2008－ARC－180 (3)
2008／10／20

画像認識アプリケーションを用いた動的切り替え可能なSIMD/MIMD 型プロセッサの MIMD モードの性能評価

野本 祥平　京 昭倫　岡崎 信一郎
NEC システムIPコア研究所 〒211-8666 神奈川県川崎市中原区下沼部1753

画像認識を用いた自動車の安全走行システムをより普及させるため，著者らは，システムの中核となる画像認識プロセッサの研究開発を行なってきている．そして，動的切り替え可能な SIMD/MIMD 型プロセッサ(XC コア)の開発により，高並列 SIMD プロセッサ(SIMD モード)による低コスト・高性能・低消費電力と，MIMD プロセッサ(MIMD モード)による柔軟な並列性の活用の双方の性質を併せ持つ画像認識プロセッサを開発中である．本稿では，XC コアに追加した MIMD モードの有効性を評価するため，XC コアに一般道路白線検出アルゴリズムを実装し，その性能評価を行なった．その結果，SIMD モードでの実行に適さない，部分白線候補検定処理を MIMD モードで実行することにより，白線検出アルゴリズム全体のリアルタイム処理(33ms 以内)が可能となること，および画像における ROI(Region Of Interest)領域のデータ転送を高速化する ROI 転送命令を MIMD モードで利用することにより，部分白線候補検定処理を，さらに約 5 倍程度高速化できることを確認した．また，部分白線候補検定処理に対し，MIMD モードでの利用 PU 数を 2 から最大 32 まで変化させたところ，処理が持つ並列性の不足が原因で，16PU 以上では性能向上が鈍化するものの，32PU で約 12.6 倍の高速化を実現できることを確認した．これらにより，高並列 SIMD プロセッサに MIMD モードを追加することで，画像認識アプリケーションの処理性能を大きく改善できることが分かった．

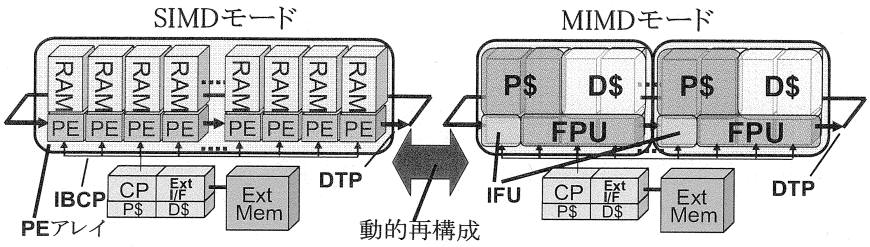
Performance Evaluation of the MIMD mode of a Dynamically Switchable SIMD/MIMD Processor by using an Image Recognition Application

Shohei NOMOTO　Shorin KYO　Shinichiro OKAZAKI
NEC System IP Core Research Laboratories, 1753 Shimonumabe Nakahara Kawasaki, Kanagawa 211-8666, Japan

To promote the expanded use of Safety systems that use image recognition in automobiles, the authors have focused on research and development of an image recognition processor that is a core technology of such systems. The authors are now working on the development of a processor calls a XC core, that achieves low cost, high performance, and low power consumption through the use of a highly parallel SIMD architecture (the SIMD mode), while achieves flexibility through morphing in to a MIMD architecture (the MIMD mode). In this paper, a robust white line detection algorithm for open roads, is implemented in the XC core, and its performance is evaluated to show the effectiveness of the MIMD mode of the XC core. The evaluation shows that the real-time processing(less than 33ms) of the robust white line detection can be achieved, by using MIMD mode to execute the verification process of white line segments, which is a part of the white line detection algorithm, and is not suitable to be executed by a pure SIMD processor. Moreover a 5x speedup of the verification process of white line segments can be achieved by using region of interest(ROI) transfer instructions which can transfer the ROI of an image efficiently, is also shown. Furthermore, each of the execution time in the MIMD mode, according to the number of PUs been used, from 2 up to 32PUs, are also measured. The measured results show that the increase in performance slow down when using more than 16PUs, mainly due to the insufficiency of parallelism in the verification process of white line segments, however, achieved a 12.6x speedup against the SIMD mode, by using 32PUs. The results demonstrate that the performance of image recognition applications can be significantly improved by adding the MIMD mode to a highly parallel SIMD processor.

1. はじめに

少子高齢化社会を迎えるにあたり，誰もが安心して使える，より安全な車システムの実現が求められている．こうした中で，カメラ画像を用いた自動車の安全走行システムの実用化が進んでいる[1][2]．これらのシステムでは，リアルタイムな画像処理・画像認識が必要となるため，高い演算性能が要求される．同時に，車載環境の厳しい熱設計をクリアする低消費電力性と，多様な画像認識アルゴリズムに対応し，開発期間短縮や保守性を向上させるための高いプログラマビリティも要求される．これらの要求を満足するため，これまでに，様々な画像認識プロセッサが開発されてきた．例えばVchip(Vision/Video Chip)[3]は，画像認識の一般的な処理(エッジフィルタ等)を専用ハードウェアとして実装している．専用ハードウェア化により，多様なアルゴリズムに対応する柔軟性を失う代わりに，想定済みの定型処理に対しては，高い演算性能と低消費電力を低コストに実現できる．また Visconti(Vision based Sensing, CONTrol, and Intelligence)[4]は，3個の 3Way-VLIW コアを持ち，各コアは 1 サイクルに，1 つのスカラー命令と，2つの SIMD 命令(8Way×8Bit 幅)を同時に実行できる．専用ハードウェアに比べピーク性能はやや悪化するが，マルチコア・VLIW・SIMD 命令により様々な粒度の並列性を活用できるため，多様な画像認識アルゴリズムに柔軟に対応し，効率よく処理できる．



※PE = Processing Element, CP = Control Processor
IBCP = Instruction Broad Cast Path, DTP = Data Transfer Path
IFU = Instruction Fetch Unit, P$ = Instruction Cache, D$ = Data Cache
FPU = Floating Point Unit, Ext I/F = External Memory Interface

図1. 動的切り替え可能なSIMD/MIMD型プロセッサの全体構成

こうした中で，我々は，画像認識プロセッサIMAP(Integrated Memory Array Processor)[5]の研究開発を行ってきた．IMAPでは，メモリと密結合したPE(Processing Element)を多数配置する高並列なSIMD型アーキテクチャを採用し，全PEに同一の動作をさせることにより，制御に要する回路規模を抑制している．その分，演算に用いる回路規模(PE数)を増やせるため，プロセッサ全体では優れたコスト性能比を実現することが可能となっている．

一方で，近年，画像認識アルゴリズムの多様化・複雑化[6]に伴い，活用できる並列性の種類が限定されるSIMD型アーキテクチャでは，必ずしもこれらの画像認識アルゴリズム全体を効率よく処理できないという課題が出てきている．このため我々は，上記の課題を解決すべく，動的切り替え可能なSIMD/MIMD型プロセッサ(XCコア)[7]の開発を行ってきた．XCコアでは，これまでのSIMD型アーキテクチャ(SIMDモード)に加え，図1に示すように，4つのPEにより1つの独立に動作するプロセッサ(PU)を再構成するMIMDモードを実装している．個々のPUが独立に動作可能となるため，これまでのIMAPが苦手としてきた，画像の部分領域毎に処理時間が大きく異なるアルゴリズム(SIMD型アーキテクチャで実装すると，最長処理時間に全体の処理時間が律速される)や，タスクレベル並列処理を要するアルゴリズムに対しても柔軟に対応することができる．

本稿では，画像認識アプリケーションの1つである一般道路白線検出アルゴリズムのXCコアへの実装を通じて，XCコアのMIMDモードの有効性を検証する．以下，2章では，XCコアのアーキテクチャ概要を述べる．3章では，一般道路白線検出アルゴリズムの詳細を説明する．4章では，当該アルゴリズムをXCコアに実装し，MIMDモードを利用する場合としない場合とで処理サイクル数を比較する．また，MIMDモードで利用するPU数を変化させた際の性能特性を評価する．最後に，本稿のまとめと今後の予定を5章で述べる．

2. XCコア・アーキテクチャの概要

本章では，動的切り替え可能なSIMD/MIMD型プロセッサ(XCコア)のアーキテクチャの概要を説明する．まず，XCコアの基本構成と，動的切り替え可能なSIMD/MIMDモードの実装方法について述べる．次に，MIMDモードにおいて，個々に独立に動作するプロセッサ(PU)での画像処理を効率化するため，各PUに実装しているデータ転送命令(ROI転送命令)について説明する．

2.1. XCコアの基本構成とSIMD/MIMDモード

図1に示すように，XCコアは，1)1つの制御プロセッサCP(Control Processor)と，2)メモリ(RAM)と密結合したPE(Processing Element)をリング状に多数結合したPEアレイとで構成される．まず，CPとPEの基本構成について述べる．

・CP(Control Processor)

CPは独自のプログラム・データキャッシュを有し，XCコア全体の制御や，SIMDモード時には各PEへの命令発行を，MIMDモード時には各PUの状態管理(起動/停止/終了)を行なう．1サイクルに最大6命令を同時発行可能な命令フェッチ部を持ち，そのうち最大6命令をCP向けに，最大5命令をPE向けに発行できる．また，最大6命令を，同時実行可能なVLIW型の命令実行機構を持つ．

・PE(Processing Element)

PEはスクラッチパッドとして利用可能なメモリ(RAM)を持ち，CPより供給される最大5命令を同時実行可能なVLIW型の命令実行機構を持つ．また，各PEはPE間データ転送パス(DTP)を介して接続され，隣接するPE同士でのデータ交換や，外部メモリ(Ext Mem)と各PEが持つメモリ間でデータのバースト転送(ライン転送)を行なう．

次に，MIMDモードにおいて，個々に独立に動作するプロセッサ(PU)の実装方法について述べる．XCコアでは，MIMDモード実現に要するハードウェアコストを最小減に抑えるため，4つのPEの回路資源を流用することで，1つのPUを実現している．回路資源の主な流用方法は，以下の3点にまとめられる．

1)PEのメモリ・レジスタファイルを流用したキャッシュ機構．SIMDモード時にスクラッチパッドとして利用しているPEのメモリ(RAM)を，MIMDモード時には，PUのプログラムキャッシュ(P$)，データキャッシュ(D$)のキャッシュメモリ部として流用する．また，PEのレジスタファイルを，PUのプログラムキャッシュとデータキャッシュのキャッシュタグ部として流用している．

2)SIMD/MIMDモードにおける制御・データパスの共有化．PEとPUが実行する命令セットの大部分(約70%)を共通化し，PEが持つ命令デコーダ部/レジスタファイル部/命令実行部を，PUでも利用可能にしている．

3)PE演算器群を流用した浮動小数点パス．PUのデータパスとして利用していない3PE分の命令実行部の流用により，単精度の浮動小数点パスを各PUに実装している．こうした独自の回路資源の流用手法により，MIMDモードを実装するのに要した回路規模を，XCコ

ア全体の約 10%程度に抑制している.

2. 2. ROI(Region Of Interest)転送命令

これまで SIMD 型アーキテクチャは，処理対象の各部分領域のサイズや内容が異なる種類の画像認識アルゴリズムを不得意としてきた．これは，各部分領域に同一の処理を施しても処理時間が相異なるため，SIMD 型アーキテクチャでは，全体の処理時間が最長処理時間に律速されてしまう為である．このような処理の一例が，図2に示すような，人物候補領域(ROI: Region Of Interest)の検定処理である．このような処理では，通常の SIMD 型アーキテクチャでは，PE アレイ全体を利用しての有効な演算を行なうことが難しい．XC コアでは，新たに実装した MIMD モードを用いて，それぞれの人物候補領域(ROI)を各 PU で独立に処理することにより，こうした画像認識アルゴリズムを効率的に処理することが出来る．

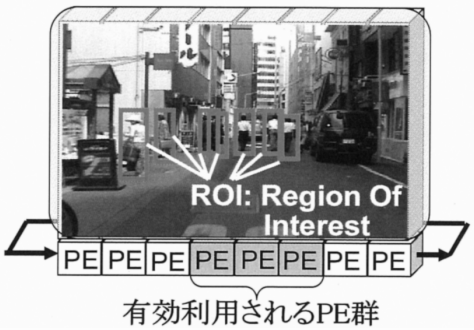


図2. 人物候補領域(ROI)の検定処理

しかし，こうした ROI 領域を処理対象としたアルゴリズムは，通常のキャッシュシステムとの相性が悪く，データ転送の非効率性により，その処理性能が律速される可能性が高い．図3に示すように，通常のキャッシュシステムでは，データ転送をキャッシュライン単位(図3の CL*)に行なうため，転送したい ROI 領域(図3の d0～d4)以外の不要なデータも転送してしまう．これにより，キャッシュと外部メモリ間の帯域が無駄に消費されてしまう．また，不要なデータもキャッシュに格納してしまうため，キャッシュの利用効率も大きく低下してしまう．

特に，MIMD モードにおける PU のキャッシュシステムは，PE のレジスタファイルを流用し，PU のキャッシュタグとして利用しているため，十分な数のキャッシュライン数を確保できているとは言えず，1キャッシュラインのサイズが 512Bytes と大きい．このため，不要なデータがキャッシュに格納される割合はさらに高くなることが予想される．

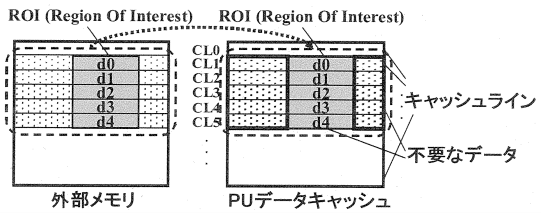


図 3. ROI 領域の転送(通常のキャッシュ)

このため，XC コアでは，ROI 領域のみを，キャッシュへ転送する ROI 転送命令(roiread/roiwrite)を実装することにより，前述のデータ転送の非効率性の影響を最小限に留める工夫を行なっている．図4に示すように，ROI 転送命令では，キャッシュライン単位のデータ転送を行わずに，ROI 領域(図3の d0～d4)のみをキャッシュへ転送すると共に，転送されたデータをキャッシュに連続的に格納することにより，メモリ帯域の無駄を低減すると共に，キャッシュの利用効率の向上を実現している．

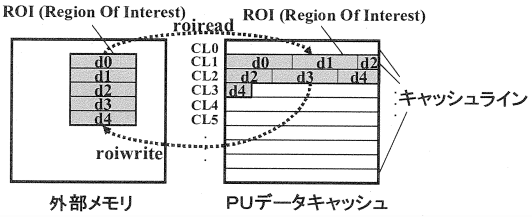


図4. ROI 領域の転送(ROI 転送命令)

3. 一般道路白線検出アルゴリズム

ここでは，XC コアの有効性を検証するために用いる一般道路白線検出アルゴリズムの全体構成と，当該アルゴリズムの中で，SIMD 型アーキテクチャが不得意とする，処理対象毎の処理時間が異なる，部分白線候補の検定処理について説明する．

3. 1. 一般道路白線検出アルゴリズムの概要

評価に用いる一般道路の白線検出アルゴリズムの全体構成について述べる．図5に白線検出アルゴリズムのフローチャートを，図7に当該アルゴリズムの処理結果画像を示す．

図5に示すように，白線検出アルゴリズムは，主に以下の7つの処理から構成される．(1)入力された道路画像(図7－a)に対して，白線特徴フィルタをかけ白線特徴量を算出する(図7－b)．(2)算出した白線特徴量の平均を閾値として，白線特徴量が閾値以上となる点を白線候補点として抽出する．次に，隣接する白線候補点群をグループ化し，グループを規定値長で縦方向に分割し，部分白線候補群を生成する(図7－c)．(3)得られた部分白線候補ごとに，画面上での位置，傾き，直線らしさを求め，道路交通法で定められた道路形状にそぐわない部分白線候補を除外(検定)する．(図7－d)．(4)検定された部分白線候補の中から，道路交通法で定められた道路形状を満たす位置関係にある部分白線候補のペアを抽出し，画面下部の部分白線から，画面上部の部分白線候補へ向けた単方向の木構造を生成し，ルートとなる部分白線候補から，木構造を辿り，白線候補群を生成する．(5)白線候補を構成する部分白線候補同士の位置関係，角度，白線特徴量から，各白線候補の得点を定義する．(6)白線候補同士の位置関係，前フレームでのレーン幅，白線候補の得点を基に，白線候補群から，自車レーンの左右の白線を選択する(図7－e)．(7)得られた白線位置を用いて，白線位置の時系列推定を行なう．ここでは，時系列推定にカルマンフィルタ[8]を用い，カルマンフィルタから得られた白線推定位置と推定分散を，次フレームの部分白線候補の検定処理，白線候補の得点付けに用いる．

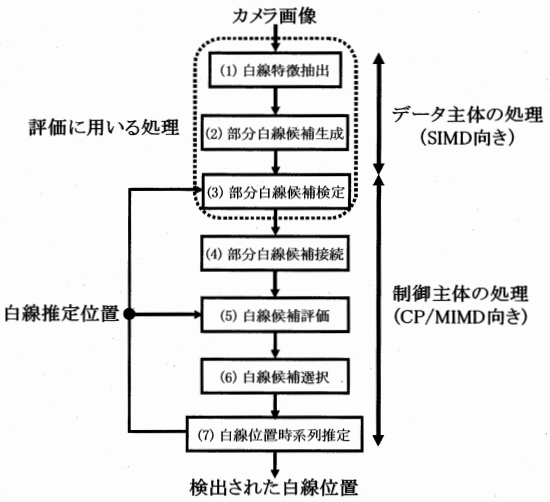


図5. 一般道路白線検出アルゴリズム

これら7つの処理は主に，データ主体の処理と，制御主体の処理の2つに別けられる．処理(1)と(2)は同一の処理を多数のデータに対して行なうため，データ主体の処理に分類でき，SIMD モード向きの処理と言える．一方，処理(3)～(7)は多数の分岐を含む処理から構成され，処理対象毎に処理時間や処理内容が異なるため，制御主体の処理に分類でき，MIMD モードあるいは CP による逐次実行向きの処理と言える．

3. 2. 部分白線候補検定の概要

ここでは，図5の白線検出アルゴリズムのうち，(3)部分白線候補検定の詳細について述べる．当該処理は，制御主体の処理(3)～(7)の実行時間の大部分(約 99%)を占めており，白線検出アルゴリズムにおいて重要な処理となっている．また，処理対象の部分白線候補毎に，処理時間が異なってくるため，SIMD 型アーキテクチャが苦手とする処理である．

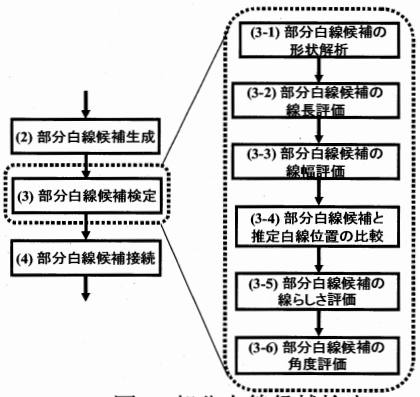


図 6. 部分白線候補検定

図6に示すように，部分白線候補検定は，以下で述べる6つの部分検定処理から構成されている．(3-1)部分白線候補の形状を解析し，その長さや幅，重心位置を求める．(3-2)(3-3) 部分白線候補の長さと幅が，道路交通法で定められた白線形状を満たさない場合には，当該部分白線候補を除外する．(3-4) 部分白線候補の重心位置が，カルマンフィルタより推定された白線位置から大きく離れている場合には，当該部分白線候補を除外する．(3-5) 部分白線候補の線らしさを計算し，線らしさが重心位置に応じて設定した規定値に満たない場合には，当該部分白線候補を除外する．(3-6)カメラの視線方向に対する部分白線候補の角度を計算し，角度と重心位置が，道路交通法で定められた白線形状を満たさない場合には，当該部分白線候補を除外する．

4. XC コアの性能評価実験

ここでは，2章で説明した XC コアに，3章で説明した一般道路白線検出アルゴリズムを実装し，その処理時間を評価する．以下では，まず白線検出アルゴリズムの XC コアへの実装方法を説明し，その後，MIMD モードを用いた場合と用いない場合の性能評価を行い，MIMD モードの有効性について考察する．

4. 1. 一般道路白線検出アルゴリズムの実装法

一般道路白線検出アルゴリズムの XC コアへの実装方法について説明する．まず，今回の評価実験では，図5で示した白線検出アルゴリズムの処理フローのうち，(1)白線特徴抽出，(2)部分白線候補生成，(3)部分白線候補検定を XC コアに実装し，それらの処理時間を測定する．これは，これら3つの処理が，白線検出アルゴリズムの処理時間の大部分(約 99%)を占めており，その傾向を分析することにより，アルゴリズム全体の処理時間の傾向を把握できると判断したからである．

次に，処理(1)～(3)の XC コアへの実装方法について説明する．(1)白線特徴抽出は，カメラ画像(図7－a)の全画素に対して，同一の白線特徴フィルタを施す処理であるため，SIMD モードで実装することとし，カメラ画像の各画素を各 PE に割り当てて並列処理する．

(2)部分白線候補生成は，(1)の処理結果画像(図7－b)の全画素に対して閾値処理を行い，閾値以上の画素全てに対して隣接する閾値以上の画素があるかを調べ，部分白線候補を生成する．同一の処理を多数のデータに対して行なうため，本処理は SIMD モードで実装し，白線特徴量画像(図7－b)の各画素を各 PE に割り当てて並列処理する．

(3)部分白線候補検定は，(2)で生成した部分白線候補ごと(図7－c)に，その形状が道路交通法が定める道路形状に従うかを検定し，従わない部分白線候補を除外する処理である．当該処理を構成する各検定処理(図6)は打ち切り型の処理であるため，部分白線候補の形状特性(大きさ，形，向きなど)によって，その処理時間が大きく異なるという特徴がある．一方で，各部分白線候補への検定処理を，それぞれ独立して行なうことが可能であるという特徴もある．このため，本処理は MIMD モードの PU への実装に適した処理である．そこで，本処理は MIMD モードで実装することとし，各部分白線候補に対する検定処理を各 PU で独立に処理し，各 PU への部分白線候補の割り当て，処理結果の収集を，CP が動的に管理する実装とする．

また，XC コアで新規実装した MIMD モードの有効性を検証するため，(3)部分白線候補検定を SIMD モードにも実装し，MIMD モードの処理結果と比較する．SIMD モードへの実装では，各部分白線候補を各 PE に割り当てて並列処理する実装が考えられる．しかし，

図7－a. 入力画像

図7－b. 白線特徴抽出結果

図7－c. 部分白線候補

図7－d. 部分白線候補の検定結果

図7－e. 白線の検出結果

図7. 白線検出アルゴリズムの実行結果

各 PE へ各部分白線候補のデータを逐次に転送しなければならないこと，およびアルゴリズムが複雑化することから，PE アレイを用いての実装は有効性が低いと判断し，SIMD モードでは，CP によって部分白線候補を逐次に処理する実装を選択する．

4．2．性能評価結果

先に述べた実装方法に従い，一般道路白線検出アルゴリズムを XC コアの SIMD モードと MIMD モードで実装し，XC コアのサイクルベース・シミュレータを用いて，テスト画像(VGA サイズ)に対する処理時間をそれぞれ測定し，比較する．また，白線検出アルゴリズムのうち，部分白線候補検定では，部分白線候補領域(ROI：Region Of Interest)を対象とした処理を行なうため，2 章で説明した ROI 転送命令により，効率的な処理を実現できる可能性がある．このため，MIMD モードでは，ROI 転送命令を用いる場合と，用いない場合の 2 通りを実装し，それぞれの処理時間を比較する．表 1 に，本評価実験で用いる XC コアの性能諸元を示す．

表1. XCコアの性能諸元

	SIMDモード	MIMDモード
PE(PU)数	128PE	32PU
命令供給	命令キャッシュ (32KB, 2way-64entry) @CP	命令キャッシュ (8KB, 2way-8entry) @PU
データメモリ	データキャッシュ (4KB, 2way-16entry) @CP スクラッチパッド (4KB) @PE	データキャッシュ (8KB, 2way-8entry) @PU
データパス	5way-VLIW	3way-VLIW
浮動小数	×	○
外部メモリ	容量：256MB，バンド幅：13.8Gbps	
動作周波数	108MHz @ 90nm CMOS	

図8に，XC コアに実装した白線検出アルゴリズムにおける，テスト画像1枚あたりの処理時間を示す．図8が示すように，SIMD/MIMD モードを適宜切り替える実装により，白線検出アルゴリズムをリアルタイム(33ms 以内)に処理することが可能となっている．これは，SIMD モードのみでは 16.6ms を要していた部分白線候補検定が，MIMD モードを利用することにより，6.47ms(ROI 転送無し)，1.25ms(ROI 転送有り)と高速に処理可能となったためであり，XC コアに実装した MIMD モードの効果は大きいと言える．また，ROI 転送命令の利用の有無により，部分白線候補検定の処理時間を大幅(約 1/5)に短縮できており，ROI 転送命令の効果も高いことが分かる．

さらに，SIMD/MIMD モードを適宜切り替える実装では，部分白線候補検定に用いる PU 数を変化させた際の性能特性を評価する．PU 数が 2PU，4PU，8PU，16PU，32PU の場合に，それぞれの処理時間を測定した．図9に，XC コアの SIMD モードに対する MIMD モードの速度向上比を，ROI 転送命令を使う場合と，使わない場合について，利用する PU 数毎にまとめる．また，外部メモリと PU 間のデータ転送競合が処理性能へ与える影響を調べるため，外部メモリのレイテンシを 0 とした場合の結果も示す．

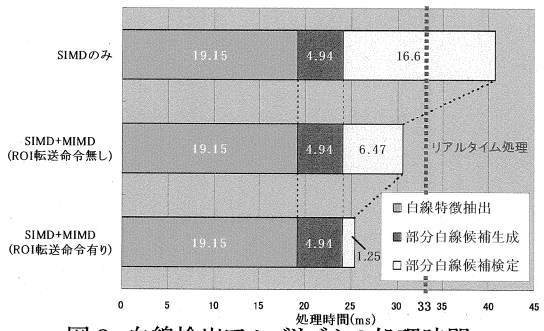


図 8. 白線検出アルゴリズムの処理時間

図9が示すように，MIMD モードを利用することにより，SIMD モードを利用する場合に比べ，大幅な性能向上(16PU で約 10.2 倍，32PU で約 12.6 倍)を実現している．また，ROI 転送命令を使うことにより，大幅な性能向上(最大で約 5 倍)が実現されている．これは，ROI 転送命令により，利用する PU 数が増加した場合でも，データ転送の競合による性能低下を大幅に抑制できた為と考えられる．

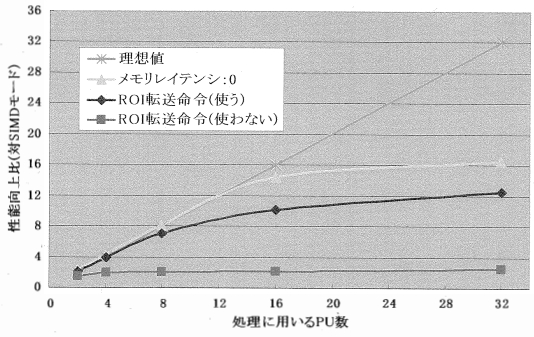


図9. MIMD モードによる性能向上比

一方で, ROI 転送命令を利用する場合でも, 利用する PU 数が 16, 32 と多くなると, その性能向上が鈍化している. 以下では, その原因分析を行なう. まず, 原因としては, 利用する PU 数の増加により, 外部メモリと PU 間のデータ転送の帯域が逼迫している可能性が考えられる. しかし, 図9が示すように, 外部メモリのレイテンシを 0 とした場合でも, 利用する PU 数が 16 より多くなると, その性能向上比は理想値から乖離し始める. このことから, 別の要因が考えられ, 表2に示す各 PU での処理サイクル数の詳細な内訳を分析することにした. 表2が示すように, CP からの処理割り当て待ちに要するサイクル数が, 利用する PU 数の増加に従い増えている. 特に, 利用する PU 数が 32 の場合には, 処理割り当て待ちが PU 毎の全処理サイクルに占める割合が 17.75%に達している. このことから, 利用する PU 数が 16, 32 と多くなるにつれて, 速度向上が鈍化する要因について, 以下の2つの可能性が考えられる. (1) 全 PU を有効活用するのに十分な処理すべき部分白線候補はあるが, それらを各 PU に割り当てるのに十分な性能が CP にない. (2)CP の割り当て性能は十分であるが, 全 PU を有効活用するのに十分な処理すべき部分白線候補がない.

表2. PUの処理サイクル数の内訳

	2PU	4PU	8PU	16PU	32PU
P$ミスストール	6.00%	5.75%	4.75%	3.38%	2.16%
D$ミスストール	0.00%	0.00%	0.00%	0.00%	0.00%
処理割り当て待ち	2.50%	4.00%	6.88%	10.88%	17.75%
プログラム本体	91.50%	90.25%	88.38%	85.75%	80.09%

この2つの仮説を検証するため, CP が部分白線候補を割り当てる処理が, CP 全体の処理サイクル数に占める割合を, 利用する PU 数毎に, 求めた結果を表3に示す. 表3が示すように, 利用する PU 数が増加するに従って, 割り当て処理が占める割合は増加する傾向にある. しかし, 利用する PU 数が 32 であっても, 割り当て処理が占める割合は, CP 全体の処理サイクル数の高々16%に過ぎない. このことから, CP の処理割り当て性能は十分であると考えられる. 以上から, 利用する PU 数が 16, 32 と多くなるにつれて, その性能向上が鈍化したのは, 全 PU を有効活用するのに十分な処理すべき部分白線候補がないことが原因であると考えられる.

表3. 割り当て処理が占める割合

	2PUs	4PUs	8PUs	16PUs	32PUs
割り当て処理	2.0%	3.0%	7.0%	11.0%	16.0%

5. まとめと今後の予定

動的切り替え可能な SIMD/MIMD 型プロセッサ(XC コア)に, 一般道白線検出アルゴリズムを実装し, XC コアの性能評価を行なった. アルゴリズム全体の内, SIMD 型アーキテクチャが不得意とする部分の処理(部分白線候補検定)を, XC コアの MIMD モードで実装することにより, 処理全体をリアルタイム(33ms 以内)に実現できることが分かった. また, 画像における ROI(Region Of Interest)領域のデータ転送を高速化する ROI 転送命令を, MIMD モードにおける各 PU で利用することにより, 部分白線候補検定を, ROI 転送命令を利用しない場合に比べ, 最大で約 5 倍高速に処理できることを確認した. さらには, 部分白線候補検定に用いる MIMD モードの PU 数を 2PU, 4PU, 8PU, 16PU, 32PU と変化させて, それぞれの処理時間を測定した. その結果, 8PU まではスケーラブルな性能向上が実現でき, 16PU 利用時で約 10.2 倍, 32PU 利用時で約 12.6 倍の高速化を実現できることを確認した. また, 利用する PU 数が 16, 32 と多くなるにつれて, 性能向上が鈍化する原因を調査したところ, 白線検出アルゴリズムおいては, 全 PU を有効活用するのに十分な処理が無いことが原因であることを確認した.

今後は, 一般道白線検出アルゴリズム以外の様々な画像認識アルゴリズムについても, XC コアへの実装を行い, XC コアのアーキテクチャ検証を進めていく予定である. その他, XC コアには SIMD/MIMD モードのそれぞれを半数の PE で実現する MIXED モードも実装されており, 複数アプリケーションを同時実装する場合などでは MIXED モードの利用も候補として検討していきたい. そしてこれらにより得た知見を基に, より優れた画像認識プロセッサの設計を進めていきたい.

参考文献

[1]Yoshikatsu Kimura, et.,al, Stereo Vision For Obstacle Detection, ITSW, Oct 2006.
[2]Takayuki Tsuji, et.,al, Development of Night-Vision System, Intelligent Transportation Systems, IEEE Transactions, Sep 2002.
[3]Shoji Muramatsu, et.,al, Image Processing Device for Automotive Vision Systems, IEEE IV, Jun, 2002.
[4]Jun Tanabe , et.,al, Visconti:Multi-VLIW Image Recognition Processor based on Configurable Processor, IEEE CICConference, Sep 2003.
[5]Shorin Kyo, et.,al, An integrated memory array processor architecture for embedded image recognition systems, Computers, IEEE Transactions, May 2007.
[6]Ryusuke Miyamoto, et.,al, Pedestrian Recognition Suitable for Night Vision Systems, IJCSNS, Jan 2007.
[7]Shorin Kyo, et.,al, A low-cost mixed-mode parallel processor architecture for embedded systems, ICS, Jun 2007.
[8]Kalman R. E, A New Approach to Linear Filtering and Prediction Problems, Transactions of the ASME, Mar 1960.