信号レベル画像処理用ソフトウェアバンクについて

6P-9

藤村 是明、大石 東作
(電子技術総合研究所)

1. はじめに

画像処理の応用範囲の拡大と処理実行環境の多様化とにより、画像処理システムも多様化している。研究段階においては、単一の処理方式を設定し、これに合わせたソフトウェアの形でアルゴリズムを蓄積することも重要である[1][2]が、実用化段階のシステムでは、実行効率向上のためのソフトウェア形態の多様化は必至であり、別の方法が必要になる。

本論文では、多様な画像処理システムを前提としたアルゴリズムの蓄積方法について論ずる。

2. 信号レベル画像処理システムの多様性

対象とする画像処理は、2次元配列としての画像データを入力・出力とする信号レベル画像処理のうち、ラスタ走査的に実行可能なものとする。また、処理のアルゴリズムを、1回のラスタ走査内での処置のアルゴリズム(基本演算アルゴリズム)と、基本演算を組み合せたアルゴリズム(複合演算アルゴリズム)とに分けて考える。

画像処理専用プロセッサを使わない汎用計算機上のシステムに話を限っても、次のような多様化要因がある。

・プロセス型/サブルーチン型
　各基本演算を実行プロセスの形で蓄積し、これを起動するコマンド列として複合演算を実現するのが、プロセス型である。各基本演算をサブルーチンとして蓄積し、これを呼び出すプログラムとして複合演算を実現するのが、サブルーチン型である。
・順次型/並行型
　複合演算の実行に際し、一つの基本演算の完了を待って次の演算実行にかかるのを、順次型と呼ぶ。これに対し、各基本演算の実行を分割し並行動作させるのが、並行型である。

この2つの多様化要因の組み合わせにより、次のような4種の型ができ、それぞれ特長がある。

①プロセス順次型
　会話型システムとしてよく用いられる。
②サブルーチン順次型
　ある規模以上の計算機システムでは、ソフトウェアの生産性・移植性が良く、実行効率も高い。
③プロセス並行型
　UNIX-OSのパイプ機能を用いた実現例がある[3]。
④サブルーチン並行型
　特殊なOS/言語機能を前提としない、小型計算機システムに適した実現例[4]がある。

実際には、OSインタフェース[2]の違い等により、さらに多くの多様化が生じるが、画像処理特有の問題ではないので、ここでは触れない。

3. ソフトウェア・バンク

多様な信号レベル画像処理システムに対し、共通のアルゴリズム蓄積を実現するには、具体的に実行可能なプログラムの蓄積よりは、高度な方法論が必要である。ここでは、汎用ソフトウェア・バンクの方法論を紹介した後、画像処理への応用を示す。

3.1. 汎用ソフトウェア・バンク

筆者の一人は、一般的ソフトウェアの生産性向上のための汎用ソフトウェア・バンクについて研究中である[5]。ここでの基本的な考え方は、

・仕様記述だけからのトップダウン技法で現実的なプログラムが得られることは少ないので、"手続き"を重視する、
・手続きとしての具体的プログラムをそのまま部品として登録するのでは、"帯に短し、タスキに長し"となる場合が多いので、再利用を考慮した形で蓄積しなければならない、

というものである。

今のところは、ソフトウェアの蓄積を、

①通常のプログラムやサブルーチンに相当する、問題や実現上の仕様変更への対応を必要としない、直接実行可能なプログラム・モジュールのバンク
②汎用性のあるアルゴリズム(ソート等)を、何種もの具体化が可能な形で蓄積する、抽象アルゴリズム・バンク
③応用問題の解決手続きの概略を、必要に応じ①・②を利用して記述した、応用システム・プロット・バンク

の三種に分けて行なっている。

3.2. 画像処理ソフトウェア・バンク

汎用ソフトウェア・バンクは現在開発中であり、画像処理にはあまり使われない高度の機能が多く含まれ、大きなシステムとなることから、特殊化・簡略版としての画像処理用ソフトウェア・バンクを考える。ここで一番重要なことは、基本演算、複合演算のそれぞれに対し、多様なシステムへの具体化が容易で、かつ人間にとっても解りやすい

A Software Bank for Signal Level Picture Processing
Koreaki FUJIMURA, Tosaku OISHI
Electrotechnical Laboratory

```
/* ex.1: local average 860122 */
[ @ local_average : 1 > 1 () ]
[ 2: $-2 ][ 2: $-2 ]
        OT = ( A+B+C+D+E+F+G+H+I ) / 9
[ * ][ * ]
        OT = E
```

図１．基本演算バンク用表現形式

表現形式を設定することである。

基本演算のバンク化にあたっては、

・システムにより画素アクセス方法やプログラム制御構造が異なること

への対応が本質的であるが、

・従来の直接実行可能プログラムでは、演算の本質部分より周辺部における例外処理に手間がかかっていたこと

などについても対策をたてると有効性が増す。

基本演算のアルゴリズムの表現形式として、図１のようなものを考えている［６］。これは、３Ｘ３近傍が揃っているところ([2:$-2][2:$-2])では、入力画素の平均値を出力とし、それ以外([*][*])は、中心の値を出力とするプログラムを記述したものである。ここで、A,B,..I,OTは、図２に示すような画素アクセスのマクロであり、対象システムにより、様々な形に展開される。

複合演算のバンク化の際の表現形式としては、図３のようなものを考えている［７］。これは、図４のような演算を矢線とするデータフロー図式を言語化したものである（中間結果を表示(QFVS4D)・格納(QWRT4D)しながら、濃淡画像の線画化を行なう処理である）。現在は、複合演算の構成要素はすべて基本演算となっている。この場合、複合演算バンクも基本演算バンクと同様、汎用ソフトウェア・バンク中の抽象アルゴリズム・バンクに対応する。

画像処理ソフトウェア・バンクを、より強力なものにするため、複合演算バンクの表現形式を拡張し、

・複合演算の構成要素として、他の複合演算を許す、

・１つの複合演算名（例 smooth ）に対し、複数の実体（例 median,average ）を許し、対話的に具体化させる、

ようにし、汎用ソフトウェア・バンク中のプロット・バンクに近づけることが考えられる。

```
+[100,100] {
<> QRED4D( in ) <IN> QFVS4D( posi, f1 ) /
   Q3WGTA( "WA" ) | Q3EDGA( "E5" ) <ED>
      QFVS4D( nega, f2 ) /
   QN3MID( thresh ) | QBRDG2 <CT>
      QFVS4D( ngbn, f3 ) /
   QWRT4D( contor ) <>;
< IN, CT > QTHPRB( thresh ) |
   QWRT4D( downth ) <>;
< <CT> QDELP1, ED>
   QRSLN3( func, cutln, thresh ) <>;
}
```

図３．複合演算バンク用表現形式

４．おわりに

画像処理用ソフトウェア・バンクの考えは今後、異種計算機がネットワークで結ばれる機会が増加するにつれ、ますます重要になると思われる。

本研究は、通産省の大型プロジェクト" 電子計算機相互運用（インターオペラブル）データベースシステムの研究開発" の下で行なわれた。研究の機会を与えられた棟上ソフトウェア部長に感謝の意を表する。

参考文献

１］田村　他：ポータブル画像処理ソフトウェア・パッケージＳＰＩＤＥＲの開発、情報処理学会論文誌、Vol.23, No.3, pp.321-328, 1982

２］S.Krusemark & R.M.Haralick: An Operating System Interface for Transportable Image Processing Software, CVGIP., Vol.23, No.1 pp.42-66, Jul. 1983

３］ W.R.Stevens & B.R.Hunt : Software Pipelines in Image Processing, CGIP., Vol.20, No.1, pp.90-95, Sep. 1982

４］藤村：ラスタ走査形画像処理のプログラミング・ツール、情報処理学会コンピュータビジョン研究会資料、21-2、1982.11

５］大石：手続きの抽象化・蓄積・再利用、情報処理学会ソフトウェア工学研究会資料、46-10, 1986.2

６］藤村：画像処理アルゴリズム記述の簡略化と実行コードの自動生成、情報処理学会第３２回全国大会資料、1N-4、1986.3

７］藤村：画像処理データ流の記述と実行形式への変換、情報処理学会第３１回全国大会資料、5N-8,1985.9

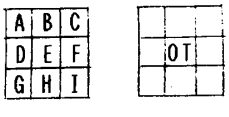

図２．画素アクセスマクロ例

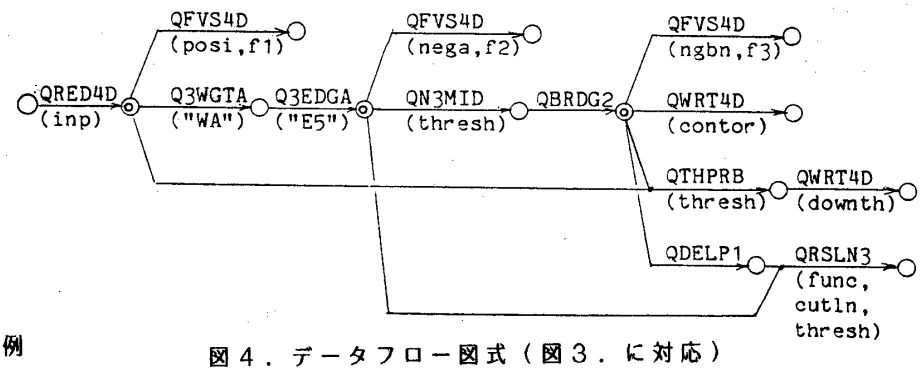

図４．データフロー図式（図３．に対応）