コンピュータビジョン 39－4
（1985. 11. 21）

データフロープロセッサを用いた図面画像の実時間変換処理

A REAL TIME TRANSFORMATION METHOD OF DRAWING IMAGES USING DATAFLOW PROCESSORS

謝　輝　　有木　康雄　　坂井　利之
Hui SHIEH　Yasuo ARIKI　Toshiyuki SAKAI
京都大学　工学部
Faculty of Engineering, Kyoto University

We have been developing a drawing image workstation by which drawing images are freely modified, changed and transformed. This workstation mainly consists of two parts. One is image processing for extracting constituents of drawing images. The other is transformation process to produce transformation plan on the description of a group of extracted constituents. To achieve high performance, it is required to integrate image processing by special processors and transformation process by general micro processor. In this paper, a real time transformation which is achieved by utilizing dataflow processors for image processing is described.

1．はじめに

オフィスにおける文書の作成ではワードプロセッサにより文書を作成・編集するだけでなく，イメージスキャナ等により図・写真を画像として入力・編集し，イメージプリンタ等に出力することのできる文書画像編集装置が商品化されている。この種の装置の機能としては，1)既存の文書より図・写真といった領域を対話的に切り出し，2)拡大・縮小・回転等を行って，3)文書中の任意の位置に配置し，文書を挿入することができるといったものであり，貼り合わせ文書を作成できるようになっている[1]。

ところが，既存の図をそのまま用いるのではなく，図の内容を一部修正して利用したいとか，図の形・サイズを変更したい，図中の文字を変更したいといった要求に対しては，既存の文書画像編集装置では対応することができない。そこで，CADのようなグラフィックスに対してではなく，イメージに対して修正・変更を自由に実行することのできる装置が望まれるようになってきた。また，これらの処理をワードプロセッサを使うような手軽さで高速に実行したいという要求もある。これらの要求を新しい文書処理装置の機能としてまとめると次のようになる。

(1)図を一定の領域に挿入するため，図形については横・縦で倍率の異なる拡大・縮小が実行でき，文字については歪まないように横・縦等倍率の拡大が行える。
(2)図中の文字列を英語から日本語といったように異なる言語の文字列に置き換える。このとき置き換えにより文字列が長くなれば，それを囲んでいる枠も拡大する。
(3)ＯＨＰやスライド用図面など各種目的別図面へ変換する。
(4)文字やシンボル(論理回路記号など)の配置に関する知識を用いて，見やすい良品質図面に変換する。
(5)対話により図中の一部を削除・修正したり，新しく挿入することができる。

このような図面処理に対する新しい機能を実現するためには，次のような問題を解決しておく必要がある。

(1)文字と図形の分離。
(2)文字の変換……これにはＯＨＰ用の文字への変換，品質の良くない文字の整形，品質の劣化が少ない文字の拡大，フォント変換，異なる言語の文字列への変換などがある。
(3)図形の変換……これには文字列を囲む枠（ブロッ

ク領域と呼ぶ）の拡大，ブロック領域の拡大を妨げる線や文字の移動，同じサイズのブロック領域に対する同一変換，矢印・点線・線巾などの線属性の変換などがある。

(4)高速処理……文字や図形の変換には局所的なマスク処理としてのイメージ処理と，構造把握に基づく大局的な変換計画とが含まれている。オフィスでの文書処理にふさわしいように，小型で高速な信号処理ユニットが必要とされる。

本稿では，新しい文書処理機能として(1)の異なる倍率での拡大・縮小，(2)の異なる言語の文字列への変換をとりあげ，主に図形の変換と高速処理の実現方法について述べる。

図形の変換では，イメージ処理として，文字・線分・接続点・ブロック領域等の図面を構成する要素を抽出し，その図面の概略構造を記述する。次にこの概略構造記述をもとに図形の変換を計画し，出力画像を生成する。即ち，イメージ処理と記述上での変換処理といった２階層の変換により，イメージ処理だけでは達成できない高度で高品質な変換処理を実現している。

高速処理では，局所的なマスク処理としてのイメージ処理はデータフロー型イメージプロセッサを用いて並列処理し，記述上での変換処理はマイクロプロセッサで実行するといった機能分担により高速処理を実現している。

２．構造把握に基づく図面画像の変換

2.1 図面変換方式の分類と比較

図面画像の変換方式は，その図面記述のレベルによって大きく次の３つに分類できる。

(1)ビット変換……図面を画像データとして表現する。図面に関する知識を必要とせず，入力画像から直接出力画像を生成する。すべての画素に対して均質な処理が施される。文書画像編集装置における拡大・縮小や切り貼り編集がこれにあたる。

(2)シンタックス変換……図面を文字，接続点，線分などの構成要素の集まりとして記述する。構成要素の性質に関する知識を用いて入力図面から図面の構造記述を作成し，構造記述上で変換を計画し出力画像を生成する。各構成要素ごとに別々の処理が施される。

(3)セマンティクス変換……図面中のシンボルを認識し，それらの接続関係を記述する。論理回路図や設計図面などの特定の図面の性質に関する知識を用いて構造記述から意味表現を抽出し，その上での変換を行う。手書き図面の清書やＣＡＤシステムでの図面変換がこれにあたる。

これら３方式は図１に示すような階層をなすと考えられる。ビット変換，シンタックス変換，セマンティクス変換の３方式を，処理速度，処理対象，変換機能の各観点から比較すると次のようになる。

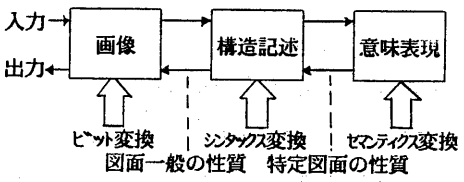


図１　図面変換における階層性

(1)ビット変換……局所処理のみであるので高速であり，処理対象も限定されないが，変換機能はビットレベルに限られ最も低い。

(2)セマンティクス変換……変換機能は最も高い。しかし，図面の表現する意味内容まで認識するためには，処理対象を限定することが必要で，また図面の構造解析やシンボル認識に時間がかかる。

(3)シンタックス変換……処理速度はビット変換とセマンティクス変換の中間であり，また，処理は基礎的で一般の図面を対象とすることができる。変換機能はセマンティクス変換より低いが，通常の目的には十分である。

以上の考察からシンタックス変換方式は処理速度，処理対象，変換機能の観点から文書中の図面変換に関しては妥当であると考えられるので，本研究では，このシンタックス変換に基づく変換方式を採用している。

2.2 シンタックス変換における処理の概要

シンタックス変換に基づくシステムは，図２に示すように次の３つの部分から構成されている。

(1)構造把握部……入力画像を解析して文字や線分といった構成要素に基づいて，図面の構造を把握し，入力図面の構造記述を作成する。

(2)図面変換部……利用者による変換機能の指示に基づき，入力図面の構造記述から出力図面の構造記述への変換を行う。

(3)画像生成部……入力図面の構造記述及び出力図面の構造記述に基づき，入力画像から出力画像への変換を行う。

システムの特徴は，1)構造把握部において，図面の各構成要素の位置と互いの位置関係を記述した構造記

述を作成する点，2)図面変換が構造記述レベルでの変換と画像データレベルでの変換の２階層で行われる点である。この２階層変換の利点は，変換に必要な情報が構造記述中にすべて含まれているので，効率的な変換ができること，および画像レベルでの変換により原画像の持つ情報を保存した変換が可能なことである。

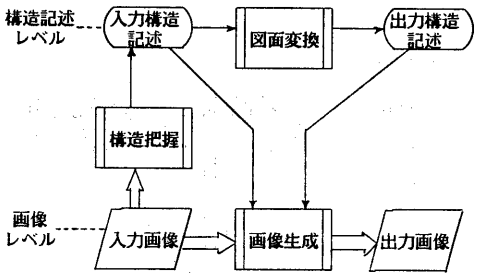


図２　シンタックス変換における処理

2.3 図面の構造記述

扱う画像は二値画像であって，図３に示すように一般に文字領域と図形領域から構成されている。図形領域はさらにいくつかの線分から構成されている。また，線分と線分の交点や，分岐点，屈曲点といった接続点と，さらに，ひとまとまりの文字領域を囲む図形枠内の領域（ブロック領域）が存在する。

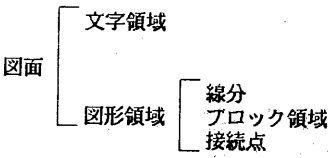


図３　図面の構成要素

図面の構造記述は，各図面構成要素の存在位置及び構成要素間の位置関係の記述である。文字領域は，その外接長方形の頂点の座標によってその位置を記述し，線分は，その始点の座標と終点の座標によって記述しておく。接続点ではその位置及び，接続する線分へのポインタを記述する。ブロック領域ではその位置を外接長方形の頂点の座標で記述し，内部に含む文字領域をポインタで記述しておく。

３．ImPPを用いた図面画像変換システム

3.1 データフロー型イメージプロセッサImPP

図面画像変換処理には画像データから構成要素を抽出するイメージ処理と，構成要素の記述に関する変換処理とがある。イメージ処理では，汎用のフォン・ノイマン型計算機を用いると，並列処理の弱さ及びフォン・ノイマン・ボトルネックのために，処理速度が著しく遅くなるといった問題がある。この問題を解決する１つの方法として，マルチ・プロセッサ構成を容易にとることができ，その上で並列処理を記述することのできるデータフロー型イメージプロセッサを用いる方法がある。本研究ではイメージ処理に新しいアーキテクチャを持つパイプライン型データフロー・プロセッサImPP[2]（Image Pipelined Processor：NEC)を用い，記述上での処理には汎用ホストプロセッサ(マイクロプロセッサ）を用いて，各プロセッサの処理能力を最大限に利用する方式を採用している。

使用しているImPPは次のような特徴を持っている。

(1)ユーザ・プログラムにより，動的に多種類のパイプライン処理を並列に行うことができる。

(2)処理の流れがデータ駆動に基づいている。

(3)マルチ・プロセッサの構成が容易にできる。

したがって，簡単かつ繰り返し演算が多く，並列に処理できる部分の多いイメージ処理に適したアーキテクチャである。

3.2 システムの実現

ホストプロセッサ及びImPPの性能を最大限に発揮し，実時間処理を実現するために，次の点を考慮している。1)イメージ処理と記述上での処理を効果的に分割すること，2)ホストプロセッサとImPPの処理が並列に行えること，3)イメージ処理アルゴリズムはImPPの特徴を十分考慮にいれたものであること。

1)は処理アルゴリズムに依存するが，ここでは，図２に示したように，構造把握，図面変換及び画像生成という３つの処理モジュールに分けられているので，構造把握と画像生成はイメージ処理，図面変換は記述上での処理とすることが効果的である。

ImPPの実行方式により，2)は自然に満たされている。ImPPのソフトウェアをファームウェア・パッケージとしてメモリ上に置き，簡単な命令によって，自動的にImPPにプログラムをローディングすることができるようになっている。さらに，数個の起動トークンによって実行が始まるようになっているので，ImPPを制御するホストプロセッサの負担が極めて小さく，ホストと

ImPPの並列処理が十分可能である。，3)はデータフロー型アルゴリズムに依存するので，これについては以降で述べる。

（1）ハードウェアの構成

図4に示すように，ホストプロセッサとして8086を用い，イメージ処理は4つのImPPを用いている。4つのImPPを並列に用いる方法には1枚の図面を4つに分割して並列処理する空間分割と，処理アルゴリズムを機能的に分割して各ImPPに分担させる機能分割の2つの方法がある。どちらの方法を用いるかはほぼ処理内容により決定されるが，空間分割では処理後の統合を，機能分割では負荷分散のバラツキを考慮しておく必要がある。

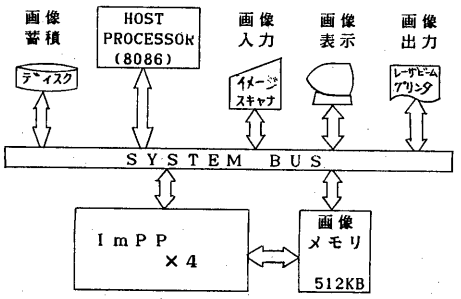


図4　ハードウェアの構成

（2）ソフトウェアの構成

図5に示すように，ソフトウェアはホストプロセッサで処理する部分とImPPで処理する部分に分かれている。HOST側ではImPPの制御，及び入力画像から得られる入力構造記述を出力構造記述へ変換する図面変換を分担する。ImPP側では入力画像の解析を行い，文字や線分といった構成要素を抽出し，図面の構造を把握する構造把握処理，及び出力構造記述より出力画像を生成する画像生成処理を分担する。

現在ImPPファームウェアパッケージとしては表1に示すようなプログラムがあり，図面変換に必要なソフトウェアについて次節以降で述べる。

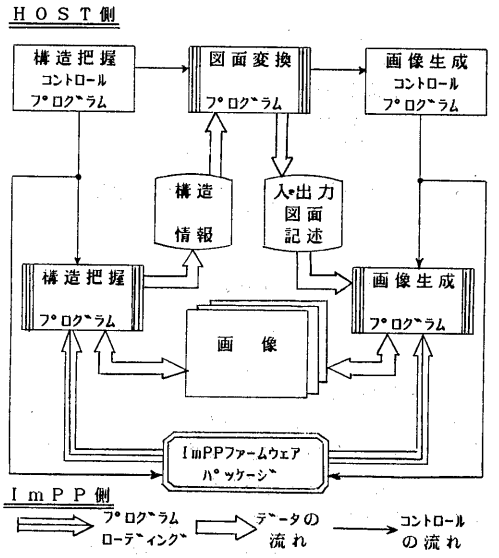


図5　ソフトウェアの構成

表1　ImPPプログラム

プログラム名	ImPP使用個数
境界線抽出	4
境界線トレース	2
圧縮画像生成	2
線分抽出	2
画像生成	4
フォントパターン埋め込み	1
画像コピー	1
クリア	1
面積投影	4
アフィン変換	4

4．構造把握

4.1 構造把握処理の概要

構造把握部では図6に示す処理の流れにより，入力画像を解析して構成要素の記述を作成する[3],[4]。まず，黒画素の連結領域と白画素の連結領域の外接長方形の存在位置を求める。以後，黒画素の連結領域を黒領域，白画素の連結領域を白領域と呼ぶ。これをもとにして文字領域及びブロック領域を抽出する。次に入力画像から文字領域を消去した画像を作りこれを走査することにより線分を抽出する。最後に，抽出された線分をもとに接続点を抽出する。黒領域と白領域の抽出及び縦・横線の抽出はイメージ処理であるので，ImPPで実行するが，その他の処理は構成要素間の演算であるのでホストプロセッサで行う。

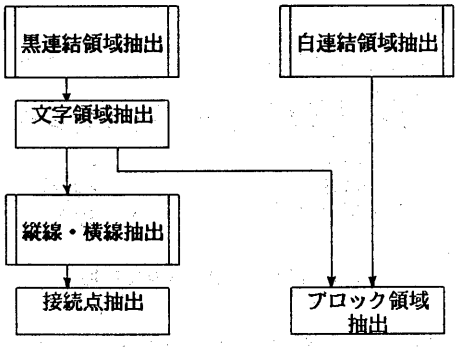


図6　構造把握処理の流れ

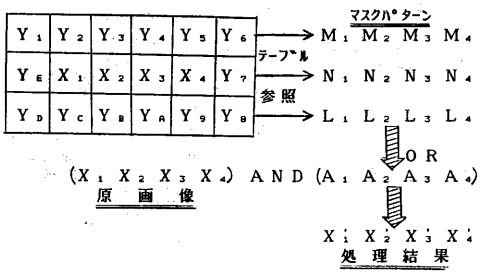


図7　境界点の抽出

4.2 連結領域の抽出

原画像から8連結の黒境界線と白境界線を生成し，境界線をトレースし，連結領域の外接長方形を求める方法を採用している。この方法は，伝幡や追跡法[5]に比べ，処理されるデータ量が少ないため高速処理に適しているといえる。黒境界線と白境界線抽出の差は，画像の黒画素を1とみるか，あるいは0とみるかの違いだけであるので，以後黒境界線についてのみ述べることにする。

（1）境界線の抽出

点X_0が境界点であるか否かを判定するには，X_0の8隣接点X_1～X_8に対して次のマスクオペレータを適用する。

$$X_0' = X_0 \cdot \overline{X_1\ X_2\ X_3\ X_4\ X_5\ X_6\ X_7\ X_8}$$

このオペレータをデータフロー型アルゴリズムに変更するためには，連続したいくつかの画素をまとめて一括処理するとともに，この処理を並列化する。即ち図7に示すように，原画像の4点X_1，X_2，X_3，X_4に対して，3×6の領域を切り出し一括処理する。処理の内容は，まず，各行の6点からテーブル参照によりマスクパターンM_i，N_i，L_i，$(1 \leq i \leq 4)$を作る。この3つのマスクパターンのORをとって，原画像のX_1，X_2，X_3，X_4とANDをとれば，処理結果が得られる。

マスクパターンは，各行の6点で示す値によって一意的に決まるので，予めImPPの内部RAMにマスク テーブルとして登録しておき，各行のビットパターンの値で参照する。またメモリアクセスの最小単位は1ワードであり，1ワード中に16画素含まれているので，上下連続する3ワードを4つの3×6の領域に分割し，並列に処理してゆき最後に統合する。

(2)境界線のトレースによる連結領域の抽出

連結領域の外接長方形を求めるアルゴリズムは抽出した境界線に対して次の手順で行う。まず図8に示すように，黒画素X_0の値を1から0に変更し，その4近傍を調べる。もし黒画素が存在すればそれをスタックに入れるとともに，連結領域のX，Y座標の最大，最小値を求めるため既に得られている黒画素のX，Y座標の最大，最小値と比較しその結果を記憶しておく。4近傍における処理が終わると，スタックから黒画素を取り出し，スタックが空になるまで同じ処理を繰り返す。スタックが空になった時点で，閉領域の外接長方形が求まりX，Y座標の最大，最小値が得られる。

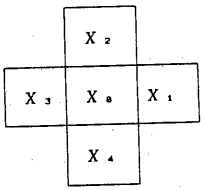


図8　画素X_0の4近傍

この境界線をトレースするアルゴリズムをデータフローで実行するためには，並列に処理可能な部分に分割し，それらの間で同期をとりながら実行する。ここでは図9に示すように，次の4つの並列に動作可能な部分に分割している。

a)SEARCHER: 画像の始端から走査を開始し，黒画素の存在するワードをみつけた時点で，その座標をTRACERに渡し終了する。

b)TRACER: 黒画素の値を1から0に変更し，その4近傍を調べ，黒画素が存在すればその座標をCMP，STACKに渡す。4近傍の処理が終わると，STACKにPOP UPトークンを出す。

c)STACK:TRACERからの座標を蓄積し，POP UPトークンがきたらTRACERにFILO (First In Last Out)の順に座標を渡す。空になったら，SEARCHERを起動して走査を始めさせ，CMPの座標を出力させる。

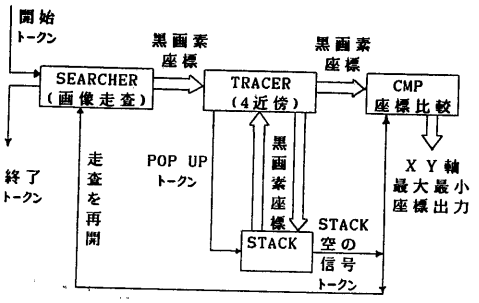


図9　トレースの動作

d)CMP: X，Y座標の最大，最小値を比較する。

この処理では，並列性が2箇所で現れている。1つはCMP,TRACERの2つの部分が同時に動作している点である。もう1つは各部分内の並列性である。即ち，TRACERは黒画素を1から0に変更する動作と，4近傍の4点を同時に調べる動作を並列に実行し，またCMPはX，Y座標の最大，最小値の比較を同時に実行している。このトレースの処理は1つのImPPで実行されるが，黒・白境界線各々に1つのImPPを用いているので，計2個のImPPが並列動作している。

4.3 文字領域及びブロック領域の抽出

文字と図形の接触はないものと考えると，文字領域は縦方向または横方向に配置された黒領域の集合からなる。従って，すべての黒領域のうち，外接長方形の縦及び横の長さがともにあるしきい値より短いものを，文字を構成する黒領域と判定する。これらの黒領域を順次マージすることにより文字領域を抽出する。

ブロック領域とは置換の単位である文字列を中央に含む領域のことである。次の手順によって抽出する。

(1)すべての白領域について，その内部に文字領域を含んでいるかどうかを調べる。内部に文字領域を含む場合，白領域自身の重心と白領域内に含まれる文字領域の重心との距離を求め，その距離がある値以下である場合にブロック領域であると判定する。

(2)次に，すべての文字領域について調べ，どのブロック領域にも含まれていない文字領域があれば，それを新たにブロック領域として登録する。

4.4 縦線・横線の抽出

正確な線の位置を抽出する必要はないので，高速化可能な局所オペレータを適用して線切片を検出し，それらを結合することにより線分を抽出する。まず，文字領域を入力画像より消去し，図形領域のみの画像を作成する。縦線分を効率よく抽出するために，図10に示すようにこの画像に対して8走査線ごとにAND操作を施し圧縮画像を生成する。横線分を抽出するためには，縦方向のAND操作を行い圧縮画像を作成する。次に，圧縮画像上で，次のような条件を満たすものを線分として抽出する。1)線の幅がある値以下，2)傾きの変化が一定の値以下，3)横方向の走査によって検出した縦線の断面(線切片)の距離が走査線間である値以下。圧縮画像を走査し，線切片をみつけたら，その線切片から次の線切片を探索してゆき，上の条件をチェックしながら線分を1本ずつ抽出する。

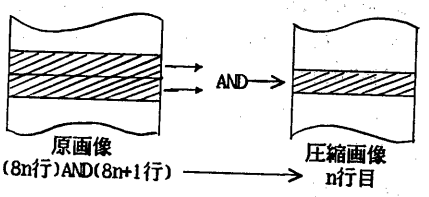


図10　縦線抽出用圧縮画像の生成

ImPPの上では，図11に示すように，3つの並列に動作しうる部分に分けられる。画像走査は図9に示したSEARCHER部の機能とほぼ同じである。探索を開始する線切片をもとに次の線切片をみつけ，比較部に渡し，線分としての条件を満たしているかどうかをチェックする。比較部では条件2)により傾きの比較を行う必要があるが，ImPP上に除算器がないため，乗算に直して比較を行っている。

斜線は縦線・横線として二重に抽出される場合があり，また若干の抽出誤りがあるが，線のおよその存在位置が分かればよいので，この程度の抽出精度で十分である。

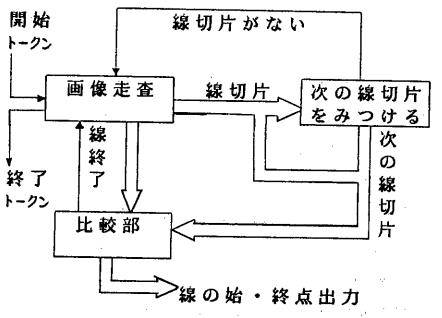


図11　線抽出の動作

4.5 接続点の抽出

接続点は線分と線分の交点や分岐点，あるいは屈曲点である。接続点の位置は座標によって記述し，線分との接続関係は線分テーブルへのポインタによって記述する。接続点の抽出は，線分テーブルをもとにして，以下の手順で行う。

(1)すべての２つの線の組合せについて，それぞれの始点・終点の座標から，交点を持つかどうかを調べる。交点を持つ場合はその交点の位置で線分を２つに分割する。これによってすべての接続点は線分の始点または終点に位置することになる。

(2)再び線分テーブルを調べて，各線の始点及び終点を順に接続点テーブルに登録してゆく。このとき１つの接続点を登録するごとに全ての線を調べて，この接続点に接続する線をポインタによって連結する。

５．図面変換及び画像生成

5.1 図面変換

図面変換部では入力図面の構造記述をもとに出力図面における構成要素を決定し，出力構造記述を作る。この入力構造記述から出力構造記述への変換は，文字領域の移動と置き換え，接続点の移動，線分の移動，ブロック領域の拡大・縮小・移動といったような基本図面操作の組合せによって実現する。

(1)図面の日英変換処理

次の手順で変換を行う。

a)利用者は各ブロック領域ごとに置き換える文字をキーボードから入力する。

b)システムは各ブロック領域について，ブロック領域の大きさと置き換え後の文字数からその整合性をチェックする。必要があるときは，ブロック領域の拡大を行ってから文字の置き換えを行う。ブロック領域の拡大に伴い，周囲の構成要素を移動するなどの操作を行う。

(2)図面の拡大・縮小

利用者が縦方向・横方向それぞれの拡大・縮小率を与える。システムはそれに基づき，接続点及びブロック領域を所定の位置に移動する。線及び文字領域の位置は，接続点及びブロックの移動にともない自動的に配置される。

5.2 画像生成

画像生成部では入力構造記述と出力構造記述に基づいて，入力画像から出力画像への画像レベルでの変換を行う。構造記述レベルにおける変換は基本図面操作の組合せで実現されているので，それらに対応する画像レベルでの以下の３つ画像生成処理を用意しておけば良い。この３つの処理はすべてImPPによって行われる。

(1)線分の複写……図面画像中の図形に関しては，線分を単位とした複写によって出力画像を生成する。各線分には入力画像と出力画像上での始点・終点位置，更に各線分の傾きが記述されている。ホストプロセッサは線の複写において，その記述内容をImPPへ渡す。ImPPは始・終点及びそれらの傾きを用いて，線の中心点を決定し，中心点の左右に存在する黒画素数を出力画像における中心点の左右に複写する。

(2)文字領域の複写……文字の置き換えが行われない場合には，入力画像の文字領域をそのまま出力画像上に複写する。複写すべき位置は入力・出力構造記述によって記述されている。

(3)文字フォントの埋め込み……文字の置き換えを行う場合には，出力構造記述には文字コードとその位置が記述されている。その場合，１６×１６のフォント・パターンをImPPのメモリにおき，文字コードにより，フォント・パターンを取り出し，それを出力画像中に埋め込む。

６．実験結果

解像度８本／mmのイメージキャナから入力した1024×1024画素の図面画像を対象に実験を行った。実験の結果から，図１２に示すような高品質な変換が実現できるとともに，表２に示すような処理時間で処理することができ，実時間処理の見通しが得られている。表２にはｽｰﾊﾟｰ･ﾐﾆｺﾝﾋﾟｭｰﾀ MS-190でシミュレーションした処理時間も比較のためにのせている。

現在の処理時間は１９秒かかるが，そのうちホストプロセッサの処理時間が１１秒かかっている。ホストプロセッサとして５MHzの8086を用いているが，将来80286 に切り換える予定である。これにより，５秒程度に短縮することが可能である。さらに画像メモリの制限により，文字フォント・パターンを常駐させることができず，画像生成中にＨＯＳＴのメモリから転送しているが，画像メモリの増設により更に高速化は可能である。従って，トータル処理時間を１０秒ぐらいで実現することも可能である。

７．おわりに

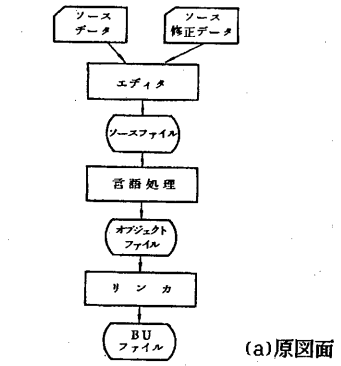
本稿ではデータフロー型イメージプロセッサを用い

て，図面画像変換処理を高速で行う方式を提案した。

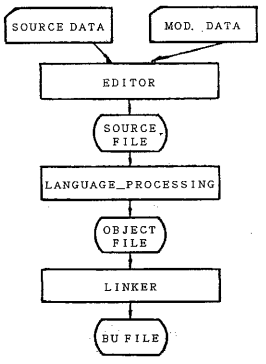
図面のイメージ処理と構成要素の記述に関する処理を，それぞれデータフロー型イメージプロセッサ及びマイクロプロセッサに分担させることにより，良好な変換機能と実時間処理の見通しが得られた。

表２　処理時間

処理内容	処理プロセッサ	処理時間(S)	MS-190(4MIPS)処理時間(S)
構造把握		１２．５	５０．６
白領域抽出	ImPP	1.5	22
黒領域抽出	ImPP	1.5	22
文字・ブロック領域抽出	ホスト	2.5	0.4
線分抽出	ImPP	1	3
接続点抽出	ホスト	6	3.2
図面変換	ホスト	2	０．１
画像生成	ImPP	４．５	３．０
合　計	－－	１９	５３．７



(a)原図面



(b)英語図面への変換結果

図１２　図面変換の例

参考文献

[1]上林,田畑,中村:“画像処理機能を持ったT-560/20マルチワークステーション”，日立評論Vol.65，No.11,1983

[2]H.Kurokawa,K.Matsumoto,T.Temma,M.Iwashita and T.Nukiyama:“The Architecture and Performance of Image Pipeline Processor”，Proc. of VLSI'83 International Conference,1983

[3]Y.Ariki，K.Wakimoto，H.Shieh and T.Sakai：“Automatic Transformation of Drawing Images Based on Geometrical Structures", Proceedings of COMPINT'85，Sep. 1985

[4]脇本,有木,坂井:“図面画像の変換処理　－日本語図面から英語図面への変換－　”，電子通信学会，パターン認識と学習研究会，PRL84-91,1985

[5]A.Rosenfeld and A.C.Kak:“Digital Picture Processing”，Academic Press,1976