

# PDF 中のテキスト情報を利用した視覚障害者のための英文 PDF 科学技術文書読取りシステム

金堀 利洋<sup>†a)</sup> 鈴木 昌和<sup>††b)</sup>

Scientific English PDF Document Reader for Visually Impaired People Utilizing Embedded Text Information

Toshihiro KANAHORI<sup>†a)</sup> and Masakazu SUZUKI<sup>††b)</sup>

あらまし Web アクセシビリティの意識が浸透しつつある一方で、電子図書など情報が PDF 文書として提供される流れが加速している。配布されている PDF 文書には文字情報が、人が読む順序と異なる順番に埋め込まれていて、視覚障害者がその内容を読み取ることが困難であるといった問題が指摘されている [2]。また、数式の情報が読み取れる形で入っていることは少ない。更に、印刷文書を OCR を用いて認識し、その認識結果を PDF 文書として生成した場合、数式がその周辺に文字認識のエラーを引き起こし、数式中以外の文字を読むことも困難になる場合も多い。今回、特に数式を含んだ科学技術文書を対象とし、PDF 文書から抽出した文字情報と、PDF 文書を画像として認識した結果を統合することで、質の高い、アクセシブルな文書情報を PDF 文書から取り出すシステムを示す。

キーワード 視覚障害者, Web アクセシビリティ, PDF アクセシビリティ, OCR, 数式認識

## 1. ま え が き

近年のインターネットを中心とした情報処理・通信技術の発達により、Web 上のコンテンツは、我々の重要な情報源の一つとなっている。視覚障害者にとっても、以前は印刷物として提供されていた情報の多くが、Web 上のコンテンツとして配信されるようになり、スクリーンリーダ等でこれらの情報に独力でアクセス可能となった [1], [2]。しかしながら、かつては文字中心であった Web コンテンツは、画像や動画などを伴うインタラクティブなコンテンツが中心となり、視覚障害者にとって、Web ページ上にある情報を把握することが困難な状況になってしまった。そういった状況の中、障害の有無によらず Web コンテンツの情報が利

用可能であること (Web アクセシビリティ) を確保するため、Web 技術の標準化団体 W3C からガイドラインが示された [3]。国内においても、日本工業規格により、「高齢者・障害者等配慮設計指針」(JIS X8341) が示され、その中で Web アクセシビリティについて定められている。こういった取組みの結果、Web アクセシビリティの意識は、次第に浸透し、Web コンテンツ上の情報へのアクセスは、改善されてきている。

一方、Web コンテンツ内で、主に資料等が PDF 文書として提供される流れも加速している。最終的に必要な情報が PDF 文書として提供されていることが非常に多くなっている。また、印刷文書を OCR によって認識し、文字情報を埋め込んだ PDF 文書として閲覧可能とした電子図書の提供も盛んになってきている。提供されている多くの PDF 文書には以下のような問題がある：(1) 印刷文書をスキャンした画像のみで、文字情報が埋め込まれておらず、スクリーンリーダで読み上げることができない。(2) 文字情報が埋め込まれていても、その順番が、人が読む順番に従って入っておらず、正確な情報を取り出せる状態になっていない。特に、図表の周辺ではキャプションが正しく入らない

<sup>†</sup> 筑波技術短期大学障害者高等教育研究支援センター, つくば市  
Tsukuba College of Technology, Research Center on Higher Education for the Hearing and Visually Impaired, 4-12-7 Kasuga, Tsukuba-shi, 305-0821 Japan

<sup>††</sup> 九州大学大学院数理学研究院, 福岡市  
Faculty of Mathematics, Kyushu University, 6-10-1 Hakozaki, Higashi-ku, Fukuoka-shi, 812-8581 Japan

a) E-mail: kanahori@k.tsukuba-tech.ac.jp

b) E-mail: suzuki@math.kyushu-u.ac.jp

ことが多い [2]. (3) 印刷文書をスキャンし, OCR で認識させ, 文字情報を埋め込んでいても, 数式など認識が困難な部分があると, その部分の文字情報だけでなく, 周辺の情報も誤って入っている. また, 多種多様な数学記号や数式構造の情報が入っていることはほとんどない. この問題については後で詳しく述べる.

Web アクセシビリティの意識の向上により, 欲しい情報にたどり着くことはある程度できるようになったが, 肝心の情報が PDF 文書化されているために結局その情報を得ることができない. 現在の PDF には用意されたアクセシビリティのための機能があるが, それを用いて作成された PDF 文書は少なく, ただ単に, OCR や生成ソフトを用いて自動的に生成された PDF 文書をそのまま使用していることがこのような事態を生み出している. しかしながら, 既にある PDF 文書に数式の情報を入力したり, 数式によって引き起こされた文字情報の間違いを修正するのは, 大変手間と時間がかかると思われる.

今回, 我々は, 特に数式を含む英文の科学技術 PDF 文書を対象とし, PDF 文書を画像として認識した結果を, 抽出した PDF 文書内に埋め込まれている文字情報を用いて修正し, 質の高い文書情報を PDF 文書から取り出し, 視覚障害者にとってアクセシブルな形式として出力することを目的としたシステムを開発した. 使用する文書認識システムは, 九州大学鈴木研究室を中心に開発をしている科学技術文書処理システム *Infty* [4] である. 認識した結果は, *Infty* の機能を用いて, 数式部分を  $\LaTeX$  の書式で文字情報を埋め込んだ PDF 文書を生成することもできる. 本論文では, このシステムの処理の流れ, 特に PDF 文書から抽出した文字情報と, PDF 文書を画像として認識した認識結果のマッチングを用いた認識結果修正手法について詳しく述べ, この手法の評価実験の結果も示す.

## 2. 数式を含んだ PDF 文書の問題

数式を含んだ PDF 文書において, 特に数式周辺で以下のような問題が挙げられる. 例えば,  $\LaTeX$  を使用し, *dvipdfm* を使用して生成した PDF 文書では, 単純な数式:  $H = \frac{h_1}{h_2} \times 1000, D = \frac{c_1 - c_2}{h_1} \times 1000,$  を変換した PDF 文書から, *Acrobat 7* を使用して「アクセシビリティ可能なテキスト」として出力した結果は,

`h|. |c|1|..c|2|H=?1000;D=?1000. |h|. |h|. |`

the weight  $\lambda$  in  $\rho$  then

$$\det(\rho A(s) - z) = \prod_{\lambda} (\rho_{\lambda}(s, z))^{m_{\lambda}}$$

where  $\lambda$  runs through the dominant weights (see [MS1]).

(a)

the weight A in g then

[本来, 数式があるべき部分]

where A runs through the dominant...

(b)

図 1 黒く反転している部分には文字情報が入っているが, 2 行目にある数式に文字情報が入っていない.

Fig. 1 The mathematical formula on the middle line is skipped and there is no hidden text for the formula.

Consider the metric  $\rho$  defined in  $\Omega$  by

$$\rho(z) |dz| = \left| \frac{du + i*du}{\Theta(c)} \right| \text{ if } z \in U(c), \dots$$

It is not difficult to show (cf., e.g., Ohtsuka [7]) that, if  $\Gamma(a, b) \neq \emptyset$ , th

(a)

Consider the metric Q defined in  $0$  by

$\backslash du + i*du \backslash \dots$

$= J Q^{-T} L$  if  $Z \in (c), -\infty < c < \infty.$

It is not difficult to show (cf., e.g., ...

(b)

図 2 数式部分に文字情報は埋め込まれているが, 数式の情報は入っていない.

Fig. 2 The hidden text for mathematical formulae do not have correct mathematical information.

となり (各 | は改行), 全くもとの数式の情報を得ることができなくなってしまう. また, OCR を用いて, 印刷文書に文字情報を埋め込んだ PDF 文書では, 図 1(a) のように数式部分に文字情報が入っておらず, その文字情報は, 図 1(b) のように, 数式部分が認識されずに, 本来, 数式の情報が入っているべき部分に全く文字情報が入っていなかったり, 図 2(a) のように, 数式部分に文字情報は入っているが, 入っている情報は, 図 2(b) のように, 数学記号や, 数式の構造に関する情報を得ることができない. また, 数式はその前後の文字認識にエラーを生じさせる.

例えば, 図 3 の, 数式  $\zeta \rightarrow \log((\zeta - i\eta_0)^{-1})$  が行のベースラインの推定を誤らせ, その後の  $d$  を (と  $i$  に認識させたり,  $p$  を  $P$  と認識させるという認識エラーを引き起こし, 結果, その文字情報は以下のよう

本来のベースライン  
 $\zeta \rightarrow \log((\zeta - i\eta_0)^{-1})$  and applying  
 誤ったベースライン

図3 数式  $\zeta \rightarrow \log((\zeta - i\eta_0)^{-1})$  が後のテキスト部分の文字認識のエラーを引き起こしている。

Fig.3 The mathematical formula  $\zeta \rightarrow \log((\zeta - i\eta_0)^{-1})$  causes recognition errors of next characters.

になってしまっている。

by ?-^1? g ((?-^o)"1) a n ( i aPPTYing ...

このように、特に理系の文書において、数式の情報を得ることができないということは、視覚障害者が理系学問を学ぶにあたって、大変致命的なことである。

### 3. PDF 文書認識の流れ

このシステムは、以下の流れで PDF 文書の認識を行う (図4)。各プロセスの詳細な説明は後の章で行う:

- (1) PDF 文書からテキスト情報をテキストファイルの形で抽出する。
- (2) PDF 文書からページ画像をマルチ TIFF 形式の画像ファイルとして生成する。
- (3) 生成したページ画像を *Infty* で認識する。
- (4) PDF 文書から抽出したテキスト情報を、*Infty* の認識結果を用いて数式や欠落しているテキスト追加して修正を加えていく。
- (5) 修正結果を最終的な認識結果として出力する。

この手法で利用するのは、PDF 文書に埋め込まれた単純なテキスト情報のみである。よって、PDF のバージョンや、生成方法を考慮する必要はないし、更には、何らかの方法でテキスト情報さえ取り出せれば、対象は PDF 文書でなくても構わない。その反面、PDF 文書がもっている文字の位置や大きさといった有用な情報を利用することはできない。本手法の特徴的な部分は、後に述べる *Word Matching Method* によって、単語ごとにテキスト情報と *Infty* の認識結果とのマッチングを行い、対応する *Infty* の認識結果の文字方形座標を PDF 文書のテキストに割り当てていくことにより、テキスト情報の修正に必要な情報を復元するところにある。また、読上げの順番に影響するレイアウトに関しては、PDF 文書でのレイアウト (読上げ順番) は関係なく、すべて *Infty* の結果を用いる。*Infty* のレイアウト解析は、基本的にページの左上から右下に向かって、2 カラムであればカラムごとに、順番を

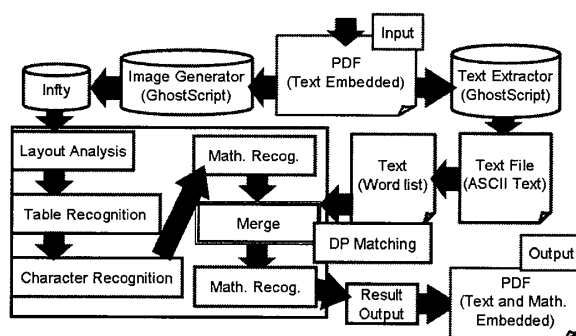


図4 PDF 文書の認識プロセス

Fig.4 Outline of PDF document recognition process.

付けていく。これによって、PDF 文書の読上げ順を改善することをねらっている。

### 4. PDF 文書からの情報抽出

PDF 文書から抽出する情報は、テキスト情報とページ画像の二つである。これらは PostScript インタプリタとしてよく用いられる *Ghostscript* [5] をバッチファイルから呼び出して行われる。まず、*Ghostscript* を呼び出し、PDF 文書からテキスト情報を ASCII テキストファイルとして抽出する。抽出したテキストファイル中の文字列をスペースで切り分ける。切り分けた各部分文字列を“単語”と呼び、テキストファイルから得られるすべての“単語”をリストとして保持する。このリストの単語と *Infty* の認識結果内の単語のマッチングを行う。次に、*Ghostscript* を用いて PDF 文書からすべてのページ画像をもった一つの 600 dpi 白黒 2 値マルチ TIFF 画像ファイルを生成する。この生成した画像ファイルを *Infty* で認識する。

### 5. *Infty* の認識結果と PDF 文書のテキスト情報を用いた最終認識結果の生成

PDF 文書から生成された画像に対して、*Infty* はレイアウト解析、文字認識、表認識、数式認識を行う。文字認識後、各単語がテキスト部であるか、そうでないかを後に説明する *Geometric Position Checking* という手法を用いて判断し、テキスト部でない単語を数式部として数式構造解析する。この解析部では、文字認識結果が変更されることもあるため [6]、解析後もう一度、テキスト/数式の判定を行う。この認識結果と、PDF 文書からのテキスト情報を用いて最終的な認識結果を生成する。基本的には、テキスト部に対しては PDF 文書のテキストを用い、数式部に対しては *Infty* の認識結果を用いるが、PDF 文書中のテキスト情報

の誤り, *Infty* の誤認識を考慮して, 以下の手順で生成を行う。

(1) 誤認識により, 本来テキスト部であるにもかかわらず, 数式部として切り分けられた場合も考慮して, いったんすべてテキスト部とみなす。このとき, 数式構造解析の結果, 分数や  $\sum$ ,  $\sqrt{\quad}$  のように, 上下の構造をもつ数式部は, 誤って数式部として切り分けられた部分ではないとして, 最終的な認識結果として採用する。

(2) テキスト部の各単語  $w$  に対して, 対応する PDF 文書中の単語  $W$  を, 生成した単語リストの中から *Word Matching Method* を用いて決定する。このとき, PDF 文書中に対応する単語が見つからなかった認識結果内の単語については, PDF 文書中で欠落した単語であると, 最終的な認識結果に加える。

(3) 対応する単語  $W$  が見つかった認識結果中の単語  $w$  の各文字  $c$  の外接方形座標を, 単語  $W$  の各文字  $C$  に対して割り当てていく。

(4) 座標が割り当てられた PDF 文書中の単語  $W$  に対して, *Geometric Position Checking* を用いてテキスト部であるか判定する。  $W$  がテキスト部であると判定された場合,  $W$  を最終的な認識結果として採用する。テキスト部ではないと判定された場合,  $W$  に対応する *Infty* の認識結果  $w$  を最終的な認識結果として採用し, 再度, テキスト部/数式部の切分けと, 数式構造解析を行う。

以上のようにして, PDF 文書のテキストに対して *Word Matching Method* を用いて *Infty* の認識結果と対応させ, 外接方形座標を復元して, PDF 文書中の単語が適切なテキスト情報であるかどうか判定する。単語に対するテキスト情報が適切でない判定された場合は, *Infty* の認識結果を用い, PDF 文書のテキスト情報の修正と, 数式情報の埋込を行う。

### 5.1 Geometric Position Checking

文字認識後, 各文字は外接方形座標をもつ。各文字に対して, その位置特徴, *top line*, *center line*, *base line*, *bottom line* が推定される。特別に, 小文字の 't' に対しては, *t-top line* が求められる (図 5 参照)。これらの特徴量は, 文字種ごとに採取した文字の高さに対する特徴量の比を用いて推定され, 数式構造解析でも使用される [6]。更に, 各行中の文字の位置特徴から多数決で, その行全体の位置特徴が推定される。*Infty* の認識率が高いことから, 行中の文字がある程度あれば, その位置特徴は信頼できる値となる。

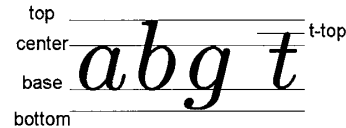


図 5 文字の位置特徴

Fig. 5 Position features of a character.

これらの位置特徴のうち, *base line* を用いて, 各単語がテキストであるか, そうでないかを調べる。単語中の各文字  $c$  に対して, その外接方形の下部の  $y$  座標を  $b(c)$  とし, 行  $L$  全体の *base line* を  $base(L)$  とすると,  $|b(c) - base(L)| > T_G$  なる文字  $c$  が単語中に存在した場合, その単語はテキストではない, と判定する。ここで  $T_G$  はしきい値で, 行  $L$  の位置特徴 *top line*, *bottom line* をそれぞれ  $top(L)$ ,  $bottom(L)$  とすると,

$$T_G := \frac{|bottom(L) - top(L)| + 16}{16}$$

と経験的に定義された値である。なお, 600 dpi の画像において,  $|bottom(L) - top(L)|$  の値は 100 程度である。単語内の各文字が行のベースライン上に, しきい値  $T_G$  の差の範囲内で並んでいるかを調べ, 並んでいなければ, その単語はテキストであると判断する。並んでいない文字があれば, 添字といった数式構造になると思われるので, テキストではないと判断する。この手法を *Geometric Position Checking* と呼ぶ。

### 5.2 Word Matching Method

各行において, 文字認識結果の列を, 空白や, テキスト部と数式部の境目で区切って“単語”とする。この認識結果内の単語に対応した単語を, PDF 文書から抽出した単語リスト内から, 以下のマッチング手法を用いて探し出す。数学記号などの特殊な文字は, バックスラッシュ '\ ' に置き換えて文字列を生成し, 単純な文字列同士として処理を行う。

#### 5.2.1 完全マッチングによる認識結果の確定

認識結果中の単語  $w_i$  に対して, 文字列として完全に一致する単語  $W_p$  を PDF 文書から抽出した単語リストの中から見つけていく。完全に一致する単語  $W_p$  が存在した場合,  $W_p$  の各文字に対して  $w_i$  の各文字の方形座標を順に割り当てたものを最終的な認識結果とする。これは結局  $w_i$  と同じものになるので, 内部的には  $w_i$  をそのまま最終的な認識結果として用いる。認識結果中の単語  $w_i$  は, レイアウト解析・行切分け結果に準じて, 基本的には行は上から順に, 単語は行中の左から右へ走査していく。この走査中に, ある  $w_i$

に対して、単語リスト中に複数の完全一致する単語が存在した場合、まず、各完全一致する単語  $W_p$  に対して、以下のようにして、完全一致度  $N_M(w_i, W_p)$  を求める。

(1)  $w_i$  の一つ前にある単語  $w_{i-1}$  と、単語リスト内で  $W_p$  の一つ前にある単語  $W_{p-1}$  とを比較する。

(2) もし、 $w_{i-1}$  と  $W_{p-1}$  が再度完全に一致した場合、 $w_{i-1}$  の文字数を  $N_P$  に加える。ここで、 $N_P$  は前方向に完全一致した文字数のカウンタとし、初期値は 0 とする。更に  $w_{i-1}$  の一つ前にある単語  $w_{i-2}$  と、 $W_{p-1}$  の一つ前にある単語  $W_{p-2}$  とを比較し、これらも完全に一致した場合、 $w_{i-2}$  の文字数を  $N_P$  に加える。

(3) このようにして、認識結果中の単語  $w_i$  の  $n$  個前の単語  $w_{i-n}$  と、PDF 文書からの単語リスト中の単語  $W_p$  の  $n$  個前の単語  $W_{p-n}$  が完全に一致する間、前にさかのぼりながら、 $w_{i-n}$  の文字数を  $N_P$  に加算していく。

(4) 同様にして、 $w_i$  の  $n$  個後ろの単語  $w_{i+n}$  についても、 $W_{p+n}$  と完全一致する間、後ろにたどりながら、 $w_{i+n}$  の文字数を、後ろ方向に完全一致した文字数のカウンタ  $N_N$  に加算していく。

(5) 認識結果中の単語  $w_i$  の前後で、対応する PDF 文書中の単語リストの単語と完全一致する単語の文字数の合計を、完全一致度として  $N_M(w_i, W_p) := N_P + N_N$  と定める (図 6)。

次に、 $w_i$  に対応する PDF 文書中の単語は、

$$N_M(w_i) : \\ = \max\{N_M(w_i, W_p) | W_p \text{ は } w_i \text{ と完全一致}\}$$

と定めたときに、 $N_M(w_i) = N_M(w_i, W(i))$  を与える PDF 文書の単語リスト中の単語とし、 $W(i)$  と書くことにする。これら単語比較の処理では、PDF 文書中の位置ではなく単語リスト中の順番を用いる。これは、

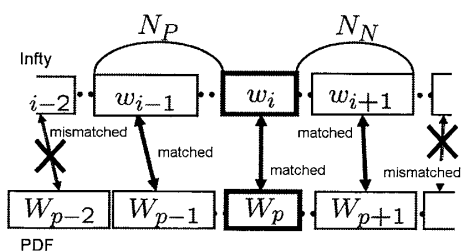


図 6  $N_M(w_i, W_p)$  の意味。この図では、各ボックスは単語を表す。

Fig. 6 Meanings of  $N_M(w_i, W_p)$ . Each box stands for a word.

PDF 文書中の行の順番の間違いを想定している。

### 5.2.2 最終認識結果として確定した単語の前後にある単語のマッチング

完全マッチングによって最終認識結果として採用された単語の前、若しくは後ろにある認識結果中の単語を  $w_i$  とする。 $w_i$  に対して、PDF 文書の単語リスト中の単語  $W_p$  を以下の方法で求める：

(1)  $w_i$  の一つ前の単語  $w_{i-1}$  が最終認識結果として確定していた場合、 $w_{i-1}$  と完全一致する単語  $W(i-1)$  の一つ後ろの単語を  $W_p$  とする。

(2)  $w_i$  の一つ後ろの単語  $w_{i+1}$  が確定していた場合、 $W(i+1)$  の一つ前の単語を  $W_p$  とする。

(3)  $w_i$  の前後の単語  $w_{i-1}$  と  $w_{i+1}$  の両方も確定していた場合、

$$W_p = \begin{cases} N_M(w_{i-1}) \geq N_M(w_{i+1}) \\ \Rightarrow W(i-1) \text{ の一つ後ろの単語,} \\ N_M(w_{i-1}) < N_M(w_{i+1}) \\ \Rightarrow W(i+1) \text{ の一つ前の単語.} \end{cases}$$

以上によって求めた  $W_p$  と  $w_i$  を動的計画法を用いたマッチング (DP マッチング) にかけて、そのマッチングのコストが、 $w_i$  と  $W_p$  によって定まるしきい値よりも小さければ、 $W_p$  を  $w_i$  に対応する単語とする。

認識結果中の単語  $w$  と PDF 文書中の単語  $W$  の DP マッチングとそのコスト  $C_{DP}(w, W)$  は以下の手順で求める：

(1)  $w$  と  $W$  の文字数が同じ  $N$  個である場合、 $w$  中の  $i$  番目の文字を  $c_i$ 、 $W$  中の  $i$  番目の文字を  $C_i$  としたとき、すべての  $i = 1, \dots, N$  に対して、 $c_i$  と  $C_i$  が同じ文字であるか、類似文字であった場合、類似文字を考慮した単語の一致として、このときのコスト  $C_{DP}(w, W)$  を 0 とする。ここで用いている類似文字の組合せを、表 1 に示す。ここで一致しなかった場合は次の処理に移る。

(2)  $w$  の文字数を  $n$ 、 $W$  の文字数を  $N$  とする。 $i = 1, \dots, n$  と  $I = 1, \dots, N$  に対して、 $w$  の  $i$  文字目までと  $W$  の  $I$  文字目までのマッチングコスト  $C_{DP}(i, I)$  を以下で定義する；

$$C_{DP}(0, 0) = 0, C_{DP}(1, 0) = C_{DP}(0, 1) = 1,$$

$$C_{DP}(i, I) = \min \left\{ \begin{array}{l} C_{DP}(i-1, I) + 1, \\ C_{DP}(i, I-1) + 1, \\ C_{DP}(i-1, I-1) + C(i, I) \end{array} \right\}.$$

表 1 類似文字の組合せ  
Table 1 Sets of similar characters.

|  |
|--|
| (c, C), (s, S), (v, V), (w, W), (x, X), (z, Z) |
| (1, l (小文字のエル))                                |
| (0 (数字のゼロ), O (大文字のオー), o (小文字のオー))            |
| (- (マイナス), - (ハイフン), - (ロングハイフン))              |

ここで、 $C(i, I)$  は  $w$  の  $i$  番目の文字  $c_i$  と  $W$  の  $I$  番目の文字  $C_I$  との比較コストで、

$$C(i, I) = \begin{cases} 0, & c_i \text{ の認識結果の第 3 候補までに、} \\ & C_I \text{ の類似文字がある、} \\ 1, & \text{上記以外。} \end{cases}$$

としている。これらの値を求めて、 $w$  の  $n$  文字目までと  $W$  の  $N$  文字目までのマッチングコスト  $C_{DP}(n, N)$  を単語全体のマッチングコスト  $C_{DP}(w, W)$  とする。次に、この  $C_{DP}(w, W)$  が以下で定めるしきい値  $T_{DP}$  より小さければ  $w$  に対応する PDF 文書中の単語を  $W$  とする。更に、単語  $w$  の文字数が 6 以下と短い場合は、 $C_{DP}(w, W)$  がしきい値  $T_{DP}$  以上であっても、 $T'_{DP}$  より小さければ、 $w$  に対応する PDF 文書中の単語を  $W$  とする。ここでしきい値  $T_{DP}, T'_{DP}$  は、

$$T_{DP} := \min \left\{ 4, \frac{\max\{n, N\}}{3} \right\},$$

$$T'_{DP} := \frac{\max\{n, N\}}{2},$$

としている。

### 5.2.3 残りの確定していない単語のマッチング

ここまでのマッチングによって、対応する単語が見つからなかった認識結果中の単語  $w$  のうち、文字数が 4 以上の単語に対して、まだ対応する単語の決まっていない PDF 文書の単語リスト中の単語の中で、

$$C_{DP}(w, W) = \min\{C_{DP}(w, W') \mid |N(w) - N(W')| \leq 2\},$$

となる  $W$  を見つける。ここで  $C_{DP}$  は 5.2.2 で定義したマッチングコストで、 $N(w)$  や  $N(W')$  はそれぞれ、単語  $w$  と  $W'$  の文字数とする。 $C_{DP}(w, W)$  が以下のしきい値  $T''_{DP}$  より小さければ、 $w$  に対応する PDF 文書中の単語を  $W$  とする；

$$T''_{DP} := \min \left\{ \max \left\{ 2, \frac{N(w)}{3} \right\}, \frac{N(w)}{2} \right\}.$$

以上においても PDF 文書の単語リスト中に対応する

単語が見つからなかった認識結果中の単語は、PDF 文書中で欠落している単語であるとして、最終的な文字認識結果として残しておく。

### 5.2.4 PDF 文書から抽出した文字への外接方形の割当

マッチングによって対応づけられた単語のペア  $(w, W)$  に対して、どちらを最終的な認識結果とするかを判断する。PDF 文書中の単語  $W$  の各文字  $C$  に対して、マッチングの結果から、認識結果中の単語  $w$  中の文字の外接方形座標を割り当てていく。割り当てられた外接方形座標と  $C$  の文字種を用いて、*Geometric Position Checking* で  $W$  がテキスト部になり得るか判断し、テキスト部であるとなれば、 $W$  を最終的な認識結果として採用する。 $W$  の各文字に対する外接方形座標の割当方は以下のとおりである： $w$  中の文字  $c$  と  $W$  中の文字  $C$  との対応が 1:1 である場合：この場合、単純に  $c$  の外接方形を  $C$  にそのまま割り当てる。 $w$  中の複数の文字  $\{c_i, \dots, c_{i+n}\}$  が、 $W$  中の 1 文字  $C_p$  に対応している場合： $C_p$  の外接方形を、 $\{c_i, \dots, c_{i+n}\}$  の各文字の外接方形の和とする。 $w$  中の一つの文字  $c_i$  が、 $W$  中の複数の文字  $\{C_p, \dots, C_{p+n}\}$  に対応している場合：各  $C_q \in \{C_p, \dots, C_{p+n}\}$  の外接方形は、 $c$  の外接方形を縦に  $n+1$  等分した方形の  $q$  番目の方形とする。以上のようにして、PDF 文書中の各単語の文字に対して外接方形を割り当てていく。

### 5.3 数式部分の認識

上記のマッチング処理によって認識結果が変更されている可能性があるため、再度、数式解析を行い、数式部であるかそうでないかの判断を行う。数式であると判断された部分は、数式解析によって更に認識結果の変更が行われるが、テキスト部分に関しては、マッチングによる修正の結果が認識の最終結果として用いられる。

### 6. 認識結果の出力

以上によって生成された認識結果を再び元画像に埋め込み、テキスト情報をもった PDF 文書を生成する。このとき、数式は *Infty* により数学記号や数式構造も認識されているので、その情報は  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  や MathML で埋め込まれ、PDF 文書内の数式の読上げも可能となる。これにより、数式の情報が埋め込まれていない PDF 文書に、数式の情報を埋め込むことができ、よりアクセシビリティの高い PDF 文書を自動的に生成することができる。

## 7. 評価実験と結果

本システムの評価を行うために、OCRによって生成されたPDF文書を再認識することによって、どの程度PDF文書に含まれるテキスト情報の精度が上がるのかを調べた。方法としてはまず、我々が文字認識システム等の評価を行うために構築した正解データベース [7] を用いた。この正解データベースは、600 dpi でスキャンされた文書画像、各文字の外接方形、文字認識、文字種などの正解、数式の構造といった情報をもっている。その中の文書画像を、*ABBY FineReader 7.0* を用いて認識し、結果をテキスト情報をもつPDF文書として出力した。*FineReader* は英語だけでなく、様々なヨーロッパ言語の文書を非常に高い精度で認識することができるOCRシステムであるが、数式に関しては非常に簡単なもの以外認識することができない。これを数式認識をしていない一般的なテキスト情報をもったPDF文書の実験サンプルとして用い、正解データベースと比較し、認識率を求めた。次に、本システムに、この生成したPDF文書を再認識させ、その認識率を求めた。ここでの認識率とは、正解データベースがもつ各文字の方形座標に対して、認識結果中の同じ方形座標にある文字が等しければ正解であるとカウントし、それを全体の文字数で割ったものとしている。

今回の評価実験では、正解データベース中の数学雑誌15冊と教科書1冊から13文書(531ページ)を用いた。これらには787,421文字、内テキスト部621,998文字、数式部165,423文字が含まれている。

ここで実験サンプルPDF文書の正解率(認識率)を求める際にも、本認識手法と同様にPDF文書から抽出したテキスト情報を用いた。しかしながら、正解データベースと比較するには、各文字の外接方形が必要となってくる。そこで、*Infty* によって認識した結果とPDF文書から抽出したテキスト情報とを、先に述べた *Word Matching Method* を用いてマッチングさせる。このとき、テキスト部においては、PDF文書から抽出した文字を常に最終結果とすることによって、対応する *Infty* の認識結果にある外接方形座標をPDF文書から抽出した文字に割り当てる。このようにしてPDF文書中のテキスト情報に外接方形座標を追加し、正解データベースとの比較を行い、認識率を算出した。また、PDF文書から抽出されるテキストはASCII文字に限られるので、比較対象はASCII文字とし、更に、認識や区別が難しい、点類("、'、.

表2 もとのPDF文書の認識率と、PDF文書のテキスト情報を利用しなかった場合と利用した場合の認識率  
Table 2 Recognition rates of original PDF documents, recognition without text information and recognition using text information.

|                          | TOTAL  | TEXT   | MATH   |
|--------------------------|--------|--------|--------|
| Sample PDF               | -      | 99.75% | -      |
| Recognition without text | 99.18% | 99.71% | 97.18% |
| Recognition using text   | 99.22% | 99.79% | 97.09% |

‘) も対象から除き、ASCII文字だけでは区別できない、マイナス記号‘-’とハイフン‘-’は同一視した。

以上のようにして、まず生成した実験サンプルの認識率を、テキスト部(TEXT)のみ求めた。次に、PDF文書の実験サンプルを、埋め込まれているテキスト情報を使用せずに *Infty* で認識し、テキスト部(TEXT)、数式部(MATH)、そして両方を含めた全体の認識率(TOTAL)を求めた。最後に、実験サンプルのPDF文書のテキスト情報を用いて *Infty* で再認識し、テキスト部、数式部、全体の認識率を求めた。その結果を表2に示す。

この評価実験中、*FineReader* で生成した実験サンプルPDF文書には以下のような誤認識の傾向が見られた：(1) 数式を含む部分がレイアウト解析の際に図と認識されてしまい、その中にあるテキスト部の文字が認識されていない、(2) 認識に使用する言語情報が数式によって乱される、形が似ている文字、‘o’と‘O’、‘w’と‘W’など、が数式の直後にあると、小文字が大文字と認識されてしまう。(3) 縦に比較的大きな数式があると、行の大きさの取得を誤り、その後の文字認識が正確に行われない(図3)。これより、数式がテキスト部の文字認識に実際に悪影響を与えていることが確かめられた。

表2において、実験サンプルのもともとの認識率(精度)と、テキスト情報を利用した認識率を見ると、*Infty* による再認識によって、数式情報の付加だけでなく、テキスト部の認識率も向上していることが分かる。しかしながら、テキスト情報を利用しなかった場合と、テキスト情報を利用した場合を比較すると、テキスト部の認識率の向上が見られるが、数式部の認識率が落ちてしまっている。これは主に、マッチングの結果、数式部において、最終的な認識結果としてPDF文書のテキスト情報の方の結果が使われてしまったことによる。例を挙げると、“ $q' = dq \bmod I_Y$ ”という式が、*Infty* では、正しく認識されていた。一方、

PDF 文書中では、“ $q' = dgmod/y$ ”となっていた。これが、マッチングの際に、*Infty* の認識結果の中に ‘ $q'$ ’ の第 2 候補として、‘ $g'$ ’ が、‘ $I'$ ’ の第 2 候補として ‘ $/$ ’が入っていたために、DP コストがしきい値以下になり、PDF 文書のテキストが採用されてしまい、結果、“ $q' = dgmod/y$ ”と認識されてしまった。このように PDF 文書中のテキストが誤っていても、認識候補中に文字があれば、認識結果として採用されてしまう問題があった。

追加実験として、テキスト部に関しては誤りのない PDF 文書に対して、数式情報を埋め込む目的で使用する場合を想定して、上記の実験でも使用した正解データベースから生成した PDF 文書を本システムでもう一度認識し、その認識率を測定した。この生成した PDF 文書では、数式は  $\text{\LaTeX}$  のコマンドで記述されている。PDF 文書のテキストを利用して認識した場合、テキスト部、数式部共に認識率 100%を示し、この実験においては、正しく入っているテキスト部の情報を崩さず、数式を埋め込むことに成功していた。

読上げ順の改善について定量的な評価は行っていないが、実際に配布されている PDF 文書を本システムで認識して比較したところ、以下のような傾向が見られた。まず、数式によって引き起こされる読上げ順の乱れに関して、例えば数式が上下の行と交じってしまっている場合は、ほぼ正しく行分けが行われ、正しい順番に修正されていた。また、異なるレイアウト中のブロックのテキストが交ざり合っているような、極端な場合に関しても、正しくブロックを分けて出力していた。これらは認識の際に PDF 文書中の読上げ順序を使わず、*Infty* のレイアウト解析結果のみを用いたことがうまく働いたからであるが、もともと PDF 文書にタグ付け [2] されており、脚注が節の終わりに挿入されるなど、正しい読上げ順になっている PDF 文書を認識にかけた場合、*Infty* による読上げ順が適用されてしまい、脚注が文書中に挿入されてしまって、かえって読上げ順を悪くするという場合もあった。これは、*Infty* のレイアウト解析が、画像中の位置によって順番を決定するためだと思われる。

## 8. む す び

今回、PDF 文書から取り出したテキスト情報と、PDF 文書を画像として認識した認識結果を組み合わせることにより、より精度の高い情報を PDF 文書から得るシステムの提案とその手法の説明、更には、評

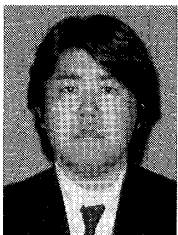
価実験も行った。既存の認識システムによって生成した PDF 文書を再認識することにより、本システムが PDF 文書中の数式部分にテキスト情報を追加するだけでなく、数式周辺の文字の誤認識もある程度修正することを示した。このシステムが提供するインタフェースを用いて、簡単に数式を含んだ PDF 文書を認識させ、数式の情報をもたない PDF 文書から、数式の情報をもった PDF 文書を生成することが可能である。しかしながら、今回 PDF 文書から抽出したテキスト情報は、*Ghostscript* の機能を利用して、ASCII テキストファイルから得るので、PDF 文書中にあるフォントや、位置といった有用な情報は使用することができなかった。今後、PDF 文書から直接情報を取得するようにすれば、認識率だけでなく、読上げ順に影響するレイアウト解析においても、より大きな効果が得られると期待できる。評価実験でも、数式部の認識においては、本手法の悪影響がわずかに見られ、これを改善するには、マッチングの対象となる単語の見直しや DP マッチングの際のコスト設定の見直しなどが必要である。また、読上げ順の改善についても、“適切な読上げ順”を考慮したレイアウト解析の改良が必要だと思われる。今回は英語文書のみ認識対象としたが、今後、特に日本語への適用を実装していきたい。

## 文 献

- [1] 渡辺哲也, “視覚障害者の Windows パソコン及びインターネット利用・学習状況,” 国立特殊教育総合研究所報告書, D-190, 2003.
- [2] 吉本浩二, 渡辺哲也, “スクリーンリーダーによる PDF 文書へのアクセシビリティについて,” ヒューマンインタフェース学会研究報告集, vol.5, no.5, pp.15–22, 2003.
- [3] “Web content accessibility guidelines 2.0,” <http://www.w3.org/TR/2006/WD-WCAG20-20060427/>
- [4] M. Suzuki, F. Tamari, R. Fukuda, S. Uchida, and T. Kanahori, “INFTY — An integrated OCR system for mathematical documents,” ACM Symposium on Document Engineering, pp.95–104, 2003.
- [5] “Ghostscript, ghostview and GSview,” <http://www.cs.wisc.edu/~ghost/>
- [6] Y. Eto and M. Suzuki, “Mathematical formula recognition using virtual link network,” Proc. ICDAR, pp.762–767, 2001.
- [7] M. Suzuki, S. Uchida, and A. Nomura, “A ground-truthed mathematical character and symbol image database,” Proc. 8th International Conference on Document Analysis and Recognition, ICDAR 2005, pp.675–679, Seoul, Korea, 2005.

(平成 18 年 6 月 27 日受付, 10 月 6 日再受付)



**金堀 利洋**

1997 岡山大・理卒。2002 九州大学数理学研究科博士課程単位取得後退学。同年、筑波技術短期大学障害者高等教育センター助手。2003 数理学博士。現在、筑波技術大学障害者高等教育研究支援センター助教授。主に視覚障害者支援システム開発に従事。

**鈴木 昌和 (正員)**

1969 京大・理卒。1971 同大学院理学研究科修士課程了。同年、同大学院理学研究科博士課程進学。1973 フランス政府給費留学生として渡仏、パリ大学第 11 部数学科博士課程。1975 フランス国立科学研究センター研究員。1977 Doctorat d'Etat ès Sciences (パリ大学 VII)。同年、九州大学工学部講師。現在、九州大学大学院数理学研究院教授。NPO 法人サイエンス・アクセシビリティ・ネット代表理事。主に、多変数関数解析、代数幾何、近年、科学技術文書認識、視覚障害者支援システム開発に従事。日本数学会、情報処理学会各会員。