# 文書OCRにおける出力テキストの整形方法

1H－9

平山唯樹
日本アイ・ビー・エム株式会社 東京基礎研究所

## 1　はじめに

印刷文書用OCRシステムを用いて文書を認識した後に、その処理結果を有効利用するために、認識結果をどのような形態で出力するのがよいかというのは大きな問題である。また、認識しながら、あるいは認識が終了した後に、認識結果を確認修正する場合に、どのような形態で表示すればより効率的な確認修正ができるかということもシステム全体から見て重要である。

認識した文字をテキストファイルとして出力するのがもっとも一般的な方法であると考えられるが、そのテキストファイルでの出力の従来の方法として(1)文字を認識した順に、上から文字列単位で出力する方法、と(2)オリジナルイメージのレイアウトをできるだけ再現して出力する方法、がある。(1)も(2)もそれぞれ認識結果を利用する際には有用な表現形式である。特に、(2)の方法では確認修正の際にオリジナル文書と比較がしやすい等の利点を持つ。従来は、(2)を実現するために文字の座標情報から位置を計算してその位置に文字を表示するという方法で実現していた。しかし、この方法は複数カラムをもつドキュメントを処理した場合などに第2カラム目以降の左端がそろわないなどの問題点をもっている。この問題点の原因としては、座標から文字数を計算する時の誤差、オリジナル文書とテキストファイルとで文字ピッチ、行ピッチ、フォントサイズ等が異なるということがあげられる。

そこで本稿では、従来(2)の方法で生じていた問題点を解決するために文字認識に先だって行なわれるレイアウト解析の結果を利用して認識された文字を整形して出力する方法を示す。

## 2　レイアウト解析から得られる情報

レイアウト解析処理は、文字認識処理に先だって行なわれる[1]。まず、文書イメージ全体を文字ブロック、図形ブロック、表ブロックに自動的に分割する。そして、各ブロックの位置、サイズそして、各ブロックがお互いに端が揃っているかどうかを調べ、それらの情報を後に続くプロセスへ渡す。

A method for formatting OCR texts
Yuki HIRAYAMA
IBM Research, Tokyo Research Laboratory, IBM Japan Ltd.

## 3　認識結果の整形

認識結果を出力するために、まず出力テキストというのが準備される。そしてブロック単位で認識結果の出力テキストへのマッピング・整形がなされる。この認識結果のマッピング・整形には大きく分けて2つのステージがある。第1のステージはブロックのソート、第2のステージは文字のマッピングである。以下に各ステージについて説明する。

### 3.1　ブロックのソート

前節で指摘したように、オリジナル文書とテキストファイルとで文字ピッチ等が異なるために、テキストブロックの出力テキスト内での幅・行数はそのテキストブロック内の文字の認識がすべて終了しなければ決定できないという問題が生じる。従って出力テキスト上でブロックが重ならないようにするためにより左、上にあるものからマッピングしていく必要がある。そこで、ブロックを左・上優先でソートして、その順に認識する。本方法では左・上に他のブロックがないブロックを一つづつ取り出していくという方法で全ブロックの順序を決定している。ある段階で、複数のブロックが候補に上がった時は、前にすでに順序づけされたブロックとの距離が近いものから先に順序づけている。それでもまだ決定することができない場合はその候補の中の任意のブロックを取り出す。

### 3.2　文字のマッピング

ここで、決定された順序にしたがってブロック単位で認識結果を出力テキストにマッピングする。マッピングする際の位置決めの拘束条件としてレイアウト解析で得られた情報を用いる。拘束条件としては、(1)他のブロックの{下／右}にある、(2)他のブロックと{左／上}端が揃っている、というものがある。これらの拘束条件は文字ブロックごとに、また表の場合は表全体のブロックとともに表のカラムごとにもっている。

このマッピングのステージは(1)ブロックの位置決め、(2)文字データの配置、(3)整形、という3つのプロセスからなる。以下に各プロセスについて説明する。

**ブロックの位置決め**　まず、各ブロックの左上の座標を決定する。最初に座標から位置を計算し、次に拘束条件によりその計算値を補正するという方法により決定する。計算した座標値を用いるとすでにあるブロックと重なってしまう場合には重ならない位置まで右、下へ移動する。さらに他のブロックと端が揃っているという条件がある場合には条件に合うように移動する。この際、どうしても端を揃えるという拘束条件に合致しない場合が生じることがある。この場合には、ブロックが重ならないようにとりあえずマッピングをしておき、後の整形のプロセスによって拘束条件に合うように変更する。この場合の詳細に関しては後の整形プロセスにおいて述べる。

**文字の配置**　ブロックの左上の座標が決まったので、この座標を基準に文字認識結果を出力テキストに配置していく。

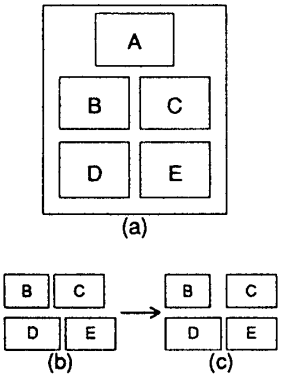


図 1 整形処理.

**整形**　この整形のプロセスは、位置決めの段階で拘束条件に満たさないものが生じた場合にのみ起動される。拘束条件に満たさない場合の例を図 1 にあげて、このプロセスを説明する。図 1-a がテキストブロックの配置の例である。この例ではブロックのマッピングの順序として A–E となっている。図 1-b は B ブロックから E ブロックを順に出力テキスト上にマッピングしたところを表している。ここで、C ブロックと E ブロックに着目する。図 1-a をみると、C ブロックと E ブロックは左端が揃っているので、E ブロックの拘束条件として「C の左端と同じ位置にマッピング」しなければならない。しかし、図 1-b の D ブロックを見ると、B ブロックよりも横方向の長さが長い。つまり E ブロックをマッピングする時に、C ブロックの左端と揃えることができないので、位置決めの段階で E ブロックは D ブロックの右横にとりあえず配置される。そして整形プロセスで再びすべてのブロックの位置決めをし直す。最初に位置決めをした時と位置が変化しているブロックのみ、ブロックの移動を行なう。ここでは、C ブロックの位置決めの際、この時には E ブロックの左端の位置が分かっているので「C ブロックは E ブロックと左端が揃っている」という条件により C ブロックの左端の位置が E ブロックの左端の位置と同じ位置に補正され、図 1-c のように右に移動される。以上が整形プロセスである。

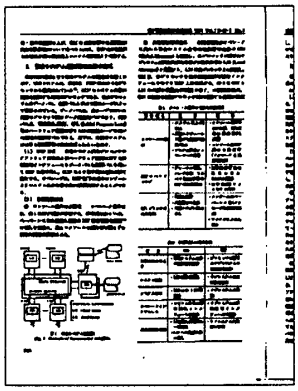

図 2 サンプルイメージ.

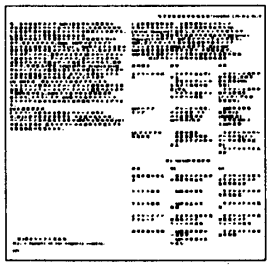

図 3 出力ファイル.

ここで、処理の具体例を示す。図 2 がサンプルのイメージであり、その処理結果の出力ファイルとして図 3 が得られた。図 3 においてオリジナルイメージのレイアウトをほぼ保持していることが確認できる。

## 4　おわりに

OCR システムで認識した結果を出力する方法は何種類も考えられ、それぞれの方法を用途に応じて使い分けるべきである。その中で、本方法のようにオリジナルのレイアウトを取り込んだ出力テキストファイルは、文字認識の確認修正、またワードプロセッサ等で再利用する場合、文書をテキストファイルとしてデータベースにしまいこみ、ビューワーで見る場合などに非常に有用であると考えられる。

## 参考文献

[1] Y. Hirayama, "A BlockSegmentation Method for Document Images with Complicated Column Structures," Proc. of ICDAR '93, pp.91-94,1993.