筑波大学大学院博士課程

システム情報工学研究科修士論文

ユーザの手元注視を減らす QWERTYベースソフトウェアキーボード

久野 祐輝

修士（工学）

（コンピュータサイエンス専攻）

指導教員　田中 二郎

2014年3月

概要

タッチパネル搭載端末の文字入力手法の 1 つとして、QWERTY 配列のソフトウェアキーボードが存在する。ソフトウェアキーボードは物理キーボードと異なり、入力した際の物理的なフィードバックが存在しない。そのため、ユーザはキーに指を合わせることが困難になる。物理キーボードにおいては、ユーザは習熟次第で手元を注視せずに入力を行うことが可能であるが、ソフトウェアキーボードではそれが難しい。一方、ソフトウェアキーボードはそのキーの位置と大きさを自由に設定してユーザに合わせることが可能である。

本研究の目的はユーザが手元を注視する回数を減らすことである。そのために、キーの位置と大きさをユーザに合わせるソフトウェアキーボードのプロトタイプとして “Leyboard”と “Meyboard”を示す。

Leyboard はソフトウェアキーボードのキーをユーザの指の位置とその周辺に配置することによって入力を容易にする。Leyboard の実装後、長期間に渡る実験を行った。この実験は Leyboard の有用性と、その問題点を明らかにした。Leyboard は既存のソフトウェアキーボードに対して有意に高速な入力が可能であるという結果を得た。しかし、手元注視を減らすという目的は Leyboard では達成されなかった。

実験の結果を受けて、手元注視を減らすために Meyboard を設計、実装した。Leyboard においては中段のキーを上下段のキーと間違える、あるいは上下段のキーを中段のキーと間違えるというエラーが頻出した。そこで Meyboard では中段のキーを排除することによりエラー率を減らし、かつキーの数が減ることにより手元注視の回数も減らすことを試みる。Meyboard では上下段のキーのいずれかにおけるフリック入力を中段のキーの入力と解釈することによって、排除された中段のキーの入力を補っている。Meyboard の実装後、手元注視が可能な状態とそうでない状態のそれぞれの条件における Meyboard の入力性能を検証する実験を行った。その結果、手元注視の可不可は入力率に有意差を与えないという結果を得た。しかし、手元注視が不可能なとき、Meyboard のエラー率は高くなることが判明した。そこでその原因を考察し、今後の Meyboard の改良案を示した。

目 次

図 目 次

表 目 次

第1章　序論

本章では本研究の対象となるQWERTY配列を基にした（これを本研究ではQWERTYベースと呼ぶ）ソフトウェアキーボードについて議論するためにまずQWERTY配列のソフトウェアキーボードについて述べる。そしてその問題点及び利点について述べ、QWERTYベースのソフトウェアキーボードにおける手元注視を減らすという本研究の目的と、そのアプローチを示す。

1.1　QWERTY配列のソフトウェアキーボード

タッチパネル搭載端末の文字入力手法の1つとして、QWERTY配列のソフトウェアキーボードが存在する。このキーの配置はQWERTY配列の物理キーボードに基づく。

QWERTY配列のソフトウェアキーボードのキーは表示空間の制約などから、特に数字や記号キーにおいてQWERTY配列の物理キーボードと全く等しい配置を持つとは限らない。但し、アルファベットキーの位置に関してはQWERTY配列に従っており、3段に並んで配置されている。3段のそれぞれを上段、中段、下段と呼ぶことにすると、アルファベットキーはQWERTY配列の上段にq、w、e、r、t、y、u、i、o、pが、中段にa、s、d、f、g、h、j、k、lが、下段にz、x、c、v、b、n、mがそれぞれ存在することになる。

1.1.1　ソフトウェアキーボードの問題点

QWERTY配列のソフトウェアキーボードはその入力において問題が存在する。それは物理キーボードと異なり、入力した際の物理的なフィードバックが存在しないことである。物理的なフィードバックが存在する物理キーボードにおいては、ユーザは習熟次第で手元を全く見ずに入力を行うことが可能である。しかし、ソフトウェアキーボードにおいては、物理的なフィードバックを得ることができない。物理キーボード入力時、ユーザはキーを入力した感触から自分の指がどのキーを押したのか把握できる。よって手元注視なしに意図するキーに指を合わせて、入力を行うことが可能になる。ソフトウェアキーボードでは自分の指がどのキーを押したのか把握するためには、入力結果を見るか手元を注視する必要がある。そのため、手元注視なしに意図するキーに指を合わせ、入力を行うことは困難になる。

1.1.2　ソフトウェアキーボードの利点

ソフトウェアキーボードは先に挙げた欠点が存在する一方、物理キーボードにはない利点も存在する。それはそのキーの位置と大きさを自由に設定できることである。よって、ソフトウェアキーボードはこれらの要素をユーザに合わせることが可能である。この利点を活用することにより、物理的なフィードバックの欠如による弊害を低減し、ユーザの手元注視を減らし、ソフトウェアキーボードの入力性能を向上することができると筆者は考えた。

1.2　入力に関する用語の定義

以下に本論文における入力に関する用語を定義する。

- タップ: ユーザが 1 本または複数本の指を画面に置いた後に、短時間内にそこから離す入力。
- スワイプ: ユーザが 1 本の指を画面上に置いたまま移動させる入力。
- フリック: ユーザが 1 本または複数本の指を画面に置き、一定時間内に一定以上の距離を移動させてそこから離す入力。
- プレス&ホールド: ユーザが 1 本または複数本の指を画面に置いた状態を一定時間以上保つ入力。

1.3　本研究の目的とアプローチ

本論文では、キーの位置と大きさをユーザに合わせるという考えに基づいたソフトウェアキーボードの入力性能を調査することを目的とする。そのアプローチとして、キーの位置と大きさをユーザの手に合わせるソフトウェアキーボードを 2 種類提案する。

1 つ目はソフトウェアキーボードのキーをユーザの指の位置とその周辺に配置することによって入力を容易にするものである。具体的には、ユーザの指の位置に中段のキーを配置し、その周辺に上段や下段のキーを配置することによって、キーの位置と大きさをユーザに合わせている。このプロトタイプを本論文では “Keyboard”の “K”を 1 文字進めた “Leyboard”と呼ぶ。

- キーをユーザの指の設置位置とその周囲に配置することによりユーザの指の位置にあったキーの配置を行う。
- 親指のキーを入力する際、人差し指から小指のキーが親指方面へ平行移動する。これによって親指のキーと、人差し指から小指のキーの同時入力を行う際のユーザの手の変形を抑えられる。さらにこの機能は複数のキーを入力する際にもキーの位置をユーザの指に合わせるという利点を持つ。本機能は本研究の中核をなす。（親指基点スライド）
- 親指回りに配置されたキーにおいて、ユーザは画面上にてキーからキーへ指を移動させる（スワイプする）ことによって、連続した入力が可能である。（親指スワイプ入力）

- 様々なキーの入力が行えるように、複数のキーセットを設けた。これらは親指回りに配置されたキーによって切り替えられる。このとき、親指スワイプ入力によってキーセットの切り替えと修飾キーの入力を 1 回のスワイプ操作にて行うことが可能になる。また、親指基点スライドによってユーザの手の変形は抑えられる。

Leyboard を実装した後に、その入力速度や入力エラー率を調べる被験者実験を長期期間に渡って行った。この実験は Leyboard の有用性と、その問題点を明らかにした。Leyboard は既存のソフトウェアキーボードに対して有意に高速な入力が可能であるという結果を得た。しかし、手元注視に関しては通常のソフトウェアキーボードと同等の注視の必要性があった。そこで、注視回数を減らすべく新たなプロトタイプを実装した。これが本論文において提案する 2 つ目のソフトウェアキーボードである。

Leyboard の被験者実験の結果、上下段のキーを中段のキーと間違える、あるいは中段のキーを上下段のキーと間違えるというエラーが多く確認された。これを回避するために、QWERTY 配列のレイアウトから中段のキーを取り去ることを考えた。上記の考えに基づき、中段のキーを排除したソフトウェアキーボードのプロトタイプを設計した。これを本論文では “Keyboard” の “K”を 2 文字進めた “Meyboard”と呼ぶ。

Meyboard の新規性は以下の点である。

- 中段のキーを排除することにより、上下段の 2 段構成のシンプルなレイアウトとなる。これにより中段のキーが存在することによる入力誤りをなくす。
- 同じタッチ位置においても入力方法によってキー入力の挙動を変える。これにより中段のキーを入力可能にする。具体的にはタップの場合はタッチ位置のキーを入力し、フリックの場合は中段のキーを入力する。
- 一度に表示されるキーの数を 20 個に減らす。この 20 個のキーは上下段のキーを表示する。そのため Space や Enter キーも排除されるが、これらの入力は複数の指によるタップによって行う。キーの数が少なくなることにより、ユーザは手元注視を減らしても正確な入力が可能となる。
- ユーザの指の設置位置を基にキーの横幅の大きさを決定することにより、ユーザの指の位置にあったキーの配置を行う。
- Leyboard と同様に、複数のキーセットを設けた。これらは左手人差し指と中指のフリックによって切り替えられる。2 本の指を動かすのみでキーセットを切り替えられるので手の移動が抑えられ、誤入力が起こりにくくなる。

本研究は QWERTY ベースのソフトウェアキーボードにおける手元注視を減らすために、キーの位置、大きさ、数及び入力方法に変更を加えるものである。本研究において提案するキーボードのユーザは QWERTY 配列の物理キーボードの入力に習熟していることを想定している。そのため、ソフトウェアキーボードのキー配置や入力方法が QWERTY 配列の物理キーボードに近いものであるほど、そのソフトウェアキーボードはユーザにとって習熟しやすいものとなる。しかし、キー配置や入力方法が QWERTY 配列の物理キーボードに近い既存のソフトウェアキーボードでは、先に挙げたように、キーへのユーザの指の合わせにくさから

手元注視回数が増大する。したがって、ユーザの習熟のしやすさと手元注視回数はトレードオフの関係にある。本研究はユーザの習熟のしやすさと手元注視回数のトレードオフを探る。

1.4 本論文の構成

以下に本論文の構成を記述する。第 2 章では既存の文字入力に関する研究を挙げ、Leyboard 及び Meyboard とそれらを対比する。次に Leyboard の機能や設計に関して第 3 章にて述べる。Leyboard を実装した後に行った、入力速度と入力エラー率を調べる被験者実験の内容及び結果を第 4 章にて記述する。実験結果からの考察と Meyboard の仕様を定めるための知見を第 5 章に記述する。そして、実装された Meyboard の設計と機能に関して第 6 章に記述する。Meyboard を実装した後に、その入力速度と入力エラー率、さらに手元注視を減らす効果を調べる被験者実験を行った。この内容や結果は第 7 章に、結果から得た知見に関する考察を第 8 章にて記述する。最後に第 9 章にて本研究をまとめ、結びとする。

第2章　関連研究

本章では既存の文字入力に関する研究を挙げ、Leyboard 及び Meyboard との関連性や違いについて述べる。

2.1　キーの形状や位置をユーザに合わせるソフトウェアキーボードの研究

本研究と同様にソフトウェアキーボードのキーの形状や位置をユーザに合わせるアプローチを採っている研究として、LiquidKeyboard[SLL11]、CATKey[GE07]、Personalized Input[FW12]、Gunawardana らの研究 [GPM10]、Go らの研究 [GT10] がある。LiquidKeyboard と Go らの研究以外にはユーザのキー入力に応じてキーの配置を変化させていくフェーズが存在する。一方、本研究ではユーザに適応させるためにキー入力を行うフェーズを必要とせずに、キー入力を伴わないキャリブレーション操作のみで配置が確定する。これはユーザの指の位置を取得するという単純なキャリブレーション操作のみでもこれらの研究と比較して十分なパフォーマンスを得られると考えたからである。何故ならばユーザの習熟度の問題もあるが、CATKey では入力率が通常のソフトウェアキーボードと比べて低く、Personalized Input では入力率が高くなってはいるものの劇的な向上が見られなかったからである。先ほど除外した 2 つのうち、LiquidKeyboard は、本研究同様にキャリブレーション操作のみでキーの配置が確定する。Leyboard ではさらに親指基点スライドによって修飾キー（Ctrl、Shift 等）とアルファベットキーの同時入力を行う際にも、キャリブレーションで取得したユーザの指の位置に合うようにキー配置を変更することが可能である。Meyboard ではフリック入力により修飾キーとアルファベットキーの同時入力を 1 回の入力として統合した。Go らの研究はソフトウェアキーボードにおいてユーザの指が触れているキー（指の触れている面積に応じて複数存在し得る）をパイメニューの形式で大きく表示することにより入力しやすくしている。これは携帯音楽プレーヤのような小型デバイスにおける利用を想定しており、パイメニューは限られた入力空間において快適な入力を可能にするために設けられた。Leyboard は物理キーボードにほぼ等しい画面の大きさを前提としている。Meyboard でも、その高さは小さくなったが、幅に関してはやはり物理キーボードにほぼ等しい画面の大きさを前提にしている。また、これらの研究で示されるソフトウェアキーボードの多くは、アルファベットを入力可能であるが、数字キーと一部の記号キーが入力不可能である。本研究ではこれらのキーも入力可能にする。

Beyond QWERTY[FLW12] もソフトウェアキーボードにおいて入力にパイメニューを利用する研究である。これは通常のソフトウェアキーボードの入力に加えて、ユーザがキーを長押

しした際に修飾キーを入力するためのパイメニューを表示する。Meyboard ではパイメニューの表示はない（キーに入力方法が最初から記載されている）が、フリック入力により修飾キーの入力などを行うため、これに近い設計となっている。また、Beyond QWERTY ではジェスチャ入力（手書き入力）による記号の入力も行われている。Meyboard はキーの数が制限されているが、ジェスチャ入力の方は入力に時間がかかるとして採用しなかった。代わりに、キーセットを切り替えて表示するキーを変更している。

2.2 タップによって入力を行うソフトウェアキーボードの研究

本研究ではキーの入力をタップにて行うこととした。TapBoard[KSL+13] は本研究と同様にタップにて入力を行うソフトウェアキーボードである。TapBoard は物理キーボードと同様に、ユーザがキーの上に指を置いた状態からの入力を行えるようにした。その方法はキーの入力をタップによって行うこととして、指を置いた状態と入力を区別することであった。そのため、両者を区別するためのタッチ時間の閾値を設けている。本研究においてキーの入力をタップによって行うのは、キャリブレーションを行うために指を置いた状態とキーの入力を区別するためである。また、Meyboard ではユーザがキーの上に指を置いた状態からの入力も可能になっている。

2.3 bimanual な操作を行うソフトウェアキーボードの研究

Don らは bimanual な操作をソフトウェアキーボードに取り入れた [DS10]。ユーザはまずキーボード上の大まかな範囲を片手で選択する。するとユーザのもう片方の手の位置にその範囲がズーム表示されるのでユーザはそこから入力するキーを確定する。Bimanual Gesture Keyboard[BCO+12] も bimanual な操作によって入力を行うソフトウェアキーボードである。そのキー配列は QWERTY 配列を格子状に並べ直し、さらに左右の 2 つに分割したものになっている。ユーザの左親指の位置に、分割されたキーボードの左部分が、右親指の位置に、分割されたキーボードの右部分が表示される。ユーザは入力する単語に含まれるアルファベットのキーを通るように、親指をスワイプさせることによって入力を行う。本研究では、キーセットを切り替えてから入力を行う際に bimanual な操作となることがあり、類似性が存在する。また、Meyboard は Bimanual Gesture Keyboard のように QWERTY 配列を格子状に並べ直したキー配列を持つ。

2.4 点字入力を基にしたソフトウェアキーボードの研究

点字入力を基にしたソフトウェアキーボードのデザインに関する研究として、BrailleType[OGN+11b] や BrailleTouch[SCF+12] がある。点字入力を基にしているので完全なアイズフリー状態における入力が可能であるが、点字を知らないユーザには習得が困難である。本研究ではユーザ

の慣れの観点から、QWERTY 配列を基としたデザインにおいて手元注視を減らすことを試みる。

2.5 種々の文字入力手法の入力性能を比較した研究

Ko らは巨大なタッチパネル端末であるテーブルトップにおける種々の文字入力手法の入力性能について調査した [KKKE11]。結果、エラー率にはさほど差はないが、QWERTY 配列が最も速度に優れていた。これは QWERTY 配列を基にする本研究のデザインの妥当性を補強する。

Oliveira らは盲人の QWERTY、MultiTap、NavTouch、BrailleType それぞれのソフトウェアキーボードにおける入力を評価した [OGN+11a]。但し、ここにおける QWERTY は split-tapping[KBW08] または double tap を用いている。split-tapping は盲人がソフトウェアキーボードにおいて入力を行う手法である。ユーザは人差し指のスワイプによって入力対象を選択し、中指のタップによって入力対象を確定する。double tap は人差し指のスワイプによって入力対象を選択し、そのタップを 2 回行うことによって入力対象を確定する手法である。MultiTap は 12 のキーにアルファベットを複数ずつ割り当て、その入力回数から入力文字を確定する。すなわち、MultiTap はテンキーを搭載した携帯電話における入力のメタファである。NavTouch はアルファベットをアルファベット順に格子状に並べ、縦方向と横方向のスワイプ入力を組み合わせることによって入力対象を選択、決定する手法である。Oliveira らが行った評価の結果、入力速度が速い順に QWERTY、MultiTap、NavTouch、BrailleType となり、エラー率の低い順に BrailleType、NavTouch、QWERTY、MultiTap となった。ここから QWERTY 配列は入力率に優れるが、画面注視が不可能であるとエラー率が高くなることが分かる。よって、QWERTY ベースのソフトウェアキーボードの手元注視を減らすためには何かしらの工夫が必要である。Oliveira らの評価結果は手元注視を減らすための工夫を行う本研究の妥当性を補強する。

2.6 QWERTY 配列のキーの数を減らす研究

Meyboard では中段のキーを排除することにより、エラー率とユーザの手元注視を減らしている。このように QWERTY 配列のキーの数を減らすソフトウェアキーボードの研究が存在する。1 Line Keyboard[LGYT11] は QWERTY 配列のキーを 1 段構成の 8 つのキーにまとめている。そのため、誤入力は生じにくくなったが、キーをまとめたために入力を候補から確定するための変換操作とそのための時間を必要とする。これは 1 Line Keyboard への習熟度を問わず必要となる時間である。片山らの研究では QWERTY 配列を左右に分割し、その左半分を排除した物理キーボードを提案した [片山 09]。このキーボードではユーザは右半分に存在するキーのみの入力を行う。単語の切れ目に到達したとき、システムは左手の入力を補完し、ユーザが意図した単語の候補を提示する。ユーザはその中から入力を確定する。これも変換操作の一種であると言える。本研究は習熟による入力率の向上が変換操作によって現れにくくならないように、変換操作を伴わない入力方法にする。また、変換操作を伴わない Meyboard

では、言語に依存しない入力が可能である。但し、その言語の単語がアルファベットによって構成される場合に限られる。Meyboard では存在しない中段のキーを入力するために、入力がタップの場合はタッチ位置のキーを入力し、フリックの場合は中段のキーを入力する仕様になっている。

2.7 フリック入力を QWERTY 配列に取り入れた研究

フリック入力を QWERTY 配列に取り入れた研究としては Gestyboard[CAP$^+$12] が挙げられる。Gestyboard はタップの場合中段のキー、フリックの場合該当方向のキーの入力となる。このため、入力するキーに応じてフリック方向を使い分ける必要がある。Meyboard は Gestyboard のように入力キーによってフリック方向を使い分ける必要はないので、フリック方向に特に制約は設けない。そのため、習熟はより簡単であると思われる。また、Blossom[桜井 13] では QWERTY 配列のソフトウェアキーボードにおいてフリック入力を利用して日本語入力を行っている。ユーザは子音となるアルファベットのキーの上から、母音を決定する 5 方向のフリック入力を使い分けることにより日本語入力を行う。Meyboard ではフリック入力は中段のキーと大文字の入力に用いられ、ユーザが使い分けるフリック方向は少ない。

2.8 物理キーなしに文字入力を行う研究

タッチパネル上における文字入力に関するものではないが、ソフトウェアキーボードと同様に物理キーが存在しない入力対象において文字入力を行う研究が存在する。これらの研究では手元にキーが表示されることはないので、ユーザは必然的に手元注視に頼らない入力を求められる。

Mujibiya らの研究 [MMR10] ではタッチパネルではない机等の平面上における指の動きをカメラで計測する。入力面はプロジェクタによって出力される赤、緑、青の縞模様によって構成する。ユーザはこの入力面上において入力を行う。カメラを用いて奥行きを計測することによって、入力面上における指先の動きと入力面への接触の有無を把握する。このようにして、入力するキーを決定している。Goldstein らの研究 [GBAT99] では、データグローブを用いて文字入力を行っている。データグローブは指に固定された圧力センサによってユーザが入力した指を判断するものである。Guesso[藤田 09] では、ユーザの指の先端にスイッチを取り付けて文字入力を行う。上記 2 つの研究において、それぞれのシステムが知ることができるのはユーザが入力を行った指に関しての情報のみである。QWERTY 配列の文字入力において、各指の担当するキーが決まっていることを利用して、システムは入力された指の順番から字句解析を行い、入力された単語を推定する。ユーザはシステムが推測によって作った候補群の中から入力を確定する。

上記の研究ではユーザは QWERTY 配列に従った指の動かし方によって単語の入力を行う。本研究においても、ユーザは QWERTY 配列に沿った文字入力を行う。但し、本研究では字句解析を行わず、ユーザがタッチパネルの画面上に指を置いた位置から入力するキーを決定す

る。そのため、アルファベットにて表現可能な範囲においては、言語に依存しない入力が可能である。

FingeRing[FS94] もユーザの指の動きを検知し、入力を行う。FingeRing はユーザの各指の付け根に指輪型のセンサを装着して指の入力の検知を行う。但し、FingeRing は QWERTY 配列に沿った文字入力手法ではなく、ユーザの各指の置き方のパターンを文字に割り当てている。

2.9 QWERTY 配列のキーボード以外の文字入力に関する研究

QWERTY 配列の物理キーボードの他にも、様々な文字入力手法が考案されている。本研究では QWERTY ベースのソフトウェアキーボードを対象としたが、ここで挙げる入力手法をソフトウェアキーボード上において再現することも手元注視を減らすための手段として考えられる。

Proschowsky[PSJ06] らや、Slay ら [SFM08] はホイールの操作による文字入力を考案した。但し、前者はプロトタイプの実装をタッチパッド上にて行っている。TwoStick[KIG07] はビデオゲームに用いられるコントローラに搭載された 2 つのジョイスティックを利用して文字入力を行う。CLURD[JK09] はハングル文字と英字の文字入力の研究である。CLURD は 5-way key という中央ボタン付きの矢印キーを利用し、ボタンを押す回数、順番、時間の組み合わせにより入力文字を決定する。ユーザはこの入力方法を覚えなければならないため、学習難易度は高いと思われる。これらの研究はボタンを押す順番や指の動きのパターンを文字に割り当てることによって入力を行う。この点はスワイプやフリック入力を用いて入力を行う本研究と類似する。

Hiraga らはキーボードの漢字無連想変換方式として T-CODE を考案した [HOY80]。日本語入力において現在一般的に用いられている変換は連想式であり、「トキ」と「時」のようにキーの組み合わせと文字の間に連想がある。一方、無連想式はこの連想関係がない。端的に言えば数ストローク （T-CODE では 2）分のキー入力を 1 つの文字に割り当てることにより無連想を実現している。連想式は習熟が容易であるが、変換に思考を割くために一定以上入力速度が上がらなくなる。無連想方式は変換の際の思考が不要なので人間やハードウェアの限界を無視すれば入力速度は無限に向上する。但し全ての文字 （かな漢字数字）に対して入力方法を覚えなければならないので学習難易度は非常に高い。本研究にて対象としたソフトウェアキーボードは QWERTY ベースであり、入力速度よりもユーザの習熟の容易さを重視している。しかし、無連想式は多数の文字を 30 のキーに割り当てるためのアイデアであり、キーセットの切り替えによって、少数のキーに多数の文字入力を割り当てるソフトウェアキーボード全般との類似性がある。

2.10 キーの形状や位置をユーザに合わせる物理キーボードの研究

キーの位置をユーザに合わせた物理キーボードに TRON キーボード [坂村 86] が存在する。そのキー配置は直線状ではなく、人間の手の形に合わせて湾曲している。TRON キーボード

の英字の配列は QWERTY ではなく DVORAK に従っている。日本語入力に関しては日本語の特性に応じて独自に論理的に定めた配列が適用される。Leyboard もキャリブレーションにより、ユーザの手に合わせてキーが湾曲する配列となる。本研究ではユーザの習熟のしやすさの観点から配列は QWERTY を基にした。

第3章　Leyboard

キーを指の設置位置とその周囲に配置するソフトウェアキーボードのプロトタイプとしてLeyboardを開発した[KST12, KST13]。本章ではこのプロトタイプの設計と機能に関して述べる。

Leyboardのキーはタッチパネル上のユーザの指の設置位置と、QWERTY配列を基に配置される。ユーザの各指の位置にはa、s、d、f、Space、Enter、j、k、l、セミコロン（;）キーが配置される。以降、これらのキーをホームポジションキー、残りのキーを非ホームポジションキーと呼ぶ。非ホームポジションキーはホームポジションキーの周囲に配置される。

各キーの形状はボロノイ図を描画することにより決定される。

3.1　キー入力

Leyboardのキー入力はタップにて行う仕様になっている。タップとしたのはキーの位置を決定するキャリブレーションと、キーの入力を衝突させないためである。

ユーザが指をタッチパネル面に接触させてから離すまでの時間で判定を行い、その時間が短かったときをタップ入力とみなす。タップ入力であるかを判定する時間の閾値は経験的に設定した。その時間は平均的には140ミリ秒であった。固定値になっていないのは実装上の問題であり、Leyboardの設計上の必然ではない。実際にはユーザが指を画面に置いている間呼び出され続けるメソッドにおいて、カウンターとなる変数の値（初期値0）を1ずつ加算している。その値が10を超えるまでの時間がタップ入力であるかを判定する閾値となっている。

タップ入力とならない例外のキーはShiftキー等の修飾キーと後述するキーセット切り替え用のキー、Back Space、Deleteであり、これらのキーでは、ユーザがそのキーの上に指を置く間入力が行われる。

3.2　キーの位置座標と領域

Leyboardではキーはそれぞれ位置座標としてx、y座標の値を持つ。これがLeyboardにおける各キーの位置を決定する。

Leyboardではユーザの入力の際に、ユーザのタッチ位置に最も位置座標が近いキーを入力の対象とする。これは、ユーザがキーを押しやすいように各キーの面積を最大限大きくするためである。キーとキーの境界がユーザに分かるように、キーの位置座標を基にしたボロノイ図が描画される。ボロノイ図を描画したのは、タッチ位置に最も位置座標が近いキーを入力

の対象とするときにキーとキーの境界はボロノイ図と等しくなるためである。ボロノイ図の描画には Fortune's algorithm[For86] を使用した。このアルゴリズムは計算量が $O(n \log n)$ であるという特徴から、高速な描画が可能である。なお、Fortune's algorithm を実装する際には公開されているコード[1]を移植した。

3.3 キャリブレーション

本研究では Leyboard のキー配置を決定する行程をキャリブレーションと呼ぶ。

3.3.1 キャリブレーションの方法

Leyboard は、ユーザが両手全ての指（10 本の指）をタッチパネル面に置いたときに、各指の位置にホームポジションキーを配置し、非ホームポジションキーをその周囲に配置する。ユーザが画面上に全ての指を置いている限り、キーの位置は確定しない。このときユーザが指を動かすとそれに合わせてキーの位置も移動する。ユーザがいずれかの指をタッチパネル面から離し、ユーザが置いた指の数が 10 本未満になるとキーの位置が確定し、キャリブレーションが終了する。

キャリブレーションを行うことによって、キーの位置がユーザの指の位置に合うようなキーの配置が作られる。すると意図したキーが入力しやすくなると考えられる。

3.3.2 キャリブレーションとキー入力の衝突回避

Leyboard においてはユーザが 10 本の指を置いた際に必ずキャリブレーションを行うとして、入力を行えなくした。入力を行えなくしたことによって、キャリブレーション時に誤ってキー入力が行われないようにしている。但し、入力を行えなくしたことによって、関連研究の TapBoard のような指を置いた状態からの入力も不可能になった。ゆえに、Leyboard では少なくともいずれかの指が浮いた状態においてユーザは入力を行う。厳密にはユーザが画面に置く指の数が 8 点を超えた場合に入力が行えなくなるようになっている。そして、ユーザが 10 本の指を置く度にキャリブレーションが行われる。

3.3.3 各タッチと指の対応

Leyboard は、ユーザが両手全ての指（10 本の指）をタッチパネル面に置いたときに、各タッチと指の対応をとる。各タッチがどの指によるものであるかは以下のように判定する。まず画面下部に最も近いタッチ 2 つは親指によるものとみなす。それ以外の 8 つのタッチを幅方向の座標値によってソートする。ソートされたタッチに対して、座標値の昇順に左手小指、左

[1] `http://blog.controul.com/2009/05/speedy-voronoi-diagrams-in-as3flash/`（参照 2014-01-19）

手薬指、左手中指、左手人差し指、右手人差し指、右手中指、右手薬指、右手小指を割り当てていく。

3.3.4 キーの配置規則

各キーの具体的な配置規則を以下と図 3.1 に示す。キャリブレーション時の人差し指と薬指の位置を結ぶ線分を考える。次にホームポジションキーの座標を通りかつ先の線分に平行な直線を各指において考える。その直線においてホームポジションキーの位置から右（左手ならば薬指から人差し指への、右手ならばその逆方向）に 17mm 離れた位置を基準とする。親指の場合は担当するキーの数が多いので、誤入力を避けるために距離を他の指よりも大きめにとって 23mm とした。17mm、23mm という数値は共に経験的に求めた。非ホームポジションキーはホームポジションキーの位置を中心にその位置から一定の角度だけ回転した位置に配置される。以下に具体的な角度を述べる。中指、薬指によって入力されるキーについては 90（w、e、i、o）、270（x、c、コンマ、ピリオド）度である。人差し指については 0（g）、60（t、u）、120（r、y）、180（h）、240（v、n）、300（b、m）度である。小指については 90（q、p）、225（スラッシュ）、315（z）度である。親指については 0（全角半角切り替えキー、右矢印）、45（Delete、右 Shift）、90（左 Alt、上矢印）、135（左 Ctrl、Back Space）、180（Num、左矢印）、225（左 Shift、Alt）、270（Fn、下矢印）、315（Tab、右 Ctrl）度である。ここでユーザの各指によって入力されるキーをそれぞれグループにまとめたとすると、各グループ（例: q、a、z）において非ホームポジションキー（例: q、z）はホームポジションキー（例: a）の位置を基準として同一円周上に配置されることになる。よって、各指で入力されるキーのグループにおいてホームポジションキーから非ホームポジションキーまでの距離は全て等しくなる。一方、従来のソフトウェアキーボードではこの距離は等しくなかった。このように距離を等しくすることにより、指の移動量が抑えられるので非ホームポジションキーの入力はより容易になると考えられる。

3.4 親指周りへのキーの配置

ユーザが入力する際に、その指の位置がホームポジションキー周辺にとどまれば誤入力が減らせると筆者は考えた。そこで Leyboard では Space キー（物理キーボードにおける親指の担当キー）の他、後述するキーセット切り替え用のキーも含めて複数のキーを親指の周りに配置した。そのため Leyboard の配列は QWERTY 配列を元にしているものの、厳密には QWERTY 配列と異なったものになった。この配列により Leyboard の全てのキーはユーザのいずれかの指の位置あるいはその周囲に存在する。ゆえに、ユーザはいかなる入力においても、従来の QWERTY 配列のキーボードのときのように手を大きく動かす必要がない。

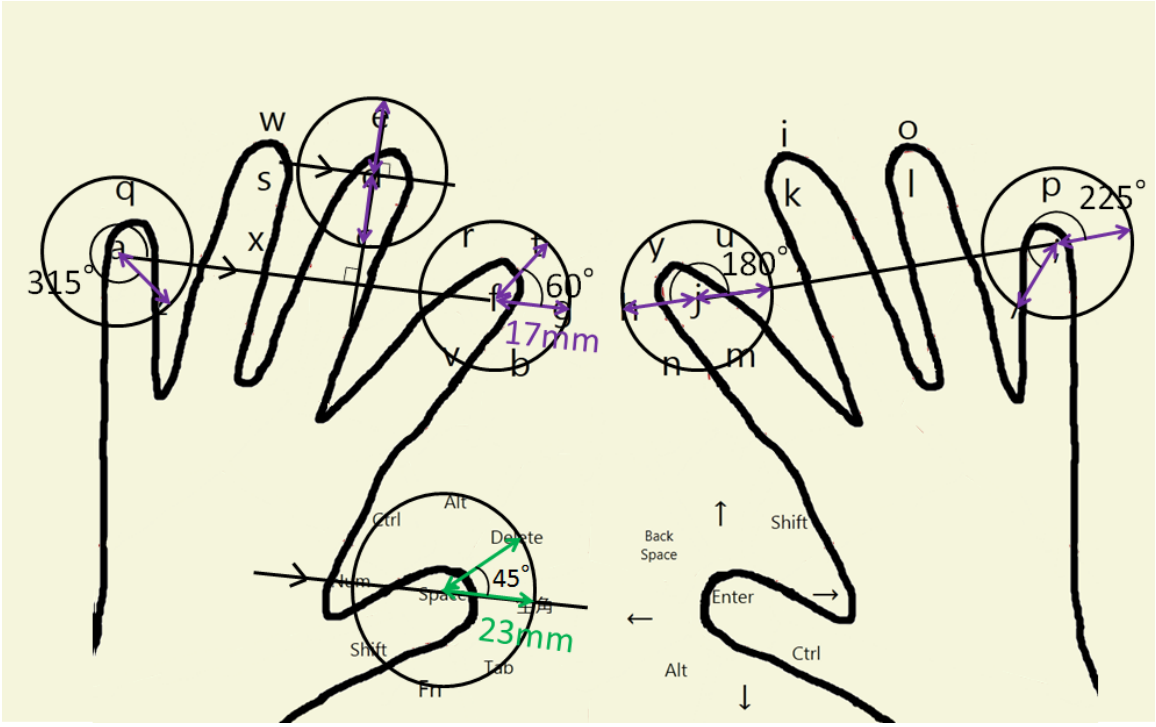

図 3.1: キーの配置規則

3.5 キーセット切り替え

テキスト入力に必要なキーの全てを一つのレイアウトとして配置すると、全てのキーがユーザの指の位置あるいはその周囲に配置するという Leyboard の提案要件を満たさなくなる。そこで筆者は Leyboard に 3 つのキーセットを用意し（図 3.2 から図 3.4）、それらを切り替えて用いるようにした。Leyboard ではユーザは入力中に必要に応じてキーセットを切り替えることになる。先述したように、筆者はキーセットを切り替えるキーを親指の周りに配置した。図 3.3 や図 3.4 上の円によって示されるキーセット切り替え用のキーをユーザが押している間、Leyboard のキーセットはそれぞれの図において示されるものに変化する。ユーザが指をキーセット切り替え用のキーから離したとき、キーセットは図 3.2 のアルファベットセットに戻る。これら 3 つのキーセットにより、Leyboard は合計 102 種類のキーが入力可能になっている。なお、キーセットを切り替えた状態においてさらに親指による入力を行うために親指スワイプ入力を設けている。

3.6 親指基点スライド

筆者は親指基点スライドと名付けた機能を実装した。一部の文字を入力する際に、2 つのキーを押す必要があることがある。例えば、f キーと Shift キーで大文字の ‘F’ を入力すると

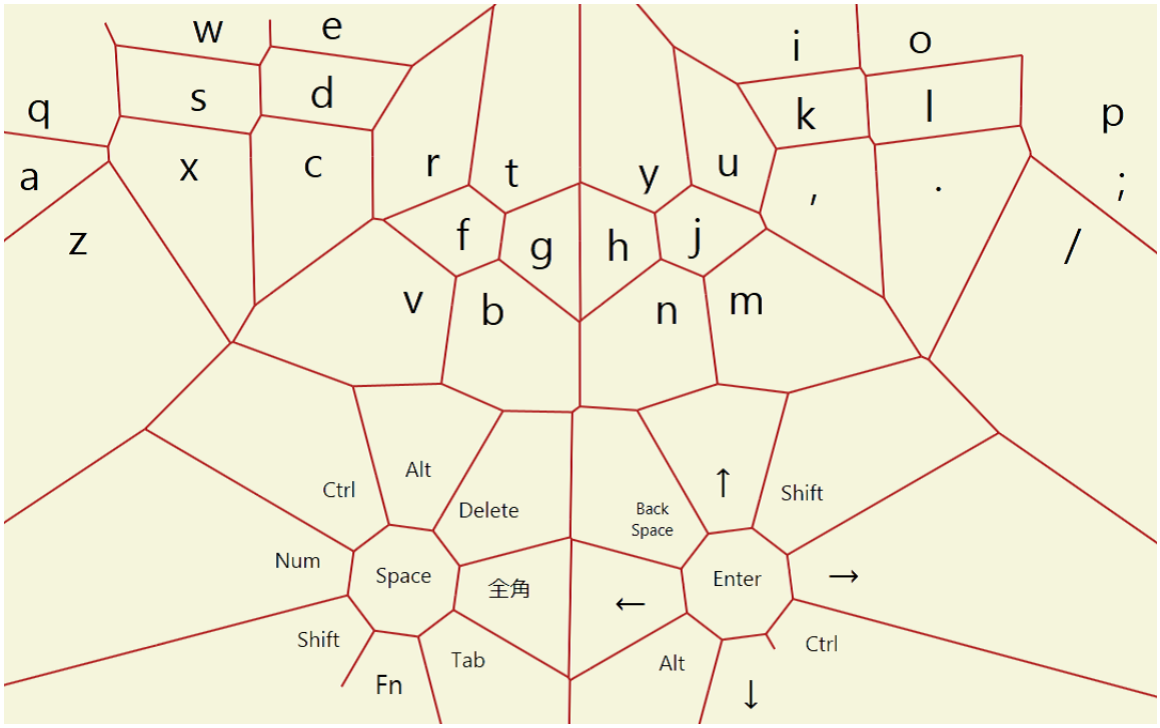

図 3.2: アルファベットセットの配列

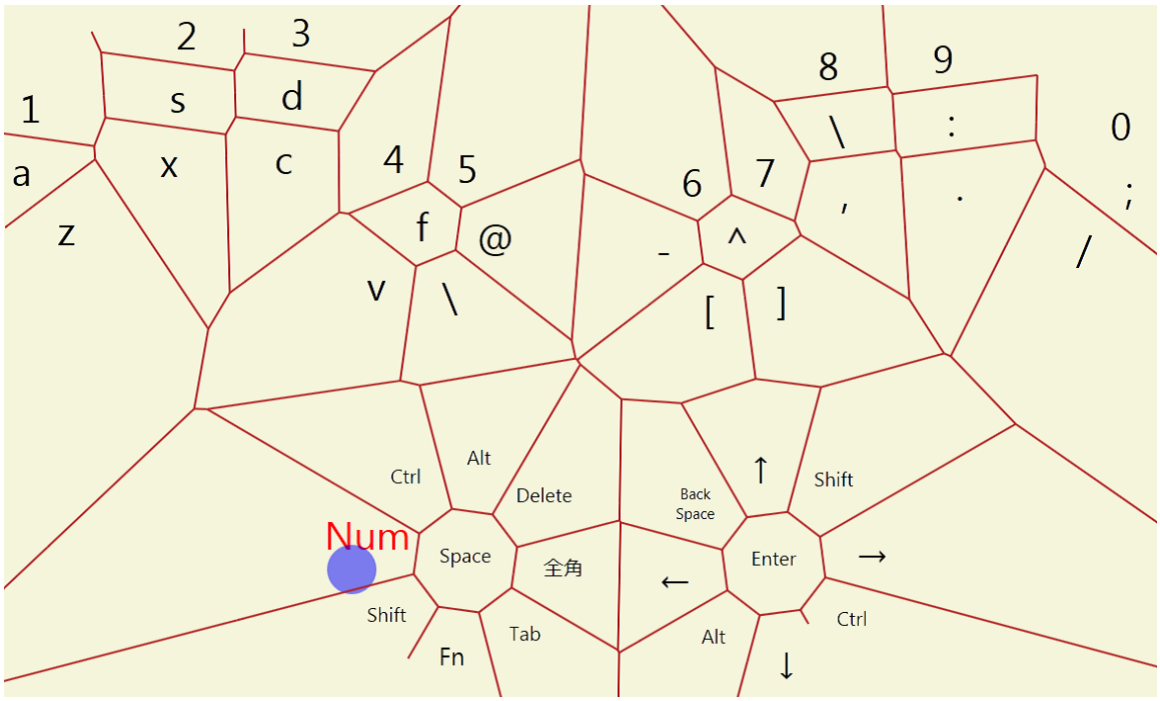

図 3.3: 数字・記号セットの配列

きである。親指基点スライドは Leyboard のレイアウトにおいてこのような 2 つのキーの同時入力を容易にする。図 3.5 に親指基点スライドの例を示す。本例では左手親指によって Shift キーの入力を行っている。このとき左手の人差し指から小指のキーの位置が、キャリブレーション時の親指との位置関係を保つように平行移動する。このように、親指基点スライドにより平行移動するキーは入力した親指と同じ手の人差し指から小指のキーである。この設計はユーザが 2 つのキーを同時入力する際に、手の姿勢を崩すことなしに入力することを可能にした。よってユーザはスムーズな入力であるがゆえに高速な入力が可能になると考えられる。親指基点スライドによって平行移動するキーの移動量と方向は、親指のホームポジションキー（Space、Enter）の位置座標からユーザの親指が現在押しているキーの位置座標までの距離と方向に等しい。親指基点スライドによるキーの平行移動はユーザが親指のキーが押している間続き、ユーザが指を離したときに平行移動していたキーは元の位置へ戻る。

3.7　親指スワイプ入力

キーセット切り替え用のキーは押している間しかキーセットを切り替えない。Shift キーはキーセット切り替え用のキーと同じ左手親指が担当するキーなので、キーセット切り替え用のキーと Shift キーを左手親指にて同時に入力することはそのままでは不可能である。キーセット切り替え用のキーと Shift キーの左手親指による同時入力を可能にするために、親指とその

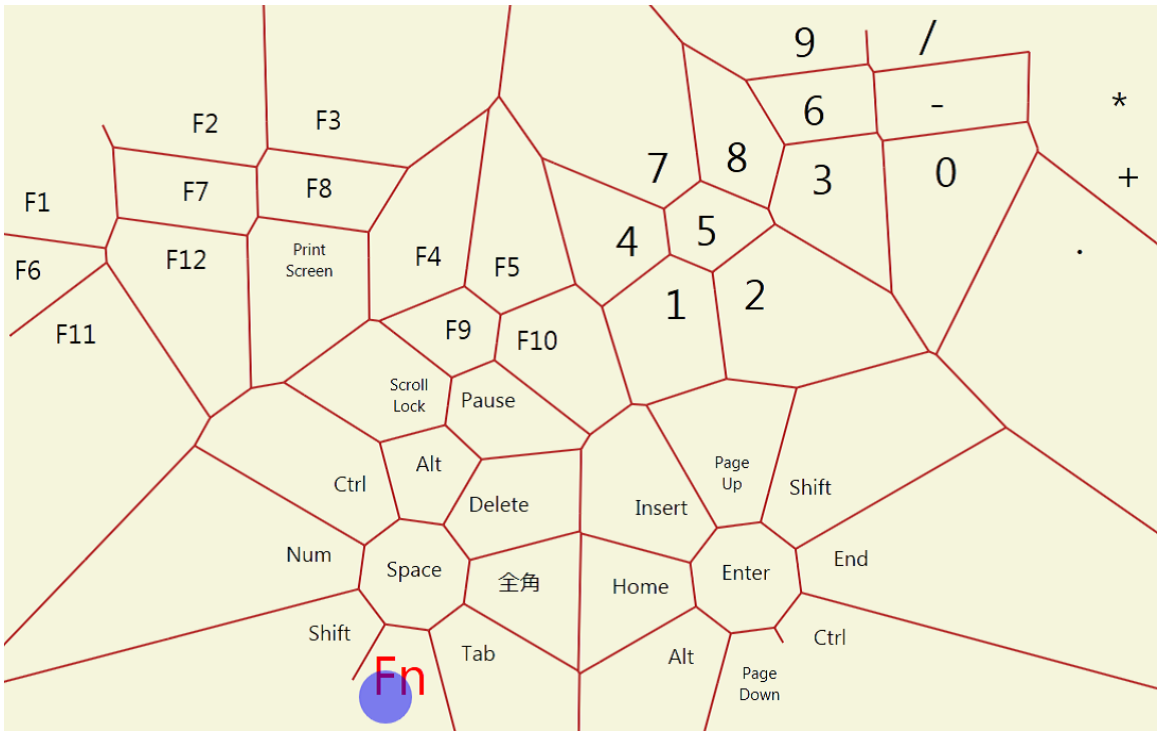

図 3.4: ファンクション・テンキーセットの配列

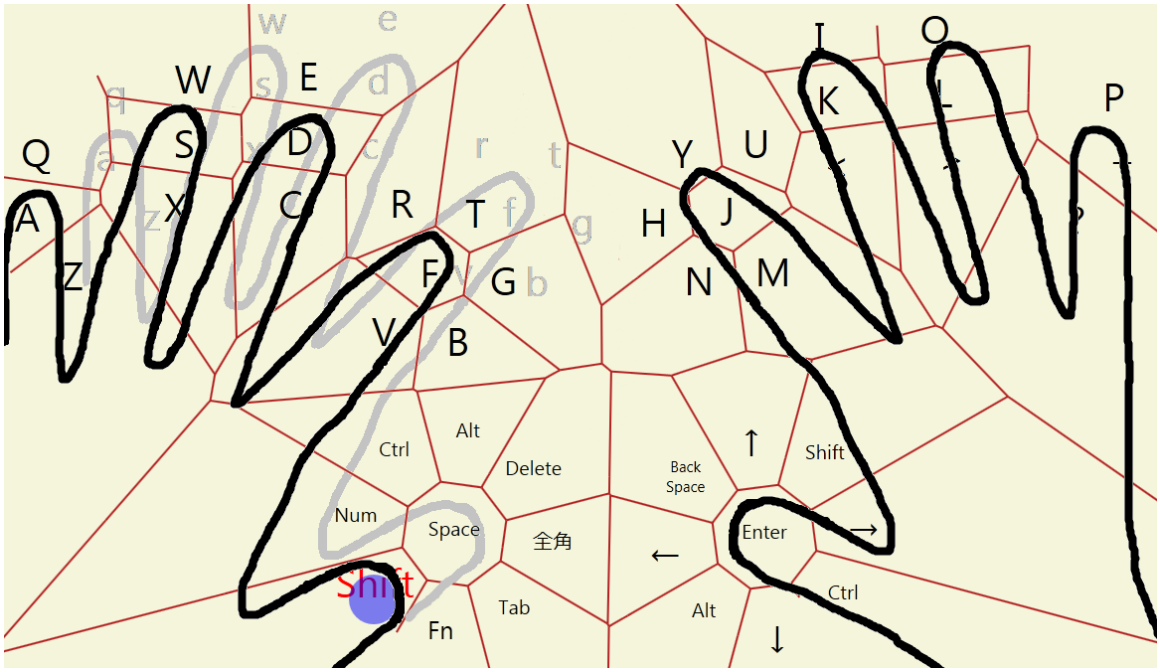

図 3.5: 親指基点スライドの例

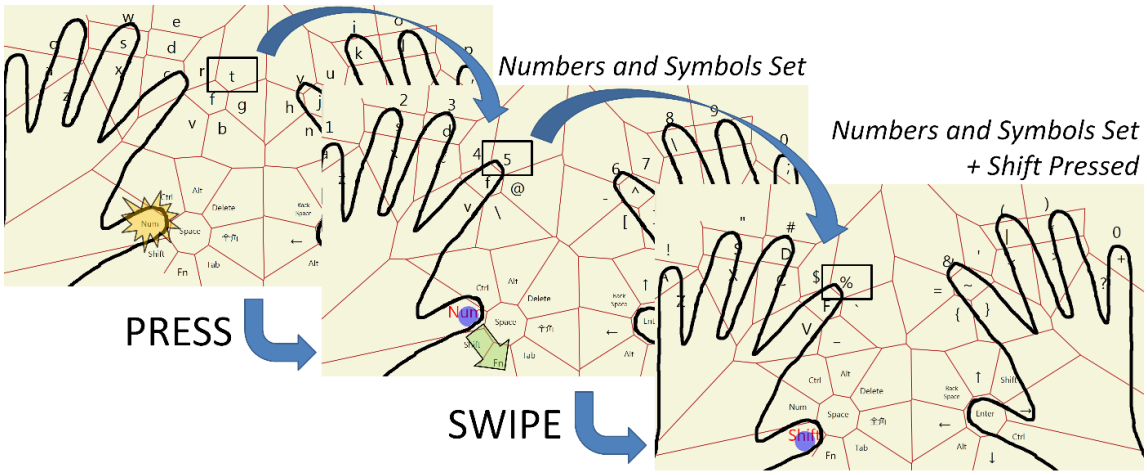

図 3.6: 親指基点スライドと親指スワイプ入力の連携例

周りのキーに、親指スワイプ入力を設けた。これは、親指をあるキーから別のキーへとスワイプさせることによってスワイプ先に存在するキーの入力を行うものである。親指スワイプ入力によって以下の 2 つのキー入力が可能になる。

- キーセットを数字・記号セットに切り替えるキー（Num）と Shift キーの同時入力
- キーセットをファンクション・テンキーセットに切り替えるキー（Fn）と Shift キーの同時入力

Shift キー以外の修飾キー（Ctrl、Alt）の入力に関してはこれらのキーを左手だけでなく右手親指にも配置しているのでキーセット切り替え時にも入力可能である。なお、左手親指にしか存在しないキーも存在する（Tab、半角全角切り替えキー、Delete）が、これらはキーセットを切り替えながらの入力が必要ではないため右手親指には設けられていない。また、本来想定された用途ではないが、片側親指のキーを複数同時に入力することも可能である。このとき、親指基点スライドは最後に入力されたキーの位置座標を基点として人差し指から小指のキーを平行移動させる。

3.8　親指基点スライドと親指スワイプ入力の連携

Leyboard は親指基点スライドと親指スワイプ入力を組み合わせることによって、ユーザの手の姿勢を崩すことなしに多種類の文字を入力可能にしている。図 3.6 に親指基点スライドと親指スワイプ入力の連携の例を示す。左手人差し指の右上の矩形内のキーに注目すると、左手親指をスワイプするにつれて、数字・記号セットに切り替わった状態、さらに Shift キーが押された状態とキーの表示が遷移している。このとき、キーの位置は常にユーザの指の位置に合ったものになっている。

3.9　フィードバック

Leyboard には入力時に視覚と音のフィードバックが存在する。

3.9.1　視覚フィードバック

従来のソフトウェアキーボードと同様に、Leyboard のキー入力時においてもユーザに視覚フィードバックを提示した。Leyboard ではユーザが画面に指を置いたとき、そのタッチ位置に青い円が表示され、入力されるキーの文字が赤くかつ大きく表示される。例えば図 3.3 ではキーセットを数字・記号セットに切り替えるキーを入力しているため、“Num”の文字が赤くかつ大きく表示されている。

3.9.2　フィードバック音

ユーザにキーの入力またはキーセットの切り替えが行われたことを通知するためにフィードバック音を設けた。ユーザがキーの入力またはキーセットの切り替えを行った場合、Leyboard のシステムはクリック音を発してユーザに入力が行われたことを伝える。

3.10　実装環境

Leyboard を開発する言語として C#を用いた。Leyboard は.NET Framework 4/WPF4 の API を利用して、マルチタッチ対応 WPF アプリケーションとして実装した。

第4章　Leyboardの被験者実験

Leyboard と既存の QWERTY 配列ソフトウェアキーボードの入力性能を比較する長期的な実験を行った。既存のものとして、筆者は Windows 7 に標準搭載されているソフトウェアキーボードを選択した。以降、これを Windows 7 キーボードとする。Windows 7 キーボードのアルファベットキーの形状は正方形になっており、1 辺の長さは 17mm であった。各ソフトウェアキーボードの幅（横幅）は共に 31cm であった。各ソフトウェアキーボードの高さ（縦幅）は Windows 7 キーボードが 10.5cm、Leyboard が 17.5cm であった。実験は約 1 年 4 か月という長期にわたって行われた。

4.1　実験環境

Leyboard を動作させるデバイスとして、Acer 社の ICONIA-F54E を用いた。ICONIA-F54E を使用した実験環境を図 4.1 に示す。ICONIA-F54E は液晶サイズが 14 インチ、解像度が 1366×768px（WXGA）の 10 点までのタッチ入力に対応するタッチパネルを搭載した端末である。ICONIA-F54E は 2 面のタッチパネルによって構成されるが、その下画面にて 10 本指のタッチを行った場合、ICONIA-F54E に用意された独自のソフトウェアキーボードが無条件に起動するようになっている。よって Leyboard を下画面にて使用することは不可能であった。そこで画面設定にて、それぞれの画面の表示を上下反転することによって上画面を見かけ上の下画面とし、そこにおいて Leyboard を動作させた。

4.2　被験者と実験タスク

本実験の被験者は筆者である。人間生活工学研究センター[1]による日本人の手の寸法データ集 2010 [2]にて用いられた形式に従った、被験者の手指の大きさを表 4.1 に示す。

筆者は本実験のタスクとして、英文パングラムの入力を選択した。パングラムとはアルファベットの全ての文字を少なくとも 1 回用いた文章である。例えば “A quick brown fox jumps over the lazy dog.”が挙げられる。このパングラムは必ず大文字を含み、また複数の記号を含むものもある。1 つのパングラムは 31 から 63 の文字によって構成されていた。パングラムの入力をタスクとするのは関連研究の CATKey において行われており、本研究ではこれに倣った。パングラムの入力をタスクとすることの利点は、各文章で全てのアルファベットが必ず 1 回

[1] `http://www.hql.jp/index.html`（参照 2014-01-19）
[2] `http://www.hql.jp/information/book/handdatabook.html`（参照 2014-01-19）


How quickly daft jumping zebras vex.
Good :14
Bad :1
Result:CharEnd
Time :122.524

図 4.1: 実験環境

表 4.1: 被験者の手指の大きさ

名称	左手（mm）	右手（mm）
手長１（茎突点）	186.4	181.8
手長２（屈曲線）	178.5	177.6
手掌長（茎突点）	108.8	108.1
指尖・指節点距離	101.6	101.9
第一指長	52.4	55.5
第二指長	70.1	68.9
第三指長	76.8	76.9
第四指長	72.7	73.6
第五指長	56.4	56.6
第二指近位関節－遠位関節長	20.7	21.2
第一指関節－指先長	29.3	28.9
第二指遠位関節－指先長	24.8	24.0
第一指爪基部長さ	14.6	14.0
第二指爪基部長さ	12.8	13.4
第三指爪基部長さ	13.4	13.4
第四指爪基部長さ	13.2	13.0
第五指爪基部長さ	11.7	11.4
茎状突起間幅	52.7	52.3
手幅（斜め）	78.3	76.4
第一指爪基部幅	15.7	15.3
第二指爪基部幅	14.6	14.3
第三指爪基部幅	15.0	14.8
第四指爪基部幅	13.8	13.8
第五指爪基部幅	12.2	12.1
第一－第五指尖端間最大距離	191.5	191.5

は入力されることである。そのため、全てのアルファベットにおいて少なくとも入力文章数と同じ数の出現が保証される。以降、10 の異なるパングラムを入力することを 1 セットとする。この 10 のパングラムは 120 用意したパングラムの中からランダムに選び出す形式によって、セットごとに決定された。

被験者は各ソフトウェアキーボードにおいて毎日 3 セットの入力を行った。本タスクでは、被験者は入力を間違えた場合、そこから正しい入力をやり直す必要がある。言い換えれば、被験者が正しい入力を行わない限りタスクは進まない。そのため被験者が入力を間違えた場合、それを通知するブザーが鳴るようにした。被験者はこの実験を 2012 年 2 月 15 日から 2013 年 6 月 13 日までの間、計 483 日間行った。これは各ソフトウェアキーボードあたり 1449 セット分に相当する。期間と実験日数が合わない理由は、期間中実験を行わなかった日が 2 日存在するためである。なお、最初の 7 日間は Windows 7 キーボード、Leyboard の順にタスクを行い、次の 7 日間はその順番を逆にするという行程を繰り返している。これは、カウンターバランス法に基づいて、ソフトウェアキーボードを使用する順番が実験結果に影響を与えるのを避けるためである。よって、順番こそ異なるが被験者は 1 日のうちに双方のソフトウェアキーボードを用いて入力を行うことになる。ソフトウェアキーボードの入力時に画面を見ることに関して、被験者に特に規定は設けなかった。結果的に、被験者は実験時に双方のソフトウェアキーボードにおいて、画面を見ながらの入力を同程度に行った。なお、Leyboard におけるボロノイ図は常に表示されていた。

毎日実験を行うにあたり、その場所や被験者の姿勢は毎回異なっている。実験を行う場所は被験者の自宅、被験者の所属する研究室内、出張先のビジネスホテルあるいは旅館であった。その中には机や椅子が存在しない場所もあり、その場合は床上や座卓等に ICONIA-F54E を設置して実験を行った。このとき被験者の姿勢は大きく変動した。日ごとに変動する実験の場所や被験者の姿勢に Leyboard のキー配置を適応させるために、毎回キャリブレーションを行う必要があると筆者は考えた。そのため、毎日の実験において、Leyboard のキャリブレーションは日ごとに行うこととした。被験者がキャリブレーションを行う際のタッチの行い方は毎回異なるため、キーの配置は日ごとに異なったものになる。

4.3 実験結果

4.3.1 入力率

筆者は入力率を 1 分間あたりに入力された単語数（wpm）として算出した。そのようにした理由は、被験者が各セットにおいて入力する文章内容が毎回異なるので、入力時間そのものは各ソフトウェアキーボードの性能を比較する基準にならないからである。wpm は Gentner によって定義された [Gen83]、1 分間あたりに入力された単語数を表す単位である。それは以下のように計算される。

$$\frac{1}{5}\frac{\text{入力誤りを除いたキーストローク数 (回/セット)}}{\text{入力の所要時間 (分/セット)}}$$

各ソフトウェアキーボードにおける入力率の変動及び対数近似に基づいた近似曲線を図 4.2 に示す。近似曲線を見ると、大きな差ではないが、Leyboard が既存のソフトウェアキーボードと比べて入力率に優れていることが分かる。各キーボードの入力率の最大値は Windows 7 キーボードが 59.34wpm、Leyboard が 61.27wpm となった。各キーボードの入力率の平均値は Windows 7 キーボードが 41.65wpm、Leyboard が 43.66wpm となった。各キーボードに十分に慣れた状態の数値として、2013 年 5 月 17 日から 2013 年 6 月 14 日までの 28 日分の各キーボードの入力率の平均値も求めた。その結果は Windows 7 キーボードが 47.09wpm、Leyboard が 49.69wpm となった。いずれの値においても Leyboard の入力率は Windows 7 キーボードよりも高くなった。

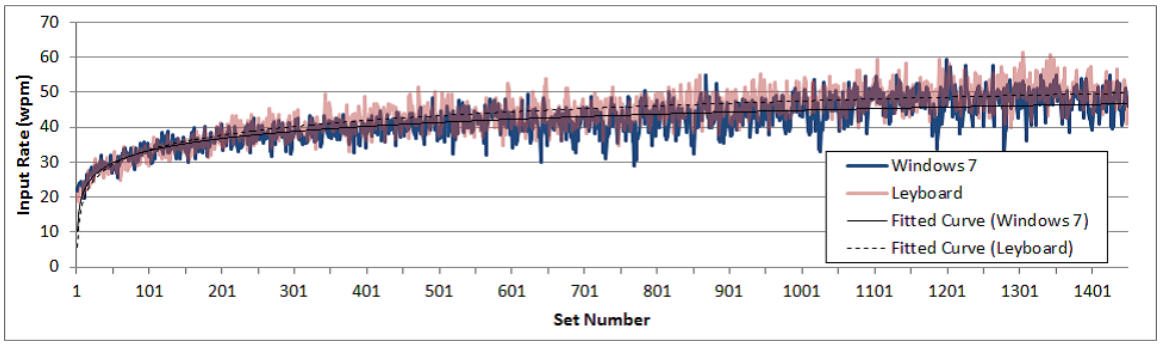

図 4.2: 各ソフトウェアキーボードの入力率（wpm）

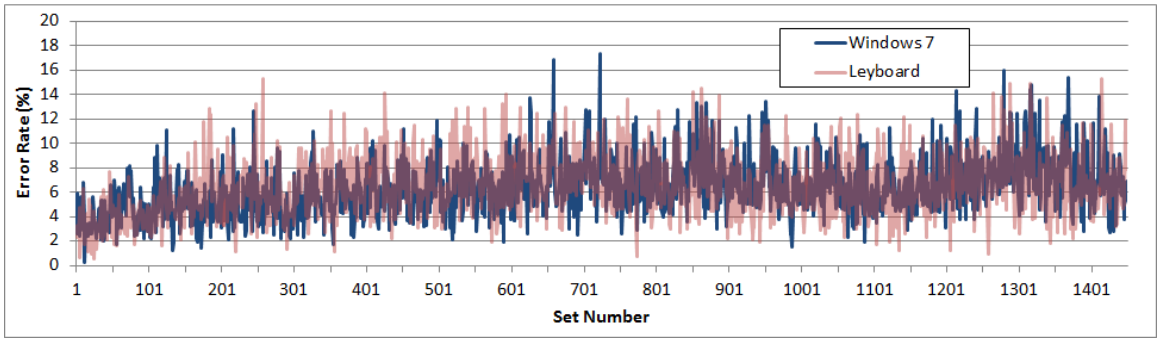

図 4.3: 各ソフトウェアキーボードのエラー率

4.3.2　エラー率

各ソフトウェアキーボードにおけるエラー率の変動を図 4.3 に示す。その平均値は Windows 7 キーボードが 6.36%、Leyboard が 6.54%であり、Leyboard の方が若干高い。

なお、被験者の物理キーボードにおける入力能力を調べるため、上記実験タスクを物理キー

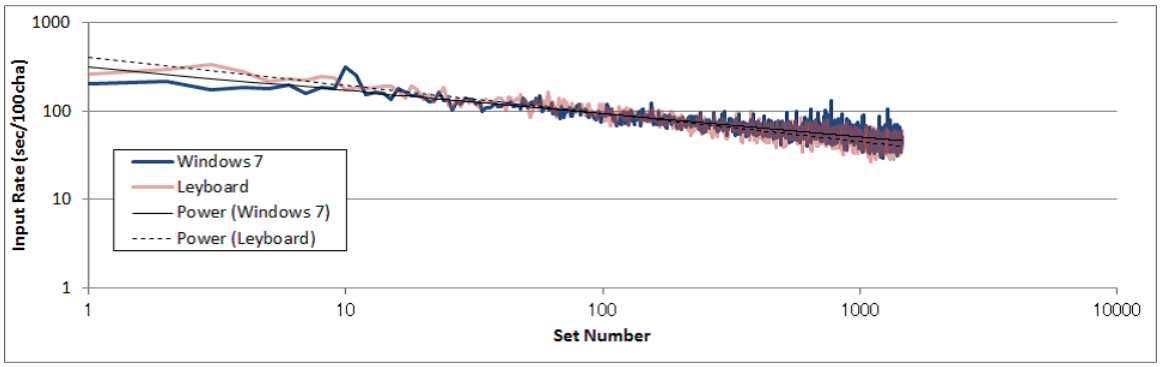

図 4.4: 各ソフトウェアキーボードの入力時間（100 文字あたりの入力時間）

ボードにて 9 セット行い、入力率とエラー率を求めた。その結果、入力率は 52.58wpm であり、エラー率は 5.98%であった。また、入力率の最大値は 55.63wpm であった。

4.3.3　入力速度の推移の妥当性

長期的に実験を行った場合、実験の値はべき乗の法則に従う。このとき、対数グラフにおいて累乗近似曲線を出力すると直線に近づくことが知られている [New81]。図 4.4 に 100 文字を入力するのにかかった時間を軸とした対数グラフを示す。累乗近似曲線が直線に近いため、本実験の経過がべき乗の法則に従っていることが分かる。よって、本実験における値の推移は妥当であると言える。また、今回の実験においては、1 週間おきに入力を行うキーボードの順番を入れ替えた。被験者は長期期間にわたり、毎日双方のキーボードの入力を行っていたため、キーボードを入力する順番を交代することによる違和感は特に抱かなかった。実際、交代が行われた日のデータを排除したうえで入力率の平均値を求めたところ、Windows 7 キーボードが 41.63wpm、Leyboard が 43.71wpm となり、全データの 41.65wpm、43.66wpm と比較して特に差はなかった。

第5章　Leyboardの被験者実験を受けての考察

本章ではLeyboardの被験者実験の結果を受けて、Leyboardが従来のソフトウェアキーボードに対して有用な点、及びLeyboardの問題点に関しての考察を行う。

5.1　キーボードごとの入力率及びエラー率の差

Leyboard と Windows 7 キーボード、それぞれのキーボードの入力率とエラー率の全データ（全1449セット分の入力率及びエラー率の数値の集合）に関して、それぞれ平均値に有意な差があるかを確かめるために、対応のあるt検定を行った。ユーザは実験が進むにつれてそれぞれのソフトウェアキーボードの入力に習熟していくので、キーボードの各セットに対応が存在する。検定の結果、Leyboardの入力率はWindows 7キーボードに対して有意に高いことが判明した（$t = -17.41, p = 2.2e-16$<0.01）。一方、エラー率の有意差はなかった（$t = -2.203, p = 0.02777$>0.01）。

上記の結果は実験全体のデータを比較したものである。十分に慣れた状態において、両者に有意差があるかを確かめるために、2013年5月17日から同年6月13日までの28日分のデータのみを抽出し、再度検定を行った。その結果、Leyboardの入力率はWindows 7キーボードに対して有意に高く（$t = -4.973, p = 3.506e-06$<0.01）、エラー率に有意差はなかった（$t = 0.7552, p = 0.4523$>0.01）。

5.2　エラー内容分析

次に、エラーの内容を分析した。筆者は実験途中の全てのキー入力内容に対してログを取った。そこで上記ログを参照し、以下のアルゴリズムに基づいて誤字と脱字の判定を行った。

1. 単語の判定は前方一致の形式で行う。
2. 入力と正解を文字単位によって比較し、正しい入力が行われている個所はスキップする。
3. 入力と正解が異なる場合、正解の次の文字に注目する。それが入力と一致した場合は脱字、それ以外は誤字と判定する。
4. 誤りが生じた場合、次に正しい入力が行われるまでの入力は全て無視する。これは脱字による入力のずれを誤りとしないためである。

例えば、テキストが “puppy”で、それに対する入力が “pupy”であった場合、このアルゴリズムは ‘p’ の脱字があったと見なす。別の例として、テキストが “lazy”で、それに対する入力が “kazy”であった場合は ‘l’ の誤字があったと見なす。

本実験では被験者は正しい文字を入力するまで次の入力を受け付けない仕様になっている。被験者が誤った入力を続ける限り入力は進まず、被験者は最終的には正しい文字を入力することになる。その結果、エラーのある入力は本来のテキストに特定の文字が挿入されたものになる。つまり、最初の例では、実際の入力は “pupypy”（pup（y）py）となり、 次の例では “klazy”（(k) lazy）となる。先のアルゴリズムに基づき、これらの例はそれぞれ 1 回の脱字、1 回の誤字と判定される。

Windows 7 キーボードには 23441 回、Leyboard には 23252 回のエラーが存在した。その内訳は以下のとおりである。Windows 7 キーボードには 17376 回の誤字と 6065 回の脱字が存在した。Leyboard には 14521 回の誤字と 8731 回の脱字が存在した。エラーの回数自体は Leyboard の方が若干少ないが、Leyboard は Windows 7 キーボードに比べて誤字が少なく脱字が多いように思われる。そこで、2 つのキーボードのエラー回数を文字別に分けてグラフ化した。誤字に関しては図 5.1 に、脱字に関しては図 5.2 に示す。なお、グラフからは誤り回数が両者ともに一定未満である文字（誤字は 50 回、脱字は 25 回）を除いた。ここで知りたいのはエラーが多くなる文字であったため、エラーの回数が非常に少ない（0.3%未満の比率である）上記の文字に関しては紙面の都合からグラフから排除した。グラフを見ると、エラー回数に極端な差がある文字が存在することが分かる。

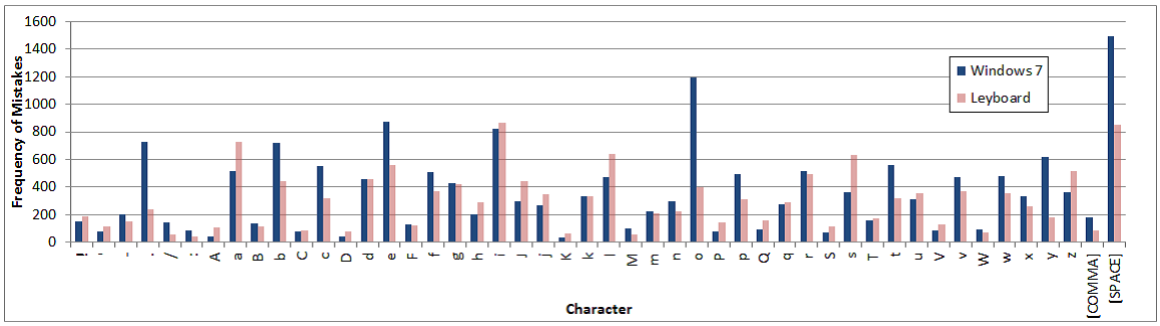

図 5.1: 誤字の分析結果

5.2.1 誤字内容の分析

Windows 7 キーボードと Leyboard の誤字内容に関して、多かったパターン上位 10 位をそれぞれ表 5.1 と表 5.2 に示す。

表 5.1 を見ると、正解と誤字の組み合わせは Space と n 以外全て横に隣り合ったキーによって構成されている。Windows 7 キーボードでは横に隣り合ったキーの誤入力が多いと言える。

表 5.2 を見ると、誤字のパターンを複数に分類できる。s と x、l と o、z と a、a と q、a と z は縦に隣り合ったキーである。J と j は Shift キーの押し損ねによって発生している。Space と

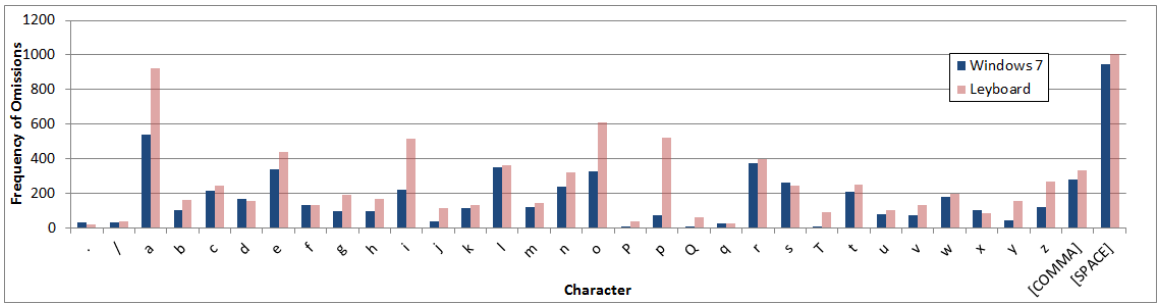

図 5.2: 脱字の分析結果

s は空白を入れるべきところに s を入力したことにより生じる。恐らく複数形にしなくてもよい単語を複数系にしたために生じた誤字である。i と o、u と y、v と b は横に隣り合ったキーである。よって Leyboard では Windows 7 キーボードより縦に隣り合ったキーの誤入力が多いと言える。そのため、縦に隣り合ったキーの誤入力が軽減できれば、Leyboard の入力率はさらに向上すると考えられる。

個々の比率や比率の累計を見ると Leyboard は Windows 7 キーボードより低いので、Leyboard の誤字の内容は Windows 7 キーボードと比べて偏りが小さい。

なお、Space と s の誤字に関して、Windows 7 キーボードでは 123 回生じていた。Leyboard の数が多い（177 回）傾向にあるものの、Space と s の誤字は双方のソフトウェアキーボードに見られる現象である。一方、J と j の誤字は Windows 7 キーボードでは 159 回生じており、Leyboard の 322 回とは大きく異なっていた。Leyboard の方が上記誤字を生じやすいと考えられる。その原因は、Windows 7 キーボードと Leyboard における Shift キーの位置が大きく異なるためと思われる。前者はアルファベットキーの外側にあり、後者は親指周りにある。そのため、Leyboard の方が j キーと Shift キーの距離が近い。距離が近いと手をほぼ動かさずに入力できるので、被験者は高速に入力しやすくなる。また、Shift キーと j キーは異なる手によって入力される。前者は左手親指であり、後者は右手人差し指である。入力する手が異なると、それぞれ別に入力できるので入力は高速になるが、入力する順番が逆になる可能性がある。入力する順番が逆になったとき、被験者は左手親指が Shift キーに触れる前に右手人差し指が j キーに触れている。あるいは、右手人差し指が j キーから離れる前に左手親指が Shift キーから離れている。この事態は入力を高速に行うことを試みたときに生じやすい。Shift キーと j キーの距離が近い Leyboard において、被験者が別々の手によって入力可能な Shift キーと j キーを高速に入力しようとした結果、入力する順番が逆になり誤字となったと考えられる。

5.2.2 脱字内容の分析

Windows 7 キーボードと Leyboard の脱字内容に関して、多かったパターン上位 10 位をそれぞれ表 5.3 と表 5.4 に示す。2 つのキーボードに共通して出現しているパターンは “ick”、“fro”、“for” のみである。個々の比率や比率の累計を見ると Leyboard は Windows 7 キーボードより

表 5.1: Windows 7 キーボード誤字内容上位

回数	正解	誤字内容	全体中の比率（%）	比率の累計（%）
823	[SPACE]	n	4.74	4.74
693	o	p	3.99	8.73
460	b	v	2.65	11.38
440	i	o	2.53	13.91
418	.	,	2.41	16.32
367	p	o	2.11	18.43
344	t	r	1.98	20.41
339	y	u	1.95	22.36
318	o	i	1.83	24.19
316	e	w	1.82	26.01

表 5.2: Leyboard 誤字内容上位

回数	正解	誤字内容	全体中の比率（%）	比率の累計（%）
510	i	o	3.51	3.51
322	J	j	2.22	5.73
318	s	x	2.19	7.92
298	l	o	2.05	9.97
241	z	a	1.66	11.63
230	a	q	1.58	13.21
186	a	z	1.28	14.49
177	[SPACE]	s	1.22	15.71
176	u	y	1.21	16.92
169	v	b	1.16	18.08

高い。よって Leyboard の脱字の内容は Windows 7 キーボードと比べて偏りが大きいほか、その特性が異なると言える。特に Leyboard の脱字における “uiz”や “mph”の比率は、Windows 7 キーボードの脱字内容上位の比率と比べても非常に高い。

表 5.3: Windows 7 キーボード脱字内容上位

回数	1 文字前	脱字内容	1 文字後	全体中の比率（%）	比率の累計（%）
60	[SPACE]	l	a	0.99	0.99
57	i	c	k	0.94	1.93
56	f	r	o	0.92	2.85
56	h	,	[SPACE]	0.92	3.77
51	t	c	h	0.84	4.61
47	f	o	r	0.77	5.38
46	e	r	[SPACE]	0.76	6.14
44	e	[SPACE]	m	0.73	6.87
42	e	d	[SPACE]	0.69	7.56
41	a	c	k	0.68	8.24

表 5.4: Leyboard 脱字内容上位

回数	1 文字前	脱字内容	1 文字後	全体中の比率（%）	比率の累計（%）
199	u	i	z	2.28	2.28
121	m	p	h	1.39	3.67
87	f	o	r	1	4.67
85	e	n	[SPACE]	0.97	5.64
84	u	i	c	0.96	6.6
71	[SPACE]	T	V	0.81	7.41
70	f	r	o	0.8	8.21
70	[SPACE]	o	f	0.8	9.01
65	i	c	k	0.74	9.75
59	n	g	[SPACE]	0.68	10.43

Leyboard において脱字が多くなった要因として、タップによって入力を行うという仕様上の問題が考えられる。ソフトウェアキーボードにおいてキーが入力される瞬間はソフトウェアキーボードごとに異なるが、それらを大きく分けると、ユーザが指を画面に置いたときに入力を行うものと指を画面から離したときに入力を行うものの 2 種類に大別することができる。Leyboard はユーザがタップ入力を行ったときのみにキー入力を実行するため、後者に分類される。この場合、キーの入力順番はユーザが指を画面から離す時刻のみによって決定さ

れる。物理キーボードにおいてはユーザがボタンを押した瞬間にキーが入力される。そのため、ソフトウェアキーボードにおいては、ユーザが指を画面上のキーに置いた順にキーが入力されることが自然に感じられる。しかし、ユーザが画面に指を置く順番とユーザが画面から指を離す順番は同じであるとは限らない。ゆえに次のような問題が発生する。ユーザが例えばu、iの順に指を置いても、指をi、uの順番に離していたならば、入力はi、uになる。するとユーザには入力が入れ替わったように感じられる。これを見かけ上の入力入れ替わりと呼ぶことにする。この実験ではエラー分析の際に、入力と正解が異なる場合において、正解の次の文字が入力と一致した場合を脱字とみなしている。よって先に挙げた、ユーザが画面に指を置いた順番と離した順番が異なることによって発生する、見かけ上の入力入れ替わりも脱字と解釈される。Leyboardにおいて脱字が多くなった要因は、この見かけ上の入力入れ替わりが多数発生したためと考えられる。

Leyboardにおいては見かけ上の入力入れ替わりが発生し得る。Leyboardと同様に指を画面から離したときに入力を行うソフトウェアキーボードには、見かけ上の入力入れ替わりを回避するために、ユーザが画面に指を置いた順番を記憶しているものがある。この種のソフトウェアキーボードはユーザが画面に指を置いたときにその順番を記憶している。そして、画面からユーザの指が離れた際に、ユーザが画面に指を置いた順番に沿って、画面から離れた指の分までの入力をまとめて行っている。するとキーの入力順番はユーザが画面に指を置いた順番によって決定されることになるため、見かけ上の入力入れ替わりは発生しない。例えばWindows 8に標準搭載されているソフトウェアキーボードはこのような仕様になっている。

Leyboardで見かけ上の入力入れ替わりが発生するのは、上記の仕様を取り入れなかったからである。取り入れなかった理由は、Leyboardではキー入力以外にキャリブレーションが求められるためである。上記の仕様は前提として、ユーザが画面に指を置く場合はキー入力時に限られるという条件を持たなければならない。つまり、ユーザの画面に指を置く行為がキー入力を意図したものであると保証することによって、ユーザが画面に指を置いた順番に沿ってまとめて入力を行うことがソフトウェアキーボードにとって可能になる。これが保証されない場合、ユーザがキー入力を意図せずに画面においた指がキー入力を誘発し、結果として誤入力になる。Leyboardにおいてはユーザが画面においた指がタップによるキー入力途中なのか、キャリブレーションを意図したものなのかがユーザが画面に指を置いたその時点では確定しない。それが確定するのは、画面からユーザの指が離れ、タップ入力と解釈される、あるいはユーザが指を置いたまま時間が経過し、タップ入力である可能性がなくなるときである。そのためユーザが画面に指を置く場合はキー入力時に限られるという前提が成立しないので、Leyboardでは上記の仕様を取り入れていない。

5.3 実験結果のまとめと手元注視を減らすための要件分析

5.3.1 実験結果のまとめ

Leyboardは十分に習熟した状態において従来のソフトウェアキーボードよりも高い入力率にて入力を行えることが判明した。よって、キーの位置と大きさをユーザに合わせるという本

研究のアプローチはソフトウェアキーボードの入力率を向上させると言える。一方、Leyboard では従来のソフトウェアキーボードと同程度の手元注視を必要とした。

5.3.2 手元注視を減らすための要件分析

Leyboard の手元注視を減らせなかった理由から、次に実装するキーボードのプロトタイプの仕様や課題を定めた。

キーの数を減らす

Leyboard の手元注視を減らせなかった理由の 1 つとして、1 つの指が担当するキーの数が多いことが考えられる。Leyboard においては表 5.5 に示すキーと指の対応において入力が行われることを想定している。これは物理キーボードと等しく、1 つの指の担当するキーの数は最小 3 つ最大 6 つである。さらに表 5.5 からは省略しているが親指が担当するキーの数は 9 つであり、非常に多い。指が担当するキーが多くなるほど、それらのキーの入力に合わせて指の位置を使い分ける必要がある。例えば担当するキーが 3 つである中指の場合、上段のキーを入力する際の指の位置、中段のキーを入力する際の指の位置、下段のキーを入力する際の指の位置の 3 つの指の位置が存在し、これらを使い分けなければならない。しかし物理的なフィードバックに乏しく、キーの位置に指を合わせにくいソフトウェアキーボードでは指の位置の使い分けが困難である。Leyboard には既存のソフトウェアキーボードと同様に物理的なフィードバック（キーを触って感知すること）は存在せず、フィードバックとして視覚フィードバックとフィードバック音が存在した。前者は手元注視を行わない場合においては効果に乏しく、後者はキーの位置に応じて異なるフィードバック（音）を返すのではなく、常に同じ音を鳴らすためユーザがキーの位置を把握する用途には有用ではない。以上の理由から、Leyboard のフィードバックは指の動かし方を使い分けることに貢献しているとは考えにくい。そのため物理的なフィードバックを設けないのであれば、使い分けるべき指の位置の数を Leyboard よりも少なくするべきである。これは 1 つのキーセットにおけるキーの数を減らすことに等しい。以上より、ソフトウェアキーボードにおいて手元注視を減らすためには、キーの数を減らすことが有効であると考えた。

表 5.5: Leyboard におけるキーと指の対応（親指は多数につき省略）

左手				右手			
小指	薬指	中指	人差し指	人差し指	中指	薬指	小指
q	w	e	r、t	y、u	i	o	p
a	s	d	f、g	h、j	k	l	;
z	x	c	v、b	n、m	,	.	/

Leyboard において誤字の原因の一つは縦に隣り合ったキーの誤入力であった。この誤入力は縦に隣り合ったキーの誤入力は中段のキーと間違えて上下段のキーを入力する、あるいは上下段のキーと間違えて中段のキーを入力することにより生じる。よって、図 5.3 のように中段のキーを除けばこの誤入力は生じ得ないと考えられる。そこで次に実装するキーボードは中段のキーを除く。するとキーセットにおけるキーの数が減るので、使い分けるべき指の位置の数が減り、ユーザの手元注視の回数を減らせると考えられる。この機能の課題は、排除した中段のキーの入力方法を決めなければならないことである。中段のキーの入力方法は上下段の入力と区別するために、上下段の入力とは異なっていなければならない。

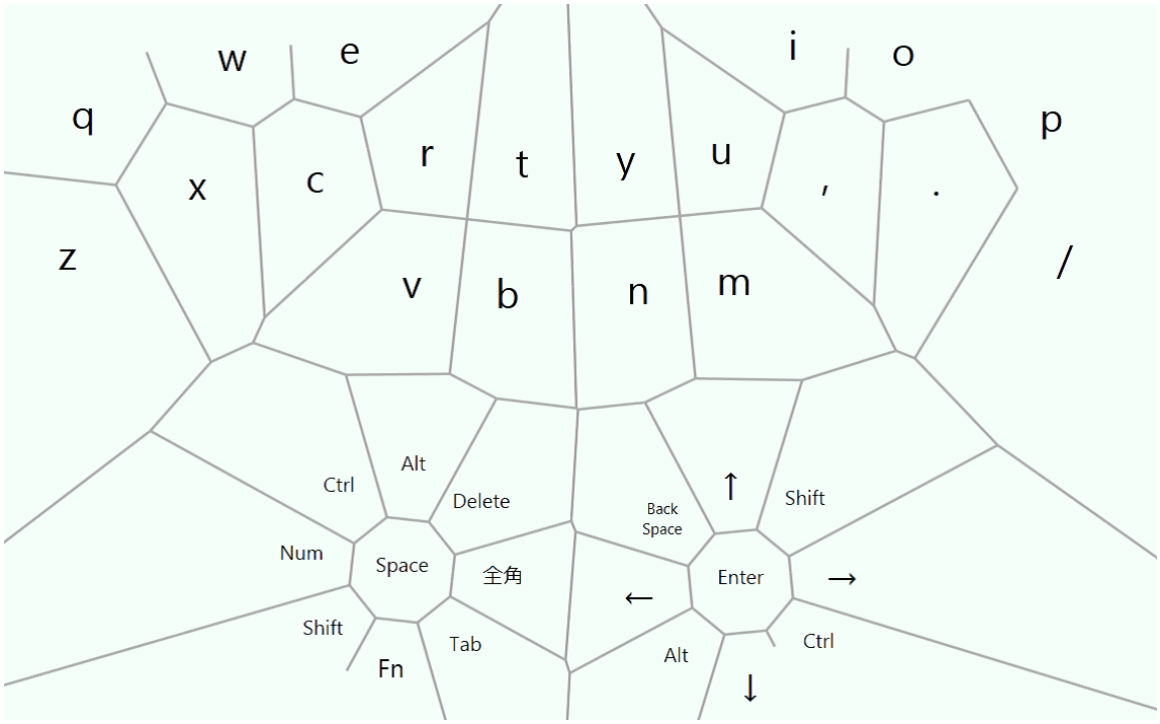

図 5.3: 中段のキーを排除した配列

不要なキーを除外する

キーの数を減らすために、不要なキーを除外することも考える。Leyboard には文字入力において全く用いられない（ゆえに実験においても全く用いられなかった）キーが含まれていた。例えば、Fn・テンキーセットに用意されたキーや、Ctrl、Alt のような修飾キーの一部がこれに該当する。これは Leyboard が物理キーボードで入力可能なキーを入力可能であるようにしたためである。しかし、ソフトウェアキーボードの既存製品は文字入力のみを前提としていると考えて良いと思われる。言い換えると、既存製品では物理キーボードを忠実に模したものを除けば、先に挙げたキーは入力不能であるものが多い。例えば Leyboard の実験に用

いた Windows 7 キーボード（図 5.4）では物理キーボードのキー配置が忠実に再現されていたが、Windows 8 ではよりシンプル化したレイアウトのソフトウェアキーボード（図 5.5、以降 Windows 8 キーボードと呼ぶ）が採用された。そこで次に実装するキーボードも文字入力のみを前提とし、それに不要なキーを除く。

図 5.4: Windows 7 キーボード

図 5.5: Windows 8 キーボード

手のひらを固定できるようにする

Leyboard ではユーザの手の一部が画面に触れると誤入力を誘発する可能性があるため、ユーザは入力に用いる指のみを適宜画面に置き、手自体は画面上を浮かせながらの入力を行った。そのため、手の位置を固定する方法がなく、入力が進むにつれて手の位置が変わりやすくなっていた。これはユーザに手の位置を調整するための画面注視を増やす要因になったと考えられる。そこで次に実装するキーボードは、手のひらの下 1/2 程の範囲を机上など（端末の画面外になる）に置き、その位置を固定しながら入力を行うことができるようにする。このとき端末は横倒しの状態にて置かれていることを前提とする。

指を置きながらの入力を可能にする

Leyboard はユーザが 10 本の指を画面に置いたときにキャリブレーションを必ず行う仕様であった。そこでキャリブレーションと入力が衝突することを避けるために、入力時に同時に画面上における指の数は最大 8 本として物理キーボードのように全ての指を置きながら入力を行うことはできなかった。物理キーボードのように指を置きながらの入力を可能にすると、手の位置が安定するのでユーザは画面注視の回数を減らせると考えられる。そこで次に実装するキーボードは、画面上に指を置きながら入力が可能なようにしたい。しかし Leyboard のように、キーボードが想定する最大本数の指をユーザが画面に置いたときにキャリブレーションを行うとすると、入力時に不要なキャリブレーションが発生する可能性がある。これを回避することが次に実装するキーボードの課題である。そのために、ユーザがキャリブレーションに必要な本数の指を置いたときに直ちにキャリブレーションを行うのではなく、ユーザが指を置いてからある程度の時間経過の後にキャリブレーションを行う。このようにして入力とキャリブレーションが衝突しないようにする。

第6章　Meyboard

Leyboard の実験結果を受けて、ユーザの手元注視を減らすソフトウェアキーボードとして Meyboard の実装を行った。

図 6.1 から図 6.4 に Meyboard の各種キーセットを示す。その配列は QWERTY 配列を基にキーを格子状に並べ、さらに中段のキーを排除したものになっている。r、v、u、m のキーは、その位置がユーザの左右それぞれの人差し指が来るべきところであることを示すガイドとなるようにキーの色を変更した。物理キーボードにおいては f、j キーに設けた突起を人差し指のガイドにすることによってユーザが指をキーに合わせやすいようにしている。r、v、u、m のキーの色変更はこれに沿っている。基本的に各指が担当するキーの個数は上下段それぞれのキーの 2 つであるが、人差し指は例外的に 4 つ（物理キーボードのおいては 6 つ）となる。そのため、色変更はそれら 4 つのキーの位置や境界を分かりやすくする効果もある。

図 6.1: 半角アルファベットセットの配列

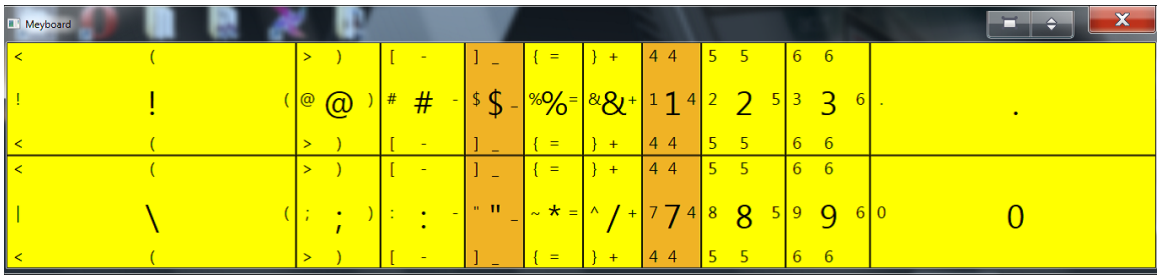

図 6.2: 半角数字・記号セットの配列

Meyboard の入力はユーザの左右の人差し指から小指の合計 8 本の範囲内にて行われる。ユーザが 10 本全ての指を使うことを想定すると、相応の面積がソフトウェアキーボードには求め

図 6.3: 全角アルファベットセットの配列

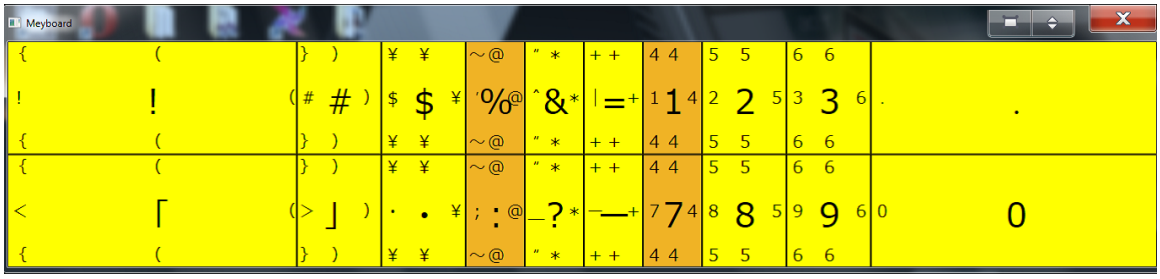

図 6.4: 全角数字・記号セットの配列

られることになる。このようなソフトウェアキーボードは画面が 1 つしか存在しない端末においては画面の大部分を自身が占有し、入力対象を表示する空間が非常に小さくなる、あるいはなくなる問題が存在する。Leyboard も ICONIA-F54E の 2 つある画面のうちの 1 つの領域を全て占有していた。但し Leyboard の被験者実験では、画面が 2 つ存在する ICONIA-F54E を用いたため問題とはならなかった。タッチパネルを搭載する端末は画面を 2 つ以上持つとは限らないので、現実に即した設計にするならばキーボードは画面を全て占有してはならない。そこで Meyboard は親指を使わない設計にして、画面を占有する領域を小さくした。その結果、Meyboard は Leyboard に比べてその高さが約 0.3 倍と小さくなった。なお、Meyboard の幅は Leyboard と違いはなく、画面の幅に等しい。

Meyboard の画面を占有する領域は Leyboard に比べて小さくなった。そのため、画面上における Meyboard の表示位置を画面下部とすることにより、手のひらを机に置き、手の位置を固定しながらの入力が可能になった。

Meyboard の各キーは格子状に配置されている。これは段ごとにキーの位置がずれている物理キーボードとは異なるが、それは以下の理由による。1 つ目は、キーの位置や形を単純化することによりユーザに分かりやすい見た目にすることである。2 つ目は、ユーザが入力に用いた指を判定する機能やキャリブレーションの処理を容易にするためである。これはキーを格子状に並べる場合、キーの境界となる直線が座標軸と平行になるからである。3 つ目は、段ごとのキーの位置を物理キーボードのようにずらすと、キャリブレーション時のユーザの指の位置から b、y キーの位置までの距離が遠くなるからである。このとき、ユーザの指はこれ

らのキーに届きにくくなり、誤入力の可能性が高くなる。また、手元注視を減らすためには指の移動量を抑えることが望ましい。何故ならば指の移動量が大きくなると、キャリブレーション時の指の位置へユーザが指を戻すことが困難になるのでキャリブレーションの効果が薄くなるからである。格子状のキー配列によって b、y キーを入力する際の指の移動量が抑えられる。

6.1 タップ入力

Meyboard ではタップ入力が上段や下段のキーの入力として用いられる。ユーザが上段や下段のキーをタップした場合に、それらのキーの入力が行われる。タップ入力には制限時間が設けられている。これをタップリミットと呼ぶことにする。タップリミットはキー入力と指の待機状態を区別するために設けた。タップ入力では、ユーザの指が画面に置かれた時刻から、タップリミットが経過するまでに指が画面から離れた場合を入力が行われたと解釈する。タップリミットは 450 ミリ秒に設定した。この値は、TapBoard においてタイピングと待機状態のトレードオフが 450 ミリ秒と結論付けられていたのでそれに従った。

6.2 フリック入力

Meyboard ではフリック入力が中段のキーの入力として用いられる。ユーザが上段か下段のいずれかのキーにおいてフリック入力を行った場合に、中段のキーが入力される。

フリック入力にも制限時間が設けられている。これをフリックリミットと呼ぶことにする。フリックリミットもキー入力と指の待機状態を区別するために設けた。フリック入力では、ユーザの指が閾値（第 7 章の実験においては 3.4mm であった）以上移動した時刻から、フリックリミットが経過するまでに指が画面から離れた場合を入力が行われたと解釈する。フリックリミットは 500 ミリ秒に設定した。フリック入力は指の移動があるため、フリックリミットはタップリミットよりも若干長い値に設定した。Meyboard のフリック入力では、ユーザが画面に指を置いた時刻を考慮していない。これはフリック入力では指の移動量を閾値とすることによって入力と待機状態を区別可能であるので、ユーザが画面に指を置いた時刻を条件に組み込む必要がないためである。

物理キーボードにおいては、ユーザはキーに指を置いた状態（待機状態）からの入力が可能である。Meyboard でもユーザの手元注視を減らすために待機状態からの入力を可能にした。待機状態からの入力を可能にすると、手の位置が安定するため、手元注視を減らすことができると考えた。中段のキーの入力にフリック入力を用いるのは待機状態からの入力を実現するための手段でもある。

待機状態から入力が可能なソフトウェアキーボードの関連研究として、TapBoard と Liquid-Keyboard にて提案されていた発展案がある。TapBoard では中段のキーを入力する際に指を一旦画面から離してからタップ入力を行う必要がある。これは待機状態と入力をユーザが画面に指を置いている時間から区別しているためである。待機状態から指を画面から離す必要が

ある点は物理キーボードと異なっている。Meyboard では物理キーボードの待機状態からユーザが中段のキーを押し込むという動作を、待機状態からフリック動作を行うことに置き換えている。そのため、TapBoard のように画面から一旦指を離してから入力を行う必要がない。LiquidKeyboard ではユーザの指が画面に置かれているとして、そのタッチ面積から待機状態と入力を区別することを、彼らの研究の応用として提案していた。これは TapBoard のように入力時に画面からいったん指を離す必要はない。しかし、待機状態と入力を区別するタッチ面積の閾値を設ける必要がある。筆者は、ユーザが画面に置いた指のタッチ面積を把握するのは困難であると考えた。そのため、Meyboard ではタッチ面積から入力内容を区別することはしなかった。

フリックする方向は概ね任意の方向であるが、左方向のみ後述する Shift 入力と解釈されるため除外する。この方向にフリック入力を行った場合は Shift 入力とみなされる。

6.3 Shift 入力

Meyboard では Shift 入力が上段や下段のキーの大文字の入力として用いられる。Shift 入力とは左方向へのフリック入力である。厳密には以下のとおりである。フリック入力において、ユーザが画面に指を置いたときの指の位置から、ユーザの指の移動量が閾値を超えた瞬間における指の位置を結ぶベクトルと、画面の幅方向に平行な、右端方向へのベクトル（大きさは任意）とのなす角を θ とする。θ の値が $3/4\pi$ から π、あるいは $-\pi$ から $-3/4\pi$ のときを左方向へのフリック入力、すなわち Shift 入力と解釈する。θ がこの範囲でなかった場合は左方向ではないフリック入力と解釈され、中段のキーが入力される。以降、本論文において特に断りがなくフリック入力とのみ記述されている場合、それは中段のキーの入力を意図した、左方向以外へのフリック入力を意味する。

左方向へのフリック入力を Shift 入力と命名したのは、大文字入力時に使用される Shift キーが、物理キーボードにおいてアルファベットキーの左側に存在することのメタファである。なお、便宜上左フリックによって行う入力を Shift 入力と呼んでいるが、Shift 入力の全てが物理キーボードにおける Shift キーを伴った入力ではない。

6.4 二段フリック入力

Meyboard では中段のキーの大文字の入力に二段フリック入力を用いる。ある方向へ閾値以上に指を動かした後に、そのまま画面から指を離さずに別方向へ指を閾値以上に動かしてから指を画面から離すと二段フリック入力となる。これらの閾値はフリック入力や Shift 入力と等しい（第 7 章の実験においては 3.4mm）。二段フリック入力となる方向の組み合わせは大きく分けて 2 つ存在する。1 つは 1 回目の指の移動方向が任意（但し左方向除く）であり、2 回目の指の移動方向が左方向の場合である。もう 1 つは 1 回目の指の移動方向が左方向であり、2 回目の指の移動方向が任意（但し左方向除く）の場合である。よって、二段フリック入力の方向の組み合わせはフリック入力と Shift 入力の方向の組みあわせになっている。

Shift 入力のように、特定の方向にフリックすることによって中段のキーの大文字の入力を行う仕様にしなかったのは、ユーザの学習コストを考慮したためである。Meyboard において QWERTY 配列をベースにしているのは、ユーザの習得を容易にするためである。そのため、ユーザが覚えるべきことはなるべく少なくすることが妥当である。ユーザが覚えるべきことが多いと学習コストが高くなる。Meyboard では中段のキーの入力がフリック入力によって行われ、上下段のキーにおける大文字の入力が Shift 入力によって行われる。これはユーザに覚えてもらう必要がある。ここで、中段のキーの大文字の入力として、新たなフリック方向を覚えてもらうことと、フリック入力と Shift 入力の方向を組み合わせた二段フリック入力を覚えてもらうことを考える。後者は中段のキーの入力と大文字の入力の組み合わせが中段のキーの大文字の入力になることを意味する。そのため、後者のほうがユーザに取って理解しやすく覚えやすいと考えた。

6.5 複数の指による入力

Meyboard では複数の指を同時に用いることにより、入力を行うキーが存在する。Meyboard にて複数の指を同時に用いる入力を取り入れたのは、Meyboard のキーの数の少なさのためである。Meyboard は中段のキーがない 20 個のキー群によって構成される 2 つのキーセットを用意している。そのため、キーは高々40 個しか存在しない。入力できるキーを増やすためにフリック入力を加えているが、単一の指による入力のみで全てのキー入力を賄うのは困難である。また、そのようなキー配列は QWERTY 配列から著しく離れたものになり、ユーザの学習コストが高くなる問題がある。加えて、Space や Enter のようなアルファベットでも数字でも記号でもないが、使用頻度が非常に高いキーは常に入力できるようにする必要がある。このようなキーがいずれかのキーセットにのみにしか存在しない場合、入力において要求されるキーセット切り替えの回数が著しく増大し、不便である。そこでいずれのキーセットにおいても入力が可能であり、かつその入力が 20 個のキーのいずれかを占有することがない、複数の指を同時に用いる入力を取り入れた。

複数の指を同時に用いるキーの入力方法を表 6.1 に示す。表 6.1 では入力に用いる指と用いない指をそれぞれ黒丸と白丸によって示している。左手の 4 つの丸は左から小指、薬指、中指、人差し指であり、右手の 4 つの丸は左から人差し指、中指、薬指、小指である。入力時に求められるユーザの指の動作は、タップ、フリック、プレス&ホールドの 3 種類である。入力内容のうち、キーセット切り替え、左手キャリブレーション及び右手キャリブレーションに関しては後述する。

これら複数の指による入力パターンを覚えることはユーザにとって負荷となるが、入力パターンを決定する際に物理キーボードのメタファを取り入れ、覚えやすくしている。全角/半角切り替えを左端、Space を中央付近、Enter を右端の位置としているが、これは物理キーボードにおいて全角/半角切り替えキーが左上に、Space キーが下段のさらに下ではあるものの中央に、Enter キーが右端に位置していることに合わせている。物理キーボードにおいて Tab はアルファベットキーの左に、Back Space と Delete はアルファベットキーの右に存在する。し

表 6.1: 複数の指による入力（黒丸は入力する指）

左手	右手	動作	入力内容
●●○○	○○○○	タップ	全角/半角
○●●○	○○○○	タップ	Tab
○○●●	○○○○	タップ	Space
○○●●	○○○○	フリック	キーセット切り替え
○○○○	○○●●	タップ	Enter
○○○○	○●●○	タップ	Back Space
○○○○	●●●○	タップ	Delete
○○○○	●●○○	フリック	矢印
●●●●	○○○○	プレス&ホールド	左手キャリブレーション
○○○○	●●●●	プレス&ホールド	右手キャリブレーション

かし、左端と右端を用いる入力パターンはすでに用いられているので、そこから指一つ分ずれたパターンに Tab と Back Space を割り当てた。Delete がまだ余っているため、Back Space と似た入力パターンを割り当てて覚えやすくした。矢印に関してはアルファベットキーの右に位置するが、その入力は主に右手の人差し指、中指、薬指によって行われる。また、タッチパッドでは 2 本指のフリックによってスクロールを行う製品が存在する。このとき用いられる指は主に人差し指と中指である。そこで、右手人差し指と中指のフリック入力を矢印に割り当てた。Meyboard ではフリック方向を 4 分割し、それぞれに 4 種類ある矢印の方向を割り当てた。

6.6 キーセット切り替え

Meyboard には 2 種類のキーセットが半角、全角文字それぞれにおいて設けられている。1 つはアルファベットセット（図 6.1、図 6.3）であり、もう 1 つは数字・記号セット（図 6.2、図 6.4）である。Meyboard には Leyboard に存在した Fn・テンキーセットが存在しない。何故ならば Fn・テンキーセットは文字入力に不要なキーによって構成されるキーセットであり、キーの数が少ない Meyboard は Leyboard 以上に入力可能なパターン数に制限があるためである。

Meyboard の各キーセットにおけるキーの配置は Windows 8 キーボード（第 5 章図 5.5）を参考にして決定した。

2 種類のキーセットを切り替えるために、左手人差し指と中指のフリック入力を用いる。これは Meyboard では親指を使わないこととしたためである。Leyboard の親指基点スライドは手の形を保ったままの入力をユーザに可能にしていた。しかし、手全体を動かすことを前提としたデザインになっているため、親指基点スライドが終了し、キーの位置が元に戻る際に手の位置を調整し直す必要があった。これは必然的にユーザの手元注視を必要とした。よって Leyboard と同様にキーセットの切り替えを採用した Meyboard では、手全体を動かすことな

しにキーセットの切り替えが行えるように、フリック入力によってキーセットを切り替えられるようにした。なお、キーセット切り替えには Shift 入力に相当する入力がないので、キーセットを切り替える際のフリック方向には一切の制約を設けていない。

6.7 キャリブレーション

Meyboard にもユーザの指にキーの位置と大きさを合わせることが可能であり、これをキャリブレーションと呼ぶ。

6.7.1 キャリブレーションの方法

Meyboard ではユーザが片手の人差し指から小指までの 4 本の指を置き、かつ各指がその位置を変えないまま、指を置いた時間から 500 ミリ秒が経過するとキャリブレーションが行われる。

Meyboard では Leyboard と異なり片手ごとにキャリブレーションが可能なようにした。これは、利便性のためであり、以下の理由に基づく。Meyboard では手元注視を減らすために、ユーザがキャリブレーションを頻繁に行うことを想定している。そのため、Leyboard のように両手の指を全て置く場合よりも素早い入力が可能な、片手の指を全て置く場合をキャリブレーションの条件として、キャリブレーションを片手ごとに行えるようにした。キャリブレーションに要する時間が長いほど、キャリブレーションが入力率を阻害する要因になるので、キャリブレーションのためのの入力は素早く行えるほうが良い。両手の指全て置く場合よりも片手の指を全て置く場合のほうが入力に時間がかからない理由は以下のとおりである。6.10 節においても述べるが、複数の指による入力を認識するときは、ユーザの各指が画面に置かれたとプログラムが判定する時刻に差があることを考えなければならない。ユーザの最初の指が画面に置かれた時刻から、最後の指が画面に置かれてから 500 ミリ秒が経過した時点の時刻までがキャリブレーションに要する時間となる。よって、ユーザが置く必要のある指の数が多いほどこの時間は長くなる。従って両手の指全て置く場合よりも片手の指を全て置く場合のほうが入力に時間がかからない。

6.7.2 キャリブレーションとキー入力の衝突回避

Meyboard ではユーザが画面に入力とは関係のない指を置きながら入力を行うことを許容している。すると、Leyboard のようにユーザが画面に一定数の指を置くことをキャリブレーションの条件とすると、入力中にキャリブレーションが行われ、キーの位置がユーザの意図とは無関係に変動する。そのため、Meyboard ではユーザが画面に一定数の指を置くこと以外にもキャリブレーションを行う条件を追加しなければならない。

Meyboard では一定数の指を置く条件に加えて、位置を一定時間変えないという条件を加えることにより、キー入力とキャリブレーションが競合しないようになっている。この時間を

プレス&ホールドリミットとする。プレス&ホールドリミットは本来プレス&ホールドの入力を判定するための閾値である。プレス&ホールドリミットを超えるまでユーザの指の移動量が閾値（フリック入力における指の移動量の閾値に等しい）未満であった場合プレス&ホールドと解釈される。Meyboard においてユーザがプレス&ホールド入力を行うのはキャリブレーションのみであるので、プレス&ホールドリミットはキャリブレーションに合わせて設定された。その値は以下のように定められた。Meyboard において文字入力に用いられる入力はタップ入力とフリック入力である。タップ入力、フリック入力のいずれの場合も、位置を 500 ミリ秒変えないという条件を満たした時点においてタップリミット（450 ミリ秒）やフリックリミット（500 ミリ秒）を超えている。そのためタップ入力やフリック入力を行ったときにキャリブレーションが行われることはない。キャリブレーションに要する時間が長いほど、キャリブレーションが入力率を阻害する要因になるのでキャリブレーションに要する時間は短いほうが良い。そのため、フリックリミットと同じ値である 500 ミリ秒をプレス&ホールドリミットにも適用した。

6.7.3　各タッチと指の対応

Meyboard には 6.9 節にて述べるように、入力に用いられたユーザの指を推測する機能がある。これを利用してキャリブレーション時に各タッチと指の対応をとる。

キャリブレーションを行うためには、ユーザが 4 本の指を置いたときに、それが片手の人差し指から小指までの 4 本の指である必要がある。ユーザの指を推測する機能を用いて上記を満たしたかをまず判断する。そのため、例えばユーザが左手、右手それぞれの人差し指と中指を画面に置いた場合は 500 ミリ秒経過してもキャリブレーションは行われない。なお、Meyboard の上記機能においては、ユーザがあるキーの上に指を置いたとして、そのキーに対応する指（表 6.2 に従う）が未入力の場合はそのキーの入力が対応する指によって行われたと解釈する。そのため、ユーザが置いた指の中に、その指と本来対応するキーの位置がずれているものがあった場合、片手の人差し指から小指までの 4 本の指を置いたことは推測できても、各指の実際の位置と推測上の指の位置が異なることがある。例えば、人差し指を中指のキーの上に置いたとき、実際においている指は人差し指であっても中指を置いたと解釈される。このとき、片手 4 本の指を全て置いたとしても、実際の指の位置と Meyboard が推測する指の位置が一致しない。そこで、キャリブレーションを行う前に各指を置いた位置の座標を画面の幅方向にソートする。こうすることにより、各指の実際の位置と推測上の指の位置を合わせている。

6.7.4　キーの配置規則

Leyboard のキャリブレーションはユーザの指の位置にホームポジションキーを配置し、その周囲に非ホームポジションキーを配置するものであった。ホームポジションキーは中段のキーであるが、Meyboard は Leyboard と異なり中段のキーが存在しない。そのため、キャリ

ブレーションにおいては Leyboard と異なった方法にてキーの位置と大きさをユーザの指に合わせる必要がある。

Meyboard ではユーザの指と指の中間の位置を通る、画面の高さ方向に平行な直線がキー間の高さ方向の境界となる仕様にした。これを満たすようにキーの幅を決定することがキャリブレーションの内容となった。一方、キーの高さは常に一定の値となっている。これはキャリブレーションによってキーの高さが小さくなりすぎることを避けるためである。Meyboard は画面を占有する領域が小さくなるように設計された。その結果、Meyboard は Leyboard に比べてその高さが著しく小さくなった。ここで、キャリブレーションによって各キーの高さをユーザの指の位置に合わせるとした場合、例えば指の位置が画面下端に近い場合にキーの高さが小さくなる。するとそのキーがユーザにとってかえって入力しにくくなると考えた。そこで、各キーの高さは常に一定の値をとることとした。

6.8 視覚フィードバック

Meyboard ではユーザの指が入力したキーの色が赤くなるという視覚フィードバックが存在する。その例を図 6.5 に示す。この例ではユーザは e キーを押している。

Meyboard は 6.9 節にて述べるように、ユーザが画面に指を置いたときにそれがどのキーの入力を意図したものか推測する。そのため、ユーザの指の位置の真下に存在するキーとは異なるキーが入力対象と判定されることがある。この場合、ユーザの指の真下に位置するキーは赤くならず、入力対象と判定されたキーが赤くなる。

なお、Meyboard は Leyboard と異なりフィードバック音は存在しない。何故ならばフィードバック音の存在するソフトウェアキーボードがまれであるためである。

図 6.5: 視覚フィードバック

6.9 ユーザが入力した指の推測

Meyboard ではキーを入力するユーザの指はキーごとに決まっていることを前提としている。Meyboard におけるキーと指の対応を表 6.2 に示す。ソフトウェアキーボードでは、物理的フィードバックが存在しない。そのため、ユーザにとってキーとキーの境界を手元注視な

しに把握することは難しい。その結果、ユーザが指を動かし過ぎて、隣の指が入力を担当するキーに指を置くことが考えられる。そこで、Meyboard ではユーザの指が画面に置かれたとき、それがユーザのどの指であるかを推測する機能を設けた。

画面にユーザの指が置かれていなかった場合は、ユーザの指の位置にあるキーからユーザの指を推測する。具体的には表 6.2 に従う。

表 6.2: Meyboard におけるキーと指の対応

左手				右手			
小指	薬指	中指	人差し指	人差し指	中指	薬指	小指
q	w	e	r、t	y、u	i	o	p
z	x	c	v、b	n、m	,	.	?

ユーザの指が押したキーからユーザの指を推測する場合、推測した指が既に画面に置かれているときがある。例えば、ユーザの左手人差し指が r キー上にあるときに、右手人差し指が t キー上に置かれたとする。このとき、ユーザの指が押したキーのみからユーザの指を推測すると、r も t も左手人差し指のキーなので、左手人差し指が 2 つ存在することになる。Meyboard ではキーの入力に用いられる指は表 6.2 のように定められているので、これはあり得ない事態となる。この場合、Meyboard はユーザの指（「対応を満たさない指」とする）の位置が既に置かれた指の位置の左右どちらに存在するかを調べる。この例では、左手人差し指が r 上に存在し、「対応を満たさない指」（右手人差し指だがこの時点の Meyboard ではそれが分からない）が t キー上に存在するので「対応を満たさない指」の位置は左手人差し指の右である。左右が分かった後に、その方向にまだ置かれていない指が存在するかを近い順に調べる。この例では、左手人差し指のすぐ右である、右手人差し指はまだ置かれていないという扱いになっている。まだ置かれていない指が存在した場合、「対応を満たさない指」をその指に割り当てる。この例では、「対応を満たさない指」に右手人差し指が割り当てられる。この例では右手人差し指が t キー上に置かれたことになっているので、Meyboard の推測結果は現実の状況に即している。仮に指が割り当てられなかった場合は逆方向を調べていく。これは、画面に対するユーザの手の置き方やユーザの指の長さによっては、指の左右関係（例: 左手人差し指の左隣は左手中指である）と実際に指が置かれた位置の左右関係が逆になる（例: 偶々手が傾いており、左手人差し指の位置が左手中指よりも左になっていた）ことがあるためである。

このように推測された指の情報を利用して、Meyboard はどのキーが入力されるかを決定する。入力が上段であるか下段であるかに関しては、画面の高さ方向における指の位置から判定するようになっている。この仕様のため、Meyboard の入力内容はユーザの指が置かれた位置にあるキーに対応するとは限らない。他の指の情報を利用してユーザの指を推測することによって、ユーザが手元注視をしていない状況においてもユーザの意図通りのキー入力が可能になる。例えば、ユーザの右手人差し指が画面に既に置かれていたとする。このとき、左手人差し指が本来右手人差し指と対応する y キーに置かれたとする。Meyboard はそれを左手人差し指による t キーの入力を意図した（t キーを入力しようとして指を動かしすぎた）もの

だと推測する。

6.10　タッチリンク

複数の指を組み合わせた入力を取り入れるために、複数の指による入力を検出する必要がある。ユーザが複数の指によって入力を行うとき、その入力を開始する時刻は理想的には等しい。しかし実際の入力においてはユーザのそれぞれの指が画面に置かれる時刻にずれが生じる。そこで複数の指が画面に連続して置かれた場合、最初の指が画面に置かれた時刻から一定時間内に画面に置かれた指を同時に画面に置かれたと解釈する必要がある。そこでタッチリンクという概念を設ける。リンクは入力に想定される指の組み合わせである。最初の指が画面に置かれた時刻から一定時間内に画面に置かれた指は全てタッチリンクの対象とする。この時間をタッチリンクスタートリミットと呼ぶことにする。

例えば表 6.1 における Space の入力のため、ユーザが左手人差し指と中指を画面に置いたとき、これらを左手人差し指と中指による 1 つのタッチリンクと解釈される。これを入力が「リンクされる」と呼ぶ。

ユーザの画面への指の置き方を表 6.1 の表記に倣い、指が画面に触れている状態を黒丸、そうでない状態を白丸によって表すとする。

例えば左手中指が画面に置かれて

○○●○　○○○○

となったときにタッチリンクが生成される。この後、タッチリンクスタートリミット内に左手人差し指が置かれて

○○●●　○○○○

となれば、人差し指の入力は中指の入力によって生成されたリンクに組み込まれる。このときを左人差し指と中指の入力が「リンクされた」、とする。タッチリンクスタートリミット超過後に置かれた指は既にあるタッチリンクと無関係であると解釈される。この場合新たなタッチリンクが生成される。

その後リンクを構成する複数の指のうち最初の指が画面に置かれた時刻から、一定時間後までにリンクを構成する指が全て画面から離れた場合、リンクを「完成した」とみなす。このとき、リンクを構成する指による入力が行われる。この時間をタッチリンクエンドリミットと呼ぶことにする。先にあげた例の場合、左手中指が画面に置かれて

○○●○　○○○○

となった時刻からタッチリンクエンドリミットが経過するまでに、左手中指と人差し指が置かれた

○○●●　○○○○

の状態になり、どちらかの指が画面から離れた

○○○●　○○○○

あるいは

○○●○　○○○○

の状態を経由し、さらにもう片方の指も画面から離れて
○○○○　○○○○
となればリンクが完成し、Space が入力される。タッチリンクエンドリミットを超過してもリンクが完成しなかった場合、そのリンクの構成に用いられた指は入力を意図したものではなく、ただ画面上に置かれただけであると解釈される。この場合、そのタッチリンクの情報は削除され、以降考慮されることはない。

以上をまとめると、左手中指が画面に置かれて
○○●○　○○○○
となった時刻からタッチリンクスタートリミットが経過するまでに左手中指と人差し指が置かれた
○○●●　○○○○
の状態を作り出し、タッチリンクエンドリミットが経過するまでにそれらの指を画面から離して
○○○○　○○○○
となればリンクが完成し、Space が入力される。タッチリンクスタートリミットが経過するまでに置かれなかった指はリンクされず、タッチリンクエンドリミットが経過するまでにリンクが完成しなかった場合はタッチリンクの情報が消滅する。

タッチリンクスタートリミットが長すぎると入力速度の制約となる上に、ユーザが素早く入力を行った際の誤入力の原因となる。それは最初のタッチダウンからタッチリンクスタートリミットが経過するまでに画面に置かれた指はリンク対象とみなされるからである。ゆえに、Meyboard の入力速度は最大（1 キー/タッチリンクスタートリミット）となる。これはタッチリンクスタートリミット内に高々1 つのキーしか入力できないことを意味する。

一方、タッチリンクスタートリミットが短すぎると、時間内にリンクの構成に必要な全ての指をユーザが画面上に置くことができず、リンクを構成することが不可能になる。

タッチリンクスタートリミットの適切な値は入力率から逆算することによって求められる。Leyboard の実験において、十分に慣れた被験者は平均 49.69wpm の入力率であった。Meyboard が理想的には同等の 50wpm の入力率となると想定する。すると Gentner による入力率の定義より、単語の入力率を 5 倍すると文字の入力率となるので、1 文字を入力するのにかかる時間は $60 \times 1000 \div (50 \times 5) = 240$ ミリ秒となる。つまり、ユーザの入力率が高々50wpm であるならば、タッチリンクスタートリミットは 240 ミリ秒以下であれば平均的には長すぎるということはない。但し、Leyboard の入力率の最大値は 61.27wpm であり、ユーザや入力内容によっては瞬間的にさらに高い入力率になった可能性も考えられる。そこで、Meyboard において想定される瞬間最大入力率を 100wpm とし、$60 \times 1000 \div (100 \times 5) \approx 120$ ミリ秒をタッチリンクスタートリミットとした。

一方のタッチリンクエンドリミットは（タッチリンクスタートリミット+フリックリミット）とした。各指の入力はタップやフリック入力とみなされる制限時間、即ちタップリミットやフリックリミットの影響を受け、それらの時間を超過した場合入力を意図したものではなく、ただ画面上に置かれただけであると解釈される。リンクを構成する指に 1 つでもこれらの時

間超過が出現すると入力が行われないため、実質これらの制限時間が制約となる。タップリミットとフリックリミットでは後者の方が長く設定されている。よってタッチリンクスタートリミットが超過する瞬間に画面にユーザの指が置かれ、タッチリンクにその指が組み込まれた場合においても、タッチリンクを構成する最初の指が画面に置かれた時刻からタッチリンクスタートリミット+フリックリミットの時間が経過するとその入力は確実に無効化されることになる。故にタッチリンクエンドリミットはタッチリンクスタートリミット+フリックリミットとすれば十分である。タップリミットやフリックリミットが設定されているにもかかわらずタッチリンクエンドリミットを設けるのは、タップやフリック動作の判定を行う前に制限時間を明らかに超過しているタッチリンクを排除するためである。

6.11 実装環境

Meyboard は Leyboard と同様に C#を用い、.NET Framework 4/WPF4 の API を利用して、マルチタッチ対応 WPF アプリケーションとして実装した。

第7章　Meyboardの被験者実験

Meyboard の入力性能と手元注視を減らす可能性について検証するため、Meyboard を手元注視に制限を設けない場合と、手元が全く見えない場合において、それぞれの入力性能を調査する実験を行った。Meyboard のキーの高さは全て 30mm であった。Meyboard のキーの幅はキャリブレーションの結果によって変わるため不定であるが、r、v、u、m キーのみ固定されており、その値は 16mm であった。Meyboard 全体の高さは 6cm、幅は 31cm であった。

7.1　実験環境

Meyboard を動作させるデバイスとして、Leyboard の被験者実験時と同様に Acer 社の ICONIA-F54E を用いた。ICONIA-F54E を使用した実験環境を図 7.1 と図 7.2 に示す。図 7.1 は手元注視に制限を設けない場合（手元注視可能条件）であり、図 7.2 は手元が全く見えない場合（手元注視不可能条件）である。後者においては、手元を布で覆うことにより手元が見えないようにした。被験者の左方には物理キーボードが設置されている。これは、被験者が Meyboard におけるキーとそれを入力する指の対応（第 6 章表 6.2）が分からなくなったときに、物理キーボードを見ることによってそれを思い出せるようにしたためである。

7.2　被験者と実験タスク

本実験の被験者は男性の 22 歳から 25 歳の合計 3 名からなる。Leyboard の実験時と同様に、人間生活工学研究センターによる日本人の手の寸法データ集 2010 にて用いられた形式に従った、各被験者の手指の大きさを表 7.1 から表 7.3 に示す。

筆者は本実験のタスクとして、英字の入力を選択した。タスクは練習タスクと本番タスクに分かれていた。各タスクでは英文または意味のない文字や単語の羅列の入力を 10 回行った。この 10 回入力を行うことを Leyboard の実験時と同様にセットと呼ぶ。

ユーザはまず手元注視可能条件において練習タスクを 1 セット、本番タスクを 2 セット、続けて手元注視不可能条件において練習タスクを 1 セット、本番タスクを 2 セットの入力を行った。以上の内容を 2014 年 1 月 8 日から同年 1 月 10 日までの 3 日間にわたって毎日行った。但し、ユーザが Meyboard に習熟しているとはいいがたい初日はユーザの疲労や実験に要する時間を考慮して各条件において練習タスクを 1 セット、本番タスクを 1 セットずつのみ行った。

練習タスクにおいては以下の内容の入力を行った。

- A quick brown fox jumps over the lazy dog.

図 7.1: 実験環境（手元注視可能条件）

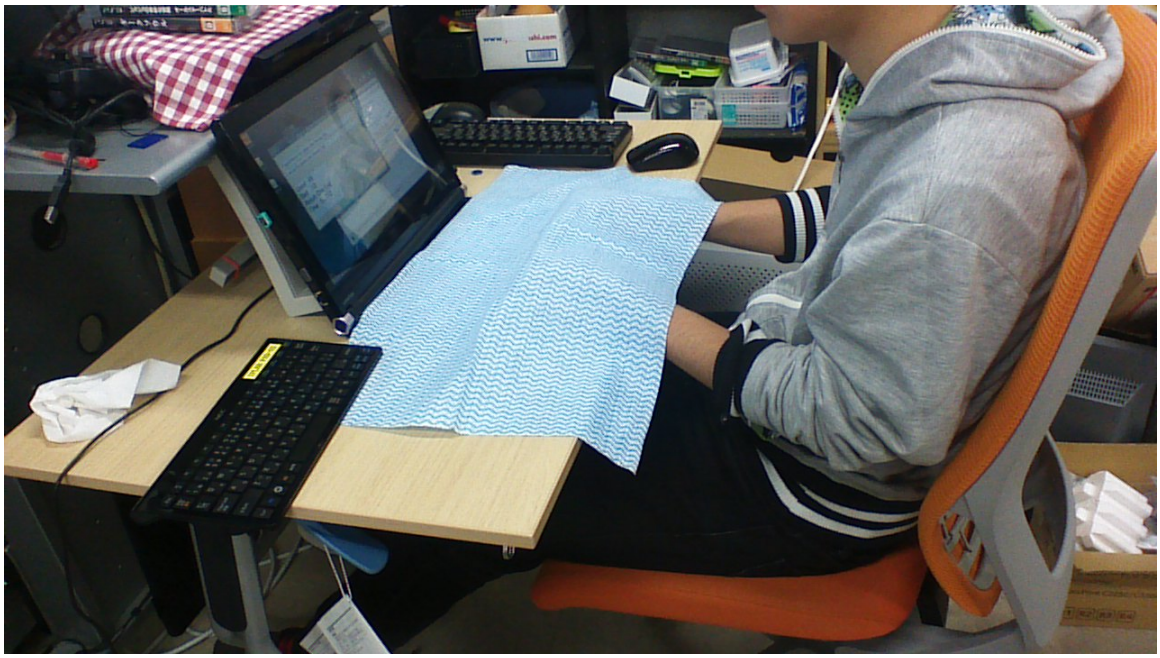

図 7.2: 実験環境（手元注視不可能条件）

表 7.1: 被験者 A の手指の大きさ

名称	左手（mm）	右手（mm）
手長１（茎突点）	181.0	185.2
手長２（屈曲線）	180.3	178.9
手掌長（茎突点）	110.8	109.0
指尖・指節点距離	94.7	93.9
第一指長	51.2	55.7
第二指長	67.0	69.2
第三指長	76.6	80.1
第四指長	68.9	70.0
第五指長	53.4	53.8
第二指近位関節－遠位関節長	16.6	18.5
第一指関節－指先長	28.3	28.0
第二指遠位関節－指先長	23.3	25.3
第一指爪基部長さ	12.7	14.0
第二指爪基部長さ	10.2	12.1
第三指爪基部長さ	10.7	11.1
第四指爪基部長さ	9.5	10.6
第五指爪基部長さ	9.0	9.5
茎状突起間幅	55.9	53.2
手幅（斜め）	84.3	85.3
第一指爪基部幅	19.8	19.6
第二指爪基部幅	15.3	15.1
第三指爪基部幅	15.5	16.5
第四指爪基部幅	16.8	15.1
第五指爪基部幅	13.3	14.9
第一－第五指尖端間最大距離	181.9	174.9

表 7.2: 被験者 B の手指の大きさ

名称	左手（mm）	右手（mm）
手長１（茎突点）	180.2	179.1
手長２（屈曲線）	168.6	173.5
手掌長（茎突点）	116.8	112.2
指尖・指節点距離	91.6	93.3
第一指長	57.3	56.1
第二指長	67.3	63.3
第三指長	70.4	71.7
第四指長	66.2	65.9
第五指長	51.4	54.7
第二指近位関節－遠位関節長	21.6	20.4
第一指関節－指先長	30.0	33.5
第二指遠位関節－指先長	23.7	24.3
第一指爪基部長さ	14.8	16.1
第二指爪基部長さ	14.1	12.2
第三指爪基部長さ	13.4	12.8
第四指爪基部長さ	13.1	12.6
第五指爪基部長さ	10.2	10.8
茎状突起間幅	57.8	57.5
手幅（斜め）	84.4	86.4
第一指爪基部幅	19.7	21.1
第二指爪基部幅	17.5	16.8
第三指爪基部幅	16.8	16.5
第四指爪基部幅	15.5	16.7
第五指爪基部幅	14.4	15.1
第一－第五指尖端間最大距離	193.8	199.0

表 7.3: 被験者 C の手指の大きさ

名称	左手（mm）	右手（mm）
手長１（茎突点）	186.0	186.1
手長２（屈曲線）	179.0	182.2
手掌長（茎突点）	126.0	123.9
指尖・指節点距離	99.8	103.8
第一指長	59.5	64.2
第二指長	69.4	68.5
第三指長	77.3	76.7
第四指長	67.4	72.9
第五指長	51.9	56.3
第二指近位関節－遠位関節長	21.4	22.3
第一指関節－指先長	29.4	30.6
第二指遠位関節－指先長	23.7	24.9
第一指爪基部長さ	14.2	14.7
第二指爪基部長さ	13.0	12.8
第三指爪基部長さ	12.9	13.6
第四指爪基部長さ	12.7	12.6
第五指爪基部長さ	9.7	11.7
茎状突起間幅	55.0	54.8
手幅（斜め）	78.2	80.3
第一指爪基部幅	18.1	17.2
第二指爪基部幅	13.7	15.4
第三指爪基部幅	15.8	15.6
第四指爪基部幅	14.9	15.2
第五指爪基部幅	13.5	13.0
第一－第五指尖端間最大距離	190.5	193.6

- Cwm fjord veg balks nth pyx quiz.
- Pack my box with five dozen liquor jugs.
- Jackdaws love my big sphinx of quartz.
- Jumbling vext frowzy hacks PDQ.
- rtfgvb alks dk yuhjnm jfgh.
- asdf jkl ruty fjgh vmbn.
- RTFGVB ALKS DK YUHJNM JFGH.
- ASDF JKL RUTY FJGH VMBN.
- Daft skull god mat envy flick had burnt duke jar.

上記は、全てのキーの入力を行うようにパングラム 5 つと、Meyboard におけるフリック入力、Shift 入力、二段フリック入力、そして担当するキーの数が多い左右の人差し指による入力が多くなるように設定された意味のない文字や単語の羅列 5 つからなる。なお、これらを入力する順番は練習タスクごとにランダムに設定された。練習タスクはユーザにとって Meyboard の入力の練習となるように設けられており、練習タスクの入力結果は実験結果においては考慮されない。

本番タスクにおいては MacKenzie らによる文章セット [MS03] を加工し、最初の文字を大文字にして最後にピリオドを加えたものを入力対象とした。タスクごとに、ここから 10 文がランダムに選択され、ユーザはその入力を行った。

本実験においても Leyboard の実験時と同様に、入力を間違えた場合、被験者はそこから正しい入力をやり直す必要がある。言い換えれば、被験者が正しい入力を行わない限りタスクは進まない。そのため、被験者は最終的には正しい文字を入力することになる。被験者が入力を間違えた場合、それを通知するブザーが鳴る。

手元を隠す本実験においては、ユーザの意図しないキーセット切り替えがユーザの誤入力によって発生した場合、タスクを完了することが不可能になる可能性がある。何故ならば、手元が見えない以上、被験者はキーセットが切り替わったことを知る手段がないからである。そこで本実験においてはキーセット切り替えを初めとする複数の指による入力を、空白（Space）の入力以外できないようにした。

ユーザはタスク中に何回キャリブレーションを行ってもよいこととした。特に、手元注視不可能条件においては頻繁にキャリブレーションを行うことが期待された。それは手元注視不可能条件時には、ユーザは自分の指をキーの位置に合わせることができているか、それを知るすべがないためである。そこでユーザはキャリブレーションを行うことにより、適宜キーの位置を指に合わせる必要があった。そのため、実験中においてキーの位置は頻繁に変動した。また、タスク中とは別に、各タスクを開始する前にもユーザはキャリブレーションを行った。

実験の全日程終了後、被験者にはアンケートに回答してもらった。

7.3　実験結果

手元注視可能条件における入力率の変動を図 7.3 に、手元注視不可能条件における入力率の変動を図 7.4 に示す。いずれの条件においても入力率は上昇傾向にあった。

手元注視可能条件における被験者ごとの入力率の平均値は被験者 A が 14.69wpm、被験者 B が 17.95wpm、被験者 C が 14.45wpm となった。手元注視不可能条件における被験者ごとの入力率の平均値は被験者 A が 14.58wpm、被験者 B が 14.85wpm、被験者 C が 12.42wpm となった。いずれの被験者においても、手元注視不可能条件の入力率は手元注視可能条件よりも低くなった。

手元注視可能条件における被験者ごとの入力率の最大値は被験者 A が 19.32wpm、被験者 B が 20.80wpm、被験者 C が 16.40wpm となった。これは全員とも実験 3 日目（最終日）に計測された。

手元注視不可能条件における被験者ごとの入力率の最大値は被験者 A が 18.10wpm、被験者 B が 20.68wpm、被験者 C が 16.54wpm となった。これは全員とも実験 3 日目（最終日）における本番タスク第 2 セット目（実験全体における最終タスク）に計測された。

手元注視可能条件における被験者ごとの入力率の最小値は被験者 A が 11.48wpm、被験者 B が 12.53wpm、被験者 C が 11.56wpm となった。これは全員とも実験初日の本番タスク第 1 セット目に計測された。

手元注視不可能条件における被験者ごとの入力率の最小値は被験者 A が 11.24wpm、被験者 B が 9.02wpm、被験者 C が 9.02wpm となった。これは全員とも実験初日の本番タスク第 1 セット目に計測された。

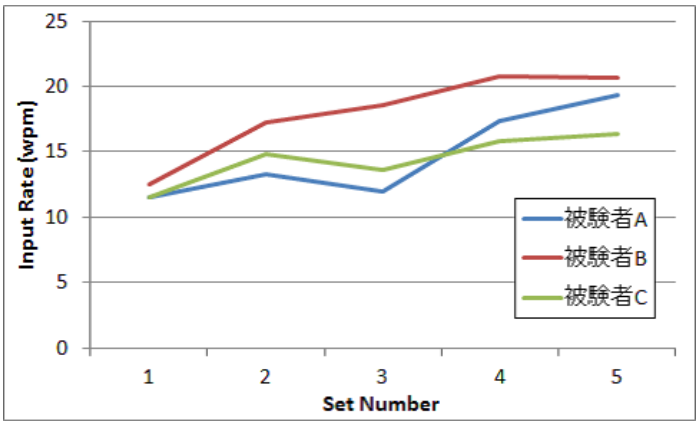

図 7.3: Meyboard の入力率（wpm）（手元注視可能）

手元注視可能条件におけるエラー率の変動を図 7.5 に、手元注視不可能条件におけるエラー率の変動を図 7.6 に示す。手元注視可能条件においては被験者 C を除いてエラー率はやや上昇傾向にあった。手元注視不可能条件においては被験者 A を除いてエラー率は下降傾向にあった。

手元注視可能条件における被験者ごとのエラー率の平均値は被験者 A が 5.74%、被験者 B

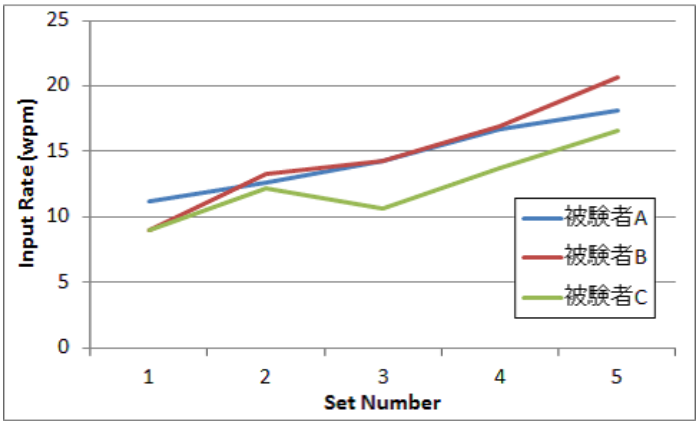

図 7.4: Meyboard の入力率（wpm）（手元注視不可能）

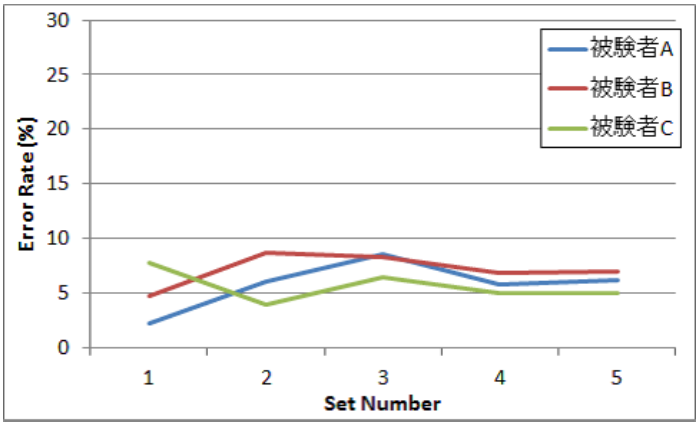

図 7.5: Meyboard のエラー率（手元注視可能）

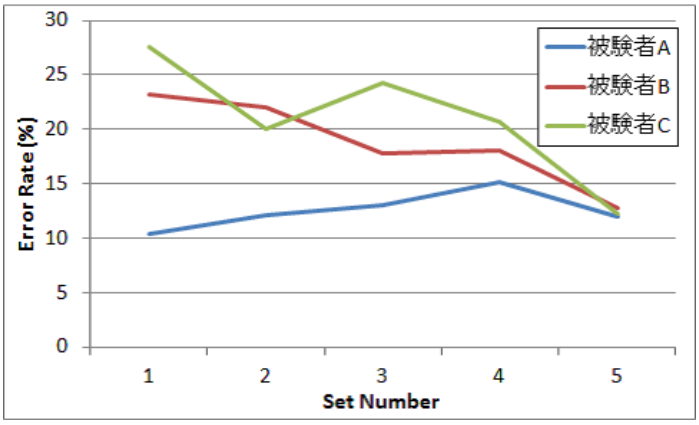

図 7.6: Meyboard のエラー率（手元注視不可能）

が 7.10%、被験者 C が 5.62%となった。手元注視不可能条件における被験者ごとのエラー率の平均値は被験者 A が 12.53%、被験者 B が 18.77%、被験者 C が 20.94%となった。いずれの被験者においても、手元注視不可能条件のエラー率は手元注視可能条件よりも非常に高くなった。

手元注視可能条件における被験者ごとのエラー率の最大値は被験者 A が 8.52%、被験者 B が 8.63%、被験者 C が 7.79%となった。これは実験初日または 2 日目に計測された。

手元注視不可能条件における被験者ごとのエラー率の最大値は被験者 A が 15.14%、被験者 B が 23.15%、被験者 C が 27.55%となった。これは被験者 C を除いて、実験初日の本番タスク第 1 セット目に計測された。

手元注視可能条件における被験者ごとのエラー率の最小値は被験者 A が 2.24%、被験者 B が 4.75%、被験者 C が 3.91%となった。これは被験者 C を除いて、実験初日の本番タスク第 1 セット目に計測された。

手元注視不可能条件における被験者ごとのエラー率の最小値は被験者 A が 10.41%、被験者 B が 12.74%、被験者 C が 12.20%となった。これは被験者 A を除いて、実験 3 日目（最終日）における本番タスク第 2 セット目（実験全体における最終タスク）に計測された。

第8章　Meyboardの被験者実験を受けての考察

本章では、Meyboard の手元注視可能条件と手元注視不可能条件のそれぞれにおける入力性能を比較し、Meyboard の設計の妥当性や問題点を検証する。

8.1　条件ごとの数値の差

Leyboard と Windows 7 キーボード、それぞれのキーボードの入力率とエラー率の全データに関して、それぞれ平均値に有意な差があるかを確かめるために、対応のある t 検定を行った。その結果、手元注視可能条件と手元注視不可能条件の入力率に有意差がないことが判明した。

- 被験者 A: $t = 0.1828, p - value = 0.8638 > 0.01$
- 被験者 B: $t = 3.8737, p - value = 0.01794 > 0.01$
- 被験者 C: $t = 3.6405, p - value = 0.02196 > 0.01$

つまり、Meyboard は手元注視を行わないときに手元注視可能時よりも入力率が劣るとは言えない。一方、エラー率においては手元注視不可能条件が手元注視可能条件に対して有意に高いことが判明した。

- 被験者 A: $t = -7.7544, p - value = 0.00149 < 0.01$
- 被験者 B: $t = -5.5762, p - value = 0.00507 < 0.01$
- 被験者 C: $t = -7.0923, p - value = 0.002087 < 0.01$

つまり、Meyboard は手元注視を全く行わなかった場合、手元注視可能時と比べてエラー率が高くなる。

以上の結果から Meyboard は手元注視を行わなくても手元注視可能時とほぼ同等の入力率となるが、その際は手元注視可能時と比べて多量のエラーが発生する。本実験ではエラーをもたらした入力に要した時間は入力率に影響を与えた（正しい入力を行わない限りタスクを進めることができない）が、エラーをもたらした入力を消すという行程が存在しなかった。そのため間違えた入力を消すのに要する時間が入力率に考慮されていない。よって、エラー修正にかかる時間を考えると実際の入力率は本実験にて測定された値よりも低くなると思われる。しかしながら、それは手元注視可能条件と手元注視不可能条件双方において成立するため、結果として入力率はやはりほぼ同等になると考えられる。

Meyboard では手元注視なしに入力を行うことは入力率に支障をきたすとは言えない。つまりユーザが Meyboard の手元注視を減らしたとしても、それが入力率に支障をきたすとは言え

ない。ここから、Meyboard は手元注視を減らす可能性がある。しかし、現状の Meyboard は手元注視なしではエラー率が高いので、エラー率を抑えるためにはある程度手元を注視する必要はある。実験が進むにつれて 3 人中 2 人の被験者にエラー率が下がる傾向があり、被験者全員に入力率向上の傾向があったので、ユーザの習熟につれて Meyboard の入力性能は向上する可能性がある。手元注視不可能条件のエラー率が手元注視可能条件のエラー率と同等まで下がるのであれば、Meyboard の入力に手元注視は影響しないことになる。すると Meyboard は手元注視を減らすソフトウェアキーボードと言えるが、この効果が本当にあるかは今後検証する必要がある。

8.2 エラー内容分析

次に、エラーの内容を分析した。Leyboard の被験者実験時と同様に、実験途中の全てのキー入力内容に対してログを取った。そこで上記ログを参照し、Leyboard の被験者実験時と同様アルゴリズムに基づいて条件ごとに誤字と脱字の判定を行った。

被験者のエラーを合計すると、手元注視可能条件では 223 回、手元注視不可能条件では 590 回となった。その内訳は以下のとおりである。手元注視可能条件では 193 回の誤字と 30 回の脱字が存在した。手元注視不可能条件では 559 回の誤字と 31 回の脱字が存在した。エラーの回数は手元注視不可能条件が圧倒的に多いが、脱字に関してはほぼ差がない。手元注視の条件別にエラー回数を文字別に分けてグラフ化した。誤字に関しては図 8.1 に、脱字に関しては図 8.2 に示す。

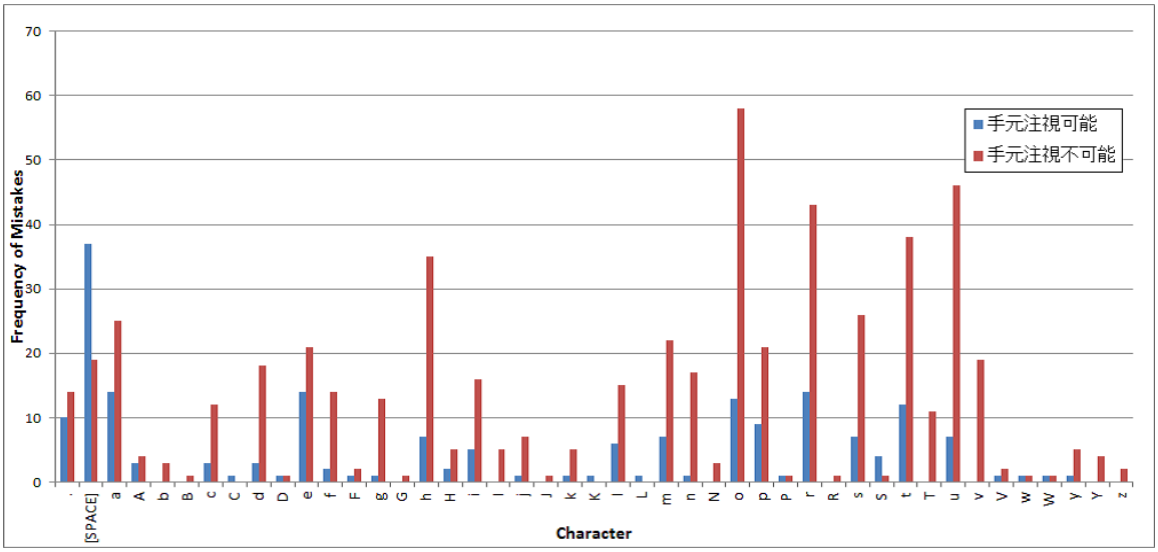

図 8.1: 誤字の分析結果

誤字に関しては手元注視不可能条件の回数が多く、また特定の文字に偏りが生じていることが分かる。

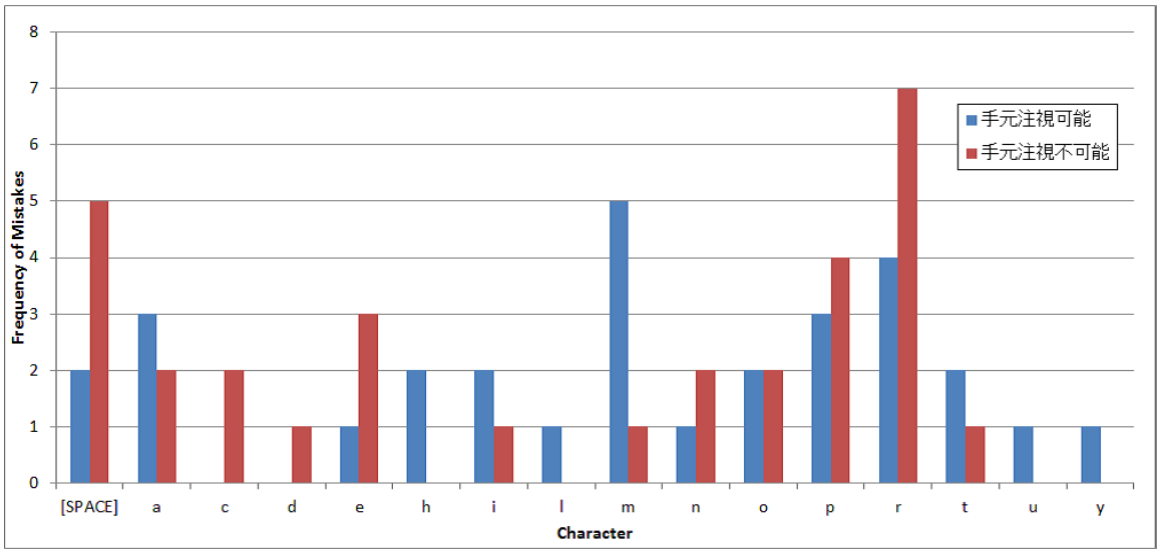

図 8.2: 脱字の分析結果

脱字に関しては手元注視可能条件において m の脱字回数が、手元注視不可能条件において r の脱字回数が多いようにも見えるが、絶対数が少ないため、ここから各条件の特徴を読み取ることは難しい。なお、Meyboard も Leyboard と同様にタップによって入力を行うため、Leyboard によって生じた、見かけ上の入力入れ替わりによる脱字が発生する可能性があることを述べておく。これはタップによって入力を行うソフトウェアキーボードの性質であるため、手元注視の条件は問わない。

8.2.1 誤字内容の分析

各条件における誤字内容に関して、多かったパターン上位 10 位をそれぞれ表 8.1 と表 8.2 に示す。

表 8.1 を見ると、正解と誤字の組み合わせの大半は横に隣り合ったキーによって構成されている。例外は Space と r、e と n、a と q、a と n である。Space と r は左手人差し指と中指のタップによって空白を入力しようとしたところ、何かしらの要因で両者にタッチリンクが構成されなかったために生じたエラーであると考えられる。被験者のアンケート回答によると、1 人の被験者が「反応が悪い」、1 人の被験者が「エラーが生じやすい」と答えていた。タッチリンクが構成されるためには、タッチリンクスタートリミット以内に入力に必要な指が全て画面に置かれる必要がある。この時間が短すぎたことが、反応が悪い、エラーが生じやすいと被験者に感じさせた原因である可能性がある。タッチリンクスタートリミットを長く設定するほど、空白に関しては入力しやすくなるが、入力率の上限がこの影響を受けるため、このトレードオフをより慎重に調べる必要がある。a と q はタップすべきところをフリックしたと推測される。但し絶対数が少ないので、ここから q キーのフリック入力が誤入力となりや

すいとは言えない。e と n、a と n は位置が全く異なるキーであるが、これらのエラーも絶対数が少ないので偶々そのような入力ミスが生じたと思われる。

表 8.2 を見ると、人差し指に対応するキーの範囲内における誤字（人差し指担当キーを別の人差し指担当キーと間違える）が多いことが分かる。人差し指は対応するキーの数が多く、それらのキー入力のための運指の使い分けが求められる。手元注視不可能条件においては、運指の使い分けを指先の感覚のみから判断しなければならないため、習熟が不足している時点ではどうしてもエラー率が高くなると考えられる。

表 8.1 と表 8.2 の双方に存在する正解と誤字の組み合わせは、o と i、p と o、u と y、r と e、m とコンマである。これらに共通する要素が手元注視の条件を問わずに Meyboard において生じやすい誤字のパターンである。この中では u と y のみが人差し指に対応するキーの範囲内における誤字である。ここから、人差し指に対応するキーの範囲内における誤字は手元注視不可能条件に多い現象であることが分かる。さて、m とコンマを除いて、これらの正解と誤字の組み合わせに共通するのは、これらが全て上段のキーであることと、正解のキーの左に位置するキーの入力を行う誤字になっていることである。

この誤字は正解のキーに対応する指がその左隣のキーに触れることによって生じるという仮説を立てた。そう考える根拠は、物理キーボード入力時において上段のキーを入力する際に正解のキーに対応する指が左上方向に移動しているためである。物理キーボードでは各段のキーが幅方向にずれて配置されている。そのため、正解のキーに対応する指は上段のキーを入力するときは左上方向に、下段のキーを入力するときは右下方向に移動する。一方、Meyboard ではキーは格子状に配置されているため、正解のキーに対応する指は上段のキーを入力するときは真上方向に、下段のキーを入力するときは真下方向に移動することが望ましい。しかし、実際は物理キーボードの影響を受け、運指は左上方向または右下方向に移動していたとする。すると、上段のキー（例: o）を入力しようとした場合はその左隣のキー（例: i）に触れることがあり得る。下段のキーである m とコンマにおいても、m を入力しようとして右手人差し指が右下方向へ動き、結果としてコンマに触れたと考えれば矛盾はない。

以上の仮説が正しいとすれば、Meyboard のキーも物理キーボードのように段ごとにずらして配置することが、ユーザの慣れやすさの観点から望ましいように思われる。一方で、第 6 章にて述べたように、段ごとのキーの位置を物理キーボードのようにずらすと、キャリブレーション時のユーザの指の位置から b、y キーの位置までの距離が遠くなる。すると、ユーザの指がこれらのキーに届きにくくなり、誤入力の可能性が高くなる。その届きにくさは物理キーボードの場合と同等と考えられるが、物理的フィードバックのないソフトウェアキーボードでは入力後にキャリブレーション時の指の位置へユーザが指を戻すことが物理キーボードと異なり困難である。よって、段ごとのキーの位置をずらすとキャリブレーションの効果が薄くなる。

筆者は以上の理由から Meyboard のキーの配置をあえて格子状にし、本章の最後に示す改良した Meyboard においても格子状のキー配置を保った。しかし、キーの配置を格子状にすることと、物理キーボードに合わせてずらすことの、どちらがよりエラー率を低減させるのかは今後検証する必要がある。

表 8.1: Meyboard 誤字内容上位（手元注視可能）

回数	正解	誤字内容	全体中の比率（%）	比率の累計（%）
24	[SPACE]	r	12.44	12.44
8	o	i	4.15	16.58
7	p	o	3.63	20.21
7	.	[COMMA]	3.63	23.83
6	u	y	3.11	26.94
6	r	e	3.11	30.05
5	e	n	2.59	32.64
5	a	q	2.59	35.23
4	m	[COMMA]	2.07	37.31
4	a	n	2.07	39.38

表 8.2: Meyboard 誤字内容上位（手元注視不可能）

回数	正解	誤字内容	全体中の比率（%）	比率の累計（%）
52	o	i	1.97	48.12
44	u	y	1.97	50.09
35	t	r	1.79	51.88
21	r	e	1.79	53.67
17	h	j	1.61	55.28
16	v	b	1.43	56.71
16	r	t	1.43	58.14
16	m	[COMMA]	1.25	59.39
15	p	o	1.25	60.64
13	s	w	1.07	61.72

8.3　被験者アンケートより得た知見

本実験においては実験の全日程終了後に被験者にアンケートを書いてもらっている。その結果をここにまとめる。各項目では必要に応じて 5 段階のリッカート尺度よる評価と、コメントを残してもらった。

8.3.1　Meyboard の疲労感に関して

5 段階のリッカート尺度における評価では、2 人が 3、1 人が 4 となった。それぞれのコメントを以下に要約する。

- フリック入力が疲労の原因となる。
- 入力失敗による精神的疲労の影響が大きい。
- 慣れによって疲労感は軽減される。

入力に必要な動作の種類や、入力の成功率が疲労感に影響を与えていたことが分かった。慣れによって疲労感は軽減されるというコメントは、Meyboard は慣れることのできる設計であるという意味合いを含んでいると思われる。

8.3.2　左手人差し指と中指のタップによる Space の入力に関して

5 段階のリッカート尺度における評価では、2 人が 4、1 人が 2 となった。それぞれのコメントを以下に要約する。

- 設計としては良いがエラーとなることが多かった。
- 従来のソフトウェアキーボードにおける Space の位置をタップすることがあった。入力は左手に限定せずに右手でも可能にしたほうが良い。
- 設計は良いが反応しないことがあった。

設計に関しては概ね好意的な評価であったが、入力失敗率の高さを問題視していた。8.2.1 節にて述べたように、タッチリンクスタートリミットの値を適切に設定することによって、入力失敗率は改善すると考えられる。また、左右問わずに入力を可能にすべきと主張した被験者がいた。物理キーボードにおいては Space キーを入力する手は片方のみに定められてはいないので妥当な指摘である。

8.3.3　フリック入力に関して

5 段階のリッカート尺度における評価では、全員 3 となった。それぞれのコメントを以下に要約する。

- 面白い。

- 実験後半時にもタップと間違えることがあった。
- フリック入力は疲労感をもたらした。手元注視不可能時に指の動きが大きくなった。

タップ入力とフリック入力を間違えるのはMeyboardにおいてどうしても生じるミスであると言える。但し、間違えたことを自覚しているため、ユーザの入力内容と実際の入力内容にユーザは矛盾を感じてはいない。手元注視不可能条件において指の動きが大きくなるのは、手元が見えないことによりフィードバックがないため、ユーザが入力した感覚を得ようとしたからであると思われる。このような心理的影響は興味深い知見である。指をどれだけ動かせばフリック入力と認識されるのかを、被験者が十分に理解したならば、この心理的影響は弱くなると筆者は考える。

8.3.4 フリック入力の方向に関して

被験者にそれぞれがフリック入力によって最も用いた方向を回答してもらった。以下に要約する。

- 上。特に理由はないが最初にこうした以上この方向を使い続けた。
- 上段のキーにおいて下。最もやり易いと思ったため。
- 上段のキーにおいて下。自分で一意に方向を決めたほうが学習しやすい。

どの被験者も一意に方向を定め、それを使い続けたことがうかがえる。

8.3.5 Shift 入力に関して

5段階のリッカート尺度における評価では、4、2、3と分かれた。それぞれのコメントを以下に要約する。

- 面白い。
- 薬指、小指の場合難しい。
- 何故左なのか分からない。

Shift入力を困難と感じた被験者は低く評価していた。そもそも左方向であることの妥当性を疑問視している被験者もいた。これに関しては、アルファベットキーの左にShiftキーが存在するメタファであることの説明があれば、被験者に受け入れられた可能性がある。

8.3.6 二段フリック入力に関して

5段階のリッカート尺度における評価では、1人が4、2人が2となった。それぞれのコメントを以下に要約する。

- Shift入力が大文字であるので、中段のキーにおいてもその方向になるのは使いやすい。

- 入力ミスが多発した。
- 方向転換がややこしい。

フリック入力と Shift 入力を組み合わせることを使いやすいと評価する被験者がいた。これは分かりやすさを意味するが、それと入力の行いやすさは別問題であることが、他の被験者のコメントからうかがえる。二段フリックにおいて、この点はトレードオフであると思われる。

8.3.7　二段フリック入力の方向に関して

被験者にそれぞれが二段フリック入力によって最も用いた方向を回答してもらった。以下に要約する。

- 上の後に左。フリック入力が上であったため。
- 下の後に左。最も自然であると思ったため。
- 下の後に左。フリック入力が上段のキーにおいて下であったため。

フリック入力と同様にどの被験者も一意に方向を定め、それを使い続けていた。なお、どの被験者も最初に指を動かす方向がフリック入力の方向と一致している。

8.3.8　キャリブレーションに関して

5 段階のリッカート尺度における評価では、2、3、4 と分かれた。それぞれのコメントを以下に要約する。

- キャリブレーション時にキーの高さも変わって欲しい。また、キャリブレーション時に r、v（または y、n）キーと t、b（または u、m）キーの境界上に人差し指が来ることが多くて嫌だった。
- キーの高さも変わって欲しい。人差し指の位置は r、v（または u、m）キー上にして欲しい。
- 特に問題はなかった。

キャリブレーション時にキーの高さを変える要望が多かった。人差し指の位置に関しての指摘もあった。r、v キーや u、m キーの幅を固定値にしたため、ユーザの指の置き方次第では人差し指の位置の下に r、v キーや u、m キーが来るとは限らなくなった。そのためコメントで挙げられたようなキーの配置になる。r、v キーや u、m キーの幅は固定値にせず、ユーザに合わせることが求められる。

8.3.9　画面上に指を置きながらの入力に関して

5 段階のリッカート尺度における評価では、4、1、3 と分かれた。それぞれのコメントを以下に要約する。

- 良い。ミスが減る気がする。
- ほぼ使わなかった。手を置きながらの運指を行う方法が分からない。
- 使うことが少なかった。キャリブレーションが発動するのを心配したため。

本機能を使わないユーザは多かった。使いどころが分からないとの意見も出た。これに関して、筆者は人差し指のキーを入力する際には手の位置が安定するので有用であると考える。また、キャリブレーションが行われることを問題視したユーザがいた。仕様上、一度キャリブレーションが行われると、ユーザのいずれかの指が画面から離れるまでキャリブレーションは行われないので、待機状態に移行しない限りは入力中にキャリブレーションは行われない。被験者に対して、この点に関しての説明が足りなかった可能性がある。

8.3.10　ユーザが入力した指の推測機能に関して

5 段階のリッカート尺度における評価では、4、1、3 と分かれた。それぞれのコメントを以下に要約する。

- どのように動いているのか分からない。
- 使い方が分からない。
- 存在を知らなかった。

本機能は全てのユーザが自覚的には用いなかった。使いどころが分からないとの意見がこちらも出た。筆者は横に隣り合ったキーを誤って入力しそうなときに用いると良いと考える。また、この機能を知らないという被験者もいた。実験を行うにあたって被験者への説明が不十分であった可能性もある。

8.3.11　Meyboard のキー配置に関して

5 段階のリッカート尺度における評価では、3、2、4 と分かれた。それぞれのコメントを以下に要約する。

- r、v、u、m キーの領域が小さい。
- 人差し指の外側のキーが狭い。
- 大体良いが、キーの位置が段ごとにずれていたほうが良いかもしれない。

人差し指が担当する r、v、u、m キーの幅が小さいと感じる被験者が多かった。この内容に関しては 8.3.12 節にて考察する。物理キーボード同様に、段ごとにキーの位置がずれていることを望む被験者もいた。8.2.1 節にて述べたように、筆者は b、y キーに指が届きにくくなることを危惧し、あえて Meyboard のキー配列を格子状にした。しかし、格子状にキーを並べることの妥当性をやはり検証するべきである。その理由の 1 つは、キーの位置が段ごとにずらすと入力しやすくなる可能性を被験者が主張したことである。そして 8.2.1 節で挙げたように、キーを格子状に並べたことが誤字をもたらした可能性があるためである。

8.3.12　入力しやすい、あるいはしにくいキーに関して

被験者にそれぞれが入力しやすい、あるいはしにくいと感じたキーに関して回答してもらった。以下に要約する。

- 人差し指によって入力するキーが全て入力しにくい。
- v が圧倒的に入力しにくい。l、s、d、k が入力しやすい。
- q、a、z、p、ピリオド、t、y が入力しやすい。r、f、v、u、j、m が入力しにくい。

被験者を問わず、人差し指が担当するキーに関しては入力しにくいと感じるものが存在していた。人差し指は担当するキーの数が多いため、特に手元注視不可能条件においては非常に入力しにくかったと思われる。また、r、v、u、m キーの幅が被験者にとって狭く、さらに入力をしにくくしていたことがうかがえる。8.3.8 節においても述べたが、やはりこのキーの幅は固定値ではなくユーザに合わせる必要がある。入力しやすいと感じたキーに関しては被験者によって意見が分かれ、人差し指のキーのように他の被験者に入力しにくいと評価されたキーを入力しやすいと評価する被験者もいた。但し、r、v、u、m キーに限定すれば先の質問の回答も合わせた場合と全ての被験者が不満を持っていた。r、v、u、m キーの幅は 16mm であった。これとユーザの人差し指の幅を比較するためには、表 7.1 から表 7.3 の第二指爪基部幅を見ればよい。第二指爪基部幅は爪の基部における指の幅である。その値は被験者 A が 15.3mm（左手）と 15.1mm（右手）、被験者 B が 17.5mm（左手）と 16.8mm（右手）、被験者 C が 13.7mm（左手）と 15.4mm（右手）であった。この値と比較すると、r、v、u、m キーの幅である 16mm は被験者の指の幅にほぼ等しい。そのため、指の位置が少しでもずれるとキーから指がはみ出すので誤入力が起こりやすくなる。よって入力が困難になるのは順当な結果であり、ユーザに合わせてこれらのキーの幅を確保する必要性が感じられる。

8.3.13　Meyboard における手元注視に関して

被験者に Meyboard には手元注視を減らす効果があると思うか回答してもらった。5 段階のリッカート尺度における評価では、2 人が 4、1 人が 5 となった。それぞれのコメントを以下に要約する。

- キーが少ないため。
- キャリブレーションが可能なため、注視回数は減らせると思う。
- キーの数を減らした効果は大きいと思う。

キーの数を減らしたことや、キャリブレーションが可能なことから Meyboard には手元注視を減らす効果があると考える被験者が多かった。実際、手元を完全に隠した手元注視不可能条件においても、エラー修正の時間を必要としない実験設計であったとはいえ、手元注視可能条件とほぼ同等の入力率にて被験者はタスクを完了することができている。

8.3.14　Meyboard におけるアイズフリー入力に関して

被験者に練習次第では Meyboard においてアイズフリー入力が可能となると思うか回答してもらった。5 段階のリッカート尺度における評価では、2 人が 5、1 人が 3 となった。それぞれのコメントを以下に要約する。

- だんだん使えるようになったと感じた。
- 性能を度外視すれば可能であると思える。手元注視不可能条件にて思ったよりも出来たと思ったため。
- 実験が進むにつれてエラー率の改善と速度の向上があったため。

手元注視不可能条件のタスクにおいて、入力率とエラー率に概ね改善が見られた（但し被験者 A はエラー率が最初から低かった）ため、被験者は練習次第ではアイズフリー入力も可能であると考えたと思われる。但し、性能を度外視することを条件づけるユーザもいた。非常に慎重な入力を要するということであり、現状のエラー率の高さを考えると妥当な指摘である。

8.4　Meyboard の改良

実験の結果や得た知見を受けて、Meyboard に必要な改良点を考察した。改良を行った Meyboard を図 8.3 に示す。

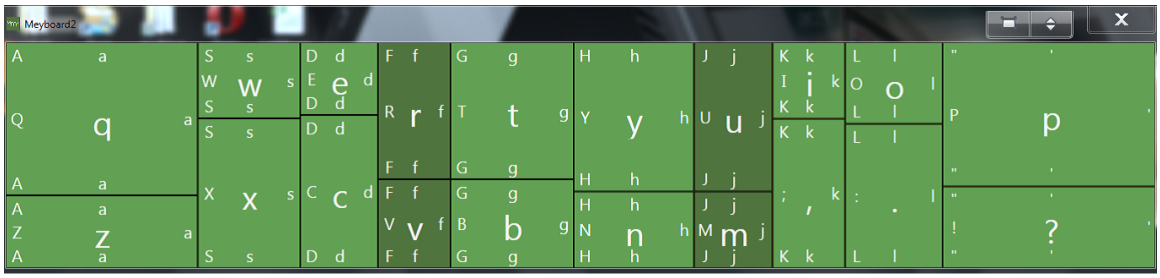

図 8.3: 改良を行った Meyboard

8.4.1　キャリブレーションにおけるキーの高さの設定

被験者からの要望が多かったため、キャリブレーション時においてキーの大きさを設定する際にキーの幅だけではなく高さも設定するようにする。具体的にはユーザの画面に置いた指の位置が上段のキーと下段のキーの境界となるようにする。しかし、ユーザの指の位置が Meyboard の画面において極端に上部あるいや下部であった場合にキーの高さが小さくなる。このとき、高さが小さくなったキーを入力することは困難になる。

そこでキーの高さが小さくなりすぎることを防ぐために、境界の位置に補正を加える。ユーザの指の位置から設定される上段と下段のキーの境界において、最も高い位置にある境界を

通る直線と、最も低い位置にある境界を通る直線を考える。両者の間が Meyboard の画面の高さ方向の中央に位置するように、全体の境界の位置を画面の高さ方向にずらす。以上によって境界の位置に補正を加え、ユーザの指が Meyboard の画面において極端に上部あるいや下部に置かれたとしてもキーの高さが小さくなりすぎることを防ぐ。

8.4.2 ユーザに合わせた r、v、u、m キーの幅の設定

実験においては、人差し指が入力を担当する r、v、u、m キーの幅が小さすぎたと述べた被験者が存在した。これらのキーの幅は固定値（16mm）であった。r、v、u、m キーの幅を固定値としたのは、それらの隣に位置し、かつ同じ人差し指によって入力する t、b、y、n キーに人差し指が届きにくくなることを避けるためであった。r、v、u、m キーの幅が大きくなると、t、b、y、n キーに人差し指が届くようにするためには人差し指を大きく動かす必要が生じるので指が届きにくくなる。一方で r、v、u、m キーの幅が 16mm では小さすぎたために入力が困難になったと主張する被験者が存在した。また、r、v、u、m キーの幅が固定値であったために、キャリブレーション時に人差し指の下がどのキーになるのか不定となることへの不満を主張する被験者も存在した。よって、r、v、u、m キーの幅も他のキー同様にユーザに合わせて設定する必要があると考えられる。

そこでキャリブレーション時のユーザの中指から、中指の担当するキーと人差し指の担当するキーの境界（ユーザの中指と人差し指の中間を通る、画面の高さ方向に平行な直線）までの距離の 2 倍の値が r、v、u、m キーの幅となるようにする。ユーザの手が大きいほど、キャリブレーション時の人差し指と中指の距離は離れると考えられる。上記の手法においては、人差し指と中指の距離が大きいほど r、v、u、m キーの幅が大きくなる。よって上記の手法はユーザの手の大きさに合わせて r、v、u、m キーの幅を設定することが可能である。

第9章　結論

本論文ではQWERTYベースのソフトウェアキーボードにおいて、ユーザの手元注視を減らすための設計について考案、検証した。そのために2種類のソフトウェアキーボードの実装を行った。それらのソフトウェアキーボードではユーザの指の位置にキーの位置と大きさを合わせることにより、物理的なフィードバックがないソフトウェアキーボードにおいてユーザの手元注視を減らすことを試みた。

最初に実装したLeyboardでは、物理キーボードにおける入力の待機状態において、ユーザが指を置くホームポジションキーをユーザの指の位置に合わせ、その周囲に他のキーを並べて配置することをキャリブレーションとした。また、手の形を維持したままのキー入力を可能とするために、親指基点スライドによる修飾キー入力時などにおけるキーの位置のユーザの手への補正や、親指スワイプ入力による1回のスワイプ操作によりキーの種類（キーセット）の変更と修飾キーの入力を同時に行う機能を設けた。

Leyboardは長期的な使用により、既存のソフトウェアキーボード以上の入力率を実現したが、手元注視を減らすことはできなかった。

次に実装したMeyboardでは、中段のキーを排除し、キーが格子状に並んだ設計にした。これにより、Leyboardにおいて頻出した中段のキーに起因する誤入力を減らし、さらに、キーの数が減ることによりユーザの手元注視が減っていても意図したキーの入力を可能にすることを試みた。Meyboardではユーザの各指間の距離からキーの位置や幅を決定することをキャリブレーションとした。排除された中段のキーの入力はフリック入力を割り当てることによって行うこととした。これにより物理キーボードのように待機状態からの入力が可能になった。このとき、ユーザは待機状態から指をフリックする、または上段か下段のキーをタップすることにより入力を行う。

Meyboardの入力率は手元注視の可不可によって変わるとは言えないことが判明した。しかし、Meyboardは手元注視を行わない場合エラー率が高くなった。但し、Meyboardは習熟につれてエラー率の改善と入力率向上の傾向があった。エラー率の改善の結果、その値が手元注視を行うときと同等にまでなるならば、Meyboardは手元注視を減らすソフトウェアキーボードと言える。Meyboardに手元注視を減らす効果があるかは今後さらに検証する必要がある。また、エラー内容の分析の結果、Meyboardにおいて物理キーボードのようにキーの位置を段ごとにずらすのではなく、キーを格子状に並べたことがエラーをもたらしたという仮説が提示された。しかしながら、Meyboardの設計においてはキーを格子状に並べたことにより誤入力を減らすことを試みたキーも存在した。キーの位置を段ごとにずらすとこの効果がなくなりかえってエラー率が高くなることが懸念された。そのため、エラー内容の分析や実験の被

験者のコメントから Meyboard の改良を行ったが、この改良においては Meyboard のキーは格子状に並べたままである。仮説検証のために、上記改良を加えた Meyboard と、キーの位置を段ごとにずらした Meyboard のエラー率を比較する必要がある。

謝辞

本研究を行うに当たっては、たくさんの方々による時間、金銭、精神的援助をいただきました。

指導教員の田中二郎先生にはキーボードにおける使用頻度の高いキーは上段に集中しているなど、Meyboard の設計を考えるうえで参考となる知見を提示していただきました。志築文太郎先生には Leyboard や Meyboard の実験設計をはじめ、本研究への数多くのご指導や助言を頂きました。修士論文を締めるところまで筆者が到達できたのは志築先生の御助力あってのことです。

また、筆者が Leyboard のデモや本研究の中間発表を行った際に、様々なご意見や知見をコンピュータサイエンス専攻の先生方から頂きました。深く感謝申し上げます。

インタラクティブプログラミング研究室の皆様にも、研究活動において様々な意見を頂きました。また被験者実験を行った際にもボランティアとしてご協力いただき、自身が忙しい身の上でありながらも、数日に渡り、総計 2 時間を超える実験に無償にて辛抱強く付きあってくださいましたことをお礼申し上げます。誠にありがとうございます。

筆者の父には研究をしていく上で、パラメータの決定法などいろいろと意見をいただきました。それだけに足らず、父としての立場からも生活や経済面において筆者を大いに支えていただきました。家族として同様に私を支えてくれた母も含めて、両親には常に感謝しています。

最後に、共に切磋琢磨した友人やその他、私の院生生活においてお世話になった様々な人々に感謝を告げて、謝辞を終わらせていただきます。本当にありがとうございました。

関連図書

[BCO+12] Xiaojun Bi, Ciprian Chelba, Tom Ouyang, Kurt Partridge, and Shumin Zhai. Bimanual gesture keyboard. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, UIST '12, pp. 137–146, 2012.

[CAP+12] Tayfur Coskun, Eva Artinger, Lorenzo Pirrilli, Daniela Korhammer, Amal Benzina, Claudia Grill, Andreas Dippon, and Gudrun Klinker. Gestyboard: A 10-finger-system and gesture based text input system for multi-touchscreens with no need for tactile feedback. In *Proceedings of the 10th Asia-Pacific Conference on Computer-Human Interaction*, APCHI '12, pp. 701–702, 2012.

[DS10] Liam Don and Shamus P. Smith. Applying bimanual interaction principles to text input on multi-touch surfaces and tabletops. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pp. 253–254, 2010.

[FLW12] Leah Findlater, Ben Lee, and Jacob Wobbrock. Beyond qwerty: Augmenting touchscreen keyboards with multi-touch gestures for non-alphanumeric input. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pp. 2679–2682, 2012.

[For86] Steven Fortune. A sweepline algorithm for voronoi diagrams. In *Proceedings of the second annual symposium on Computational geometry*, SCG '86, pp. 313–322. ACM New York, NY, USA, 1986.

[FS94] Masaaki Fukumoto and Yasuhito Suenaga. "FingeRing": a full-time wearable interface. In *Conference Companion on Conference of Human Factors in Computing Systems*, CHI '94, pp. 81–82, 1994.

[FW12] Leah Findlater and Jacob Wobbrock. Personalized input: improving ten-finger touchscreen typing through automatic adaptation. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pp. 815–824, 2012.

[GBAT99] Mikael Goldstein, Robert Book, Gunilla Alsiö, and Silvia Tessa. Non-keyboard QWERTY touch typing: A portable input interface for the mobile user. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, pp. 32–39, 1999.

[GE07] Kentaro Go and Yuki Endo. Catkey: Customizable and adaptable touchscreen keyboard with bubble cursor-like visual feedback. In *Proceedings of the 11th IFIP TC 13 International Conference on Human-computer Interaction*, INTERACT'07, pp. 493–496, 2007.

[Gen83] Donald R. Gentner. Keystroke timing in transcription typing. In William E. Cooper, editor, *Cognitive Aspects of Skilled Typewriting*, pp. 95–120. Springer-Verlag, 1983.

[GPM10] Asela Gunawardana, Tim Paek, and Christopher Meek. Usability guided key-target resizing for soft keyboards. In *Proceedings of the 15th International Conference on Intelligent User Interfaces*, IUI '10, pp. 111–118. ACM New York, NY, USA, 2010.

[GT10] Kentaro Go and Leo Tsurumi. Arranging touch screen software keyboard split-keys based on contact surface. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '10, pp. 3805–3810, 2010.

[HOY80] Yuzuru Hiraga, Yoshihiko Ono, and Hisao Yamada. An assignment of key-codes for a japanese character keyboard. In *Proceedings of the 8th Conference on Computational Linguistics*, COLING '80, pp. 249–256, 1980.

[JK09] Hyunjin Ji and Taeyong Kim. Clurd a new character-inputting system using one 5-way key module. In *Proceedings of the 13th International Conference on Human-Computer Interaction*, HCI International 2009, Part III, pp. 39–47, 2009.

[KBW08] Shaun K. Kane, Jeffrey P. Bigham, and Jacob O. Wobbrock. Slide rule: Making mobile touch screens accessible to blind people using multi-touch interaction techniques. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*, Assets '08, pp. 73–80, 2008.

[KIG07] Thomas Költringer, Poika Isokoski, and Thomas Grechenig. Twostick: Writing with a game controller. In *Proceedings of Graphics Interface 2007*, GI '07, pp. 103–110, 2007.

[KKKE11] Sungahn Ko, KyungTae Kim, Tejas Kulkarni, and Niklas Elmqvist. Applying mobile device soft keyboards to collaborative multitouch tabletop displays: design and evaluation. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '11, pp. 130–139, 2011.

[KSL+13] Sunjun Kim, Jeongmin Son, Geehyuk Lee, Hwan Kim, and Woohun Lee. Tapboard: Making a touch screen keyboard more touchable. In *Proceedings of the 2013 annual conference on Human factors in computing systems*, CHI '13, pp. 553–562, 2013.

[KST12] Yuki Kuno, Buntarou Shizuki, and Jiro Tanaka. Leyboard: A software keyboard that places keys at positions of fingers and their surroundings. In *Proceedings of the 10th Asia-Pacific Conference on Computer-Human Interaction*, APCHI '12, pp. 723–724, 2012.

[KST13] Yuki Kuno, Buntarou Shizuki, and Jiro Tanaka. Long-term study of a software keyboard that places keys at positions of fingers and their surroundings. In *Proceedings of the 15th International Conference on Human-Computer Interaction*, HCI International 2013, Part V, pp. 72–81, 2013.

[LGYT11] Frank Chun Yat Li, Richard T. Guy, Koji Yatani, and Khai N. Truong. The 1line keyboard a QWERTY layout in a single line. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pp. 461–470, 2011.

[MMR10] Adiyan Mujibiya, Takashi Miyaki, and Jun Rekimoto. Anywhere touchtyping: Text input on arbitrary surface using depth sensing. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pp. 443–444, 2010.

[MS03] I. Scott MacKenzie and R. William Soukoreff. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, pp. 754–755, 2003.

[New81] Allen Newell. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, pp. 1–55, 1981.

[OGN+11a] João Oliveira, Tiago Guerreiro, Hugo Nicolau, Joaquim Jorge, and Daniel Gonçalves. Blind people and mobile touch-based text-entry: acknowledging the need for different flavors. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*, ASSETS '11, pp. 179–186, 2011.

[OGN+11b] João Oliveira, Tiago Guerreiro, Hugo Nicolau, Joaquim Jorge, and Daniel Gonçalves. Brailletype: unleashing braille over touch screen mobile phones. In *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part I*, INTERACT '11, pp. 100–107, 2011.

[PSJ06] Morten Proschowsky, Nette Schultz, and Niels Ebbe Jacobsen. An intuitive text input method for touch wheels. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pp. 467–470, 2006.

[SCF+12] Caleb Southern, James Clawson, Brian Frey, Gregory Abowd, and Mario Romero. An evaluation of brailletouch: mobile touchscreen text entry for the visually impaired.

In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*, MobileHCI '12, pp. 317–326, 2012.

[SFM08] Hannah Slay, Greg Foster, and Edison Mukadah. Investigating the viability of scroll-wheel interfaced mobile phones for text entry. In *Proceedings of the 2008 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries: Riding the Wave of Technology*, SAICSIT '08, pp. 220–228, 2008.

[SLL11] Christian Sax, Hannes Lau, and Elaine Lawrence. LiquidKeyboard: An ergonomic, adaptive QWERTY keyboard for touchscreens and surfaces. In *Proceedings of the Fifth International Conference on Digital Society*, ICDS '11, pp. 117–122. XPS, 2011.

[坂村 86] 坂村健. BTRON における入力方式 － TRON キーボードの設計－. 情報処理学会研究報告ヒューマンコンピュータインタラクション（HCI）, Vol. 1986, No. 41, pp. 1–8, 1986.

[桜井 13] 桜井雄介, 増井俊之. Qwerty ソフトキーボード上のフリック日本語入力システム. 情報処理学会研究報告. HCI, ヒューマンコンピュータインタラクション研究会報告, Vol. 2013, No. 5, pp. 1–4, 2013.

[藤田 09] 藤田晋也, 赤池英夫, 角田博保. 運指情報と統計的手法を用いたウェアラブル機器向けの日本文入力手法の提案と評価. 情報処理学会研究報告. HCI, ヒューマンコンピュータインタラクション研究会報告, Vol. 2009, No. 28, pp. 9–16, 2009.

[片山 09] 片山拓也, 村尾和哉, 寺田努, 西尾章治郎. 片手用キーボードのための入力補完手法. 情報処理学会研究報告. HCI, ヒューマンコンピュータインタラクション研究会報告, Vol. 2009, No. 17, pp. 1–6, 2009.

付録

次ページ以降に、第 4 章にて用いた英文パングラムと、第 7 章の評価実験において被験者に配った実験同意書及びアンケートを載せる。

第 4 章にて用いた英文パングラム

- Cwm fjordbank glyphs vext quiz.
- Squdgy fez, blank jimp crwth vox!
- Jink cwm, zag veldt, fob qursh pyx.
- Junky qoph-flags vext crwd zimb.
- Cwm fjord veg balks nth pyx quiz.
- NBC glad. Quiz - Why? Fox PM TV jerks.
- Jumbling vext frowzy hacks PDQ
- PR flacks quiz gym: TV DJ box when?
- Zing, dwarf jocks vex lymph, Qutb.
- Zing, vext cwm fly jabs Kurd qoph.
- Blowzy night-frumps vex’d Jack Q.
- Dwarf mobs quiz lynx.jpg, kvetch!
- Fjord Nymphs XV beg quick waltz.
- Fjord q-klutz bahs given cwm pyx.
- Frowzy things plumb vex’d Jack Q.
- G.B. fjords vex quick waltz nymph.
- Glum Schwartzkopf vex’d by NJ IQ.
- Jerk gawps foxy Qum Blvd. chintz.
- JFK got my VHS, PC and XLR web quiz.
- Jocks find quartz glyph, vex BMW.
- J.Q. Vandz struck my big fox whelp.
- J.Q. Schwartz flung D.V. Pike my box.
- Jump dogs, why vex Fritz Blank QC?
- Mr. Jock, TV quiz PhD, bags few lynx.
- New job: fix Mr. Gluck’s hazy TV, PDQ!
- Quartz glyph job vex’d cwm finks.
- Quartz jock vends BMW glyph fix.
- The glib czar junks my VW Fox PDQ.
- TV quiz jock, Mr. PhD, bags few lynx.
- Nymphs blitz quick vex dwarf jog.
- DJs flock by when MTV ax quiz prog.
- Big fjords vex quick waltz nymph.
- Bawds jog, flick quartz, vex nymph.
- Junk MTV quiz graced by fox whelps.
- Bawds jog, flick quartz, vex nymphs.
- Waltz, bad nymph, for quick jigs vex!
- Fox nymphs grab quick-jived waltz.

- Brick quiz whangs jumpy veldt fox.
- Glib jocks quiz nymph to vex dwarf.
- Bright vixens jump; dozy fowl quack.
- Vexed nymphs go for quick waltz job.
- Quick wafting zephyrs vex bold Jim.
- Quick zephyrs blow, vexing daft Jim.
- Quick blowing zephyrs vex daft Jim.
- Sphinx of black quartz, judge my vow.
- Sex-charged fop blew my junk TV quiz.
- Jack fox bids ivy-strewn phlegm quiz.
- How quickly daft jumping zebras vex.
- Two driven jocks help fax my big quiz.
- “Now fax quiz Jack!” my brave ghost pled.
- Jack, love my big wad of sphinx quartz!
- Vamp fox held quartz duck just by wing.
- Five quacking zephyrs jolt my wax bed.
- The five boxing wizards jump quickly.
- Jackdaws love my big sphinx of quartz.
- My jocks box, get hard, unzip, quiver, flow.
- Kvetching, flummoxed by job, W. zaps Iraq.
- My ex pub quiz crowd gave joyful thanks.
- Cozy sphinx waves quart jug of bad milk.
- A very bad quack might jinx zippy fowls.
- Pack my box with five dozen liquor jugs.
- Few quips galvanized the mock jury box.
- Quick brown fox jumps over the lazy dog.
- The jay, pig, fox, zebra, and my wolves quack!
- Blowzy red vixens fight for a quick jump.
- Sex prof gives back no quiz with mild joy.
- The quick brown fox jumps over a lazy dog.
- A quick brown fox jumps over the lazy dog.
- Boxers had zap of gay jock love, quit women.
- Joaquin Phoenix was gazed by MTV for luck.
- JCVD might pique a sleazy boxer with funk.
- Quizzical twins proved my hijack-bug fix.
- The quick brown fox jumps over the lazy dog.
- Waxy and quivering, jocks fumble the pizza.
- When zombies arrive, quickly fax judge Pat.

- Heavy boxes perform quick waltzes and jigs.
- A quick chop jolted my big sexy frozen wives.
- A wizard’s job is to vex chumps quickly in fog.
- Sympathizing would fix Quaker objectives.
- Pack my red box with five dozen quality jugs.
- Quads of blowzy fjord ignite map vex’d chicks.
- Fake bugs put in wax jonquils drive him crazy.
- Watch “Jeopardy!”, Alex Trebek’s fun TV quiz game.
- GQ jock wears vinyl tuxedo for showbiz promo.
- The quick brown fox jumped over the lazy dogs.
- Woven silk pyjamas exchanged for blue quartz.
- Brawny gods just flocked up to quiz and vex him.
- Twelve ziggurats quickly jumped a finch box.
- My faxed joke won a pager in the cable TV quiz show.
- The quick onyx goblin jumps over the lazy dwarf.
- The lazy major was fixing Cupid’s broken quiver.
- Amazingly few discotheques provide jukeboxes.
- Foxy diva Jennifer Lopez wasn’t baking my quiche.
- Cozy lummox gives smart squid who asks for job pen.
- By Jove, my quick study of lexicography won a prize.
- Painful zombies quickly watch a jinxed graveyard.
- Fax back Jim’s Gwyneth Paltrow video quiz.
- My girl wove six dozen plaid jackets before she quit.
- Six big devils from Japan quickly forgot how to waltz.
- “Who am taking the ebonics quiz?”, the prof jovially axed.
- Why shouldn’t a quixotic Kazakh vampire jog barefoot?
- Big July earthquakes confound zany experimental vow.
- Foxy parsons quiz and cajole the lovably dim wiki-girl.
- Cute, kind, jovial, foxy physique, amazing beauty? Wowser!
- Have a pick: twenty six letters - no forcing a jumbled quiz!
- A very big box sailed up then whizzed quickly from Japan.
- Jack quietly moved up front and seized the big ball of wax.
- Few black taxis drive up major roads on quiet hazy nights.
- Just poets wax boldly as kings and queens march over fuzz.
- Bored? Craving a pub quiz fix? Why, just come to the Royal Oak!
- Grumpy wizards make toxic brew for the evil Queen and Jack.
- Crazy Fredericka bought many very exquisite opal jewels.
- Sixty zippers were quickly picked from the woven jute bag.

- The job of waxing linoleum frequently peeves chintzy kids.
- Back in June we delivered oxygen equipment of the same size.
- Just keep examining every low bid quoted for zinc etchings.
- How razorback-jumping frogs can level six piqued gymnasts!
- A quick movement of the enemy will jeopardize six gunboats.
- All questions asked by five watched experts amaze the judge.
- The wizard quickly jinxed the gnomes before they vaporized.

第 7 章における実験同意書及びアンケート

ソフトウェアキーボードの入力における精度と速度のデータ収集へ協力のお願い

文責：久野祐輝

この度は実験にご協力いただき、ありがとうございます。

本実験では、本研究において開発したソフトウェアキーボード（以下、Meyboard）に対して手元を隠した状態とそうでない状態の 2 通りの条件にて使用していただきます。それらを用いて行うタスクは、いずれの場合も、英字の入力となっています。

実験中の様子はビデオカメラにて録画させていただきます。

実験内容

まず練習として空白を含めた英字の入力を 10 回行っていただきます。これは意味のない文字列と英文の双方が含まれます。

次に英文を入力していただきます。英文を 10 回入力することを 1 セットとして、これを 2 セット行っていただきます。入力はできるだけ速く、かつ正確に行ってください。

以上を条件ごとに合計 2 回行います。セット間や条件間は休憩時間を自由に取ってよいものとします。

本実験は 3 日間に渡って行われます。それぞれの日において以上の内容を行っていただきます。

データの取り扱いに関して

本実験において得られた入力データ及び録画された動画は実験の分析、論文、学内外の発表において使用する可能性があります。その際は個人が特定できないように処理します。

実験に関して、上記内容を理解、同意した上で実験に参加していただける場合は、下部の署名欄に署名をお願いいたします。

平成　　年　　月　　日

所属　　署名

説明者：所属　システム情報工学研究科　コンピュータサイエンス専攻　署名

アンケート

1. 年齢と性別についてお答えください。
   年齢：_____歳　　性別：男・女　（該当する方を〇で囲んでください）

2. 熟練を「手元を全く見ずに素早い入力が可能である」、未習得を「手元を見ないと入力することができない」状態として、物理キーボードのタイピングの習熟度をお答えください。
   熟練　５・４・３・２・１　未習得　（該当する番号を〇で囲んでください）

3. タブレット端末（画面サイズ 7 インチ以上をそうとみなす）を所持（貸与含む）している場合、その機種をお答えください。端末を複数所持している場合は自分が最も用いる機種をお答えください。
   機種：_____・持っていない（未所持の場合〇で囲んでください）

4. 3 で答えた機種において自分が最も習熟していると思うソフトウェアキーボードをお答えください。3 の回答が「持っていない」の場合、本質問は飛ばしてください。
   QWERTY 配列・フリック入力・GODAN・その他：_____
   （該当するものを〇で囲んでください。その他の場合

5. 1 日あたりにおいて 4 の入力手段が搭載されている計算機を利用している平均時間をお答えください。3 の回答が「持っていない」の場合、本質問は飛ばしてください。
   _____時間

6. 熟練を「物理キーボードにほぼ等しい素早い入力が可能である」、未習得を「1 文字の入力に数秒かかるほど遅い」状態として、4 の入力手段の習熟度をお答えください。3 の回答が「持っていない」の場合、本質問は飛ばしてください。
   熟練　５・４・３・２・１　未習得　（該当する番号を〇で囲んでください）

7. Meyboard を使用した際の疲労感を評価してください。高いほど疲れやすいとします。
   評価：　高　５・４・３・２・１　低　（該当する番号を〇で囲んでください）

理由：

8. Meyboard の左手人差し指と中指のタップによって空白の入力を行う機能を評価してください。

   評価：　肯定　５・４・３・２・１　否定　（該当する番号を〇で囲んでください）

理由：

9. Meyboard のフリック入力によって中段のキーを入力する機能を評価してください。これに中段のキーの大文字の入力は含まれません。

   評価：　肯定　５・４・３・２・１　否定　（該当する番号を〇で囲んでください）

理由：

10. Meyboard のフリック入力によって中段のキーを入力する時に最もよく用いた方向を矢印にてお答えください。またそうなった理由もお答えください。

回答と理由：

11. Meyboard の左フリック入力によって上下段のキーの大文字を入力する機能を評価してください。

    評価：　肯定　５・４・３・２・１　否定　（該当する番号を〇で囲んでください）

理由：

12. Meyboard の左フリック入力とフリック入力を組み合わせることによって中段のキーの大文字を入力する機能を評価してください。

評価： 肯定 5・4・3・2・1 否定 （該当する番号を○で囲んでください）

理由：

13. Meyboard の左フリック入力とフリック入力を組み合わせることによって中段のキーの大文字を入力する時に最もよく用いた方向を矢印にて、理由も含めてお答えください。

回答と理由：

14. Meyboard のキャリブレーション機能を評価してください。

評価： 肯定 5・4・3・2・1 否定 （該当する番号を○で囲んでください）

理由：

15. Meyboard のキーの上に手を置きながら入力ができる機能を評価してください。

評価： 肯定 5・4・3・2・1 否定 （該当する番号を○で囲んでください）

理由：

16. Meyboard の入力する指を予測して入力するキーを決定する機能を評価してください。

評価： 肯定 5・4・3・2・1 否定 （該当する番号を○で囲んでください）

理由：

17. Meyboard のキー配列(位置、大きさなど)を全般的に評価してください。
評価： 肯定 5・4・3・2・1 否定 （該当する番号を○で囲んでください）

理由：

18. Meyboard において特に入力しやすいキーあるいは入力しにくいキーがあればお答えください

回答：

19. 本実験にて Meyboard をどの程度習得できたか、熟練を「手元を全く見ずに素早い入力が可能である」、未習得を「手元を見ないと入力することができない」状態として、習熟度をお答えください。
熟練 5・4・3・2・1 未習得 （該当する番号を○で囲んでください）

20. あなたは Meyboard は従来のソフトウェアキーボードよりも手元を見る回数を減らせると思いますか。お答えください。
評価： 肯定 5・4・3・2・1 否定 （該当する番号を○で囲んでください）

理由：

21. あなたは練習次第で手元を全く見ずに Meyboard を入力可能になると思いますか。お答えください。
評価： 肯定 5・4・3・2・1 否定 （該当する番号を○で囲んでください）

理由：

22. Meyboard を利用して他に、良かった点、改善すべき点があればお答えください。

良かった点：

改善すべき点：

アンケートは以上です。ご協力ありがとうございました。