キーボードの打鍵情報を活用した図形型コマンド入力方式

片山 拓也
村尾, 和哉
寺田, 努
塚本, 昌彦

(Citation)
ヒューマンインタフェース学会論文誌, 14(1-4):167-176

(Issue Date)
2012

(Resource Type)
journal article

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/90002134

Vol.14 No.2, 2012

原著論文

キーボードの打鍵情報を活用した図形型コマンド入力方式

片山拓也[*1] 村尾和哉[*1] 寺田 努[*1*2] 塚本昌彦[*1]

A Shape Command Input Method using Key Entry Information of Physical Keyboard

Takuya Katayama[*1], Kazuya Murao[*1], Tsutomu Terada[*1*2] and Masahiko Tsukamoto[*1]

Abstract – **Keyboard is mainly used for text input, and has function keys and shortcut keys to reduce the number of manual operation. However, it is difficult and troublesome to memorize relations between keys and functions for beginners. In this paper, we propose two intuitional input methods on the physical keyboard in addition to usual *texttyping*: *stroke* that traces a shape on the keyboard, and *stamp* that presses a shape on the keyboard. Our system automatically classifies user inputs into texttyping, stroke, or stamp from key entry information, enabling us to seamlessly input those commands without additional devices.**

Keywords : **キーボード，コマンド入力，パターン認識**

1. はじめに

現在，キーボードはその入力速度と操作性の高さからコンピュータの入力デバイスとして不可欠なものとなっている．キーボードは主に文字入力に用いられるが，素早く機能を実行するためにファンクションキーやショートカットキーと呼ばれる入力方式が用いられる．ファンクションキーは文字キーとは別の汎用キー（「F1」など），ショートカットキーは「Ctrl」などの修飾キーと文字キーの組合せ（「Ctrl+C」など）であり，それぞれにコンピュータの機能や動作を割り当てられる．しかし，これらの入力方式にはユーザへの敷居を高くする要因がある．第一の要因は機能に対応する入力キーの記憶にある．ある程度習熟したユーザは，頻繁に用いる機能の入力キーを経験から記憶しているが，コンピュータの習熟度が低いユーザにとってそれらの記憶は煩わしい．第二の要因はキー位置の把握にある．ショートカットキーには “Copy” の頭文字から容易に連想される「Ctrl+C」のような入力も存在するが，入力に用いるキーの位置を把握しておかなければ素早い入力は行えない．そのため，キー配列を覚えていないユーザにはファンクションキーやショートカットキーの有効性が最大限発揮されない．

そこで，本論文では新しい入力方式として，キーボード上で行う図形型コマンド入力を提案する．図形型コマンド入力はその直観性からマウスジェスチャや加速度センサを用いたジェスチャ入力などに幅広く用いられている．我々は，通常の文字入力を行う「タイピング」に加えて，キーボードの上をなぞる「ストローク」，およびキーボードの範囲を押す「スタンプ」の 2 種類の図形型コマンド入力を提案する．複数キー上で指を滑らせたり，同時に周囲の複数のキーを押すという入力は，従来キーボードでは行われていなかったが，ピアノなど他のボタン型道具では採用されている入力方法である．提案方式は，それぞれ指の軌跡と手の形状を入力するので，機能から連想される図形を直観的に関連付けて使用できる．直感性には大きく二つのレイヤが存在する．一つは使用するデバイスと操作の対応関係における直観性であり，もう一つは操作の意味と機能の対応関係における直観性である．提案システムでは，従来は一つ一つのキーを押して使用していたキーボードの使用方法とは異なる操作を行うため前者の直観性は考慮していない．後者の直感性は，ジェスチャを用いて下方向に動かせば操作の対象が下方向への影響を受けるように，ユーザが操作と機能の対応関係を連想できることから実現している．また，提案方式は入力場所に依存しないので使用に際してキー配列を覚える必要はない．さらに，提案方式は，入力の特性から「タイピング」,「ストローク」,「スタンプ」のうち，どの入力が行われたのかを自動的に識別するため，文字入力とコマンド入力をモード切り替えなしでシームレスに使用できる．本論文ではコンピュータを所持しているが，使用時間が一日一時間にも満たないような者を初心者と定義する．提案システムを用いることで，初心者は現在所持している機器を用いて，コストをかけずに直観的な入力をキーボードに導入できる．また，提案システムは幅広い習熟度のユーザに対応できるように，単純なインタフェースやタイムラグの低減を目指す．本論文ではシステムのプロトタイプ

*1: 神戸大学大学院工学研究科
*2: 科学技術振興機構さきがけ
*1: Graduate School of Engineering, Kobe University
*2: PRESTO, Japan Science and Technology Agency

としてコマンドと機能を関連付けるインタフェースを実装し，ユーザ評価によって識別精度や使い勝手を調査した．

以下，2 章で関連研究を紹介し，3 章で構築したコマンド入力方式について説明し，4 章で提案システムの評価について述べる．そして，5 章で提案システムの応用例を述べ，最後に 6 章で本論文のまとめを行う．

2. 関連研究

本章では，キーボードにおける入力機能拡張，キーストロークダイナミクスの利用，図形型コマンド入力についての関連研究を述べる．

2.1 キーボードにおける入力機能拡張

これまでにキーボードに文字入力以外の様々な機能を拡張する研究が数多く行われている．これらの機能拡張によってキーボードの操作回数やデバイス間の手の移動回数を減らせる．以下にいくつかの例を挙げて説明する．

ポインティング機能を付加する研究として，Pointing Keyboard [1] がある．これはキーボード上に 2 次元座標検出のための赤外線センサを重ねた構造になっており，キーボードを「押す」動作と「なでる」動作の違いを検出し，キーボード面上でキー入力とポインティングの両操作を可能にする．また Touch-Display Keyboards [2] では，各キーの上面にディスプレイとタッチセンサを取り付けることでキーにボタンやウェブリンク，イメージなどを自由にマッピングし，表示している．さらに，タッチセンサを使ったジェスチャ入力や，キーボードのディスプレイに作業中の画面を表示することでキーボード上でポインティングを行える．他に市販されているものに FingerWorks 社の TouchStream [3] がある．これは従来のキーボードの操作に倣って盤面を叩けば文字入力に利用できるが，2 本指でなぞるとポインティング，3 本指で叩けばダブルクリックなど数十種類の入力が可能なキーボードである．しかし，これらのシステムは専用の盤面を設計しており，従来のキーボード上では使用できない．それに対して，提案システムは既存のキーボードから得られる打鍵情報を用いるため，新たな装置を必要としない．

ThumbSence [4] は，ノート PC 用のポインティングデバイスとして広く普及しているタッチパッドの左右のマウスボタンの操作時にキーボードのホームポジションから手が離れるという欠点を解決する技術である．具体的には，タッチパッドに指が触れている間は，キーボードにマウスボタンの機能やコマンド入力を割り当てる．これにより，キーボードのホームポジションに手を置いたまま様々な操作が可能になるが，キーとコマンドの割当てを覚えるのは煩雑である．同様のことがショートカットキーにも言える．

ExpressiveTyping [5] は，ノート PC の落下の衝撃を感知してハードディスクを緊急停止させるために搭載される内蔵型加速度センサの値を用いて打鍵圧を取得し，打鍵圧に応じたフォントサイズの変更を行うシステムである．この技術のように，ユーザが直観的に操作できるシステムが提案されており，提案システムの取組みもその一環といえる．

2.2 キーストロークダイナミクス

キーボードのどのキーがいつ押されていつ離されたかといった打鍵情報はキーストロークダイナミクスと呼ばれ，様々な研究に応用されている．主な応用例として認証がある．キーストロークダイナミクスは個人ごとに異なり，手書きのサインと同レベルの識別精度がある [6]．多くのキーストロークダイナミクス認証が固定テキストを対象にしているのに対して，ユーザが自由に入力したテキストを用いて認証を行う機構も提案されている [7]．この認証方法にはプライバシーに関わる生体情報ではなくて後天的な情報を用いること，万が一他人に知られたとしても模倣が困難であることなどの利点がある．さらに，ユーザの感情の状態を認識する試みも行われている [8]．この手法では，キーストロークダイナミクスを用いて 7 種類の感情の状態を 2 レベルで認識できる．

提案システムでは，キーストロークダイナミクスを用いて，ユーザが「タイピング」,「ストローク」,「スタンプ」のうちのどの入力を行っているのかを認識する．

2.3 図形型コマンド入力

図形型コマンドはユーザが機能から連想される図形を入力する直観的なコマンド入力として，マウスジェスチャや加速度センサを用いたジェスチャ入力など広い範囲で用いられている．マウスジェスチャはマウスボタンと上下左右のカーソル移動の組み合わせの入力で，例えば，マウスボタンを押しながら左右に動かすことで Web ブラウザ上で「1 ページ進む/戻る」の機能を実行する．加速度センサを用いたジェスチャ入力では，例えば，センサが搭載されたデバイスを使って右向三角を描くと再生し，四角を描くと停止を行うといった機能を有する動画再生アプリケーションが提案されている．このような図形型コマンドはユーザが使用するデバイスに応じて使い分けることでスムーズに入力できる．マウスジェスチャはウェブページ閲覧などマウス操作が中心の作業中にコマンドを入力する際に適したものであり，それに対して，提案システムは文章の入力やメールの作成などキーボード操作が中心の作業中にコマンドを入力する際に適している．さらに，提案システムはマウスが使用できないなど，デバ

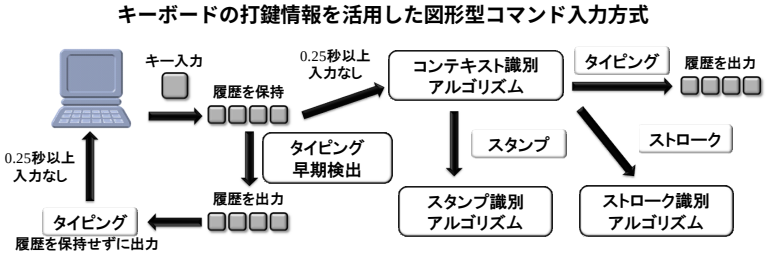


図 1　システムの動作フロー
Fig. 1　Operation flow of proposal system

イスの使用に制限がある環境でも有用である．

キーボードに図形型コマンド入力を組み合わせた例に SHARK と呼ばれる入力を用いたコマンド入力 [9] がある．SHARK はソフトウェアキーボード上の文字入力手法で，キーをタップするのではなく，目的のアルファベットキーをつなげるように一筆書きの要領でなぞって入力する．システムはその入力を図形として扱い，何の単語が入力されたのかを認識する．SHARK で「Ctrl」から始めて，機能名の全部，あるいは一部を入力するとコマンド入力として扱う．この入力方式によってソフトウェアキーボード上でのコマンド入力の入力速度は向上したが，これを用いるためにはキー配列を把握している必要がある．これに対して，提案システムはキー配列を覚えることなく使用できる．また，提案システムは物理キーボードを用いた入力方式なので，粗い解像度の図形を扱う必要がある．

3.　コマンド入力方式の設計

3.1　システム概要

本論文では，物理キーボード上でストロークとスタンプの 2 種類の図形型コマンド入力を実現するシステムを提案する．ストロークはキーボードの上を一筆書きの要領でなぞるように順に押して入力し，スタンプはキーボードの範囲を目的の形状で同時に押して入力する．つまり，それぞれが指の軌跡，手の形状の図形を入力するコマンドであると言える．提案システムの特徴を以下に挙げる．

- 直観性
  コンピュータの用語を知らないユーザでも，機能から連想される形を入力することでコマンドを実行できる．例えば，ストロークの利用例として，キーボード上を左右になぞってブラウザのページの履歴を戻ったり進んだり，「○」を描くようになぞることでお気に入り登録を行うといったものが考えられ，スタンプの利用例として，手を開いた状態でキーボードを押してウィンドウの最大化をして，手を握った状態でキーボードを押してウィンドウを閉じるといったものが考えられる．
  機能から連想される形は多数存在するが，同一の人物は特定の機能からは同一の図形を連想すると考えられる．また，中には連想し難い機能も存在するので，提案手法は各ユーザが提案コマンドを直観的に連想できる機能を選択的に用いるのが好ましいと考えられる．
- コマンド自動識別
  提案システムでは，2 種類のコマンド入力動作を文字入力動作と識別する機能をもっている．そのため，コマンドをシームレスに入力でき，モード切り替えの煩雑さも解消され，スムーズにコマンド入力できる．
- キー非依存
  提案システムでは，コマンドの識別の際に入力されたキーの座標列を場所に依存しない特徴量に変換しているため，入力キーに依存せずに使用できる．この特徴も利用者に対するコマンドの敷居を下げる要因の 1 つになり，各キーの位置を把握していない初心者でもコマンドを容易に扱える．
- 書き順制約
  ストロークを識別する際に，入力された図形に書き順の特徴量を加えた識別アルゴリズムを採用している．これによって，同一の図形を入力しても書き順を間違えると別のストロークと識別される．しかし，同じ水平方向の直線でも「右から左に伸びる直線」と「左から右に伸びる直線」を区別でき，入力の方向を考慮したコマンドを登録できる．

(a) ストローク　　(b) スタンプ
図 2　提案システムの利用例
Fig. 2　The usage of the propoed commands

3.2　システム構成

提案システムの動作フローを図 1 に示す．前述した通り，タイピングは単語や文章を入力している状態，ストロークは図形を一筆書きで描くように打鍵した状態（図 2(a)），スタンプは手の形を保ってキーボードを複数キー同時に押した状態（図 2(b)）を表す．以下，この 3 種類の状態をコンテキストと呼ぶ．

提案システムでは，シームレスな入力実現のためにストロークやスタンプが入力された際にキーの入力

を無効化してコマンドに置き換える必要がある．そこで，キーイベントが発生したら実行せず一時的にシステムで保持する．以下，保持したキーイベント列を履歴と呼ぶ．その後，キーイベントが一定時間（初期値は 0.25 秒に設定）発生しなければ，その時点までの履歴のコンテキストを「コンテキスト識別アルゴリズム」を用いて識別する．履歴のコンテキストがタイピングと判断された場合は保持した履歴を出力し，ストローク，スタンプと判断された場合はそれぞれ「ストローク識別アルゴリズム」,「スタンプ識別アルゴリズム」を用いて入力された図形を識別し，判定された図形コマンドに関連付けられたイベントを発生させる．ここで，タイピングをしているにも関わらず文字を出力せず履歴を蓄積すると，キー入力が画面に反映されずにユーザに違和感を生じさせるため，タイピングに関しては可能な限り早期に検出する必要がある．そのため，提案システムは 1 打鍵ごとにタイピングか否かを判断してタイピングを早期で検出する「タイピング早期検出機構」を備えている．ここで，タイピング早期検出機構もコンテキストを識別する機構だが，混乱を防ぐため，以下では入力の切れ目にタイピング，ストローク，スタンプの 3 種を識別する機構をコンテキスト識別アルゴリズム，1 打鍵毎にそれまでの入力がタイピングかどうかを検討してタイピング時のタイムラグを最小化する機構をタイピング早期検出機構と定義する．以下，コンテキスト識別アルゴリズム，タイピング早期検出機構，ストローク識別アルゴリズム，スタンプ識別アルゴリズムについて詳しく述べる．

3.3　コンテキスト識別アルゴリズム

提案システムでは，新たに特別な装置を用いることなく，履歴に含まれるキー座標とキー押下時間の特徴量を抽出してコンテキストを識別する．識別のための予備実験として，6 名の被験者のタイピング，ストローク，スタンプの 3 種類の入力を 30 回ずつ行った場合のキー入力履歴を収集した．ここでタイピングは自由に文章を打った時，ストロークとスタンプは被験者が思いつく自由な図形を入力した際に得られた履歴を用いた．得られた入力履歴から，以下に示す特徴量を抽出した．

・t_{Press}：各キーの押下時間の平均
・$t_{Interval}$：キーダウンからキーダウンまでの間隔の平均
・t_{Span}：最初のキーダウンから最後のキーダウンまでの時間
・$c_{Distance}$：キー遷移の移動距離の平均
・$c_{Deviation}$：キー座標の分散
・n_{Once}：同時に押されたキーの最大値
・n_{All}：履歴に含まれるキー数

図 3　キーと座標の対応
Fig. 3　The key coordinate

被験者は，男性 5 名女性 1 名で男性の中には左利きの者を 2 名含み，いずれもコンピュータを日常的に使用するものである．ストロークやスタンプの入力の特徴にはコンピュータの習熟度ではなく，ユーザの利き手や手の大きさ，指の太さといった身体的特徴が影響を与える．また，タイピングは入力速度が高いほど 1 つの履歴に複数のキーイベントが含まれるため高度な識別が要求される．そこで，本予備実験では，より習熟度が高いユーザの使用にも耐えうる設計のために，本研究で定義した初心者よりも入力速度が高い被験者を採用した．

抽出された特徴量の平均，標準偏差，最大，最小を表 1 に示す．ここで，図 3 に示すように，キー座標 (x,y) は (0,0)〜(75,25) の範囲であらかじめマッピングしており，$c_{Distance}$，$c_{Deviation}$ は，入力文字の座標列 $S = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ が与えられた時にそれぞれ以下の式で求められる．

$$c_{Distance} = \frac{1}{(n-1)} \sum_{k=1}^{n-1} \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2}$$

$$c_{Deviation} = \frac{1}{2n} \sum_{k=1}^{n} (x_k - \bar{x})^2 + (y_k - \bar{y})^2$$

$$\left(\bar{x} = \frac{1}{n} \sum_{k=1}^{n} x_k, \quad \bar{y} = \frac{1}{n} \sum_{k=1}^{n} y_k \right)$$

また，n_{Once} はあるキーのキーアップイベントの前に別のキーのキーダウンイベントが発生した際に同時打鍵とし，$n_{All} = 1$ の時には特徴量として t_{Press} しか計算できないので別項目として扱い，$t_{Interval}$，t_{Span}，$c_{Distance}$，$c_{Deviation}$ の計算の際には除外する．

表 1 より，以下の特徴があることが分かった．
・t_{Press}：タイピングに比べてストローク，スタンプの値が小さくなった．これは，タイピングの際は一つ一つの入力キーが意味をもつのに対して，ストローク，スタンプの際は入力したキーのそれぞれは意味をもたず，入力動作の一部であることが原因と考えられる．
・$t_{Interval}$：タイピングに比べてストロークとスタンプの際の平均値が小さく，特にスタンプの際は全てのキーを同時に打鍵するため，極めて小さい値をとった．
・t_{Span}：スタンプの際の値が小さい．これはスタンプ入力時の $t_{Interval}$ が小さい理由と同様である．
・$c_{Distance}$：ストロークとスタンプの際の平均値が小さく，ストロークの際の標準偏差が非常に小さい．ストローク入力時は指をキーボード上で滑らせて入力す

表 1　予備実験結果
Table 1　The result of pilot study

項目	入力方式	平均	最小	最大	標準偏差
t_{Press} (msec) ($n_{All} \geqq 2$)	タイピング	110.7	60.5	411.5	30.3
	ストローク	84.3	36.3	208.3	27.0
	スタンプ	72.1	10.3	277.8	67.3
t_{Press} (msec) ($n_{All} = 1$)	タイピング	100.2	52.0	226.0	27.6
	ストローク	–	–	–	–
	スタンプ	25.9	9.0	131.0	37.1
$t_{Interval}$ (msec)	タイピング	146.2	40.0	358.0	50.8
	ストローク	55.2	16.9	152.9	27.4
	スタンプ	8.2	0.0	107.5	12.0
t_{Span} (msec)	タイピング	773.0	40.0	8736.0	854.3
	ストローク	695.9	77.0	2175.1	415.9
	スタンプ	24.0	0.0	215.0	35.3
$c_{Distance}$	タイピング	20.1	0.0	53.2	10.8
	ストローク	6.0	5.1	7.6	0.3
	スタンプ	8.5	3.6	16.4	2.6
$c_{Deviation}$	タイピング	12.7	0.0	32.0	6.2
	ストローク	10.2	4.6	17.1	2.7
	スタンプ	5.5	1.8	11.5	2.1
n_{Once}	タイピング	1.6	1.0	3.0	0.6
	ストローク	3.7	1.0	10.0	1.7
	スタンプ	3.6	1.0	8.0	1.4
n_{All}	タイピング	5.1	1.0	68.0	5.9
	ストローク	13.9	4.0	39.0	6.2
	スタンプ	3.8	1.0	11.0	1.6

るため常に隣接するキーに移動しており，スタンプ入力時は片手で入力しているため片手の範囲内のキーのみが押されるためである．

・$c_{Deviation}$：スタンプの際の平均値が小さく，ストロークとスタンプの際の標準偏差が小さい．

・n_{Once}：ストロークとスタンプの際の最大値が大きいため，平均値も大きい．

・n_{All}：ストロークの際の最小値が大きい．これは，形をなぞる動作では一定以上のキーを押す必要があるためである．また，スタンプの際の値の平均が少ない．これは，一般のキーボードでは内部の電気回路的な制約により，押下したキー全ては入力できないことが原因として挙げられる．通常，安価なキーボードでは 3 キー程度までの同時押しまでしか保証されていない設計になっており，全てのキーの同時押しを読み込むためには全てのキーに電流逆流防止のダイオードを入れる必要がある．

得られた特徴を考慮して，コマンド入力を識別するアルゴリズムを以下のように設計した．なお，$n_{All} = 1$ の時は抽出できる特徴量が t_{Press} のみでタイピングとスタンプを完全に識別できないため，使用中の大部分を占めると予想されるタイピングを優先してパラメータを設定した．また，同様の理由から，$n_{All} \geqq 2$ の時のスタンプの識別の閾値も予備実験の $t_{Interval}$，t_{Span} の最大値よりも低く設定した．

```
if (n_All = 1) {
  if (t_Press > 50) typing
  else stamp
}else {
  if (t_Interval > 15 and t_Span > 75
    and c_Distance < 8) stroke
  else if (t_Interval < 80 and t_Span < 200) stamp
  else typing
}
```

3.4　タイピング早期検出機構

コンテキスト識別アルゴリズムのみでコンテキストの識別を行うと，キーストローク間隔が一定時間以上開くまでキー入力のコンテキストが認識されないので，タイピングを行った際には，入力が途切れるまで入力が画面に反映されない．具体的には，表 1 の t_{Span} の項に入力待ちの時間（0.25 秒に設定）を加えた分の遅延，つまり，平均約 1 秒，最大で約 9 秒の遅延が発生する．キーボードの使用時間の大部分のコンテキストはタイピングであると考えられるため，このタイムラグがユーザに与えるストレスは甚大なものである．そこで，入力の途切れを待たずに逐次的にタイピングの検出を行い，入出力のタイムラグを解消するタイピング早期検出機構を以下のように設計した．

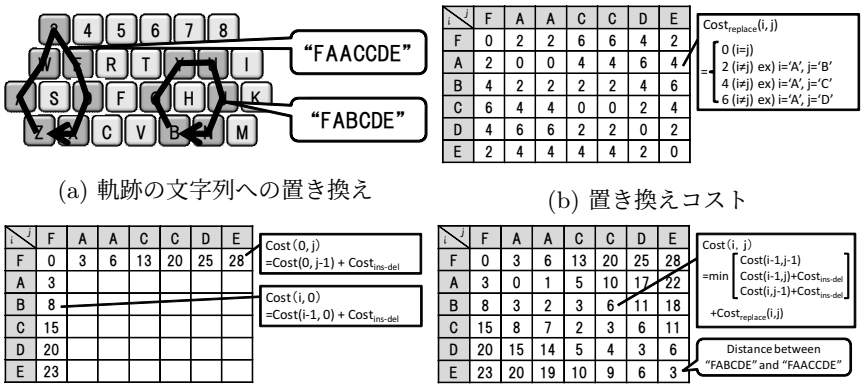
提案する 2 種類のコマンド入力には，2 つの共通の特徴がある．1 つ目は前述した予備実験結果から得られた $c_{Distance}$ が小さいという特徴である．もう 1 つは同一キーの連続打鍵が発生しないという特徴である．ストロークは指を滑らせて入力するので同一キーを打鍵するためには一度隣接キーへ移動しなければならない．スタンプは一度の打鍵動作で複数のキーを押すため，同一キーの複数打鍵は発生しない．そこで，キー入力の履歴の座標列 $S' = \{(x'_1, y'_1), (x'_2, y'_2), \ldots, (x'_n, y'_n)\}$ がある状態で，新たにキー (x'_{n+1}, y'_{n+1}) が押された時に，以下のいずれかの条件を満たす場合は，その履歴をタイピングであるとして，現在のキー入力の履歴をシステムで保持せず，すぐに画面に表示するという処理を加えた．

- $\sqrt{(x'_{n+1} - x'_n)^2 + (y'_{n+1} - y'_n)^2} > 30$
- $x'_n == x'_{n+1}$ and $y'_n == y'_{n+1}$

さらに，タイピングからコマンド入力の動作の間には若干の時間差があるため，提案システムでは，一度タイピングと判定されたらその後もタイピングが続けられるとする．タイピング早期検出機構によってタイピングが検出された後，キーイベントが一定時間（初期値は 0.25 秒に設定）発生しなくなるまでに発生したキー入力は全てタイピングであるとみなし，システムで履歴を保持せずに画面に出力する．

3.5　ストローク識別アルゴリズム

コンテキスト識別アルゴリズムでストローク入力と判断されたキー入力の履歴は，DP マッチングを用いた文字列比較 [10] を提案するストロークの識別に特化させた手法によって識別される．DP マッチングを用いた文字列比較では，文字列の不一致の原因として「挿入・削除」と「置き換え」の 2 種類を定義し，それぞれ



(a) 軌跡の文字列への置き換え　(b) 置き換えコスト

(c) コストの計算過程　(d) コストの蓄積計算結果

図 5　ストロークの距離の導出過程
Fig. 5　The calculation process of distance between strokes

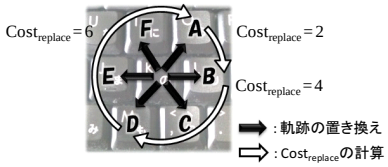


図 4　軌跡の置き換え
Fig. 4　The conversion of a trajectory

に適当なコスト $Cost_{del-ins}$，$Cost_{replace}$ を定めて文字列同士の乖離度をスコア化する．一般的なキーボードのアルファベットキーは周囲のキーと 6 方向で接しているため，文字列の軌跡を図 4 のように，右上に移動した場合は ‘A’ に，同様に右に移動した場合，右下に移動した場合，左下に移動した場合，左に移動した場合，左上に移動した場合はそれぞれ ‘B’，‘C’，‘D’，‘E’，‘F’ に置き換える．例えば，キーボード上を円形になぞって「‘B’‘G’‘Y’‘U’‘J’‘N’‘B’」の順にキーが押された場合，その軌跡は “FABCDE” という文字列に変換される．また，$Cost_{del-ins}$ を 1 に設定し，$Cost_{replace}$ を軌跡のずれの角度の大きさにより図 4 のように 3 段階に設定した．システムにはあらかじめ全てのストローク入力の軌跡を文字列に変換したものと出力するイベントのラベルの組を保持しておき，未知のストローク入力の軌跡に対して保持している全ての軌跡との距離を求め，最も近いストローク入力のラベルを認識結果とする．以下にストローク入力間の距離を求める際の詳細な計算手順を具体例を用いて説明する．「‘B’‘G’‘Y’‘U’‘J’‘N’‘B’」と「‘Z’‘A’‘W’‘3’‘E’‘D’‘X’‘Z’」の 2 種類のストロークの距離を計算する場合，まず 2 つのストロークの軌跡を文字列に変換する（図 5(a)）．そして，それらの $Cost_{replace}$ を求める（図 5(b)）．その後，表の左上から右下に向かって $Cost_{replace}$ と $Cost_{del-ins}$ の和を計算する（図 5(c,d)）．ここで，挿入・削除は表を右，あるいは下に動くことを意味し，左上からの移動の場合はそのコストがかからない．そして，最終的に表の一番右下で計算されたコストが 2 つのストロークの距離である．

3.6　スタンプ識別アルゴリズム

コンテキスト識別アルゴリズムでスタンプ入力と識別された履歴は，さらに特徴量を抽出してスタンプの際の図形である手の形状を識別する．抽出する特徴量は，t_{Press}，n_{All}，X 座標，Y 座標それぞれの分散，重心位置の 6 つである．ここで，ある系列値 $a_1, a_2, \cdots, a_n$ の重心位置 a_{pos} はその系列の最小値 a_{min} と最大値 a_{max} を用いて以下の式で計算される．

$$a_{pos} = \frac{1}{a_{max} - a_{min}}\left(\frac{1}{n}\sum_{i=1}^{n} a_i - a_{min}\right)$$

あらかじめ特徴量と正解のスタンプの組をシステムに学習させておき，未知のスタンプを認識する際は，その未知のスタンプ入力の履歴の特徴量と学習した特徴量のユークリッド距離を計算する．その後，k 近傍法 (k-nearest neighbor algorithm) を用いて推定された正解ラベルのスタンプが入力されたと判断し，そのラベルに応じた機能を実行する．k 近傍法は，入力データの近傍 k 個にある学習データ群のラベルの多数決によって入力データのラベルを決定するものである．ここで，$k = 3$ とした．

4.　評価

提案システムの評価として，コマンドの記憶調査と識別精度調査を行った．これらはいずれも 6 名の被験者（A〜F）から採取したデータを用いて行った．被験者はいずれも 20 代男性で，被験者 A，B，C，D はコンピュータの使用時間が一日平均一時間未満，被験者 E，F は平均一から二時間の者である．また，被験者 A，B，C はキー配列を把握していない者，被験者 E，F はキー配列は把握しているがタッチタイピングはできない者，被験者 D はタッチタイピングができる者である．キーボードは Panasonic Let’s note CF-S9 の純正キーボードを使用した．

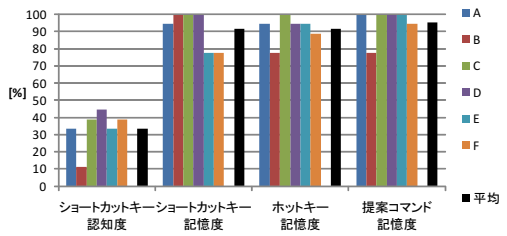


図 6　各コマンドの記憶調査
Fig. 6　The rememberable test

4.1　記憶調査

記憶調査はテキストエディタ，ブラウザ，音楽プレーヤの 3 種類のアプリケーションからそれぞれ 6 種類，計 18 種類の機能に関して行った．なお，使用した機能はテキストエディタは「保存」,「コピー」,「ペースト」,「元に戻す」,「やり直し」,「行選択」，ブラウザは「お気に入り登録」,「前に戻る」,「次に進む」,「ホームページ」,「現在のタブを閉じる」,「更新」，音楽プレーヤは「再生」,「停止」,「開く」,「次の曲」,「早送り」,「巻き戻し」の 18 種類である．

まず，被験者は，機能名が書かれた用紙にそれぞれのショートカットキーを知っている範囲で記述し，記述後に各ショートカットキーの正解とその隣に 2 つの空欄が書かれた紙が配布される．ここで，各被験者があらかじめ知っていたショートカットキーの割合を認知度とする．次に，被験者は，各機能に対して自由にショートカットキーのようなキーコマンドと提案コマンド（ストローク，スタンプの組合せは自由）を定義して記述する．以下，被験者が各機能に対して定義したキーコマンドをホットキーと呼ぶ．ホットキーは，ショートカットキーをそのまま記述してもよいこととする．提案コマンドは,「ストローク：右に伸びる直線」や「スタンプ：手を握った形」のように記述するか，記述が難しい場合は図形を描いて定義する．被験者が記憶できたと宣言した後に用紙を回収する．その後，1 時間の時間をおき，記憶の有無を調査した．1 時間の間に他の作業に集中させるため，4.2 節に述べる識別精度調査の入力データを収集した．記憶の有無は，機能名の一覧とその隣に 3 つの空欄が書かれた紙にショートカットキー，ホットキー，提案コマンドを覚えている範囲で記述させて確認した．なお，各被験者にはキーボードを見せながら実験を行った．

認知度と各コマンドの記憶度の割合の結果を図 6 に示す．図 6 から，各ショートカットキーの認知度は平均 33%と低かった．認知度が 80%を超えるショートカットキーも存在したが，半数のコマンドの認知度は 0%となった．一定時間後の記憶調査では，ショートカットキーが 91.7%，ホットキーが 91.7%，提案コマンドが 95.3%で被験者がコマンドを記憶していた．詳しくみると，ホットキーや提案コマンドの定義において被験者間で類似点がみられた．テキストエディタでは 5 名の被験者が「元に戻す」と「やり直し」に対称的なストローク（4 名が左右に伸びる直線，1 名が＜と＞）を登録し，3 名の被験者が「コピー」と「ペースト」にスタンプで握った手と開いた手を登録していた．また，1 名のショートカットキーを知っている機能にショートカットキーの頭文字（「保存」なら ‘S’）を割り当てていた．ブラウザでは,「前に戻る」,「次に進む」に全被験者が左右に伸びる直線のストロークを割り当てていた．また,「現在のタブを閉じる」には 4 名の被験者が握った手でのスタンプを登録し，3 名の被験者がお気に入り登録に星型のストロークを登録していた．音楽プレーヤでは 5 名の被験者が「再生」,「停止」にスタンプで開いた手と握った手をそれぞれ登録し，残りの 1 名は実際のプレイヤーにあるような三角と四角のストロークを登録した．また，5 名の被験者が「早送り」と「巻き戻し」に左右に伸びる直線を登録し,「開く」には 4 名の被験者が時計回りの円のストロークを登録していた．このような類似点からも図形型コマンドの直観性が期待できる．

表 2　タイピングの評価
Table 2　The evaluation of texttyping

被験者	タイピング			コンテキスト誤認識 (%)
	タイムラグあり		タイムラグなし (%)	
	コ認識 *1(%)	タ早期 *2(%)		
A	21.8	35.4	42.3	0.5
B	35.8	26.3	37.9	0.0
C	34.8	24.8	35.0	5.4
D	16.2	28.7	54.0	1.1
E	22.8	30.3	45.6	1.4
F	23.0	31.4	45.6	0.0
平均	25.7	29.5	43.4	1.4

∗1：コンテキスト認識アルゴリズム，∗2：タイピング早期検出機構

4.2　識別精度調査

4.2.1　タイピング

提案システムを用いることによるタイピングの際のコマンド誤認識回数および入出力のタイムラグを評価するために，被験者に自由に文章をタイピングさせてデータを取得した．各被験者の打鍵数 100 回あたりのタイムラグを伴ってコンテキスト識別アルゴリズムあるいはタイピング早期検出機構でタイピングと判定された打鍵数，タイムラグを伴わずにタイピングとして出力された打鍵数，コマンド入力であると誤認識された打鍵数を表 2 に示す．タイムラグが発生しない打鍵はタイピングと早期検出されてから入力が途切れるまでの打鍵である．結果から以下の知見を得た．

・コマンド入力との誤認識：平均して 100 打鍵毎に 1.4 打鍵の割合でシステムがタイピングをコマンド入力と誤認識し，その全てがスタンプ入力との誤認識であった．その際の履歴は “ry” や “om” のようなものだった．コンテキスト識別アルゴリズムの誤認識を減らす

表 3 タイピングのタイムラグ
Table 3 The delay in texttyping

被験者	タイムラグ [msec.] (%)					
	-5	6-200	201-400	401-600	601-800	801-
A	26.5	18.1	39.0	12.4	3.2	0.8
B	29.7	5.9	39.8	16.5	5.1	3.0
C	27.9	21.0	44.5	5.1	1.1	0.4
D	32.8	21.7	32.4	9.0	3.3	0.8
E	31.8	17.6	35.1	11.6	3.0	0.9
F	29.5	20.9	36.3	8.6	3.6	1.1
平均	29.7	16.9	38.3	10.9	3.1	1.1

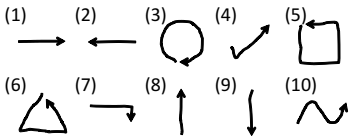


図 7 使用したストロークコマンド
Fig. 7 The stroke commnads

には，コンテキストを識別するまでの待ち時間を長くするという対策が考えられる．それによって新たに発生したキーイベントによって誤認識を防いだり，タイピングの早期検出が発生する．しかし，タイピングが早期検出されない場合のタイムラグが大きくなるというトレードオフが存在する．あるいはコンテキストの遷移を考慮して，タイピングが長時間続いている時は閾値を調整するといった処理も対策として考えられる．

・タイピング早期検出機構の有効性：表 2 から，多くの履歴がコンテキスト識別アルゴリズムを使用する前にタイピング早期検出機構によってタイピングと認識されていることが分かる．タイピング早期検出機構により，提案システムによる入出力のタイムラグを大幅に低減できた．

・タイムラグ：表 2 に示す通り，約 55%の打鍵がタイムラグを伴って出力された．タイムラグの長さの割合を調べ，結果を表 3 に示す．表 3 から，約 85%のタイムラグが 400ms 以下に抑えられていることが分かる．打鍵の開始から出力までのタイムラグによる違和感の評価を行った．被験者は 5 名で，タイムラグを 0.1 秒から 1.0 秒まで 10 段階に変化させて，違和感を感じたタイムラグを調査した．被験者はいずれもコンピュータを日常的に使用する者である．キー入力のタイムラグの影響はコンピュータの習熟度が高いほど深刻であるので，習熟度が高い被験者を採用した．実験の結果，5 名中 4 名がタイムラグが 0.4 秒以上の時に違和感を感じ，残りの 1 名は 0.5 秒になって初めて違和感を感じた．この結果から，提案システムの使用による違和感は少ないと言える．実際に被験者に対するアンケートの結果，全ての被験者が提案システムの使用によるタイムラグは気にならない範囲だと回答した．

4.2.2 ストローク

図 7 に示す 10 種類のストロークコマンドを 10 回ずつ入力した際の識別精度を調査した．なお，各ストロークの学習に 3 回分の入力を用いた．結果を表 4 に示す．

表 4 識別精度（ストローク）
Table 4 The recognition accuracy (stroke)

	(1)	(2)	(3)	(4)	(5)	(6)
(1)	98.3%	0.0%	0.0%	0.0%	0.0%	0.0%
(2)	0.0%	98.3%	0.0%	0.0%	1.7%	0.0%
(3)	0.0%	1.7%	95.0%	1.7%	0.0%	0.0%
(4)	3.3%	0.0%	0.0%	68.3%	0.0%	0.0%
(5)	6.7%	36.7%	0.0%	1.7%	31.7%	0.0%
(6)	5.0%	15.0%	0.0%	5.0%	6.7%	31.7%
(7)	16.7%	0.0%	1.7%	1.7%	5.0%	0.0%
(8)	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
(9)	0.0%	1.7%	0.0%	0.0%	0.0%	0.0%
(10)	1.7%	0.0%	1.7%	15.0%	1.7%	0.0%
	(7)	(8)	(9)	(10)	タイピング	スタンプ
(1)	0.0%	0.0%	0.0%	0.0%	0.0%	1.7%
(2)	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
(3)	0.0%	0.0%	1.7%	0.0%	0.0%	0.0%
(4)	3.3%	23.3%	1.7%	0.0%	0.0%	0.0%
(5)	5.0%	16.7%	1.7%	0.0%	0.0%	0.0%
(6)	5.0%	31.7%	0.0%	0.0%	0.0%	0.0%
(7)	53.3%	0.0%	21.7%	0.0%	0.0%	0.0%
(8)	0.0%	98.3%	0.0%	0.0%	0.0%	1.7%
(9)	1.7%	0.0%	96.7%	0.0%	0.0%	0.0%
(10)	0.0%	10.0%	0.0%	70.0%	0.0%	0.0%

表 5 識別精度の遷移（ストローク）
Table 5 The accuracy transition (stroke)

	登録コマンド数ごとの識別精度					
	1	2	4	6	8	10
(1)	98.3%	98.3%	98.3%	98.3%	98.3%	98.3%
(2)	100.0%	99.8%	99.4%	99.1%	98.7%	98.3%
(3)	100.0%	95.9%	95.0%	95.0%	95.0%	95.0%
(4)	100.0%	90.4%	78.5%	72.3%	69.4%	68.3%
(5)	100.0%	67.0%	46.6%	39.6%	35.1%	31.7%
(6)	100.0%	52.8%	38.0%	35.0%	33.2%	31.7%
(7)	100.0%	83.9%	68.1%	61.3%	57.0%	53.3%
(8)	98.3%	98.3%	98.3%	98.3%	98.3%	98.3%
(9)	100.0%	99.6%	98.9%	98.1%	97.4%	96.7%
(10)	100.0%	89.4%	80.1%	75.4%	72.4%	70.0%
平均	99.8%	87.7%	80.3%	77.4%	75.7%	74.3%

また，登録コマンド数を 1 種類から 10 種類に変化させた時の識別精度の遷移を表 5 に示す．なお，登録コマンド数が 1 種類の時の識別精度はコンテキスト識別精度に等しい．コンテキスト識別精度は 99.8%，コマンドを 10 種類登録時のストローク識別精度は 74.3%となった．また，10 種類のうちの 5 種類のストロークコマンドについては識別精度が 95%を超えた．結果から以下の知見を得た．

・コンテキスト誤認識：短いストロークコマンドを素早く入力した際にスタンプと誤認識する事例が存在した．具体的にはそれによって (1) と (8) の識別精度が低下している．

・図形の適性：表 4 から，(1)，(2)，(3)，(8)，(9) の精度が 95%以上であるのに対して，(5) と (6) の精度が 31.7%と極端に低い．被験者の入力がストロークコマンドの角で一度止まり，不完全な履歴で識別される事例が多く存在した．同様の理由で (4) と (7) の精度も低かった．入力が途切れたストロークコマンドを連結させる機構が必要だと考えられる．

4.2.3 スタンプ

ストロークの評価と同様にスタンプについても 5 種類のスタンプコマンドを 10 回ずつ入力した際の識別

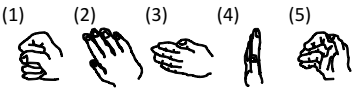


図 8　使用したスタンプコマンド
Fig. 8　The stamp commnads

表 6　識別精度（スタンプ）
Table 6　The recognition accuracy (stamp)

	(1)	(2)	(3)	(4)	(5)	タイピング	ストローク
(1)	30.0%	10.0%	8.3%	20.0%	20.0%	6.7%	5.0%
(2)	18.3%	18.3%	13.3%	25.0%	11.7%	8.3%	5.0%
(3)	3.3%	10.0%	48.3%	6.7%	13.3%	13.3%	5.0%
(4)	13.3%	0.0%	1.7%	61.7%	15.0%	6.7%	1.7%
(5)	15.0%	8.3%	10.0%	30.0%	21.7%	13.3%	1.7%

表 7　識別精度の遷移（スタンプ）
Table 7　The accuracy transition (stamp)

	党則コマンド数ごとの識別精度				
	1	2	3	4	5
(1)	88.3%	57.9%	42.2%	34.6%	30.0%
(2)	86.7%	51.3%	32.5%	22.9%	18.3%
(3)	81.7%	61.3%	53.6%	53.6%	48.3%
(4)	91.7%	80.0%	68.1%	68.8%	61.7%
(5)	85.0%	56.3%	40.8%	30.8%	21.7%
平均	86.7%	61.3%	47.4%	42.1%	36.0%

精度を調べた．各スタンプの学習は 20 回分の入力を用いた．入力したスタンプコマンドを図 8 に，結果を表 6 に示す．また，登録コマンド数を 1 種類から 5 種類まで変化させた時の識別精度の遷移を表 7 に示す．コンテキスト識別精度は 86.7%，コマンドを 5 種類登録時のスタンプ識別精度は 36.0%と，ストロークの識別精度に比べると低い値になった．結果から以下の知見を得た．

・コンテキスト誤認識：表 6 から 10%の入力がタイピングと誤認識されていることが分かる．履歴の解析から，それらは極端に大きな t_{span} を特徴としてもっていることが分かった．また，3.7%の入力がストロークと誤認識されているが，それらは極端に大きな $t_{interval}$ をもっていた．被験者の多くが慎重に入力するあまり，履歴の N_{all} が 1 の場合にタイピングと識別されていた．さらに，キーボードの内部回路の制約から，押したキーの全てが反応していなかったり，実際には押していないキーが反応したりしていた．これは，前述したキーボード毎に保証されている同時押しキー数を超えたために内部回路で電流が逆流したためだと考えられる．

・図形の適性：表 6，表 7 から，打鍵する範囲が狭いスタンプコマンドほど識別精度が高くなっていることが分かる．前述した通り，従来のキーボードは多数のキーの同時押しを想定していないため，(2) のような打鍵範囲が広いスタンプコマンドの入力時には予期していない動作をする頻度が高くなる．このことから，打鍵範囲が狭いスタンプコマンドが提案システムには適していると言える．

上記の問題を解決するためには，打鍵圧の値などキーイベントの履歴以外の要素を使うことも有効である．打鍵圧を用いることで，それをパラメータとした機能の変更も可能になる．

4.3　キーボードによる違い

3 種類のキーボードに対して 6 名の被験者のコマンド入力の識別精度を調査した．被験者は 3.3 節の予備実験の被験者と同一である．使用したキーボードは，キーが薄くキー間に溝が少ないキーボード A（Panasonic Let's note CF-W7 の純正キーボード），キーが分厚くキー間にやや溝があるキーボード B（Logitech Cordless Desktop S510），キーが薄くセパレートキータイプのキーボード C（Sony VAIO VGN-Z90NS の純正キーボード）である．結果，スタンプの識別精度に統計学的な差は見られなかったが，キーボード C のストロークの精度がキーボード A と C に比べて低くなった．これは，セパレートキーのキー間の隙間によって，ストロークの図形の情報の一部が失われたためだと考えられる．また，数名の被験者からキーボード B 上でのストロークの入力が困難であるという意見が得られた．提案システムの利用にはキーが薄くキー間に溝が少ないキーボードが適していると言える．

5.　応用例

5.1　図形コマンドの自動関連付け

提案コマンドの記憶調査から，図形と機能の関連付けにある程度の共通の嗜好があることが分かった．それを詳細に調査することで，各アプリケーションの機能名から関連付ける図形の候補を選定できる．ユーザが図形と機能の関連付けを行う際にその候補を提示することで，学習の際の手間を低減し，提案システムの利便性を向上させられる．

5.2　場所依存のスタンプ入力

本論文のスタンプの評価では，コンテキスト認識精度は 85.7%であったにも関わらず，5 種類の図形の認識の精度は 36.0%と低くなった．また，被験者に対するアンケートからストロークに比べてスタンプは形の自由度が少ないという意見が得られた．これらから，スタンプの入力を図形型コマンドとして扱うのではなく，場所依存のコマンドとして扱う，という活用方法が考えられる．例えば「左」「中央」「右」という粗い粒度で識別して，文字の左揃え，中央揃え，右揃えを行うなどの使用例が考えられる．入力されたキーは考慮せず，その座標のみを特徴量として扱うので，キー配列を把握していないユーザも直観的に使用できる．

6.　おわりに

本論文では，従来の文字入力である「タイピング」に加えて「ストローク」と「スタンプ」という 2 種類の直観的なコマンド入力を提案し，特別な装置を用いることなく，文字入力中にシームレスにコマンド

入力が行える方式を構築した．評価から，ストロークはコンテキスト識別精度が 99.8%，10 種類のコマンド登録時の図形の識別精度のうち 5 種類が 95%を超えるなど，非常に高い精度で識別できることを確認した．スタンプはコンテキスト識別精度が 86.7%，5 種類のコマンド登録時の図形の識別精度が 36.0%であった．今後は，認識精度向上のための改良や提案システムを活用したアプリケーションの実装などを行う予定である．なお，提案システムのプロトタイプは http://ubi.eedept.kobe-u.ac.jp/sas/に公開している．

参考文献

[1] 塚田有人, 星野剛史: Pointing Keyboard: キー入力／ポインティングが可能な入力デバイス, 第 10 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS 2002), pp. 61–66 (2002).

[2] F. Block, H. Gellersen, and N. Villar: Touch-Display Keyboards: Transforming Keyboards into Interactive Surfaces, Proceedings of the 28th international conference on Human factors in computing systems, pp. 1145–1154 (2010).

[3] FingerWorks, Inc: TouchStream Stealth, http://www.fingerworks.com.

[4] J. Rekimoto: ThumbSense: Automatic Mode Sensing for Touchpad-based Interactions, CHI 2003 extended abstracts on Human factors in computing systems, pp. 852–853 (2003).

[5] K. Iwasaki, T. Miyaki, and J. Rekimoto: Expressive Typing: A New Way to Sense Typing Pressure and Its Applications, Proceedings of the 27th international conference extended abstracts on Human factors in computing systems, pp. 4369–4374 (2009).

[6] R. Joyce, G. Gupta: Identity Authentication Based on Keystroke Latencies, Communications of the ACM, Vol. 33, Issue 2, pp. 168–176 (1990).

[7] D. Gunetti and C. Picardi: Keystroke Analysis of Free Text, ACM Transactions on Information and System Security, Vol. 8, Issue 3, pp. 312–347 (2005).

[8] C. Epp, M. Lippold, and R. L. Mandryk: Identifying Emotional States using Keystroke Dynamics, Proceedings of the 2011 annual conference on Human factors in computing systems, pp. 715–724 (2011).

[9] P. O. Kristensson and S. Zhai: Command Strokes with and without Preview: Using Pen Gestures on Keyboard for Command, Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 1137–1146 (2007).

[10] P. A. V. Hall and G. R. Dowling: Approximate String Matching, ANC Computing Surveys, Vol. 12, pp. 381–402 (1980).

（2011 年 8 月 29 日受付，2012 年 1 月 16 日再受付）

著者紹介

片山　拓也

2008 年大阪大学工学部電子情報エネルギー工学科卒業．2010 年同大学大学院博士前期課程修了．現在，神戸大学大学院工学研究科博士後期課程に在籍．コンテキストアウェアサービス，ユーザインタフェース，ウェアラブルコンピューティングの研究に興味をもつ．

村尾　和哉

2006 年大阪大学工学部電子情報エネルギー工学科卒業．2008 年同大学院情報科学研究科博士前期課程修了．2009 年より独立行政法人日本学術振興会特別研究員 DC2．2010 年同大学院情報科学研究科博士後期課程修了．2010 年より独立行政法人日本学術振興会特別研究員 PD．2011 年より神戸大学大学院工学研究科学術推進研究員．2011 年神戸大学大学院工学研究科助教となり，現在に至る．博士（情報科学）．ウェアラブル・ユビキタスコンピューティング，コンテキストアウェアネスの研究に従事．IEEE 等，3 学会の会員．

寺田　　努　（正会員）

1997 年大阪大学工学部情報システム工学科卒業．1999 年同大学院工学研究科博士前期課程修了．2000 年同大学院工学研究科博士後期課程退学．同年より大阪大学サイバーメディアセンター助手．2005 年より同講師．2007 年神戸大学大学院工学研究科准教授．現在に至る．2004 年より特定非営利活動法人ウェアラブルコンピュータ研究開発機構理事を兼務．博士 (工学)．アクティブデータベース，ウェアラブルコンピューティング，ユビキタスコンピューティングの研究に従事．IEEE 等，5 学会の会員．

塚本　昌彦　（正会員）

1987 年京都大学工学部数理工学科卒業．1989 年同大学院工学研究科修士課程修了．同年シャープ（株）入社．1995 年大阪大学大学院工学研究科情報システム工学専攻講師，1996 年同専攻助教授，2002 年同大学院情報科学研究科マルチメディア工学専攻助教授，2004 年神戸大学電気電子工学科教授となり，現在に至る．2004 年より特定非営利活動法人ウェアラブルコンピュータ研究開発機構理事長を兼務．工学博士．ウェアラブルコンピューティングとユビキタスコンピューティングの研究に従事．ACM，IEEE 等，8 学会の会員．