平成 13 年度

学士学位論文

ピクセルベース動画圧縮方式と そのデータ駆動型実現法

Pixel Based Compression Scheme and Its Data-Driven Implementation

1020309　　中村 勲二

指導教員　　岩田 誠

2002 年 2 月 8 日

高知工科大学 情報システム工学科

要 旨

ピクセルベース動画圧縮方式とそのデータ駆動型実現法

中村 勲二

近年、インターネット・モバイルネットワークの高速化・大容量化に伴って、リアルタイム性を重視した映像通信サービスの需要が高まっている。しかし、複雑な処理を要する映像圧縮・伸張処理がボトルネックとなっており、処理の高速化が急がれている。

標準化されている MPEG[4][5] 等の映像圧縮方式は、汎用性を重視し周波数変換や動き補償のような特定の用途には不要な負荷の高い処理を伴うため高速化が難しい。

本研究では、リアルタイムインターネット授業など動きの小さい用途に特化したピクセルベース動画圧縮方式 PBC を提案する。本方式は、BTC（Block Truncation Coding : ブロック切り詰め符号化）[2] をベースとした、画素データから直接圧縮符号を生成するアルゴリズムを持つため、処理負荷が低く高速化が可能である。また、フレーム間の差分が少ない部分、すなわち動きの少ない部分の BTC 符号を切り捨てることで、高い圧縮率を達成できる。

PBC を実現するプロセッサとしては、超高速・省電力のデータ駆動型メディアプロセッサ DDMP[1] が有望である。DDMP は、従来のノイマン型プロセッサとは異なる独自のアーキテクチャを持ち、取り分け映像処理においてその真価が発揮される。

本稿では、DDMP への PBC の実装手法を提示すると共に、シミュレーションによりその性能を評価した。結果、DDMP による高速な PBC の実現が可能なことを証明できた。

キーワード　データ駆動, BTC, ピクセルベース動画圧縮方式, 2 世代化

Abstract

Pixel Based Compression Scheme and Its Data-Driven Implementation

Kunji NAKAMURA

In recent years, the demand of real-time video communication services via high speed networks is increasing. Improvement in the speed of video compression/extension processes which are the bottle-necks are urged. Standard video compression schemes, such as MPEG[4][5], need complicated processes such as DCT and motion compensation for keeping video quality, so they are difficult to enable high speed processing.

In this research, I propose PBC (Pixel Based Compression) based on BTC (Block Truncation Coding)[2] which is a video compression scheme specialized for small motion video, such as real-time Internet Classes. BTC reduces the load by direct encoding from pixel data. Moreover, PBC enables the high rate of compression by throwing away BTC codes when differences between frames are small.

DDMP (Data-Driven Media Processor)[1] of high-speed and power saving is promising to implement PBC. DDMP has the unique architecture different from the conventional Von Neumann type processor. The architecture has the high affinity especially with video processing.

In this paper, I show an effective data-driven implementation of PBC and evaluate the performance by simulation. As the result, I proved that the high-speed implementation of PBC by DDMP was possible.

key words Data-Driven, BTC, Pixel Based Compression Scheme, Pair-Packet

目次

第 1 章　序論 **1**

第 2 章　既存の動画圧縮方式 **5**

2.1　周波数変換ベースの動画圧縮方式 5

2.1.1　MPEG 5

2.1.2　H261、H263 6

2.1.3　Motion-JPEG 7

2.1.4　周波数変換ベース動画圧縮方式の特徴 7

2.1.5　問題点 8

2.2　BTC ベースの動画圧縮方式 10

2.2.1　Smart Compression 11

2.2.2　圧縮率見積もり 13

2.2.3　問題点 14

第 3 章　ピクセルベース動画圧縮方式 **15**

3.1　Pixel Based Compression : PBC 15

3.2　PBC の処理内容 16

3.2.1　RGB-YCrCb 変換処理 17

3.2.2　BTC 18

3.2.3　Class Specify（キーフレーム用） 18

3.2.4　Class Specify（デルタフレーム用） 19

3.3　PBC の圧縮率 20

第 4 章　データ駆動型 PBC の実現法 **22**

4.1　DDMP 22

4.1.1 DDMP の基本アーキテクチャ 22
4.1.2 DDMP と映像処理の親和性 23
4.2 データ駆動型 PBC 実現法の要件 26
4.2.1 RGB → YCrCb 27
4.2.2 並列 BTC 27
4.2.3 Class Specify Key 29
4.2.4 Class Specify Delta 29
4.3 DDMP4G 向け PBC 最適化手法 30
4.3.1 データ駆動型 PBC の DDMP4G への実装方法 31
4.3.2 2 世代化 32
4.3.3 2 世代化特化命令セット 33
4.3.4 連続メモリ参照命令セット 35

第 5 章 性能評価 **37**

5.1 DDMP4G チップへの BTC 実装実験 37
5.2 シミュレーションによる性能評価実験 37
5.2.1 シミュレータの原理 37
5.2.2 シミュレーション手法 38
5.2.3 動画圧縮アルゴリズムの性能評価 39
5.2.4 提案命令セット追加による PBC の性能比較 39

第 6 章 結論 **41**

謝辞 **44**

参考文献 **46**

図目次

2.1 周波数変換ベース動画圧縮方式 7

2.2 BTC 基本原理 10

2.3 BTC の画質 11

2.4 Smart Compression 12

2.5 多層 BTC 13

2.6 ブロックのクラス 13

3.1 Pixel Based Compression 16

3.2 PBC における時間的冗長部分削減法 17

3.3 差分値 1 19

4.1 DDMP のナノプロセッサ構成図 22

4.2 DDMP の並列概念図 23

4.3 映像処理における世代識別子の割り当て方 24

4.4 メモリアクセスの方法 24

4.5 データ駆動型 PBC 26

4.6 並列 BTC 28

4.7 デルタフレーム用クラス判定処理 30

4.8 DDMP4G 31

4.9 2 世代化 33

4.10 2 世代化特化命令セット（基本演算用） 34

4.11 2 世代化特化命令セット（メモリアクセス用） 35

4.12 連続メモリ参照命令セット 36

表目次

2.1 周波数変換ベース動画圧縮方式の圧縮率 8
2.2 SC の圧縮率見積もり 14

3.1 クラス分類条件とブロックの圧縮率 19
3.2 クラス分類条件とブロックの圧縮率（デルタフレーム） 20
3.3 PBC の圧縮率 21

5.1 性能評価 39
5.2 性能評価（PBC） 40

6.1 インターネット授業用動画圧縮方式の要件 42
6.2 PBC の圧縮率（今後の展望） 43

第 1 章

序論

近年、商用ネットワークは著しい発達を遂げた。大都市における有線ネットワークでは毎秒ギガビット単位の大容量・高速通信が可能となり、モバイル通信においては W-CDMA 方式により毎秒メガ単位のデータ通信が可能となった。そして現在、その高速ネットワークを利用した、

- テレビ電話
- 各種監視システム
- 過疎地域の教育環境向上を目指したインターネット授業
- 視聴者が参加可能な TV 番組

など、リアルタイム性を重視した各種映像通信サービスの需要が高まっている。しかし、映像通信において複雑な処理を要する映像の圧縮・伸張処理がボトルネック、すなわち遅延の原因となっており、処理の高速化が急務となる。

現在標準化されている映像圧縮方式の中で特に代表的なものとして MPEG[4][5] が挙げられる。MPEG は現在、MPEG1、MPEG2、MPEG4 が標準化されており、MPEG7 の標準化が進行中である。中でも、前述したネットワークによる映像通信を目的として定められた方式が MPEG4 であり、同方式を利用した映像通信サービスも幾つか商標化されている。

MPEG の基本的な符号化アルゴリズムはすべて、動き補償や DCT 等に基づいている。これらの処理により、重要な画像情報の損失なく圧縮率を向上可能な一方、負荷が高く処理の高速化が難しいという問題点を持つ。更に、標準化されているためネットワーク上で盗聴されれば容易に復号可能であるという欠点を伴う。そのため、有料配信を目的とした映像

や、プライバシ重視の映像を MPEG 形式で送信する場合、映像データへの暗号化処理を要し、それが更に高速化を困難なものとしている。

そこで本研究では、標準化を目的としない高速な映像圧縮アルゴリズムの構築を目標として、BTC（Block Truncation Coding : ブロック切り詰め符号化）[2] に着目した。BTC は 1979 年に米パーデュ大学の O. Robert Mitchell と Edward J. Delp らによって提案されたグレイスケールの静止画向けの圧縮方式である。MPEG のベースとなる静止画圧縮方式 JPEG[5][6] は、画像を周波数変換し視覚的に重要な情報を低周波領域に集め、量子化により不要な高周波成分を切り捨てて圧縮符号を生成するという複雑なアルゴリズムを要する。それに対して BTC は、画像を 4 × 4 のマクロブロックに分割し、ブロックデータから直接圧縮符号を生成するアルゴリズムを持つため処理負荷が低い。よって、BTC ベースの動画圧縮方式はリアルタイム性を重視する用途に有効である。

本稿では、BTC をフルカラーに拡張し、低負荷なアルゴリズムで高圧縮を可能としたピクセルベース動画圧縮方式 PBC（Pixel Based Compression）を提案する。本方式は、BTC 同様 4 × 4 マクロブロックを基本処理単位とし、フレーム間で BTC 符号の差分を取り、差分が少ないブロック、すなわち動きの少ないブロックの符号を切り捨てる処理を付加することで圧縮率の向上を可能とした。付加した処理は少ない命令での実装が可能な低負荷な処理であるため、BTC とほぼ同様の処理性能が見込める。

この方式を実現するプロセッサとしては、超高速・省電力の DDMP（Data-Driven Media Processor : データ駆動型メディアプロセッサ）[1] が有望である。

映像圧縮・伸張処理に用途を特化した ASIC（Application Specific Integrated Circuit : 特定用途向け集積回路）による処理の高速化・省電力化は可能で、既に MPEG4 向きの ASIC を搭載した機器も商用化されている。しかし、新しい技術、方式、サービスが次々と開発される現代においては、用途を特化した ASIC では設計・開発コストが高すぎる。サービス内容に応じて処理を拡張・変更する場合、ハードウェアを設計し直す必要があるからである。

それに対して DDMP は、従来のノイマン型プロセッサとは異なる独自のアーキテクチャを採用することで高速処理・低消費電力を実現している。ノイマン型プロセッサがプログラムカウンタにより逐次的に命令を取り出して実行するのに対し、DDMP は行き先情報、世代識別子等を含むパケット形式のデータを取り扱う。行き先情報はプロセッサ内のルータと照合してデータの行き先を決定するのに使用される。これにより、データは独立してプロセッサ内を巡回することが可能となる。世代識別子は、フィールド・ライン・ピクセルの 3 次元で構成され、同じ識別子を持つデータ同士に対してのみ処理が実行されるという特性を持つ。これらの識別子により、各データに独立性が与えられ複数の処理をパイプライン並列に実行可能となり、取り分け映像処理においてその真価が発揮される。現在開発されている DDMP4G は、8600MOPS の処理性能を実現している。

またプログラム可能なため、処理内容の拡張・変更がソフトウェアにより対応できる。例えば、サービス内容に応じて特定用途に有効な PBC と汎用的な MPEG を使い分けることで、各種映像通信サービスへの柔軟な対応が可能となる。

第 2 章では、現在提案されている動画圧縮方式を、周波数変換ベース、BTC ベースに分類して取り上げる。周波数変換ベースの方式は、用途に応じて高画質・高圧縮率が可能であるが、膨大な量の演算を必要とし、処理の高速化が困難である。BTC ベースの方式としては、現在 Smart Compression[3] という方式が提案されているが、これは圧縮率の向上のため高負荷な処理を伴う。

第 3 章では、第 2 章で触れた問題点を踏まえた上で、簡易な処理の付加により高い圧縮率を実現可能な PBC を提案すると共に、この方式の基本的なアルゴリズムを述べる。

第 4 章では、データ駆動方式向けの PBC の並列アルゴリズムを提案すると共に、これを DDMP4G 向けに最適化する手法として、以下を検討する。

- メモリによる負荷制御法
- マルチプロセッサ上で効果的なパイプライン処理を実現するための、処理分散法
- 隣接する画素を同時に演算する 2 世代化とそれに特化した命令セット

- メモリアクセス回数の極小化を目的とした連続メモリ参照命令セット

第 5 章では、DDMP4G シミュレータにより、第 4 章で提案したデータ駆動型 PBC アルゴリズム及び命令セットの性能評価結果を示し、その有効性を明らかにする。データ駆動型 PBC アルゴリズムの評価については、他の圧縮方式との性能比較を行う。命令セットの評価については、PBC を以下の 4 パターン実装しその性能を比較する。

- DDMP4G 既存命令（従来方式）
- DDMP4G 既存命令（2 世代化）
- DDMP4G 既存命令＋ 2 世代化特化命令（2 世代化）
- DDMP4G 既存命令＋ 2 世代化特化命令＋連続メモリ参照命令（2 世代化）

第 6 章では、第 5 章までで得られた結果をもとに PBC の実用性、残された課題、今後の展望に触れる。

第 2 章

既存の動画圧縮方式

2.1 周波数変換ベースの動画圧縮方式

動画圧縮方式の国際標準である H.261、H263、MPEG 等はすべて、前後の画面の違いを検出する動き補償、画像情報を周波数情報に変換する DCT（Discrete Cosine Transform : 離散コサイン変換)、不要な高周波成分の情報を切り捨てる量子化、生起確率の高い符号に短い符号を割り当てる可変長符号化という 4 つのアルゴリズム要素に基づいている。圧縮率の目安としては、DCT と量子化で 1/10〜1/20、動き補償で 1/2、可変長符号化で 2/3〜1/2 の圧縮が可能である。したがって、総合的に 1/30〜1/80 程度の圧縮が可能である。本節では、これらの動画圧縮方式のうち、MPEG、H.261、H263、Motion-JPEG について取り上げ、その応用分野、特徴、問題点を述べる。

2.1.1 MPEG

MPEG とは、ISO/IEC（国際標準化機構/国際電気標準会議）の動画像音声符号化グループ（Moving Picture coding Experts Group）で、動画・音声のデジタル圧縮方式の標準化を目的として 1988 年に設立された。同グループによって標準化されている動画像圧縮符号化方式を指す場合もこの略称が用いられる。現在までに MPEG1、MPEG2、MPEG4 を標準化し、MPEG7 の標準化を進行中である。

MPEG1 は、CD-ROM や DAT（Digital audio tape）などの蓄積メディアを対象とした圧縮方式で、転送速度は 1.5Mbps 程度、画質は VHS のビデオ並みである。1992 年に ISO の正式規格として採用された。

MPEG2 は、放送分野への応用を目的として開発された符号化方式であり、再生時に動画と音声合わせて 4〜15Mbps 程度のデータ転送速度を要する。1994 年に標準化された。MPEG2 の適用対象としては、次世代のテレビ放送 HDTV や広帯域 ISDN を利用した高画質の映像転送、および高画質デジタル映像メディアの DVD などがある。欧米ではすでに次世代デジタルテレビの標準規格として採用が決まっており、日本では 1996 年にサービスを開始したデジタル CS（Communication Satellite：通信衛星）で用いられている。

MPEG4 は、携帯電話、PHS、電話回線などの通信速度の低い回線による低画質・高圧縮率の映像の配信を目的とした規格であり、動画と音声合わせて 64kbps 程度のデータ転送速度を目指している。MEPG4 の最も大きな特徴は、多様なアプリケーションをサポートするために、様々な要素技術を組み合わせたプロファイルとレベルを複数提供している点である。シンプルプロファイルにあたる Version1 が 1999 年 3 月、Version1 の上位互換規格であり高機能なプロファイルをサポートしている Version2 が 2001 年 3 月に正式に確定した。

2.1.2　H261、H263

H261 は、テレビ電話/テレビ会議などの通信を対象とした動画圧縮方式で、転送速度は 64kbps から 2Mbps に限定されている。画像フォーマットは通常は QCIF（Quarter Common Intermediate Format : 176 画素× 144 画素)、オプションで CIF(Common Intermediate Format : 352 × 288 画素）を扱い、フレームレートは最大 30fps となっている。また、この方式は MPEG の基本ともなっている。1990 年に ITU-T（International Telecommunication Union Telecommunication Standardization sector）によって標準化された。

H263 は、H261 の発展型として作られたもので、H261 と同様に低ビットレートでのテレビ会議等の動画像伝送を目的としている。H263 は H261 と比較して、30〜50%程度の情報量で同等の品質を実現可能で、sub-QCIF・4CIF・16CIF 等、H261 より多彩な画像フォーマットをサポートしている。

2.1.3　Motion-JPEG

Motion-JPEG は、静止画フォーマットである JPEG を高速に伸長処理して連続再生することにより動画のように見せる圧縮方式である。前後のフレームとの差分情報を利用しないため、MPEG 等と比較して圧縮率は低いが、圧縮・伸張処理が低負荷で、比較的簡単な設備でリアルタイム圧縮が可能なため、個人向けのビデオキャプチャカード等の出力形式に採用されている。

2.1.4　周波数変換ベース動画圧縮方式の特徴

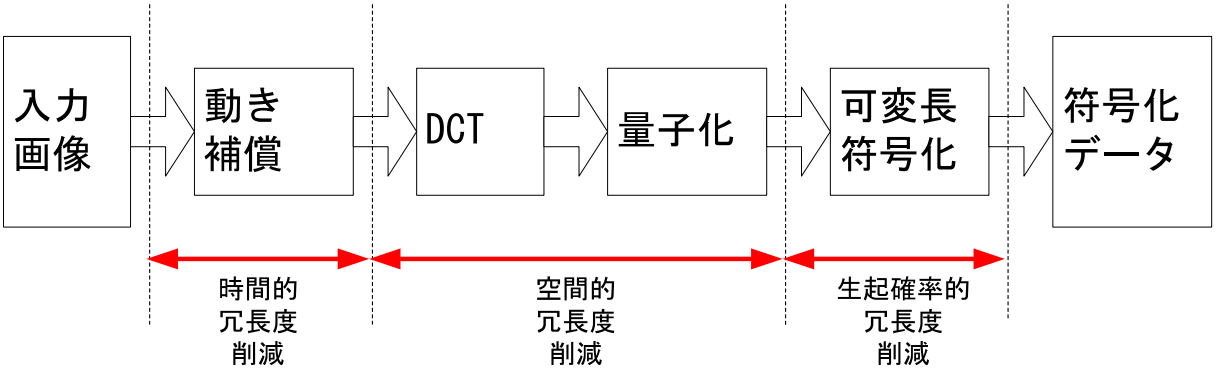


図 2.1　周波数変換ベース動画圧縮方式

図 2.1 に周波数変換ベース動画圧縮方式の基本的なプロセスを示す。動き補償においては、連続する 2 つのフレーム間における物体の動きの方向と大きさを表す動きベクトルを検出し、動きベクトルで示される 1 フレーム前の物体を予測値とし、その予測誤差を求める。これにより、時間方向に対する冗長度を削減する。DCT では、動き補償で得られた予測誤差を周波数成分に変換し、低周波成分に重要な画像情報を集める。DCT で抽出された空間的な冗長部分は、量子化により削減される。量子化では、画像情報の性質に基づいて低周波成分は細かく、高周波成分は粗く切り捨てる。可変長符号化では、量子化されたデータの生起確率に応じて、生起確率の高いデータには短い符号を、低いデータには長い符号を割り当てることで圧縮率を向上する。このように、人間の視覚的に不要な画像情報を抽出し削減す

ることで高画質を保持し、あらゆる用途への応用を可能としている。

表 2.1 に本節で取り上げた方式の圧縮率を示す。画質重視の MPEG2 や処理速度重視の Motion-JPEG では圧縮率を下げず、圧縮率重視の MPEG4 や H261 では画像情報を多く削減するというように用途に応じて圧縮率と画質の調整が可能である。

表 2.1　周波数変換ベース動画圧縮方式の圧縮率

	MPEG2	MPEG4	H261	Motion-JPEG
圧縮率	1/30〜1/40	サービス内容による	1/9〜1/80	1/15〜1/40

また、量子化や可変長符号化については、

1. 汎用的な量子化テーブル・可変長符号化テーブルを使用する。
2. 圧縮する画像データに応じて専用の量子化テーブル・可変長符号化テーブルを作成する。

という 2 つの方法があり、1 の方法は処理の高速化が可能な反面圧縮率・画質を高められないという欠点を持ち、2 の方法は圧縮率・画質を高められるが処理の高速化が困難であるという欠点を持つ。

これらを用途に応じて使い分けることでより有効な動画圧縮が可能となる。

2.1.5　問題点

周波数変換ベースの方式は、用途に応じて画質・圧縮率を高められる汎用的な符号化方式であるが、動き補償や DCT は多くの演算処理を必要としているため高速化が困難である。量子化テーブル・可変長符号化テーブルを画像データに応じて作成する場合は処理負荷が更に高くなる。

また、標準化されているためネットワーク上で盗聴されれば容易に復号可能であるという欠点がある。したがって、有料配信を目的とした映像、プライバシ重視の映像を送信する場合は映像データへの暗号化処理が必要となり、それが更に高速化を困難なものとしている。

現在、処理の高速化は ASIC によってのみ実現可能であるが、ASIC の場合、各種サービス内容に応じて別々にハードウェアを設計する必要がある。

DDMP4G による圧縮処理の実現については、ハードウェアがまだ発展途上にあるため、DCT 及び MPEG4-Version2 で規定されている SA-DCT（Shape Adaptive DCT：形状適応 DCT）のデータ駆動方式による実現法 [8] の確立に留まっている。

2.2　BTC ベースの動画圧縮方式

BTC は、1979 年に米パーデュ大学の Mitchell と Delp らにより発表されたグレイスケールの静止画に特化した画像圧縮方式である。図 2.2 に BTC の基本符号化原理を示す。

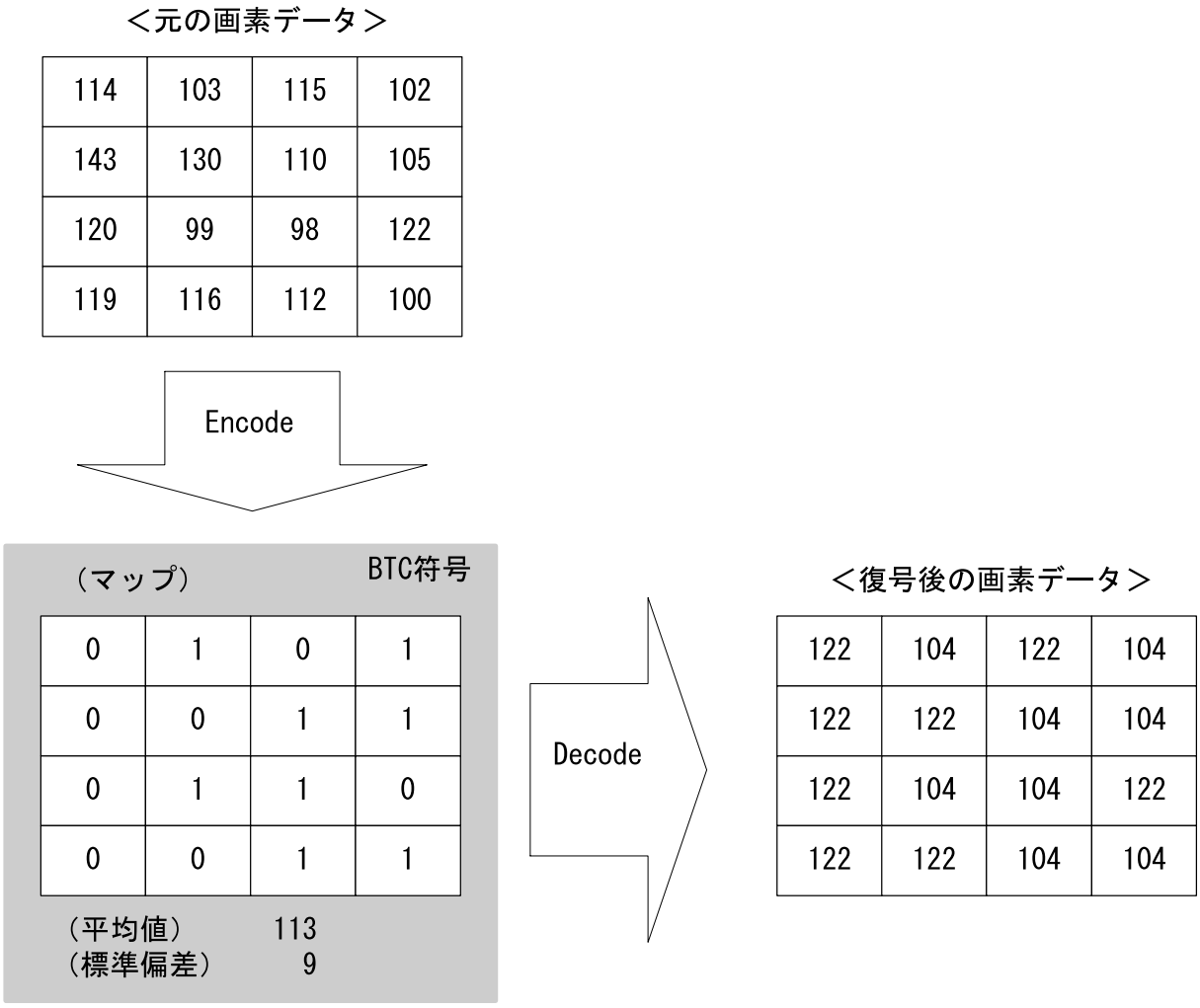


図 2.2　BTC 基本原理

まず、画像を 4 × 4 のマクロブロックに分割し、各ブロック毎に、

- 平均値
- 画素の濃度を 0 と 1 で表現したマップ
- 画素の強弱差の平均である標準偏差

を算出する。

復号する時には、マップの値に基づいて次のように各画素に値を割り当てる。

- マップが 0 →平均値＋標準偏差
- マップが 1 →平均値－標準偏差

この方式は、画素データから直接圧縮符号を生成するアルゴリズムを持つため、処理負荷が高い DCT や量子化をベースとした JPEG 等の静止画圧縮方式と比較して、高速な処理が可能である。また、不可逆な方式であるが、図 2.3 に示すように実用的な画質の保持を可能としている。

図 2.3　BTC の画質

よって、BTC ベースの動画圧縮方式は次のような用途への応用が可能である。

- 高い画質を必要としない用途、すなわち画像の鑑賞でなく、画像からの情報取得を目的とした用途
- リアルタイム性を重視する用途、すなわち高速な処理を必要とする用途
- 小型機器に圧縮処理機能等を実装する場合、又はそれを必要とする用途

2.2.1　Smart Compression

Smart Compression（以下 SC）は、処理能力の低い ASIC による高速処理を目指し、動きの少ない用途に特化して動き補償等を省いた BTC ベースの静止画・動画圧縮方式である。応用分野は限定されており標準化はされていない。

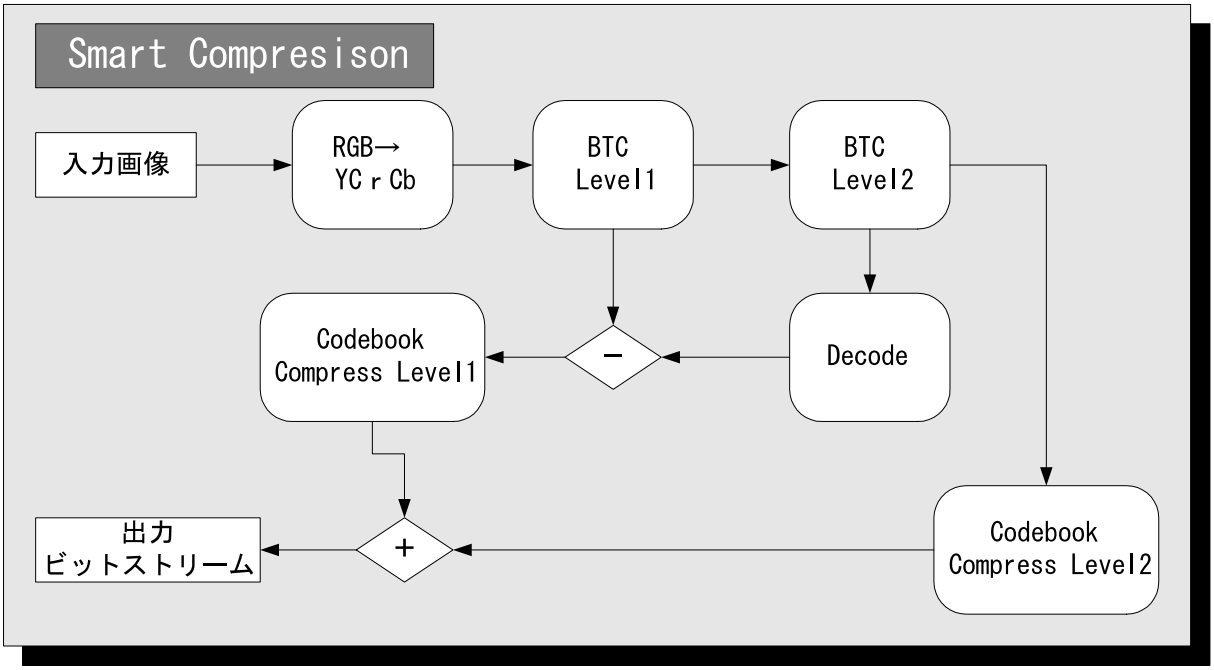


図 2.4　Smart Compression

前述したように BTC はグレイスケール専用の方式であるが、SC においてはフルカラーの画像を扱えるように処理内容を拡張している。

SC では、圧縮率を上げるために 16 × 16 のマクロブロックを基本処理単位とし、BTC 処理を 2 層に分けて行う多層 BTC（Multi-Level BTC）アルゴリズムを用いる。

図 2.5 に多層 BTC の基本的な処理内容を示す。まず、レベル 1BTC で得られた平均値 4 × 4 個をレベル 2 ブロックとして BTC 符号化し、レベル 2 ブロックを一度復号して復号後の画素データとレベル 1BTC 平均値との差分を取る。これにより、レベル 1BTC の平均値を 0 へ漸近し圧縮効果を上げることが可能となる。

更に、SC では可変長符号化も取り入れているが、可変長符号化テーブルは通常処理負荷の軽減を重視し汎用的なものを用いる。

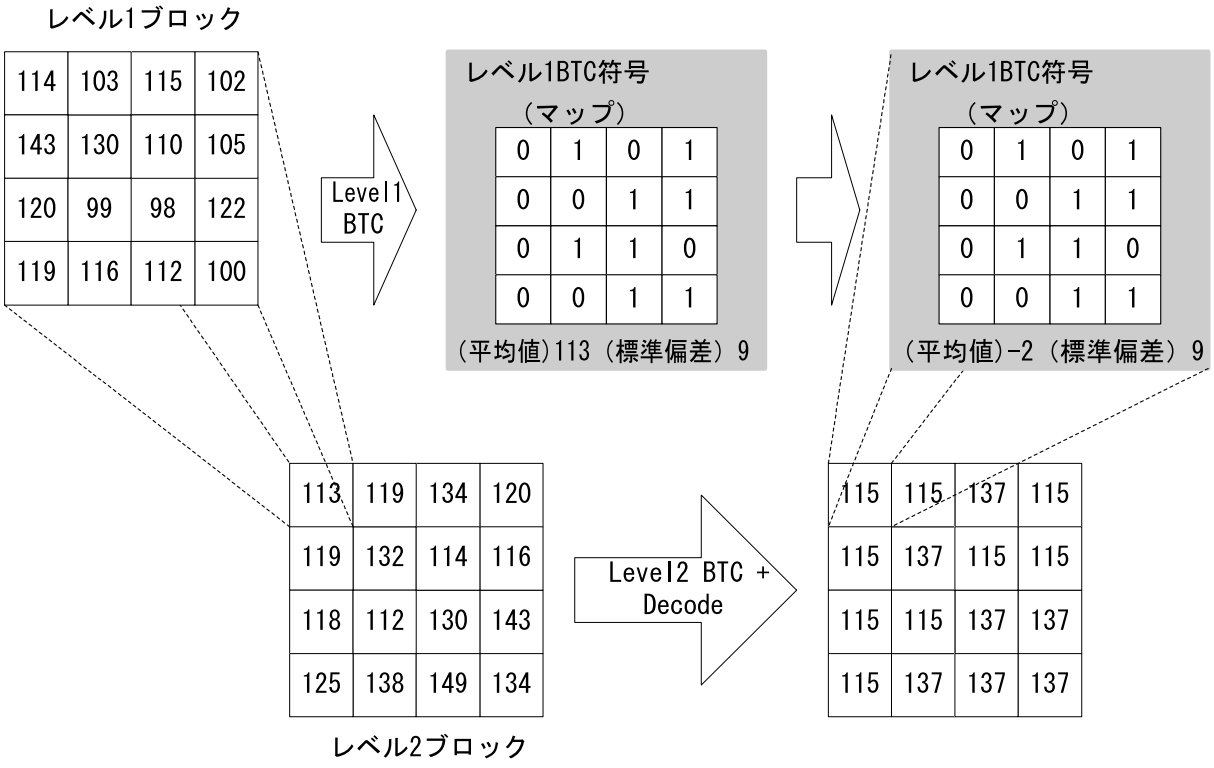


図 2.5　多層 BTC

2.2.2　圧縮率見積もり

SC では、BTC 符号に基づいてブロックを幾つかのクラスに分類し、クラスに応じて不要な符号を切り捨てる。

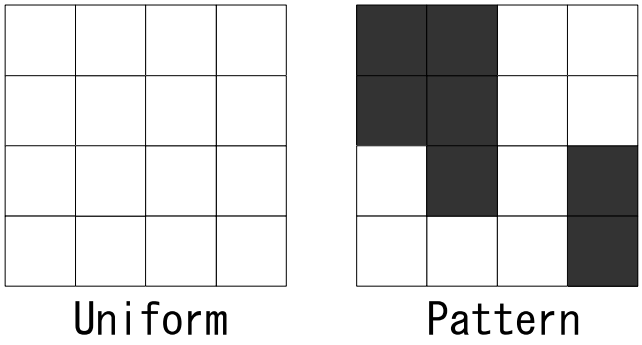


図 2.6　ブロックのクラス

例えば、図 2.6 左に示す画素濃度が均一な Uniform ブロックの場合、BTC 符号のうち濃度パターンを表現するマップ、濃度差を表現する標準偏差が不要となり、これらを切り捨て

て圧縮率の向上が可能となる。

したがって、SC の圧縮率は画像のクラス占有率に依存する。例えば、Uniform ブロックが多い場合圧縮率は大幅に上がり、逆に Pattern ブロックが多い場合高い圧縮率は見込めない。

表 2.2 に、以下の手法を用いた SC の圧縮率見積もりを示す。

- 圧縮率の最小値はレベル 2 を含めすべてのブロックが Pattern とする。
- 圧縮率の最大値はレベル 2 を含めすべてのブロックが Uniform とする。
- 可変長符号化により 2/3 圧縮されたものとする。
- 多層 BTC は、レベル 1 ブロックの平均値をすべて 0 とする。

表 2.2 SC の圧縮率見積もり

	SC（レベル 1BTC のみ）	SC（多層 BTC）
圧縮率見積もり	1/9〜1/24	1/12〜1/384

2.2.3 問題点

本研究では既に、以下の 2 種類の SC を DDMP4G に実装し、シミュレーションにより処理性能を評価した。

1. ブロックを多層化しない SC
2. すべての処理を実装した SC

1 の場合、高い処理性能を実現したが、圧縮率は Motion-JPEG 以下となる。2 の場合、圧縮率は高いが、処理負荷が高く高速化が困難であるという結果が出た。これは前節でも触れたとおり DDMP4G がまだ発展途上のプロセッサであることに起因する。

以上の結果を踏まえて、本研究では BTC を多層化せず圧縮率の向上を可能とする方式の構築を目指した。

第 3 章

ピクセルベース動画圧縮方式

3.1 Pixel Based Compression : PBC

周波数変換ベースの動画圧縮方式は、重要な画像情報の損失なく圧縮率を向上することが可能であるが、動き補償や DCT 等複雑な処理を必要とするため高速化が困難である。更に、これらの多くは標準化されているためセキュリティ面の問題を伴う。ネットワーク上で盗聴されれば容易に復号可能であり、有料配信を目的とした映像や、プライバシ重視の映像を送信する場合、映像データへの暗号化処理が必要となる。これが更に、処理の高速化を困難にしている。

そこで本研究では、標準化を目的としない高速な映像圧縮アルゴリズムの構築を目標とし、BTC に着目した。前述したように、BTC は画素データから直接圧縮符号を生成するアルゴリズムを持つため処理負荷が低く、不可逆な方式であるが実用的な画質の保持を可能としている。

よって、BTC ベースの動画圧縮方式は次のような用途に有効である。

- 画像全体に渡り、高い画質を必要としない用途
- リアルタイム性を重視する用途

BTC ベースの方式としては現在 SC が存在するが、圧縮率を高めるために多層 BTC や復号処理等高負荷な処理を必要とし、高速化が難しい。

そこで、BTC をベースとして簡易な処理の付加により圧縮率の向上を可能としたピクセルベース動画圧縮方式（Pixel Based Compression、以下 PBC）を提案する。

3.2 PBC の処理内容

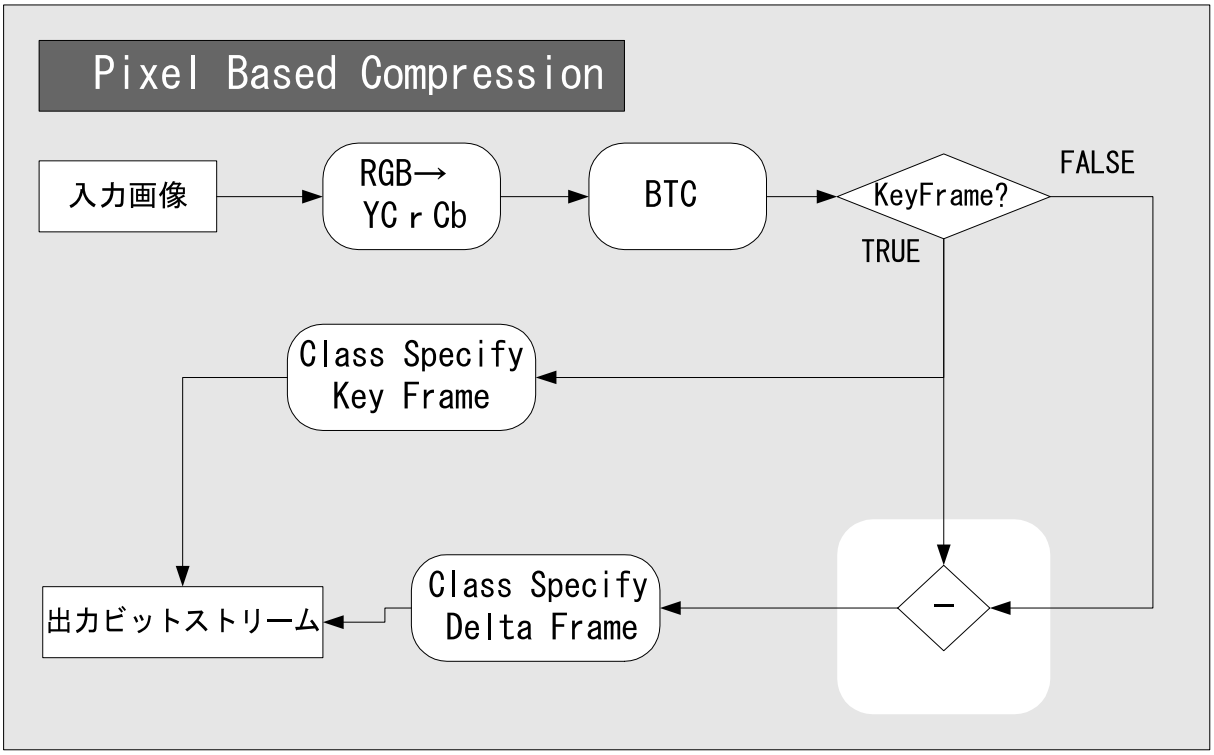


図 3.1 Pixel Based Compression

PBC 処理の流れは図 3.1 のようになる。まず、RGB の色空間を輝度・色差からなる YCrCb の色空間に変換し、BTC 符号化処理を行う。次に、Class Specify 処理により BTC 符号のうち時間的・空間的な冗長部分を削減する。

周波数変換ベースの方式では、時間的冗長部分を動き補償により削減し、空間的冗長部分を DCT と量子化により削減するがこれらは多量の演算を必要とする。

それに対して Class Specify は、簡易な処理による冗長部分の削減が可能である。まず、ブロックの濃度差が小さい場合これを空間的冗長部分として削減する。更に、1 秒につき画質重視のキーフレームを 1 フレーム取り、残りを圧縮率重視のデルタフレームとする。そして、図 3.2 に示すようにデルタフレームとキーフレームの差分を取り、差分が少ない部分、すなわち動きの少ない部分を時間的冗長部分として削減する。

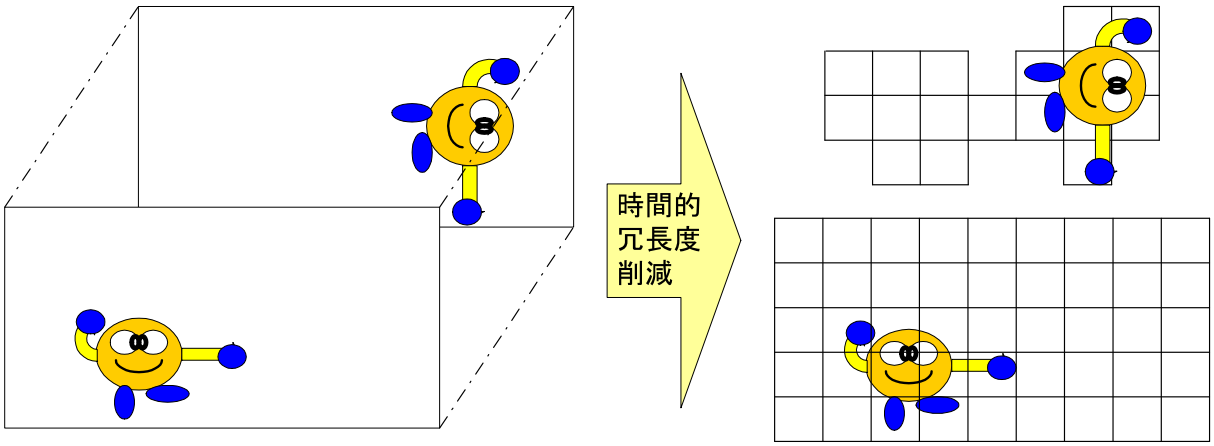


図 3.2 PBC における時間的冗長部分削減法

3.2.1 RGB-YCrCb 変換処理

PBC ではグレイスケール専用の BTC を拡張し、3 次元の色空間を扱えるようにしている。入力された RGB（赤・緑・青）の色空間は、輝度 Y と色差 Cr・Cb からなる YCrCb の色空間に変換する。人間の視覚は、輝度成分に敏感で色差成分に鈍感という特性を持つ。この特性を利用し不要な色差情報を削減することで、圧縮率の向上が可能となる。変換式は以下のものを用いる。

- RGB → YCrCb

  $Y = 0.2990R + 0.5870G + 0.1140B - 128$

  $Cr = 0.5000R - 0.4187G - 0.0813B$

  $Cb = -0.1687R - 0.3313G + 0.5000B$

- YCrCb → RGB

  $R = Y + 128 + 1.402000Cr$

  $G = Y + 128 - 0.34414Cb - 0.71414Cr$

  $B = Y + 128 + 1.77200Cb$

Y 成分の変換式で 128 減算しているが、これは YCrCb の画素値を符号付き 8 ビット（−128〜127）で表現するためである。

3.2.2 BTC

BTC では、画像を 4 × 4 マクロブロックに分割し、ブロック毎に平均値、画素の濃度パターンを 0 と 1 で表現したマップ、画素の濃度差を表す標準偏差を算出する。PBC ではこれに加え、Cr・Cb の平均値を算出する。各値の生成式は以下のようになる。

- 平均値（YCrCb）

  平均値 $= \sum_{i=0}^{3} \sum_{j=0}^{3} (i,j) \div 16$

- マップ

  $\mathrm{Y}(i,j) \geq$ Y 平均値$\rightarrow \mathrm{MAP}(i,j) = 0$

  $\mathrm{Y}(i,j) <$ Y 平均値$\rightarrow \mathrm{MAP}(i,j) = 1$

- 標準偏差（Y）

  Y 標準偏差 $= \{\sum_{i=0}^{3} \sum_{j=0}^{3}$（Y 平均値 $-\mathrm{Y}(i,j)$）$\times \mathrm{MAP}(i,j)\} \div 8$

3.2.3 Class Specify（キーフレーム用）

PBC では、BTC 符号に基づいてブロックを幾つかのクラスに分類し、クラスに応じて不要な符号を切り捨てることで圧縮率を上げる。

キーフレーム用のクラス判定処理では、ブロック内の画素濃度差が大きい場合は No-Uniform（均一でない）クラス、小さい場合は Uniform（均一）クラスを割り当てる。Uniform クラスの場合、BTC 符号のうち濃度パターンを表現するマップ、濃度差を表現する標準偏差が不要となり、これらを切り捨てることで空間的な冗長部分を削減する。表 3.1 にクラス分類条件とブロックの圧縮率を示す。

表 3.1 クラス分類条件とブロックの圧縮率

クラス	必要な符号	圧縮率	分類条件
No-Uniform	YCrCb 平均値, マップ, Y 標準偏差, クラス	1/8	Y 標準偏差 $>$ 閾値
Uniform	YCrCb 平均値, クラス	1/16	Y 標準偏差 $\leq$ 閾値

3.2.4 Class Specify（デルタフレーム用）

デルタフレームのクラス判定処理では、空間的冗長部分の削減に加え、時間的冗長部分の削減を行う。同一空間上にあるキーフレームの BTC 符号との差分値を取り、これが小さい場合その符号は時間的に冗長なものと判断し切り捨てる。差分値として、以下の 2 つの値を算出する。

- 差分値 1：ビットカウント (MAP(デルタフレーム) $\oplus$ MAP(キーフレーム))

  ※ ビットカウント ()：括弧内の 1 の個数を数える。
- 差分値 2：|Y 平均値 (デルタフレーム) − Y 平均値 (キーフレーム)|

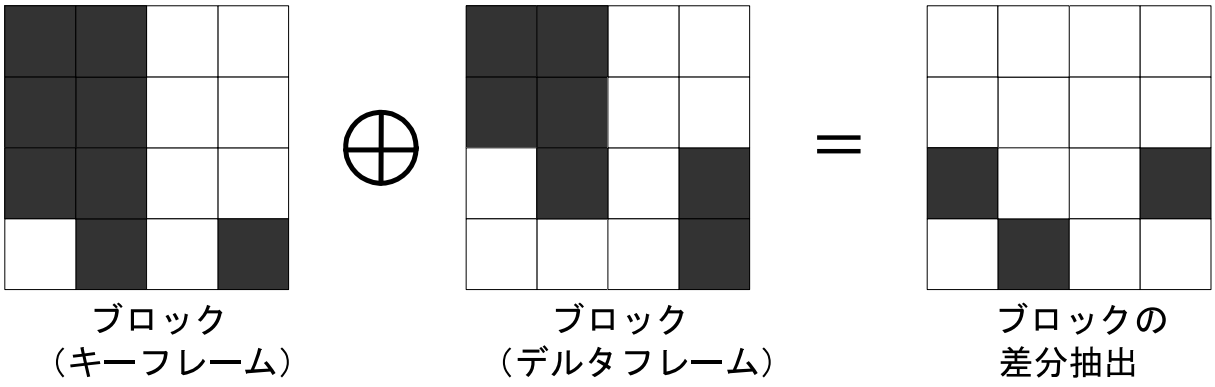


図 3.3 差分値 1

差分値 1 は、図 3.3 に示すようにブロックの濃度パターンを照合して算出する。差分値 2 は、ブロックの Y 平均値を照合し絶対値を算出する。差分値 1、2 共に閾値以内ならば、そのブロックには Static（静的）クラスを割り当て、差分が大きい場合そのブロックは Dynamic（動的）であると判断する。Dynamic ブロックは、キーフレームと同様画素の濃度差にしたがって、Dynamic（動的）クラスと Dunamic-Uniform（動的かつ均一）クラスに分類する。

表 3.2 にクラス分類条件とブロックの圧縮率を示す。

表 3.2　クラス分類条件とブロックの圧縮率（デルタフレーム）

クラス	必要な符号	圧縮率	分類条件
Static	クラス	1/192	差分値 1・差分値 2≤ 閾値
Dynamic	YCrCb 平均値, マップ, Y 標準偏差, クラス	1/8	差分値 1> 閾値又は差分値 2> 閾値、Y 標準偏差 > 閾値
Dynamic-Uniform	YCrCb 平均値, クラス	1/16	差分値 1> 閾値又は差分値 2> 閾値、Y 標準偏差 ≤ 閾値

Static クラスの場合、BTC 符号はほぼキーフレームと同じなので、すべて切り捨ててクラス符号のみを残す。これにより時間的な冗長部分が削減される。

3.3　PBC の圧縮率

PBC では、BTC 符号にクラスを割り当てることで空間的・時間的に冗長な部分を削減し、圧縮率の向上を可能としている。よって、圧縮率はクラスの占有率に依存する。

以下の手法を用いて算出した PBC の圧縮率を表 3.3 に示す。

- Dynamic クラス、すなわち動いている部分の占有率ごとに圧縮率を算出。
- 圧縮率が最小値の場合、ブロックのクラスは Dynamic と Static。
- 圧縮率が最大値の場合、ブロックのクラスは Dynamic-Uniform と Static。

- 各符号長は、クラスが 2 ビット、マップが 16 ビット、平均値・標準偏差は 7 ビットとする。

表 3.3　PBC の圧縮率

動き部分占有率（%）	圧縮率（キーフレーム）	圧縮率（デルタフレーム）
50	1 / 8 〜1 / 16	1 / 16 〜1 / 30
40	1 / 8 〜1 / 16	1 / 19 〜1 / 36
30	1 / 8 〜1 / 16	1 / 25 〜1 / 46

動いている部分の占有率が 30%のとき、1/30〜1/40 の圧縮率が可能な MPEG2 とほぼ同等となる。

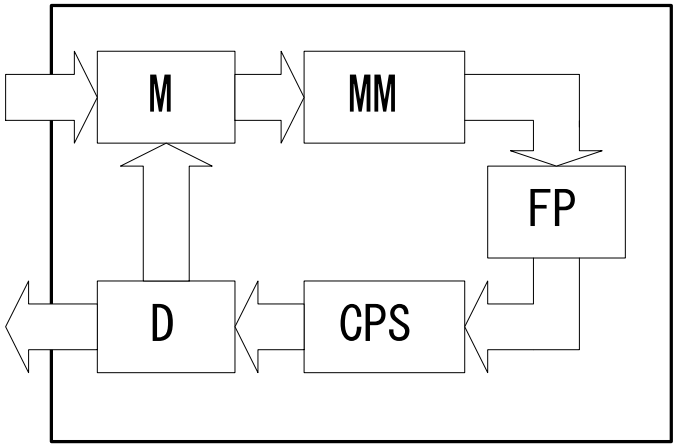
第 4 章

データ駆動型 PBC の実現法

4.1 DDMP

4.1.1 DDMP の基本アーキテクチャ

DDMP は、従来のノイマン型プロセッサとは異なる独自のアーキテクチャを採用することで、高速処理・低消費電力を実現している。ノイマン型プロセッサがプログラムカウンタにより逐次的に命令を取り出して実行するのに対し、DDMP は行き先情報、世代識別子等を含むパケット形式のデータを取り扱う。したがって、パイプライン処理の効率的な実現が可能である。



MM : Matching Memory （待ち合わせメモリ）
FP : Functional Processing Unit （演算部）
CPS : Cache Program Storage （プログラム記憶部）
M : Flow Merging Module （合流部）
D : Flow Diverting Module （分岐部）

図 4.1 DDMP のナノプロセッサ構成図

図 4.1 に DDMP のナノプロセッサ構成を示す。パケット内の行き先情報は、図 4.1 の分岐部 D においてデータの行き先の決定に使用される。行き先情報は処理内容に応じて CPS 部で変更され、これによりデータは独立してプロセッサ内を巡回する。世代識別子は、フィールド・ライン・ピクセルの 3 次元で構成される。DDMP は、同じ行き先情報・世代識別子を持つデータ同士に対してのみ処理を実行する特性を持つ。図 4.1 の MM 部に処理対象となるデータのうち先着のものを格納し、同一の識別子を持つデータが MM 部に到達した時、格納していたデータを読み出し FP 部で演算処理する。これらの識別子により、図 4.2 に示すように各データに独立性が与えられて空間的・時間的な並列処理が可能となる。現在開発されている DDMP4G は、8600MOPS の処理性能を実現している。

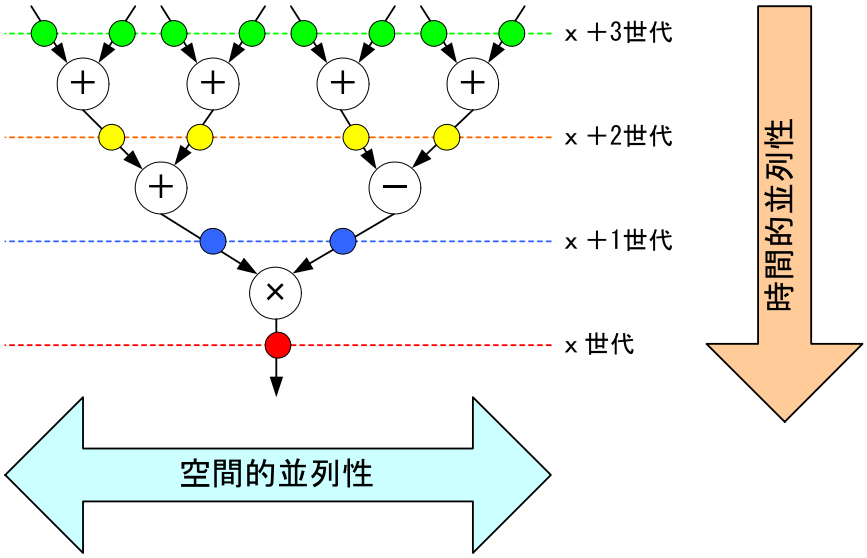


図 4.2 DDMP の並列概念図

また DDMP は、有意なデータを持つデータラッチのみを駆動させる自己タイミング型パイプライン機構 [1][7] との親和性が高く、これの採用により低消費電力を実現している。

4.1.2 DDMP と映像処理の親和性

前述したように、DDMP では世代識別子と呼ばれる 3 次元の識別子を用いる。映像処理においては、ピクセル番号で水平方向、ライン番号で垂直方向、フィールド番号で時間方向

を識別する。(図 4.3) これにより、各画素は独立性を持ち並列処理が可能となる。

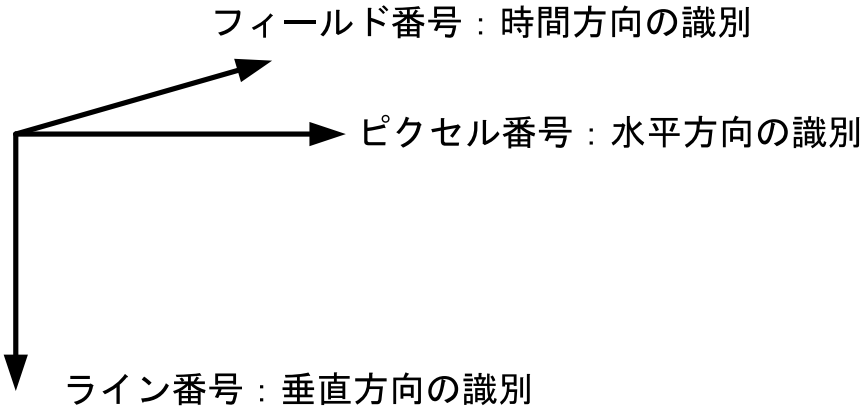


図 4.3 映像処理における世代識別子の割り当て方

また、このアーキテクチャの有効性を高めるため、DDMP には世代識別子に定数を加算した値をアドレスとしてアクセス可能なメモリを搭載している。(図 4.4)

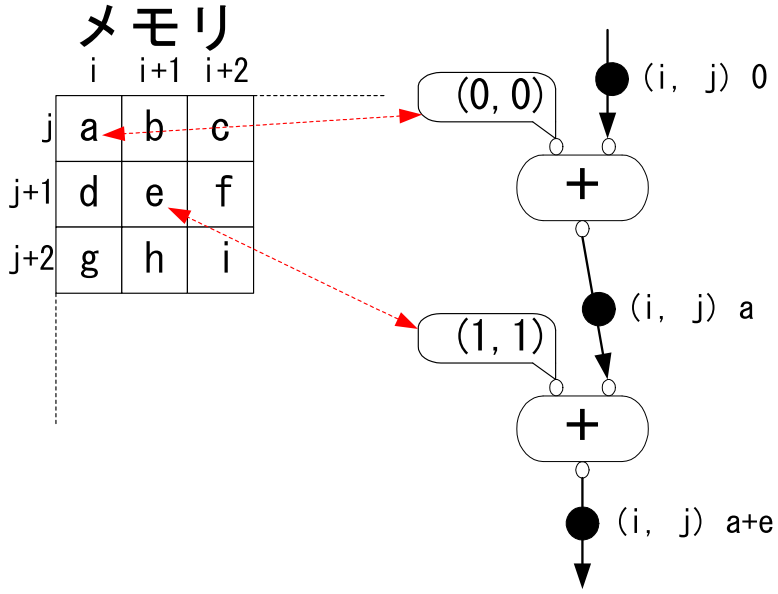


図 4.4 メモリアクセスの方法

DDMP4G には、8k 個のデータを格納できる内部メモリを 2 つと、用途に応じて容量を変更可能な外部メモリを 1 つ搭載している。これらのメモリを有効に利用し、プロセッサ内を巡回するパケット数を極小化することで、処理性能の向上が可能となる。

3 次元の世代識別子とそれを使ってアクセス可能なメモリを搭載することで、単純なアドレス指定型線形メモリを搭載したプロセッサに比べ、DDMP は映像処理との親和性を高めている。

4.2　データ駆動型 PBC 実現法の要件

PBC は、色空間の並列性、4 × 4 マクロブロックの並列性、並びにマクロブロック内の各画素の並列性が極めて高く、図 4.5 に示すような並列アルゴリズムが可能である。

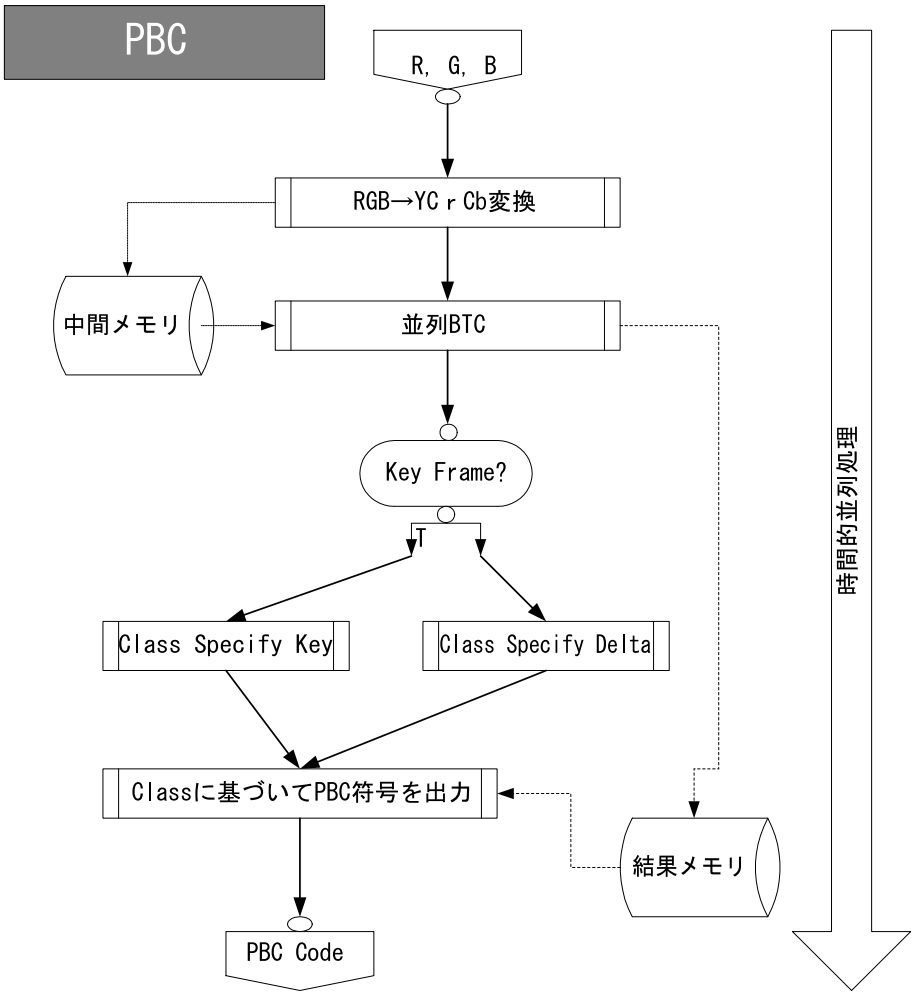


図 4.5　データ駆動型 PBC

まず、RGB の画素データを入力し、これを YCrCb の色空間に変換する。この処理において、各画素は独立に並列処理できる。次の BTC 処理以降は 4 × 4 マクロブロックを基本処理単位とし、これを時間的並列処理する。このとき、プロセッサ内を巡回するパケットは 1 ブロックにつき 1 パケットのみとし、残りのデータは中間データ用のメモリに待避させ必要に応じて読み出すことで、プロセッサの余剰負荷を削減する。BTC 処理で得られた符号

は結果データ用のメモリに待避させ、フレームの種類に応じて Class Specify 処理を行い、得られたクラスを基にメモリから BTC 符号を読み出す。

以下に、各処理毎の詳細なデータ駆動型実現手法を述べる。

4.2.1 RGB → YCrCb

RGB → YCrCb の変換式は 3 章に示したが、DDMP4G には一般的に使用頻度が低く回路規模が増大するため除算命令がなく、整数型のデータのみを扱うため 3 章の式を適用できない。このような DDMP で色空間の変換を行う場合、8 ビット右シフト（256 で除算）を用いて近似値を算出する。

- RGB → YCrCb

  Y=(77/256)R+(150/256)G+(29/256)B−128

  Cr=(1/2)R−(107/256)G−(21/256)B

  Cb=(-43/256) R−(85/256)G+(1/2)B

- YCrCb → RGB

  R=Y+128+(359/256)Cr

  G=Y+128−(88/256)Cb−(183/256)Cr

  B=Y+128+(454/256)Cb

4.2.2 並列 BTC

BTC は高い並列性を持ち、図 4.6 に示す並列 BTC アルゴリズムが有効である。

平均値算出部では、マクロブロックの代表パケットでメモリ内のデータ読み出しながら加算を 15 回繰り返す。YCrCb の色空間にデータ依存関係は無く、並列処理が可能である。次にマップと差分値を 1 画素ごとに各 16 並列で処理する。この部分では、4 × 4 個のパケッ

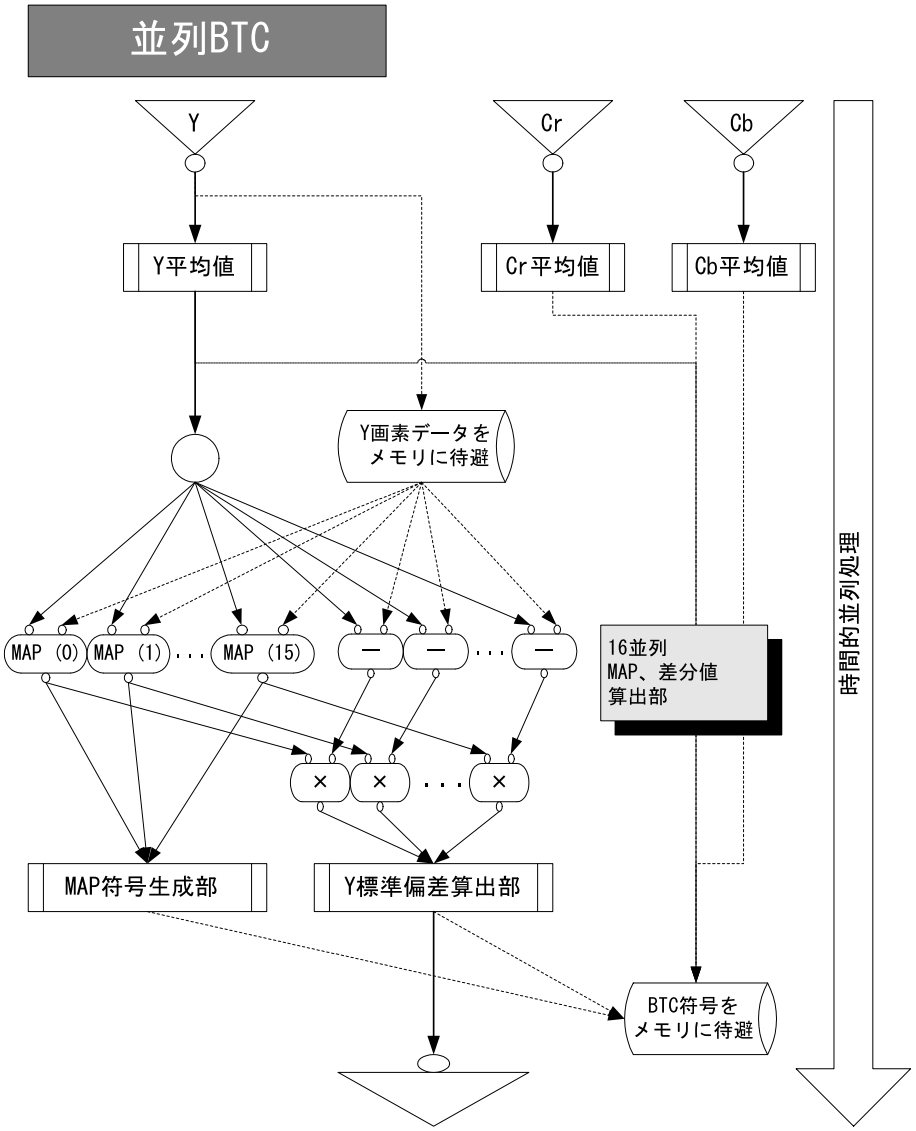


図 4.6　並列 BTC

トが同時にプロセッサ内を流れるため、BTC において最も高負荷となる。DDMP4G では、次節で紹介する 2 世代化を用いて負荷を軽減する。2 世代化により隣接する 2 つの画素を 1 パケットに格納し、8 並列の算出機構で 16 並列と同等の処理が可能となる。マップ符号生成部では、16 個のマップを 1 つのマップ符号にまとめる。DDMP4G では、データ長が 12 ビットなので、8 ビットのマップ符号を 2 つ生成する。Y 標準偏差算出部では、16 個の差分値とマップを乗算した値の合計を算出する。

生成した BTC 符号はすべて一度メモリに格納し、最後に Class Specify でブロックの種

別を判定しそれに応じて符号を出力する。

4.2.3 Class Specify Key

キーフレーム用クラス判定処理では、メモリから Y 標準偏差を読み出し、値が閾値を超えれば No-Uniform クラスと判定し BTC 符号をすべて読み出す。値が閾値以下ならば Uniform クラスと判定し BTC 符号のうち平均値のみを読み出す。最後にクラス符号を割り当てる。本研究における試作プログラムでは、以下のようにクラス符号を割り当てている。

- No-Uniform クラス : 0
- Uniform クラス : 1

4.2.4 Class Specify Delta

デルタフレーム用クラス判定処理は、図 4.7 に示すようにメモリから逐次データを読み出しながら実行する。

差分値 1 の算出は、結果メモリからキーフレームとデルタフレームのマップを読み出し、排他的論理和を取って差分値の 1 の個数を数える。差分値 2 は、結果メモリからキーフレームとデルタフレームの平均値を読み出し、差分の絶対値を取る。差分値 1・2 いずれかが閾値を超える値ならば Dynamic と判断し、差分値 1・2 共に閾値以内ならば Static と判断する。Dynamic の場合、Y 標準偏差を読み出しこれが閾値を超えれば Dynamic クラスと判定し BTC 符号をすべて読み出す。閾値以下ならば Dynamic-Uniform クラスと判定し BTC 符号のうち平均値のみを読み出す。Static の場合、BTC 符号は読み出さない。最後にクラス符号を割り当てる。本研究における試作プログラムでは、以下のようにクラス符号を割り当てている。

- Dynamic クラス : 0
- Dynamic-Uniform クラス : 1
- Static クラス : 2

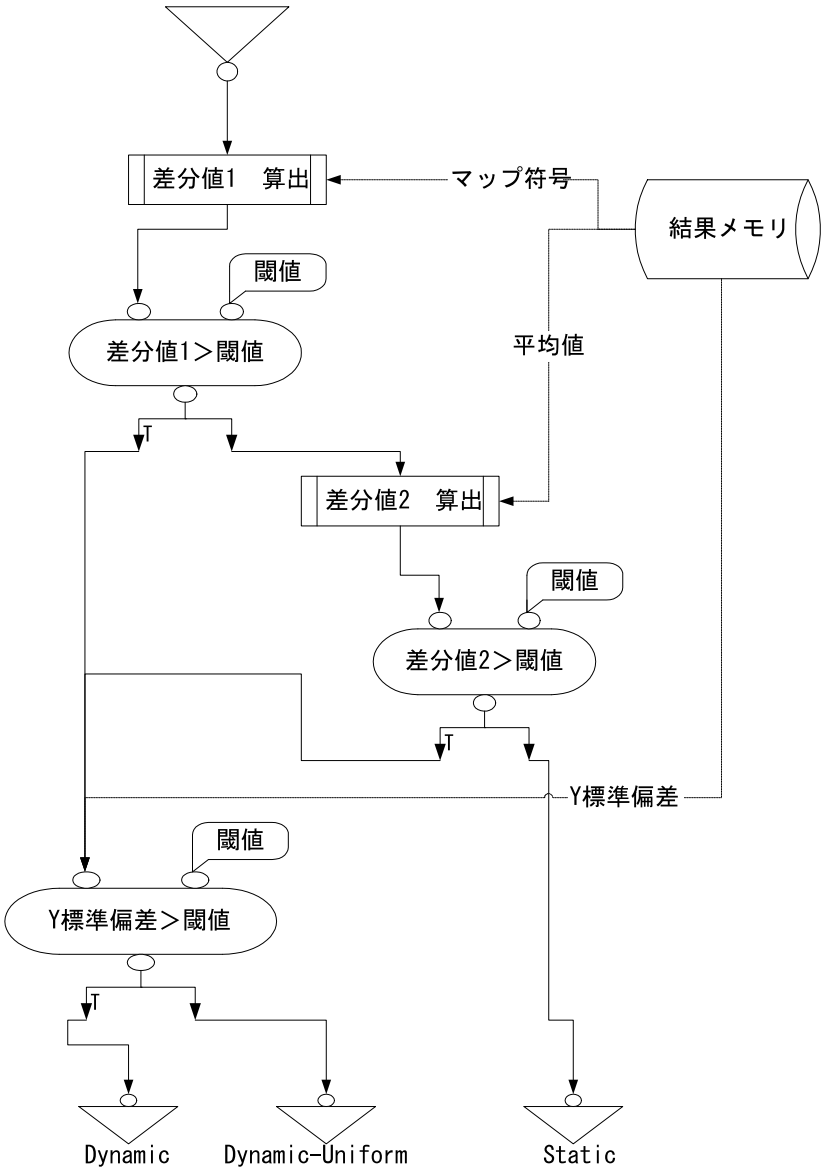


図 4.7　デルタフレーム用クラス判定処理

4.3　DDMP4G 向け PBC 最適化手法

2 章で述べたように、DDMP4G はまだ発展途上のハードウェアである。そこで、1 パケットに 2 つのデータを格納可能な 2 世代化と呼ばれる DDMP4G 特有の方式を有効活用すると共に、簡単なハードウェア機構の追加により実現可能な命令セットの付加により、DDMP4G による PBC の高速化を実現する。

4.3.1　データ駆動型 PBC の DDMP4G への実装方法

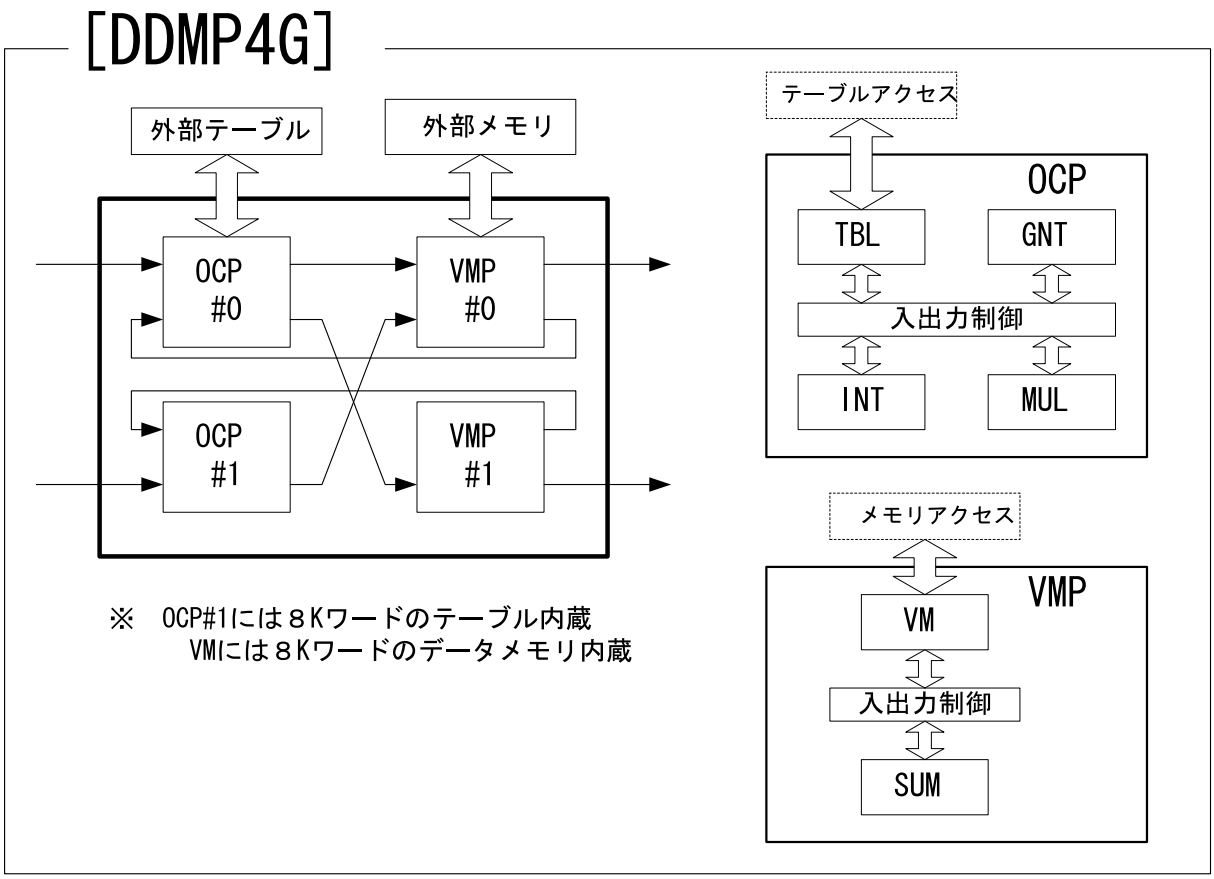


図 4.8　DDMP4G

図 4.8 に DDMP4G のプロセッサ構成を示す。DDMP4G には OCP、VMP と呼ばれるマクロプロセッサが各 2 個、そしてマクロプロセッサ内に図 4.1 に示したナノプロセッサが OCP に 4 個、VMP に 2 個搭載されている。OCP では、整数演算、テーブルメモリアクセス、世代操作等を実行し、VMP ではメモリアクセス等を実行する。

PBC においては、RGB-YCrCb 変換を OCP0 に、負荷の高い BTC を OCP1 と VMP1 に、クラス判定処理を OCP0 と VMP2 に割り当てる。図 4.5 に示した中間メモリは VMP1 の内部メモリを用いる。キーフレームの BTC 符号はデルタフレームのクラス判定処理に必要なので、結果データは VMP0 からアクセス可能な大容量の外部メモリに格納する。

4.3.2 2 世代化

DDMP4G は、2 世代化（図 4.9）と呼ばれる 1 つのパケットに 2 つのデータを格納し、それらを同時に処理する方式を可能としたハードウェア機構を備えている。それぞれのデータの世代番号は、偶数世代とそれに 1 を加えた奇数世代であり、画像処理においては隣接する画素データを 2 世代パケットに格納する。2 個のデータはそれぞれ、偶数世代を第 0 データ、奇数世代を第 1 データと呼ぶ。DDMP4G には、2 世代化用の命令として、2 つの 1 世代パケットから 2 世代パケットを生成する命令が用意されており、この命令を実行することでプロセッサ内を巡回するパケット数を半分間引いて、処理負荷の半減が可能となる。

2 世代パケットは、条件分岐部において 2 つのデータの条件が異なる時、分解されて 1 世代パケットとなる。したがって、以下のような処理には 2 世代化の適用が難しい。

- 処理内容がデータに依存する処理
- 隣接する画素データに対して異なる処理を行う処理

PBC 処理のうち、RGB → YCrCb 変換処理、BTC 処理などは上の条件に当てはまらないため、2 世代化との親和性が高くこれを適用することで処理の高速化が可能となる。

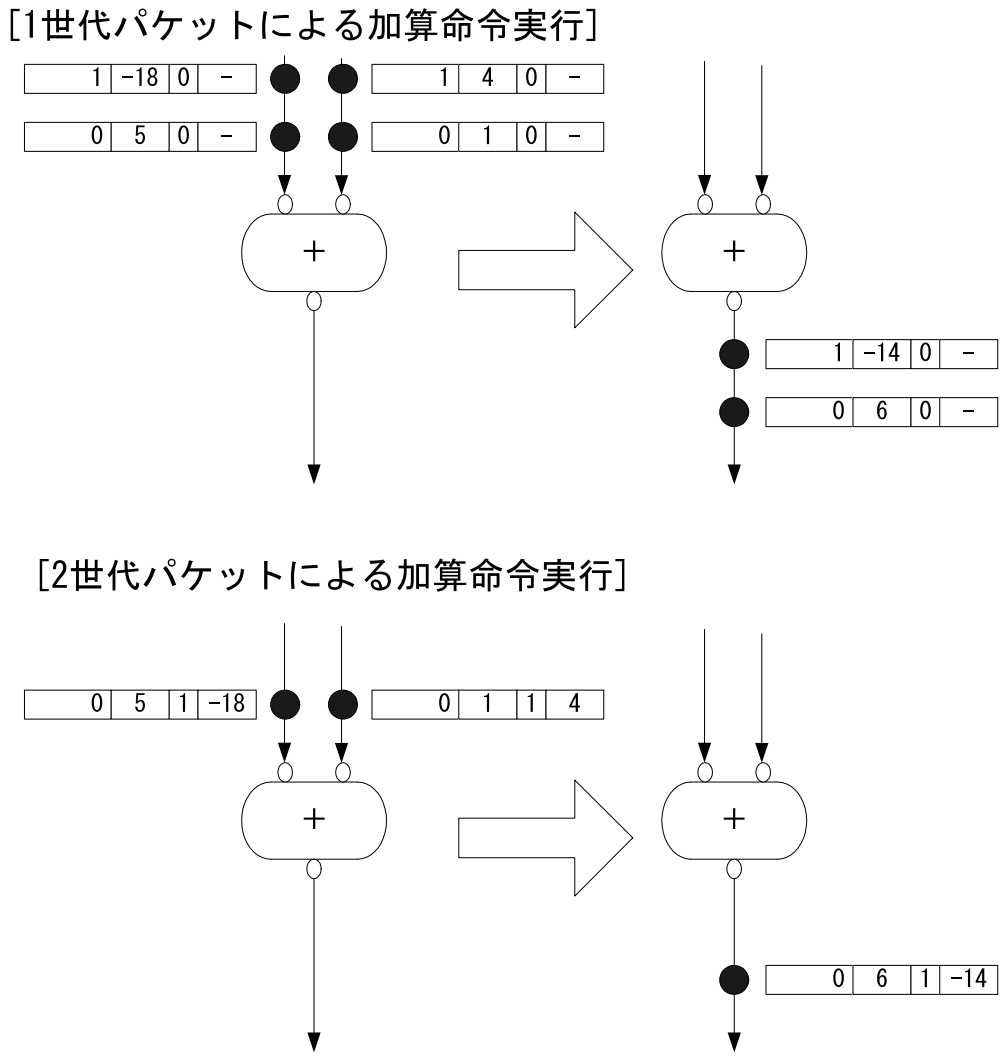


図 4.9　2 世代化

4.3.3　2 世代化特化命令セット

前述したように、2 世代化はパケットを半分間引いて最大で 2 倍の処理性能向上を可能とする有効な方式である。しかし、DDMP4G の既存命令を用いて PBC に 2 世代化を適用す

る場合、一度 2 世代パケットを分解し再び 2 世代パケットを生成するといった 2 世代化特有の処理を要し、2 世代化本来の目的である負荷軽減の妨げとなっている。

そこで、PBC と 2 世代化の親和性を高めるための 2 世代化特化命令セットを提案する。図 4.10 に基本演算用の 2 世代化特化命令セットを示す。

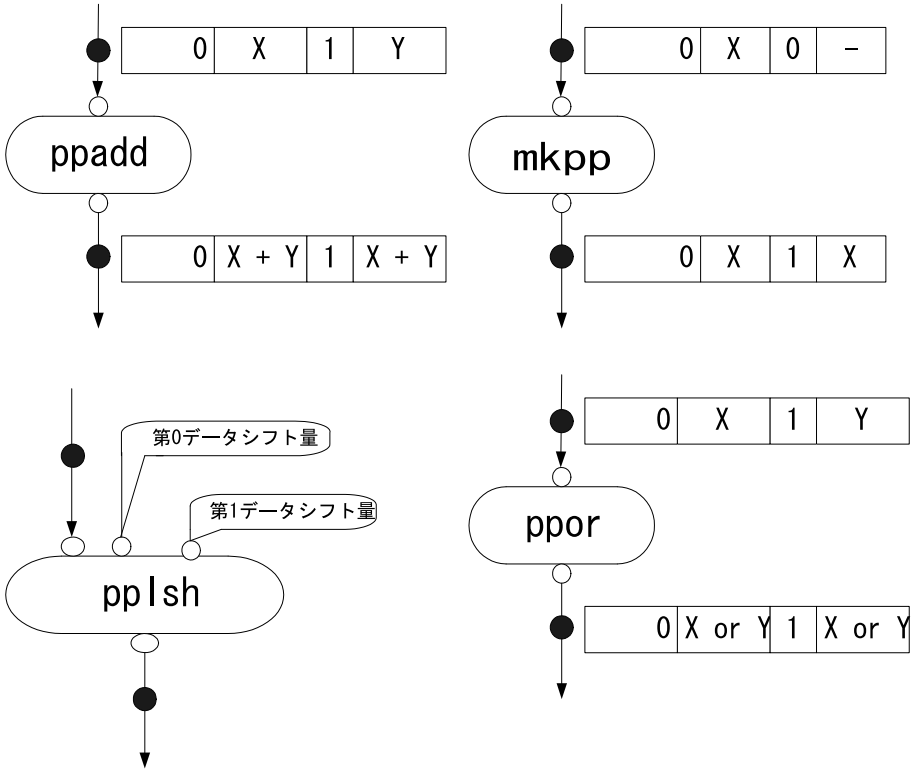


図 4.10　2 世代化特化命令セット（基本演算用）

ppadd（Pair-Packet ADD）は、2 世代パケット内の 2 つのデータを加算する命令である。DDMP4G の既存命令で同様の処理をする場合、一度パケットを分解し加算した後再び 2 世代化する必要がある。PBC では、BTC の Y 平均値算出部及び標準偏差算出部、デルタフレーム用クラス判定処理の差分値 1 算出部に用いる。

mkpp（MaKe Pair-Packet）は、1 世代パケットに対して行われる処理で、世代番号の LSB（Least Significant Bit : 最下位ビット）に 0 を代入し、2 世代フラグに 1 を代入し第 0 データと同一のデータを第 1 データにコピーする命令である。PBC では、Cr・Cb の平均値算出部で用いる。

pplsh（Pair-Packet Logical SHift）は、第 0 データと第 1 データに別々に定数を与え定数分論理シフトを行う命令である。ppor（Pair-Packet OR）は、2 世代パケット内の 2 つのデータの論理和を取る命令である。PBC では、共に BTC のマップ符号生成部で用いる。

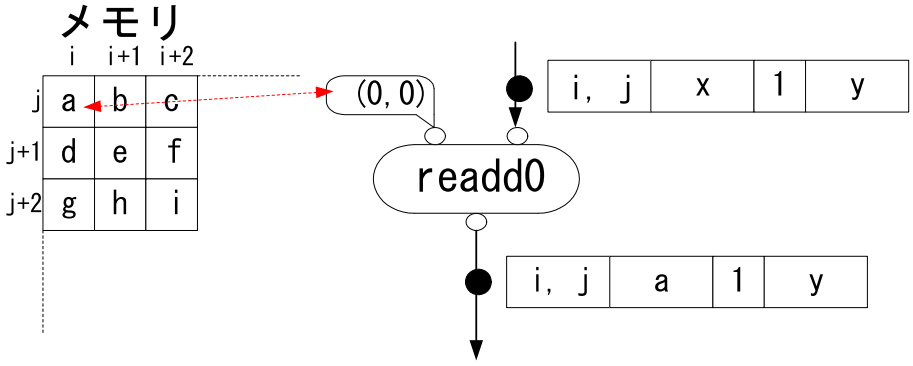


図 4.11　2 世代化特化命令セット（メモリアクセス用）

次に示す 2 種類の命令は共にメモリアクセス用の命令である。writed0（WRITE Data 0）は、2 世代パケット内の第 0 データのみをメモリに書き込む命令である。readd0（READ Data 0）（図 4.11）は、2 世代パケットの偶数世代番号でメモリにアクセスし、読み出した値を第 0 データにコピーする命令である。これらの命令は BTC 符号のメモリ書き込み・読み出し等に用いる。

4.3.4　連続メモリ参照命令セット

PBC では、多くのメモリアクセスを要し、これに多数の命令を要するため処理性能向上が難しい。そこで、1 命令でメモリを連続して参照可能な命令セットを提案する。

図 4.12 に、連続メモリ参照命令セットを示す。sqadd（SQuare read and ADD）は、世代番号を始点とし、世代番号に定数を加算したアドレスまでのデータを読み出し、これらを加算する命令である。この命令は、BTC の平均値算出部に適用する。

sqread（SQuare READ）は、世代番号を始点とし、世代番号に定数を加算したアドレスまでのデータを読み出す命令である。この命令は、クラス判定処理後符号を読み出す部分に適用する。

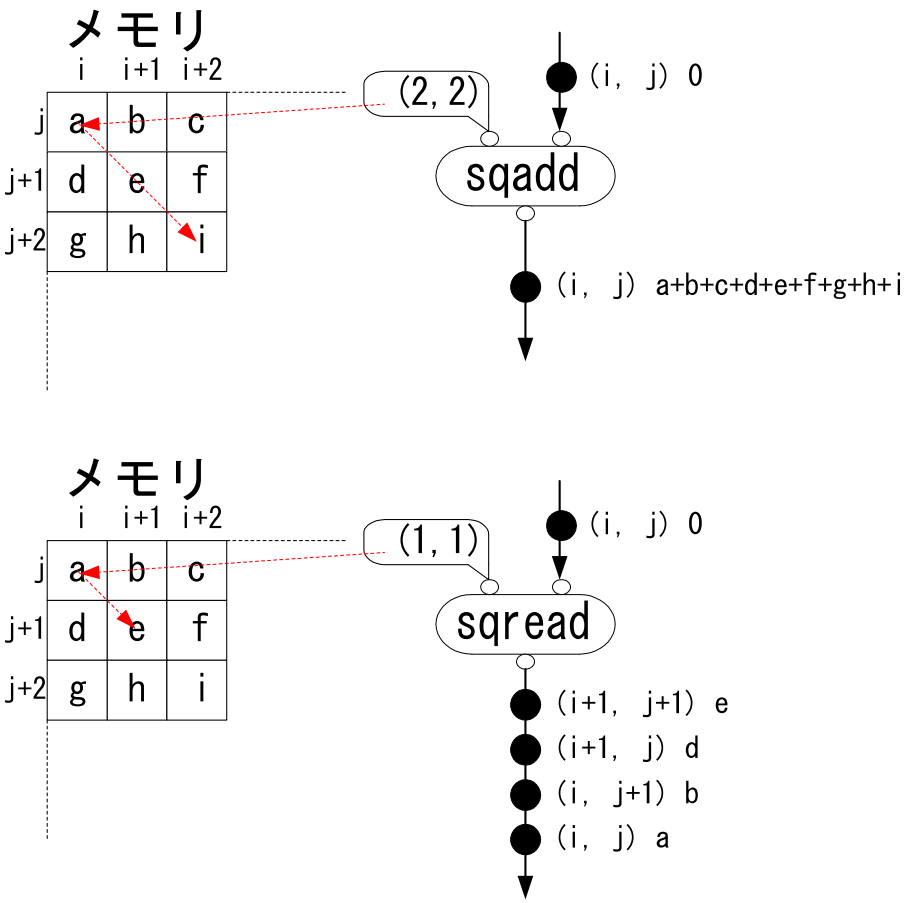


図 4.12 連続メモリ参照命令セット

第 5 章

性能評価

5.1 DDMP4G チップへの BTC 実装実験

2 章で紹介した SC の DDMP4G への実装実験の一環として、DDMP4G2 チップを用いてこれに BTC を実装する実験を行った。チップ 1 には RGB-YCrCb 変換と BTC 符号化のプログラム、チップ 2 には BTC 復号と YCrCb-RGB 変換のプログラムを実装した。

チップ 1 に実装した BTC 符号化プログラムは 4 章に示した並列 BTC のプロトタイプで、2 世代化向けに最適化し、マップ符号生成部は除いたものである。

結果、毎秒 640 × 240 × 15 フレームの動画の符号化・復号に成功し、BTC が DDMP チップで実現可能なことが証明された。

5.2 シミュレーションによる性能評価実験

5.2.1 シミュレータの原理

本研究のシミュレーションは、DDMP4G のフローグラフシミュレータで行う。このシミュレータは DDMP4G のチップ構成を基にした、フローグラフシミュレーションツールである。プログラムとして、データフローをノードと矢印で示したフローグラフを作成し、シミュレータがこれを解析して入力データを演算する。

入力データは世代番号・エントリー番号・データからなるパケットデータであり、本研究では、1 画素につき色空間の数、すなわち 3 パケットのデータを同時に入力する。これは、DDMP4G のデータ長が 12 ビットしか扱えないためである。

5.2.2 シミュレーション手法

実験には、640 × 64 の画像データを用いた。BTC 以降のブロック処理では、1 ブロックにつき右下の画素データ 1 パケットのみをプロセッサに流して処理を行う。よって、4x−1 (x=1, 2, ...) ラインのパケットが入力された時最も高負荷となり、それ以外のラインでは、YCrCb 変換後データをメモリに待避しパケットを消去するため負荷は低く、640 × 64 画像により 640 × 480 画像とほぼ同等のスループット性能を算出できる。

PBC の性能測定には、デルタフレームのフィールド番号を持つ画像データを入力する。これは、デルタフレームの処理がキーフレームのものと比較して高負荷なためである。

測定式は以下を用いる。

1. レスポンスタイム:

   マクロブロックデータが入力されて出力されるまでの時間 (s)

2. スループット:

$$\frac{1}{\text{マクロブロックの平均出力時間}}(blocks/s)$$

3. フレームレート（640 × 480）:

$$\frac{\text{スループット}}{640 \times 480 \div 16}(frames/s)$$

4. フレームレート（SC）（640 × 480）:

$$\frac{1}{640 \times 480 \times \text{パケット投入間隔}}(frames/s)$$

1 は性能評価指標の一つで、データが入力されてから出力されるまでの時間であり、本研究では 1 マクロブロック当りのレスポンスタイムを測定する。

2 も性能評価指標であり、単位時間当りに処理可能なデータ量を示す。ただし SC に関しては、処理に応じてブロックサイズが変化するためこの値は算出していない。

3 は、1 秒当りに処理可能な 640 × 480 サイズの動画のフレーム数であり、これは 2 の値を基に算出する。4 は SC 用のフレームレートで、これはパケット投入間隔を基に算出する。

5.2.3　動画圧縮アルゴリズムの性能評価

本節では、以下の 4 種類の動画圧縮アルゴリズムの性能を比較した。(表 5.1)

- PBC
- 並列 BTC
- SC（多層 BTC を行わす、レベル 1 ブロック符号化のみ行うもの）
- SC

プログラムはすべて DDMP4G の既存命令を用いて 1 チップに実装した。

結果、PBC は並列 BTC やレベル 1 のみの SC とほぼ同等の性能を実現し得ると考えていたが、実際には大幅な性能格差があった。SC と比較すると、2.8 倍の処理性能を可能とした。

表 5.1　性能評価

	RT (μ s)	TP (kblocks/s)	FR (f/s)
PBC	337	396	20
並列 BTC	246	555	28
SC（レベル 1 のみ）	310	438	24
SC	4518	-	7

5.2.4　提案命令セット追加による PBC の性能比較

本節では、4 章で提案した 2 世代化特化命令セット及び連続メモリ参照命令セットの有効性の検証を目的とし、PBC を以下の 4 パターン実装してその性能を比較した。(表 5.2)

- パターン 1 : DDMP4G 既存命令（2 世代化をしない従来方式）
- パターン 2 : DDMP4G 既存命令（2 世代化）
- パターン 3 : DDMP4G 既存命令＋ 2 世代化特化命令（2 世代化）
- パターン 4 : DDMP4G 既存命令＋ 2 世代化特化命令＋連続メモリ参照命令（2 世代化）

結果、パターン 1 と比較してパターン 2 は 1.3 倍、パターン 3 は 1.7 倍のスループット性能向上を可能とし、2 世代化とその特化命令セットの有効性を証明できた。

パターン 4 については、パターン 3 と比較して 1.06 倍という結果であり大きな性能の格差はなかった。しかし連続メモリ参照命令セットは、メモリアクセス回数の多い SC 等に適用すれば処理性能向上が見込めるであろう。

表 5.2　性能評価（PBC）

PBC	RT (μ s)	TP (kblocks/s)	FR (f/s)
パターン 1	454	287	14
パターン 2	337	396	20
パターン 3	256	490	25
パターン 4	242	520	27

第 6 章

結論

本稿では、標準化を目的としない高速な動画圧縮方式 PBC を提案すると共に、本方式のデータ駆動型実現法について述べた。

第 2 章では、既存の動画圧縮方式を周波数変換ベースと BTC ベースに分けて取り上げ、その特徴と問題点に触れた。周波数変換ベースの方式は、汎用性重視のため膨大な演算量を必要とし、処理の高速化が困難であるという問題点が存在した。また、これらの殆どは標準化されているため、セキュリティ面での問題点も残っていた。BTC ベースの方式としては、現在 SC が提案されているが、これは高い圧縮率を実現するため高負荷な多層 BTC 処理を伴い、DDMP4G による高速化は困難という結果であった。

第 3 章では、第 2 章で取り上げた動画圧縮方式の問題点を踏まえて、BTC に簡易な処理を付加することで圧縮率の向上を可能としたピクセルベース動画圧縮方式 PBC を提案し、その基本アルゴリズムを述べた。

第 4 章では、データ駆動方式向けの PBC アルゴリズムを提案すると共に、これを現在開発されている DDMP4G 向けに最適化する手法を 2 つ述べた。1 つは、2 世代化と呼ばれる DDMP4G 特有の処理方式の活用で、もう 1 つは簡単なハードウェア機構の追加により実現可能な命令セットの提案である。

第 5 章では、ハードウェアによる実験及びシミュレーションによる実験の結果を取り上げた。ハードウェアによる実験では、BTC 符号化・復号処理のプログラムを実装し、その性能を検証した。シミュレーションによる実験では、第 4 章で提案したデータ駆動型 PBC 及び命令セットの性能検証を行った。

本研究の成果としては、まず提案した動画圧縮方式 PBC が DDMP4G で高速に実現可能

なことを証明した。また、並列 BTC 及び BTC 復号処理のプログラムを DDMP4G チップに実装し、640 × 240 × 15fps の処理性能を確認した。更に、2 世代化とそれに特化した命令セットの提案、連続メモリ参照命令セットの提案を行い、その有効性を証明した。

本研究で得られた性能評価をもとにした PBC の応用分野としては、例えばリアルタイム・インターネット授業等が挙げられる。表 6.1 に、リアルタイム・インターネット授業用動画圧縮方式の要件を示す。

表 6.1　インターネット授業用動画圧縮方式の要件

画像サイズ	フレームレート	圧縮率要件
640 × 480	15fps	1/22

インターネット授業においては、黒板の文字並びに教師の上半身が受講者に見えることが重要であり、画像サイズは 640 × 480、フレームレートは 15 が最適値となる。圧縮率については、10Mbps の通信速度を持つ標準的な教育用ネットワークを想定して設定した。ただしこれは理論値であり、実際の通信速度は半分の 5Mbps 位であると思われる。そこで、以下の式に基づいて圧縮率を算出した。

$$\text{圧縮率} = \frac{\text{通信速度}}{\text{横サイズ}\times\text{縦サイズ}\times\text{フレーム数}\times 1\text{ 画素のデータ量}}$$

$$= \frac{5 \times 10^6}{640 \times 480 \times 15 \times 24} = \frac{1}{22}$$

PBC では、動き部分占有率 30%以下のとき応用可能である。

本研究に残された課題としては、以下の 2 つが挙げられる。

1. PBC の画質検証
2. PBC 復号処理のデータ駆動型実現法の確立

1 については、現在キーフレームが復号可能なことを確認したのみで、動画データを用いてデルタフレームの画質を検証し、問題があればブロックのクラス判定処理を見直す必要がある。この検証は、今後 JAVA で PBC を実装して行う予定である。

2 については、既に DDMP4G チップに BTC 復号処理プログラムを実装しているが、これはマップ符号を用いない機能限定版であるため、これを拡張し高い処理性能が不可能ならばアルゴリズムの見直しが必要となる。

今後の展望としては、PBC に可変長符号化を取り入れることで圧縮率の向上が見込める。ただし、画像ごとに可変長符号テーブルを作成すると処理負荷が高くなるため、殆どの画像に適用しうるテーブルを予め作成しこれを用いる方法が有効である。これにより、表 6.2 に示す程度の圧縮率が可能となり、応用分野の拡大が見込めるであろう。

表 6.2　PBC の圧縮率（今後の展望）

動き部分占有率（%）	圧縮率（キーフレーム）	圧縮率（デルタフレーム）
50	1 / 12 〜1 / 24	1 / 24 〜1 / 45
40	1 / 12 〜1 / 24	1 / 28 〜1 / 54
30	1 / 12 〜1 / 24	1 / 37 〜1 / 69

画質向上の手法については、SC ではフィルタリング処理の付加によりこれを実現するが、DDMP4G で試作版を実装した結果多数の命令を要するため実用的でない。よって、DDMP 向けのノイズ除去・エッジ強調フィルタを検討する必要がある。

また、本稿で提案した新命令は今後 DDMP チップへの追加が検討される。新命令はいずれも汎用性があり、追加されれば様々なデータ駆動型処理への応用が期待できる。

将来的には、現在の DDMP4G 並みの処理性能を持つ DDMP は超小型化し、これに PBC を実装して人間の目では確認出来ないほどの小型カメラへの搭載が可能となるであろう。これは、監視カメラとして犯罪防止に一役買うことが期待できる。また、処理能力の高い DDMP が開発されれば、現在の PBC に処理を付加することで高速・高画質・高圧縮率のデータ駆動型 PBC の実現が可能となるであろう。

謝辞

本研究に於いて、懇切なる御指導、御鞭撻を賜った高知工科大学の 岩田 誠 教授に、心より感謝の意を表します。

本研究の基礎としているデータ駆動型アーキテクチャを提唱され、様々な御示唆を賜った寺田 浩詔 教授 に心より感謝の意を表します。

本研究を進めるにあたり、様々な御助言、御指導を賜った高知工科大学 情報システム工学科の 坂本 明雄 教授、島村 和典 教授、岡田 守 教授、門田 幹夫 教授、David Greene 教授、清水 明宏 教授、Lawrence Hunter 教授、竹田 史章 教授、菊池 豊 助教授、篠森 敬三 助教授、Ruck Thawonmas 助教授、明神 千代 助教授、福本 昌弘 助教授、浜村 昌則 講師、任 向実 講師、大森 洋一 助手、妻鳥 貴彦 助手、近藤 剛 実験講師に、心より感謝の意を表します。

本研究の方向性を模索する上で重要なヒントとなった、Smart Compression の DDMP4G への実装実験に携わった、シャープ株式会社 ネットワークシステム LSI 開発センター所長 宮田 宗一 氏、同課長 浦谷 宗宏 氏に、心より感謝の意を表します。

御多忙な中、本研究の指導に貴重な時間を費やして頂き、また研究に際限無きことを御示唆頂いた岩田研究室の 橋本 正和 氏に、感謝の意を表します。

ソフトウェア開発環境向上の研究に携わる立場から、多くの貴重な御意見を賜ると共に、Smart Compression の実装実験に多大なる御援助を頂いた岩田研究室の 三宮 秀次 氏 に、感謝の意を表します。

共にデータ駆動型並列アルゴリズムの研究に携わる立場から、多くの貴重な御意見を賜った岩田研究室の 細美 俊彦 氏、森川 大智 氏、志摩 浩 氏、原田 香織 氏に、感謝の意を表します。

データ駆動型ハードウェアの研究に携わる立場から、多くの貴重な御意見、御協力を賜った岩田研究室の 別役 宣奉 氏、小倉 通寛 氏、長野 光 氏に、感謝の意を表します。

日頃から多くの御意見、御支援を頂いた岩田研究室の方々、荒木 俊介 氏、岩井 秀樹 氏、大石 祐子 氏、西山 直人 氏、宮崎 康徳 氏、山岡 正明 氏に、感謝の意を表します。

参考文献

[1] H. Terada, S. Miyata, and M. Iwata, "DDMP's: self-timed super-pipelined data-driven multimedia processors," *Proc. of IEEE*, 87(2), 282–296 (1999).

[2] Edward J. Delp, and O. Robert Mitchell, "Image Compression using Block Truncation Coding," *IEEE Trans. on Comm.* , 27(9), 1335–1342 (1979).

[3] β enchmark Labs, Inc. "Smart Compression Specifications Revision 1.0 Part ," (1999).

[4] 三木弼一 編, "MPEG-4 のすべて," 工業調査会 (1999).

[5] 小野定康, 鈴木純司 編, "JPEG/MPEG2 の実現法," オーム社 (1995).

[6] ケイワーク 編, "JPEG 概念から C++による実装まで," ソフトバンクパブリッシング株式会社 (1998).

[7] 別役 宣奉, 岩田 誠, "自己タイミング型パイプライン機構におけるデータ転送制御回路に関する研究," 高知工科大学 情報システム工学科 学士学位論文 (2000).

[8] 橋本 正和, 岩田 誠, "形状適応 DCT のデータ駆動型並列実現法," 高知工科大学 情報システム工学科 学士学位論文 (2000).