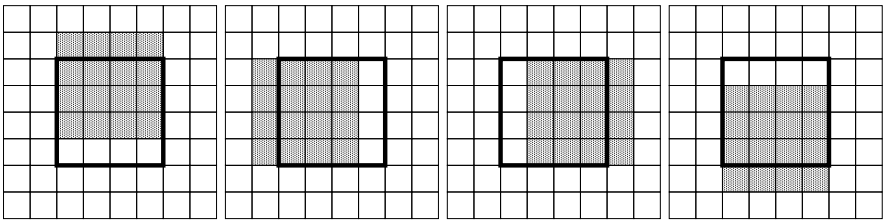

図 4.10: ダイヤモンドサーチの各探索点

1. 探索中心と各探索点 (上下左右方向の 4 点) との差分絶対値和の比較を行う.

2. 比較の結果，探索中心が最小となれば，探索は終了となる．探索中心が最小でないならば，3 へ進む.

3. 最小となった探索点を新たな探索中心として探索を行う．探索中心の移動により新たに増えた 3 点とだけ差分絶対値和の比較を行う.

4. 2 へ戻る.

4.3 ヘキサゴンサーチ

ヘキサゴンサーチとは，探索中心と各探索点 (6 点) との差分絶対値和の比較を行い，探索中心が最小となる位置を探索する方法である.

ヘキサゴンサーチの各探索点を図 4.11 に示す．太線で囲まれた所は探索中心であり，色の付いている所は探索点である.

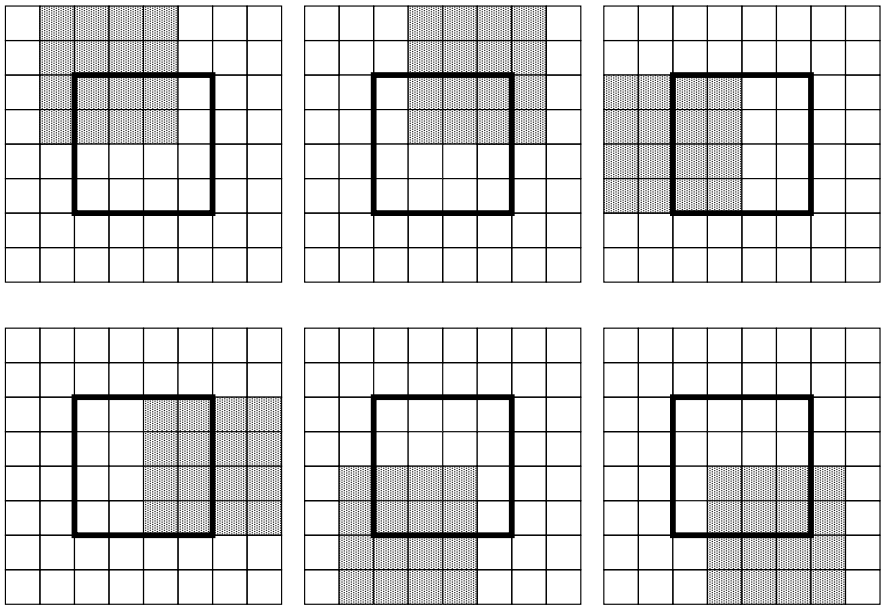

図 4.11: ヘキサゴンサーチの各探索点

ヘキサゴンサーチのアルゴリズムを以下に示す.

1. 探索中心と各探索点 (6 点) との差分絶対値和の比較を行う.

2. 比較の結果，探索中心が最小となれば，探索は終了となる．探索中心が最小でないならば，3 へ進む.

3. 最小となった探索点を新たな探索中心として探索を行う．探索中心の移動により新たに増えた 3 点とだけ差分絶対値和の比較を行う.

4. 2 へ戻る.

4.4 優位性比較

スクエアサーチは，ダイヤモンドサーチ，ヘキサゴンサーチに比べると差分絶対値和の計算を行う必要となる探索点数が多い．しかし，スクエアサーチは，探索点当たりの参照画像からのロード画素数がダイヤモンドサーチ，ヘキサゴンサーチより少ない．

各動き探索方法について探索点当たりの平均ロード画素数を 8x8 の場合について示す．ダイヤモンドサーチは表 4.1 に，ヘキサゴンサーチは表 4.2 に，スクエアサーチは表 4.3 に示す．

表 4.1: 探索点当たりの平均ロード画素数 (ダイヤモンドサーチ)

Ending Search Points	4	7	10	13	16
Average Loading Pixels	25	16	12	10	9

表 4.2: 探索点当たりの平均ロード画素数 (ヘキサゴンサーチ)

Ending Search Points	6	9	12	15	18
Average Loading Pixels	24	19	17	15	14

表 4.3: 探索点当たりの平均ロード画素数 (スクエアサーチ)

Ending Search Points	8	11	13	14	18
Average Loading Pixels	13	11	9	9	8

8x8 の場合について探索面積当たりの演算量と探索面積当たりの所要クロックサイクル数を表 4.4 に示す．探索面積は探索点 1 点当たりの面積に探索点数を掛けた値とする．探索面積で割った値で比較を行うとスクエアサーチが，ダイヤモンドサーチ，ヘキサゴンサーチより最も小さい値となる．

表 4.4: 探索面積当たりの演算量と所要クロックサイクル数

	Area	Operation	Clock Cycle
Square Search	512	566	2194
Diamond Search	256	358	1382
Hexagon Search	384	492	1902

5 提案手法

動き探索処理において，差分絶対値和の計算に使用される参照画像の画素値データを再利用する手法を提案する。この提案手法は，差分絶対値和の計算を行うための参照画像の画素値データの読み込み処理の回数を減らし，動き探索処理の高速化を図るものである．

1 回目は，参照画像の画素値データ全ての読み込み処理を行う必要がある．

2 回目以降は，探索中心の移動した方向により必要となる参照画像の画

素値データが変わり，新たに必要となった部分のみ参照画像の画素値データの読み込み処理を行う．移動方向が上下方向の場合には，新たに必要となる行データのみ読み込み処理を行う．移動方向が左右方向の場合には，新たに必要となる列データのみ読み込み処理を行う．移動方向が斜め方向の場合には，上下方向と左右方向の処理を組み合わせて新たに必要となるデータの読み込み処理を行う．

6　実装

6.1　スクエアサーチ

オープンソースの x264 ソフトウェアエンコーダには，動き探索方法としてダイヤモンドサーチ，ヘキサゴンサーチは実装されている．しかし，スクエアサーチについては実装されていない．そこで，新たにスクエアサーチを実装した．

1 回目の探索では，上下左右斜め方向の 8 点の差分絶対値和の計算を行う．その後，探索中心と 8 点との比較を行い，差分絶対値和が最小となる点を探す．探索中心が最小となれば，探索は終了である．探索中心が最小でなければ，最小となった探索点の差分絶対値和の値に探索中心の値を更新する．そして，再び探索を行う．

2回目以降の探索では，探索中心の移動方向により処理が変わる．上下左右方向の場合は，新たに増える探索点は3点である．新たに増えた3点の探索点のみ差分絶対値和の計算を行い，探索中心と比較を行う．探索中心が最小となれば，探索は終了である．探索中心が最小でなければ，最小となった探索点の差分絶対値和の値に探索中心の値を更新する．そして，再び探索を行う．斜め方向の場合は，新たに増える探索点は5点である．新たに増えた5点の探索点のみ差分絶対値和の計算を行い，探索中心と比較を行う．探索中心が最小となれば，探索は終了である．探索中心が最小でなければ，最小となった探索点の差分絶対値和の値に探索中心の値を更新する．そして，再び探索を行う．

プログラムコードを以下に示す．COST_MV_X8，COST_MV_X3，COST_MV_X5では，各探索点の差分絶対値和を求めている．COPY2_IF_LTでは，探索中心と探索点の差分絶対値和の値の比較を行い，小さいほうの値をbcostに格納する．bcostの値は探索中心の差分絶対値和の値である．bmx，bmyは探索中心の座標を表す．

```
COST_MV_X8( -1,0, -1,1, 0,1,  1,1, 1,0, 1,-1, 0,-1, -1,-1, costs );
COPY2_IF_LT( bcost, costs[0], dir, 0 );
COPY2_IF_LT( bcost, costs[1], dir, 1 );
```

```
COPY2_IF_LT( bcost, costs[2], dir, 2 );
COPY2_IF_LT( bcost, costs[3], dir, 3 );
COPY2_IF_LT( bcost, costs[4], dir, 4 );
COPY2_IF_LT( bcost, costs[5], dir, 5 );
COPY2_IF_LT( bcost, costs[6], dir, 6 );
COPY2_IF_LT( bcost, costs[7], dir, 7 );
if( dir != -2 )
{
   bmx += squ2[dir+1][0];
   bmy += squ2[dir+1][1];
   for( i = 1; i < i_me_range && CHECK_MVRANGE(bmx, bmy); i++ )
   {
        const int odir = mod8m1[dir+1];
        dir = -2;
        if( odir % 2 == 0 )
        {
           COST_MV_X3_DIR( squ2[odir+0][0], squ2[odir+0][1],
                           squ2[odir+1][0], squ2[odir+1][1],
```

```
                    squ2[odir+2][0], squ2[odir+2][1],
                    costs );
    COPY2_IF_LT( bcost, costs[0], dir, odir-1 );
    COPY2_IF_LT( bcost, costs[1], dir, odir   );
    COPY2_IF_LT( bcost, costs[2], dir, odir+1 );
}
else
{
    COST_MV_X5( squ2[odir-1][0], squ2[odir-1][1],
                squ2[odir+0][0], squ2[odir+0][1],
                squ2[odir+1][0], squ2[odir+1][1],
                squ2[odir+2][0], squ2[odir+2][1],
                squ2[odir+3][0], squ2[odir+3][1],
                costs );
    COPY2_IF_LT( bcost, costs[0], dir, odir-2 );
    COPY2_IF_LT( bcost, costs[1], dir, odir-1 );
    COPY2_IF_LT( bcost, costs[2], dir, odir   );
    COPY2_IF_LT( bcost, costs[3], dir, odir+1 );
```

```
            COPY2_IF_LT( bcost, costs[4], dir, odir+2 );
            }
          if( dir == -2 )
             break;
          bmx += squ2[dir+1][0];
          bmy += squ2[dir+1][1];
    }
}
```

6.2 データの再利用

探索点を左側，中央，右側の 3 つの部分に分け，左側の探索点用，中央の探索点用，右側の探索点用の参照画像の画素値データを格納する 3 つの変数 (left，center，right) を用意する．差分絶対値和の計算に使用される参照画像の画素値データの内，どの部分のデータを 3 つの変数 (left，center，right) に格納するかを示したものを図 6.12 に示す．色の濃い部分は探索中心である．

変数宣言を以下に示す．

```
vec_u8_t left[bh+2];
```

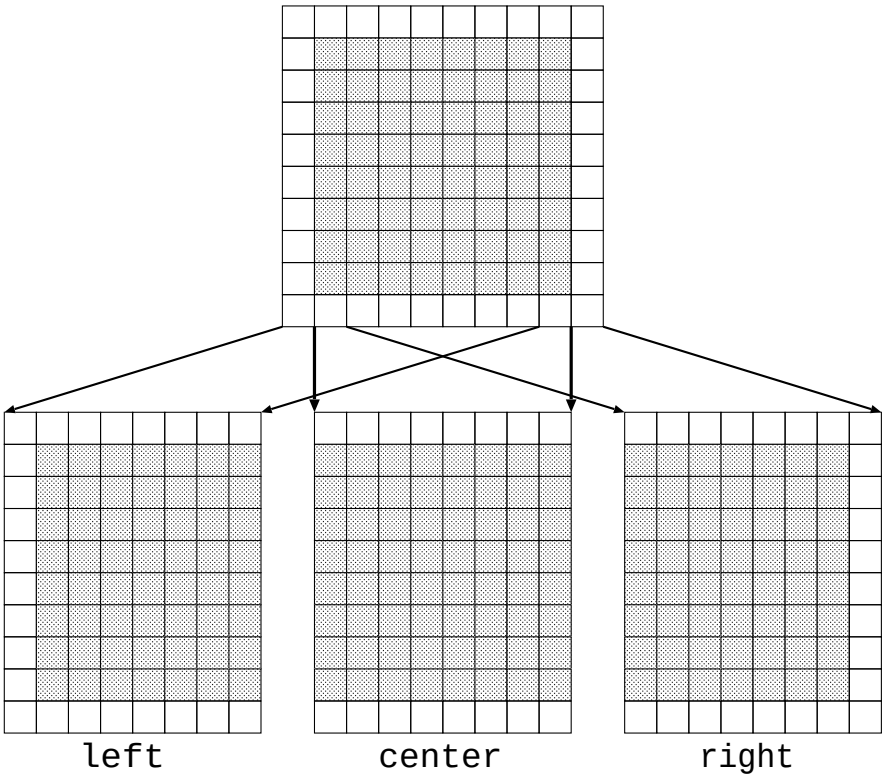

図 6.12: 各変数の参照画像の画素値データ格納部分

```
vec_u8_t center[bh+2];

vec_u8_t right[bh+2];
```

1 回目の探索では，参照画像の画素値データを全て読み込み，変数 left，center，right に格納する．左側の探索点の差分絶対値和の計算に使用する参照画像の画素値データは，変数 left に格納する．中央の探索点の差分絶対値和の計算に使用する参照画像の画素値データは，変数 center に格納する．右側の探索点の差分絶対値和の計算に使用する参照画像の画素

値データは，変数 right に格納する．

2 回目以降の探索では移動方向により参照画像の画素値データを格納する変数が変わる．移動方向が左方向の場合は，変数 right のデータが不要になるので，変数 right に新たに必要となった参照画像の画素値データを格納する．移動方向が右方向の場合は，変数 left のデータが不要になるので，変数 left に新たに必要となった参照画像の画素値データを格納する．移動方向が上方向の場合は，各変数の不要となったデータの所に新たに必要となった参照画像の画素値データを格納する．移動方向が下方向の場合は，各変数の不要となったデータの所に新たに必要となった参照画像の画素値データを格納する．移動方向が斜め方向の場合は，上下左右方向のそれぞれの処理を組み合わせて処理を行う．

プログラムコードを以下に示す．VEC_LOAD は参照画像の画素値データを変数に格納する．pix7，pix6，pix5 は参照画像の画素値データを取って来る先頭を示す．移動方向が左方向の場合は pixv が right，移動方向が右方向の場合は pixv が left となる．pix は参照画像の画素値データを取って来る先頭を示す．移動方向が上方向または下方向の場合は，y を不要データの位置にする．

1 回目の探索：

```
for( y = 0; y < i_height+2; y++ )
{
    VEC_LOAD( pix7, left[y], i_width, vec_u8_t );
    pix7 += i_stride;
    VEC_LOAD( pix6, center[y], i_width, vec_u8_t );
    pix6 += i_stride;
    VEC_LOAD( pix5, right[y], i_width, vec_u8_t );
    pix5 += i_stride;
}
```

2回目以降の探索：

移動方向が左方向または右方向の場合：

```
for( y = 0; y < i_height+2; y++ )
{
    VEC_LOAD( pix, pixv[y], i_width, vec_u8_t );
    pix += i_stride;
}
```

移動方向が上方向または下方向の場合：

```
VEC_LOAD( pix, pixv[y], i_width, vec_u8_t );
```

7　性能評価

評価動画には，Whale，Ice hockey を使用した．フレーム数は，200 フレームである．動き探索方法には，ダイヤモンドサーチ，ヘキサゴンサーチ，スクエアサーチを使用した．スクエアサーチについては，データの再利用を行わないものとデータの再利用を行うものを使用した．各動き探索方法についてエンコード時の探索処理時間と PSNR 値を示す．Whale については表 7.5 に，Ice Hockey については表 7.6 に示す．

表 7.5: 探索処理時間と PSNR 値 (Whale)

	Diamond	Hexagon	Square	Square (reuse)
Time(sec)	42.573	44.466	44.090	43.639
psnr(dB)	26.955	27.029	26.960	26.963

表 7.6: 探索処理時間と PSNR 値 (Ice Hockey)

	Diamond	Hexagon	Square	Square (reuse)
Time(sec)	34.326	35.612	35.405	34.480
psnr(dB)	37.201	37.248	37.210	37.209

8　おわりに

参照画像の画素値データの再利用により，動き探索処理の時間を短縮することができた．これにより動き探索処理の速度を向上させることが

でき，参照画像の画素値データの再利用により，動き探索処理の高速化が可能であることを示した．

今回実装した参照画像の画素値データを再利用するプログラムには，データ再利用処理部分に条件分岐が存在する．また，処理関数のマクロ化を行っていない部分も存在する．そのため，条件分岐の削減や関数のマクロ化により，動き探索処理の更なる速度の向上が可能であり，これにより更なる高速化が期待できる．

今後の課題としては，データ再利用処理部分の条件分岐の削減や関数のマクロ化を行うことが必要である．そして，動き探索処理の更なる高速化を図ることも必要である．

謝辞

本研究を行うに当たり日頃ご指導頂きました近藤利夫教授，大野和彦講師，佐々木敬泰助教に深く感謝致します．また，日頃お世話になりました計算機アーキテクチャ研究室の皆様に感謝致します．

参考文献

[1] 黒澤泰彦，渡辺幸男，田胡治之,「次世代プロセッサ Cell Broadband Engine」，東芝レビュー，Vol.6 No.6，pp9-15，2006

[2] 大久保榮 (監修), 角野眞也, 菊池義浩, 鈴木輝彦,「改訂版 H.264/AVC 教科書」, 2006