筑波大学 情報学群 情報メディア創成学類

卒業研究論文

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開発と評価

久野 祐輝

指導教員　志築 文太郎　三末 和男　田中 二郎

2012年2月

概要

本研究ではマルチタッチが可能なタッチパネルにおいてタッチタイピングが容易なソフトウェアキーボードを作成する。従来のソフトウェアキーボードは、物理キーボードのように触覚によってキー位置を感知することが不可能であるためタッチタイピングが困難である。

本研究はキーをユーザの指の設置位置とその周囲に配置させることによって、タッチタイピングを容易にすることを試みる。ソフトウェアキーボードのキー位置を物理キーボードと同じにした場合、先にも述べたようにキーの感知が不可能であるため、意図したキーを入力しにくい。一方、キー位置がユーザの手の形状に適応していれば、キーの感知が不可能であっても、意図したキーが入力しやすくなると考えられる。

システムのプロトタイプを実装し、その評価実験を行った。結果、現状のシステムにおいては通常のソフトウェアキーボード以上の入力パフォーマンスを発揮することがなかった。そこでその原因を考察し、システムの改良を行った。

目 次

図 目 次

表目次

第1章　はじめに

本研究では、マルチタッチが可能なタッチパネル搭載端末を対象として、タッチタイピングが容易なソフトウェアキーボードを作成する。

まず、本論文で用いる用語を説明する。本研究では指を利用したディスプレイへのタッチ操作が可能である計算機をタッチパネル搭載端末と呼ぶことにする。また、本研究では画面の大きさが 9.5 インチ以上 20 インチ未満である端末をスレート端末と呼ぶ。よってスレート端末以上の大きさの端末は画面の大きさが 9.5 インチ以上であるものを指す。

スレート端末の例として Apple 社の iPad や富士通社の ARROWS Tab が挙げられる。スレート端末より大きな端末の例として NEC 社の X-info Table や SONY 社の VAIO L が挙げられる。また、タッチパネル搭載端末の中には複数のタッチを同時に入力するマルチタッチに対応したものがある。iPad や ARROWS Tab はマルチタッチに対応している。スレート端末以上の大きさのタッチパネル搭載端末はマルチタッチに対応した製品が多い。本研究ではマルチタッチに対応したスレート端末以上の大きさのタッチパネル搭載端末が対象となる。

これらの製品において、文字入力には主にソフトウェアキーボードが用いられている。本章ではソフトウェアキーボードの説明を行い、その問題点を挙げる。そして挙げられた問題点を改善する本研究の目的とアプローチを述べる。

1.1　ソフトウェアキーボード

ソフトウェアキーボードはスレート端末以上の大きさのタッチパネル搭載端末における主要な文字入力手法である。それは物理キーボードを模してキーとなるボタンを配置した文字入力システムである。タッチパネル搭載端末のユーザは QWERTY 配列の物理キーボードに慣れていることが多い。そのため、ソフトウェアキーボードのキー配置は主に QWERTY 配列に従っている。スレート端末以上の大きさのタッチパネル搭載端末は QWERTY 配列を表示するために十分な大きさを持つため、ソフトウェアキーボードが主要な文字入力手法となった。本研究では以降、ソフトウェアキーボードはタッチパネル搭載端末において用いられているものに限定する。

なお、スマートフォンのような画面の大きさが 7 インチ未満の小型端末においては、QWERTY 配列を表示するには画面が小さい。そのため、ソフトウェアキーボードとは別の入力手法が利用されることがある。その例としてはフリック入力や Swype 入力が挙げられる。

1.2　ソフトウェアキーボードの問題点

ソフトウェアキーボードは現状においては物理キーボードに匹敵する入力パフォーマンスを発揮しない。物理キーボードには高速かつ正確な文字入力方式であるタッチタイピングが存在する。キーの形状がきわめて近しいにもかかわらず、ソフトウェアキーボードが物理キーボードに比べて性能が劣るのは物理キーボードのようにタッチタイピングを行うことが困難であるためであると考えられる。

タッチタイピングは物理キーボードにおいて素早いキー入力を可能とする入力技法である。タッチタイピングは、指の起点位置としてホームポジションを設けている。ホームポジションは入力待機時に指を乗せる決まったキーのことを指す。QWERTY 配列におけるホームポジションの指とキーの対応を表 1.1 に示す。以降これらのキーをホームポジションキーと呼称する。なお、ホームポジションではないキーを非ホームポジションキーとする。ユーザはホームポジションキーを基準としてその位置から指を動かして入力を行うことによって特定のキーを特定の指で入力することが可能である。

表 1.1: ホームポジションの指とキーの対応

左手				右手			
小指	薬指	中指	人差し指	人差し指	中指	薬指	小指
A	S	D	F	J	K	L	;

QWERTY 配列における、タッチタイピングにおける指と入力するキーの対応を表 1.2 に示す。なお、表 1.2 に記載されていないキーの入力方法に関しては、スペースを親指が、アルファベットキーの周囲に配置されたキーを左右それぞれの小指が担当する。その他の矢印、Insert、Home、End、Page Up、Page Down 等のキーに対応する指が厳密に定められてはいない。

表 1.2: タッチタイピングにおける指と入力するキーの対応

左手					右手				
小指	薬指	中指	人差し指		人差し指		中指	薬指	小指
1	2	3	4	5	6	7	8	9	0
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	,	.	/

ソフトウェアキーボードにおいてタッチタイピングが困難になる原因は、ソフトウェアキーボードの各キーが個別に実体を持たないためである。よって、ソフトウェアキーボードのキーは物理キーボードのキーのように触って感知することが不可能である。ソフトウェアキーボードのキーは視認することによって、感知することは可能であるが、視認なしにホームポジションに指を合わせることは困難である。

1.3 目的とアプローチ

本研究ではスレート端末以上の大きさの端末を対象として、タッチタイピングが可能であるソフトウェアキーボードを開発する事を目的とする。そのために、以下の要件を満たす文字入力システムを開発する。

- キーが指の設置位置とその周囲に配置されている
- QWERTY 配列の物理キーボードを基としたデザインである

ソフトウェアキーボードはキー位置を動的にかつ自由に決定することが可能である。そこで、キー位置をユーザの指を設置した位置に配置することによって、タッチタイピングを容易にすることを試みる。ソフトウェアキーボードのキーは感知が不可能である。そのため、物理キーボードとキー位置を同じにした場合、意図したキーを入力しにくく、入力ミスが多くなる。一方、ユーザが指を設置する際の手の形状にキー位置が適応されていれば、キーの感知が不可能であっても意図したキーが入力しやすくなると考えられる。

マルチタッチを利用して、ユーザのそれぞれの指の位置を認識する。システム側がユーザの指先の位置を認識してそこをホームポジションキーの位置とする。ホームポジションキーの位置を基点に、他のキーを配置していく。

今後、タッチ点数に応じて○点タッチという呼称を用いる。例えば、左右 10 本の指先をタッチパネル面に設置した場合、それを 10 点タッチと呼称する。10 点タッチが行われた場合、ソフトウェアキーボードのキーを指の位置とその周囲に自動的に配置する方式をとる。

1.4 本論文の構成

第 1 章においては研究の背景となるソフトウェアキーボードを説明し、その問題点とそれを改善する本研究のアプローチを示した。第 2 章はタッチパネルにおける文字入力や、本研究と似たアプローチを取る文字入力システムに関する研究を挙げて本研究の位置づけを示す。第 3 章ではアプローチに基づいてシステムの設計を行う。第 4 章では実装したシステムに関して説明する。第 5 章では実装したシステムを用いて評価実験を行い、その結果を述べ考察を行う。第 6 章では評価実験の結果を受けて今後の課題を述べる。最後に第 7 章において本研究をまとめる。

第2章　関連研究

本研究はタッチパネル上における文字入力において、文字入力手法の一つであるタッチタイピングを適用可能なソフトウェアキーボードの開発を試みるものである。そこで、関連研究を分類し、ソフトウェアキーボードのキーの形状や位置を入力しやすいようにユーザに合わせる研究、タッチパネル以外の面上においてタッチタイピングを行う研究、及び少ない数のキーにおいて文字入力を行う研究のそれぞれについて述べる。

2.1　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]、BLT-Adaptive キーボード [遠藤 06]、Gunawardana の研究 [GPM10]、Go らの研究 [GT10] がある。

Sax らの LiquidKeyboard は、本研究と同様にユーザの 10 点タッチによって各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。LiquidKeyboard は、物理キーボードの入力時と同様に、ホームポジションキー上に指を待機させた状態から入力を行うことを想定している。つまり、ホームポジションキーはタッチされた状態となる。そのため、ホームポジションキーの入力方法を課題としている。ホームポジションキーの位置に指を待機させた状態から、入力するアルファベットに対応する指のみを動かして入力を行い、その後ホームポジションに指を戻す入力方法をとったとする。なお、指をホームポジションキーに戻すのはその指が担当するキーが連続した場合、それらがすべて入力された後とする。上記条件において “kilogram”を入力しようと意図した場合、ユーザは “ikolgrfmj”と入力を行うことになる。この誤入力を回避するために Sax らは 3 つの手法を考案した。

- 指のタッチ面積を取得し、それが一定以上大きくなるとキーが押されたと解釈し入力を行う。
- ホームポジションキーも指の設置位置の周囲に配置する。
- 辞書を用いて入力を意図した単語を解釈する

3 番目の手法に関して補足する。まずはあらゆるタッチ操作を入力と解釈する。次に辞書を用いて、ホームポジションへのタッチが入力であるかを判断する。その結果、例えば “ikolgrfmj” という入力があった場合、これを辞書と対応付けて “kilogram”と解釈する。

最終的に LiquidKeyboard は 1 番目のアプローチが一番妥当であると結論付けていた。しかし、本研究では指のタッチ面積が大きくなるように指を強く押し付ける入力方法は入力に時間がかかると判断し、キャリブレーション後にいったん指を浮かしたうえで入力を行うアプローチを採ることとする。

なお、LiquidKeyboard にはアルファベットと句読点、スラッシュのキーしか存在せず、数字や記号の大多数を入力する手段がない。また、この論文の時点においてはシステムに対して評価がされていない。本研究では、これらのキーを拡張し、かつシステムに対しての評価を行う。

遠藤らの BLT-Adaptive キーボードはタッチパネル面上のキーの形状をユーザの入力に応じて変化させ、ユーザの手の形状に合わせるアプローチを採っている。キーは一定の形状をしておらず、キーの中心座標（重心）からキーの領域を決定している。これはキーの形状及び配置の変更時にキー同士が重なることや、どのキーにも該当しない空白の領域が肥大化することを防ぐためである。システムはユーザが入力した位置を記憶し、その平均を取って新たなキーの重心としている。このアプローチによって、指の位置とキーの位置のずれを抑え、ずれを修正するためにキーを視認する回数を減らしている。また、触覚の代わりにバブルカーソル状の視覚的フィードバックを用いている。なお、本研究では既に設定されたキーの形状を徐々に変化させるのではなく、キャリブレーションを行うごとにキー配置を新たに設定し直している。

Gunawardana の研究では動的にキーの大きさを変動させている。大きさを変動させるキーを決定するために入力予測を行い、次に入力される確率が高いキーを大きく、低いキーを小さくする事によってユーザの誤入力率を低下させるアプローチをとっている。本研究ではタッチ位置に最も近く位置するキーを入力キーとする。よって、キーを配置する際の指の置き方によってキーの範囲が動的に決定される。

Go らの研究では、タッチする指に覆われた範囲から QWERTY 配列中における入力キーを複数選択する。ユーザは選択されたキーの中から入力するキーを改めて決定する。また、入力画面の大きさに応じて QWERTY 配列中の複数のキーを一つにまとめ、必要に応じて全体のキーの数を少なくしている。よって画面の大きさに因らずに意図するキーを選択可能にすることによって、誤入力を減らしている。本研究では画面の大きさをスレート端末以上に限定しており、また全体のキーの数が変動することはない。しかし、画面の大きさや指の運動特性を考慮し、誤入力を防ぐために特定のキーの入力が多段階の入力によって行われる点が共通している。

2.2 タッチパネル以外の面上においてタッチタイピングを行う研究

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、本研究と同様に物理キーが存在しない入力対象においてタッチタイピングの実現を目的とする研究として、Mujibiya らの研究 [MMR10] と Goldstein らの研究 [GBAT99] がある。

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。入力面をプロジェクター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。入力面上にはキーを示す視覚的な要素が表示されない。そのため、手元を見たとしてもキーの位置は分からない。本研究ではどの指によってタッチがなされたかの判断をキャリブレーション時における指のタッチ位置から行っている。

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。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。システムが知ることができるのは入力が行われた指に関しての情報のみである。タッチタイピングにおいて各指は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。例えば、左手中指、左手薬指、左手人差し指と入力がされた場合、最初の入力が {E, D, C} のいずれか、 2 番目の入力が {Q, A, Z} のいずれか、 3 番目の入力が {R, F, V, T, G, B} のいずれかである。ここから入力された単語の候補を作成し、最も確率が高いものを入力された単語と推定する。単語バイグラムやトライグラムを利用することにより、システムは最終的に単語を CAT と推定する。本研究ではキャリブレーション時の指の位置からタッチした指を判断しているほか、タッチ位置から入力されたキーを判断している。

2.3 少ない数のキーにおいて文字入力を行う研究

少ない数のキーにおいて文字入力を行う、本研究とは異なったアプローチをとっている研究として、Li らの The 1Line Keyboard[LGYT11]、Fukumoto らの FingeRing[FS94]、Szentgyorgyi らの研究 [SL07]、五味らの提案する Nbis 入力方式 [五味 08] がある。これらの研究はタッチパネル上における入力時においての誤入力を防ぐために、キーの数を減らしている。また、これらの研究の中にはタッチパネルを用いていないものもあるが、それらの研究内容自体はタッチパネル上における文字入力においても適用可能であるため本節にて述べる。

Li らの The 1Line Keyboard は表 1.2 において示した、QWERTY 配列のタッチタイピングにおける各指が担当するキー群をそれぞれ 1 つのキーにまとめたソフトウェアキーボードである。故に The 1Line Keyboard は 8 つのキーが一列に並んだ構成をしている。ただし、厳密には図 1.2 のキー群から数字を除いた構成になっている。ユーザは入力を意図した単語中に含まれた文字を内包するキーを入力していく。システムは入力されたキーから単語の候補を作成する。入力後、ユーザはフリック入力を行うことで入力する単語を選択、確定する。本研究では複数のキーセットを切り替えることによって入力を行うが、QWERTY 配列のキーの数を減らしてはいない。

Fukumoto らの FingeRing は Goldstein らの研究と同様に手に装着したデバイスを利用して文字入力を行う研究である。ただし、データグローブは用いずに、FingeRing と Fukumoto らが名づけた指輪状のデバイスをユーザのそれぞれの指にはめるものである。FingeRing には極

小の加速度計が内蔵されており、それを用いて指の動きを検知することによって入力を行う。また、Goldstein らの研究が両手にデータグローブを装着して QWERTY 配列に従って入力するものであるのに対して、FingeRing は片手のみに装着される。FingeRing は QWERTY 配列には従わずに、入力する指の組み合わせによって文字を決定する。

Szentgyorgyi らの研究も FingeRing と同様に片手のみを用いて文字入力を行う研究である。このシステムは入力する指の組み合わせによって文字を決定する点も FingeRing と同様であるが、こちらは指の組み合わせに加えて、長押しや素早い連続入力といったキーを入力するリズムも入力パターンとして用いている。

五味らの提案する Nbis 入力方式では、キーを押す時間によって入力内容を変えるアプローチをとっている。キーの数や入力する文字、機能の個数を任意に設定可能なモデルを五味らは考案した。例えば、日本語入力において、五味らは各キーに子音を対応させ、それらを押す時間によって母音を決定し入力を行う手法を提案している。

Fukumoto らの FingeRing、Szentgyorgyi らの研究、五味らの提案する Nbis 入力方式はキーの数の少なさという制約下において様々な文字の入力を可能とするために入力パターンの組み合わせを利用するアプローチをとっている。これらは高速かつ正確な入力を実現する可能性があるが、習熟に時間がかかる。この点が、QWERTY 配列に準じた設計を採り、ユーザの QWERTY 配列への慣れを利用する本研究と異なる。Li らの The 1Line Keyboard は少ないキー数と QWERTY 配列を組み合わせ、誤入力率の低さとユーザの慣れという、双方の利点を取り入れようとする研究であった。本研究の方が圧倒的にキー数は多いものの、この点において両者のアプローチは似通っている。ただし、本研究においては複数のキーを 1 箇所にまとめてはいるが、その選択が明示的な操作によって行われる点が異なる。

第3章　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

本章では第1章中のアプローチに基づいて、本研究において作成するソフトウェアキーボードの設計を行う。アプローチにおいて提示した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に必要な機能や入力方法をここで提案する。ここで提案された手法を取り入れたソフトウェアキーボードを以降、Leyboard と呼称する。

3.1　キャリブレーション

10 点タッチが行われた際に、各タッチ点がどの指によるものであるかを把握し、その位置にホームポジションキーを配置する。以降、このキーを配置する行程をキャリブレーションと呼称する。10 点タッチを行うことによって、ホームポジションキーの入力時にタッチパネル面上において指がタッチされる位置が分かる。

ホームポジションキーの周囲に非ホームポジションキーを配置する。手の角度に応じて適宜、ホームポジションキーを中心として非ホームポジションキーを回転する。非ホームポジションキーを回転させることによって、手の向きに関係なく、入力が行い易い位置にキーが配置される。

キーが配置された状態においても 10 点タッチを行った場合は再びキャリブレーションを行う。手の位置や入力姿勢が崩れたときに再キャリブレーションを行えば、再び入力に適したキー配置のもとで入力を続けることができる。

3.2　親指の周囲のキー配置

QWERTY 配列のタッチタイピングにおいて、親指はスペースキーを担当するが、Leyboard においては親指の周りにスペースキーの他、多数のキーを配置する。親指の非ホームポジションキーとして、Shift、Ctrl といった物理キーボードにおける小指の担当するキー及び Tab、Alt、矢印といったタッチタイピングにおいて対応する指が特に決められていないキーを配置した。親指周りに配置した理由を以下に述べる。前者に関しては、小指はその長さが他の指に対して短いので、タッチ入力が比較的不得手であるためである。後者に関しては他の指が担当するキーの数を増やすと、手元を見ずに入力することが困難になるためである。また、人差し指は 6 つのキーを担当しているためこれ以上担当キーを増やすのは不適当である。中指や薬

指は自身の左右に別の指が存在するために横の動きが不自由であり、キーを追加する場所として不適切である。故に、比較的キーの設置場所が空いており指を動かす自由度が高い親指を選択した。

3.3　キーの位置調整

本研究ではキーをユーザが指を設置した位置の周囲に配置し、タッチ位置に最も近いキーを入力するキーとする。ここで、より正確な入力を行うためには、それぞれのキーがユーザの指が届く範囲に存在しなければならない。そこで、ホームポジションキーを基準とした、非ホームポジションキーの位置を自由に調整可能にすることによってユーザが入力を行い易いキー位置の設定を可能にする。なお、以下において特に指定がなかった場合、タッチとは 10 点未満のタッチのことを指す。10 点タッチ時においてはキャリブレーションが優先される。

3.3.1　キーの位置補正

タッチ時にタッチ位置に距離が最も近いキーをタッチ点へ移動させる。指の移動量に応じたキーの間隔を反映する。

3.3.2　キーのドラッグ

1 点タッチ時にドラッグを行うことによってキーを任意の場所へ動かすことを可能とする。

ただし、各指が担当するキーは表 1.2 のタッチタイピングのキーと指の対応に常に従う。そのため、ホームポジションキーから離れた位置にキーを設置してもその位置はホームポジションキーの位置に依存する。

3.4　キーの振る舞いの分類

Leyboard では、キーが押されたときにその入力が行われるのではなく、キーが入力されるタイミングや入力のされ方はキーによって異なる。物理キーボードにおいても、全てのキーが押されたときに同じ振る舞いをするわけではないが、ほとんどのキーは押したままの状態にすることによって連続入力される。しかし、この振る舞いをそのまま適用すると 10 点タッチを行った場合において、入力とキャリブレーションが競合する。その結果キャリブレーション時に意図しない入力が発生する可能性がある。そこでほとんどのキーに対してはそのような競合が発生しないような振る舞いを適用する。ただし、入力の利便性を考えて一部のキーにはまた別の振る舞いを適用する。

3.4.1　指を離した時に入力するキー

タッチを行った指がタッチパネル面から離れた時にキーが入力される。ただし、タッチ時間が一定以上長い場合、そのキーの入力を行わない。他 2 つの振る舞いに該当しないキーの全てがこれに該当する。タッチ時間が一定以上長い場合は入力が行われないことを利用して、例えば両手の親指で手の位置を固定しながら入力が可能となる。この場合、親指に配置されているキーを入力する場合は一度親指をタッチパネル面から離す必要がある。指を離した時に入力を行うことによって、キャリブレーションの処理と入力の処理が競合しないようにする。

3.4.2　タッチを行った時に入力するキー

Ctrl、Alt、Shift キーが該当する。タッチを行った時点においてキーを押したとし、指がタッチパネル面から離れた時点においてキーを離したとする。これらのキーは他のキーと組み合わせて入力を行うことが多いため、このような振る舞いをする必要がある。

3.4.3　タッチを続けた時に連続入力するキー

キーが押されている限り、連続的にキーを入力し続ける。小指担当の Back Space、Delete、Enter キーが該当する。これらのキーは親指付近にも配置されているが、これらは誤入力を防ぐためにこの振る舞いをしない。親指付近に配置されているキーにおいても、例外的に矢印、Page Up、Page Down キーが該当する。ここで挙げたキーは連続入力されることが多く、単一入力を繰り返す形式をとると入力時に手に負荷をかける原因となる。よってこのような振る舞いをする必要がある。

3.5　キーセット

文章等を入力するにはアルファベットのキーのみではキーの種類が不足する。そこで複数のキーセットを用意し、それらを切り替えて用いる。

キーを指の設置位置とその周りに配置する本研究では、キーは整列した状態によって配置されない。故にキーが空間を占める領域が大きいので多数のキーを一度に表示できるだけの空間が存在しない。また、入力するキーの数を増やすと、ソフトウェアキーボードはキーを感知できないために誤入力が増大する。さらに、Leyboard はホームポジションキーの位置から相対的に他のキーの位置を決定するため、それぞれのキーの位置の計算に計算機のリソースを割くことになる。そのため、一度に描画するキーの数が多くなると、キーの位置の計算が完了するまでに時間がかかる。その結果、指の動きに処理が追従できなくなる。キーセットを切り替える以外に、一度に描画するキーの数を増やす方法も考えられるが、以上の理由からこれは妥当ではないと判断した。

また、キーの入力時、親指がタッチしているキーの位置に応じて人差し指から小指が担当するキーの位置が変更されるようにする。この機能は Shift キーなどを押しながらの入力を行

う場合や、後述するキーセットを切り替える機能を利用する場合に役に立つと考えた。その理由は以下のとおりである。物理キーボードにおいてはキーを触覚によって感知することが可能である。そのため、手の姿勢を例えば Shift キーを押す場合において崩したとしても元の姿勢に戻すことは容易である。しかし、ソフトウェアキーボードにおいてはキーを触覚によって感知することが不可能である。そのため、Shift キーを入力する場合においても手の姿勢を保持しなければ、元の姿勢に戻すことが困難である。よって人差し指から小指が担当するキー全体を動かすことによって手の姿勢を保持したまま Shift キーなどの入力を行えたほうが良いと考えた。

第4章　実装

第3章において述べた設計を基にソフトウェアキーボードLeyboardを実装した。本章では第3章において提案した機能を実現するために用いたタッチパネル搭載端末とアルゴリズムを述べる。また、キーセットに関しても実装した内容をここで述べる。

4.1　システム構成

4.1.1　使用デバイス

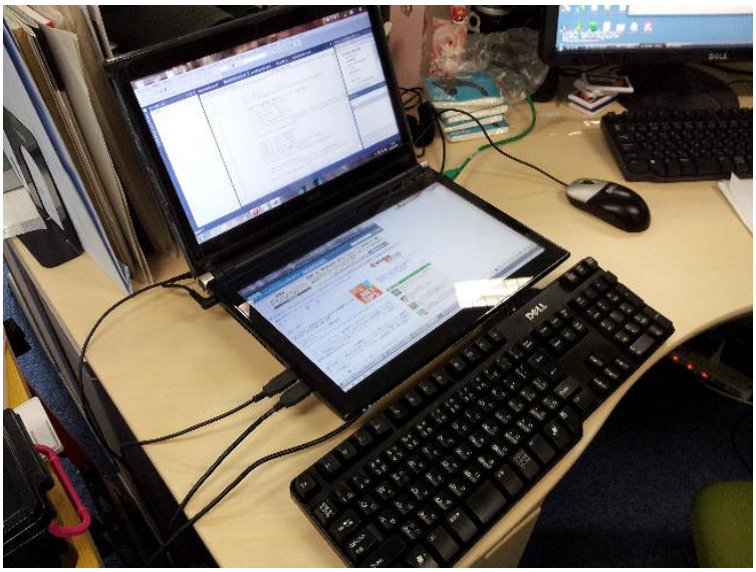

図 4.1: 使用デバイス

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を図4.1に示す。ICONIA-F54Eは両面が解像度1366×768のタッチパネルによって構成されたノートPCであり、上画面が10点までのタッチ入力に対応している。下画面も10点までのタッチ入力に対応しているが、5点以上のタッチ入力はICONIA-F54Eに用意されたランチャーおよびソフトウェアキーボードの利用に特化されているために、一般的なアプリケーションにおいては5点タッチ以上を認識させることができない。そのため、Leyboardは上画面において使用することが前提となった。

4.1.2 開発環境

Leyboard を開発する言語として C#を用いた。C#は.NET Framework 4/WPF4 の API を利用することによって、マルチタッチ対応アプリケーションを容易に作成可能である。そこで Leyboard は.NET Framework 4/WPF4 の API を利用して、WPF アプリケーションとして実装した。

4.1.3 画面位置と領域

Leyboard はキーの配置領域として ICONIA-F54E の片面全体を用いる。ただし、いくつかの機能の切り替えを行うインタフェースは別画面に表示する。

4.2 指の識別アルゴリズム

10 点タッチ時に各タッチ点が対応する指を識別するアルゴリズムを実装した。

4.2.1 処理の行程

以下の行程によって指の位置を識別する。

1. 10 点のタッチ点群を X 座標の値によってソートする。
2. タッチ点群を始めの 5 点と後ろの 5 点に分ける。始めの 5 点を左手、後ろの 5 点を右手とする。
3. それぞれの 5 点において、Y 座標の値が最も大きい点を探す。これを親指の点とする。
4. 親指の点を除いた点が人差し指から小指の点に相当する。基本的に X 座標の値のソート結果から、値の低い順に左手は小指から、右手は人差し指から当てはめていく。ただし、中指の Y 座標の値が人差し指の Y 座標の値を超えると判断された場合は、Y 座標の値が大きい方の点を人差し指とする。

行程の 4 番目において中指と人差し指の Y 座標に注目した理由は以下の通りである。中指の Y 座標の値が人差し指の Y 座標の値を超える場合、人差し指よりも中指の方が親指に近い位置になる。これは通常の手の置き方においてあり得ない。そのため、この場合は Y 座標の値が大きい方の点を人差し指と考えるのが妥当である。

4.2.2　指識別アルゴリズムを適用可能なケースと適用可能でないケース

手の置き方としては様々な場合が想定されるが、その中において入力時の手の置き方としてはあり得ないものが存在する。手の追い方として考えられる各場合を検証し、指識別アルゴリズムを適用可能でないケースは全て入力時の置き方としては想定する必要のないものであることを示す。

1. 手を画面に対して垂直に置いた場合

   図 4.2 のような置き方が最も一般的である。この場合、座標点を X 座標の値によってソートすれば指の識別が可能になる。

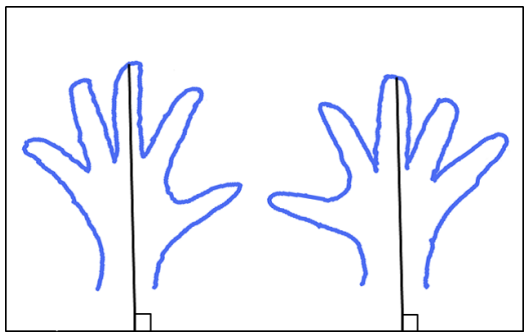

図 4.2: 手を画面に対して垂直に置いた場合

2. 手を画面に対して内側に傾けて置いた場合

   図 4.3 のように手を置いた場合、Y 座標の値を考慮する必要が生じる。左手に注目すれば、人差し指の X 座標の値が親指の X 座標の値を超える可能性があるため、X 座標の値のみからでは指を識別できない。親指は全ての指において最も下に位置するので、Y 座標の値から識別する。また、中指の X 座標の値も人差し指の X 座標の値を超える可能性がある。そこで人差し指の候補と中指の候補の中において、Y 座標の値が大きい方を人差し指とする。何故ならば、人差し指は中指よりもユーザの手前に位置するからである。

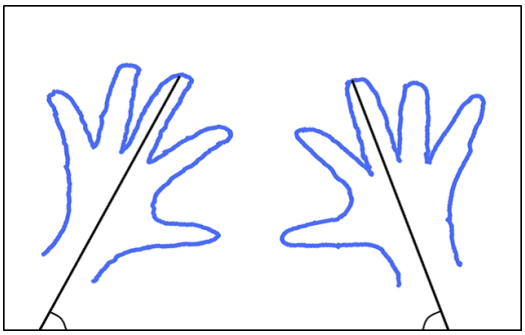

図 4.3: 手を画面に対して内側に傾けて置いた場合

3. 手を画面に対して真横に置いた場合

　手の内側への傾きがさらに急になり、図 4.4 の置き方のように、手の置き方が完全に真横になった場合、指識別アルゴリズムは対応できない。この場合、人によっては薬指の X 座標が人差し指の X 座標を超えてしまいアルゴリズムが破たんする。この時、手を置いた人間の姿勢は手を真横に向けるために肘を前方に突き出す形になる。これは非常に負担がかかる姿勢であり、キーボードを使用する状況としては現実的ではない。よってこのような状況は想定しない。

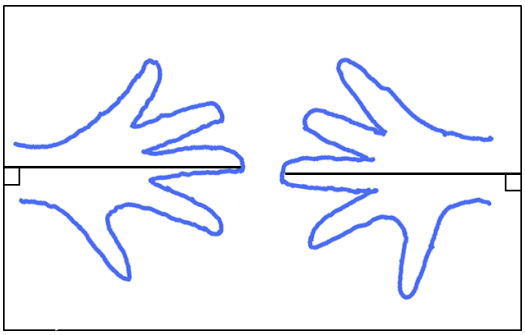

図 4.4: 手を画面に対して真横に置いた場合

4. 手を画面に対して逆さまに置いた場合

　手を画面に対して真横に置いた場合を想定していないため腕をさらに傾けて、図 4.5 のように手が逆さまになった場合も想定していない。もっとも、このような体勢によって入力を行うユーザはいないと思われるので問題はないと判断した。

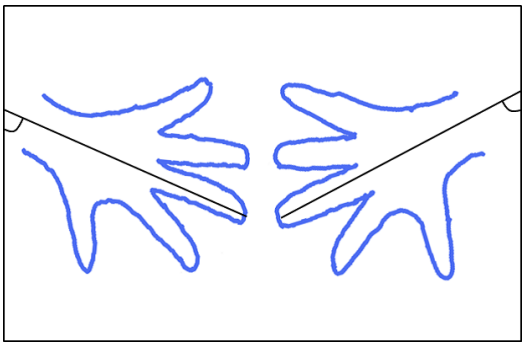

図 4.5: 手を逆さまに置いた場合

5. 手を画面に対して外側に傾けて置いた場合

　図 4.6 のような置き方である場合にも対処できない。この場合、Y 座標の値が最も大きくなるのは小指の可能性があり、人差し指の Y 座標の値が中指の Y 座標の値を下回る可能性があるためである。しかし、この場合を排除してもさほど問題はないと考えている。何故ならば、これは手首を外側に回転させる体勢であり、ユーザに負荷を強いる。よって、ユーザがこの体勢によって操作を行うことが現実的でないためである。

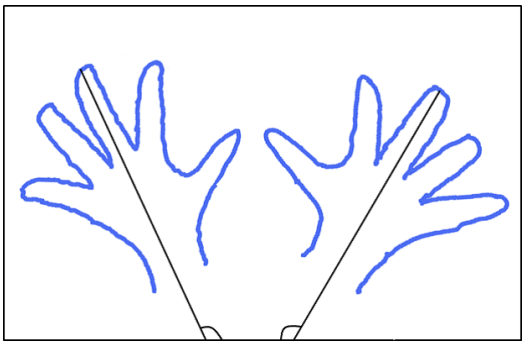

図 4.6: 手を画面に対して外側に傾けて置いた場合

6. 左右の手を交差して置いた場合

　この置き方の場合にも対処できない。しかし、両手を交差させて物理キーボードを入力するユーザは全くいないか、いたとしても非常に少数であると思われるので、問題はないと判断した。

4.3　キー位置の決定法

10 点タッチ時にホームポジションキーを配置し、その位置を基準として周囲に非ホームポジションキーを格子状に配置する。親指周辺のキーは指が届きやすいように円周上にキーを配置した。

4.3.1　キーの相対座標と絶対座標

各キーは絶対座標と相対座標をそれぞれパラメータとして持つ。絶対座標はキーのウィンドウ上の位置を意味する。相対座標はキーが対応する指のホームポジションキーの位置を基準とした、相対的な指の位置に関する情報である。なお、ホームポジションキーの相対座標は原点 (0,0) である。

初期設定では、非ホームポジションキーはホームポジションキーを基準に 50 ピクセル間隔において格子状に配置される。キーの間隔を 50 ピクセルとしたのは、1 辺 50 ピクセルの正方形領域が著者の人差し指の腹の領域を収めるのに十分な大きさであったためである。親指の担当するキーに関しては周囲に多数のキーが存在するため、入力ミスを防ぐために 75 ピクセルをキーの間隔とした。

また、各キーは担当する指をパラメータとして所持している。これは 0 から 9 の整数によって表され、左手小指から左手親指を 0 から 4 とし、右手親指から右手小指までを 5 から 9 としている。このパラメータの値を基に基準となるホームポジションキーの位置を取得する。

4.3.2　親指を基準としたキーの相対座標

全てのキーはホームポジションキーを基準とした相対座標を持つ。ホームポジションキーの内部においても親指の担当するホームポジションキーの位置を基準とした、人差し指から小指のホームポジションキーの相対座標を設けた。これを利用してキーの入力時、親指が存在するキーの位置に応じて人差し指から小指が担当するキーの位置が変更されるようにした。

4.3.3　手の傾き補正

キーの配置時に人差し指と小指の位置から手の傾きを計算し、キーの配置に反映させる。この時に手を回転させると、手の回転に合わせて非ホームポジションキーの位置も回転する。

片方の手に置いて人差し指と小指のタッチ点を結ぶ直線がタッチパネルの長辺に対して平行であった場合の傾きを 0 rad として、左手の小指の座標を (L_{lx}, L_{ly})、左手の人差し指の座標を (L_{ix}, L_{iy})、右手の人差し指の座標を (R_{ix}, R_{iy})、右手の小指の座標を (R_{lx}, R_{ly}) とすると、左手の回転角度 (rad) は

$$tan^{-1}(\frac{L_{iy} - L_{ly}}{L_{ix} - L_{lx}}) \tag{4.1}$$

右手の回転角度 (rad) は

$$tan^{-1}(\frac{R_{ly} - R_{iy}}{R_{lx} - R_{ix}}) \tag{4.2}$$

となる。

4.3.4 キー配置の手順

キーの配置時における座標計算から実際のキーの配置までの手順を以下に示す。

1. 10 点タッチを行う
2. 各指の位置を確定する
3. 左右の手それぞれに対して人差し指と小指の位置から手の傾きをそれぞれ計算する
4. 全てのキーに対して以下の処理を行う
   (a) 相対座標を読み込む
   (b) 相対座標を手の傾きの角度分回転させた座標を計算する
   (c) そのキーと同じ指が担当するホームポジションキーの絶対座標に上記の座標を加えた座標をキーの絶対座標とする
   (d) 算出した絶対座標の位置にキーを配置する

4.3.5 相対位置関係の破壊/非破壊

キーに対する位置補正やドラッグ操作を行った場合に、それぞれホームポジションキーと非ホームポジションキーとの相対位置関係の破壊の有無を設定できるようにする。

キーの位置補正時

相対位置関係破壊機能がオンの場合 (図 4.7)、非ホームポジションキーを補正する場合は該当する非ホームポジションキーのみを移動させ、ホームポジションキーとの相対座標を更新する。手の傾きによる影響を相対座標の値から取り除くために、更新される相対座標には、現在の相対座標に対して手の傾きの分の角度を逆回転させた座標が適応される。図 4.7 の例では、B キーの相対座標が変更された。それに伴い B キーの位置のみが移動した。

相対位置関係破壊機能がオフの場合 (図 4.8)、非ホームポジションキーを補正する場合は該当する非ホームポジションキーの他にそのキーを入力する指に対応するホームポジションキーと非ホームポジションキーの全てを移動させ、相対位置関係を保つ。図 4.8 の例では、B キーと同じ指が担当するホームポジションキーの絶対座標のみが変更された。それに伴いキー全体が移動した。

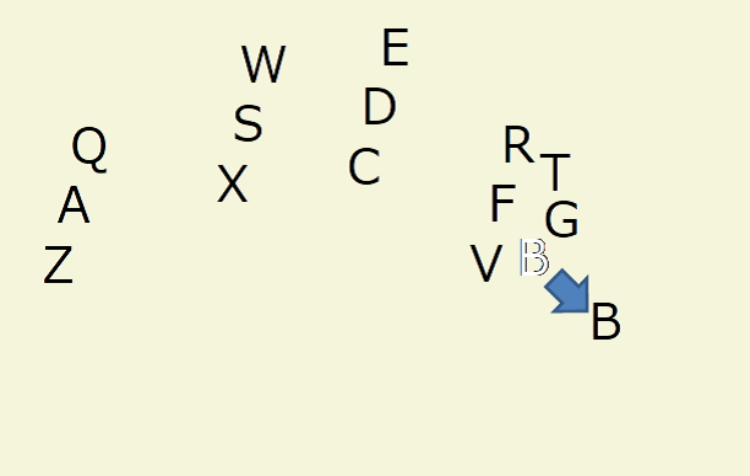

図 4.7: 相対位置関係破壊機能がオンの場合 (キーの位置補正)

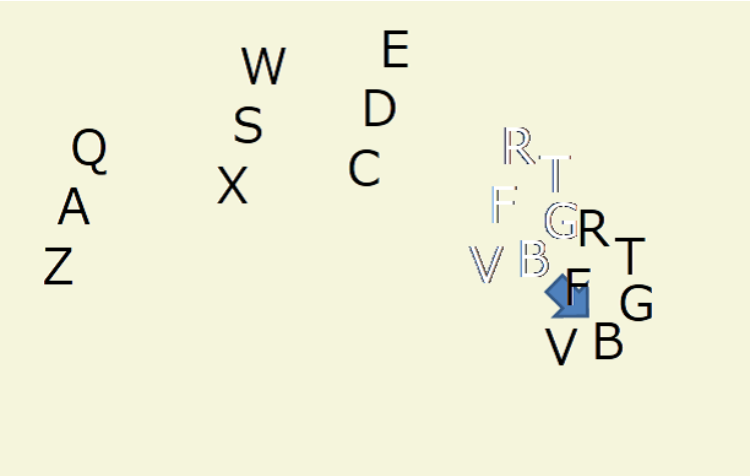

図 4.8: 相対位置関係破壊機能がオフの場合 (キーの位置補正)

ホームポジションキーの補正に関してはいずれの場合もホームポジションキーだけではなく非ホームポジションキーの位置も同時に変更される。よってホームポジションキーと非ホームポジションキーの相対関係は維持される。これはホームポジションキーの補正を行う場合に相対位置関係を破壊すると、補正を行うたびにキーの位置関係が大きく変わるためにかえって使いにくくなると判断したためである。これを変更するにはキーのドラッグを行えばよい。

キーのドラッグ時

相対位置関係破壊機能がオンの場合 (図 4.9)、該当するホームポジションキーのみを移動させ、そのキーを入力する指が担当する全ての非ホームポジションキーの相対座標を更新する。手の傾きによる影響を相対座標の値から取り除くために、更新される相対座標には、現在の相対座標に対して手の傾きの分の角度を逆回転させた座標が適応される。これはキーの位置補正時と同様である。図 4.9 の例では、ホームポジションキーである F キーの絶対座標と、全ての非ホームポジションキーの相対座標が変更された。それに伴い、F キーの位置のみが移動した。

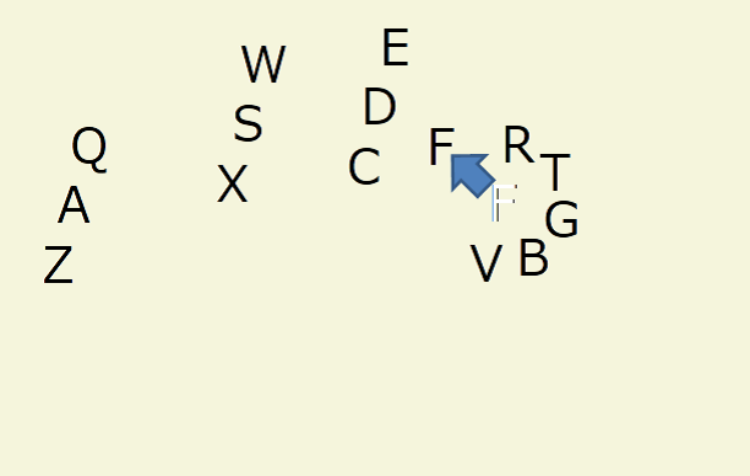

図 4.9: 相対位置関係破壊機能がオンの場合 (キーのドラッグ)

相対位置関係破壊機能がオフの場合 (図 4.10)、該当するホームポジションキーの他にそのキーを入力する指に対応する非ホームポジションキーの全てを移動させ、相対位置関係を保つ。図 4.10 の例では、ホームポジションキーである F キーの絶対座標のみが変更された。それに伴いキー全体が移動した。

キーの補正に関してはいずれの場合も該当する非ホームポジションキーのみを移動させ、ホームポジションキーとの相対座標を更新する。つまり相対位置関係は破壊される。これは相対位置関係が破壊されない場合、ホームポジションキーをドラッグする場合と挙動が全く同じになるため意味がないと判断したためである。キー全体を移動させたいならば相対位置関係破壊機能をオフにしてホームポジションキーをドラッグすればよい。

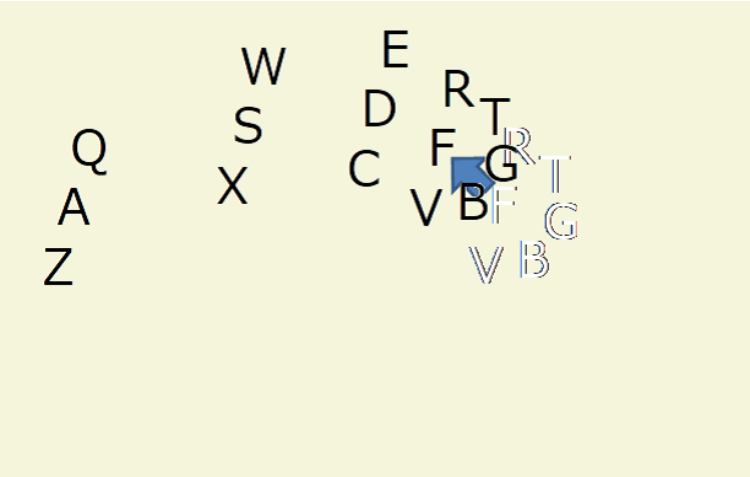

図 4.10: 相対位置関係破壊機能がオフの場合 (キーのドラッグ)

4.4 各キーセットにおいて使われるキー

文字セット、数字・記号セット、ファンクション・テンキーセットの 3 つのキーセットを用意した。Caps Lock キー、2 つ存在する Windows キー、無変換キー、変換キー、カタカナひらがなローマ字キー、メニューキーは他のキーを利用することによって機能的に代替可能であるか、OS に依存するキーである。そのため、いずれのキーセットにおいても用いなかった。

Leyboard は 1 つのキーセット当たり 52 のキーを持つ。このうちスペースや Enter キーのようにどのキーセットにおいても使われるキーを考えると、自由に入れ替えが効くキーは 34 である。現在主流である日本語入力用の物理キーボードは 109 キーボードであり、これはキー数が 109 存在する。ここから先に述べた Leyboard に用いなかった 7 つのキーを引くとキー数は 102 になる。$\frac{102}{34} = 3$ であるので、3 つのキーセットは 102 のキー数を補うために妥当な数であると言える。

文字セット

文字セットの配置を図 4.11 に示す。キーの並びは QWERTY 配列のアルファベット部分に沿っている。

数字・記号セット

数字・記号セットの配置を図 4.12 に示す。キーの並びはキーボード配列上段の数字キーの配置を基にしている。キーボード配列上段の数字キーに対応しない記号に関しては右手が主に担当するようにキーの配置を決定した。図 4.12 においてはアルファベット等の QWERTY 配列のキーが一部混ざっているが、これは数字・記号セットのキーの個数が少なく、そのため全てのキーを置き換える必要がないためである。このような場合、以前のキーの配置が残される。ファンクション・テンキーセットから数字・記号セットに切り替えた場合も同様である。

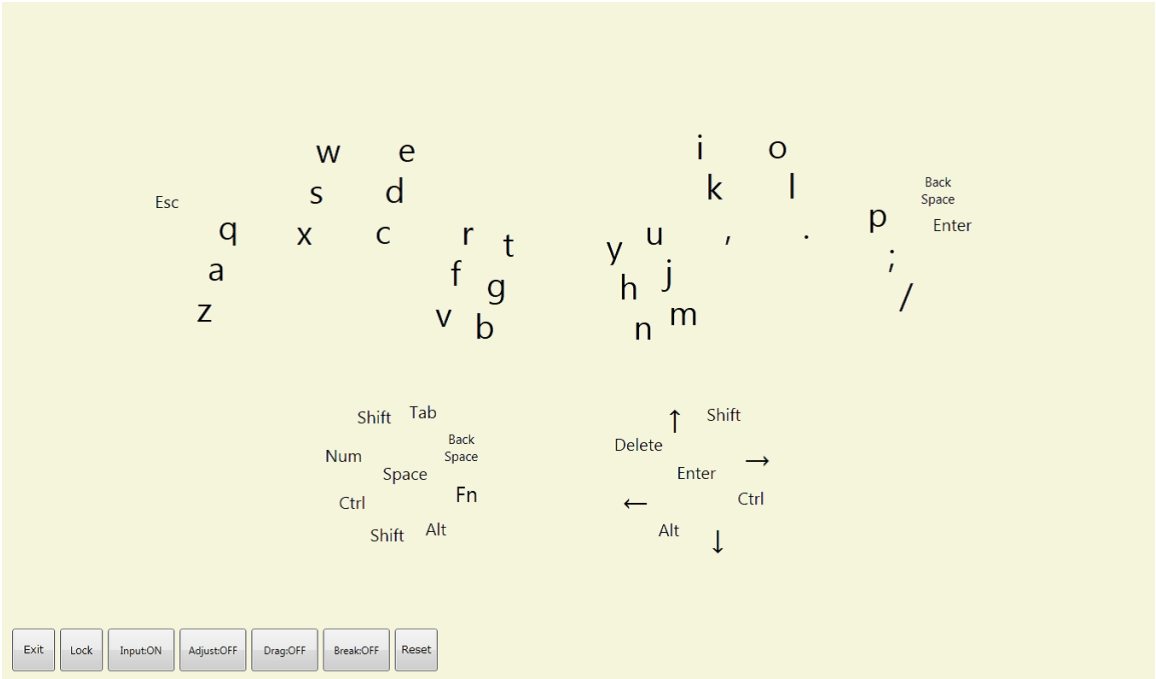

図 4.11: 文字セット

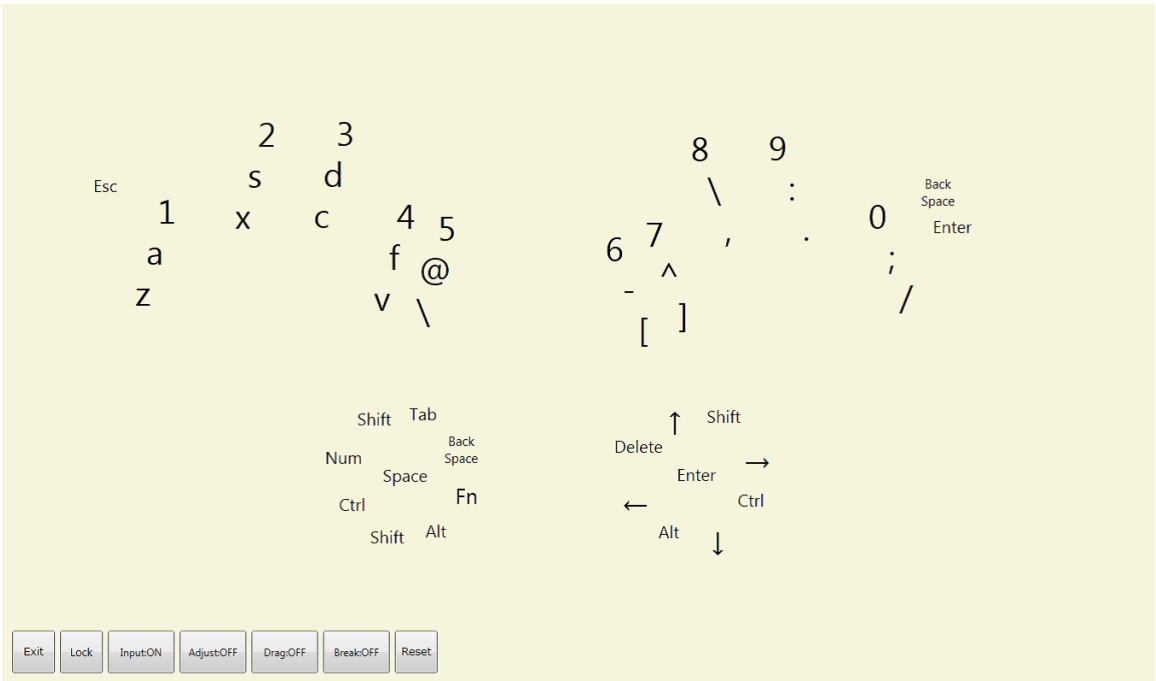

図 4.12: 数字・記号セット

ファンクション・テンキーセット

ファンクション・テンキーセットの配置を図 4.13 に示す。キーボードの右側にはテンキーの配置を基に数字キーを、左側には F1 から F12 までのキーを配置した。また、他キーセットに存在しなかったキーを割り当てた。

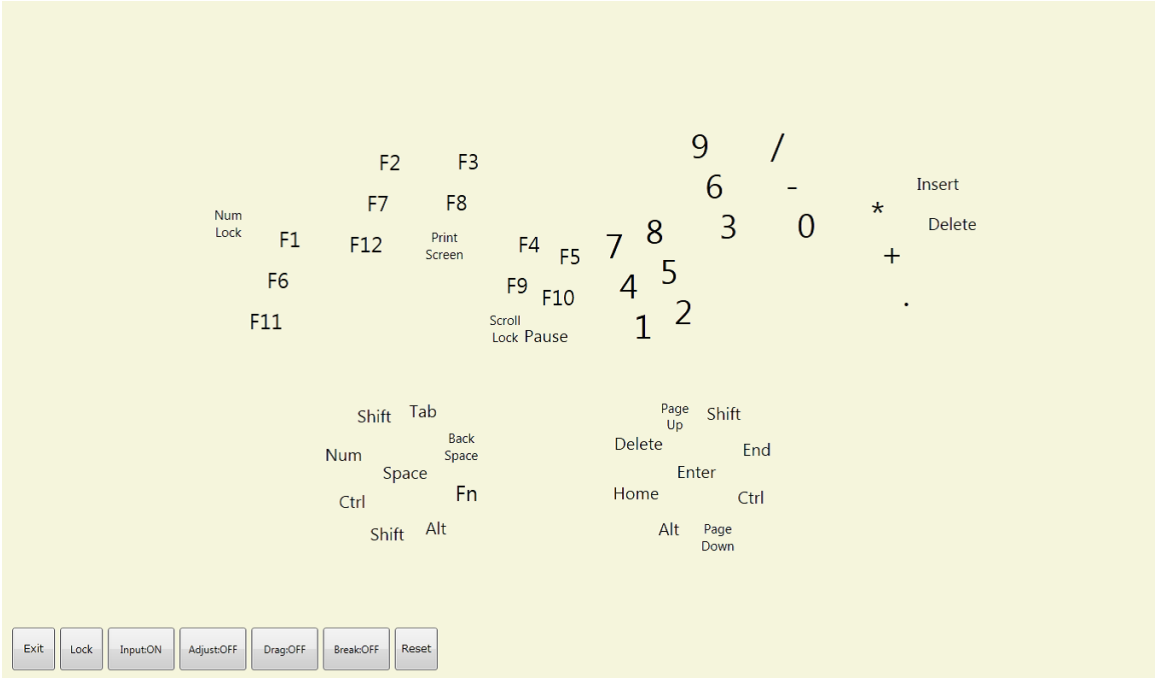

図 4.13: ファンクション・テンキーセット

4.5 キーの入力

4.5.1 入力の仕組み

タッチ点を中心とした正方形領域の範囲内に存在するキーに対してタッチ点との距離を算出し、その中において最も距離が近かったキーをテキストエディタ等の外部アプリケーションへの入力の対象とする。該当するキーが存在しなかった場合は入力を行わない。

外部アプリケーションへのキーの入力は WindowsAPI の関数である SendInput 関数を利用して行った。SendInput 関数はキーストローク、マウスの動き、ボタンのクリックなどのイベントを合成してアプリケーションに送ることが可能である。Leyboard においては SendInput 関数を用いてキーストロークをイベントとして送ることにした。

WindowsAPI には指定されたウィンドウを作成したスレッドをフォアグラウンドにしてからそのウィンドウをアクティブにする SetForegroundWindow 関数が存在する。これを利用して

キー入力時に入力対象のウィンドウをアクティブにする。Leyboard に対するタッチ操作を行った場合、Leyboard がアクティブになってしまうためキーストロークを外部アプリケーションに送ることができない。そのため、入力が行われるごとに SetForegroundWindow 関数を用いて外部アプリケーションをアクティブにしている。

4.5.2 キー入力の手順

Leyboard のシステム中の入力に関係するメソッド (以下、入力メソッド) が呼び出されてから、キーが入力されるまでの手順を以下に示す。

1. 外部アプリケーションのプロセスを取得する
2. SetForegroundWindow 関数によって外部アプリケーションのプロセスをアクティブにする
3. SendInput 関数によって入力対象にキーストロークを送信する
4. 外部アプリケーション上においてキーが入力される

4.6 キーセットの切り替え

各種キーセットを切り替えるためのキーを用意した。Space キーを入力すると文字セットに、Num キーを入力すると数字・記号セットに、Fn キーを入力するとファンクション・テンキーセットに切り替わる。該当するキーセットに既に切り替わっている場合、これらはスペースキーとして機能する。また、親指の担当するキーに限り、指をあるキーから他のキーへスライドさせると、スライドした先に存在するキーの入力が可能である。これによってキーセットを切り替えた上に Shift キーを押す行為が一連の動作によって実行可能になる。

4.7 音声フィードバック

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音声フィードバックを設けた。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、システムは入力音を発してユーザにこれを伝える。

4.8 システム中の各ボタンの機能

Leyboard に設けた各種機能の干渉や誤作動を避けるために、ボタンを設けてオン/オフを切り替えられるようにした。システム中のボタン群を図 4.14 に示す。

各ボタンの機能は以下のようになっている。

- システムを終了する (図 4.14a)

図 4.14: ボタン一覧

- ウィンドウのドラッグ機能の使用のオン/オフを切り替える (図 4.14b)
- 入力機能のオン/オフを切り替える (図 4.14c)
- キーの位置補正機能のオン/オフを切り替える (図 4.14d)
- キーのドラッグ機能のオン/オフを切り替える (図 4.14e)
- 相対位置関係破壊機能のオン/オフを切り替える (図 4.14f)
- キーの配置を初期設定の状態に戻す (図 4.14g)

ウィンドウ内の任意の位置をマウスまたはタッチによってドラッグするとウィンドウを動かすことができる機能を備えている。この機能が常に働いていると、キーの配置や位置の調整、入力時にこの機能が干渉するので図 4.14b のボタンによって機能のオン/オフを切り替えられるようにした。

図 4.14g のリセットボタンを押すとキーが消滅し、キーが配置されていない起動直後の画面に戻る。その後キャリブレーションを行った場合、キーの配置も初期設定の状態に戻っている。

第5章　評価実験

本章では第4章において実装したLeyboardの評価実験について述べる。Leyboardを被験者に使ってもらい、入力速度やエラー率の評価を行った。その結果を述べ、Leyboardの有用性に関して考察する。

5.1　実験目的

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を収集することによって、既存のソフトウェアキーボードとLeyboardの性能の比較を行う。

本実験では、既存のソフトウェアキーボードとして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。

5.2　Windows 7キーボードの仕様

実験において留意すべきWindows 7キーボードの特徴について述べる。

Windows 7キーボードのキーはそのほとんどがタッチを続けた時に連続入力を行う振る舞いをする。また、物理キーボードとは異なり、Windows 7キーボードのShiftキーは一度入力されると次のキーが入力されるまで入力状態が固定される。これはマルチタッチに対応していないタッチパネル端末においてもソフトウェアキーボードを用いるための工夫であると考えられる。

Windows 7キーボードにおいてはキーの大きさがWindows 7キーボードのウィンドウの大きさに応じて変化する。

5.3　実験内容

被験者に英文パングラムを入力させた。パングラムとは特定の言語における記号を除いた全ての文字を少なくとも1回以上用いて構成した文章である。英語においてはAからZまでの全てのアルファベットを少なくとも1回以上用いて構成した文章になり、例えば“A quick brown fox jumps over the lazy dog.”がパングラムである。パングラムを入力対象とすることによって、被験者はスラッシュ(/)やアットマーク(@)などの一部キーを除き、文字セット中のほぼ全てのキーを入力することになる。

本実験ではそれぞれの内容が異なる英文パングラムを 10 回入力することを 1 セットとし、被験者に各ソフトウェアキーボードにおいて 6 セットの入力を行わせた。このうち、最初の 2 セットは練習セットとした。

1 セットの中において、120 文用意したパングラムの中からランダムに 10 文を選択し、被験者に提示した。パングラムの文中には大文字小文字の使い分けがなされており、記号を含むものも存在する。パングラム中に含まれていた記号は、エクスクラメーション (!)、ダブルクオーテーション (”)、ピリオド (.)、カンマ (,)、コロン (:)、セミコロン (;)、アポストロフィ (’)、クエスチョン (?) である。1 文中の文字数は空白や記号を含めて 31 文字から 63 文字の範囲である。

被験者に対しては入力をできるだけ速く、かつ正確に行うように指示した。なお、セットの間に被験者は休憩時間を自由に取ってよいものとした。

実験条件を統一するために Windows 7 キーボードの位置を変更することは許可したが、その大きさは固定した。今回は、実験に用いたタッチパネル搭載端末の画面の横幅と同じ大きさにした場合に、アルファベットのキーの形状が正方形になるように、Windows 7 キーボードのウィンドウの大きさを調整した。キーの 1 辺の大きさは 75 ピクセルであった。

Leyboard のキーの位置補正機能とドラッグ機能は使用しなかった。これは初期設定のキー配置の妥当性を検証するためである。

5.4 実験手順

被験者を甲グループと乙グループの 2 つのグループに分けて、それぞれ以下のタスクを行わせた。

5.4.1 甲グループ

1. Windows 7 キーボードを使用して入力を 6 セット行う
2. Leyboard を使用して入力を 6 セット行う
3. アンケートの記入を行う

5.4.2 乙グループ

1. Leyboard を使用して入力を 6 セット行う
2. Windows 7 キーボードを使用して入力を 6 セット行う
3. アンケートの記入を行う

被験者を 2 つのグループに分けて、タスクを異なった順番で行わせたのは、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。

5.5 被験者

被験者は大学生 5 名、大学院生 5 名の合計 10 名のボランティアからなる。年齢は 21 歳から 24 歳の範囲であった。性別は男性が 8 名、女性が 2 名であった。実験後に行ったアンケートによると、全員が文字入力システムの中においては QWERTY 配列の物理キーボードに最も慣れていると答えている。また、同様に実験後に行ったアンケートにおいて被験者に自身のタッチタイピングの習熟度を、5 を「手元を全く見ずに素早い入力が可能である」、1 を「手元を見ないと入力することができない」状態として、5 段階のリッカート尺度によって評価させると、5 と答えた被験者が 1 人、4 と答えた被験者が 5 人、3 と答えた被験者が 2 人、2 と答えた被験者が 2 人であった。3 段階目以上と答えた被験者が全体の 8 割を占めるので、彼らはほとんどが平均以上のタッチタイピング技能を持っていると考えられる。

被験者を無作為に 5 名ずつ甲、乙グループにそれぞれ分けた。

5.6 実験環境

デバイスには実装と同じく ICONIA-F54E を用いた。なお、ICONIA-F54E は両面がタッチパネルのノート PC であるが、下画面はその仕様上、自作したシステムに対しては 4 点までのタッチにのみ反応する。Leyboard は 10 点タッチを認識可能な端末においてのみ動作する。ICONIA-F54E は上画面が 10 点までのタッチを認識可能であるため、Leyboard は上画面においてのみ動作する。そこで上下の画面をそれぞれ回転させ、PC 本体も上下を逆にして配置した。

ソフトウェアキーボードはいずれの場合も画面上部に表示させた。PC 本体の配置が上下逆になっているので被験者には画面下部にキーボードが存在するように感じられる。実験環境の光景を図 5.1 に示す。

評価実験を行うにあたって、実験用のアプリケーションを用意した。このアプリケーションの内部には 2 つのテキストボックスが存在し、それぞれ同じ例文が表示されている。被験者が入力を行うと、片方のテキストボックス中の例文から既に入力された文字が消えていく。大文字小文字の表記や空白、記号の入力を表示された文章の通りに行わない限り、被験者は次の文字を入力できない。全ての文字を入力し終わると、アプリケーションは次の例文を表示する。また、このアプリケーションは正解と間違いのそれぞれの入力数及び所要時間を把握し、10 文の入力が終了した時にファイルに記録するようになっている。さらに、入力されたキーのログを取るようにもなっており、この内容もアプリケーションの終了時にファイルに書き出す。

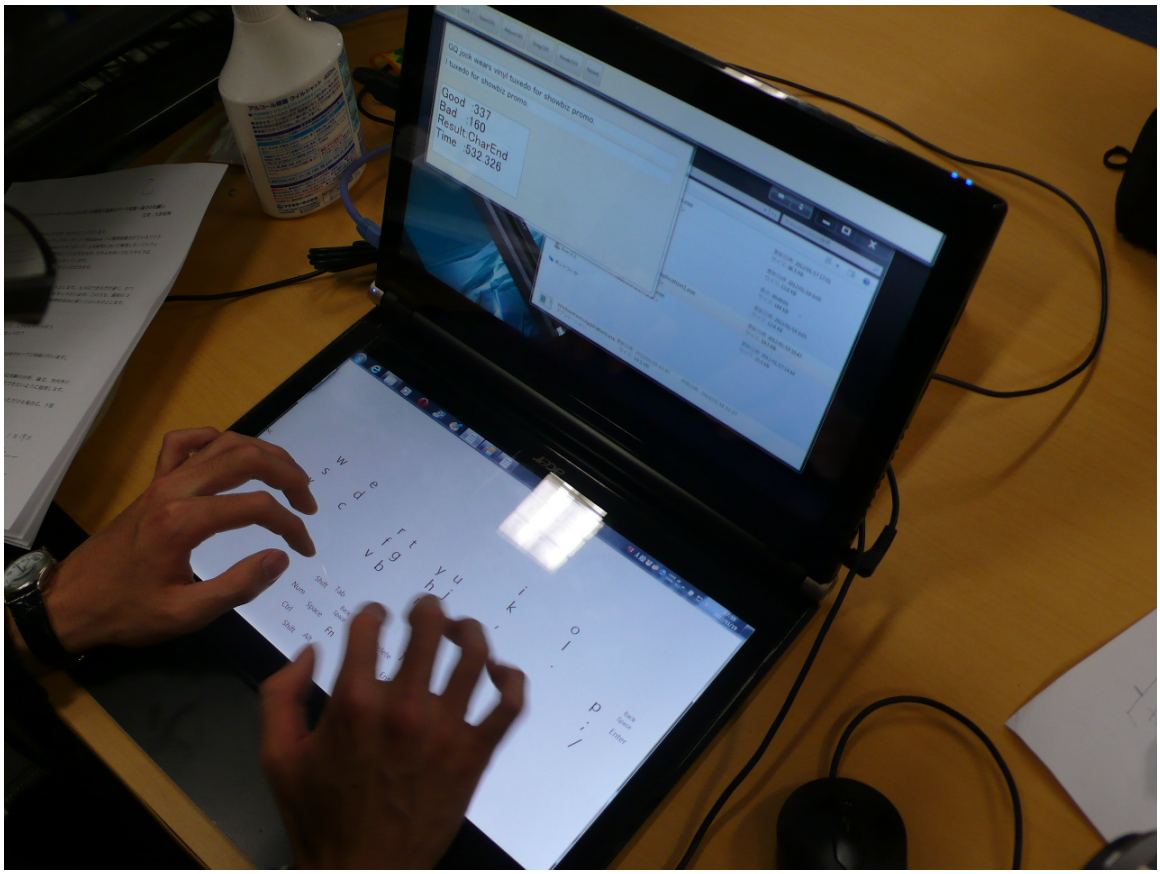

図 5.1: 実験環境

実験用アプリケーションは画面下部に表示された。PC 本体の配置が上下逆になっているので被験者には画面上部にアプリケーションが表示されているように感じられる。

5.7 結果

5.7.1 入力率

被験者が各セットにおいて入力する文章内容はそれぞれ異なるので、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。そこで、入力率を比較する。入力率とは単位時間あたりに入力された単語数を表す値である。Gentner は空白も含めて 1 分間あたりに入力されたキーストローク数を 5 で割った値を 1 分間に入力された英単語数即ち words per minute(wpm) とみなした [Gen83]。本実験ではこの wpm を入力率として、ソフトウェアキーボードごとに比較する。なお、1 分間あたりに入力されたキーストローク数は

$$\frac{\text{全体のキーストローク数}}{\text{入力時間 (分換算)}} \tag{5.1}$$

によって求められる。

各ソフトウェアキーボードにおける平均入力率を図 5.2 に示す。平均的に Windows 7 キーボードの入力率の方が Leyboard に対して高い。

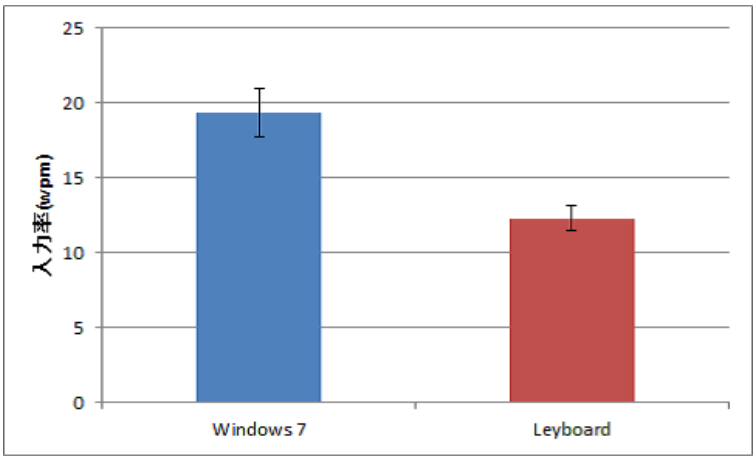

図 5.2: 各ソフトウェアキーボードにおける平均入力率

各ソフトウェアキーボードにおける平均入力率の変動を図 5.3 に示す。Windows 7 キーボードに上昇傾向は見られないが、Leyboard には上昇傾向がみられる。

各ソフトウェアキーボードにおける被験者別の平均入力率を図 5.4 に示す。t 検定を行ったところ、両条件の平均の差は有意であった (両側検定: $t(9) = 14.4, p < .001$)。したがって、Windows 7 キーボードの方が Leyboard よりも入力率が高いと言える。

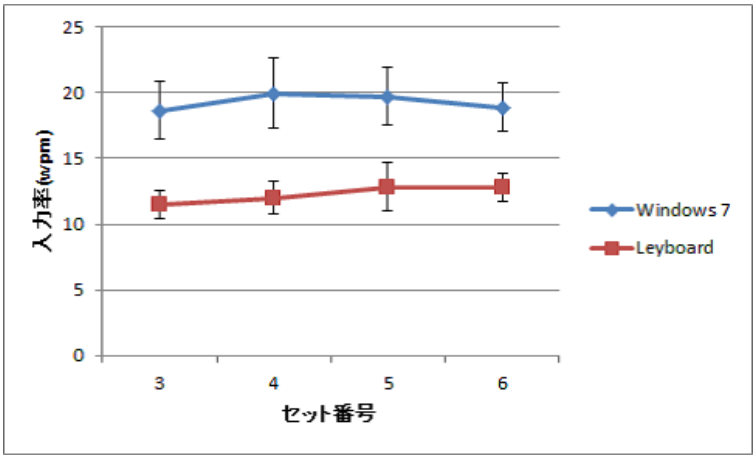

図 5.3: 各ソフトウェアキーボードにおける平均入力率の変動

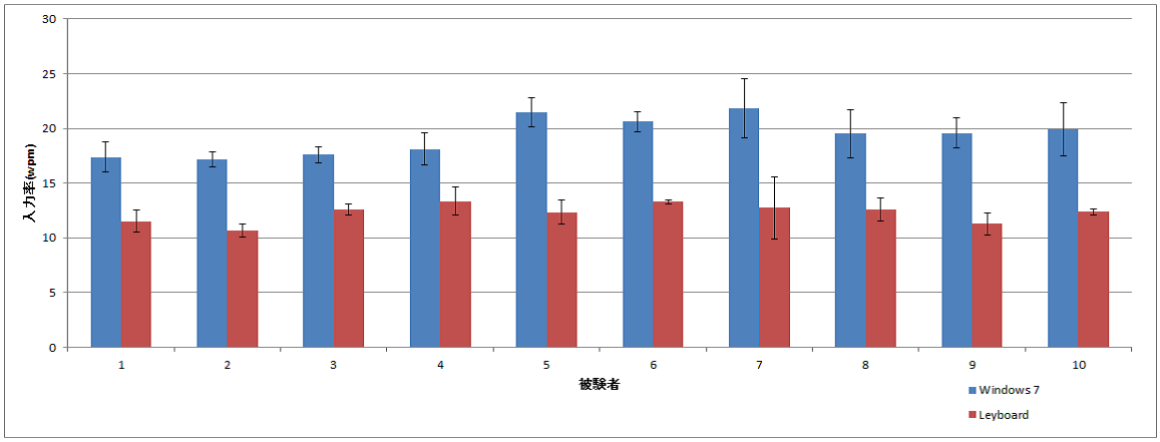

図 5.4: 各ソフトウェアキーボードにおける被験者別の平均入力率

5.7.2　エラー率

エラー率は百分率によって表す場合、

$$\frac{\text{誤入力を行った回数}}{\text{全体のキーストローク数}} \times 100 \tag{5.2}$$

によって求められる。ここではこの百分率で表したエラー率をソフトウェアキーボードごとに比較する。

各ソフトウェアキーボードにおける平均エラー率を図 5.5 に示す。平均的に Leyboard のエラー率の方が Windows 7 キーボードに対して高い。

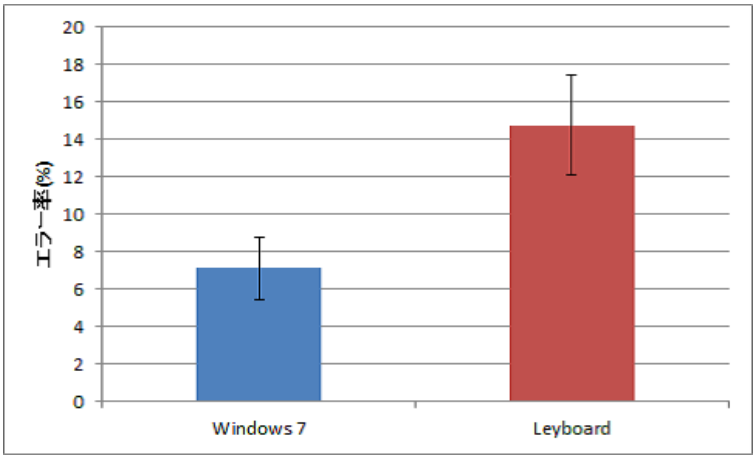

図 5.5: 各ソフトウェアキーボードにおける平均エラー率

各ソフトウェアキーボードにおける平均エラー率の変動を図 5.6 に示す。Leyboard のエラー率は Windows 7 キーボードに比べて減少する傾向にあった。

各ソフトウェアキーボードにおける被験者別の平均エラー率を図 5.4 に示す。t 検定を行ったところ、両条件の平均の差は有意であった (両側検定: $t(9) = -8.15, p < .001$)。したがって、Windows 7 キーボードの方が Leyboard よりもエラー率が低いと言える。

5.7.3　アンケート

実験終了後に被験者に対して行ったアンケートの結果を以下にまとめる。

Windows 7 キーボードを使用した際の疲労感の評価

この項目における肯定的コメントとは、疲労感が少ないと評価した人によるものである。

- 肯定的コメント

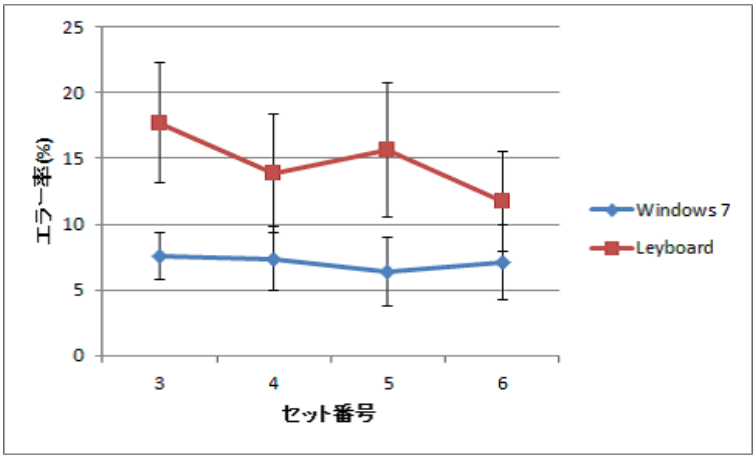

図 5.6: 各ソフトウェアキーボードにおける平均エラー率の変動

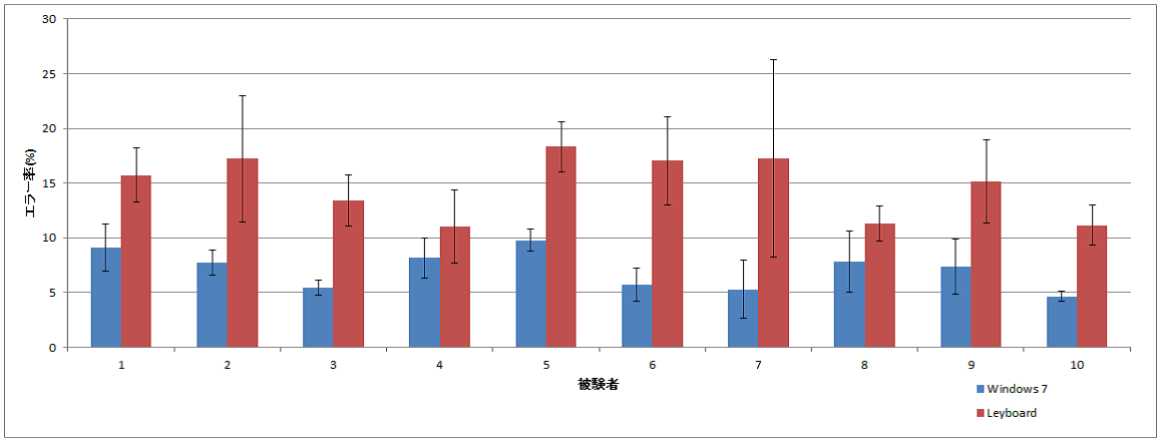

図 5.7: 各ソフトウェアキーボードにおける被験者別の平均エラー率

  - ボタンが大きいので視認しやすい。
  - 位置が固定されているので安定して入力を行える。

- 否定的コメント

  - 普段使っているキーボードと配列が異なるので慣れない。
  - 常に指先を浮かせておく必要があり、疲れる。
  - スペースキーが入力しにくい。
  - ホームポジションの位置が見ないと分からない。
  - キーを入力したつもりでも実際は入力されていなかった。
  - 物理キーボードのように指先では入力できず、指の腹を用いて入力する必要がある。

Leyboard を使用した際の疲労感の評価

先の項目と同様に、この項目における肯定的コメントとは、疲労感が少ないと評価した人によるものである。

- 肯定的コメント

  - 指の動きが自然である。

- 否定的コメント

  - 慣れるのに時間がかかった。
  - キーを探すのに時間がかかった。
  - 親指のキーを入力する際に、手全体を動かさなければならない。
  - 手によってキーが隠される。
  - 自分が意図するキーを入力しにくい。
  - A、P、F、J キーの入力に失敗する確率が高い。
  - キーの間隔が狭い。
  - 入力の成否が分かりにくい。
  - 目的のキーの位置以外触れることができない。
  - キーの位置を自由に決めることができても、入力が進むにつれて手の位置がずれていくため意味がない。
  - 手を広げて浮かせる姿勢は疲れる。

Leyboard の親指が存在するキーの位置に応じて人差し指から小指のキーの位置が変動する機能の評価

- 肯定的コメント
  - タッチタイピングには必要であると感じた。
  - 手の位置に対応してキーボードが出てくるのは良いと思った。
  - 機能自体は良い。
  - 入力しやすい。
- 否定的コメント
  - キーを視認する際に戸惑う。
  - 自分にとって入力しやすかったキーの位置が変動してしまうために入力しにくくなる。
  - 文字に逃げられる。
  - Shift キーを親指で入力するため、B、V、N、M キーが入力しにくい。
  - 手の形が変わるため入力がかえって難しくなる。
- その他のコメント
  - 良くも悪くも気にならなかった。

Leyboard の親指を固定しながらによるキーの入力機能の評価

実装としては、指を離した時に入力するキー全てに設けられている機能であるが、アンケートでは親指とした。なお、有効な回答は 1 つのみであり、被験者の中では 1 人のみがこの機能を利用した。

- 肯定的コメント
  - なし。
- 否定的コメント
  - 変な姿勢を強いられる。

Leyboard の親指スライドによるキーの入力機能の評価

- 肯定的コメント
  - 動作自体は心地よい。
  - キーセットの変更の際に、それを行っている実感がある。
  - Shift キーを入力する際に Windows 7 キーボードの Shift キーの仕様よりも使いやすい。
- 否定的コメント
  - ミスが多発した。
  - 親指だとミスが多いので人差し指を使うこともあった。
  - キーの数はもっと少ないほうが良い。
  - 疲れる。
  - 使い方を考える必要がある。
  - 指を設置した時の手の形と、Shift キーを入力した時の手の形は異なると思う。

Windows 7 キーボードのキー配列の全般的な評価

- 肯定的コメント
  - 慣れた配列なので使いやすい。
  - 一部キーのタッチタイピングが行い易い。
  - キーの場所が固定されているために入力しやすい。
- 否定的コメント
  - キーが小さい。
  - スペースキーが小さい。
  - 普段使用しているコンピュータが Apple 社の Mac であるため慣れない。
  - 慣れていないために見ながら出ないと入力ができない。
  - Shift と A キー、Ctrl と Alt と Caps Lock キーを間違えることがよくあった。
  - Caps Lock キーが邪魔である。

Leyboard のキー配列の全般的な評価

- 肯定的コメント
  - キーの横の間隔は広いのでいくつかのキーは視認せずに入力が可能であった。
  - 運指を行い易い。
  - 配列自体は良い。
- 否定的コメント
  - 小指の外側に位置するキーが指の設置位置によっては画面外に配置される。
  - ホームポジションキーが入力しにくい。
  - キーの縦の間隔が狭い。
  - ホームポジションキーと非ホームポジションキーの間隔をもっと広げてほしい。
  - キーセットの切り替えが負担である。
  - 記号の入力がしにくい。
  - キーの大きさが小さい。
  - 自分の指が邪魔になってキーが見えない。
  - 親指の関節が硬いので入力が辛い。
  - 普段のキーボード入力の際に小指担当キーを中指にて入力することが多いので、小指の位置に配置された担当キーが入力しにくい。
  - タッチタイピングの手法を忠実に守った入力手法を行っていないので、普段とは違う指を用いて入力をしなければならないキーがあった。
- その他のコメント
  - Windows 7 キーボードのように整然とキーが並んでいることが良いとは限らない。
  - 十分に慣れることができなかった。

Windows 7 キーボードに関してその他の良かった点及び改善すべき点

- 良かった点
  - Shift キーを押した場合に入力される文字がキートップに初期設定の状態で載っているため分かり易い。
  - キーが均等に配置されている。
  - キーの面積が広く、その配列が覚えやすい。
  - キーの間隔が広い。

  - キーに枠があるためにキーの領域が分かる。
  - キーの位置が固定されている。

- 改善すべき点
  - キー入力時に視覚以外のフィードバックがほしい。
  - キーの枠と枠の間の部分をタッチしてしまい、入力がされなかった場合があった。

Leyboard に関してその他の良かった点及び改善すべき点

- 良かった点
  - 小指を広げられる点が良かった。
  - 思ったよりも慣れるのが速い。
  - 入力に対する音のフィードバックが良かった。
  - 親指、人差し指、中指の担当するキーは入力しやすかった。
  - 親指スライドによってキーを探すのが簡単である。
  - 指の位置に対応している。
  - キーの上段と下段には手元を見なくても入力可能なキーがあった。
  - 親指の位置に Shift やキーセットを切り替えるキーがあるので操作しやすかった。
  - 操作ミスをしても空振りをするだけであり、誤ったキーが入力されることがない。
  - 自由な位置にキーを配置できる。
  - スペースキーが入力しやすい。
  - ホームポジションに正しく指を設置する訓練に使えるかもしれない。

- 改善すべき点
  - 入力可能状態のオン/オフは一目見れば分かるようにしてほしい。
  - キーとキーの境界が分かるようにしてほしい。
  - 自分に合ったキー配列をモデル化するのには時間がかかるだろう。
  - 入力時に自分が認識したキーの位置の少し上に実際のキーがあるように感じた。
  - Shift キーの位置が悪く、入力によってはキー全体が入力しにくい位置に変動してしまう。
  - 右手親指の Shift キーは入力しづらい。
  - 入力時のフィードバックをもっと分かりやすくしてほしい。

- Shift キーの入力やキーセットの切り替えの時には音声フィードバック入力時とは異なったものにしたほうが良い。
- スペースキーの周りの間隔はもっと広くても良いと感じた。
- Shift キーを入力したままの状態にしないと入力ができないのが辛かった。
- 本質的な問題ではないかもしれないが、全体的なレスポンスの速度が遅い。
- 小指のキーが非常に入力しにくい。
- 自分は親指同士を近づける傾向があるようなので両親指の外側はキーを配置したくない。
- 手の腹が画面に触れてもタッチ検出をしないでほしい。

5.8 考察

Leyboard は入力率やエラー率に関して Windows 7 キーボードに劣るという結果が導かれた。その原因を考察する。その際、アンケートから得られたコメントを参考にした。

5.8.1 誤入力に関する考察

ホームポジションキーが入力しにくいというコメントが見られた。実際にキー入力のログを見た所、隣り合ったキーの誤入力が多く、特にホームポジションキーの入力において顕著であった。具体的にはホームポジションキーを入力しようとして、誤って下段のキーを入力している。これは、Leyboard のキーの中心同士の間隔が 50 ピクセルと近すぎたために、fat finger problem が生じたためであると考えられる。事実、キーの 1 辺の大きさが Leyboard より大きい 75 ピクセルであった Windows 7 キーボードに対してはキーが大きいので入力がしやすいというコメントがなされている。

また、入力時に自分が認識したキーの位置の少し上に実際のキーがあるように感じたとのコメントがあることから、ユーザが意図したタッチ点とデバイスが検知したタッチ点に生じるずれも原因となっていることが分かる。

誤入力の多さは疲労感の原因にもつながっていることがコメントから読み取れる。

5.8.2 キーの配置に関する考察

自らの手がキーを視認するのに邪魔であるというコメントが複数見られた。実験中の被験者を観察したところ、図 5.8 のように、キーの位置を視認するために腕を水平状態から体の外側に向かって傾ける傾向があった。キーの配置に慣れなかったというコメントがあったので、被験者がキーの位置を把握できなかったことが原因の一環と考えられる。

指を設定した位置とその周りにキーを配置することに関しては好意的なコメントもあった (10 人中 4 人) ので、ユーザがキーの配置に慣れることでこの問題は改善されると考えられる。

図 5.8: 腕を傾ける被験者

ただし、今回の実装においてはキーの間隔が狭く設定されていたために、結局は視認して入力を行わないと精度が上がらなかったものと思われる。

キーセットの切り替えに関しては負担になるというコメントが多かった。キーが 1 つの盤面上に揃っている Windows 7 キーボードとの比較を行っているのでこのようなコメントが返ってくるのはある程度予想されることである。キーの配置位置に対しては好意的なコメントがあるので、キーセットの切り替えがより円滑に行われるように工夫することによって操作性が改善されると考えられる。

被験者の中には、我流のタイピング手法を用いているためにキー配置をタッチタイピングの手法に合わせた Leyboard のキー配置が入力しにくいとコメントした者がいた。

また、ホームポジションに正しく指を設置する訓練に使えるかもしれないと述べて、タッチタイピングの訓練への応用という興味深い提案を行うコメントが存在した。

5.8.3 機能に関する考察

親指が存在するキーの位置に応じて人差し指から小指のキーの位置が変動する機能に関しては肯定的な意見も否定的な意見もあった。入力しやすい、入力しにくいという意見が同時に存在しており、興味深い結果となっている。否定的な意見の中には、親指を固定しながら入力を行うので、指が全て浮いた状態と比べて入力しにくくなるキーが出現するというものがあり、検証の必要性が感じられる。

親指スライドによるキーの入力機能は動作の心地よさや操作の達成感の観点からの肯定的なコメントはあったが、実用性に関してはミスが多くなる、疲れる等の否定的なコメントが見られた。実際、今回の実装においては使わなくとも入力に支障をきたさない機能であった

ため、この機能を取り入れていくならば適切なデザインを考える必要がある。

手の姿勢を固定するために、指を固定しながらによるキーの入力機能を設けた。これはLeyboardにおける一定時間以上タッチされると入力を行わないキーの性質を利用したものである。しかし、この機能を利用した被験者は1人のみであり、その1人も姿勢が変になると否定的なコメントを残している。この機能はキャリブレーション時における誤入力を防ぐための機能としては有用である。しかし、入力においても活かせる機能であったとは言えない。

5.8.4 その他

Leyboardにおいて用いられている手法とは直接関係はないが、実装したシステムのデザインに問題があるとのコメントがあった。また、システムのレスポンスの遅さ自体が入力において問題となっているのではないかという、実装上の問題を指摘するコメントがあった。今回は既存のソフトウェアキーボードとの性能の比較という観点からWindows 7キーボードを比較対象として選択したが、QWERTY配列の物理キーボードと同じキー配置のソフトウェアキーボードをLeyboardと同じキーストロークイベント送信アルゴリズムを用いて実装し、比較対象として用いたほうが実験条件としては良かったと思われる。

手の腹が画面に触れてもタッチ検出をしないでほしいというコメントがあったが、実装上では手の腹は無視されるようになっている。しかし、デバイスであるICONIA-F54Eの画面と画面の境界付近における手の腹の接触が、手の腹が触れていない側の画面において、指によるタッチとして検出されてしまうハードウェア的問題が発生した。この場合、画面端という極端な位置にタッチ点が出現することになる。これに対してはプログラム側において、このようなタッチ点を無視する等の対策をすることが考えられる。

第6章　Leyboardの改良

第5章における実験結果とアンケートのコメントを受けた考察から、Leyboardの改良を行った。改良したLeyboardを図6.1に示す。

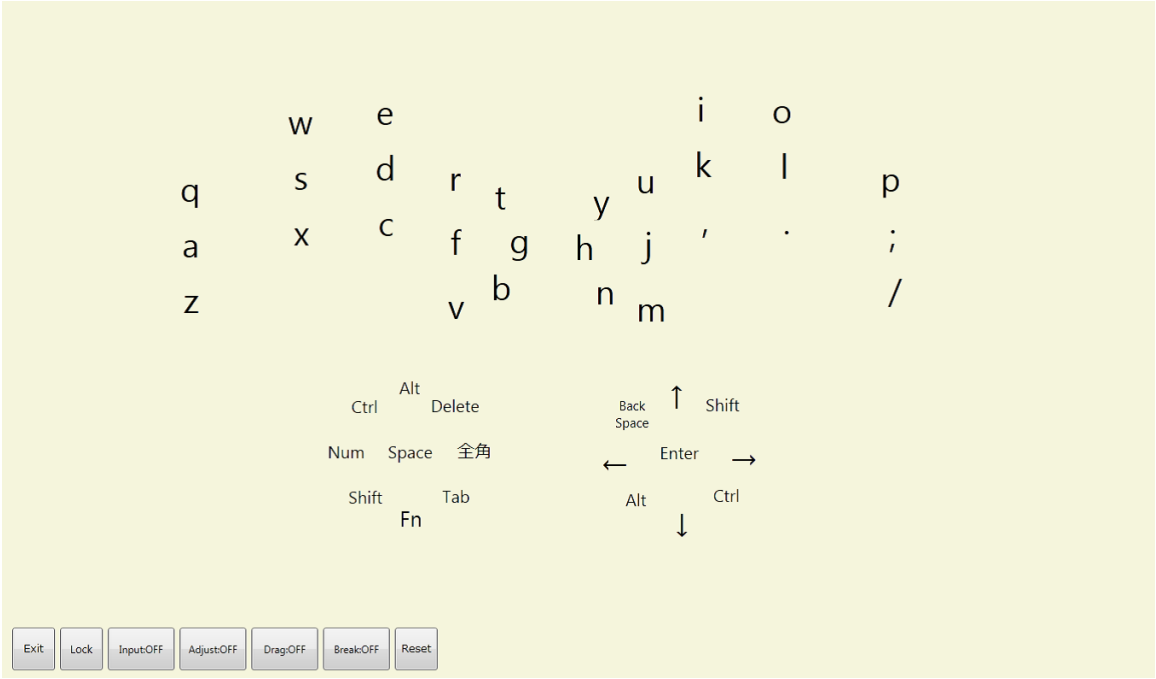

図6.1: 改良版Leyboard

まず、キーの間隔が狭すぎたためにホームポジションキーが入力しにくい問題の対処として、キーの間隔を広げた。中指から小指の担当するキーはキーの間隔を65ピクセルに、人差し指の担当するキーはキーの間隔を75ピクセルに変更した。人差し指の担当するキーのみキーの間隔が広くなっているのは、人差し指の担当するキーの数が多いため、キーが密集するからである。評価実験の結果より、キーの間隔を近づけすぎるとfat finger problemによって入力ミスが増加する。逆に遠ざけすぎると指がキーに届かず入力ができなくなる。また、人差し指は親指を除いた4本の指の中において端に位置し、さらに指の中においては長さがある。そのため、キーの間隔を他の指の担当キーに対して大きく取っても入力に支障をきたさないと判断した。キーの間隔を大きく取った場合、キーを格子状に配置するとホームポジションキーの斜めに位置するキーに指が届きにくい。そこで、親指の担当するキーと同様に、キー

を円周上に配置した。

小指の担当するキーはホームポジションキーと非ホームポジションキー 2 つの合計 3 つに減らした。これは、小指は指の中において短いために、複数のキーを入力するには指が届きにくいと考えたからである。また、入力するキーの数を減らした方が誤入力を少なくすると考えたからでもある。減らされたキーは親指の担当キーとするか他のキーセットに移動させた。このため、親指周りのキー配置が大きく変わっている。

キーセットの切り替えが負担になるとのコメントがあった。ここで、例えばキーセットを文字セットから数字・記号セットに切り替え、入力を行った後に文字セットに戻す動作を考えると、必要な行程数が多い。そこで、最後の文字セットに戻す行程を自動的に行うことによって行程数を減らすことを試みた。文字セットから数字・記号セットまたはファンクション・テンキーセットに切り替える際には、該当するキーセット切り替え専用のキーを入力する必要がある。そこで Shift キーと同様に該当するキーを入力している間のみキーセットを切り替えるようにし、指を離した時にキーセットが自動的に文字セットに戻るようにした。しかし一部記号のように、数字・記号セットに切り替えた後に、Shift キーを入力する必要のある文字が存在する。この場合、キーセットを切り替えた後に、Shift キーを入力するために手を離すと、キーセットが文字セットに戻ってしまう。これらの文字はキーセットを切り替えるキーの隣に Shift キーを設けることによって、親指スライドを用いた一連の流れにおける入力を可能にした。

第7章　まとめと今後の課題

本研究では、タッチタイピングをスレート端末以上の大きさのタッチパネル端末において実現するソフトウェアキーボードとして Leyboard を示した。Leyboard はユーザの指の設置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。また、親指周りにキーを多数配置することによって、手の移動量を抑えながらもアルファベットだけでなく多数のキーの入力を可能にした。キー位置をユーザの手の形状に適応させるため、キーの基本配列である QWERTY 配列を壊さない範囲において、ユーザがキーの位置を自由に設定できるようにした。本研究は両手によって入力を行うため、10 点マルチタッチが可能である端末を対象とする。よって、現在発売されている製品において実現可能である。

Leyboard の実装を行った後に、その精度と入力速度の評価実験を行った。実験では、マイクロソフト社の OS である Windows 7 に標準搭載されているソフトウェアキーボードと Leyboard を比較した。その結果、検証時点における Leyboard は統計的には Windows 7 のソフトウェアキーボードに入力性能の面で劣っていたが、被験者から多くの知見を得ることができた。そこで、その知見を基に Leyboard の改良を行った。

今後は、改良された Leyboard の性能を評価し、その妥当性を検証する。第 5 章の評価実験のコメントにおいて、処理プロセスの遅さが障害となっている可能性が指摘されていたので、実験の際には条件をそろえるために比較用のソフトウェアキーボードを Leyboard と同様のキーストロークイベント送信アルゴリズムを用いて実装する必要がある。

今回の実験においてキーの位置補正機能やキーのドラッグ機能の有用性は未検証であった。そこで、ユーザの手の動きに合ったキー配列が初期設定時のキー配列と比べて入力に与える影響を検証する実験も行う。

また、第 5 章の評価実験のコメントより、完全なタッチタイピングではない我流の入力方式を用いるユーザには Leyboard は扱いにくいという知見が得られた。そこで、ホームポジションキーと非ホームポジションキーを自由に設定可能なインタフェースを持つことによって、個人の癖に合わせることが可能であるソフトウェアキーボードの設計を考える。

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。

志築文太郎先生には本研究への数多くのご指導や助言を頂きました。また、提案した手法を実現するうえで必要であったデバイスの調達を行った際に、著者からの申請に対しても快く応じていただき、その後不幸にもデバイスの不調に見舞われた際にも適切な対応策をすぐに提示してくださいました。本研究を締めるところまで著者が到達できたのは間違いなく志築先生の御助力あってのことです。田中二郎先生には各種文字や記号の入力方法に関して様々な助言をいただきました。三末和男先生には比較対象を明確化することという本研究に対する重要な助言と、本研究の入力手法に関しての意見をいただきました。以上、指導教員の方々に感謝いたします。

また、私が実装途中であった Leyboard のデモを行う機会があった際に、直接の指導教員ではないのにもかかわらず、高橋伸先生からは様々なご意見を頂きました。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は、研究活動において様々な意見をいただいたほか、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 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。特に同期の野口杏奈さんには彼女自身の研究の過程において作られたプログラムの提供を、実験用にプログラムを調整していただくことも含めて行っていただきました。誠にありがとうございます。

著者の父には研究をしていく上でいろいろとアドバイスをいただきました。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著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、両親にはいつも感謝しています。

最後に、共に切磋琢磨した友人方やその他、私の大学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。

参考文献

[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.

[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.

[Gen83] Donald R. Gentner. Keystroke timing in transcription typing. In William E. Cooper, editor, *Cognitive Aspects of Skilled Typewriting*, pp. 95–120. 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, 2010.

[GT10] Kentaro Go and Leo Tsurumi. Arranging touch screen software keyboard split-keys based on contact surface. In *Proceedings of the 28th of the International Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA ’10, pp. 3805–3810, 2010.

[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.

[SL07] Christine Szentgyorgyi and Edward Lank. Five-key text input using rhythmic mappings. In *Proceedings of the 9th International Conference on Multimodal Interfaces*, ICMI ’07, pp. 118–121, 2007.

[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, 2011.

[遠藤 06] 遠藤裕貴, 郷健太郎. タッチ画面におけるバブルカーソル状適応型キーボードの提案. 情報処理学会研究報告. HI, ヒューマンインタフェース研究会報告, pp. 25–32, 2006.

[五味 08] 五味雄一, 寺田努, 塚本昌彦. ボタンの押下時間を利用した並列入力可能な文字入力方式. 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, pp. 57–62, 2008.

付録

次ページ以降は、第 5 章において述べた評価実験に関係する書類である。1 つ目は被験者に配った実験同意書及びアンケートである。2 つ目は著者が被験者に対して実験の説明を行った際の補助として、説明に必要である事項をメモした内容を書き起こしたものである。

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、既存のソフトウェアキーボード（Windows 7 に標準搭載されているソフトウェアキーボード。以下、Windows 7 キーボード）と本研究において開発したソフトウェアキーボード（以下、Leyboard）を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、英文パングラム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

英文パングラムを 10 回入力することを 1 セットとします。入力はできるだけ速く、かつ正確に行ってください。キーボードごとに入力を 6 セット行います。このうち、最初の 2 セットは練習セットとなります。セット間には休憩時間を自由に取ってよいものとします。

実験手順

・甲グループの方

1. Windows 7 キーボードを使用して、入力を 6 セット行う
2. Leyboard を使用して、入力を 6 セット行う
3. アンケート記入

・乙グループの方

甲グループの 1 と 2 の手順を逆に行います。他は甲グループと同様に行います。

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　情報学群　情報メディア創成学類　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、タッチ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. 文字入力において自分が最も習熟していると思う入力手段をお答えください。
   物理キーボード・ソフトウェアキーボード・その他：_____
   （該当するものを〇で囲んでください。その他の場合は具体的にお答えください）

4. 3 にて物理キーボードまたはソフトウェアキーボードと答えた方にお聞きします。使用しているキーボードのレイアウトをお答えください。
   QWERTY・その他：_____
   （該当するものを〇で囲んでください。その他の場合は具体的にお答えください）

5. 1 日あたりにおいて 3 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。
   _____時間

6. Windows 7 キーボード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：

7. Leyboard 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. Leyboard の親指の位置に応じてキーの位置が変動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：

9. Leyboard の親指を固定しながらによるキーの入力機能を使いましたか？使った場合にはこの機能を評価してください。

   この機能を　使った・使っていない　（該当する方を〇で囲んでください）

   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：

10. Leyboard の親指スライドによるキーセットの切り替えまたは Shift キーの入力機能を使いましたか？使った場合にはこの機能を評価してください。

    この機能を　使った・使っていない　（該当する方を〇で囲んでください）

    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：

11. Windows 7 キーボードのキー配列を全般的に評価してください。

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

理由：

12. Leyboard のキー配列を全般的に評価してください。

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

理由：

13. Windows 7 キーボードを利用して他に、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：

14. Leyboard 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。

被験者に通達すべきことのリスト

- 例文は大文字や記号も含めて提示された文章に忠実に入力することを説明する。
- 入力は正確さを優先し、次に速さを重視することを説明する。
- 被験者に何セット終わったかを把握できるように、セットが終わるごとに何セットまで終わったのか紙に書くことを説明する。
- Leyboard のキーセットの切り替えに関して説明する。
- Leyboard のキーの入力方法を説明する。特に記号の入力方法に関しては詳細を説明する。
- Leyboard の親指の位置に応じてキーの位置が変動する機能について説明する。
- Leyboard はいずれかの親指担当キーを複数同時に入力した場合、親指の位置に応じてキーの位置が変動する機能が異常作動するので行わないことを説明する。
- Leyboard の親指を固定した状態を維持しながらによるキーの入力機能について説明する。
- Leyboard の親指スライドによる親指担当キーの入力機能について説明する。
- Leyboard 特有のキー配置について、どの位置に何のキーが存在するかを説明する。
- Leyboard の空白の入力は現在のキーセットに切り替えるキーへの入力によって行われることを説明する。具体的には数字・記号セットの時は数字・記号セットへと切り替える Num キーが空白として機能し、Space キーはキーセットの文字セットへの切り替えとして機能することを説明する。
- Leyboard において万が一入力ができなくなった場合は Ctrl、Alt、Shift キーを順番に入力すれば復帰することを説明する。(Ctrl、Alt、Shift キーのいずれかの入力が固定されるバグが原因)
- Windows 7 キーボードは Shift キーが一度入力されると次のキーを入力するまで固定されることを説明する。
- Windows 7 キーボードを利用するタスクにおいて入力ができなくなった場合は例文入力システムからフォーカスが離れているため、例文入力システムのウィンドウをタッチすることによってフォーカスを再び与えることを説明する。