5H−5

DSPとCPUによるジェスチャ認識プログラムの作成

林 悠平† 立川 純† 田中 康一郎‡ 有田 五次郎†
† 九州工業大学 情報工学部 知能情報工学科
‡ 九州工業大学 マイクロ化総合技術センター

1 はじめに

人間とコンピュータのコミュニケーションを行う"「棒」入力システムのためのジェスチャ認識システム"[1]がある．このシステムは画像を利用したリアルタイムアプリケーションであり，入力されたRGBデータをコンピュータで処理しやすいYUVデータに変換している．その後，アプリケーション独自の処理が開始される．システム全体の負荷が大きいのに加え，本来の処理以外のデータ変換なども行わなくてはならない．パーソナルコンピュータを用いてCPUだけで処理を行っているため，現在のレスポンスタイムでは満足されていない．

そこでDSPでデータ変換などの画像処理部分を行い，CPU負荷を軽減することにより，システムの高速化を試みることにした．今回すでにパーソナルコンピュータ上で動作しているジェスチャ認識システムの一部をDSPで作成した．本稿では，DSPによる設計を行い高速化を図った結果について報告する．

2 画像処理アプリケーション

このシステムは人間の動作により，コンピュータとコミュニケーションを行うものであるが，処理時間の向上が望まれている．そこでシステム中の入力画像から「棒」領域を抽出する(入力データを変換し，棒の色との距離を求める)処理部分を作成することにした．一連の処理の流れを図2に示し，各処理について詳しく説明する．

まず，式(1)により入力されたRGBデータをYUVデータに変換する．YUVデータはそれぞれ輝度・輝度と青色成分との距離・輝度と赤色成分との距離を表し，データサイズは各々8ビットとした．

$$\begin{cases} y = 0.3 \times r + 0.59 \times g + 0.11 \times b \\ u = 0.5781250 \times (b - y) \\ v = 0.7265625 \times (r - y) \end{cases} \quad (1)$$

次に，領域を抽出するために式(2)により参照データ(ry,ru,rv)と入力データ(ty,tu,tv)との距離を求める．参照データもRGBで与え，同様にYUVデータに変換して距離計算を行う．

$$\begin{cases} t_1 = (ry - 128) - (ty - 128) \\ t_2 = ru - tu \\ t_3 = rv - tv \\ dis = \dfrac{255 \times 10}{\sqrt{(t_1)^2 + (t_2)^2 + (t_3)^2 + 10}} \end{cases} \quad (2)$$

最後に求めた距離情報を縦横方向に投影し，図1に示すようなヒストグラムを生成する．

また，処理が正しく行われているかを確かめるために距離情報をモノクロ画像(pgmフォーマット)として出力する．

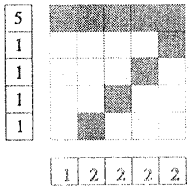

図 1: 距離情報の投影

3 DSPにおける設計

本研究で対象としたのは、RGBデータからヒストグラムを作成するまでである．本来のシステムでは入力はカメラから取り込んだ画像であるが，作成したプログラムではRGBデータを持つ静止画像(ppmフォーマット)をファイルから読み込むことにした．また使用するデータの精度は8ビットなので計算時間を短縮するために，YUV変換の式(1)をそのまま使用するのではなく，整数演算のみで行う式(3)に変形した．

$$\begin{cases} y = (307 \times r + 604 \times g + 113 \times b) \gg 10 \\ u = (592 \times (b - y)) \gg 10 \\ v = (744 \times (r - y)) \gg 10 \end{cases} \quad (3)$$

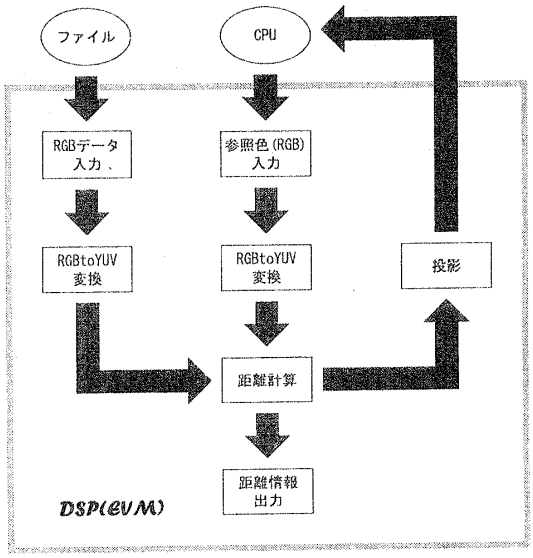

図 2: 処理の流れ

3.1 開発環境

2節で述べたアプリケーションを図3に示すDSP C6701 (TI社 TMS320C6701)[2]を搭載した．Evaluation Module (EVM)上で実行させた．また，EVMの統合開発環境であるCode Composerを用いて，コード開発・デバッグ・実行・処理時間計測を行った．プログラミング言語にはC言語を用い，専用Cコンパイラを使用した．

A Programming Approach for a Gesture Recognition Application with a DSP and a CPU. By Yuhei Hayashi†, Jun Tachikawa†, Koichiro Tanaka‡, Itsujiro Arita† (†Department of Artificial Intelligence, Kyushu Institute of Technology. ‡Center for Microelectronic Systems, Kyushu Institute of Technology., 680-4 Kawazu, Iizuka, Fukuoka 820-8502, Japan)

DSP C6701 の特徴として次のようになものが挙げられる.

- 8命令同時実行可能
- 最大動作周波数 200[MHz]
- 浮動小数演算命令対応

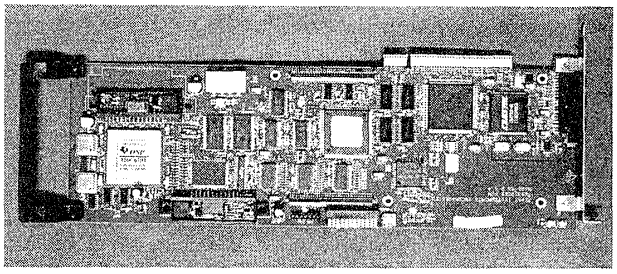

図 3: Evaluation Module (TMS320C6701)

3.2 プログラム作成

DSPで高速な処理を行うためには，CPUで処理する場合の開発とは異なり，TMS320C6701のアーキテクチャを意識したコード生成を行う必要がある．特に今回は，次に示すような点を考慮しながらプログラムを作成した．

1. データ型 乗算ユニットは16ビットの演算をするため，*int* × *int* の演算では3回の乗算を実行しなければならない．よってデータの範囲が16ビットまでならば *short* 型を使用する．
2. レジスタ レジスタは32ビットであるため，long, double型(40,60ビット)の変数を使った演算にはレジスタが2つ必要になる．精度に応じた型の変数を使用する．
3. メモリの依存関係 メモリの依存関係を削減することによって並列に実行可能な命令を増やすことができる．最も簡単な方法としては，関数内で値を書き換えない変数は const 宣言を使用する．
4. その他 整数の割り算には時間がかかるため，2の倍数の演算にはシフトを使用する．

3.3 比較対象

DSP C6701の性能を把握するために，同一のプログラムをパーソナルコンピュータ上で実行し，処理時間の評価を行った．対象としたマイクロプロセッサは，Intel社の PentiumII [450MHz] と PentiumII [233MHz] であり，WindowsNT Workstation 4.0 ServicePack4 上で実行させた．コンパイラにはMicrosoft社の VisualC++ 6.0使用した．

4 結果

2節で述べたアプリケーションを実行させたところ，図4に示すような画像が出力された．640 × 480の画像を入力とし，入力画像の青色バインダに類似した色を参照色として与えた結果である．DSP・CPUそれぞれで処理した結果は同じものであった．

今回作成したプログラムの実行時間を表1に示す．これはシステム全体の処理時間ではなく，画像を入力してからヒストグラムが生成されるまで時間である．DSPとPCで全く同じコードを実行した結果である．DSPの処理時間はサイクル数で計測し，その値と最大動作周波数から処理時間を求めた．CPUでの処理時間はライブラリを用いて直接計測した．

今回の画像処理アプリケーションでは3.2節で述べたこと以外に，関数呼び出しを軽減してプログラムを作成した．当初はYUV変換・距離計算・投影を各々計算していたが，画素数が非常に多いため1画素ごとにRGBから距離までを一度に求めるようにした．この変更により関数呼び出しにかかる時間を軽減したこと以外に，中間結果を保存する必要がなくなるためメモリアクセスの削減ができその時間も短縮できた．

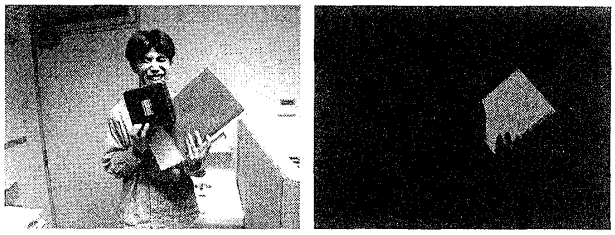

図 4: 実行結果(左:入力画像，右:出力画像)

表1の **Original** は最適化を意識せずに作成したプログラム，**Optimal** は3.2節の内容や関数呼び出しの削減など最適化を意識したプログラムでの実行時間である．

表 1: 処理時間

Processor	Original [sec]	Optimal [sec]
DSP C6701 [200MHz]	1.889	0.752
Pemtium II [233MHz]	0.701	0.379
Pemtium II [450MHz]	0.360	0.185

YUV変換などを独立に処理するのではなく，CPUでジェスチャ認識などの本来の処理を行いながら，外部のDSPでその前処理を同時に行えばシステム全体の処理効率は上がると考えられる．また，今回のアプリケーションがどの程度DSPの能力を引出せているのか知る必要がある．

5 おわりに

DSPによる画像処理アプリケーションの作成およびCPUとの比較を行った．DSPを単独で用いるのではなく，CPUと組み合わせたシステム設計が重要である．今後システムにどのような形でDSPを組み込めば効率がよくなるかを確かめるために，DSPを組み込んだシステムの開発を行う必要がある．DSPの潜在能力を調べれば規模や用途に応じて使い分けが可能となる．また，DSPとCPUだけでなくFPGAのようなハードウエアとソフトウエアを組み合わせたエミュレータボード [3] を使用した設計も検討している．

謝辞

今回設計した画像処理アプリケーションを快くご提供してくださった本学知能情報工学科大橋健氏に感謝します．

参考文献

[1] 大橋健, 中程啓, 吉田隆一, 江島俊朗:「棒」入力システムのためのジェスチャ認識の実現, 情報処理学会論文誌, pp. 567 – 576 (1999).

[2] Texas Instruments, Inc.: *http://www.ti.com/*.

[3] 田中康一郎, 平野孝明, 浅野種正, 有田五次郎：システムLSI時代に向けたFPGAとDSPによるシステム設計教育, in *Proceedings of The Seventh Japanese FPGA/PLD Design Conference & Exhibit*, pp. 53 – 60 (1999).