情報処理学会論文誌 Vol.55 No.4 1353–1364 (Apr. 2014)

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

久野 祐輝[1,a)] 志築 文太郎[2,b)] 田中 二郎[2,c)]

受付日 2013年6月21日, 採録日 2014年1月8日

概要：本稿において我々はユーザがより高速に入力が可能なソフトウェアキーボードである “Leyboard” 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．これにより，入力がより正確になり，結果として入力が高速になると我々は考えた．我々は Leyboard のプロトタイプを用いて長期間にわたる実験を行った．実験の結果，Leyboard で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った．

キーワード：タッチパネル，文字入力，ソフトウェアキーボード，長期実験

A Software Keyboard that Places Keys at Positions of Fingers and their Surroundings

Yuki Kuno[1,a)] Buntarou Shizuki[2,b)] Jiro Tanaka[2,c)]

Received: June 21, 2013, Accepted: January 8, 2014

Abstract: In this paper, we present a software keyboard called Leyboard that enables users to type faster. Leyboard makes typing easier by placing keys at the positions of fingers and their surroundings. With this keyboard, users can type more accurately and thus faster than with ordinary software keyboards. We implemented a prototype and performed a long-term user study. The study has proved the usefulness of Leyboard that enables the user to type statistically significantly faster than the ordinary software keyboard.

Keywords: touch screen, text entry, software keyboard, long-term study

1. 序論

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と同一であるが，物理的なフィードバックを得ることができない．そのためユーザは入力を意図するキーに指を合わせて，正確に入力することが困難である．

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できるので，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，入力速度を向上させられると考えられる．

本稿では，上記の考えに基づいたソフトウェアキーボードの入力速度を調査するために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示す．さらに，長期的な実験によって提案手法のキーボードの有用性を調べる．

本研究の新規性は以下の点である．

- 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

1 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki 305–8573, Japan
2 筑波大学システム情報系
Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba, Ibaraki 305–8573, Japan
a) kuno@iplab.cs.tsukuba.ac.jp
b) shizuki@cs.tsukuba.ac.jp
c) jiro@cs.tsukuba.ac.jp

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）．

- 親指周りに配置されたキーにおいて，ユーザは画面上で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．これらは親指周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替えと修飾キーの入力を 1 回のスワイプ操作で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．

以下に本稿における用語を定義する．タップとはユーザが 1 本の指を画面に接触させた後に短時間内にそこから離す入力である．スワイプとはユーザが 1 本の指を画面上に接触させたまま移動させる入力である．

2. 関連研究

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とっている研究として，LiquidKeyboard [18]，CATKey [8]，Personalized Input [5]，Gunawardana らの研究 [10]，Go らの研究 [9] 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，Leyboard 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，キー入力をともな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 では，Leyboard 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．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うる）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ー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．一方，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．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 [4] 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．Go らの研究も含めて，これらのパイメニューを経由する入力はその展開と入力に時間を要するため，特にそのような操作を必要としない Leyboard の方が入力効率は良いと考えられる．

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 [17] や BrailleTouch [19] 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高速な入力を可能にすることを試みる．

本研究のアプローチと異なり，ソフトウェアキーボードに皮膚感覚へのフィードバックを付与する研究としては McAdam らの研究 [14], [15] や SLAP widgets [20] がある．本研究ではフィードバックの不在による入力の困難性を，キーの位置をユーザの指の位置に合わせることにより抑えている．

本研究には親指スワイプ入力というジェスチャ入力の要素が存在する．同様に文字入力にジェスチャ入力を用いた研究として Gestyboard [2]，Bimanual Gesture Keyboard [1]，Beyond QWERTY がある．本研究におけるジェスチャ入力は複数の操作をまとめるものであり，文字入力自体には使われていない．

本研究ではキーの入力をタップにて行うこととした．TapBoard [11] 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，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できるようにした．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．本研究におけるタッチ時間の閾値はこれらを区別するためにある．本研究では，これは経験的に設定した値であった．一方，TapBoard では被験者実験を行うことによって適切な閾値を求めている．

3. Leyboard

我々は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を開発した [12], [13]．本節ではこのプロトタイプの設計と機能について述べる．以降，本プロトタイプを “Keyboard” の “K” を 1 文字進めて “Leyboard” と呼ぶ．

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見なす．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 [6] 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 [3] 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，ユーザが両手すべての指（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．ユーザが画面上にすべ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，キー入力を行えなくした．キー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．ただし，キー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キー入力を行う．厳密にはユーザが画面に置く指の数が 8 点を超えた場合にキー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 本の指を置くたび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，ユーザが両手すべての指（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見なす．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 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

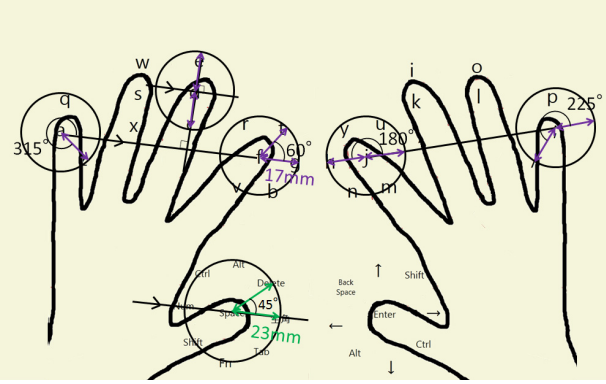

図 1　キーの配置規則
Fig. 1　The rules of the key placement.

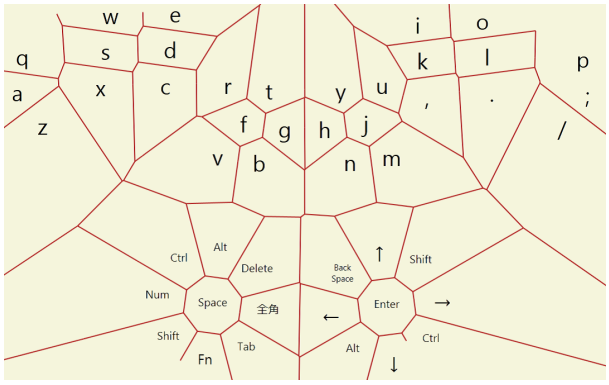

図 2　アルファベットセットの配列
Fig. 2　The layout of alphabet set.

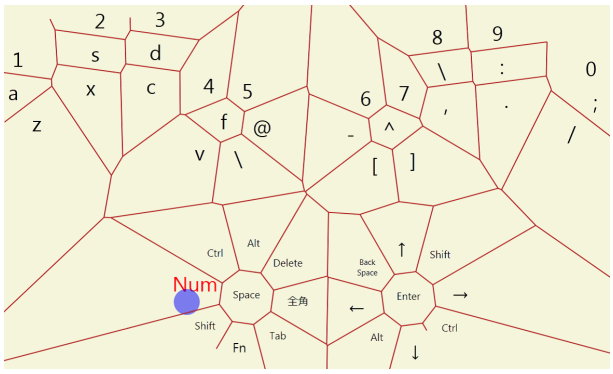

図 3　数字・記号セットの配列
Fig. 3　The layout of numbers and symbols set.

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 17 mm 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 23 mm とした．17 mm，23 mm という数値はともに経験的に求めた．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すべて等しくなる．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我々は考えた．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）のほか，後述するキーセット切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 のすべ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．

3.5　キーセット切替え

テキスト入力に必要なキーのすべてを 1 つのレイアウトとして配置すると，すべ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．そこで我々は Leyboard に 3 つのキーセットを用意し（**図 2**，**図 3**，**図 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，我々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 や図 4 上の円によって示されるキーセット切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替え用のキーから離したとき，キーセットは図 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

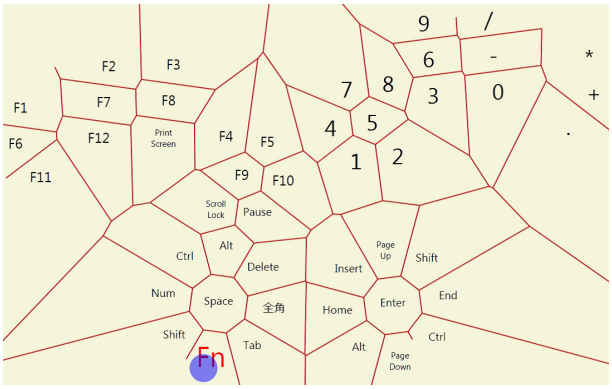

図 4　ファンクション・テンキーセットの配列
Fig. 4　The layout of functions and numerical keypad set.

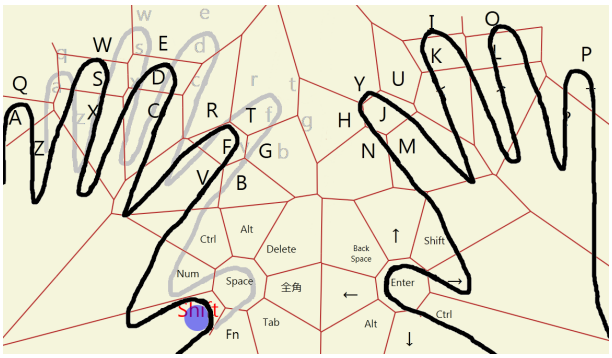

図 5　親指基点スライドの例
Fig. 5　The example of thumb based sliding.

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

我々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．たとえば，f キーと Shift キーで大文字の ‘F’ を入力するときである．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 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た．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，親指基点スライドによって，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が平行移動する．

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 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，キーの位置はつね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

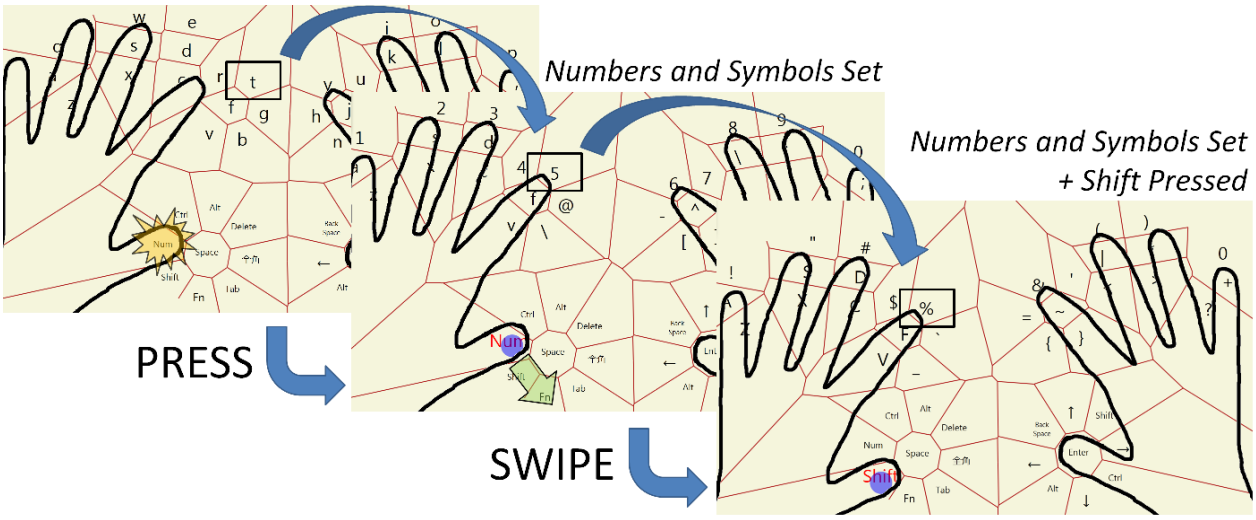

図 6　親指基点スライドと親指スワイプ入力の連携例
Fig. 6　The example of the combination of thumb based sliding and thumb swipe input.

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．たとえば図 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．

4.　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．

5.　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，我々は 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 17 mm であった．各ソフトウェアキーボードの横幅はともに 31 cm であった．縦幅は Windows 7 キーボードが 10.5 cm，Leyboard が 17.5 cm であった．実験は約 1 年 4 カ月という長期にわたって行われた．

5.1　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 **7** に示す．ICONIA-F54E は液晶サイズが 14 インチ，解像度が 1,366 × 768 ドット（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．

図 7　実験環境
Fig. 7　Experiment environment.

5.2　被験者と実験タスク

本実験の被験者は著者の 1 人である．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 [22] 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 1** に示す．

表 1 被験者の手指の大きさ
Table 1 Size of the hands of the participant.

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

我々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すべての文字を少なくとも 1 回用いた文章である．たとえば “A quick brown fox jumps over the lazy dog.” があげられる．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，本研究ではこれにならった．パングラムの入力をタスクとすることの利点は，各文章ですべてのアルファベットが必ず 1 回は入力されることである．そのため，すべ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．いい換えれば，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 1,449 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つねに表示されていた．

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我々は考えた．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．

5.3 実験結果

5.3.1 入力率

我々は入力率を 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 [7]，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および対数近似に基づいた近似曲線を**図 8** 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 59.34 wpm，Leyboard が 61.27 wpm 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 41.65 wpm，Leyboard が 43.66 wpm となった．各キー

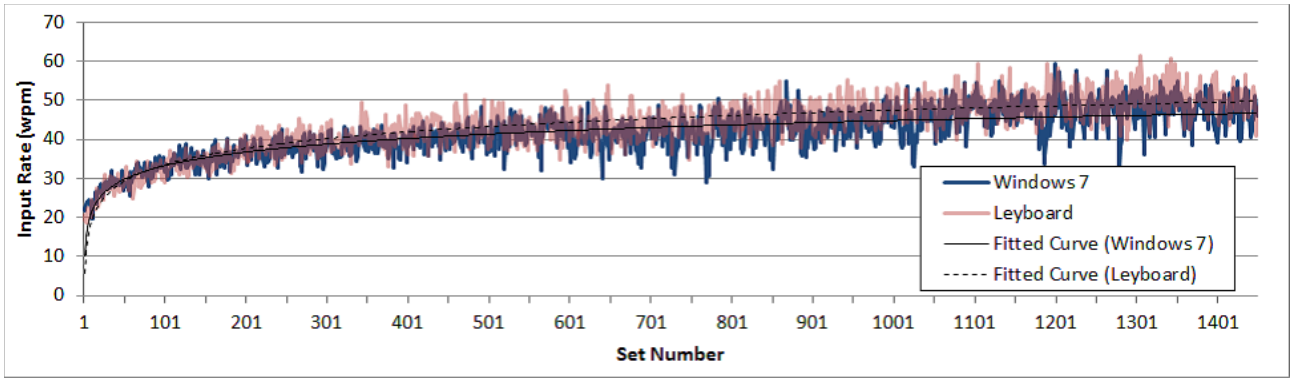

図 8　各ソフトウェアキーボードの入力率（wpm）
Fig. 8　Input rate (wpm) on each software keyboard.

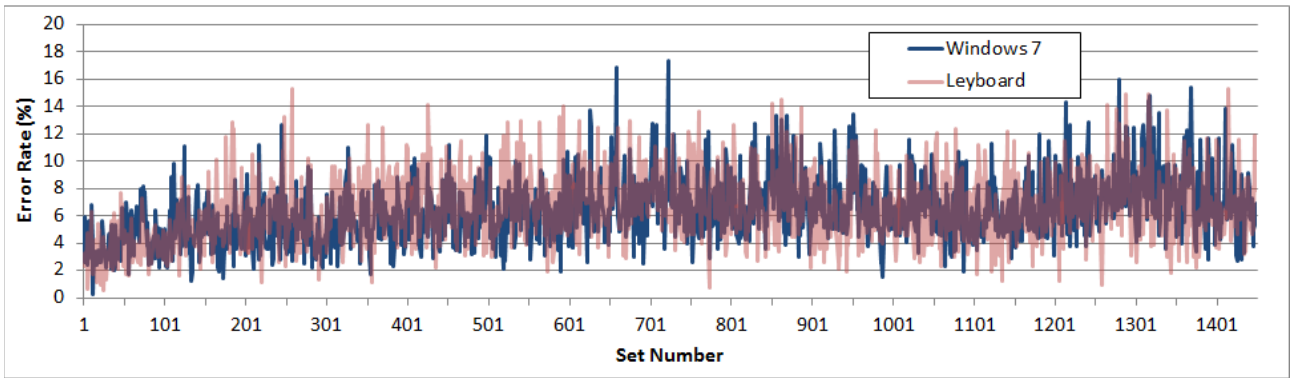

図 9　各ソフトウェアキーボードのエラー率
Fig. 9　Error rate on each software keyboard.

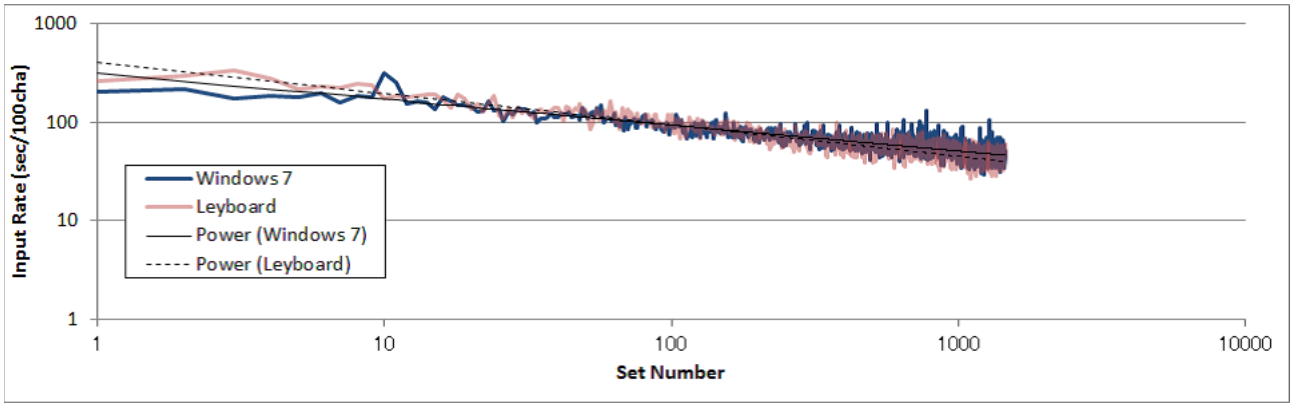

図 10　各ソフトウェアキーボードの入力時間（100 文字あたりの入力時間）
Fig. 10　Input rate (seconds per 100 characters) on each software keyboard.

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 47.09 wpm，Leyboard が 49.69 wpm 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．

5.3.2　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 **9** 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 52.58 wpm であり，エラー率は 5.98%であった．また，入力率の最大値は 55.63 wpm であった．

5.3.3　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 [16]．図 **10** に

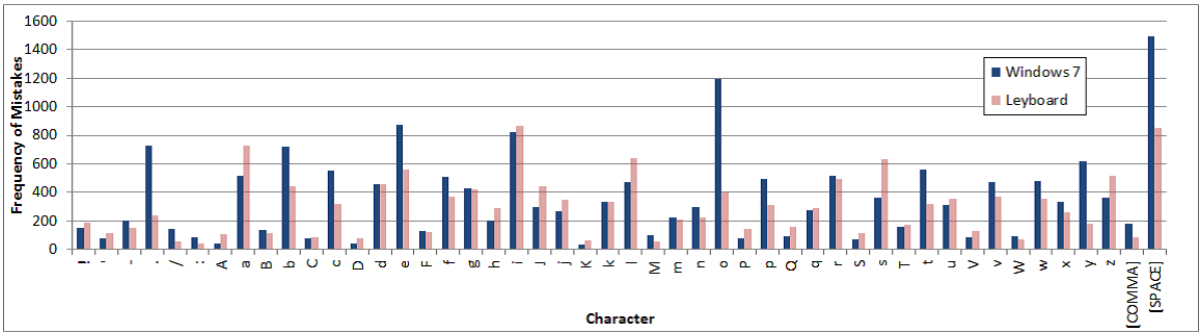

図 11　誤字の分析結果
Fig. 11　The result of analysis of mistakes.

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いえる．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 41.63 wpm，Leyboard が 43.71 wpm となり，全データの 41.65 wpm，43.66 wpm と比較して特に差はなかった．

6.　考察

6.1　キーボードごとの入力率およびエラー率の差

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 1,449 セット分の入力率およ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（$t = 0.7552,\ p = 0.4523 > 0.01$）．

6.2　エラー内容分析

実験中に生じたエラーの内容を分析するため，我々は実験途中のすべてのキー入力内容に対してログを取った．そし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箇所はスキップする．
(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すべて無視する．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．

たとえば，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．

Windows 7 キーボードには 23,441 回，Leyboard には 23,252 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 17,376 回の誤字と 6,065 回の脱字が存在した．Leyboard には 14,521 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 11** に，脱字に関しては**図 12** 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

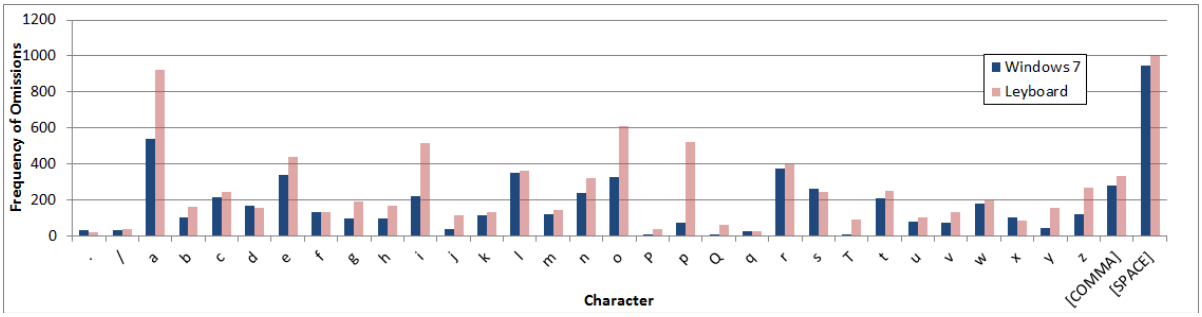

図 12　脱字の分析結果
Fig. 12　The result of analysis of omissions.

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．

6.2.1　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 2** と**表 3** に示す．

表 2 を見ると，正解と誤字の組合せは Space と n 以外すべ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いえる．

表 3 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 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．おそ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いえる．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．Shift キーと j キーの距離が近い Leyboard において，被験者が 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．

表 2　Windows 7 キーボード誤字内容上位
Table 2　Top errors in Windows 7 keyboard.

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 3　Leyboard 誤字内容上位
Table 3　Top errors in Leyboard.

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

6.3　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 4** と**表 5** 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いえる．特

表 4 Windows 7 キーボード脱字内容上位
Table 4 Top omissions in Windows 7 keyboard.

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 Leyboard 脱字内容上位
Table 5 Top omissions in Leyboard.

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

に Leyboard の脱字における “uiz” や “mph” の比率は非常に高い．

脱字はキーをタップし損なうことによって発生する．よって脱字の比率が大きくなる入力はタップしにくい要素が存在すると考えられる．タップしにくくなる要素として 1 つ考えられるのは指の移動量である．指の移動量が大きくなる入力を素早く行おうとすると，キーをタップする前にさらに次のキーを入力しようと指が動き結果としてタップをし損なうことがある．Leyboard ではユーザの指の位置にホームポジションキーを設置し，その周囲に非ホームポジションキーを配置した．その結果，キー間の距離が Windows 7 キーボードと Leyboard のそれぞれで大きく異なるキーが出現した．“uiz” や “mph” の場合を考えると，m–p 間，p–h 間，u–i 間の距離はキーが直線状に並んでいる Windows 7 キーボードよりも大きくなることが図 2 より分かる．そのため，これらのパターンは Leyboard において特に脱字が起こりやすくなったと思われる．この解決策としては，たとえばタップ位置と入力の傾向に応じてタップ位置を補正することが考えられる [21]．なお i–z 間の距離に関しては入力する手が異なるため，この問題とは関係がない．

7. 結論と今後の課題

本研究では，QWERTY 配列のソフトウェアキーボードの入力速度を向上させる方法としてキーの位置をユーザの指の位置に合わせることをあげ，それを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とその周囲にキーを配置する．また，親指周りのキーの配置，親指スワイプ入力，親指基点スライドによって，ユーザは手の変形を抑えながらも 102 種類のキーが入力可能である．

Leyboard の実装を行った後に，その入力速度の長期的な評価実験を行った．その結果，Leyboard の入力率は従来のソフトウェアキーボードよりも高いことが判明した．Leyboard におけるエラーの原因は主に縦に隣り合ったキーの入力ミス，そして一部のキーが入力しにくいことである．後者に関してはキーの配置がユーザの手の形に合わせて曲がっているために特定のキー間の距離が大きいためと考えられる．よって今後は Leyboard の入力率の優位性を維持しつつ，これらの入力を確実に行うための改良を考える．また，今回は実験の入力文章にパングラムを用いたため，特定の単語の出現率が高くなっている．そこで一般的な文章を集めたコーパスによる検証も行う．今回の実験において，被験者は双方のキーボードの入力を行ったが，これが熟練度の上達を阻害した可能性がある．そのため次に実験を行う際には被験者をグループ別に分けることを考えている．

参考文献

[1] Bi, X., Chelba, C., Ouyang, T., Partridge, K. and Zhai, S.: Bimanual gesture keyboard, *Proc. 25th Annual ACM Symposium on User Interface Software and Technology, UIST ’12*, pp.137–146 (2012).

[2] Coskun, T., Artinger, E., Pirrilli, L., Korhammer, D., Benzina, A., Grill, C., Dippon, A. and Klinker, G.: Gestyboard: A 10-finger-system and Gesture Based Text Input System for Multi-touchscreens with No Need for Tactile Feedback, *Proc. 10th Asia-Pacific Conference on Computer-Human Interaction, APCHI ’12*, pp.701–702 (2012).

[3] Dachev, H.: Speedy Voronoi diagrams in as3/flash, Hristo Dachev (online), available from ⟨http://blog.controul.com/2009/05/speedy-voronoi-diagrams-in-as3flash/⟩ (accessed 2014-01-19).

[4] Findlater, L., Lee, B. and Wobbrock, J.: Beyond QWERTY: Augmenting Touch-Screen Keyboards with Multi-Touch Gestures for Non-Alphanumeric Input, *Proc. 2012 ACM Annual Conference on Human Factors in Computing Systems, CHI ’12*, pp.2679–2682 (2012).

[5] Findlater, L. and Wobbrock, J.: Personalized input: Improving ten-finger touchscreen typing through automatic adaptation, *Proc. 2012 ACM Annual Conference on Human Factors in Computing Systems, CHI ’12*, pp.815–824 (2012).

[6] Fortune, S.: A sweepline algorithm for Voronoi diagrams, *Proc. 2nd Annual Symposium on Computational Geometry*, *SCG '86*, pp.313–322, ACM (1986).

[7] Gentner, D.R.: Keystroke Timing in Transcription Typing, *Cognitive Aspects of Skilled Typewriting*, Cooper, W.E. (Ed.), Springer-Verlag, pp.95–120 (1983).

[8] Go, K. and Endo, Y.: CATKey: Customizable and Adaptable Touchscreen Keyboard with Bubble Cursor-like Visual Feedback, *Proc. 11th IFIP TC 13 International Conference on Human-computer Interaction*, *INTERACT'07*, pp.493–496 (2007).

[9] Go, K. and Tsurumi, L.: Arranging touch screen software keyboard split-keys based on contact surface, *CHI '10 Extended Abstracts on Human Factors in Computing Systems*, *CHI EA '10*, pp.3805–3810 (2010).

[10] Gunawardana, A., Paek, T. and Meek, C.: Usability Guided Key-Target Resizing for Soft Keyboards, *Proc. 15th International Conference on Intelligent User Interfaces*, *IUI '10*, pp.111–118, ACM (2010).

[11] Kim, S., Son, J., Lee, G., Kim, H. and Lee, W.: TapBoard: Making a Touch Screen Keyboard More Touchable, *Proc. 2013 Annual Conference on Human Factors in Computing Systems*, *CHI '13*, pp.553–562 (2013).

[12] Kuno, Y., Shizuki, B. and Tanaka, J.: Leyboard: A Software Keyboard that Places Keys at Positions of Fingers and their Surroundings, *Proc. 10th Asia-Pacific Conference on Computer-Human Interaction*, *APCHI '12*, pp.723–724 (2012).

[13] Kuno, Y., Shizuki, B. and Tanaka, J.: Long-Term Study of a Software Keyboard that Places Keys at Positions of Fingers and their Surroundings, *Proc. 15th International Conference on Human-Computer Interaction*, *HCI International 2013*, Part V, pp.72–81 (2013).

[14] McAdam, C. and Brewster, S.: Distal tactile feedback for text entry on tabletop computers, *Proc. 23rd British HCI Group Annual Conference on People and Computers*, *BCS-HCI '09*, pp.504–511 (2009).

[15] McAdam, C. and Brewster, S.: Mobile phones as a tactile display for tabletop typing, *Proc. ACM International Conference on Interactive Tabletops and Surfaces*, *ITS '11*, pp.276–277 (2011).

[16] Newell, A.: Mechanisms of skill acquisition and the law of practice, *Cognitive Skills and Their Acquisition*, pp.1–55 (1981).

[17] Oliveira, J.A., Guerreiro, T., Nicolau, H., Jorge, J. and Gonçalves, D.: BrailleType: Unleashing braille over touch screen mobile phones, *Proc. 13th IFIP TC 13 International Conference on Human-Computer Interaction - Volume Part I*, *INTERACT '11*, pp.100–107 (2011).

[18] Sax, C., Lau, H. and Lawrence, E.: LiquidKeyboard: An Ergonomic, Adaptive QWERTY Keyboard for Touchscreens and Surfaces, *Proc. 5th International Conference on Digital Society*, *ICDS '11*, pp.117–122, XPS (2011).

[19] Southern, C., Clawson, J., Frey, B., Abowd, G. and Romero, M.: An evaluation of BrailleTouch: Mobile touchscreen text entry for the visually impaired, *Proc. 14th International Conference on Human-computer Interaction with Mobile Devices and Services*, *MobileHCI '12*, pp.317–326 (2012).

[20] Weiss, M., Wagner, J., Jansen, Y., Jennings, R., Khoshabeh, R., Hollan, J.D. and Borchers, J.: SLAP widgets: bridging the gap between virtual and physical controls on tabletops, *Proc. 27th International Conference on Human Factors in Computing Systems*, *CHI '09*, pp.481–490 (2009).

[21] Yin, Y., Ouyang, T.Y., Partridge, K. and Zhai, S.: Making touchscreen keyboards adaptive to keys, hand postures, and individuals: A hierarchical spatial backoff model approach, *Proc. SIGCHI Conference on Human Factors in Computing Systems*, *CHI '13*, pp.2775–2784 (2013).

[22] 人間生活工学研究センター：日本人の手の寸法データ集2010，人間生活工学研究センター（オンライン），入手先〈http://www.hql.jp/index.html〉（参照 2014-01-19）.

久野 祐輝 （学生会員）

1989 年生．2012 年筑波大学情報学群情報メディア創成学類卒業．同大学大学院システム情報工学研究科コンピュータサイエンス専攻博士前期課程在学中．ヒューマンインタフェースに関する研究に興味を持つ．

志築 文太郎 （正会員）

1971 年生．1994 年東京工業大学理学部情報学科卒業．2000 年同大学大学院情報理工学研究科数理・計算科学専攻博士課程単位取得退学．博士（理学）．現在，筑波大学大学院システム情報系准教授．ヒューマンインタフェースに関する研究に興味を持つ．日本ソフトウェア科学会，ACM，IEEE Computer Society，電子情報通信学会，ヒューマンインタフェース学会各会員．

田中 二郎 （正会員）

1975 年東京大学理学部卒業．1977 年同大学大学院理学系研究科修士課程修了．修士論文題目「タッチ打鍵法による日本文入力法の研究」．1984 年米国ユタ大学大学院計算機科学科博士課程修了．ユタ大学では関数型プログラミング言語の並列実装に関する研究に従事．PhD. in Computer Science．1985〜1988 年に（財）新世代コンピュータ技術開発機構で並列理論型プログラミング言語の研究開発に従事．1993 年から筑波大学に勤務．現在，筑波大学システム情報系教授．ACM，IEEE，電子情報通信学会，日本ソフトウェア科学会各会員．