平成 26 年度　卒業論文

形態素解析を利用した文章範囲選択手法の改良

平成 27 年 2 月 16 日

10111017

清弘 祥太

指導教員 三浦 元喜 准教授

九州工業大学 工学部 総合システム工学科

概 要

タブレット端末において，ソフトウェアキーボードによる文字入力は，一般的なキーボードによる文字入力に比べて負荷が高い．そのため，画面に表示されている文字列を選択し，コピーペーストして再利用することが頻繁に行われる．しかし従来のタブレット/スマートフォン用 OS における文字選択方式は選択対象文の文法や文脈を考慮して設計されていない．本研究では，先行研究における日本語形態素を考慮した文字選択方式を改良するとともに，従来研究における実験条件を詳細化することによって，タブレットにおける文字列範囲選択手法の知見を明確にすることを目的とする．改良として，ひらがな 1 文字の助詞に相当する単語を直前の単語と連結してグルーピングする手法を提案し，文字単位・単語単位，改良した単語単位の 3 条件での被験者実験を行った．一元配置の分散分析を行った結果，上記 3 手法の作業時間の平均値に有意差が認められた．また多重比較を行った結果，上記 3 手法のそれぞれについて，作業時間の平均値に有意差が認められた．このことから，単語単位および改良単語単位による文字列選択手法の有効性を確認し，今後のタブレット／スマートフォンのインタフェース設計に有益な知見を得た．

目 次

第1章　序論

本論文は主にタブレット端末における文字列範囲選択手法について論ずるものである．第 1 章では，研究の背景と，目的について述べる．

1.1　背景

近年，タブレット端末やスマートフォンが普及している．その理由として，どこでも手軽に持ち運びができ，インターネットに接続できるため，外出中に現地の情報を調べたり，連絡をとったりするのが容易にできることが挙げられる．また，アプリケーションをインストールすることによって，必要な機能を追加したり，カスタマイズすることも簡単に行える．特に近年では，Twitter や Facebook といった，マイクロブログサービスやソーシャルネットワークサービス (SNS) が広く浸透し，ユーザが感じたことや体験したことを気軽に書き込み，共有するといったことが広く行われている．

このことから，タブレット端末やスマートフォンを介した情報の取得や，情報の入力による共有・提供に対する重要性が高まっている．しかし，タブレット端末やスマートフォンにおける入力方法については，以下に述べる問題がある．

タブレット端末やスマートフォンにおいては，できるだけ多くの情報を一度に表示することを重視し，物理キーボードを搭載せず，タッチパネル方式のディスプレイのみが搭載されることが多い．そのため，文字入力を行う際には，画面上にキーボードを表示し，指やペンでその仮想的なキーボードを操作する，いわゆるソフトウェアキーボードを用いることが一般的となっている．

しかし，ソフトウェアキーボードによる文字入力は，一般的なキーボードによる文字入力に比べて負荷が高い．その理由として，キーを押したときの操作に対するフィードバックが無いことや，キーの触感により，どのキーを触っているかの手がかりが得られないことが原因として考えられている．

そのため，ユーザは可能であれば，画面上に表示されている文字列を選択し，それをコピーペーストして再利用しようとするし，実際に頻繁に行われる．しかし従来のタブレット/スマートフォン用 OS における文字選択方式は，選択対象文の文法や文脈を十分に考慮して設計されていない．

例えば iOS では，範囲選択を開始する際のロングタップ操作のときのデフォルト選択では単語単位で行われるが，そのあとのハンドル移動では文字単位となる．AndroidOS では，ロングタップ時のデフォルト選択は同一文字種 (ひらがな，カタカナ，漢字等) が連続した範囲となり，その後のハンドル移動はやはり文字単位となる．

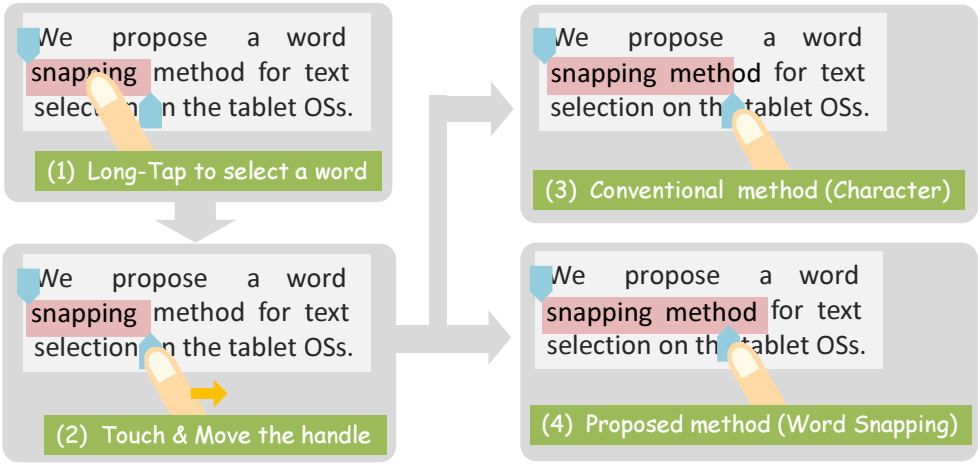

図 1.1: タブレット端末における文字選択方式の概要：文字単位 (Character) と単語単位 (Word Snapping)

最所らは，ロングタップ後のハンドル移動について，文字単位ではなく単語単位のみで動かすようにすること（図 1.1）を提案し，タブレット上での実験を行っている [1, 2]. その結果，一文字毎の文字選択手法に比べて，単語単位での文字選択手法のほうが，タスク完了時間の平均値が有意に減少したと述べている．また使いやすさについて尋ねたアンケート結果でも，単語単位のほうが使いやすいという結果が得られている．しかし，最所らの実験では，行跨ぎタスクの場合は，単語単位の選択は適しておらず，また詳細な検討を行った結果，単語数が少ない場合（1 単語や 2 単語）のときのみ有意差があるが，そうでない場合には有為差がでないという結果となっている．

1.2 研究の目的

そこで我々は，最所らの実験について，より詳細な検討を行うための追実験を行い，単語単位の選択は，本当に単語数が少ない場合のみ有効なのか，また，行跨ぎタスクについて，本当に単語単位の選択は適していないのかといった点について，確認を行うことを本研究の目的として設定した．

これらの疑問点が解決されることによって，タブレット端末やスマートフォンへの単語単位の文字列選択手法の有効性が確認され，より良いインタフェースの開発につながることが期待できる．

第2章　関連研究

本章では，本論文で述べるタブレット端末における文字列範囲選択手法に関連する研究について概説する．

2.1　指による操作の困難さ

タブレットやスマートフォンの操作は，主に指で操作することが多い．画面のスクロールや拡大縮小といった操作は，とくにターゲットを意識しなくても操作できるが，アイコンボタンの選択やハンドルの操作といった，細かい画面上のオブジェクトを操作する場合に発生する問題は，一般に Fat Finger Problem [3] として認知されている．Fat Finger Problem には，主として２つの要因がある．ひとつは，指によって画面上の細かなオブジェクトが隠れてしまい，操作が難しくなるという問題で，もうひとつはタッチパネルに対する指の操作は，スタイラスペンによる操作と比べて，広い領域で画面をタッチすることになるため，繊細な調整に向いていないという点である．iOS の文字選択において，ハンドルやカーソルを詳細に調整する際に表示される拡大レンズは，一般にはエリアカーソル [4, 5] と呼ばれる，Fat Finger Problem や微細なターゲットに対する操作を円滑に行うための技術である．また関連して，ターゲット選択の正確性を高めるための手法も提案されてきた [6, 7] が，タブレット OS における文字選択タスクについての研究は，いわゆるスマートフォンの普及時期以降のものが多い．

Cockburn ら [8] は，指タッチパネルとスタイラス，マウス操作の 3 者についてタップ，ドラッグ，回転ドラッグ（アナログ時計の針を回すような操作）を比較した一般的な調査結果を示している．彼らは，指タップ操作は他の操作に比べて早いが，誤差エラーも多いと結論づけている．タブレットやスマートフォンの画面に表示される文字は一般に小さいため，文字単位の選択は画面を拡大しない限り，細かな操作が要求される．

2.2 文字列選択タスク

Fuccella らは，ソフトキーボード領域におけるマルチタッチジェスチャを用いて，カーソル移動やコピー・ペースト等のショートカット操作を提供する方式を提案し，評価実験を行っている [9]．1 本指の横スワイプは文字毎で，2 本指は単語毎の移動といったジェスチャが割り当てられている．これらの操作は単語毎に行う必要があるため，複数の単語を一気に選択範囲に含めるといった操作は (行単位の選択を除くと) 考慮されていない．Scheibel らは，ジョイスティック風の操作ウィジェット (Virtual Stick) によってカーソル移動タスクの正確性を高める研究を行っている [10]．Virtual Stick と指操作との比較実験を行い，ハンドルの移動距離が短い場合で，文字サイズが小さい場合に，指による直接指定よりも有効であったと報告している．

島らは従来のロングタップによる範囲選択開始ではなく，3 点タッチを起動ジェスチャとする範囲選択手法を提案している [11]．ダブルタップやロングタップとの比較実験を行い，被験者のコメントを収集している．

2.3 文脈（コンテキスト）を利用した文字列選択

プログラミングを支援するための統合開発環境 (IDE) においては，テキストの文脈（コンテキスト）を利用した文字列選択の概念が提案されている．Wallace ら [12] は，ソースコードの文脈を理解し，プログラマがコードブロック内のある箇所をダブルクリックしたり，トリプルクリックしたりすると，プログラムの意味内容に応じて 1 行の命令全体や，カーリーブラケットで囲まれたブロック全体の文字列を自動的に選択してくれるエディタを開発している．こうした研究はソースコードのコピー＆ペーストによる再利用性を高める働きをしているといえる．Kerr と Stuerzlinger [13] は，この考えを拡張し，元のコピー文字列をペーストした際に，ペースト先の文脈を考慮して自動的にエラーを回避する手法を提案している．

こうした文脈を利用した文字列選択は，プログラミング言語のように明確なルールが定まっている領域では有効に作用することが確認されている．しかし，自然言語については，まだその有効性は確認されていない．また，タブレット端末における操作についても，確認されていなかった．

最所は，これらの文脈を利用した文字列選択の手法を，自然言語の 1 つである日本語文章に拡張して，タブレット上での実験を行った [1]．その結果，一文字毎の文字選択手法に比べて，単語単位での文字選択手法のほうが，タスク完了時間の平均値が有意に減少したと述べている．また使いやすさについて尋ねたアンケート結果でも，単語単位のほうが使いやすいという結果が得られている．しかし，最所の実験では，行跨ぎタスクの場合は，単語単位の選択は適しておらず，また詳細な検討を行った結果，単語数が少ない場合（1 単語や 2 単語）のときのみ有意差があるが，そうでない場合には有為差がでないという結果となっている．

第3章　提案手法

本研究の目的は，最所らの実験について，より詳細な検討を行うための追実験を行うことにある．本章では，その詳細な検討を行ううえで，本研究で独自に追加した文字列選択手法について述べる．

3.1　本研究で独自に追加した文字列選択手法

最所らの研究では，単語単位 (Word) による方式を新たに提案し，文字単位 (Char) による方式と比較することが主目的であった．今回の研究では，単語単位のグルーピング効果について，より詳細な検討を行うことを目的として，単語単位で，ひらがなは前の単語と連結する手法を追加して，実験を行うこととした．

図 3.1 に，最所らが提案した，単語単位による方式で分割した文章の例を示す．図 3.2 に，我々が導入する，ひらがな 1 文字の単語は前の単語と連結する方式で分割した文章の例を示す．それぞれ，縦棒が各単語（グルーピング）の切れ目を示している．

ひらがな 1 文字の単語は前の単語と連結する方式の利点について述べる．日本語の文章を単語に分割する際には，通常，形態素解析器を利用する．形態素解析器は，日本語文章を形態素に分解し，それぞれの形態素の種類を付与することができる．しかし，形態素の分解は，一般的な日本語文字列を選択・コピーする場合の単位としては細かすぎる場合がある．

そこで，日本語文章に含まれる「助詞」を，その前の単語に連結することによって，読点で区切られる程度の意味のかたまり（グルーピング）を作成することが有効ではないかと考えた．ここで，グルーピングが大きすぎると，ユーザが選択したい文字列の粒度にそぐわない場合がある．そのため，グルーピングは小さくても大きくても，ユーザの文字列範囲選択に対する要求を満たすことはできないと考えられる．

本来は，形態素解析器の結果を利用して，助詞のみを連結するといった方法が適切で

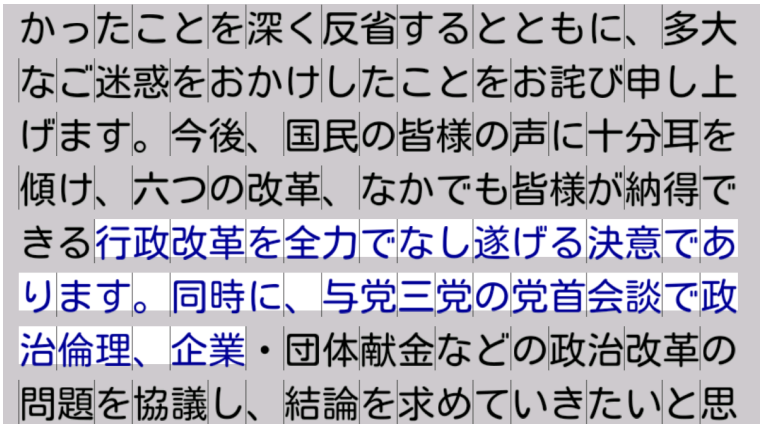

図 3.1: 単語単位 (Word)

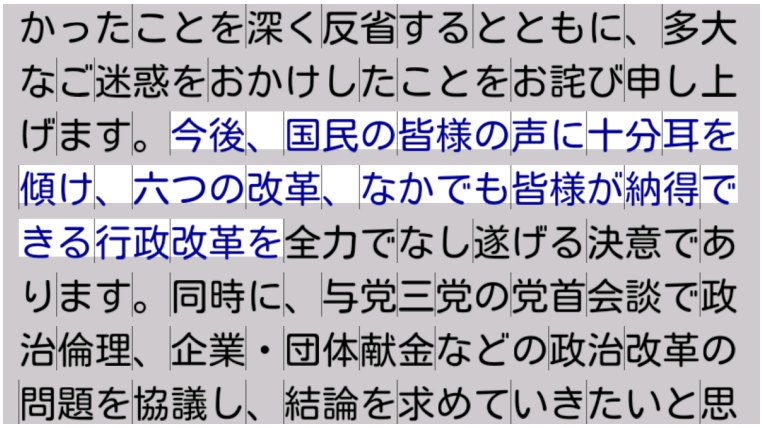

図 3.2: 単語単位で，ひらがな 1 文字の単語は前の単語と連結 (Word2)

あるが，今回は，ひらがな 1 文字の形態素は，助詞として扱うという処理で実験を行った．なぜそのようにしたかというと，形態素解析器を用いて分かち書きを行う（その後，ひらがな 1 文字のみを抽出する）処理のほうが，分かち書きに加えて，個々の形態素の種類を出力する処理に比べて，計算量が少なくて済むからである．近年のスマートフォンやタブレットの性能は向上しているとはいえ，対象となる文章すべてにおいて，形態素解析を行うことはバッテリーの消費やプロセッサの処理負担という観点からも，できるだけ軽量なアルゴリズムを選択すべきであると考える．

実際に，図 3.1 と図 3.2 を比べてみると，ほとんどの場合において，ひらがな 1 文字の

形態素は助詞に含まれることがわかる．また，グルーピングの大きさという観点において，ひらがな２文字以上の助詞は別の単語として個別に選択できるほうが利便性が高いということもある．

第4章　実験システムの改良

本章では，正確な実験を行うための，実験システムの改良について述べる．

4.1　実験タスクの設定

従来の実験において，実験タスクを構成する文章と，選択範囲は試験実施者がそれぞれ設定していた．そのため，タスク数を増やしにくいことや，実験条件によっては，特定の条件にたいして有利な，恣意的なタスクが含まれてしまう危険性があった．

今回の研究では，実験タスクの設定を，ソフトウェアが自動的に構成するようにした．具体的には，実験タスクを構成する文章は，あらかじめ用意しておいた文書群から長さを自動的に調整して作成した．また，単語の区切りについては，Mecab の出力をそのまま利用した．

さらに，実験タスクの妥当性を高めるため，選択範囲の条件についても統制した．具体的には，選択範囲の開始位置と終了位置について，画面の左側にある場合と右側にある場合とで，タスク遂行にかかる時間が変わってくることが考えられる．そのため，選択範囲の開始位置と終了位置について，各条件で平等になるように，実験タスクを作成するようにした．

よって，タスク生成の際のパラメータは，(1) 実験タスクを構成する文章の文字数，(2) 選択範囲の開始位置 (左 or 右)，(3) 選択範囲の終了位置 (左 or 右)，(4) 選択範囲に含まれる改行の数　の 4 点のみとなる．

4.2　実験条件の追加

従来の実験においては，操作手法として (1) 文字単位 (char) (2) 単語単位 (word) の 2 つのみを比較していた．しかし，単語単位にまとめることによる効果を明確に確認するに

は，１単語の文字数はある程度長いほうが望ましい．そのため，本研究では，上記の２手法に加えて，前章で述べた，(3) 単語単位だが，単語がひらがな１文字だった場合は直線の単語と連結する (word2)，という手法を追加することにした．

4.3　単語区切り位置の明示

またプロトタイプ実験の結果，提案手法の設計および実験で用いるタスクについて，改善すべき点が明らかとなった．具体的には，長い単語が連続する場合で，複数の行にまたがる選択指示範囲のとき，上下への意図しないハンドルのジャンプが頻繁に発生した．またこれに関連して，終端ハンドルを画面右側から次の行の左側にドラッグで移動する操作の負荷が高いことがわかった．これを改善するため，図 4.1 に示すように単語区切り位置を明示し，ユーザがどこにハンドルを移動可能かをフィードバックすることにした．

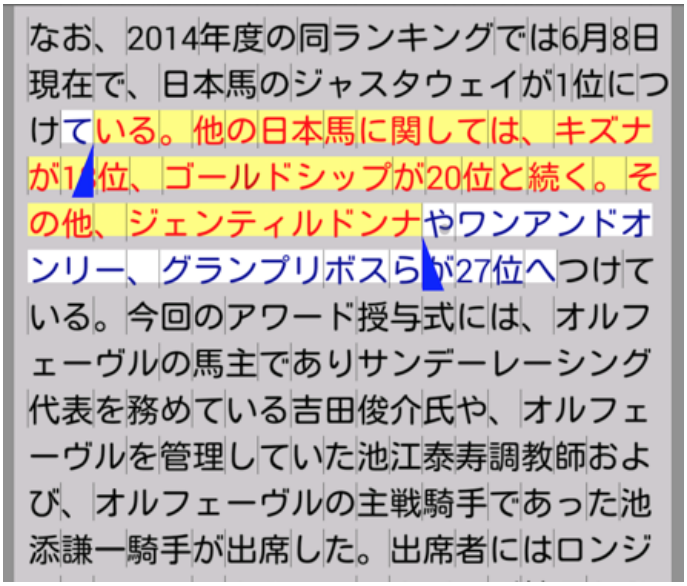

図 4.1: 単語区切り位置の明示

今回の実験システムでは，Mecab を用いて実験タスクを構成するため，長い単語が出力されることはほとんどない．ただし，固有名詞のように長い単語として認識された場合に，上記の問題が発生するおそれがあったため，今回はフィードバック機能を有効にして実験を行うことにした．

なお，単語区切り位置のフィードバックが常に表示されることは，通常の閲覧に影響を及ぼす可能性があるため，今回はロングタップによって文字列選択が開始されているときのみ，表示するように設定した．

4.4 選択範囲上でのスワイプ操作で選択範囲を調整する機能

従来の実験システムにおいて，ロングタップにより初期選択を行った後，終端ハンドルの位置が画面右端になった場合で，タスク遂行のためにその終端ハンドルをすこしだけ右に動かす（すなわち，次行の先頭部分に移動する）必要があったとき，操作が難しいことが確認された．このような場合に，終端ハンドルを直接ドラッグ操作する以外の方法で，終端ハンドルを移動できることが望ましい．デスクトップ環境においては，マウスで文字列選択した後で，SHIFT キーを押しながら，左右のカーソルキーを押すことによって，終端位置を微調整する機能がある．

これに近い操作感をタブレット環境でも実現するため，我々は選択済みの文字列上で，左右スワイプ操作を行うことによって終端ハンドルまたは始点ハンドルを微調整する機能を追加した．具体的には，最初のロングタップによって初期選択を行った際，ロングタップ位置の文字よりも左側の選択済み文字列上で左右スワイプを行ったときは，始点ハンドルがスワイプ操作量（距離）に応じて左右に移動する．逆に，ロングタップ位置の文字よりも右側の選択済み文字列上で左右スワイプを行ったときは，終端ハンドルがスワイプ操作量（距離）に応じて左右に移動する．

図 4.2 は，上記のスワイプ操作を行う前で,「パーセント」の「セ」上でロングタップしたときの画面である．ここで,「セ」の右側の領域で，右スワイプ操作を行うと，図 4.3 に示すように，終端ハンドルが右側に移動する．ここでは,「ト」の上からスワイプを開始した．

4.5 その他

その他，以下に挙げる問題点や懸念点を解消した．

- 初期選択時の誤動作防止：行間をロングタップしたときに，その上の行が必ず選択されるようにした．これまで，行間に対してロングタップの判定が行われない場所

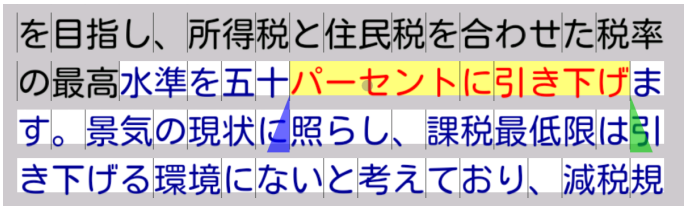

図 4.2: 選択範囲上でのスワイプ操作で選択範囲を調整する機能（スワイプ前）

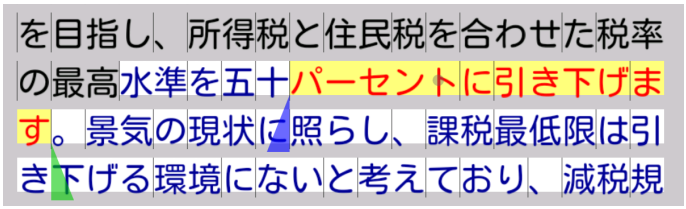

図 4.3: 選択範囲上でのスワイプ操作で選択範囲を調整する機能（スワイプ後）

があり，ユーザの初期選択に影響を及ぼしていた可能性がある．

- 選択範囲調整ハンドルの操作におけるホールドミスの防止：選択範囲調整ハンドルが小さいため，ユーザがハンドルをつかんだと思っていても，うまくつかめていない場合があった．その問題を解決するため，ハンドルをタップ＆ドラッグするときに反応する領域を大きくした．とくに終端ハンドルについては，始点ハンドルよりも頻繁に利用されることが予想されるため，誤動作をさけるために領域を注意深く設定した．なお，終端ハンドルの反応領域と，始点ハンドルの反応領域が万一重なってしまった場合には，終端ハンドルの反応を優先させるようにした．

- 選択範囲調整ハンドルの下にある文字列が読みにくい問題を解消するために，ハンドルを半透明表示するようにした．

- 始点と終端の２つの選択範囲調整ハンドルについて，色を変更し，区別しやすくした．これにより，終端ハンドルを素早く見つけることが可能となった．

- ロングタップによる初期選択の範囲を，従来は単語２つ分であったが，初期選択範囲は大きいほうが操作しやすいことがわかってきたので，改良システムでは単語３つ分とした．ちなみに char の実験でも，初期選択範囲は word の３単語とすること

で，ロングタップ初期選択における手法による影響を極力小さくしている．

- 文字列選択タスクにおいて何らかの問題が発生した場合に，次のタスクに移動したり，途中で実験を終了できるようにするため，SKIP ボタンと MENU ボタンを配置した．ただし，実験ではこれらの機能を使用していない．

4.6 実装した実験システムの詳細

本節では，スクリーンショットを用いながら，実装した実験システムの詳細について説明する．動作確認は Nexus7 で行っている．

起動時の画面を図 4.4 に示す．この画面では，7 種類の文書セットのうち，どの文書セットを用いてタスクを生成するかを選択することができる．例として，文書セット 3 を選択した画面が図 4.5 である．

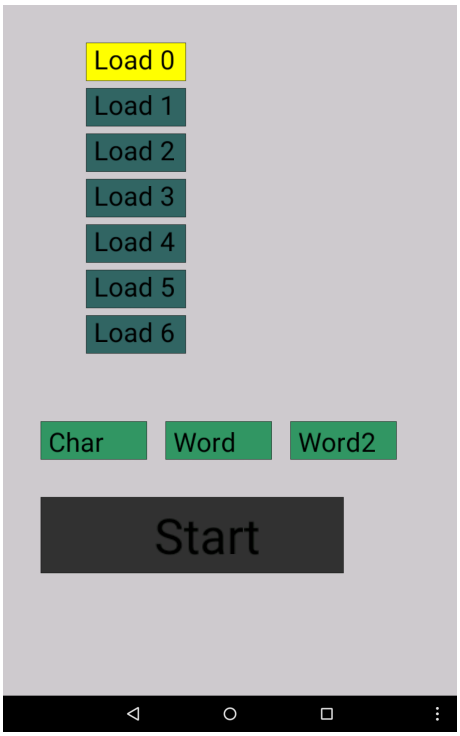

図 4.4: 起動時の画面

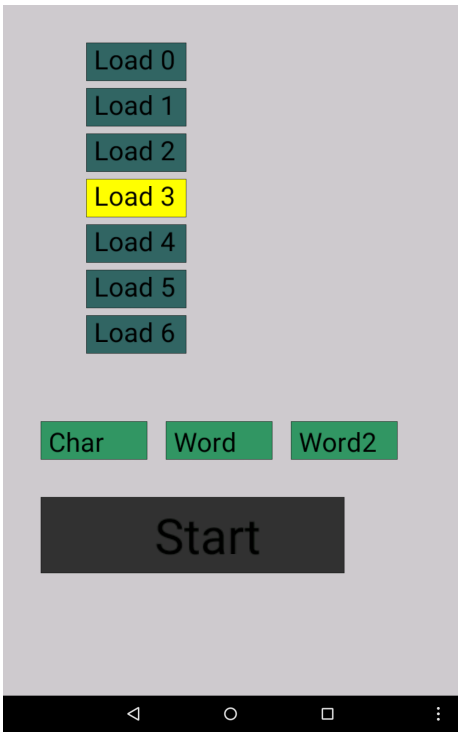

図 4.5: タスク生成時の文書を選択

次に，文字列範囲選択手法を 3 種類 (Char/Word/Word2) のなかから選ぶ．図 4.6 では例

として，Word を選択した様子を示している．すると，選択した文書セットを Word 手法を用いてタスクを生成する．タスクを生成すると，Start ボタンが有効になる．

ここで，Start ボタンを押すと，図 4.7 に示すように，タスクに含まれる一つ目の試行について，文章と，試行を完了するための選択範囲が青文字白背景で表示される．また，画面下には，現在実行しているタスクにおける，文字列範囲選択手法や，Answer ボタン，非常時に初期画面に戻るための Menu ボタン，現在の試行を中止し，次の試行に移行する Skip ボタンと，その反対で前の試行に戻る Back ボタンが表示される．ただし，今回の実験においては，画面下に表示されているボタンを使用する必要はない．

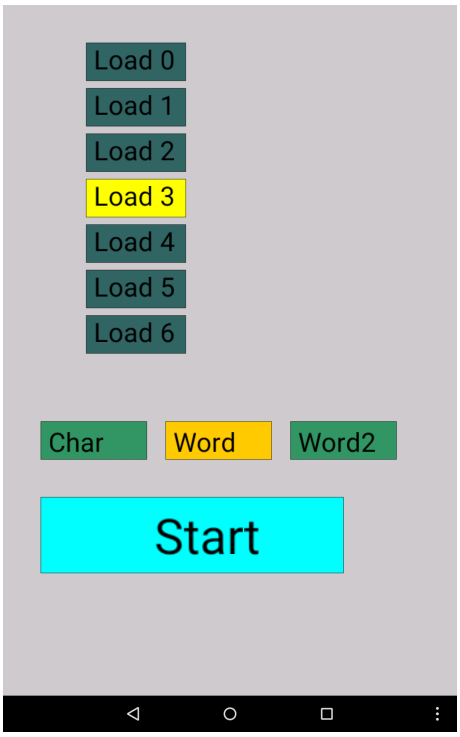

図 4.6: 文字列範囲選択手法を選択した画面

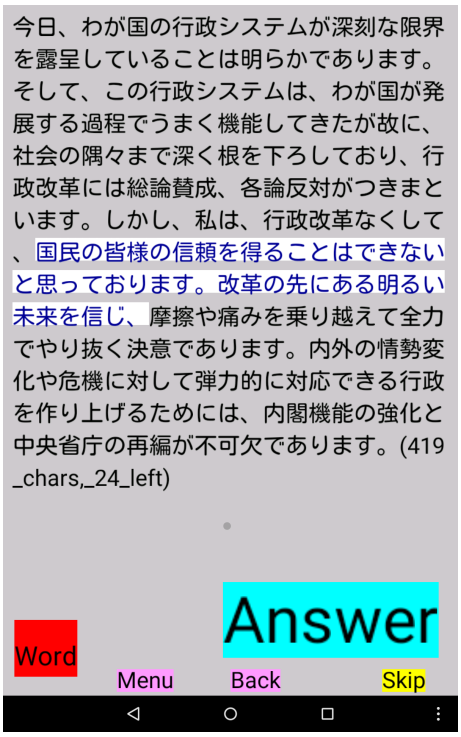

図 4.7: タスクが開始され，一つ目の試行が表示された画面

図 4.8 に,「国民の皆様」の「の」の上でロングタップして，初期範囲選択したときの画面を示す．青色半透明の始点ハンドルと，緑色半透明の終端ハンドルが表示される．また，範囲選択領域が赤文字黄背景で強調表示されている．ここで，緑色の終端ハンドルをドラッグすると，図 4.9 のように，選択範囲が変化する．

一つの試行は，始点ハンドルと終端ハンドルをドラッグによって操作して，青文字白

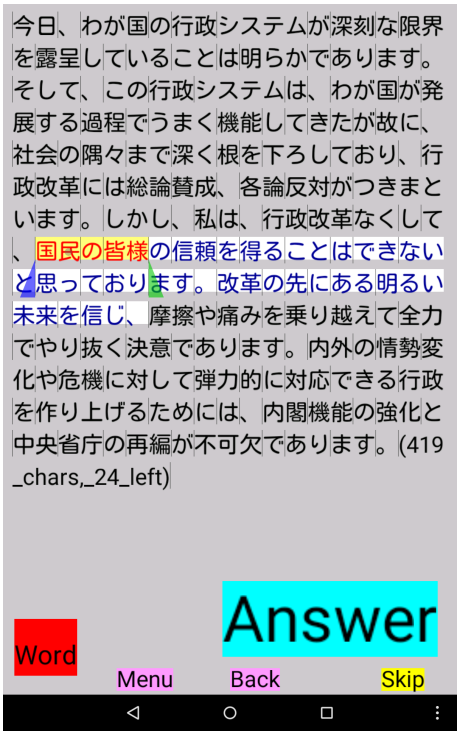

図 4.8: ロングタップによって初期範囲選択を実行

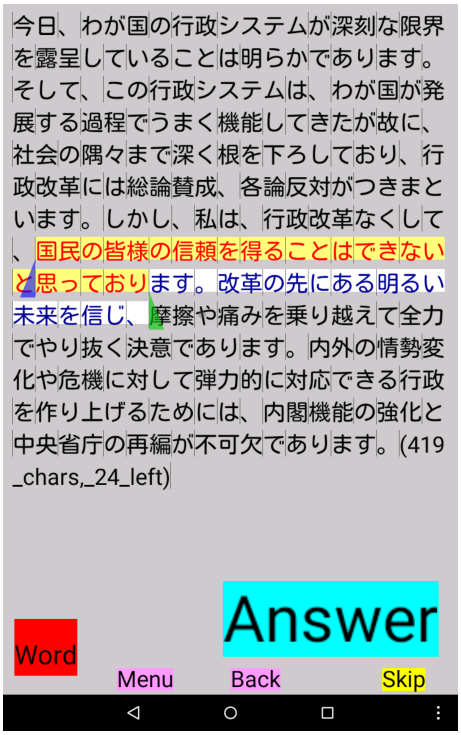

図 4.9: 終端ハンドルを操作し，選択範囲を変化させた

背景の文字列をすべて範囲選択した状態にしたのち，画面から指を離したときに自動的に終了し，次の試行が開始される．もし，範囲選択が試行で定められた領域を超えた場合には，図 4.10 に示すように，超えた部分（オーバーシュート部分）を赤文字ピンク背景で強調表示する．

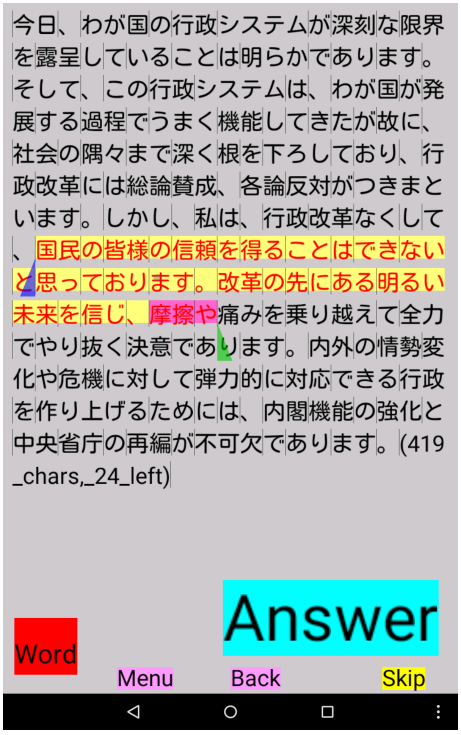

図 4.10: オーバーシュート表示

第5章　実験

第 5 章では，これまでに説明した実験システムを用いて，被験者実験を行った結果について述べる．

5.1　実験方法

我々は文字単位のハンドル移動と単語単位のハンドル移動について，比較実験用のアプリケーションを Processing for Android で作成し，Nexus 7 上で被験者実験を行った．

前章で述べたように，今回の実験においては，以下の条件

- 選択範囲の開始位置が画面の左側 or 右側
- 選択範囲の終了位置が画面の左側 or 右側
- 選択範囲が 1 つの改行を含む場合と，３つの改行を含む場合

の（$2 \times 2 \times 2 = 8$ 通り）について，それぞれ同じ回数（3 回ずつ）行ってもらうことによって，選択範囲の開始・終了位置の場所と，改行の量（文章の量）に対する条件を統制した．8 通りの試行を 3 回ずつ行ったので，合計で 24 回の試行によって 1 セットのタスクが構成される．

12 名の被験者には，３つの文字範囲選択手法 (char/word/word2) について，1 回ずつ，合計 3 セットのタスクを，十分な休憩をはさみながら行ってもらった．

5.2　実験システム

実験用の端末としては，第 4 章で述べたように，7 インチタブレット (Nexus7) を用いた．被験者には，24 回の文字列選択試行を 1 つずつ提示し，正確に範囲選択が行われて

いれば，決定 (Answer) ボタンを押さなくても自動的に次の試行が現れる実験設定とした．以前の実験では，選択を完了してから，画面下のボタンを押す必要があったが，自動的に次の試行に移行することにより，ボタン押し作業にかかる時間の影響が緩和され，タスク本来の遂行時間が正確に測れるようになった．なお，24 回の試行を完了すると，自動的に初期画面図 4.4 に戻るようにしている．

実験プログラムによって，各試行で記録した情報は，以下の通りである．

- msec: 試行完了にかかった時間（ミリ秒）
- touch: 画面に触れた回数（ロングタップやドラッグ）
- h1touch: 始点ハンドルに触れた回数
- h2touch: 終端ハンドルに触れた回数
- h1dragdist: 始点ハンドルをドラッグ操作した距離の合計
- h2dragdist: 終端ハンドルをドラッグ操作した距離の合計
- overshoot: オーバーシュートしていた累積時間
- miss: Answer ボタンを押したときに，正しく範囲選択していなかった回数 (今回の実験ではすべて 0)
- num: 試行番号
- mode: 手法 (Char/Word/Word2)
- lines: 改行の数 (1/3)
- beginpos: 選択範囲の開始位置が画面の左側 (1) or 右側 (2)
- endpos: 選択範囲の終了位置が画面の左側 (1) or 右側 (2)
- len: 試行で定義された選択範囲に含まれる文字数
- linenum: 試行で用いられた文章の ID

実験における文書は，Wikisource[1] の所信表明演説を利用した．句点で分割したのち，一文ずつ長さを調べながら連結していき，390 文字を超えたら 1 つの文書としてファイルに保存していく．このとき，テキストファイルの 1 行が 1 文書に対応する．合計で 400 文書を含むテキストファイルを作成した．

5.3 実験結果

実験の結果，12 名 ×3 セット ×24 試行=864 個の試行データが取得できた．

このうち，試行完了までにかかった時間が，6000 ミリ秒以上のデータが 16 個存在した．

この 16 個の試行データの平均ミリ秒は 8,729 で，標準偏差は 4,465 であった．

これらのデータは，試行完了までになんらかのトラブルにより，通常よりも長い時間がかかってしまったものと思われるため，以降は，この 16 個のデータを外れ値として除外し，残りの 848 個の試行データをもとに，分析を行うこととした．

手法がタスク遂行時間に与える影響の分析　図 5.1 に，この 6000 ミリ秒未満の時間で完了した 848 個の試行データにおける，char/word/word2 手法の比較を示す．

一元配置の分散分析を行った結果，char/word/word2 手法の作業時間の平均値に有意差が認められた．(F(2,33)=13.81, $p < .001$)

有意水準 5%において，ボンフェローニの多重比較を行った結果，char と word ($p < .001$)，char と word2 ($p < .001$)，word と word2 ($p = .027$) のそれぞれについて，作業時間の平均値に有意差が認められた．

このことから，提案手法である word 手法は，従来手法である char 手法に比べて，作業時間を短縮するのに有効であるといえる．また，word2 手法のほうが word 手法よりも，作業時間を短縮する効果が高いことがわかった．

改行の数がタスク遂行時間に与える影響の分析　我々は，char/word/word2 のそれぞれの手法について，選択範囲が 1 つの改行を含む場合と，3 つの改行を含む場合の 2 つの場合について，作業時間の平均値に差があるかどうかを調べた．

[1] http://ja.wikisource.org/wiki/カテゴリ:内閣総理大臣所信表明演説

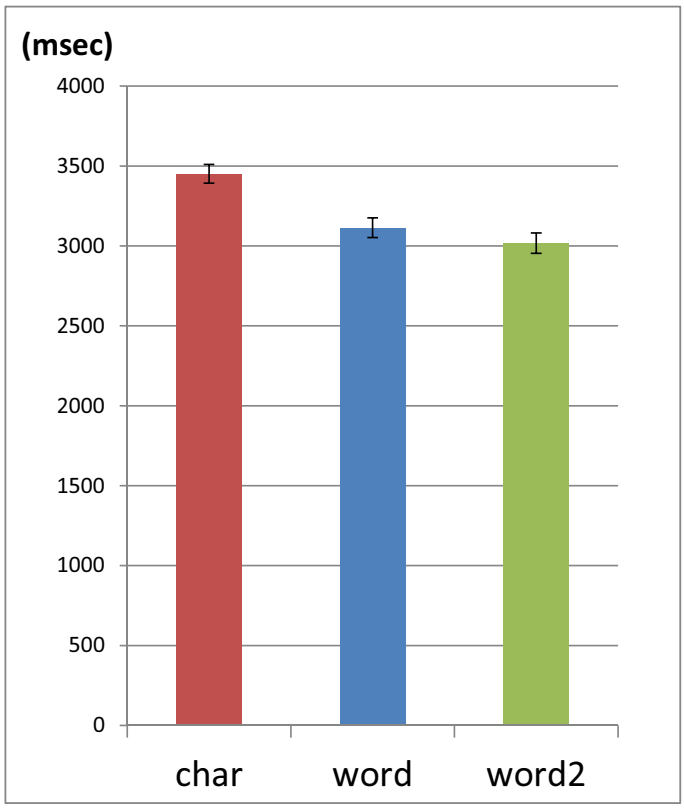

図 5.1: 6000 ミリ秒以内に完了したタスクにおける，char/word/word2 手法の比較

３つの手法 × ２つの改行数の合計 6 パターンにわけて，各被験者の作業時間の平均値を計算し，対応のある t 検定を行った．その結果，char (t(11)=0.63, $p = .540$), word (t(11)=0.27, $p = .793$) , word2 (t(11)=0.24, $p = .815$) すべてにおいて，有意水準 5%において有意差は認められなかった．このことから，改行の数が 1 つの場合と 3 つの場合とでは，作業時間における影響はないことが明らかとなった．

図 5.2 に，選択範囲の行数による影響を示す．

選択範囲の開始位置と終了位置がタスク遂行時間に与える影響の分析　つぎに我々は，char/word/word2 のそれぞれの手法について，選択範囲の開始位置と終了位置が作業時間に与える影響について調査した．

３つの手法について，２つの開始位置 × ２つの終了位置，４パターンにわけて，各被験者の作業時間の平均値を計算し，一元配置の分散分析を行った結果，char (F(3,44)=2.01,

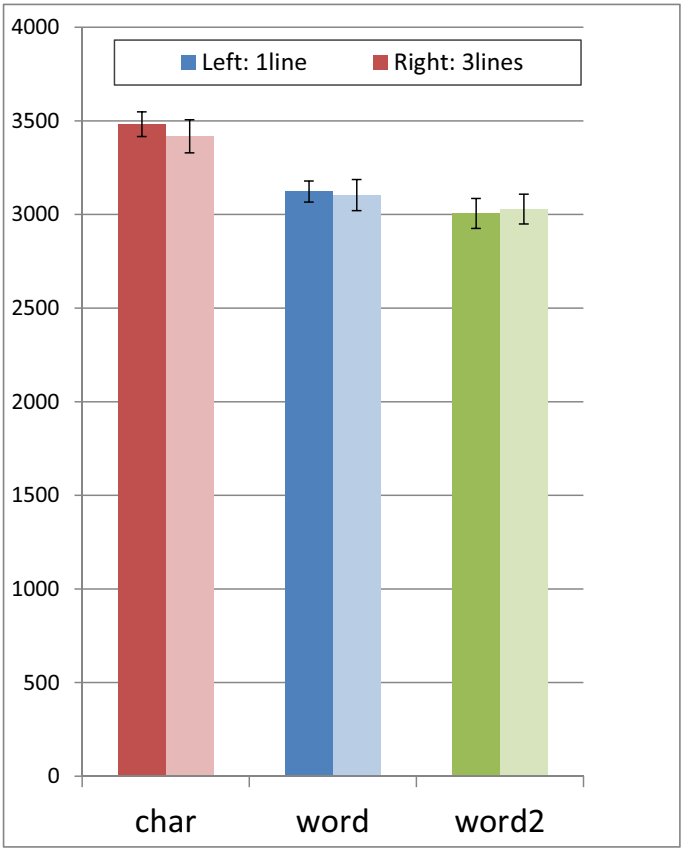

図 5.2: 選択範囲の行数

$p = .126$)，word (F(3,44)=0.27, $p = .848$)，word2 (F(3,44)=1.55, $p = .216$) すべてにおいて有意差は認められなかった．

ただし，char の LL と LR には有意傾向があった．LL は終了位置ハンドルをすぐ下に動かせばよかったので，左右の移動が少なかったことが主要な原因と考えられる．

図 5.3 に，選択範囲の位置と，作業時間の関係を示す．

5.4 考察

以前の実験においては，手法がタスク遂行時間に与える影響は限定的であり，明確な有意差は認められなかったが，今回の実験においては，手法がタスク遂行時間に与える影響について，統計的な有意差が現れた．その原因として，今回の実験では以前の実験

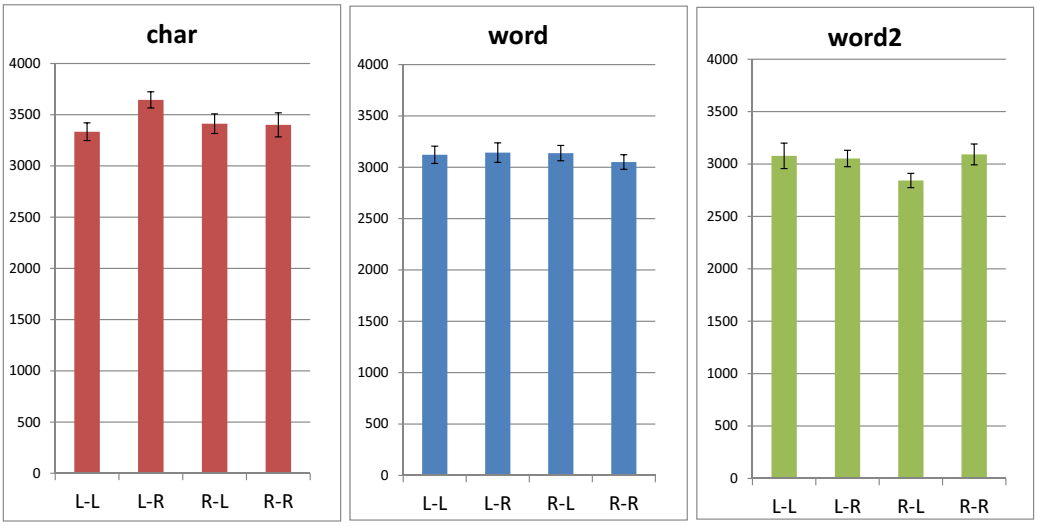

図 5.3: 選択範囲の位置

に比べ，タスクに含まれる試行の質（難易度）が統制できており，分散があまり大きくならなかったことが挙げられる．

また，本質的な文字列範囲選択タスク以外の要因（例えば，Answer ボタンを押すまでの時間や，ロングタップの失敗，ハンドル操作におけるホールドミスやハンドル移動が思い通りの場所に行かない状況など）が解決されたことによって，やはり本来のタスクに含まれる試行内容と，それを実施するうえでの手法の影響のみを実験データに残すことができたことが挙げられる．

また，以前の実験では含まれていなかった，word2 条件と，word 条件の比較が行えた点も有意義であった．これらの違いは単に，ひらがな 1 文字の単語を直前の単語に連結するか否かといった点だけであり，単語に含まれる文字数の平均値にそれほど大きな差はない．しかしながら，word2 のほうがタスク遂行時間が短くなったという事実から，やはり単語のグルーピング単位の大きさが，タスク遂行時間に直接影響していることが明らかとなったといえる．

このことから，今後の指針や課題として，グルーピング単位をどこまで大きくしても大丈夫かを検証したり，それに伴い失われる文字単位での選択の自由度を回復するような新しい手法を開発するといった研究の方向性が見いだせた．とくに後者については，早くハンドルを動かしているときはグルーピング単位が大きくなり，ゆっくり，かつ長時

間ハンドルをドラッグしていたら，単語内の文字単位で選択したいのではないかとシステム側が推測して，グルーピング単位を徐々に小さくするといった，動的なグルーピング単位設定の有効性や実用性を検証することが必要であると感じた．

第6章　まとめと今後の課題

本研究では，最所らが行った文字列範囲選択におけるグルーピングの影響を調べる実験について，より詳細な検討を行うための追実験を行い，単語単位の選択は，本当に単語数が少ない場合のみ有効なのか，また，行跨ぎタスクについて，本当に単語単位の選択は適していないのかといった点について，確認を行った．その結果，単語数によらず，また行跨ぎの有無によらず，単語単位でのグルーピングは文字列範囲選択タスクの作業時間を軽減できることを確認した．

また，本研究で独自に追加した，1 文字の助詞（ひらがな）を直前の単語と連結する手法は，従来の単語単位でのグルーピングに比べて，作業時間の軽減に貢献することも確認できた．

これらの疑問点が解決されたことによって，タブレット端末やスマートフォンへの単語単位の文字列選択手法の有効性が確認された．これらの結果は，今後タブレット端末やスマートフォンのより良いインタフェースの開発につながることが期待できる．

今後の課題としては，前章でも述べたように，グルーピング単位をどこまで大きくしても大丈夫かを検証することや，早くハンドルを動かしているときはグルーピング単位が大きくなり，ゆっくり，かつ長時間ハンドルをドラッグしていたら，単語内の文字単位で選択したいのではないかとシステム側が推測して，グルーピング単位を徐々に小さくするといった，動的なグルーピング単位設定の有効性や実用性を，実装および実験によって検証していくことが挙げられる．

謝辞

卒業論文を完成するにあたり，ご指導ご教授くださりました三浦准教授に御礼申し上げます．また，輪講や中間発表においてご指導やご教授を下さりました情報セクションの先生方に御礼申しあげます．加えて，本論文のデータ収集実験や評価実験において，被験者としてご参加頂きました三浦研究室の学生と情報セクションの学生にお礼を述べたいと思います．最後に，私の意思を尊重してくださり学部修了を応援して頂き，経済面や生活面において，ご支援をして頂いた家族に心から感謝申し上げます．

参考文献

[1] 最所賢至. タブレット型端末の特性を考慮した文章範囲選択手法. 九州工業大学工学部総合システム工学科卒業論文, February 2014.

[2] Motoki Miura and Kenji Saisho. A Text Selection Technique Using Word Snapping. In *Proceedings of the 18th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES2014)*, pp. 1644–1651, September 2014.

[3] Katie A Siek, Yvonne Rogers, and Kay H Connelly. Fat finger worries: how older and younger users physically interact with PDAs. In *INTERACT 2005*, pp. 267–280. Springer, 2005.

[4] Paul Kabbash and William AS Buxton. The “prince” technique: Fitts' law and selection using area cursors. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 273–279. ACM Press/Addison-Wesley Publishing Co., 1995.

[5] Leah Findlater, Alex Jansen, Kristen Shinohara, Morgan Dixon, Peter Kamb, Joshua Rakita, and Jacob O Wobbrock. Enhanced area cursors: reducing fine pointing demands for people with motor impairments. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pp. 153–162. ACM, 2010.

[6] Daniel Vogel and Patrick Baudisch. Shift: A Technique for Operating Pen-based Interfaces Using Touch. In *Proceedings of CHI 2007*, CHI '07, pp. 657–666, New York, NY, USA, 2007. ACM.

[7] Hrvoje Benko, Andrew D. Wilson, and Patrick Baudisch. Precise Selection Techniques for Multi-touch Screens. In *Proceedings of CHI 2006*, CHI '06, pp. 1263–1272, New York, NY, USA, 2006. ACM.

[8] A. Cockburn, D. Ahlström, and C. Gutwin. Understanding Performance in Touch Selections: Tap, Drag and Radial Pointing Drag with Finger, Stylus and Mouse. *Int. J. Hum.-Comput. Stud.*, Vol. 70, No. 3, pp. 218–233, March 2012.

[9] Vittorio Fuccella, Poika Isokoski, and Benoît Martin. Gestures and Widgets: Performance in Text Editing on Multi-Touch Capable Mobile Devices. In *Proceedings of CHI 2013*, CHI '13, pp. 2785–2794, 2013.

[10] Jean-Baptiste Scheibel, Cyril Pierson, Benoît Martin, Nathan Godard, Vittorio Fuccella, and Poika Isokoski. Virtual Stick in Caret Positioning on Touch Screens. In *Proceedings of the 25ième Conférence Francophone on L'Interaction Homme-Machine*, IHM '13, pp. 107:107–107:114, New York, NY, USA, 2013. ACM.

[11] 島佳吾, 箱田博之, 栗原拓郎, 志築文太郎, 田中二郎. 3 点タッチを起動ジェスチャとする範囲選択. 情報処理学会研究報告 HCI, pp. 1–8, July 2014.

[12] Glen Wallace, Robert Biddle, and Ewan Tempero. Smarter Cut-and-Paste for Programming Text Editors. In *Proceedings of 2nd Australasian Conference on User Interface*, AUIC '01, pp. 56–63, Washington, DC, USA, 2001. IEEE Computer Society.

[13] Reid Kerr and Wolfgang Stuerzlinger. Context-sensitive Cut, Copy, and Paste. In *Proceedings of C3S2E Conference*, C3S2E '08, pp. 159–166, New York, NY, USA, 2008. ACM.