GPGPU を活用した音響尤度計算の高速化に関する一検討

八木良一　柳原広昌
（株）KDDI 研究所

1．はじめに

近年グラフィックス・ハードウェア（GPU）の分野では，コンピュータ・グラフィックス処理のためのパイプライン化や，複数の並列演算器，MIMD（Multi Insutruction/ Multiple Data），SIMD（Single Instruction/ Multiple Data）命令など，ハイパフォーマンスコンピューティングのための高性能演算機能が組み込まれるなど，特に浮動小数点数の演算性能において，CPU の数十倍の演算能力を実現している．一方，GPU をコンピュータ・グラフィックス以外の汎用演算に利用する GPGPU（General Purpose GPU）[1]の研究[2]も盛んに進められ，携帯端末用 GPGPU を搭載したプラットフォームも登場するなど，様々な用途，目的に応じた環境も整備されつつある．

そこで本稿では，GPGPU の演算性能，並列性に着目し，GPGPU を使った統計的手法による音声認識処理における音響尤度計算のうち，正規分布からの出力確率の計算部分を高速化するための一手法を提案すると同時に，その有効性の検証結果を報告する．

2．音響尤度計算

(1) 概要

音声認識は，入力された単語系列に対し，その単語系列の並びが発生する確率（言語尤度）と，その単語系列の並びから予測できる音素毎の多次元特徴ベクトルが得られるであろう確率（音響尤度）を求め，それらの組み合わせが最大となる単語と，単語や音素等を単位とする音響特徴が記述された統計的モデル（以降，音響モデル）を連続的に照合し，音声に最も適合する単語系列を認識結果として出力するものである．

音響尤度計算[3]とは，音響モデルに記述された音素毎の特徴量（以降，モデル特徴ベクトル）の出力確率分布と，入力音声から算出した特徴量（以降，入力特徴ベクトル）を照合し，音響モデルから出力される確率を計算する処理である．

(2) 尤度計算式

音響的特徴の時間変化を確率的にモデル化した隠れマルコフモデル（Hidden Markov Model: HMM）で表される音響モデルでの尤度計算は，次の演算式(1)で示される．x は互いに独立する n 次元（n=38）の入力特徴ベクトル，μ はモデル特徴ベクトルの平均，σ はモデル特徴ベクトルの分散である．

$$p(x)=\prod_{n=1}^{38}\frac{1}{\sqrt{2\pi\sigma_n^2}}\exp\left\{\frac{(x_n-\mu_n)^2}{\sigma_n^2}\right\} \qquad \text{演算式（1）}$$

さらに，演算式(1)を対数化し尤度にすると，次の演算式(2)の通り x の 2 次関数の和で表される．

$$\log p(x)=\sum_{n=1}^{38}\left\{-\frac{1}{2}\log\left(2\pi\sigma_n^2\right)+\frac{(x_n-\mu_n)^2}{\sigma_n^2}\right\} \qquad \text{演算式（2）}$$

この演算式(2)の括弧{ }内の第一項は定数項であり，予め計算しておけるため，演算式(2)の括弧{ }内の第二項が実際の音響尤度計算のための演算式となる．さらに，この演算式(2)は，音素毎に用意されている音響モデル内の状態系列の個数と，混合分布数（8～32 程度[3]）分の膨大な演算を実行する必要がある．音声認識の精度向上を図る上では，音響モデルの状態数，混合分布数を増やす必要があり，モバイル端末（例えば ARM9 200MHz : 200MFLOPS）など処理能力の限られた実行環境ではリアルタイム処理が困難となる．例えば，8kHz, 1 サンプルあたり 16bit でサンプリングされた音声データが，10ms 毎に 38 次元の入力特徴ベクトルに変換され，仮に 1 つの状態系列が 500 の状態，混合分布数 8 で構成される音響モデルの場合であれば，1 秒の音声データに対して 100×500×8=400000 回の尤度計算が行われることになる．これは 60M FLOPS 程度の処理に相当し，さらに，音声認識処理全体に占める音響尤度計算の処理比率を 10%と仮定し，処理全体を FLOPS に換算すると，リアルタイム実行には 600M FLOPS 程度の CPU が必要と試算できる．

(3) 一般的な高速化実装手法

音響尤度計算の高速化には，一般的に次のようなアプローチがある．①予め音響尤度計算の演算結果をテーブル化し，それを参照することで計算回数を削減する手法や，②音響尤度計算の演算式を，浮動小数点数の演算や除算を除く演算式に置換，または演算式を分解し，個々の計算コストを削減する手法がある．しかしながら，上記一般的な高速化実装手法を適用したとしても，携帯電話などのリソースの限られたモバイル端末では，CPU の演算処理能力やメモリ容量などに制限があることから，さらなる音響尤度計算の高速化が必要であった．

3．提案手法

前述の問題点への一つのアプローチとして，GPGPU の活用を検討し，今回音声認識処理の中で並列化の比較的容易な音響尤度計算について GPGPU を用いた高速化を提案する．

本提案手法では，三次元グラフィック処理で用いられる画像処理（フラグメントシェーダ）のマトリックス並列演算機能をベクトル処理と見做して，複数並列処理のカスケード接続の処理形態を構成する．さらに，音響尤度計算における並列・順次処理の入出力値を GPGPU 上のテクスチャ（画像データ）を保持するためのメモリ領域へ効率的にマッピングすることで，高速化を図る．図

"A Fast Method for Acoustic Likelihood Calculation Using GPGPU"
Ryoucihi Yagi, Hiromasa Yanagihara.
KDDI R&D Laboratories Inc.

1 に，GPGPU を用いた音響尤度計算処理のシステム構成例を示し，以下に提案手法の処理手順を概説する．

① モデル特徴ベクトルをテクスチャ A のメモリ領域に保存する．
② 入力音声を入力特徴ベクトルに変換し，テクスチャ B のメモリ領域に保存する．
③ テクスチャA，B のメモリ領域に保存されたモデル特徴ベクトル，入力特徴ベクトルを元に，音響尤度計算を並列実行する．
④ ③の結果をテクスチャCのメモリ領域に保存する．
⑤ テクスチャCのメモリ領域に保存された計算結果を元に，音響尤度計算結果のΣ演算を実行し，合計した音響尤度計算結果を得る．

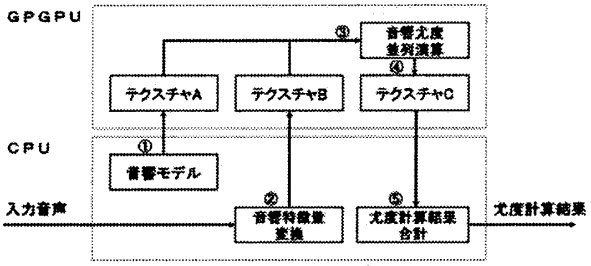

図 1 システム構成例

さらに，図 2 に，並列・順次処理におけるテクスチャの具体的な利用例を示し、以下にマッピングを中心に詳説する．

(1) マッピング
- 入力特徴ベクトルｘ ｎをｍ個分複製し、テクスチャｘの画素（m,n）に配置する．
- モデル特徴ベクトルｙ mnの平均μmn，分散σmnをテクスチャμ，σの画素（m,n）に配置する．

(2) 並列・順次演算
- テクスチャｘ，μの画素に設定されたｘ mn，μmnに対し，演算式(2)第二項（ｘ mn－μmn）の演算を一括実行し，演算結果をｚ mnとする．
- ｚ mn，σmnに対し，演算式(2)第二項（Zmn×Zmn）／（σmn×σmn）の演算を一括実行し，演算結果をＰmnとし，テクスチャｐに保存する．

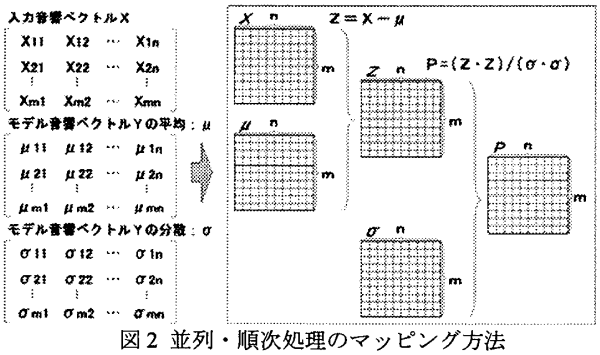

図 2 並列・順次処理のマッピング方法

4. 実験と考察

提案手法の有効性を確認するため，以下条件にて実験を行い，正規分布からの出力確率計算部分の処理時間を，CPU で実行した場合と提案手法で実行した場合とで比較した．なお，実験環境は，CPU：Pentium4 3.8GHz，メモリ：1 GB，GPU： NVIDIA 社製 GeForce 7800 GTX GPU を搭載した ELSA 製 GLADIAC 970 GTX ボードを使用した．

(1) 実験条件

8kHz，16bit でサンプリング（1Frame=10ms）された音声サンプルを 124Frame 入力した．また,音響モデルの状態系列の個数mは 546 個，入力特徴ベクトルの個数ｎは 38 個とし，各テクスチャのメモリ領域は，256×256=65536byte を用いた．124Frame 分の尤度計算処理時間の計測結果を図 3 に示す．なお，テクスチャの生成時間，テクスチャへのマッピング処理時間は含んでいない．

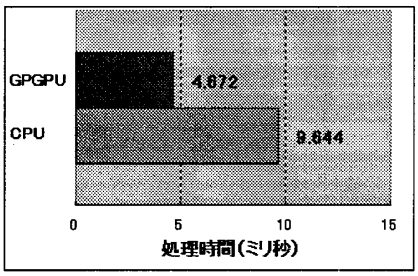

図 3 実験結果（尤度計算処理時間）

図 3 より，提案手法では CPU 単体で実行した場合と比べ，約 1/2 の処理時間で音響尤度計算を実行できることが分かる．従って，音声認識処理全体の処理時間に関しては約 5%の削減効果を期待できる．

5. おわりに

GPGPU を使った統計的手法による音声認識処理における音響尤度計算を高速化するための一手法を提案すると同時に，その有効性の実験を行った．音響尤度計算における並列・順次処理を，テクスチャへ効率的にマッピングすることにより，CPU 単体の実行時と比べ，処理時間を，約 1/2 （全体としては約 95%）に削減できることが確認できた．

今後は，入力特徴ベクトルの複数入力対応や利用テクスチャ数の削減など，さらなる最適化・軽量化を図る．

参考文献

[1] H. Nguyen 編：”GPU Gems 3”, Addison-Wesley, pp.765-907, 2007.

[2] 内之宮 仁志,西尾 孝治,小堀 研一：” GPU を用いたボリューム細分割の高速化に関する研究”,情報処理学会第 69 回全国大会講演論文集, 5Y-1, 第 4 分冊 , pp.245-246, 2007.

[3] 鹿野 清宏 , 河原 達也, 山本 幹雄, 伊藤 克亘, 武田 一哉 , 情報処理学会編：” IT Text 音声認識システム”, オーム社, pp.13-15, 2001.