



## 概 要

従来の文書の編集機能には様々なワープロソフトが用いられているが、それらのシステムではキーボードやマウスなどの間接的な手法をとってしまっている。また、現在ではタブレット PC などのペン入力インタフェースが多くなってきているため、ペン入力による文書編集手法の重要性は増してきていると考えられる。

そこで本研究では、手書きによる校正記号を書き込むことでの文書の校正が可能なシステムを提案した。これを利用することで、紙の上での手書きの感覚で文書の校正を行うことができる。

# 目次

第1章	はじめに	1
1.1	研究の背景	1
1.2	現在用いられている編集システム	1
1.2.1	一般的なワープロソフト	1
1.2.2	関連研究	3
1.3	研究の目的	4
1.4	論文の構成	4
第2章	手書き校正記号による文書編集システムの提案	5
2.1	本システム的设计	5
2.2	校正記号の例	6
2.3	使用する校正記号	8
2.4	表示画面の構成	9
第3章	試作システムの実装	11
3.1	実装環境	11
3.2	システム的设计	11
3.2.1	画面の表示	11
3.2.2	主要クラスの説明	11
3.3	手書き校正記号の解析	12
3.3.1	改行	12
3.3.2	削除	13
3.3.3	挿入	15
3.3.4	入れ替え	18
3.4	校正結果の表示	19
第4章	実験	20
4.1	システムの使用例	20
4.2	評価	20
4.3	今後の課題	21
第5章	まとめ	22
	謝辞	23
	参考文献	24

## 図表目次

図 1-1	Microsoft Word での編集例	2
図 1-2	Microsoft Word の校正機能使用例	3
図 2-1	システム使用イメージ	6
表 2-2	校正記号の例	8
図 2-3	校正記号：改行	8
図 2-4	校正記号：削除	9
図 2-5	校正記号：挿入	9
図 2-6	校正記号：入れ替え	9
図 2-7	表示画面の構成	10
図 3-1	改行：ストロークの入力	12
図 3-2	改行：校正結果	13
図 3-3	削除：ストロークの入力	14
図 3-4	削除：校正結果	14
図 3-5	挿入：ストロークの入力	15
図 3-6	挿入：挿入要素の入力	16
図 3-7	挿入：校正結果	17
図 3-8	挿入：挿入要素の再編集	17
図 3-9	入れ替え：ストロークの入力	18
図 3-10	入れ替え：校正結果	19
図 4-1	システムの動作実験	20

# 第1章 はじめに

## 1.1 研究の背景

近年、計算機は様々な社会に広く普及してきた。それによって、様々な文書を書く際にも計算機を用いて書かれることが多くなってきている。また、プラズマディスプレイなどの大画面ディスプレイなども普及してきていて、電子ホワイトボードシステムとして使用されることも多くなっている。このような場合に、キーボードやマウスなどの入力インタフェースでは大きな画面に対しては扱いづらいため、画面に対して直接的な操作が可能となるペン入力インタフェースが注目されてきている。しかし、計算機で行われる作業として一般的である文書の編集において、ペン入力インタフェースが活かされているものは少ない。

また、計算機上での文書の編集が主流になっている今でも、文書の校正の際には一度プリントアウトし、紙の上で校正作業を行うということも多く行われている。これは、文書に対して手で校正を行うということが重要視されているということだと考えられる。ここでいう校正とは、ある程度まで完成された文書に対して手直しをすることであり、編集とは違う意味で扱っている。

そこで本論文では、計算機上での校正という作業に焦点をおき、ペン入力を利用して手書き校正記号を画面内に書き込むことで校正を行うシステムを提案した。

## 1.2 現在用いられている編集システム

### 1.2.1 一般的なワープロソフト

一般的に用いられる様々な文書編集ソフトを用いることで、単純な文書を作成・編集することは可能である。しかし、文書の校正という面において支援されているものは少ない。また、ワープロソフトを使用する際の入力にはキーボードやマウスが使用される。それらの入力は画面に対して直接的ではなく、カーソルの移動などに手間がかかってしまうことがある。

現在広く用いられている Microsoft Word[1]は、文書編集に関して様々な機能が追加されている。しかし、図 1-1 に示すように、様々な機能が追加されていくと同時にメニューなどの項目が複雑化していき、メニューを選択するまでの時間が大きくなってしまったり、メニューを間違えて選択しやすくなったりするなど、結果としてその機能を利用するためにユーザにかかる負担が大きくなっている。

またそれらのエディタの共通の問題点として、文書の校正を行うと校正する前の画面が失われてしまい、文書全体のイメージやレイアウト等を忘れてしまう可能性があるということが挙げられる。例えば、文章を書き直すために、一度ある程度まで完成させた文章を消して書き直したとする。すると書き直した後の文書と書き直す前の文書を比べ

ように思っても、書き直す前の文書はすでに失われてしまっているために比較ができず、その書き直しの結果文書がより良くなっているのか、悪くなっているのかの判断もしづらくなる。この問題はアンドゥ・リドゥ機能で多少解消されるが、根本的な部分での解決にはなっていないと思われる。

図 1-2 に示されるように、Microsoft Word には文書校正を支援する機能が付属されていて、校正を行った箇所やその校正の種類などが示されるようになっている。しかし、やはり入力インタフェースとしてはキーボードが用いられ、画面に対して直接的に操作することはできない。

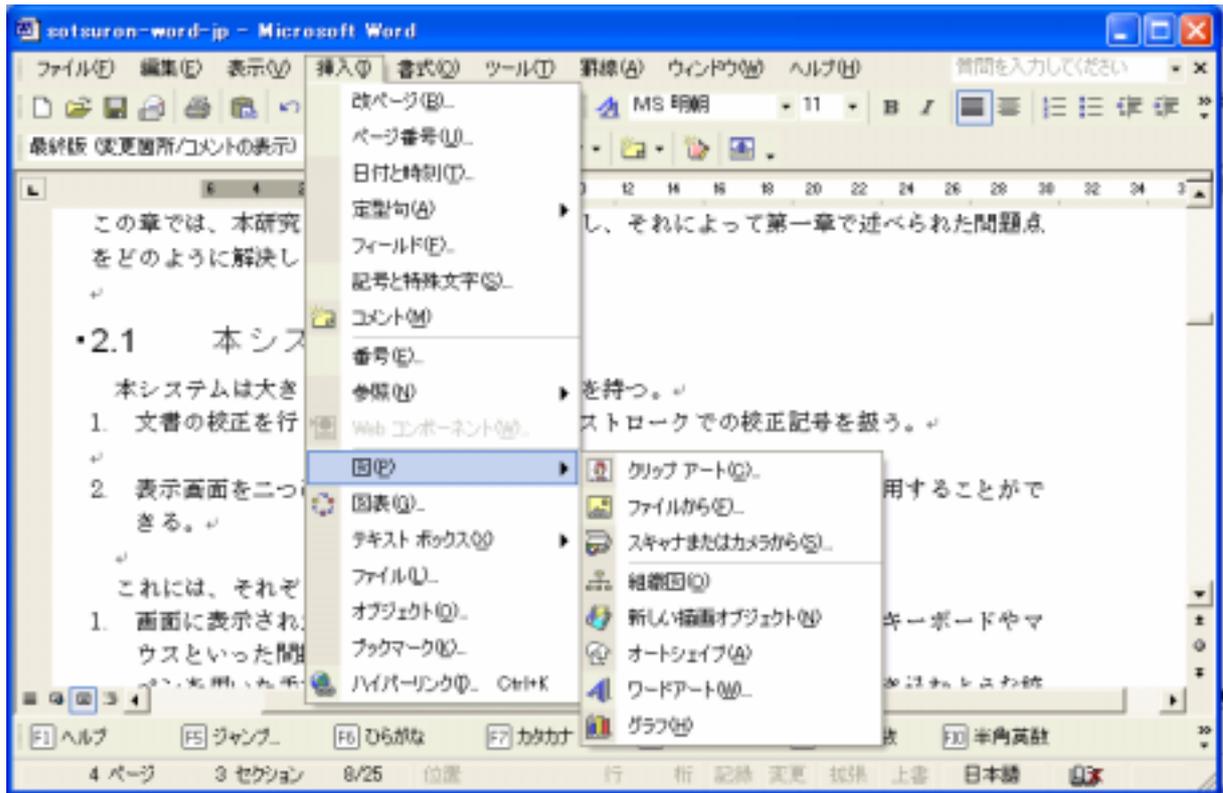


図 1-1 Microsoft Word での編集例

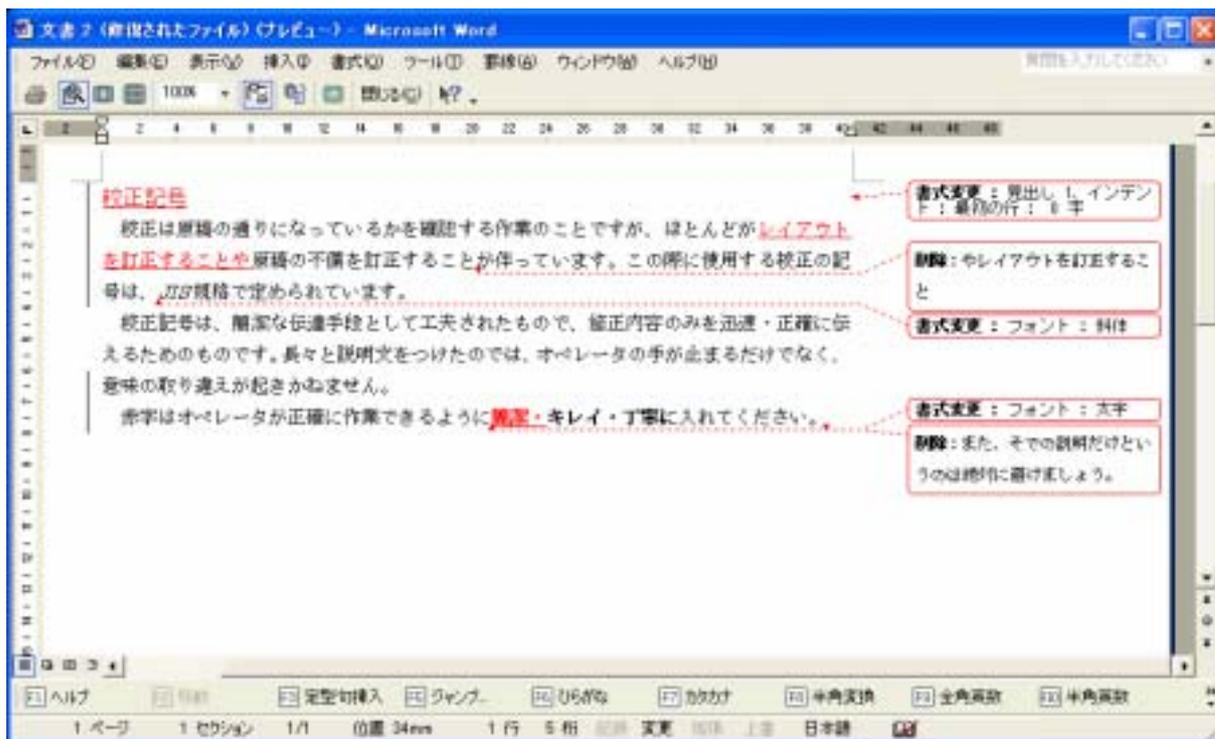


図 1-2 Microsoft Word の校正機能使用例

### 1.2.2 関連研究

一般的なワープロソフト以外にも、ペン入力を用いた編集を行う研究は多くされている。

Moran らが開発したペン入力を使用した電子ホワイトボードシステム Tivoli[2]は、グループでの使用を目的とし、大画面での資料の整理が行えるシステムである。これは大画面の共有ディスプレイを利用し、手書きストロークによって表示されている資料を囲むことで選択することや、簡単なストロークで資料を区切ることでの資料の整理、また資料のアイコン化などの機能がある。

葛貫らの研究に、手書きの校正記号を用いたオンライン編集方式[3]がある。これは、表示された文書に対して校正記号を書き込むことで文書の校正が行われるものであるが、校正前のレイアウトを保存することができないという問題がある。

また、Hardock らの、手書きの注釈を入れることにより文書の編集を行う MATE [4]というシステムもある。これは本研究と同じく二つの表示ビューを持っているが、編集を進めていくと画面が複雑化し編集が続けにくくなることや、文字入力にキーボードでの入力が必要であるなどの問題がある。

文書の添削に関する研究としては、矢野らの構築した、日本語作文教育のためのネットワーク型添削支援システム CoCoA[5]が挙げられる。これは、留学生などが書いた作文に対して教師が添削を行うことを支援するシステムであり、電子メールを用いて学習者と教師とのやりとりが行われる。電子メールでやりとりされる文書は CCML 文書と

いう形式で書かれており、CoCoA のビューアおよびエディタを利用することで、添削の内容を確認・作成することができる。このシステムはあくまで学習者と教師の間での学習のための利用を目的としたものであるため、個人での利用には対応していない。

### 1.3 研究の目的

本研究では、文書の編集作業の中でも、特に校正作業を支援するためのシステムを提案、設計する。

そのため、扱う文書としてはすでにある程度完成している文書を入力とし、システムを用いて校正を行い、その結果として文書をよりよいものとして完成させることを目的とする。

また入力方法としてはペン入力のみを想定し、キーボードなどの併用をせずに扱えるシステムを目標とする。

### 1.4 論文の構成

本論文の構成は以下のようになっている。第二章では本研究で提案する文書校正システムの提案及びシステムの概要を説明し、第三章でそのシステムの具体的な実装を説明する。第四章では実際にシステムを使用した評価を行い、第五章でまとめる。

## 第2章 手書き校正記号による文書編集システムの提案

この章では、本研究で提案するシステムを説明する。

### 2.1 本システムの設計

本システムは、タブレット PC のようなペン入力インターフェースを用いての使用を想定しており、キーボードやマウス等での入力は極力使用しないものとする。

その特徴としては、文書の校正を行うための入力として、ペン入力を用いた手書きストロークでの校正記号を扱うということがある。また、表示画面を二つにし、校正前の文書と校正後の文書を同時に見ながら使用することができる。

これには、以下のような利点がある。

- ・ 画面に表示された文書の校正したい場所を直接示すことができるため、キーボードやマウスといった間接的なカーソル移動での指示を必要としない。
- ・ ペンを用いた手書き入力で文書の校正を行うことで、実際の紙の上書き込むような校正の感覚を持つことができる。
- ・ 校正記号そのものに編集機能の意味が備わっているため、メニューなどでの明示的な機能の指定が必要ない。そのため、選択ミスによる思考過程の妨げが生じない。
- ・ 校正前の文書を見ながら作業ができるため、文書全体のイメージを確認することができる。

図 2-1 は、システムを使用しているときのイメージである。画面は二画面構成になっており、左側の白地のウインドウには校正記号を書き込み、右側の黄色のウインドウにはその校正結果が表示される。ここでは左側のウインドウに削除の校正記号と改行の校正記号が書き込まれており、それぞれの校正結果が右側のウインドウに反映されている。詳しくは、以降の章で説明する。

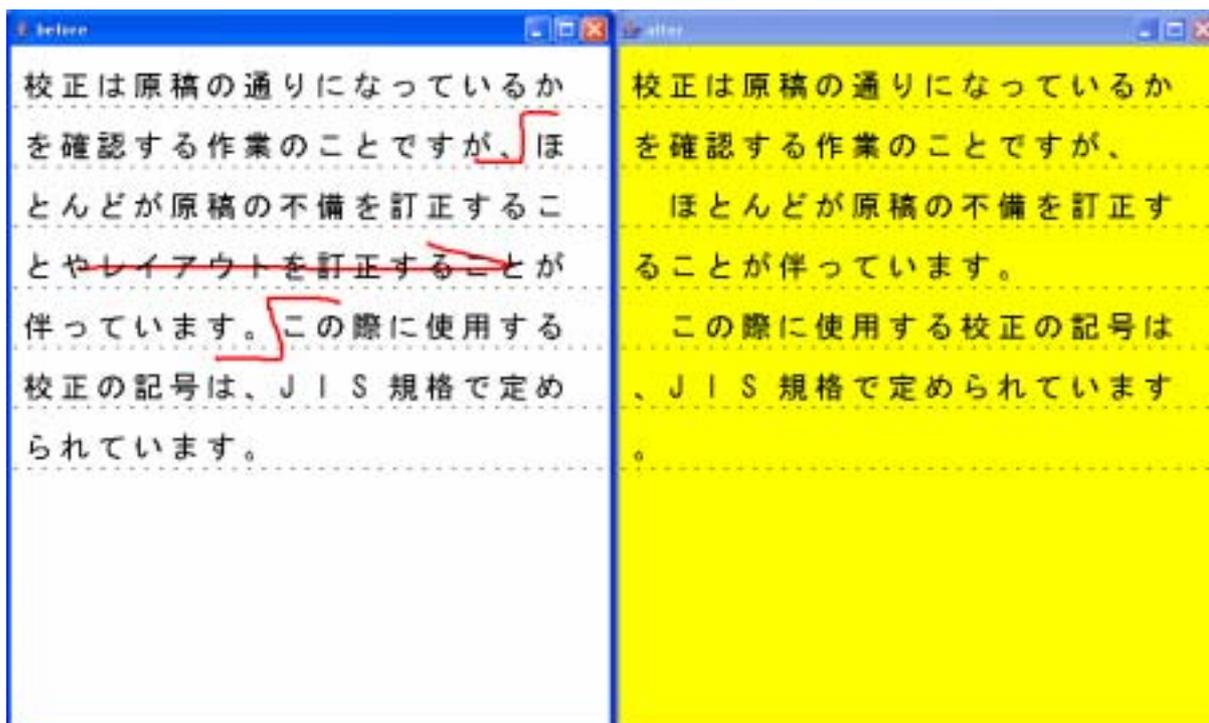


図 2-1 システム使用イメージ

## 2.2 校正記号の例

ここでは、現在一般的に用いられている校正記号の例を紹介する。

以下に示したようなこれらの校正記号は JIS 規格で定められており（JIS Z 8208-1965）一般的に紙の上で書かれた文書に対しての校正の際に用いられる。[6]

校正記号	意味
	文字・記号などをかえ、または取り去る。
	書体または大きさなどをかえる。
	字間に文字・記号などを入れる。
	転倒した文字・記号などを正しくする。

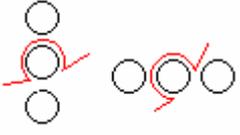
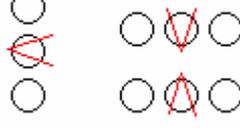
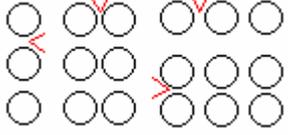
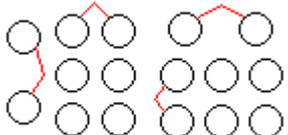
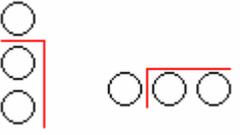
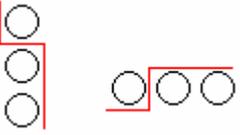
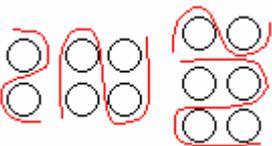
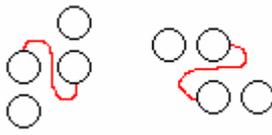
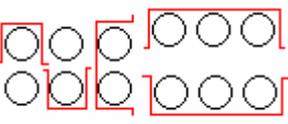
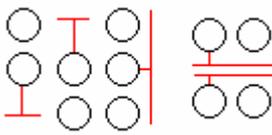
	<p>不良の文字・記号などをかえる。</p>
	<p>右付き・上付きまたは下付きにする。</p>
	<p>字間・行間などをあける。</p>
	<p>字間・行間などをつめる。</p>
	<p>次の行へ移す。</p>
	<p>前の行へ移す。</p>
	<p>行を新しく起こす。</p>
	<p>文字・行などを入れかえる。</p>
	<p>行を続ける。</p>
	<p>指定の位置まで文字・行などを移す。</p>
	<p>指定の位置まで文字・行などを移す。</p>

表 2-2 校正記号の例

## 2.3 使用する校正記号

JIS で定められた校正記号はあくまでも紙の上での校正という目的に対してのものであり、コンピュータ上で使用することに対して必ずしも適切とはいえない。

本研究内容において使用するべき校正には、改行、削除、入れ替え、挿入、文字の書体・大きさの変更、字間・行間の変更、文字の移動などの様々な種類が要求される。それらの校正記号の形状についても、コンピュータ上で扱う際に適しているのかどうか考慮が必要である。

現時点では、使用する頻度が高いと思われる以下の四種類の校正記号について採用をした。なお、解析のしやすさやストロークの書きやすさなどの理由から、上で述べた代表的な校正記号とは多少違っているものもある。

### 1. 改行

図 2-2 の赤線が示すような記号を書くことで、改行を意味する。記号により切り分けられた左側にある部分まではそのまま、右側に分けられた部分から新しい列に移される。これは、表 2-2 における「行を新しく起こす」に相当する記号である。

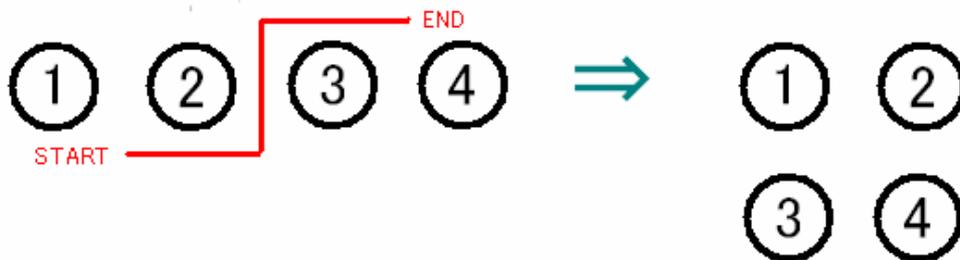


図 2-3 校正記号：改行

図中の START、END は線の始点、終点を表している。

### 2. 削除

図 2-3 の赤線が示すような記号を書くことで、削除を意味する。線が重なった部分の文字列は不要なものとし、消去される。これは、表 2-2 における「文字・記号などをかえ、または取り去る」に相当する記号である。ただし、このシステム上では文字を取り去るだけで、文字・記号などをかえる機能はついていない。しかし、文字の変更の機能については、次に説明する挿入の機能を併用することで実現可能である。



図 2-4 校正記号：削除

### 3. 挿入

図 2-4 の赤線が示すような記号を書くことで、挿入を意味する。縦線によって区切られた位置に新たな要素を挿入する。円で囲まれた部分には、挿入したい要素を書き込む（図中では ☆ で表されている）。これは、表 2-2 における「字間に文字・記号などを入れる」に相当する記号である。



図 2-5 校正記号：挿入

### 4. 入れ替え

図 2-5 の赤線が示すような記号を書くことで、入れ替えを意味する。線によって左側に囲まれた文字列と右側に囲まれた文字列がそれぞれ入れ替えられる。これは、表 2-2 における「文字・行などを入れかえる」に相当する記号である。

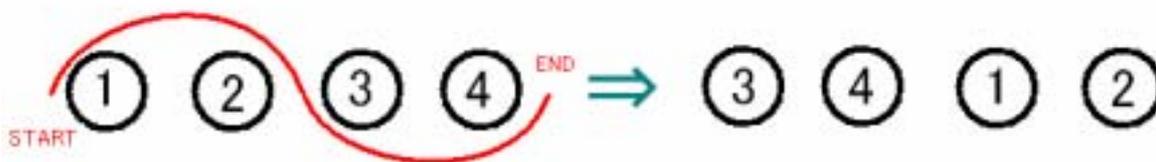


図 2-6 校正記号：入れ替え

## 2.4 表示画面の構成

ここでは、表示されるウインドウについて説明する。

本システムには、表示画面が二つのウインドウで構成されている。（図 2-7）

左側に表示されている白地のウインドウは校正前の文書であり、ここに校正記号を書き込むことで文書の校正を行う。書き込まれた校正記号及び校正前の文書は常に保存され、文書の構造が変わることはない。

右側に表示されている黄色のウインドウは校正後の文書であり、左側のウインドウに書き込まれた校正記号による校正結果を適宜反映し、文書構造が変化する。また、こちらのウインドウは校正結果の表示のみを行うため、ストロークの書き込みをすることはできない。

現在の画面構成は左側に校正記号を書き込み、右側に結果を表示しているが、ペンを使用して校正記号を書き込む際には右利きの場合と左利きの場合で書きやすさが異なってしまうということが考えられる。現在の実装ではそれに対応して適切なウインドウ位置に変更する機能はついていないが、将来的には自由にウインドウ位置の設定ができるような実装を考えている。

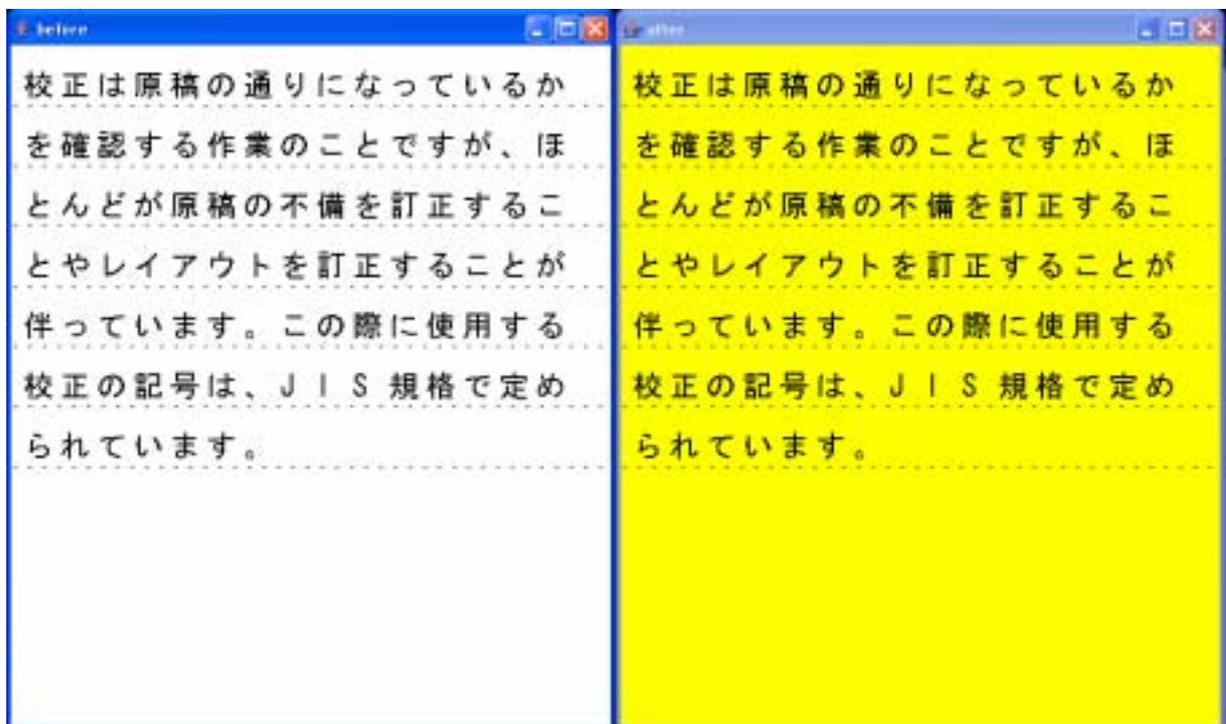


図 2-7 表示画面の構成

## 第3章 試作システムの実装

### 3.1 実装環境

システムの実装には Java TM 2 SDK, Standard Edition Version 1.4.2 を用いた。また、手書き入力インタフェースを実装するためのライブラリとして SATIN[7]を用いた。

### 3.2 システムの設計

#### 3.2.1 画面の表示

表示されている2つのウィンドウは、SATIN の Sheet クラスを継承し、様々な機能を追加した ScribblerSheet クラスを用いている。

Sheet クラスは JPanel クラスを継承し拡張されたクラスであり、ストロークの入力などに関する機能を実装している。

文字の表示に関しては、一つ一つの文字に対して MyObject というクラスを割り当て、一つ一つの文字を別々に管理している。MyObject クラスは JComponent クラスを継承したクラスであり、表示される一つの文字及び表示位置を与え、ScribblerSheet クラスに追加することで対応する文字が表示される。

#### 3.2.2 主要クラスの説明

- ScribblerSheet クラス

Sheet クラスを継承したクラスであり、画面の表示に関するほとんどのデータを持ち、段落を表す Paragraph クラスや行を表す Line クラスなども持つ。

また、解析結果に基づく個別の処理のメソッドもこのクラスに記述されている。

- Mysgi クラス

StandardGestureinterpreter クラスを継承したクラスであり、ストロークの解析を行う。

それぞれの解析結果に応じて、ScribblerSheet クラスのメソッドを呼び出す。

- MyObject クラス

JComponent クラスを継承した一つの文字を表すクラスであり、表示する座標、表示する文字、フォント、文字の大きさなどを保持する。

- MyManager クラス

両方の ScribblerSheet のインスタンス（校正前、校正後のウィンドウ）を保持し、それぞれに対して文字列の表示、再描画などを行う。

### 3.3 手書き校正記号の解析

手書きストロークの解析には、SATIN に付属のツール `quill` を使用している。

`quill` を使用してジェスチャを登録することで、Sheet に登録された `StandardGesture-Interpreter` が Sheet 上で書かれたストロークを自動的に解析し、その結果最も類似度の高いジェスチャ名を返してくれる。

解析手順としては、

1. 手書きストロークにより校正記号を入力する
  2. 入力されたストロークを解析し、類似度の高いジェスチャ名を受け取る
  3. 受け取ったジェスチャ名によって、それぞれの機能を実行する
- という順になっている。

また、解析されなかったストロークについては画面に表示として残らないようにしている。

以下に、それぞれの機能の実行例を示す。

#### 3.3.1 改行

書かれたストロークの中から左右に区切られると思われる適切な X 座標を探し、その座標の後の文字以降を次の行に送る。

実行例を以下の図 3-1、図 3-2 に示す。

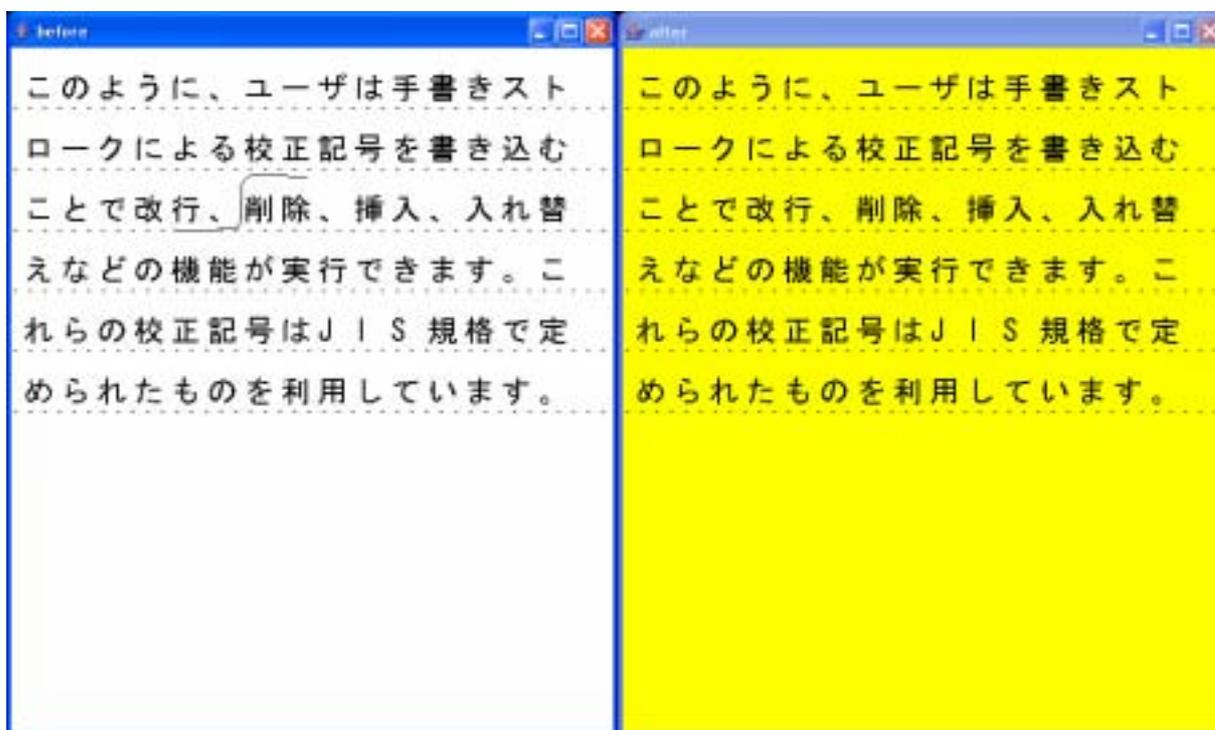


図 3-1 改行：ストロークの入力

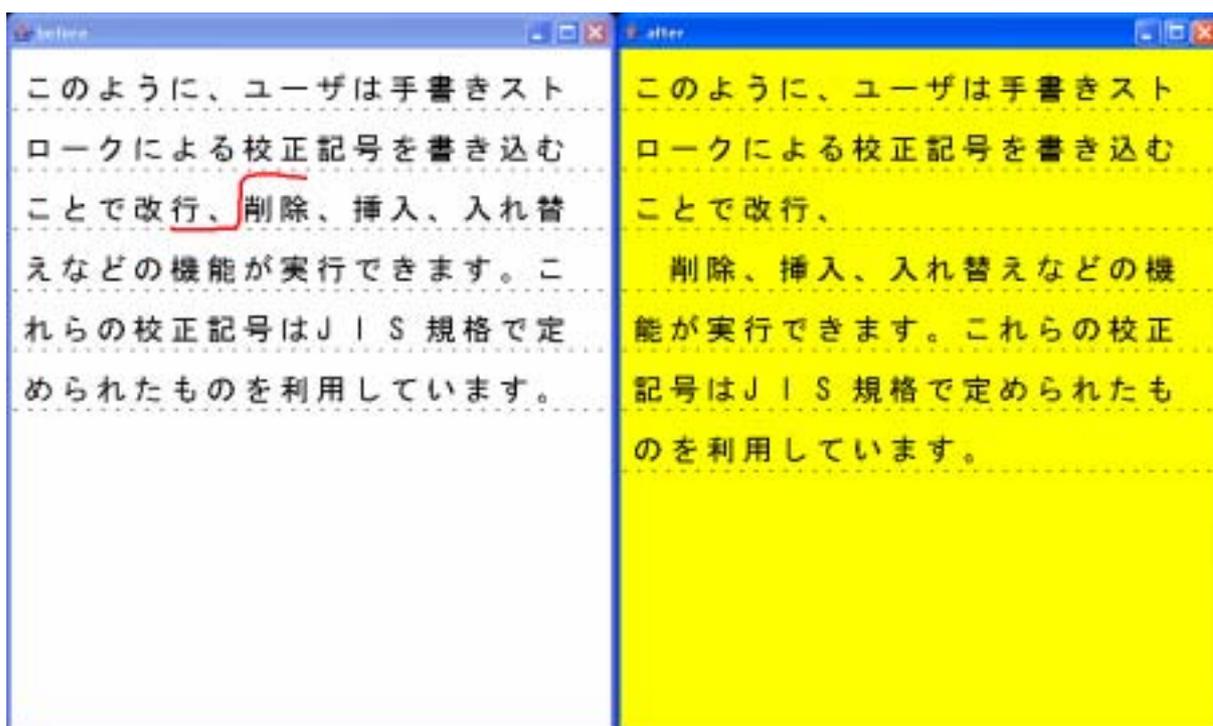


図 3-2 改行：校正結果

図 3-1 はストロークを入力している最中のものであり、ペンをドラッグしている間は、そのストロークが黒い線で表されている。

図 3-2 はペンを放し、ストロークの入力が終わった後のものであり、解析されたストロークは赤く色が変化し、線が太く表示される。またそれと同時に校正後の画面にその解析結果が反映される。これらの動作については、以下のいずれの記号についても同様である。

校正結果である右側のウインドウでは、入力された校正記号のある位置から先の文字列が次の行へ移されていることがわかる。

### 3.3.2 削除

ストロークが重なっている部分の文字列を探し、校正結果にはその文字を表示しないようにする。

実行例を以下の図 3-3、図 3-4 に示す。

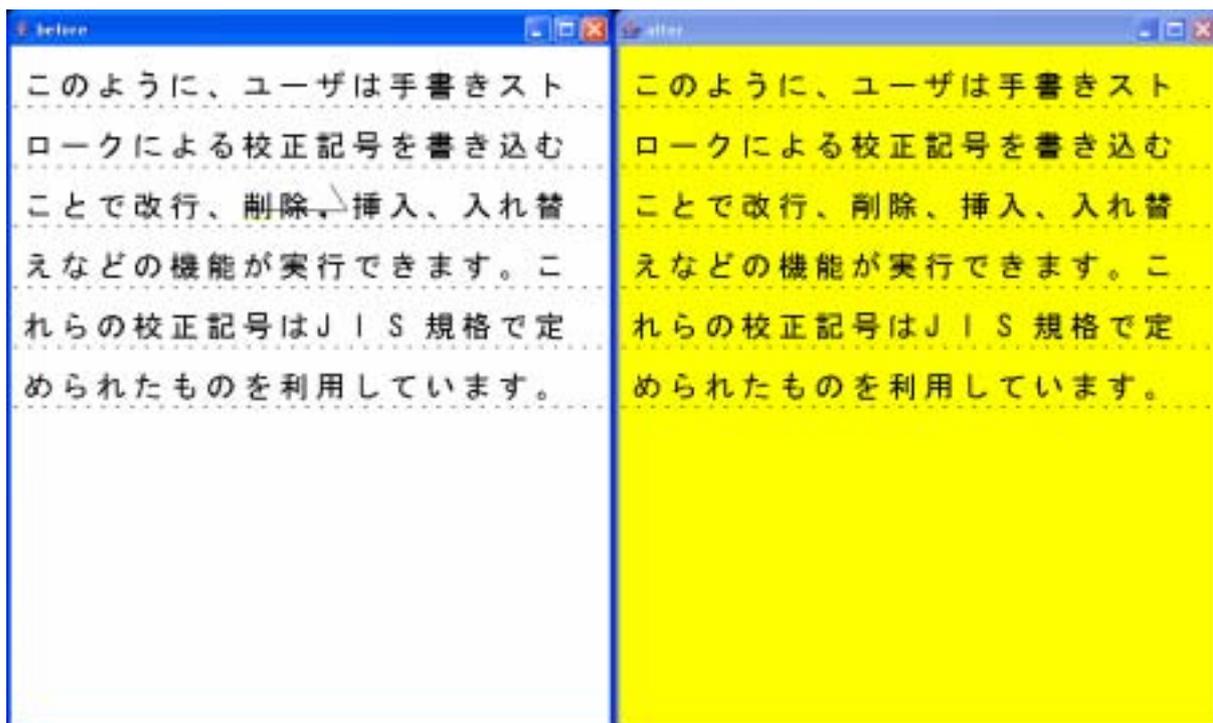


図 3-3 削除：ストロークの入力

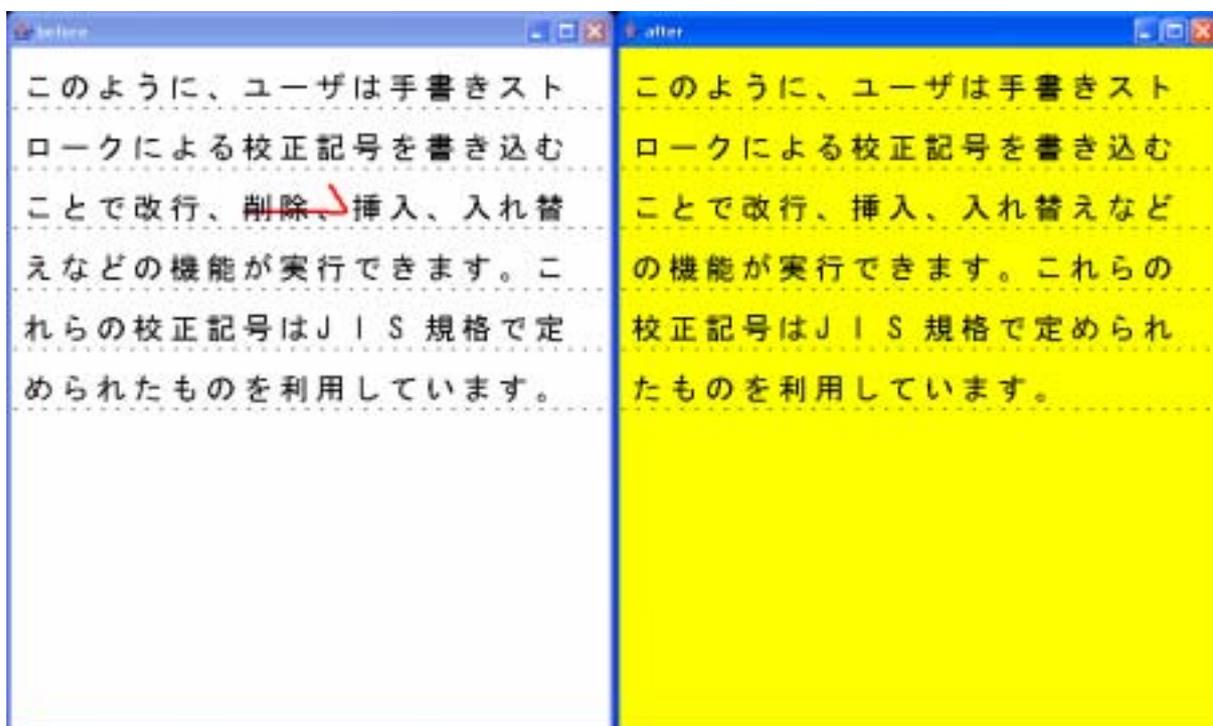


図 3-4 削除：校正結果

図 3-4 は、削除の校正記号によって変更された後の画面の例である。ストロークが重なっている「削除、」の部分が削除され、文字が詰められていることがわかる。

### 3.3.3 挿入

手書き校正記号により示された位置に、自由な手書きストロークを挿入する。  
実行例を以下の図 3-5、図 3-6、図 3-7、図 3-8 に示す。

ストロークが書かれただけでは校正結果に変化は起こらないが、書かれたストロークの下部（挿入要素を書き込む部分）をタップ（マウスクリック）することで新しくウィンドウが開く。また、タップする部分にカーソルが入ったときにストロークの色が赤から青に変化し、ユーザにタップするべき場所がわかりやすいようにした。

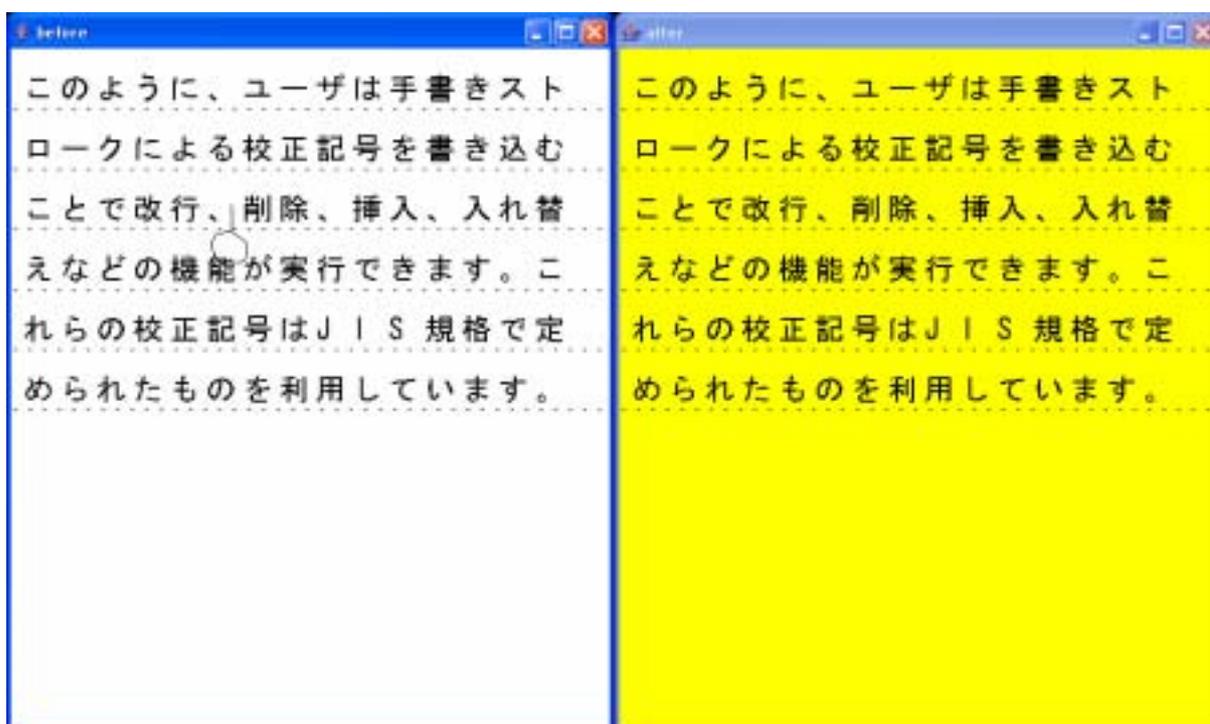


図 3-5 挿入：ストロークの入力

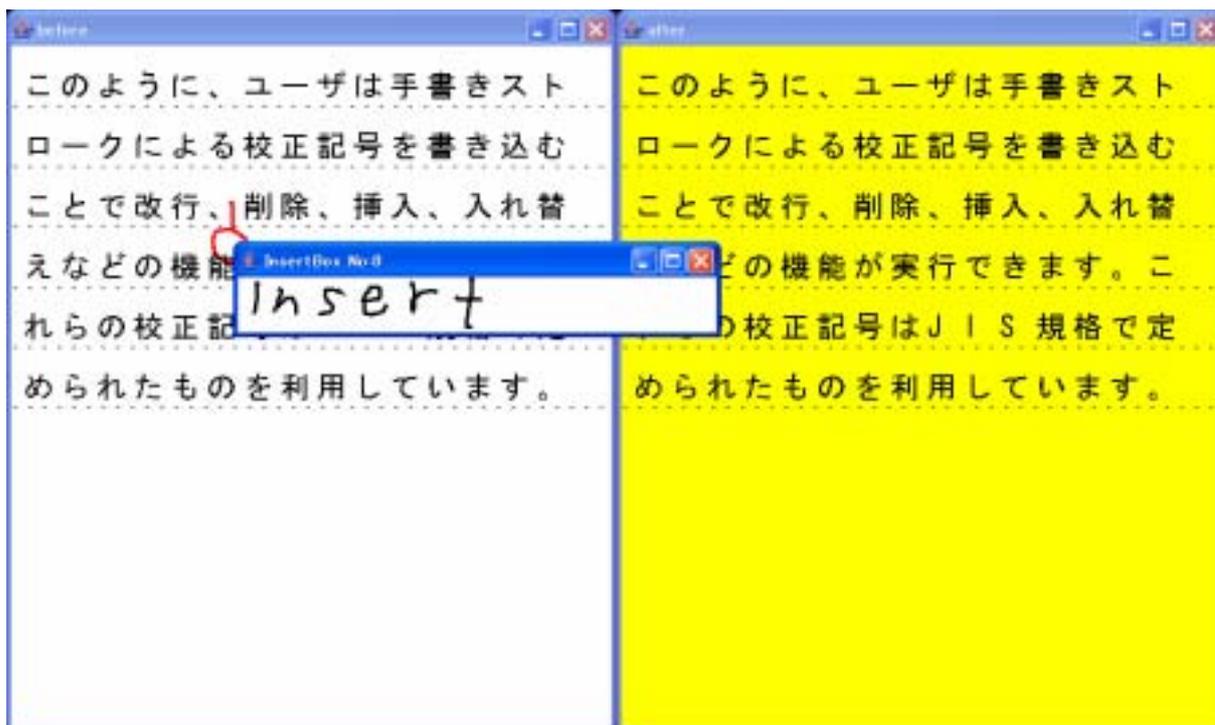


図 3-6 挿入：挿入要素の入力

図 3-6 は、新しく開いたウインドウに手書きストロークを書き込んだ例である。このように、挿入したい要素を自由なストロークで書き込むことができる。書き終わったら、ウインドウ内でタップすることで、ウインドウ内に書かれたストロークが校正結果に挿入して表示される。挿入する位置はストロークの開始位置である点を基準に判定している。

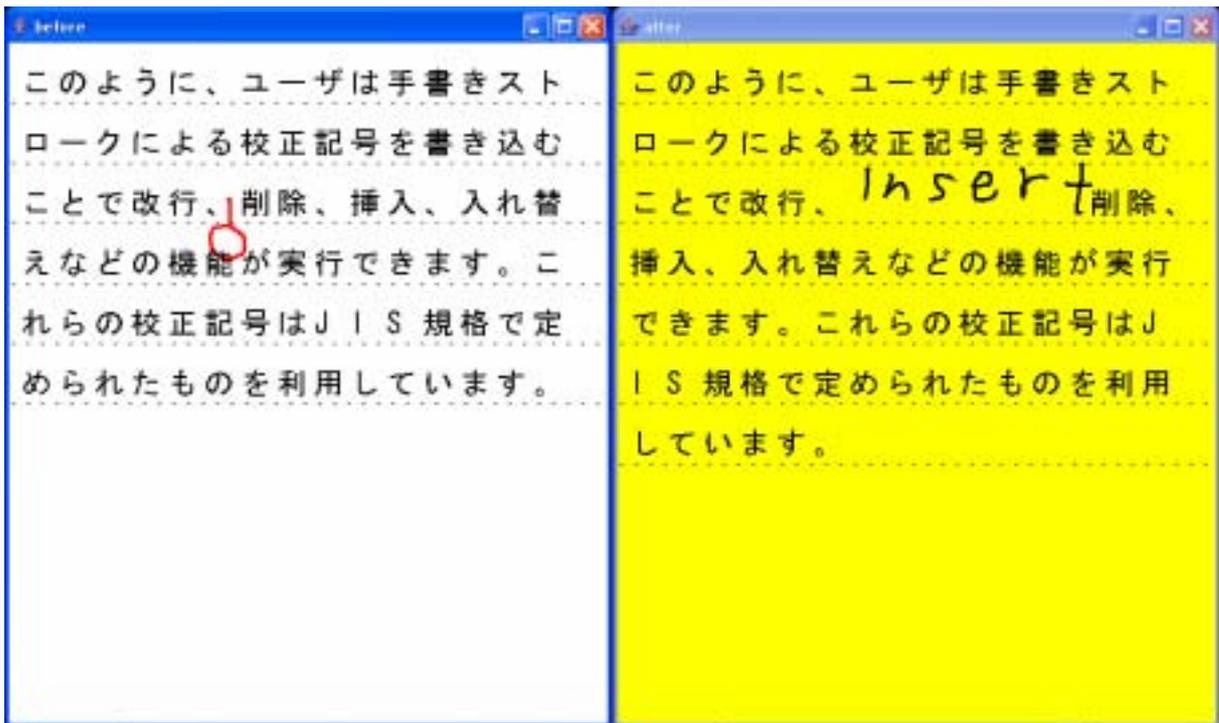


図 3-7 挿入：校正結果

図 3-7 は、手書きストロークが挿入された結果である。このように、指定した位置から要素が挿入され、その要素の大きさに応じて以降の文字が寄せられる。もし、挿入する要素の横幅が大きすぎて画面の右側にはみ出してしまう場合には、自動的に次の行へと行替えが行われる。

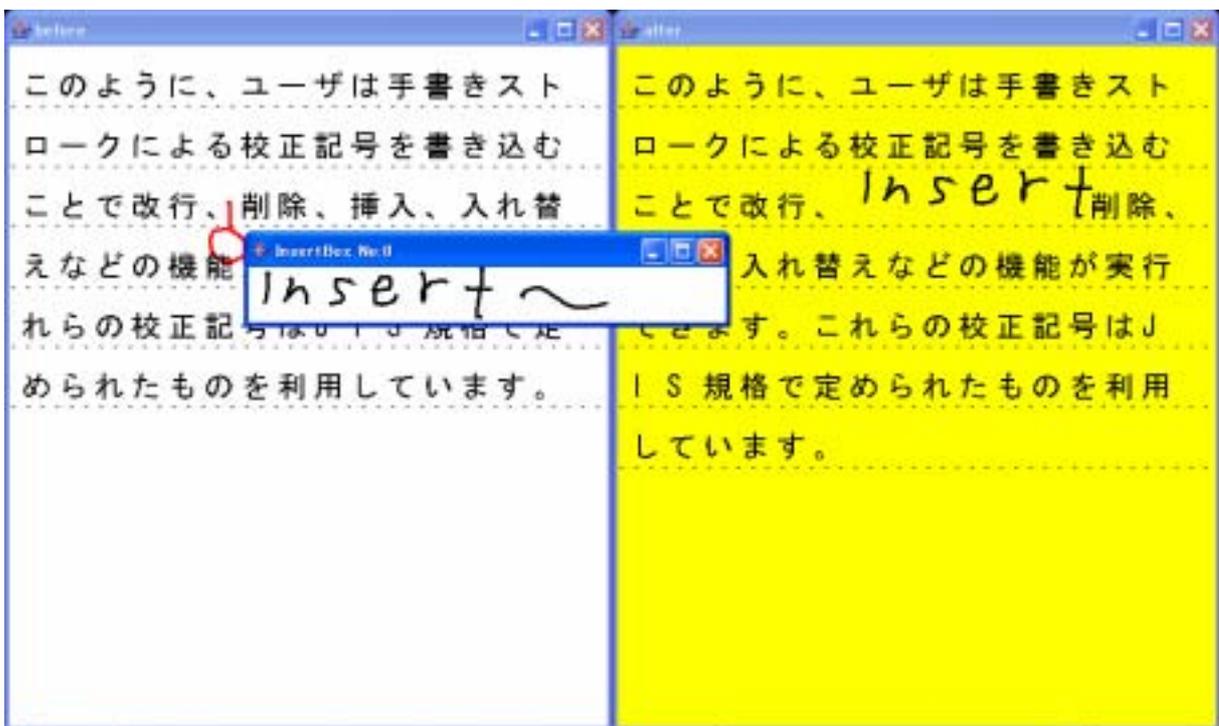


図 3-8 挿入：挿入要素の再編集

また、図 3-8 で示すように改めて校正記号をタップすることで再び挿入要素入力用のウインドウが開き、挿入要素を追加することも可能である。挿入要素を追加した後に再びウインドウ内をタップすると、挿入要素が追加された後のものに変更される。

現時点での実装では手書きストロークがそのまま表示されるようになっているが、将来的には手書き文字を解析し、整形された文字を挿入するようになることを想定している。

### 3.3.4 入れ替え

書かれたストロークの中から左右に区切られると思われる適切な X 座標、およびストロークの最左の X 座標、最右の X 座標を取得し、区切られた部分から左側に囲まれた文字列と右側に囲まれた文字列を入れ替える。

実行例を以下の図 3-9、図 3-10 に示す。

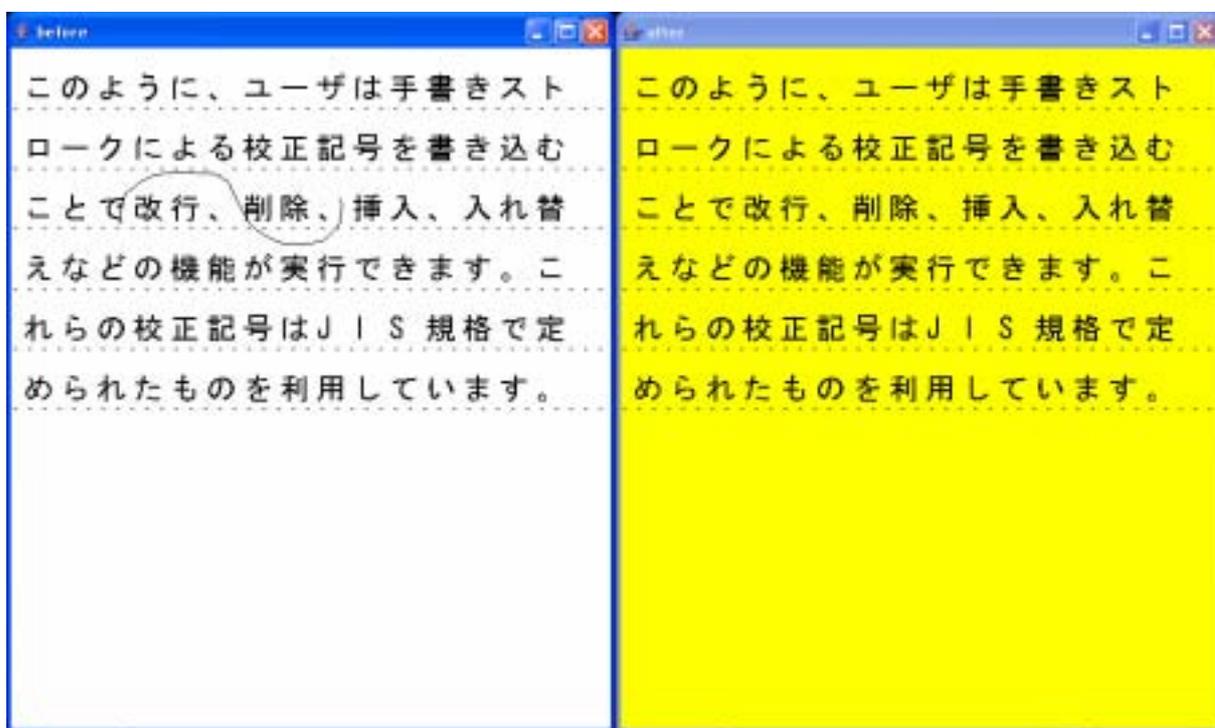


図 3-9 入れ替え：ストロークの入力

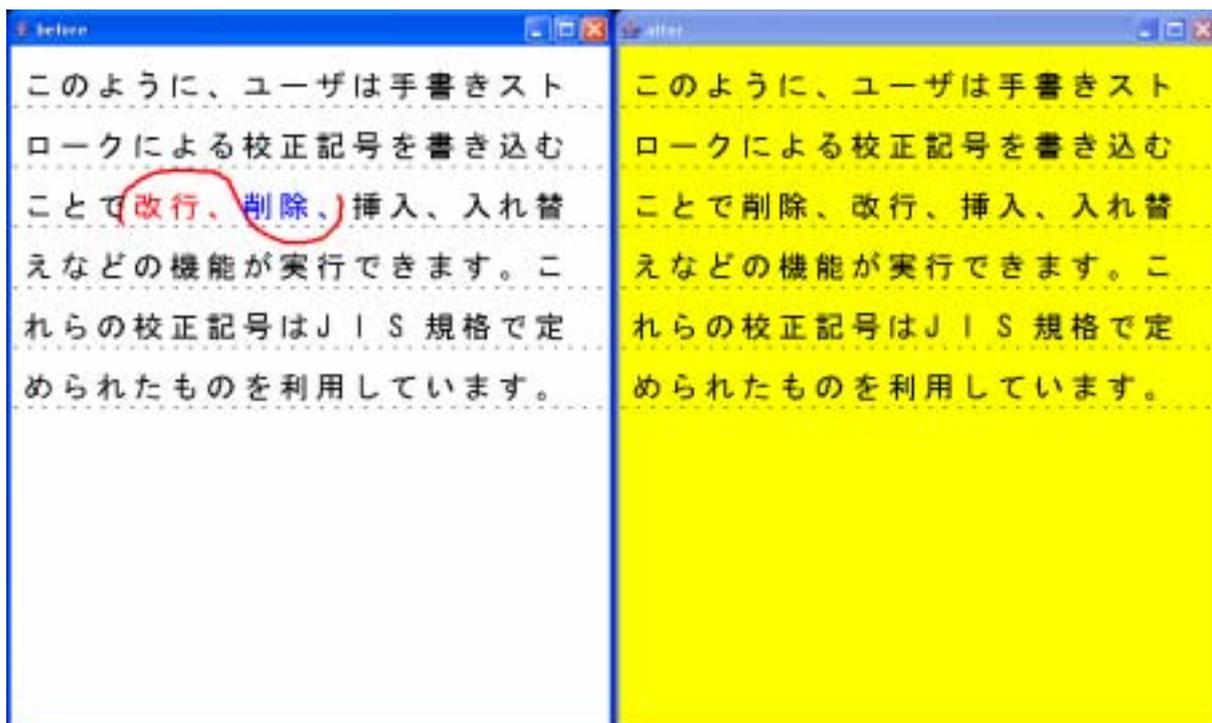


図 3-10 入れ替え：校正結果

図 3-10 は入れ替えた後の結果を示している。「改行、」の部分と「削除、」の部分が入れ替わっていることがわかる。

左側の画面上では入れ替えられる文字がわかりやすいよう、それぞれ赤と青に文字色を変更している。

### 3.4 校正結果の表示

校正結果の表示方法としては、書き込まれた校正記号の種類を解析した後、その種類によって校正後の画面にその校正結果を反映させるという順番で行われている。ウインドウに文字列を配置するための入力としては String クラスの文字列を用いているため、文字列にそれぞれの校正機能を表す記号を挿入し、その記号によってそれぞれの校正結果を表示させるようにしている。

しかし、現在の方法では校正記号が複数個重なってしまったときの処理がうまくいかないため、実装の改善が必要である。

## 第4章 実験

### 4.1 システムの使用例

本システムを実際使用し、文書の校正を行った結果を図 4-1 に示す。

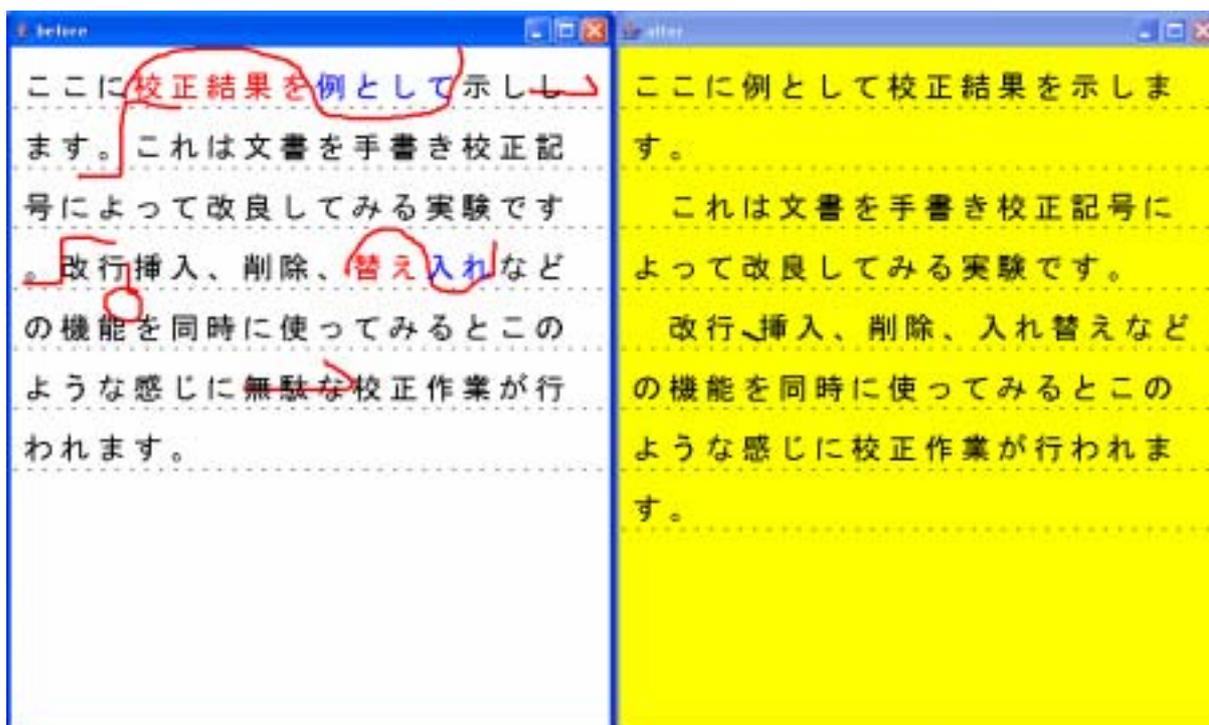


図 4-1 システムの動作実験

左側の画面に書かれたそれぞれの校正記号による校正結果が、右側の画面に反映されている。

### 4.2 評価

実際にシステムを使用してみた結果、手書き校正記号のストロークの解析は比較的うまくいった。

しかし、現時点での実装では、たくさんの校正を行っていく場合に、途中で校正作業のやり直しができないために実用的なものとはいえない。

また、ちょうど行が切り替わる場所にある文字列が分割されてしまう（たとえば、図 4-1 中における“校正記号”という文字列が、行替えのために“校正記”と“号”に分けられている）ため、入れ替えなどの校正記号が書き込めない場合があるという問題がある。これは、一行の横幅が一定であるために起こる問題で、その幅を変更することができるようにするか、あるいはその他の対処が必要である。

### 4.3 今後の課題

上で述べたように、快適な校正を行うためには、一度行った校正結果の破棄やアンドゥ・リドゥの機能の追加などが必要であると思われる。行の切り替わりに関する問題に対しては、ウインドウサイズの変更に対応できるようにすることでの解決を現在考えている。

また、それ以外の機能として、ファイルでの入出力、文字の大きさの変更、文字の移動、整形された文字の挿入などの様々な機能が不足しているため、今後の実装ではそれらの機能の追加もしていかなければならない。

また、実装する機能が増えてくるにつれ、それに対応するストロークの数が増えてくる。すると、それぞれの機能に対して適切なストロークを割り当てる必要があるため、どのようなストロークが適切であるかを考えなくてはならない。それに伴い、ユーザには多くのストロークを覚えなければならないという負担も生じてくるため、これについても考慮が必要である。

ストロークの認識についても、まだ認識率が高くなく誤りや認識できないといったことが多いため、認識率の向上も課題の一つである。

また、現時点の実装では、書き込まれる校正記号の量が増えるにつれ画面が複雑になり見づらくなるという問題を解決できていないため、それに対する解決策も考える必要がある。

## 第5章 まとめ

本論文では、ペン入力インタフェースを扱った文書校正システムとして、手書き校正記号を書き込むことによる校正が可能となるような文書校正手法について提案し、その試作システムの実装について述べた。現段階の実装では実際に使用するシステムとして不十分であるが、様々な機能や適切なストロークを追加していくことで、より手書きに近い感覚での校正が行えるようになると考えられる。

## 謝辞

本論文の執筆にあたり、指導教員である田中二郎教授から様々な助言をいただきました。ここに深く感謝を申し上げます。また、研究全般にわたり助言を下された志築文太郎先生、三末和男先生ならびに高橋伸先生に心から感謝いたします。田中研究室の皆さんにもお世話になりました。とくに同じ VS グループのメンバーの皆さんには、幾度となく助言をいただきました。皆様、本当にありがとうございました。

## 参考文献

[1] Microsoft Word

<http://www.microsoft.com/japan/office/default.msp>

[2] Moran, T. P., Chiu, P., and van Melle, W. “Pen-based interaction techniques for organizing material on an electronic whiteboard” (UIST '97) page 45 - 54.

[3] 葛貫壮士郎, 横山孝典, 正嶋博, 福永泰 “JIS 校正記号準拠のオンライン手書き編集方式 情報処理学会論文誌” Vol.27 No.10 (1986)

[4] Gary Hardock, Gordon Kurtenbach, and William Buxton. “A Marking Based Interface For Collaborative Writing” (UIST '93)page 259 - 266.

[5] 矢野米雄, 緒方広明, 榊原理恵, 脇田里子 “日本語作文教育のためのネットワーク型添削支援システム CoCoA の構築” 教育システム情報学会 (先端的ネットワーク技術特集号), Vol.14, No.3, page 21-28. (1997)

[6] 校正記号の参考: 校正舎 <http://www.kouseisya.jp/>

[7] Jason I. Hong and James A. Landay. “SATIN: A Toolkit for Informal Ink-based Applications.” (UIST '2000) page 63 - 72.

<http://guir.berkeley.edu/projects/satin/>