SIMD型命令を利用した文書画像処理の高速化

High Speed Document Image Processing Using SIMD Instructions

土屋　美由紀*　　山合　敏文*
Miyuki TSUCHIYA　　Toshifumi YAMAAI

要　旨

文書をスキャンして保存する際に，文字認識(OCR)処理を行って透明テキストを貼り付けることでテキスト検索が可能となるテキスト付きPDFが普及している．このテキスト付きPDFの生成をMFP上で実装するために画像処理の高速化が望まれていた．そこで処理速度のボトルネックとなっていた処理を中心に，ソースコードの最適化と，Intelプロセッサ用のSIMD型命令であるSSEを利用した並列化処理によって高速化を行った．その結果，二値画像での処理時間を従来の51.1%まで縮めることができた.

ABSTRACT

The searchable PDF which enables text retrieval by embedding text information through scanning and succeeding OCR (Optical Charactor Recognition) becomes widespread recently. High speed image processing is required for searchable PDF generation on an MFP (Multi Functional Peripheral). Speed limiting steps in the processes are identified and speed enhanced through implementation optimization and introducing parallel processing using SSE (Streaming SIMD Extensions) available on Intel CPUs. Experiment results show process time reduction up to 51.1% compared to the conventional process on a binary image.

* 研究開発本部 基盤技術研究センター
Core Technology R&D Center, Research and Development Group

1. 背景と目的

リコーでは長年にわたって文書画像処理技術を研究してきた．その中でもOCR(光学式文字認識)技術はスキャンした文書画像の文字を認識するために広く使われている．このOCR技術を応用したものにテキスト付きPDFがある．テキスト付きPDFとは，認識した文字を透明テキストとして，スキャンした文書画像に付加したPDFのことである．こうすることで，利用者はPDF内のテキストを検索することができる．

紙の文書を電子化するにあたり，検索可能な形で保管したいというニーズは非常に高い．さらに，連携PCやサーバを用いる方法に比べて利用者の使い勝手が良いという理由で，MFP単体上での実装が望まれていた．しかし，MFP本体のCPUのみを利用して追加のハードウェア無しでこのテキスト付きPDF作成機能を実現する場合，処理スピードがネックになっていた．

高速なテキスト付きPDF作成機能をMFP単体上で実現できれば，利用者の作業効率が向上するとともに，他のサーバ製品との連携などのドキュメントソリューションに展開することも容易になる．そこで，テキスト付きPDF作成処理の高速化のために，ボトルネックとなっていたOCR処理を中心に改良を行うことを目的とする．高速化の目標値は，平均処理時間の30%の削減と設定する．

2. OCR処理と処理時間の課題

2-1 OCRの処理フロー

一般的なOCR処理のフローについてFig. 1を用いて説明する.[1)2)]

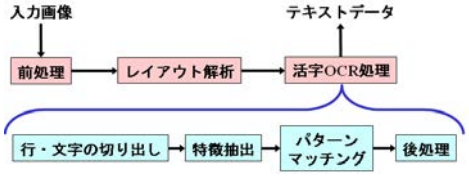

Fig. 1 Processing flow of OCR.

まず，スキャンした文書画像を入力画像として受け取り，画像の回転の補正やノイズ除去などの前処理を行う．次にレイアウト解析を行う．レイアウト解析とは，文書画像中からテキストの段組や表や図のまとまりを分類して抽出する処理であり，この処理によってテキスト領域が抽出される．また，テキスト領域には人が読む際の読み順が自動的に付与される．その後，テキスト領域について，活字OCR処理を行う．活字OCR処理では，まずテキスト領域から1文字分の領域を切り出す．そして1文字分の領域の画像から認識に使用する特徴量データを抽出し，あらかじめパターン辞書として登録されている文字の特徴量データと比較し（パターンマッチング），認識候補となる文字を出力する．その後，後処理で認識結果を言語的な知識を用いて修正を行い，文字列として確定させてテキストデータとして出力する．

2-2 従来版の処理性能解析

処理のボトルネックとなっている箇所を見つけるために，従来版の処理時間のプロファイル測定を行った．二値画像の評価原稿を入力したときのOCR処理のプロファイルをFig. 2に示す．認識対象文字は平仮名・カタカナ・英数字・記号・漢字第一水準・第二水準とし，その他の実行時の各種パラメータは，リコーからドキュメント管理ソフトとして発売しているRidoc Global Scanと同等であるように設定した．

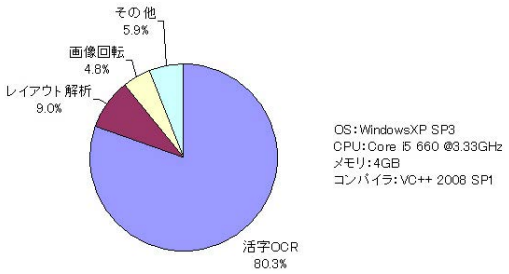

Fig. 2 Profile of the conventional version.

この結果から，最も処理時間のかかっている機能の上位3つは，活字OCR，レイアウト解析，画像回転であることがわかる．

活字OCRは切り出した文字の認識を行う機能であり，画像中の文字数が多ければ呼び出される回数も増加する．レイアウト解析は画像を領域ごとに分割・分類する機能である．画像回転は原稿画像が90度単位や微小角だけ回転してスキャンされた場合に，正しい向きに修正する機能である．その他にはノイズ除去や表処理などの機能が含まれるが，どれも処理時間への寄与率は低かった．

二値画像の処理においては，活字OCRの処理が80%以上を占めていた．個別の画像では，表が含まれている画像では表処理用の機能の寄与率が大きいという傾向が見られたが，どの画像でも上位3つはほぼ上記の3機能であった．

3. 高速化開発

3-1 高速化開発の方針

プログラムを高速化するには，その対象となるプログラム言語とハードウェアを理解したうえで方針を決める必要がある．今回対象とする文書画像処理では言語にC/C++を使用している．C/C++をベースにした一般的な高速化手法を大別すると，次のようにMFPの本体CPU以外のハードウェアを利用する手法と，本体CPUだけで処理を行う手法に分かれる．以下それぞれの特徴について述べる.[3) 4) 5)]

(1) 本体CPUと異なるハードウェアを利用する手法
  (1A)プログラムのASIC化
  (1B)追加ハードウエア(GPGPUなど)の利用
(2) 本体CPUのみで対応する手法
  (2A)CPUの拡張命令となるSIMD型命令の利用
  (2B)スレッド並列処理の利用
  (2C)ソースコードの最適化

(1A)は専用のASICを作るため，最も高速化が期待できる．しかし，開発期間が他に比べて長いこと，改良などのアップデートがしにくいことがデメリットである．

(1B)の例であるGPGPUは近年普及してきており，OpenCL™などの利用によってある程度汎用性の高いソースコードで実現できる．しかし処理によってはCPUとGPGPUの間のメモリ転送がボトルネックとなることや，ハードウェア自体のコストがかかることがデメリットである．また，本体CPUよりも高速のCPUを用いて処理する手法はコスト面がデメリットになる．

(2A)SIMDとはsingle instruction multiple dataの略で，1回の命令で複数データに対する処理を行うものであり，大量のデータに似たような処理を行う使い方で性能を発揮する．

x86系CPUではSSE，ARM®ではNEON™というSIMD型拡張命令セットがあり，メインCPUのみでSIMD型の並列命令を実現できる．しかしそれぞれのCPU専用のソースコードは他のCPUには適用できず，ポータビリティが損なわれるというデメリットがある．

(2B)近年のCPUはマルチコア化が進み，スレッド並列処理が盛んになってきている．複数のCPU(コア)が使用できる環境で容易に高速化できるのがメリットである．逆に1CPU(コア)しか使えない環境では高速化の効果がないことと，スレッドプログラミングを必要とすることがデメリットである．

(2C)入出力仕様を変更せず内部を最適化していくリファクタリングなどもこれに含まれる．ポータビリティを損なわずに高速化を実現できるのがメリットである．しかし高速化プログラミングのノウハウが必要であり，また対象の完成度によってはさらなる高速化は望めないこともある．

今回の開発方針は，追加ハードウェアを用いないこと，シングルCPUでの高速化が求められること，メインのプラットフォームはx86系CPUであること，文字認識結果への影響を最小限にすること，の4点を重視して検討を行った結果，(2A)x86系SSE命令の利用，(2C)ソースコードの最適化　の2つの手法を用いて高速化を実現することとした．

対象に関しては，2-2節で記した処理時間解析により，処理の多くを占めていた活字OCR，レイアウト解析，画像回転の3つの機能に対して重点的に高速化を行うという方針で開発を進めることにした．

以下の節で2つの高速化手法それぞれについて主要な改良点を述べる．

3-2　ソースコードの最適化による高速化開発

ソースコードの最適化は一般的にリファクタリングと呼ばれる，プログラムの入出力仕様を変えずに内部の冗長な処理を省くなどの整理方法を実施した．以下に今回採用した具体的な高速化手法を列挙する．

- 浮動小数点演算の固定小数点化
- 整数除算のシフト演算化
- 大容量メモリ配列について，確保・解放タイミングの再考と総回数の削減
- 画像データの不要な参照・複製の廃止
- 画像中の同じ画素を複数回参照する場合に，CPUのキャッシュサイズを考慮したブロック単位の処理にすることによるキャッシュの有効活用
- 文書画像には一般的に白背景が多いことを利用した，画素のAND処理などでのスキップ処理
- 汎用パラメータ入力の関数の固定パラメータ入力の専用関数への変更

いずれも一般的に知られている高速化手法であり特殊なものではないが，高速化開発においては後述のように一定の成果が得られた．

3-3　SSE命令の利用による高速化開発

3-3-1　SSE命令を用いた高速化手法

SSEとはStreaming SIMD Extensionsの略で，インテルのCPUの拡張命令セットであり，複数の浮動小数点演算を並列に実行できる機能などがある．SSE命令を用いた高速化は，並列処理が可能な部分に適用すると特に効果が高いと言われており，並列処理が可能なことが多い画像処理分野に適するものだと考えられる．SSEは他に特殊なデバイスを必要とせず，またSSEによる並列化処理のノウハウを蓄積すれば他のプラットフォームにも応用しやすいという利点もある．今回の開発では，MFPだけでなく汎用ソフトへの展開を見据え，2000年以降に発売された多くのx86系CPUで使用できるSSE2までの命令を利用することとした．また，SSE命令を利用した実装は，Intrinsicと呼ばれる組み込み関数を利用する方法と，アセンブラで記述する方法がある．我々の文書画像処理技術はマルチプラットフォームでの動作が求められるため，今回の開発では，プラットフォーム間でソースの互換性がある程度確保できるIntrinsicを採用した．

x86系CPUにおいてSSE命令を使用することで，128bitレジスタを8本使用することができる．例えば8bit変数なら1本のレジスタに16個，32bit変数なら1本のレジスタに4個の変数を格納することができ，これらのデータに対し同一の命令で同時に処理することができる．Fig. 3は32bit変数同士の足し算をSISDとSIMDで行ったときの比較である．4組の32bit変数の足し算を行う場合，SISDでは4命令必要だが，SIMDでは1命令で処理をすることができる．このようなレジスタとSSE命令を利用して，演算結果に依存関係のない部分の並列化を行い，処理速度の向上を図った．

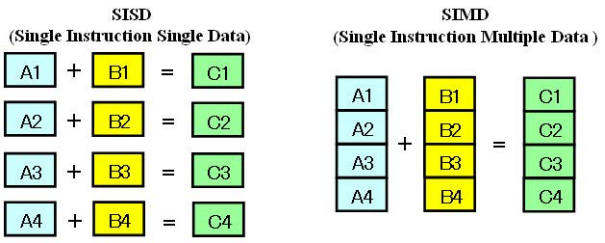

Fig. 3　Example of SIMD.

3-3-2　活字OCRの高速化

活字OCR処理に対してさらに細かく処理時間のプロファイルを測定したところ，処理速度のボトルネックになっている部分はパターンマッチング処理であるということがわかった．活字OCR処理のうち，パターンマッチング処理が占める割合は27.0%であった．

パターンマッチング処理では，1文字に切り出された画像から特徴抽出を行い，その特徴量データと，あらかじめパターン辞書として登録されている文字の特徴量データとの距離計算を行って，最も特徴量間の距離が近い文字を探索する．パターンマッチングの処理フローと辞書データについてFig. 4に示す．

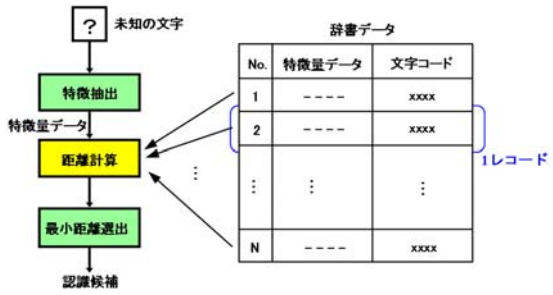

Fig. 4　Pattern dictionary and pattern matching.

パターン辞書には認識候補となるすべての文字（例えば，平仮名・カタカナ・英数字・記号・漢字第一水準すべて）が，特徴量と文字コードのセットという形で登録されている．1文字に対応する特徴量と文字コードのセットが1つのレコードとなっており，認識候補文字数分のレコードが登録されている．通常，日本語の文書に対してOCR処理を行う場合，レコード数は数千に及ぶ．認識処理では未知の入力文字の特徴量に対して，辞書に登録されているすべての文字の特徴量との距離を計算するため，多くの処理時間がかかっていた．

未知の文字と辞書に登録されている文字の距離計算の方法をFig. 5に示す．1文字の特徴量は64次元のベクトルとなっており，ベクトルの各要素は8bitで構成される．距離計算では，この2つの64次元ベクトルのユークリッド距離を求める．各要素の差を2乗したものを64個分加算したものが距離となる．

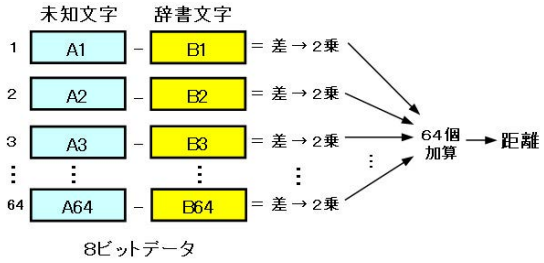

Fig. 5　Distance calculation of features.

ここで，「差を取って2乗する」という演算が64個分繰り返されるが，個々の演算に依存関係はないため，並列処理が可能だと考えられる．そこでこの距離計算にSSE命令を用いることで，高速化を実現した．具体的にはFig. 6のように，128bitレジスタに16bit×8個のデータを格納し，差を取って2乗するという演算を8並列で行った．これを8回繰り返すことによって64次元ベクトルデータの演算が可能となる．

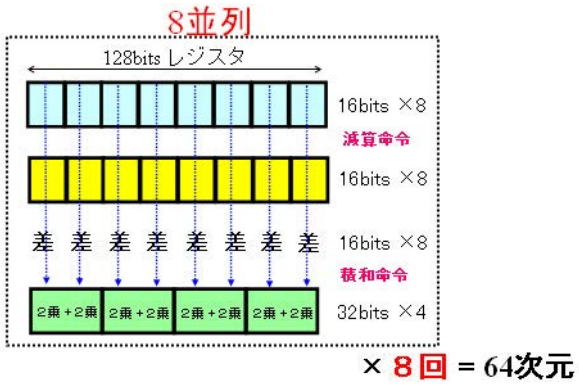

Fig. 6　Distance calculation using SSE.

その他にもパターンマッチング処理を除いた上位2関数も含めた全3関数に対してSSE命令を利用した高速化を実施した．その結果，活字OCR関数単体での処理時間が従来の55.0%に短縮できた．

3-3-3　画像回転・その他の機能の高速化

90度単位の画像回転について，SSE命令の利用による高速化を行った．128bitずつ並列処理が行えるため，1bit画像の回転処理では128画素を一度に演算することができる．この結果，90度回転では従来の44.5%，270度回転では従来の45.8%まで処理時間を短縮することができた．180度回転については元々の処理がある程度高速だったこともあり，従来の95.1%の処理時間に留まった．

その他，OCR処理への寄与率はそれほど高くないものの，並列処理の効果が高いと考えられる関数に対してSSE命令を用いた高速化実装を行った．具体的にはフィルタ処理，圧縮処理，白黒反転処理などである．

4. 結果

今までに述べてきた手法で高速化実装を行った後の処理時間測定結果をTable 1に，プロファイルをFig. 7に示す.

Table 1　Processing time of OCR.

	平均処理時間	処理時間比率
従来版	2.68 sec	-
高速版	1.37 sec	51.12 %
高速版（SSEなし）	2.51 sec	93.66 %

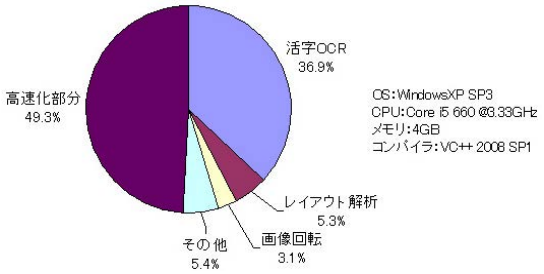

Fig. 7　Profile of the high speed version.

評価に用いた32枚のスキャン画像に対してOCR処理を行い，従来版で85.8秒(1枚平均2.7秒)であった処理時間を43.8秒(1枚平均1.4秒)まで短縮することができた.

Table 1を見ると，今回行った高速化開発により，二値画像で従来の51.12%まで処理時間が削減されたことがわかり，OCR処理の処理時間が大きく改善されたことが確認できた．また，SSEなしでも従来の93.66%の処理時間となっており，ソースコードの最適化による高速化の効果も確認できた.

Fig. 7はFig. 2の処理時間を100%としたときに各メソッドの処理時間がどのくらいになったのかを表している．つまり，Fig. 2の測定時より，全体で49.3%にあたる時間を高速化によって削減できており，例えば活字OCRはFig. 2では80.3%だったのがFig. 7では36.9%となっていることから，これらの画像での活字OCRの処理にかかる時間は従来の36.9/80.3=46.0%になったことがわかる.

また，Fig. 7のプロファイル結果によると，高速化後も処理時間比率が高かった順に活字OCR，レイアウト解析，画像回転となっており，これは従来と変わっていない．しかしFig. 2とFig. 7から各関数を見ると，活字OCR(80.3→36.9)，レイアウト解析(9.0→5.3)，画像回転(4.8→3.1)など，各関数の処理時間が大きく削減できたこともわかる．特に活字OCRは大幅に高速化が実現できている.

5. おわりに

以上のように, SSE命令の利用とソースコードの最適化の2つの手法を用いて高速化開発を行い，処理時間の大幅な短縮を実現した.

今回高速化開発を行ったこれらの機能は, imagio MP C5002/C4002/C3302/C2802シリーズの機能として製品搭載され，追加ハードウェア無しでのテキスト付きPDF作成処理においてトップレベルの処理時間を実現することができた．また今後のMFP製品への展開や，ドキュメント管理システムソフトウエアにおけるOCR処理高速化も順次計画されている．さらに，ここで培ったSSEなどによる高速化のノウハウは他の画像処理モジュールへの展開も可能であり，ドキュメントソリューション製品の競争力強化につなげたい.

参考文献

1) 齋藤高志, 山合敏文, 立川道義：Document Image Segmentation and Layout Analysis (Special Issue on Document Analysis and Recognition), *IEICE transactions on information and systems,* Vol.E77-D, No.7, pp.778-784 (1994) .
2) 伊藤仁志, 大羽成征, 石井信, 山形秀明：多段階手続きによる日本語活字文字認識, 電子情報通信学会技術研究報告 パターン認識・メディア理解, Vol.102, No.707, pp.79-84 (2003) .
3) Steve Oualline, 望月康司, 谷口功：C 実践プログラミング, 第 3 版, オライリー・ジャパン (1998).

4) Intel Corporation: Intel® C++ Compiler XE 12.1 User and Reference Guides, http://software.intel.com/sites/products/documentation/hpc/composerxe/en-us/2011Update/cpp/win/index.htm (参照 2012-08-07).
5) 中田潤也, 北山洋：IA-32 SIMD リファレンスブック 上・下, カットシステム (2007).

注1) Windows はMicrosoft Corporationの登録商標です.
注2) Intel, SSEはIntel Corporationの商標です.
注3) ARM, NEONはARM Limitedの商標です.
注4) OpenCLはApple Inc.の商標です.