2009－HCI－132 (2)
2009／3／13

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

藤田 晋也[†] 赤池 英夫[‡] 角田 博保[‡]

[†] 電気通信大学 電気通信学研究科 情報工学専攻 [‡] 電気通信大学 電気通信学部 情報工学科

本研究では，ウェアラブル機器向けの新しい日本文入力手法を提案する．近年，携帯電話やスマートフォンのような携帯情報端末が急速に普及してきており，ウェアラブルコンピュータも登場し始めている．このような環境では，フルキーボードに代わる優れた入力手段が求められている．そこで，曖昧性をもった運指情報と，これを絞込むための統計的手法を用いた入力手法を提案，実装し，有用性の評価を行った．評価実験により，平均で 1 分間あたり 33.5 文字の漢字かな混じり文を入力できることが示された．これは，同一被験者群におけるフルキーボード入力速度の 42.9%に相当し，実用に値するような入力速度が得られることを確認した．

The Proposal and Evaluation of A Japanese Text Input Method for Wearable Devices Using Fingering and Statistical Technique

Shinya Fujita[†] , Hideo Akaike[‡] and Hiroyasu Kakuda[‡]

[†]Department of Computer Science, Graduate school of Electro-Communications, The University of Electro-Communications

[‡]Department of Computer Science, The University of Electro-Communications

In this research, we propose a new Japanese text input method for wearable devices. Recently, handheld terminal such as cellular phone and smartphone is widely used, and wearable computers begin to appear. In such environment, a more suitable alternative to a full keyboard is required. Then, we proposed the input method using ambiguous fingerwork information and statistical techniques that restrict the information. As a result of an evaluation experiment, it turned out that 33.5 Japanese characters per a minute could be typed in on an average. It reached to 42.9% of the full keyboard input speed in the same test subject group, and we confirmed that practicable input speed was obtained with the proposed method.

1. はじめに

本研究では，モバイル環境やウェアラブルコンピュータで使用できるような，新しい日本文入力手法 “Guesso” を提案し，その実装と評価を行った．

近年，携帯電話やスマートフォンのような携帯情報端末が急速に普及してきており，ウェアラブルコンピュータも登場し始めている．このような環境では，両手で高速入力が可能なフルキーボード[⋆1]に代わる文章入力手段が求められている．しかし，既存手法には，入力速度が著しく劣る，習熟までに非常に時間がかかる，入力のために手元を確認する必要がある，といった欠点や問題点を持ったものが多い．

これらの問題を踏まえ，QWERTY キーボードに熟練したユーザの経験を利用し，高速入力が可能になるような日本文入力手法 “Guesso” を提案し，その実装と評価について報告する．

Guesso は，両手の各指先の位置にスイッチを取り付けたデバイスを装着し，“目の前に QWERTY キーボードがあると仮定” して日本文のローマ字入力を行ったときの運指を検出し，運指の組み合わせから，かな候補を日本文として尤もらしい順に提示する．

かな候補提示の手法については，自然言語処理の形態素解析で用いられるコスト最小法を，かなに応用した手法を採用しており，内部に辞書を持たない，連文節変換が可能などの独自性を有している．

2. 関連研究

非フルキーボード入力デバイスとしては，スマートフォン等に利用されるミニキーボード，Twiddler[1)]，

⋆1 本研究ではタッチタイピングができるフルサイズのキーボードのことを表す．

Virtual Keyboard[2], FingeRing[3] 等が挙げられるが，タッチタイピングができずデバイスを見る必要がある，入力方法が複雑で習熟までに時間がかかるなどの短所があるものが多い．

曖昧性のある少数キー入力手法としては，T9[4]，POBox[5]，FtoKey[6] が挙げられる．これらの手法は全て連文節入力ができず，曖昧性の解消には辞書を使用している．辞書の使用により，性能が辞書の内容に大きく左右される，辞書のメンテナンスが非常に難しい，といった難点が生じやすい．

3. 提案手法 “Guesso”

3.1 設計方針

提案手法 “Guesso” は，

- QWERTY キーボードの操作方法にできる限り従い，QWERTY 入力に熟練したユーザの学習コストを下げる
- フルキーボードに近い入力速度が得られる
- 机や膝の上など，空間以外の任意の場所に対して打鍵[*1]できる
- 入力デバイスを注視せずに入力できる

ことを基本方針とし，さらに，曖昧性のある少数キー入力手法であるが，関連研究とは異なり，

- かな確定部分と漢字変換部分を完全分離し，内部に辞書を持たず，外部漢字エンジンによる連文節変換が可能
- かな候補の順位付けアルゴリズムの使用

といった独自機能を持つことを方針として設計した．

単文節入力の場合は，いつでも文節を意識して入力を行わなければならないという負担がある．また，文章の修正を行うような場面や，直前の確定状況によっては，そもそも文節として成立しない部分の入力を行いたい状況にもなりうる (たとえば，“入力の方法” という文字列の入力で “の方法” の部分を入力するような場面)．これらの理由から，本手法では連文節変換を採用することにした．

また，内部に辞書を持たないという方針により，

- サイズの大きな辞書ファイルを搭載する必要が無い
- 文節に左右されない自由な文字列を入力できる
- 辞書に存在しない単語の入力も比較的簡単

のような利点も得られる．

外部漢字変換エンジンの使用による利点は，連文節変換が可能になることの他に，

- 精度の良い変換が可能になる
- 変換エンジンの学習機能が容易に使用できる

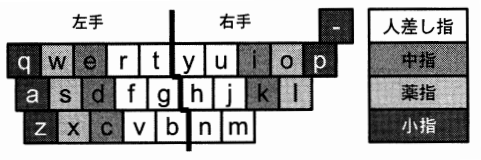

図 1 スイッチと文字の割り当て

- 目的に応じて変換エンジンを切り替えることができる

などが挙げられる．

3.2 手法の概要

Guesso では，両手の指先にあたる位置に各 1 個，計 10 個のスイッチを取り付けたデバイスを使用する．これを装着することで，机の上や大腿部等，任意の場所で打鍵することが可能になる．

親指以外の 8 個のスイッチが文字キーに対応する．文字用のスイッチが 8 個のみのため，これらだけでは入力したい文字を一意に決定することができない．このため，1 つのスイッチに複数の文字が割り当てられている．スイッチと文字の割り当ては図 1 のように，通常の QWERTY キーボードで打鍵を行う指と対応している[*2]．

ユーザがキーボードで日本文のローマ字打ちをするのと同様の指使いで打鍵すると，システムはその運指情報により，かな文字列の候補を生成し，より日本文として尤もらしい順に提示する．かなを確定した後は，外部のかな漢字変換エンジンを用いて連文節漢字変換を行う．これらの入力操作，確定後のカーソル操作，修正操作を 10 個のスイッチの組み合わせのみで行うことができる．

3.3 使用場面

この手法は，ヘッドマウントディスプレイ (HMD) を利用するような，ウェアラブルコンピュータでの使用を想定している．HMD と Guesso のデバイスを装着することで，歩行中や電車の中等でいつでも日本文の入力が可能になる．

4. デバイスの実装

手袋の指先部分へスイッチを取り付けたデバイスを作成した (図 2)．

指先に取り付けるスイッチへは，“クリック感” を導入した．キーボードのようなクリック感を導入することで，意図しない入力を減らし，軽快な入力を可能にすることが期待できる．クリック感を与えるために，メンブレンキーボードなどに使われるラバードームを

---

*1 本研究では各指で机などを叩くことでデバイスの指先のスイッチが押されることを “打鍵” と呼ぶ．

*2 ユーザの癖に応じて事前調整することが可能

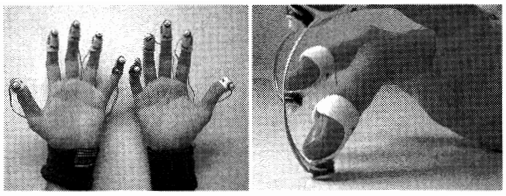

図 2 Guesso のデバイス

図 3 ゴム部品

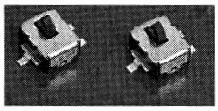

図 4 小型検出スイッチ

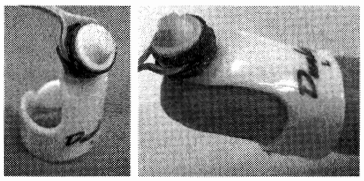

図 5 Guesso のデバイス (指先)

参考に，ゴム部品を作成した．これは，ゴム形状を設計し，樹脂を卓上フライス盤で削り出して型を作成し，その中に模型用のシリコーンゴムを流し込んで成型した (図 3)．ゴム部品と小型の検出スイッチ (図 4) を組み合わせ，さらにギター用のフィンガーピックに接着することで，指先に装着するためのデバイスが完成した (図 5)．

5. システムの実装

デモプログラム及び被験者実験プログラムは Java 言語で実装した．詳細な仕様を以下に示す．

5.1 かな候補の順位付け

Guesso の入力には曖昧性があるので，生成されるかな候補は膨大になってしまう (例:「KAKUDAKENN」と入力できるような打鍵を行うと，かなとして成立する候補数は 724 個)．自前の辞書で絞込む方法は連文節変換への対応が難しくなるため，かな候補に順位付けを行うアルゴリズムを採用した．

5.1.1 アルゴリズム

アルゴリズムは，自然言語処理の形態素解析で用いられるコスト最小法を，統計的パラメータを用いて，かなに応用した方法を用いた．

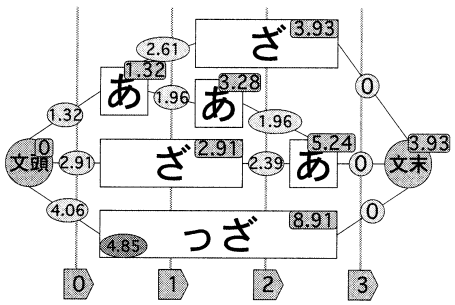

図 6 ラティス構造 (左小指 (Q,A,Z) を三回打鍵)

打鍵を行ったときの内部状態のデータ構造は図 6 のようになる．このようなデータ構造をラティス構造という．ローマ字の組み合わせによっては “ZZA →っざ” のように，2 文字以上のかなからなるノードも生成される．リンク上の数字はリンク間のコスト，ノードの左下の数字はノードコストを表す．ノードコストの表示が無いノードのコストは 0 である．ノードコストは，2 文字以上のかなからなるノードの場合にだけ発生する．縦線がポインタで，下端にあるラベルがポインタ番号である．各ノードの右上の数字については後述する．

各リンクとノードに適当なコストを与え，文頭から前向きに Viterbi アルゴリズムを行った後，A* アルゴリズムを適用することで，N-best 解を得ることができる[7]．

Viterbi アルゴリズム

Viterbi アルゴリズムを以下に示す．

(1) 文頭の部分最小コストを 0 とする
(2) 指定ポインタ $p = 0$ とする
(3) あるポインタで終わるノードを $w_{e1}, w_{e2}, \cdots$，そのポインタから始まるノードを $w_{s1}, w_{s2}, \cdots$ とし，p から始まる w_{si} を列挙する
(4) 各 w_{si} について次の操作を行う
- w_{si} の左に連結される w_{ej} を列挙する．このとき，各 w_{ej} について，
  - w_{ej} の部分最小コスト
  - w_{ej} と w_{si} のリンクのコスト
  - w_{si} のノードコスト

  の和の最小値を求め，その値を w_{si} の部分最小コストとする

(5) p に 1 加え，文末まで手順 3，4 を繰り返す

図 6 の各ノードの右上の数字が，文末までアルゴリズムを適用したときの部分最小解である．

A* アルゴリズム

Viterbi アルゴリズムで得られた各ノードの部分最小解の値をもとに，A* アルゴリズムによって N-best 解を得ることができる．

A* アルゴリズムは，探索の出発地点からある地点までの部分コスト g と，その地点から目標地点までの予想コスト h を計算し，その和 $g+h$ が最小の地点から探索を進めていくという方法である．文末からあるノードまでのコストが g，そのノードの部分最小解が h に対応する[*1]．

N-best 解を出力する A* アルゴリズムを以下に示す．

(1) 文末の左につながるノードを OPEN リストに入れる
(2) OPEN リストが空なら終了
(3) OPEN リストから $f=$ (文末からのコスト + ノードの部分最小解) が最小のノードを取り出し，OPEN リストから除く
(4) 取り出したノードが文頭であれば経路 (文字列) を出力，N 回目の出力であれば終了
(5) 取り出したノードに左につながるノードがあればそれを OPEN リストに入れ，手順 2 へ

5.1.2 コ ス ト

コストには，隣り合うかな 2 文字の頻度 “かな 2-gram(bigram)” を使用する．あるかな k_a の出現頻度 (コーパス内の出現回数) を $C(k_a)$，かな列 k_ak_b の出現頻度を $C(k_ak_b)$，かな k_{i-1} の次に k_i が並ぶ確率を $p(k_i \mid k_{i-1})$ と表記すると，

$$p(k_i \mid k_{i-1}) = \frac{C(k_{i-1}k_i)}{C(k_{i-1})} \tag{1}$$

となる．コスト最小法はマルコフモデルと本質的に同じであるため，かな候補順位付けアルゴリズムでは，かな列 $K(=k_1k_2\cdots k_n)$ の出現確率 $P(K)$ は次のように計算できる．

$$P(K) = \prod_{i=1}^{n} p(k_i \mid k_{i-1}) \tag{2}$$

かな k_a,k_b の隣り合う確率 $p(k_b \mid k_a)$ をコスト値として使用することもできるが，積の繰り返しを行うと誤差が大きくなってしまうため，積を和に変換するために対数をとり，$p(k_b \mid k_a)$ を $p'(k_b \mid k_a)$ に置き換えた値をコスト値とする．

$$p'(k_b \mid k_a) = -log_{10}p(k_b \mid k_a) \tag{3}$$

5.1.1 節のアルゴリズムは，コストが小さい順に経路を見つけるものなので，マイナスをつけて，対数値から得られる負の値を正の値に変換している．式 (2) は次のように置き換えることができる．

$$P(K) = 10^{-\sum_{i=1}^{n} p'(k_b|k_a)} \tag{4}$$

コスト値 $p'(k_b \mid k_a)$ は 5.1.1 節で述べたアルゴリズムで使用することができる．$p'(k_b \mid k_a)$ をコストとして使用する事により，より日本文として尤もらしいかな文字列の順位を高くすることができる．

コーパスとしては「青空文庫[*2]」から新字新仮名遣いで記述された文献を用いた．テキストはかな漢字交じり文で書かれているので，これを形態素解析エンジン MeCab[*3] を用いてかなに変換し，そこから 2-gram を計算した．計算に使用した文字数は約 1 億である．

このコストを用いたときに，打鍵によって生じる文字候補列の例とそれに対するかな候補を図 7 に示す．

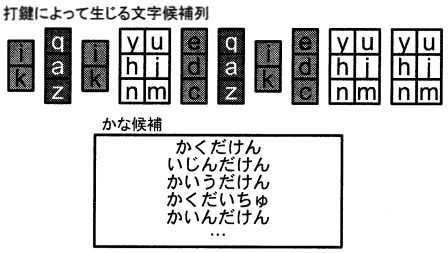

図 7 打鍵によって生じる文字候補列の例とそれに対するかな候補

5.1.3 絞 込 み

入力したいかな文字列の順位が低く，提示されない場合は，文頭から絞込み操作を行うことができる．絞込みはノード単位で行う．システムは絞込みできるノードのリストを，前のノードの最後の文字と，絞込めるノードの 1 文字目のかな 2-gram のコストが低い順に提示する．

5.2 入 力 操 作

全ての操作方法をまとめたものを表 1 に示す．ただし操作によりモードが切り替わる場合には，遷移先のモードを示している．右親指のスイッチにだけ，押し続けている間だけ一時的に別のモードに切り替える機能を割り当てた．残りのスイッチはクリック操作 (押して離すという一連の動作) 後に各機能が実行される．名前に “【】” がついているモードは，右親指を押下することにより遷移できるモードを示す．また，右親指のモード遷移のうち，“■” 記号は右親指を押す操作，“□” 記号は右親指を離す操作によるモード遷移

⋆1 部分最小解により全てのノードで最適な h が求められているため，A アルゴリズムではなく，A* アルゴリズムとなる．

⋆2 http://www.aozora.gr.jp/

⋆3 http://mecab.sourceforge.net/

を表す．入力システムでは，どの状態でも常に操作方法が示されているので，熟練するまではそのガイド表示を見ながら入力することが可能になっている．

他に特記すべき事項を以下に列挙する．

- 各指の打鍵による機能はモードごとにある程度共通化されている
- どのモードでも，右親指を押下した状態で左親指を打鍵することで，バックスペース (BS) もしくは 1 段階前のモードに戻るという操作に対応する
- カーソルモードでは，確定済みの文字列表示に対して，カーソル移動や消去を行うことができる
- 絞込み状態を維持しながらの打鍵列の追加や削除ができる
- かな入力モードでのバックスペースは，1 文字の消去ではなく，1 打鍵の消去を行う
- 絞込みモードで左中指を打鍵することにより，確定前の入力が全てキャンセルされ，未入力モードへ遷移することができる
- かな候補選択モードでは，入力したいかな候補を選択した状態で，左小指や左薬指を打鍵することにより，ひらがなでの無変換確定やカタカナでの無変換確定を行うことができる
- 漢字変換操作では，文節の移動や，文節の伸縮を行うことができる
- 未入力モードで左親指を打鍵することにより，記号選択モードへ遷移でき，“、”や“。”などの記号を入力することができる

6. アルゴリズムの性能評価実験

かな候補の順位付けアルゴリズムの性能を検証するために，あるかな文字列 A に対して打鍵列を生成し，そこからかな候補を生成したときの候補の中での A の順位の具合を調査した．調査候補とした文字列の集合は，辞書 ipadic [*1] に存在する単語の読みを抽出した集合と，ランダムに生成したかな文字列の集合である．

まだ絞込みされていないノードの数 (= 文字列 A のノード数 − 絞込み回数) を残り絞込み回数としたとき，単語集合の場合の残り絞込み回数に対する各ノード数の 10 位以内出現確率の変化を図 8 に示す．ただし，ノード数が 12 までの場合のみ表示している．10 位以内出現確率とは，文字列 A がその絞込み時点で 10 位以内に出現した割合のことである．10 位以内の根拠は，被験者実験での 1 ページあたり候補表示数が 10 個であること，つまり，10 位以内出現確率は文字

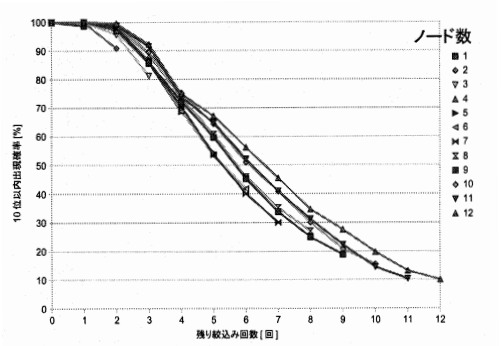

図 8 単語集合の残り絞込み回数によるノード数別出現確率の変化

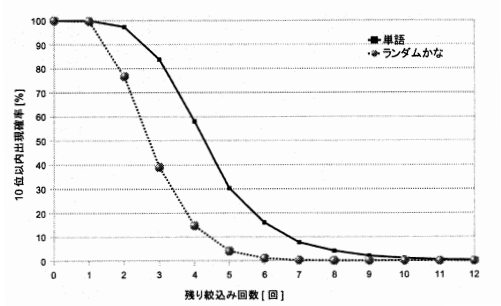

図 9 残り絞込み回数による出現確率の変化

列 A がその絞込み時点で候補の 1 ページ目に表示される確率である．どのノード数の場合も，同様の傾向が見られ，残り絞込み回数について調べた場合，出現確率に影響を与えるパラメータにはノード数は関係無いことが確認できたので，ノード数について統合することができる．全てのノード数について統合した結果を図 9 に示す．

単語集合が対象の場合，残り絞込み回数 5 回の場合には約 30%の確率，残り絞込み回数 4 回になると，約 60%の確率で 10 位以内に対象文字列が得られることがわかる．ランダムかな集合と比べると，残り絞込み回数 3〜5 の場合に，約 25〜45 ポイント高くなっていることがわかる．

7. 被験者実験

Guesso(机の上で入力)，ミニキーボード diNovo mini [*2] (両手で把持し両手親指のみで入力を行うタイプ)，フルキーボードによる各 3 セッションからなる被験者入力実験を行った．各セッションでは漢字かな交じり文の短文 (平均 20 文字) の入力を 20 回行う (合計 400 文字)．各短文の入力では，確定時に例文と

*1 http://chasen.aist-nara.ac.jp/chasen/distribution.html.ja

*2 http://www.logicool.co.jp/index.cfm/keyboards/keyboard/devices/3848&cl=jp,ja

表 1　各モードでの操作

モード	左小指	左薬指	左中指	左人差指	左親指	右親指	右人差指	右中指	右薬指	右小指
未入力	[QAZ]	[WSX]	[EDC]	[RFVTGB]	記号選択モード	■カーソルモード	[YHNUJM]	[IK]	[OL]	[P-]
【カーソル】	HOME	END	↑ or ←	↓ or →	BS	□未入力モード	←	→	Enter	Enter
記号選択	確定	確定	次ページ候補	前候補	次候補	■記号解除モード	確定	確定	確定	確定
【記号解除】	×	×	×	×	未入力モード	□記号選択モード	×	×	×	×
かな入力	[QAZ]	[WSX]	[EDC]	[RFVTGB]	かな選択モード	■絞込みモード	[YHNUJM]	[IK]	[OL]	[P-]
かな候補選択	かな無変換確定	カナ無変換確定	次ページ候補	前候補	次候補	■絞込みモード	漢字変換モード	漢字変換モード	漢字変換モード	漢字変換モード
【絞込み・消去】	絞込み1字解除	絞込み全解除	未入力モード	次頁絞込み候補	BS	□かな入力モード	絞込み候補1	絞込み候補2	絞込み候補3	絞込み候補4
漢字変換	×	前ページ候補	次ページ候補	前候補	次候補	■文節調整へ	確定	確定	確定	確定
【文節調整】	先頭文節選択	末尾文節選択	左文節選択	右文節選択	かな選択モード	□漢字変換モード	左文節選択	右文節選択	文節を縮める	文節を伸ばす

異なる入力が行われている場合は確定操作を受け付けず，次の短文の入力を行うことができない．カーソル移動やバックスペース等による修正はいつでも可能である．目標の漢字が変換で出てこない場合などは，その漢字を含む熟語を入力してから不要な部分を消去するというような入力も可能である[*1]．

被験者はタッチタイピングに熟練した学生 9 名 (男性 8 名，女性 1 名) である．全員が QWERTY キーボードのタッチタイピングに熟練している．漢字変換エンジンには学習機能を切った Anthy [*2] を使用した．事前に Guesso は 17 回，ミニキーボードは 10 回の学習セッションを行っている．

実験は，Guesso の学習セッション，Guesso の比較セッション，ミニキーボードの学習セッション，ミニキーボードの比較セッション，フルキーボードの比較セッションの順で行った．

7.1　結果と考察

グラフに表示したエラーバーは，被験者間の標準偏差を表す．有意差の検定に関しては，特に明記が無い限り有意水準は $p < 0.05$ とする．

7.1.1　Guesso の学習結果

比較セッションの 1，2，3 セッション目を，学習セッションの 18，19，20 セッション目とみなして，学習による効果の考察を行う．

Guesso の学習セッションでの，各被験者の平均入力速度 (KPM) の変化を図 10 に示す．KPM は，“漢字かな混じり文字数 / 入力時間 [分]” によって得られる値である．ここで，入力時間は，提示された短文の入力開始から修正も含めて完了までにかかった時間で，修正に要した時間を含む．第 1 セッションでの平均速度は約 13KPM 程度だが，第 15 セッションあたりで約 30KPM 程度まで上昇し，その後は停滞している様子が分かる．入力速度に関しては非常に早く成熟が見られていると言える．ある被験者の最大入力速度とし

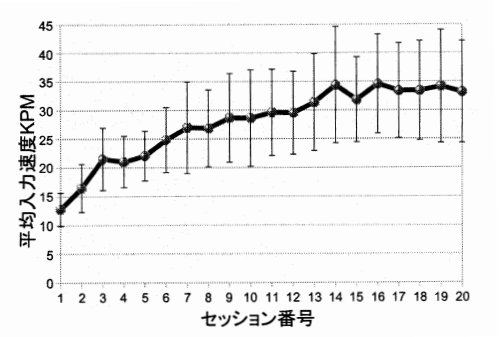

図 10　各被験者の入力速度の変化

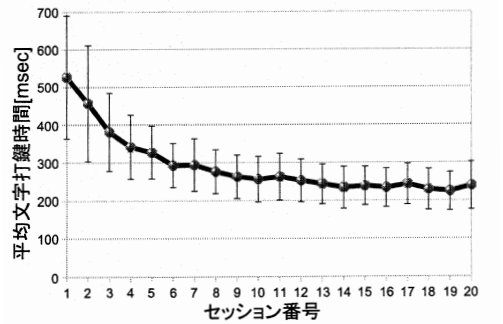

図 11　各被験者の平均文字打鍵時間の変化

ては，50.9KPM という結果が得られている．

各被験者の平均文字打鍵時間の変化を図 11 に示す．文字打鍵時間とは，通常の文字打鍵（変換操作やカーソル操作以外）が連続した場合のそれぞれが行われた時刻の間隔のことである．第 1 セッションではばらつきが大きいが，10 セッション程度で収束している様子が分かる．デバイスによる打鍵にも短い学習期間で慣れることができるといえる．

7.1.2　各手法の比較

各手法の全被験者の平均入力速度 KPM を図 12 に示す．手法を要因とする 1 要因の被験者間分散分析を行ったところ，主効果が有意 ($F(2,8) = 36.7, p < 0.001$) であった．さらにライアン法による多重検定を行ったところ，各手法間全てに有意差が認められた．Guesso の入力速度は 33.5KPM であり，ミニキーボードに対

*1 例:“攻” と入力したいときに，“こうげき” を変換して “攻撃” とし，“撃” を消去する．

*2 http://anthy.sourceforge.jp/

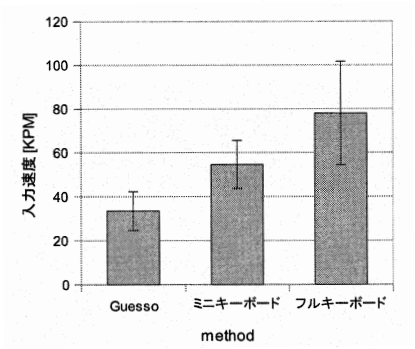

図 12 入力速度の比較

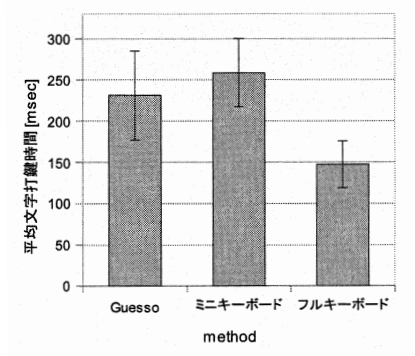

図 13 文字打鍵時間の比較

しては 61.3%，フルキーボードに対しては 42.9%の速度が得られた．Guesso の最も良い被験者の結果としては，フルキーボードに対して 65.6%の速度が得られた．このことから，ある程度の実用に値するような入力速度を得られることを確認した．しかし，ミニキーボードよりも入力速度が遅いという結果については，設計方針の “フルキーボードに近い入力速度が得られる” を達成してるとは言いにくい．

各手法の全被験者の平均文字打鍵時間を図 13 に示す．手法を要因とする 1 要因の被験者間分散分析を行ったところ，主効果が有意 ($F(2,8) = 43.8, p < 0.001$) であった．さらにライアン法による多重検定を行ったところ，フルキーボードとミニキーボード，Guesso とフルキーボードには有意差が確認されたが，Guesso とミニキーボードに関しては有意差が確認されなかった．Guesso はミニキーボードに対しては 89.3%，フルキーボードに対しては 157%の打鍵時間が得られた．有意差は無いものの，ミニキーボードよりは若干良い値が得られたが，理想的には，フルキーボードと同等の値が得られることが望まれる．今後の課題にしたい．

7.1.3 Guesso の詳細解析

Guesso は，文字打鍵時間が長い割に，入力速度は

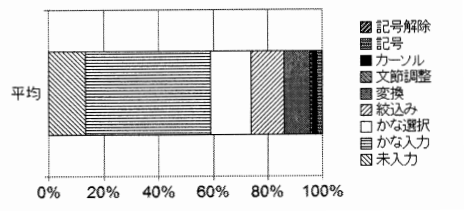

図 14 各モードの時間割合の平均

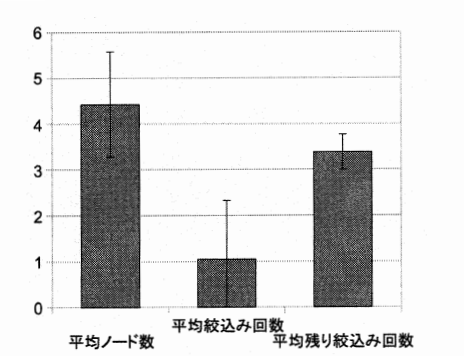

図 15 平均ノード数，平均絞込み回数，平均残り絞込み回数

遅かった．この原因を調べるため，比較セッションの各モードの時間割合を調べた結果，図 14 のようになった．この結果から，かな候補の選択や絞込みに時間がかかっていることがわかった．このことから，かな候補の順位付けアルゴリズムの改良や，かな選択を経由しない直接漢字候補提示の実装により，より高速入力できる日本文入力手法が得られると考えられる．

Guesso で漢字確定があったとき，その確定に必要となったかなのノード数と絞込み回数，残り絞込み回数を調べた．比較セッションの各被験者の平均ノード数，平均絞込み回数，平均残り絞込み回数を調べた結果，図 15 のようになった．ノード数の平均値は 4.42 回，絞込み回数は 1.05 回，残り絞込み回数は 3.38 回となった．ほとんどの被験者の平均絞込み回数は 1 回以内となっており，絞込みによって候補数を抑えるよりは，短い文字列を打鍵して候補数を抑える傾向が強いことが伺える．平均残り絞込み回数に注目すると，どの被験者も約 3〜4 回という結果が得られている．これは，図 9 に示した 10 位以内出現確率での約 55〜85%の高い確率の部分に相当していることがわかる．

7.1.4 アンケート結果

実験終了後，Guesso について記述回答形式のアンケートを行った．アンケートの各設問について回答をまとめ，考察する．

Guesso のデバイスについて

“ゴムによる反発力の感触は何かの役に立ったか” と

いう設問については，全ての被験者が，反発力によるフィードバックが役立ったと回答した．デバイスにクリック感を与えたことは有効であったと言えるだろう．

“装着の仕方についてどう思うか” という設問については，8名の被験者が，装着しにくいと答えた．装着しにくい理由としては，ケーブルが煩わしい，フィンガーピックが指の上で回転して使用しにくい，サイズが指に合わない，各指に1つずつ装着するのが面倒などが挙げられた．装着が難しいことはウェアラブルデバイスとして致命的な問題なので，今後必ず改良すべき問題点である．

“フィンガーピック形状の他に使いやすそうな形状はあるか” という設問については，装着しやすい手袋形状を挙げるコメントが多かった．他にも，手袋と指先ベルトを組み合わせて，スイッチ位置の調整や指先への固定をしやすくする，接触スイッチではなく何かのセンサで打鍵を検知する，空中で入力できるように，指先の曲げを使用する，というアイディアが得られた．

その他にも，接触による触覚フィードバック以外にも何かのフィードバックがあるほうが良いというコメントが得られた．ウェアラブル装置としては，音のフィードバックは人ごみでは騒音となりうるので，振動フィードバックの採用が有用ではないかと考えられる．

Guesso のシステムについて

“日本文の入力に連文節変換は必要だと思うか” という設問に対しては，全ての被験者が，必要である，もしくは，あったほうが良いと答えた．

“速く入力するためのコツはあるか” という設問については，文節ごとなど短い文字列ごとに確定するとよい，リズムよく打鍵する，通常のキーボードでのタッチタイピングをイメージして打鍵する，かな無変換機能を多用する，といったコメントが得られた．短い文字列で入力すると入力しやすいというコメントは多かったが，これは，連文節変換を行えるようにしたという Guesso の特徴を薄めてしまっている．長い文字列でも入力しやすくするという改良も必要であると考えられる．

Guesso の入力手法に関連するアイディア

指使いを覚えられるため，タッチタイピング初心者の学習に良いのではないかというコメントや，POBox のようにまだ入力していない部分を辞書によって補完する機能を組み合わせるというアイディアが得られた．他にも，タッチパネル上での入力や，仮想キーボードとしての利用案を得ることが出来た．

7.1.5 そ の 他

文字入力に関する実験では，入力速度の他にエラー率が重要な評価項目となるが，今回の入力実験では，修正や，修正を応用した形での入力を許可したので，エラー率という独立した値の導出は非常に難しかったため行わなかった．ただし，入力速度 KPM の値には修正に要した時間も含まれるため，入力速度 KPM は入力速度とエラー率の両方を含んだ値として捉えることもできる．

8. おわりに

本研究では，モバイル環境での使用を想定した日本文入力手法 “Guesso” を提案し，実装，評価を行った．その結果，独自機能の連文節変換を実装し，ある程度の実用に値するような入力速度を得ることができた．しかし，フルキーボードのような高速入力には及ばず，改善すべき課題が見つかった．

また，被験者は連文節変換を必要としているが，短い文字列の方が入力を確定しやすく，連文節変換を生かしきれなかった，という考察を得ることができた．今後の改善点としたい．

展望としては，3-gram(trigram) や形態素数パラメータ (形態素数が少ない文字列のほうがより日本文らしいというアイディアから，外部ソフトによる形態素解析を行い，形態素数を順位付けのパラメータとして併用する) の採用による，かな候補順位付けの改良，かな候補の選択を経由しない直接漢字候補提示の実装，太股の上や歩行中などのより実際的なウェアラブル環境での入力による評価を行いたい．

参 考 文 献

1) HandyKey Corporation:Twiddler2, http://www.handykey.com/
2) i-Tech Dynamic:Virtural Keyboard, http://www.virtual-laser-keyboard.com/
3) 福本, 平岩, 曽根原: “FingeRing”: A Keyboard Device for Wearable Computers, 電子情報通信学会 Vol.J79-A, No.2,pp. 460-470,1996.
4) Nuance Communications: T9, http://www.nuance.com/t9/
5) 増井俊之: ペンを用いた高速文章入力手法, インタラクティブシステムとソフトウェア IV, 日本ソフトウェア科学会 WISS ’96, pp. 51-60,1996.
6) 青木亮磨:運指情報を利用した推測による日本語入力手法 “FtoKey” の提案と評価, 電気通信大学修士論文,2007.
7) 永田昌明:前向き DP 後向き A* アルゴリズムを用いた確率的日本語形態素解析システム, 情報処理学会研究報告,94-NL-101-10,pp.73-80,1994.