厚生労働科学研究費補助金

感覚器障害研究事業

多機能高精度自動点訳エンジンの開発

平成 20 年度～22 年度　総合研究報告書

研究代表者　石川　准

平成 23（2011）年　5 月

厚生労働科学研究費補助金

感覚器障害研究事業

多機能高精度自動点訳エンジンの開発

平成２０年度～２２年度　総合研究報告書

研究代表者　石川　准

平成２３（２０１１）年　５月

目　次

厚生労働科学研究費補助金（感覚器障害研究事業）
総合研究報告書

多機能高精度自動点訳エンジンの開発

研究代表者　　石川　准　　静岡県立大学　国際関係学部　教授

研究要旨

本研究は、近年の以下のような新しいニーズの高まりに対応した多機能高精度自動点訳エンジンの開発を目的とする。

・各種ドキュメントが有する構造情報、レイアウト情報、テキスト情報等を生かした高精度自動点訳の実現

・大量コーパスを用いる今日的形態素解析エンジンの自動点訳への応用による自動点訳の変換精度の向上

・点字コーパスによる点訳分かち書き誤りの発見と修正

研究分担者

宮本　修（筑波技術大学 障害者高等教育研究支援センター　特任助教）

A. 研究目的

本研究は以下の３点を目的とする。

１．Open XML 等の構造情報を自動点訳に利用するための点訳 XML の策定とその有効性評価の研究

点訳 XML を策定し、Open XML やテキスト DAISY から点訳 XML への変換を行うモジュールを開発する。

２．点訳精度向上のための新しい形態素解析手法の研究

読みの誤り、分かち書きの誤り等の点訳誤りを項目別に自動的に算出できるツールを開発し、市販・公開されている点訳ソフトウェアの性能を客観的に評価する。

研究代表者がこれまで開発してきた既存の自動点訳エンジンの形態素解析アルゴリズムの性能を解析する。

大量のコーパスデータを用い、自動学習機能を有する今日的な形態素解析手法を解析し、それを応用して、確率論的方法論に基づく新しい自動点訳エンジンを開発する。

専門分野ごとにコーパスを整備し、ドキュメント種別判別の精度を高め、各専門分野特有の読み方に対応し点訳精度を向上させる。

３．点訳分かち書き誤りの発見の研究

全国視覚障害者情報提供施設協会が運営する視覚障害者情報総合ネットワークサピエの点字図書データを用いて大量点字コーパスを作成する。

点字コーパスを用いた分かち書き誤り検出手法を開発する。

以上の成果に基づき高度化するニーズに答える多機能高精度自動点訳システムを開発する。

B. 研究方法

上記目的を達成すべく以下のように研究を進める。

・B-1 点訳 XML による構造情報を活用した点訳（別紙参照）

・B-2 自動点訳結果評価システム（別紙参照）

・B-3 今日的な形態素解析の調査（別紙参照）

・B-4 形態素解析エンジン MeCab を使った自動点訳エンジンの開発と評価（別紙参照）

・B-5 点字コーパスによる分かち書き誤り検出と修正（別紙参照）

・B-6 複数候補の提示と修正 UI 開発（別紙参照）

（倫理面への配慮）

研究の過程で知り得た個人情報の守秘義務を遵守した。

C. 研究結果

点字コーパスによる点訳分かち書き誤り検出技術を開発し、性能を評価した。その結果分かち書き誤りの２３％から４６％を自動修正できた。これは実用的性能といえる。これは本研究の最大の成果である。

さらに点字コーパスは読み下しの誤りを減らすためにも利用できる。点字コーパスにまったく存在しない読みと多く存在する読みでは後者を選択する、あるいは複数候補の提示順を点字コーパスの情報をもとに決定するなどの応用が考えられる。複数候補提示インタフェースを既存の点訳システムに組み込む実装はすでに行っており、現在候補リストを点字コーパスに基づき提示する実装に着手して

いる。

一方、最新の形態素解析エンジンを改良して試作した自動点訳エンジンは、現段階では既存の点訳エンジンの変換精度に達しなかった。高い変換精度を達成するには、形態素解析の性能に加えて、コーパスまたは辞書の性能、例外処理など、多くの労力を要する。

２０万語の医学等専門辞書を整備し、ドキュメントの内容ごとに専門辞書を切り替えて用いることで点訳精度が向上することを確認した。

構造化点訳 xml の仕様を策定し、OpenXML ドキュメント等からのインポート機能を既存の点訳システムに実装し、構造情報、レイアウト情報を利用する自動点訳の「読みやすさ」向上を確認した。

D. 考察

今後点訳コーパスによる点字分かち書き修正機能を市販の自動点訳エンジン等に実装していく。

また自動点訳を利用しない点訳者に対しても、分かち書き誤りチェッカーを開発して提供すれば、点訳校正を効果的に支援できる。

E. 結論

テキスト・構造情報抽出、自動点訳、読み・分かち書きの修正の３段階それぞれにおける技術的改良を行った。それにより従来より多機能で高精度な自動点訳エンジンが実現した。

点訳コーパスによる分かち書き誤り判別技術には新規性があり、学術的価値がある。

一方読みの精度向上には、辞書の整備と形態素解析アルゴリズムの改良を地道に継続する必要があり、分かち書き誤りの修正のような顕著な効果を短期間で実現する方法はいまのところ見当たらない。ただし、点字コーパスは読みの確からしさの評価にも応用できると期待できるので、今後既存点訳システムに実装してその有効性を検証する計画である。

F. 研究発表

１．論文発表

宮本　修，コーパスを使った点訳の分かち書き誤り修正システム，情報処理学会論文誌 投稿中

２．発表

・石川　准，宮本　修，多機能高精度自動点訳エンジンの開発，平成 20 年度感覚器障害研究・研究成果発表会，KKR ホテル東京

・石川　准，宮本　修，多機能高精度自動点訳エンジンの開発，平成 21 年度感覚器障害研究・研究成果発表会，KKR ホテル東京

・石川　准，宮本　修，多機能高精度自動点訳エンジンの開発，平成 22 年度感覚器障害研究・研究成果発表会，KKR ホテル東京

G. 知的所有権の取得状況

なし

B-1 点訳 XML による構造情報を活用した点訳

■ 目的

点訳の対象となる、一般に利用されている文書の多くには構造情報が存在している。この構造情報とは、見出し情報、箇条書き属性、参照情報、目次情報、脚注情報、表、強調、ルビ、文書名やリリース日などのメタ情報を指す。点字ではこのような構造情報に対して、それぞれ独自の規則で構造情報を表現する。
従来の自動点訳システムでは、対象となる文書ファイルからテキストを抽出し、これを点訳するという処理で点訳が実現されていた。しかし、点訳に際して単なるテキストだけを処理対象としたのでは、構造情報に対応した正しい点訳が行えない。本研究では、対象となる文書ファイルから構造情報付きの点訳 XML を生成し、この点訳 XML を点訳することで、構造情報付の文書を正しく点訳する方法を実現した。

■ 研究内容

◆ 対象とした文書形式

今回構造情報付きの点訳対象とした文書形式は、Microsoft Word の Open XML 型式のファイルと DAISY 3 の文書型式である。
今日 Microsoft Word は、広く構造情報を表現した日本語文書作成に利用されている文書形式であり、一般のパソコン利用者にとってなじみ深い文書形式である。また、Microsoft Word 2007 以降は、Open XML としてその文書形式が公開されており、また、ISO/IEC 29500:2008 の国際規格となっている。Open XML を処理対象とすることで、一般の利用者にとって身近な構造情報付き文書を点訳できるようになった。
一方、DAISY 3 は、視覚障害者を始めとするプリント・ディスアビリティ向けの録音図書型式である。従来、DAISY 3 の文書を点訳する場合には、原文テキストを別途入手して点訳する必要があったが、本研究の成果により DAISY 3 の文書を直接構造情報付きで点訳できるようになった。

◆ 本システムの処理方法

本システムでは、直接点訳対象の文書形式を点訳するのではなく、前段階の処理として対象の文書形式を一度、テキストと構造情報を含む点訳 XML と呼ぶ型式に変換する。後段の処理として点訳エンジンは、この点訳 XML を構造情報を正しく扱いながら点訳を行う。
この構成は、システムを新たな文書形式に対応させるのに、前段の点訳 XML 変換部分を新たに開発するだけで良いというメリットがある。実際、Open XML 型式を実現した後に、DAISY 3 の文書形式への対応を行ったが、DAISY 3 独自の原ページ情報という新たな構造情報を追加した以外は、後段のシステムに対してはまったく手を加えずに機能の拡張が行えた。

■ 研究成果

本研究による成果は、2010 年 2 月にリリースされた Extra for Windows Version 5.1 に反映されている。Extra for Windows Version 5.1 の Open XML 型式の構造情報付き読み込み機能は、本研究に基づくものである。Open XML 型式のファイルを点訳する際には、内部的に点訳 XML ファイルを生成し、これを自動点訳する。Extra for Windows Version 5.1 には点訳 XML ファイルを直接開いて点訳を行う機能が用意されている。この機能を用いると、別のシステムによって生成された点訳 XML を読み込み、構造情報付きで点訳を行える。本研究により開発した DAISY3 の点訳機能は、点訳 XML を生成する前段のみのシステムである。このシステムで DAISY3 から生成した点訳 XML ファイルを、Extra for Windows Version 5.1 で読み込むと、DAISY3 の構造情報付きの点訳が実現できる。

■ 点訳 XML(Structured Braille Translation Input Format)仕様

◆ 構造化点字 XML とは

次世代 Extra 自動点訳エンジンでは、文書ファイルの見出しなどの構造情報を自動点訳に利用する。機能の実現には、構造情報付きの文書情報を入力ファイルとして扱えなければならない。構造情報付きの文書としては、Microsoft Word/Excel/PowerPoint, PDF, RSS, DAISY などの様々なファイルを想定している。
特定の文書ファイル形式に依存した形で直接点訳エンジン内で処理を行う実装を行うと、新たに他の文書ファイル型式に対応するにはまた同様の開発を行わなければならない。特定の文書形式を直接点訳エンジンが扱うのは汎用性の面で開発効率が悪いので、いったん実際の文書ファイルから構造情報付きのテキスト情報を XML ファイルの形式で抽出したものに対して Extra 自動点訳エンジンが点訳を行う構造とする。

本仕様書は、この中間形式の構造情報付き点訳入力 XML(Structured Braille Translation Input Format: 以降 SBTIF)の仕様を規定するものである。

◆ ファイルフォーマット
SBTIF ファイルは、「JIS X 4159:拡張可能なマーク付け言語(XML)」で規定される XML の規格に準拠するものとする。
ファイルの拡張子には.sbtif を用いる。

◆ XML 宣言、文字コード、DTD
ファイルの文字コードは、BOM 付きの UTF-8 とする。ファイルの開始は 16 進数表現で EF BB BF 3C 3F 78 6D となる。
XML 宣言は次のようになる。
<?xml version="1.0" encoding="utf-8" ?>
DTD 宣言は次のような記載となる。
<!DOCTYPE sbtif SYSTEM "sbtif-1.0.dtd" >

◆ 要素名、属性値、XML 構成の概要
SBTIF ファイルのサンプル例は次の通りである。

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE sbtif SYSTEM "sbtif-1.0.dtd" >
<sbtif
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:dcq="https://purl.org/dc/terms/" >
  <info>
    <orig_file_name>元文書ファイル名</orig_file_name>
    <filter>sbtif ファイルを生成したプログラム名</filter>
    ...略...
    <dc:title>タイトル</dc:title>
    <dc:language>ja</dc:language>
    <dc:creator>作者名</dc:creator>
    <dc:description>要約</dc:description>
    ...略...
  </info>
  <header>ヘッダー内容</header>
  <body>
    <para id="1" type="h1">第 1 章</para>
    <para id="2">本文</para>
    <para id="3" type="h2">第 1 節</para>
    <para id="4">本文</para>
    <para id="5" type="h3">第 1 項</para>
    <para>本文</para>
    ...略...
  </body>
</sbtif>
```

このように SBTIF ファイルの要素名、属性名は全て小文字を用いる。
また、SBTIF ファイルの要素名、属性値のうちのいくつかは Dublin Core を利用する。
基本構造として、sbtif 要素(html 要素ではない)が最上位に配置され、この下に info 要素、header 要素、body 要素の順で並ぶ。
info 要素の下には、文書全般に関する情報が格納され、body 要素の下には本文の内容情報が格納される。

要素名	意味
info	文書全体に関する情報

header	ヘッダー情報
body	本文情報

◆ info

info 要素の下には、文書全般に関する情報が格納される。info 要素の子ノードとしては、独自の要素と、Dublin Core で規定されるメタデータの 2 種類が配置される。子ノードの出現順序は任意である。

◆ 独自要素

要素名	意味
orig_file_name	変換元の文書ファイル名
filter	sbtif ファイルを生成したプログラム名

◆ Dublin Core 要素

Dublin Core で規定される文書のメタデータを記述する。
sbtif 内での基本 Dublin Core 要素は dc:で修飾した名前空間を使用する。
また、拡張 Dublin Core 要素は dcq:で修飾した名前空間を使用する。

要素名	意味
dc:title	タイトル、リソースに与えられた名前
dc:creator	リソースの内容に主たる責任を持つ人や組織
dc:subject	リソースの内容に含まれるトピック(キーワード、分類コード)
dc:description	要約や説明文
dc:publisher	リソースを利用可能にする責任者(個人、組織、サービスなど)
dc:contributor	リソースの内容に協力、貢献している人や組織、サービス
dc:date	リソースの作成日、公開日など主要な出来事に関する日
dc:type	分野コード
dc:format	元文書ファイルの MIME Type
dc:identifier	リソースへの曖昧さのない参照(URI, ISBN)
dc:source	リソースが派生作品であるときの元リソースへの参照
dc:language	リソースの言語(ISO 639 の言語コードを用いる)
dc:relation	関連するリソースへの参照
dc:coverage	リソースの内容がカバーする範囲もしくは対象
dc:rights	権利に関する情報
dcq:created	作成日
dcq:issued	正式発行日
dcq:modified	更新日

日付のフォーマットは ISO-8601/W3C-DTF 形式(http://www.w3.org/TR/1998/NOTE-datetime-19980827)を使用する。

年のみ
　　YYYY (例: 1997)
年月
　　YYYY-MM (例: 1997-07)
年月日
　　YYYY-MM-DD (例: 1997-07-16)
年月日+時分+タイムゾーン
　　YYYY-MM-DDThh:mmTZD (例: 1997-07-16T19:20+01:00)
年月日+時分秒+タイムゾーン
　　YYYY-MM-DDThh:mm:ssTZD (例: 1997-07-16T19:20:30+01:00)
年月日+時分秒+1/100 秒+タイムゾーン

YYYY-MM-DDThh:mm:ss.sTZD (例: 1997-07-16T19:20:30.45+01:00)
dc:format の値には次に示すような MIME Type を指定する。

拡張子	MIME Type
.doc	application/msword
.xls	application/msexcel
.ppt	application/mspowerpoint
.pdf	application/pdf
.rtf	application/rtf
.docx	application/vnd.openxmlformats-officedocument.wordprocessingml.documen
.xlsx	application/vnd.openxmlformats-officedocument.spreadsheetml.sheet
.pptx	application/vnd.openxmlformats-officedocument.presentationml.presentation

dc:language の値には ISO 639 で規定される次のような 2 文字の言語コードを指定する。

コード	言語
en	英語
ja	日本語

dc:language の情報は、点字に変換する際に点訳モードの自動判定に利用される。
dc:type の情報は、点訳を行う際の専門辞書の自動判定に利用される。dc:type の分野コード情報は SBTIF の変換プログラムに対する引数、オプションなどの形式で指定できることが奨励される。現在計画している分野コードは次の通りである。

分野コード	分野名
medicine	医学
jurisprudence	法学
sociology	社会学

◆ 必須 info 子要素
info 子要素として必須なものは次の通りである。

要素名	意味
orig_file_name	変換元の文書ファイル名
filter	sbtif ファイルを生成したプログラム名
dc:title	タイトル、リソースに与えられた名前
dc:language	リソースの言語(ISO 639 の言語コードを用いる)
dc:format	元文書ファイルの MIME Type

これ以外の要素については省略が可能である。

◆ body
body 要素の下には、文書内での出現順に段落、表のようなブロック要素が複数個配置される。
ブロック要素として利用できるものは次の通りである。

要素名	意味
para	段落
table	表

◆ para 要素
次の例のように para 要素は段落を表現する。

<para id="1" type="h1">詩の書き方</para>
<para id="2" >七・五調などの定型詩は五マス目から書き出する。</para>
<para id="3" >定型詩の表題は書き出し五マス目と区別するため七マス目から書き出する。</para>

---

para 要素の子要素には段落内容が記述される。段落の内容にはテキストの他、後述するインライン要素が含まれる。
para 要素は必ず id 属性を持つ。id 属性値にはユニークな数字(重複の無い数字)が設定される。
段落には type 属性を付与できる。type 属性の無い段落要素は、通常の本文として扱われる。
type 属性の値として指定できるのは以下の通りである。
属性値 意味・役割

---

h1　見だし 1
h2　見だし 2
h3　見だし 3
h4　見だし 4
li1　箇条書き 1
li2　箇条書き 2
li3　箇条書き 3
title　タイトル
center センタリング
right　右寄せ
hr　水平線
index　目次

---

h1 は一番大きな単位の見だしである。h2 は h1 より一段階小さな単位の見だし、h3 は h2 よりさらに小さな単位の見だし、h4 は h3 よりさらに小さい単位の見だしである。
li1 は箇条書きに用いる。箇条書きの中に、箇条書きを入れ子にするには li2 を用いる。li2 の下にさらに箇条書きを入れ子にするには li3 を用いる。
h1～h4、li1～li3、index の type 属性を持つ para 要素には、次の属性が付与される。
属性　意味・役割

---

intro　前置文字列
startnum 開始数字番号

---

startnum 属性で、章節などの番号や、番号付きリストの開始番号を指定できる。
startnum 属性が明示されていない場合には、直前の同じ属性の番号をインクリメントした数値が使用される。
intro 属性値は段落の前置文字列を意味する。前置文字列中の数字は、各レベルの番号に置き換わりる。
番号付き箇条書きの例:

---

1 大会概要:東京マラソン 2009
2 主催:(財)日本陸上競技連盟、東京都
3 開催日時: 2009 年 3 月 22 日(日)
4 制限時間: マラソン:7 時間、10km: 1 時間 40 分

上記のような番号付きの箇条書きを para 要素で表現すると、次のようになる。

<para id="10" type="li1" intro="1 " startnum="1">大会概要:東京マラソン 2009</para>
<para id="11" type="li1" intro="1 ">主催:(財)日本陸上競技連盟、東京都</para>
<para id="12" type="li1" intro="1 ">開催日時: 2009 年 3 月 22 日(日)</para>
<para id="13" type="li1" intro="1 ">制限時間: マラソン:7 時間、10km: 1 時間 40 分</para>

番号無し箇条書きの例:

・大会概要:東京マラソン 2009
・主催:(財)日本陸上競技連盟、東京都
・開催日時: 2009 年 3 月 22 日(日)
・制限時間: マラソン:7 時間、10km: 1 時間 40 分

上記のような番号無しの箇条書きを para 要素で表現すると、次のようになる。

```
<para id="10" type="li1" intro="・">大会概要:東京マラソン 2009</para>
<para id="11" type="li1" intro="・">主催:(財)日本陸上競技連盟、東京都</para>
<para id="12" type="li1" intro="・">開催日時: 2009 年 3 月 22 日(日)</para>
<para id="13" type="li1" intro="・">制限時間: マラソン:7 時間、10km: 1 時間 40 分</para>
```

◇ 目次

type 属性値に index が指定される目次に、どの段落に対する目次なのかという情報が重要である。
参照先の段落を示すために、次のような idref の属性を使用する。

属性	意味・役割
idref	段落番号

idref の属性値には、参照先段落の id 値を指定する。
例:

```
<para id="10" type="index" idref="20">第一章 : 都市の生活</para>
<para id="11" type="index" idref="34">早朝の風景</para>
<para id="12" type="index" idref="36">昼間の風景</para>
<para id="13" type="index" idref="50">第二章 : 町中探検</para>
<para id="14" type="index" idref="55">銭湯めぐり</para>
<para id="15" type="index" idref="62">公園めぐり</para>
```

◆ table 要素

表を記述するのには table 要素を使用する。
table 要素の構成は、XHTML の table にほぼ準拠したものになる。

```
<table id="58">
<tr><th>セル見だし 1</th><th>セル見だし</th></tr>
<tr><td>セル内容</td><td>セル内容</td></tr>
<tr><td>セル内容</td><td>セル内容</td></tr>
<tr><td>セル内容</td><td>セル内容</td></tr>
</table>
```

table 要素に対して id 属性が必ず付与される点が XHTML とは異なる点である。
id 属性値は、para 要素と同様にユニークな数字が設定される。この id 属性値は、para 要素の id 属性値とは重複しない。
テーブルで使われる要素は次の通りである。

要素	意味・役割
tr	テーブルの行
th	テーブルのセル見だし
td	テーブルのセルデータ

◆ インライン要素
元文書の段落中に何らかの属性情報が付いている場合には点訳時にこの情報を生かせる可能性がある。
このような、段落中の属性情報をインライン要素と呼ぶ。
インライン要素には次のような要素がある。

要素	意味・役割
em	段落のデフォルト属性とは異なる属性が付いている
a	ハイパーリンク
ruby	ルビ
page	ページ区切り

◇ em 要素
元文書の段落中で、特定の箇所だけ特殊な属性が付いている場合には該当箇所を em 要素の子要素として表現する。
なお、段落全体に同じ属性が付与されている場合は、構造情報としての意味合いが薄いので、em 要素を用いての属性を表現は行わない。
em 要素には、必ず type 属性が指定され、元文書のどのような表現であったかを示す。

type 属性値	意味・役割
italic	イタリック
bold	ボールド
under	アンダーライン
color	文字色/背景色の変更
font	フォント・サイズの変更
strike	打ち消し線
other	その他の属性変更（網掛けなど）

例：

```
<para id="201"><em type="under">マラソン:</em>東京都庁～飯田橋～皇居前～日比谷～品川～銀座～日本橋～浅草雷門～築地～豊洲～東京ビッグサイト(日本陸上競技連盟/AIMS 公認コース) </para>
<para id="202"><em type="under">10km:</em>東京都庁～飯田橋～皇居前～日比谷公園（公認条件に適合せず記録は公認されない)</para>
```

この例では、段落頭の「マラソン」と「10Km」の 2 箇所がアンダーラインの属性が付与されていることを示する。
◇ a 要素
ハイパーリンクを示すのに a 要素を用いる。
ハイパーリンクの例:

```
<para id="133">皇居マラソン参加のお申し込みは、インターネットの<a href="http://www.foo.bar/">登録サイト</a>よりお願いする。</para>
```

◇ ruby 要素
ruby 要素はルビを表現するのに使用する。
ruby 要素の下にルビの対象範囲を示す rb 要素と、ルビ文字列を示す rt 要素を用いて表現する。
(XHTML ではルビのタグを解釈できないブラウザ用に rp 要素を用いるが、SBTIF では使用しない。)
ルビの例:

```
<para id="203"><ruby><rb>米国</rb><rt>アメリカ</rt></ruby>は映画産業が盛んである。</para>
```

◇ page 要素
点訳では必要に応じて原文のページ番号を付加することがある。これを実現するため、新しいページが始ま

ったことを page 要素を使って示する。
page 要素には、ページ番号を示す pagenum 属性が必須である。

<para id="433">組織委員会を行い、2009 年の<page pagenum="10" />大会要項が決定しましたのでお知らせする。</para>

◆ header 要素
header 要素には元文書のヘッダーの内容を記載する。
例:

<header>マラソン大会参加概要</header>

---

header 要素の子要素にはインライン要素を含められる。

◆ Microsoft Word の Open XML ファイル(.docx)からの情報抽出
Microsoft Word の Open XML ファイル(.docx)からの情報抽出に関する注意事項は次の通りである。

◆ 修正記録
修正記録付きの文書ファイルの場合には最後の確定内容のみが抽出され、過去に削除された文字列情報、スタイル情報などは抽出しないものとする。

◆ 見だしレベル・箇条書きレベル
Microsoft Word の見だしレベルは 1～9 までが定義されている。SBTIF に見だしレベルの情報を変換するにあたっては、1～4 のみを対象とし、これより大きな見だしレベル 5～9 は通常の本文と同様として扱いる。
また、箇条書きレベルも Microsoft Word では 1～5 までが定義されている。こちらも 4～5 の箇条書きレベルは本文と同様として扱いる。

◆ 文末脚注、脚注、引用
・文末脚注：<annotation>
・脚注：<footnote>
Word の[参考資料]-[脚注の挿入]、[文末脚注の挿入]で設定された部分
<para id="1">脚注が付いている文章<annotation>脚注の内容</annotation></para>
・引用：<blockquote>
スタイルが、「引用文」もしくは「引用文 2」に設定されている部分
<para id="2"><blockquote>引用を設定している部分</blockquote></para>

◆ 数式
・数式：<math>

◆ プロパティ情報
Word の文書プロパティ情報は、次のように Dublin Core 要素に対応させる。
要素名　　Word の文書プロパティ情報

---

dc:title ファイル概要-タイトル
dc:creator　　ファイル概要-作成者
dc:subject　　ファイル概要-分類
dc:description ファイル概要-サブタイトル
dc:publisher　ファイル概要-管理者
dc:contributor
dc:date　　ファイル概要-更新日時
dc:type　　(変換プログラム側で指定)
dc:format　　application/vnd.openxmlformats-officedocument.wordprocessingml.documen
dc:identifier

dc:source	
dc:language	(ja もしくは en を指定)
dc:relation	ユーザ設定-リファレンス
dc:coverage	
dc:rights	
dcq:created	ファイル概要-作成日時
dcq:issued	ユーザ設定-完了日
dcq:modified	ファイル概要-更新日時

◆ セクション区切り

Open XML ではセクション区切り情報を用いて、文書レイアウトを変更している。
Word の SBTIF 化にあたっては、セクション区切りはなにも情報を抽出しない。

◆ カスタム XML マークアップ

Open XML ではカスタム XML マークアップというしくみで、ユーザ定義の XML 情報を Open XML 中に埋め込める。
Word の SBTIF 化にあたっては、カスタム XML マークアップに関してはなにも情報を抽出しない。

B-2 自動点訳結果評価システム

■ 目的

従来、自動点訳の結果に対して正しい点訳結果かどうかを判断するには、熟練した点訳者が人手でチェックの作業を行う必要があった。この方法では、人手での作業であるため、大量の点訳結果に対する判定が難しいことや、点訳者の違いによる判断の違い、チェック漏れなどの問題があった。

自動点訳システムの点訳精度を向上する場合、ある試みが点訳精度の向上に結びつくのかどうかという判断は、できるだけ大量の点訳結果を機械的に判定し評価するシステムが欠かせない。そこで、本研究では、熟練した点訳者が点訳した点訳結果をお手本とし、このお手本データに対して、自動点訳結果のデータを突き合わせ、どのような誤りがあるかを自動的に集計するシステムを開発した。

この自動点訳結果評価システムによって、今回の一連の点訳精度の向上に対する研究が、正しい精度向上に貢献する改良あるいは取り組みであったのか、あるいはそうでなかったのかといった点を客観的に評価できるようになった。

■ 研究内容

熟練した点訳者による点訳結果をNABCCコードのテキスト型式で保存する。また、評価対象の自動点訳の結果についてもNABCCコードのテキスト型式で保存する。

NABCCコードは北米点字コードと呼ばれるコード体系で、点字を表現するために点字プリンタ、点字ディスプレイなどで用いられる汎用的な点字コードであり、6点点字の全てのパターンを表現できる。NABCCコードを使用することで、特定のシステムに依存しない評価システムが実現できる。

保存した2つのNABCCコードのテキストファイルを引数に指定して、btevalという集計プログラムを起動すると、即座に集計情報が出力される。集計内容には、行(段落)単位の評価結果と、文全体に関する評価結果（サマリー情報）がレポートされる。評価方法のアルゴリズムとしては、UNIXシステムのdiffで知られる差分抽出アルゴリズムを元に、点訳評価用に独自の工夫を加えたアルゴリズムを用いている。

行単位の評価結果としては、行内のどの部分が誤りであったか、また誤りの内容として誤りワード数、切りすぎ、切らなすぎ、切り間違い、読み間違えの数がそれぞれレポートされる。

文全体に関する評価結果としては、トータルの行数、トータルのワード数、違いのあった行の数、違いのあった行の割合、切りすぎ箇所の数、切らなすぎ箇所の数、切り間違いの箇所の数、読み間違え箇所の数、切りすぎのあった行の数、切らなすぎのあった行の数、切り間違いのあった行の数、読み間違えのあった行の数についてレポートされる。

■ 研究成果

自動点訳結果評価システムを用いて、一連の研究で開発した改良版自動点訳エンジンの性能評価、コーパスを用いたマス開け補正システムの性能評価、汎用的な形態素解析エンジンであるMeCabを用いた自動点訳エンジンの性能評価を行うことができた。これらの性能評価は、点訳者による属人的な評価結果ではなく機械的な評価結果であるため、極めて客観性が高く、評価精度も高いものであると考えられる。

■ 自動点訳結果評価システムの使用方法

コマンドラインよりbtevalというコマンドを使い、2つのNABCCコードの点字ファイル(.BRL形式)の内容を比較分析し、結果をレポートする。
片方のファイルを正しく点訳された内容とし、もう一つの点訳内容を検証する。

◆ コマンドラインオプション

bteval のコマンドラインオプションは次の通り。

bteval [options]
--version, -V　バージョン情報を表示。
--help, -h　コマンドラインヘルプを表示。
-T　プログラムの内部テストを行う。
-l=filename　手本となる NABCC ファイルを指定。(必須)
-t=filename　評価対象の NABCC フィアルを指定。(必須)
-s=filename　点訳元のテキストファイルを指定。(省略可能)
-r=filename　レポートを出力する先のファイルを指定。
指定しない場合には標準出力に出力。
-d　合致しなかった行に関する差分解析情報をレポートする。
-i　行頭・行末の空白文字を無視。

◆ レポートファイルの形式

○ 行の差分解析情報(-d オプションを付けた場合)

-d オプションを付けて bteval コマンドを実行した場合にはレポートファイルに手本と評価対象の行が合致しなかった行に関する、差分解析情報も出力される。

行の差分解析情報は次のように 5 行に渡って内容が表示される。
点訳元のテキストファイルを指定した場合には 6 行に渡って内容が表示される。

line:1　← 行番号(何行目で差分が生じたかを示す。)
text:赤とんぼ　← 点訳対象テキストの該当行内容
lead:A* T0"!　← 手本の行内容
target:A*T0"!　← 評価対象の行内容
> =|A*| -| | =|T0"!| .　← 差分内容
word:2 too much cut:0　too short cut:1　bad cut:0　bad reading:0
←ワード数、切りすぎ、切らなすぎ、切り間違い、読み間違えの数

差分内容は次のような表現で構成される。
文字列は|で囲んで表現する。(|は大文字系の NABCC では使われないため)

> ← 行頭を示す。
. ← 行末を示す。
=文字列　← 手本と評価対象に共通して存在する文字列であることを示す。
-文字列 ← 手本から削除された部分文字列であることを示す。
+文字列 ← 手本に追加された部分文字列であることを示す。

切りすぎ、切らなすぎはそれぞれ空白文字のみで構成される文字列が、何カ所追加されているかを調べる。
読み間違えは、削除と追加が連続している箇所に着目してカウントを行う。

○ サマリー情報

サマリー情報はレポートファイルの最後に全体を通した情報が記録される。
それぞれ、以下のような内容が記録される。

Summary information.
The lead NABCC file:akatonbo-extra51.brl ← 手本の NABCC ファイルの名前
The target NABCC file:akatonbo-ibukiTenC.brl ← 評価対象の NABCC ファイルの名前
The source text file:akatonbo.txt ← 点訳元のテキストファイルの名前
Total line number:33 ← トータルの行数

Total word number:207 ← トータルのワード数
Total different line number:33 ← 違いのあった行の数
Different lines percentage:100% ← 違いのあった行の割合
Total too much cut points:42 ← 切りすぎ箇所の数
Total too short cut points:21 ← 切らなすぎ箇所の数
Total bad cut points:3 ← 切り間違いの箇所の数
Total bad reading points:17 ← 読み間違え箇所の数
Total too much cut lines:33 ← 切りすぎのあった行の数
Total too short cut lines:16 ← 切らなすぎのあった行の数
Total bad cut lines:3 ← 切り間違いのあった行の数
Total bad reading lines:11 ← 読み間違えのあった行の数

■ 自動点訳結果評価システムの後処理システムの使用方法

後処理システムは、btevalの出力情報を人間に分かりやすいように翻訳するシステムである。

btevalの出力情報はNABCCコードにより表記されており、人間には分かりにくい。このNABCCコードをひらがなに直し、表示する。
また差分情報から誤りの種類を解析し、表示する。

◆機能と起動方法

起動は、コマンドラインよりReportFormatterというコマンドを指定する。
入力はリダイレクトでbtevalの出力ファイルを指定する。
出力もリダイレクトにより適切なファイル名を指定する。

○ 起動の例

>ReportFormatter < in.txt > out.txt

◆ 出力形式

出力形式はbtevalの出力に、以下の項目が追加されたものである。

・手本点訳のひらがな表記
・評価対象点訳のひらがな表記
・誤り解析情報

誤り解析情報は以下のとおりである。
[誤りの種類] [手本となる点訳の該当部分のひらがな表記]<-->[評価対象の点訳の該当部分のひらがな表記]

このうち、[誤りの情報]は以下の4種類である。

1.キリ(切りすぎ、切らなすぎ、切る位置の相違)
2.カッコ
3.読み
4.読みキリ

1.の「キリ」はマス空けの誤りである。これには「切りすぎ」、「切らなすぎ」、「切る位置の相違」の3通りが含まれる。「切る位置の相違」とは、例えば「回転待ち時間」が正解では「かいてんまち　じかん」となっていたが、対象点訳では「かいてん　まちじかん」と誤っていた場合に、「切りすぎ」誤り1個と「切らなすぎ」誤り1個の合計2個ではなく、「切る位置の相違」誤り1個とカウントする。
「切りすぎ」は「キリM」
「切らなすぎ」は「キリS」

「切る位置の相違」は「キリ B」
と、それぞれ表記する。

2.の「カッコ」は、マス空けの誤りのうち、カッコの処理によるものについては、特に「カッコ」と出力したものである。"(" (丸カッコ開き)は、点字規則では、内容により前と続ける場合と、前をひとマス空ける場合の二通りがある。この誤りは文脈に依存する場合が多いので、通常のマス空けの誤りとは区別して出力している。

3.の「読み」は読みの相違による誤りである。

4.の「読みキリ」は読みの相違とマス空けの誤りの複合である。

◆出力例

text:赤とんぼ
あか　とんぼ　←手本点訳をひらがなで表記したもの。後処理システムにより追加
lead:A* T0"!　←手本点訳を NABCC コードで表記したもの。
target:A*T0"!　←評価対象点訳を NABCC コードで表記したもの。
あかとんぼ　　←評価対象点訳をひらがなで表記したもの。後処理システムにより追加
>=|A*| -| | =|T0"!| .
キリ　あか　とんぼ　<-->　あかとんぼ
↑エラーの種類、手本点訳、評価対象点訳のそれぞれ該当部分をひらがなで表記したもの。後処理システムにより追加

B-3 今日的な形態素解析の調査

■ 目的
石川によって開発を続けている EXTRA 自動点訳エンジンは、点訳精度の面で一定の成果をあげているが、さらなる精度向上には形態素解析部分が大きな弱点になっていると考えられる。そこで、点訳エンジンの精度向上のために、今日的な形態素解析を利用して自動点訳を行う方法について検討するため、今日的な形態素解析手法に関しての調査を行った。

■ 研究内容
本格的な日本語情報処理のさきがけである、形態素解析システム JUMAN を始め、ChaSen(茶筌)、MeCab(和布蕪)といったオープンソースの形で入手できる形態素解析システムについて自動点訳システムとして利用できるかどうかという視点で調査及び評価を行った。

■ 研究成果
□ 第 1 章 はじめに

◆ 1-1 目的
今後の次世代自動点訳ソフトウェアの開発に向けて、自動点訳処理の大きな部分を占める形態素解析手法の調査を行った。本調査報告書は、この調査結果を報告するものである。

◆ 1-2 自然言語処理の具体例
今日、多くの分野でコンピュータを使った自然言語の処理が行われている。具体的な例を挙げると以下のような分野で、自然言語処理が利用されている。
・ かな漢字入力
・ 自動翻訳
・ 音声合成システム
・ 音声認識システム
・ 全文検索エンジン
・ 自動要約
・ 自動応答システム
分野によっては、既に実用の領域に入って何年も経過しているものもあれば、まだ十分利用できるレベルに至っていない分野もある。また、自然言語の種別によって実用度に大きな開きがある。
日本語は、欧米の言語と異なり、たとえば単語と単語がマス開けされない言語であるため、単純な単語を認識するだけでも大きな計算処理が必要とされる特性や、主語が省略される度合いが大きいなど、計算機による処理が困難な部類の言語である。

◆ 1-3 自動翻訳システムの構造
自然言語処理の方法には様々な方法が考案されているが、自動翻訳を例に取ると、次のような 5 ステップにより、原文を解析するシステムが多い。
・ 形態素解析
・ 構文解析
・ 格解析(意味解析)
・ 変換処理
・ 生成処理
形態素解析は、入力文を単語に分割し、その単語の品詞を決定することである。この処理は、対象言語によって内容と難しさが異なる。英文では各単語は空白文字でわかち書きされるため、単語の切れ目を解析には多くの処理を必要としないが、日本語の場合には、単語間の特別な区切り文字が存在しないため、辞書を用いて解析を行いながら分かち書きを行い、同時に品詞を決定する必要がある。
構文解析は、形態素解析の結果得られた品詞列に基づき、原文の構文的構造を分析し、構文解析木として表現する処理を指す。構文解析木は、品詞列を文法に基づいて分析することで得られる。しかし、文法だけからは文に対する構文解析木を一意に求めることができない場合が存在し、これを係受けの曖昧さと呼ぶ。構文解析の前段階である形態素解析の結果が複数候補考えられるような場合には、係受けの曖昧さと相乗的に複数の解析結果の候補が得られる。