Vol.2012-HCI-146 No.17
2012/1/20

PICo: 2キーの入力タイミングに基づくコマンド入力手法

片山拓也[†1] 寺田努[†1,†2] 塚本昌彦[†1]

現在，文字入力に用いられるキーボードには，操作時間やデバイス間の移動回数を減らすためにコマンド入力機構が備わっている．しかし，コマンド入力に用いられる修飾キーやファンクションキーはキーボードの端に位置することが多く，ホームポジションを崩して入力を行うため，入力速度を落とす要因になる．そこで，本研究では，二つのアルファベットキーの入力タイミングをコマンド入力として扱う方式を提案する．本稿では，システムのプロトタイプを作成し，ユーザによる評価実験から提案手法の有効性を検証する．

PICo: a Command Input Method based on Input Timing of Two Different Keys

Takuya Katayama,[†1] Tsutomu Terada[†1,†2] and Masahiko Tsukamoto[†1]

The keyboard, which is mainly used for text input, has command input function to reduce the operation time and the movement beteween input devices. However, the modifier keys and function keys, which are used on command input, are placed on the edge of keyboard. It might cause the input speed degradation because it is difficult to keep fingers on the home position in inputting a command. We propose a command input method based on the combination of input timing between two alphabet keys. We developed a prototype system, then verified the effectivity of the proposed method.

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

1. はじめに

現在，文字入力デバイスとして広く用いられているキーボードには，文字入力機構のほか，操作時間やデバイス間の手の移動回数を減らすためにコマンド入力機構が備わっている．例えば，修飾キーと特定のキーの組合せ入力で実現されるショートカットキーや，1キーに機能が関連付けられているファンクションキーがある．しかし，それらの入力に用いられる修飾キーやファンクションキーは，通常はキーボードの端に位置することが多く，ホームポジションを崩したり，視線をキーボードに移して入力を行うため，入力速度を低下させる要因となる．また，従来のキーボードの使用方法では，修飾キーとの組合せ入力以外で複数キーの組合せ入力は行われておらず，複数キーの入力タイミングが考慮されていなかった．そこで，本研究では，ユーザが正確にキー位置を把握しており，ホームポジションを維持したまま入力できるアルファベットキーに注目し，二つのアルファベットキーの入力タイミングをコマンドとして扱う方式（以下，PICo: Parallel Input Command）を提案する．PICoでは，二文字のアルファベットに対して機能を関連付けているため，一文字のアルファベットに対して機能を関連付けるショートカットキーと比べて，コマンドが記憶しやすいという効果が見込まれる．提案システムは，ユーザの入力が従来の文字入力かPICoかを自動で識別するので，通常の文字入力を支障をきたすことはない．

本研究では，システムのプロトタイプとして従来の文字入力とPICoを識別する機構を実装し，識別精度を調査した．そして，ユーザ評価から提案手法の有効性を検証した．以下，2章で関連研究を紹介し，3章でPICoの識別機構と識別精度の評価について説明し，4章で構築したプロトタイプを紹介し，5章で提案手法の有効性の評価について述べる．そして，6章で考察を述べ，最後に7章で本研究のまとめを行う．

2. 関連研究

ThumbSence[1)]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を操作する際に，キーボードのホームポジションから手が離れて入力速度が低下するという問題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を割り当てる．この技術によって，タッチパッドに親指で触れながら他の指でキーボードに割り当てられたマウスボタンを押す操作ができる．ThumbSenceではタッチパッドに指が触れている間はマウスボタンの機能以外にも各キーにコマンドを割り当てているので，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，コマンドに対するキー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が，この煩雑さの原因の一つに一文字のアルファベットに対して一つの機能を割り当てていることがある．ショートカットキーは “開く” に “Ctrl+O(Open)”，“保存” に “Ctrl+S(Save)” のように機能の頭文字を割り当てていることが多いので，もしも頭文字が同じ機能が存在する時には，修飾キーの組合せを変えるか，別のアルファベットに機能を割り当てなければならない．

速記用のキーボードとして開発されたステノワード[7] は，文字入力キーが 10 個しかなく，右側 5 個のキーに母音，左側 5 個のキーに子音が割り当てられたキーボードである．このキーボードは文字キーが 10 個しかないので，通常のキーボードのように指を上下左右に動かす必要がなく，指は常にホームポジションに置いたままで入力でき，同時押しを取り入れることで 10 個のキーだけで日本語を高速に入力できる．これから，ホームポジションから指を出来るだけ離さないことが高速入力にとって重要な要因であることが分かる．

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[7] と呼ばれる入力を用いたコマンド入力[8] 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として扱う．このシステムを用いることで，ソフトウェアキーボード上でのコマンドの入力速度が向上した．速度が向上した理由の一つに，機能名をそのままコマンド入力として扱うことができ，記憶しやすくなったことが挙げられる．同様に，提案システムでは機能名から連想される 2 文字を割り当てられるので，記憶のしやすさだけでなく，入力動作の開始を早める効果も期待できる．

Touch-Display Keyboard[3] 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，各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，キーボード上でマウスとタッチの組合せ入力を実現している．市販されているシステムには FingerWorks 社の TouchStream[4] 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，既存のキーボードに適用できない．また，各キーにディスプレイとタッチセンサを取り付けるには大きなコストがかかる．そのため，一般的な家庭や職場で取り入れるのは現段階では難しい．一方，提案システムは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せず，既存のキーボードに適用できる．

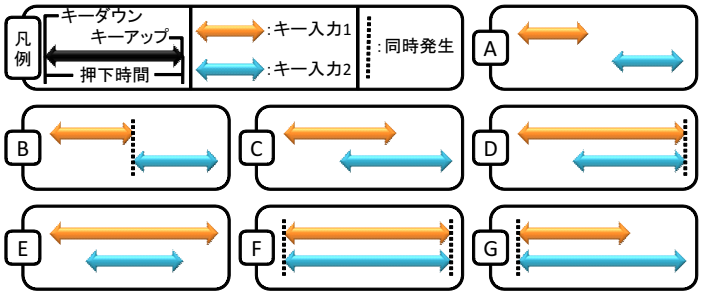

図 1 2 キーの入力タイミング
Fig. 1 The input timing of two keys

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う打鍵情報はキーストロークダイナミクスと呼ばれ，様々な研究に用いられている．キーストロークダイナミクスの主な適用例として個人認証がある．キーストロークダイナミクスは個人ごとに癖が存在し，異なることが分かっており，固定テキストのキーストロークダイナミクスを学習することで，手書きのサインと同レベルの識別精度が実現できることが確認されている[5]．キーストロークダイナミクスを用いた認証方法は，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り，注目されている．別の適用例としてユーザの感情を認識する試みも行われている[6]．この手法では，キーストロークダイナミクスを用いて七種類の感情状態を二レベルで認識できることが確認された．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文字入力と PICo を識別する．

3. 提案コマンド入力手法

3.1 システム概要

システムはあるキー入力があった際に，文字入力か PICo かを自動で識別する．文字入力と識別されたキー入力は反映し，PICo と識別されたキー入力は反映せずに関連付けられた機能を実行する．2 つのキーを 1 度ずつ押す際のタイミングのバリエーションは**図 1** に示す

Vol.2012-HCI-146 No.17
2012/1/20

7 種類であるが，タイミング A，B は通常の文字入力中に多く発生し，それ以外のタイミングは通常の文字入力においてほとんど発生しないと考えられる．そこで，タイミング C〜G を PICo におけるコマンドとして扱う．PICo にアルファベットキーのみを用いた場合，タイミング C，D，E，G は $26 \times 25 = 650$ 通り，タイミング F は $26 \times 25 \div 2 = 325$ 通り，合計で 2925 通りのコマンドが実現できる．さらに，0 〜 9 の数字キーを加えた場合は合計で 5670 通りのコマンドが実現できる．PICo はタッチタイピングができるアルファベットキーを用いているので高い入力速度が期待でき，二文字のアルファベットに対して機能を割り当てることで，コマンドが覚えやすいという効果も見込まれる．ショートカットキーの多くは，コマンドを連想しやすいように機能の頭文字と修飾キーの組合せが用いられている．しかし，頭文字が等しい二つの機能にショートカットキーを割り当てる際は，修飾キーの組合せを変えるか，別のアルファベットを割り当てなければならない．それに対して，PICo では二文字のアルファベットに対して機能を割り当てるため，キーの競合が起こりにくい．

3.2 コマンド識別機構

システムはキーボードの入力情報（キーダウン・キーアップ）を監視し，あるキー（$K1$）のキーダウンイベントが発生した際に，すぐにはキー入力を反映せずに保持する．ここで，$K1$ の次に入力されるキーを $K2$ とし，以下，Ki のキーダウン，キーアップイベントをそれぞれ Ki_D，Ki_U と記述する．その後，$K2_D$ が発生する前に $K1_U$ が発生した際は，$K1$ の入力を反映する．一方，$K2_D$ が $K1_U$ よりも先に発生した際は，$K1_U$，$K2_U$ の発生後に入力タイミングの検証を行い，タイミング B であれば $K1$ の入力を反映し，$K2$ を保持する．タイミング C〜G であれば $K1$，$K2$ の入力を反映せずに PICo として扱う．また，$K1_U$，$K2_U$ が発生する前に $K3_D$ が発生した場合は $K1$ の入力を反映する．なお，PICo に用いるアルファベットキーと数字キー以外の入力は直ちに反映させて，タイムラグを軽減する．

3.3 識別精度の評価

20 代の被験者 5 名から得られた入力履歴を用いて PICo の識別精度を評価した．これらの被験者はいずれもキーボードのタッチタイピングができる者である．タッチタイピングができない者の場合は文字入力時にタイミング A のみが発生し，識別精度が 100%になると考えられるためである．各被験者からは，PICo の入力としてタイミング C〜G を 10 回ずつ計 50 回と，自由文章入力（日本語，英語）時の平均 1200 キーの入力履歴を収集した．ここで PICo の入力時は 2 つのキーと 5 種類のタイミングをランダムに画面に提示して入力させた．被験者から得られた PICo 入力時の $K1$，$K2$ のキーイベントの時間関係を**表 1** に示す．得られた結果から，コマンド識別機構に用いる各閾値を以下のように決定する．

表 1 PICo の入力タイミング
Table 1 The input timing of PICo

タイミング		発生間隔 (ms)			
		平均	最小	最大	標準偏差
C	$K1_D - K1_U$	541.8	187	1300	218.8
	$K2_D - K2_U$	529.3	156	1290	189.7
	$K1_D - K2_D$	245.6	94	490	111.4
	$K2_D - K1_U$	296.3	63	1130	157.0
	$K1_U - K2_U$	233.0	94	410	75.0
D	$K1_D - K1_U$	583.9	344	920	178.8
	$K2_D - K2_U$	352.1	188	830	108.7
	$K1_D - K2_D$	250.9	47	490	124.0
	$K2_D - K1_U$	350.6	184	830	108.8
	$K1_U - K2_U$	14.1	0	109	19.9
E	$K1_D - K1_U$	772.4	406	1320	245.2
	$K2_D - K2_U$	274.5	78	560	98.3
	$K1_D - K2_D$	262.0	109	480	97.8
	$K2_D - K1_U$	510.4	266	880	163.9
	$K1_U - K2_U$	235.9	109	440	85.4
F	$K1_D - K1_U$	383.5	142	840	196.2
	$K2_D - K2_U$	379.7	137	740	188.2
	$K1_D - K2_D$	19.5	0	200	28.8
	$K2_D - K1_U$	378.7	153	750	184.3
	$K1_U - K2_U$	8.7	0	50	10.0
G	$K1_D - K1_U$	432.6	156	2990	430.0
	$K2_D - K2_U$	642.6	188	3110	448.2
	$K1_D - K2_D$	23.2	0	130	27.8
	$K2_D - K1_U$	428.4	94	2970	435.6
	$K1_U - K2_U$	214.2	63	400	88.5

表 2 文字入力時の押下時間
Table 2 The pressed span in text-typing

押下時間 (ms)			
平均	最小	最大	標準偏差
110.4	0	750	34.3

同時発生の判定

PICo のタイミング D，F，G の入力履歴から，ユーザが同時発生を意識したキーイベントが実際にはどの程度の時間差で発生しているかを調査した．表 1 から，キーダウンキーイベントは平均 21.3ms，標準偏差 28.2ms の時間差で，キーアップイベントは平均 11.4ms，標

準偏差 15.9ms の時間差で発生することが分かった．キーダウンの方が平均値が大きくなったが，これは外れ値によるものであると考えられる．実際のデータには 100ms 以上の外れ値が全体の 2.0%存在し，これらは入力ミスによるものと考えられる．それらの外れ値を除外すると，平均 13.8ms，標準偏差 14.2ms となった．外れ値が発生した理由として，被験者に PICo を 50 回連続で入力させたことによる疲労が考えられる．以上の結果から，同時発生として扱う閾値を 50ms に設定した．

押下時間制約

文章を高速で入力するとタイミング C~G のキー入力が発生する可能性があるので，各キーの押下時間から入力を識別する機構を導入する．文章入力時の各キーの押下時間を調べたところ**表 2** のようになった．PICo 入力時の各キーの押下時間は表 1 の中の $K1_D - K1_U$，$K2_D - K2_U$ である．それぞれを比較すると，PICo 入力時の方が文章入力時よりも長い押下時間になっていることが分かる．これは，PICo を入力する際に被験者がタイミングを強く意識しているためだと考えられる．そして，実際に得られた値を元に押下時間制約の閾値を設定した．システムはタイミング C~G の入力があった時にも，2 キーの押下時間がどちらも閾値に満たない場合はその入力を文章入力として扱う．設定した閾値はタイミング C と G が 200ms，D と E が 300ms，F が 150ms である．

PICo 認識

上記の閾値に適合する PICo の入力は 95.6%となり，同時発生と押下時間制約の閾値に適合しなかった入力はそれぞれ 3.6%と 0.8%となった．押下時間制約による誤認識よりも同時発生の閾値による誤認識が多く発生したが，これは前述した通り被験者に PICo を 50 回連続で入力させたことによる疲労が原因に挙げられる．同時発生の閾値による誤認識は 2 名の被験者に集中して見られ，この 2 名の被験者が集中を欠いた時に誤認識が発生していたと考えられる．その 2 名を除いた 3 名の被験者では同時発生の閾値による誤認識は見られず，適合する入力は 99.3%となった．

文章入力認識

文章入力時における連続する 2 キーの入力タイミングの発生割合と，それぞれの文章入力の際の押下時間制約に適合した入力の割合を**表 3** に示す．表 3 から文章入力時にもタイミング C~G の打鍵は発生していることが分かる．押下時間制約を考慮しない場合は，タイミング C~G の入力は全て PICo として扱われてしまうので，全体の適合率は 87.2%となる．一方，押下時間制約を考慮した場合は，適合率は 99.4%に向上しており，押下時間制約の有効性が確認された．また，この 0.6%の誤認識のうちの 94.9%は 2 名の被験者に集中しており，個人のキーボード入力の癖によるものであると考えられる．この 2 名を除いた 3 名の被験者の文章入力の適合率は 99.95%となった．個人のキーボード入力の癖を学習して，それに応じて押下時間制約の閾値を調整する必要があると言える．

表 3 文章入力時の発生割合と適合率
Table 3 The appearance rate and accuracy in text-typing

タイミング	出現割合（%）	適合率（%）
A	52.4	100.0
B	34.8	100.0
C	6.3	92.9
D	3.6	100.0
E	0.0	100.0
F	1.0	92.9
G	2.0	97.0

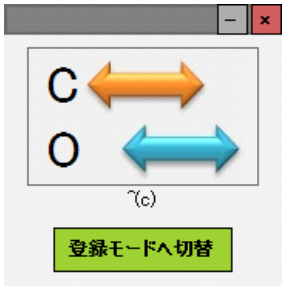

図 2 プロトタイプシステムの画面例
Fig. 2 The screenshot of prototype system

4. 実　　装

これまでに述べた入力識別機構を組み込んだプロトタイプシステムを構築した．システムの動作画面の例を**図 2** に示す．この動作画面は C キーと O キーをタイミング C で入力して「Ctrl+C」を実行している様子である．動作モードは登録モードと実行モードがあり，画面下にあるボタンを押すと動作モードが切り替わる．

登録モードでは，PICo を入力するとその PICo に関連付ける機能を入力する画面に切り替わる．プロトタイプでは関連付ける機能として，SendKeys メソッドの引数を入力する．SendKeys メソッドは，引数のキー入力をエミュレートするメソッドで，修飾キーの組合せ

入力なども実現できるのでショートカットキーの代替として利用できる．

動作モードでは，PICo の入力があった際に入力されたキーの組合せとタイミングを図で描画し，確認のためにエミュレートされるキー入力をその下に示す．入力された PICo に機能が割り当てられていない場合は，その入力を文章入力として扱う．そのため，実際に使用する際の認識精度は前章で述べた 99.4%よりも更に高くなる．

5. 提案コマンドの有効性

5.1 実験概要

PICo の有効性を評価するために実験を行った．被験者は 4 名で，いずれも 20 代，キーボードのタッチタイピングができる者である．被験者はメモ帳（Microsoft Windows に付属するテキストエディタ）起動中に，“文章を入力→機能を実行→文章を入力” のタスクを行う．タスク 10 回を 1 セットし，各タスクにおける機能の実行を “マウス・タッチパッド”，“ショートカットキー”，“PICo” と変えて 3 セット，計 30 回のタスクを行う．タスクは**図 3** に示すように，メモ帳の右下に表示して被験者に指示する．各試行で入力させる文章は，アルファベット 15 文字をランダムに生成したものとする．実行する機能は，“新規”，“開く”，“上書き保存”，“名前を付けて保存”，“印刷”，“メモ帳の終了”，“元に戻す”，“切り取り”，“コピー”，“貼り付け”，“検索”，“置換”，“行へ移動”，“すべて選択”，“日付と時刻”，“右端で折り返す”，“フォント”，“ステータスバー”，“バージョン情報” の 19 個の中からランダムに選ばれる．ショートカットキー，PICo のセットでは，タスクを行う前に上記の 19 種類の機能に対してコマンドをユーザが自由に割り当てる．入力履歴から，文章入力後から機能実行までに要した時間，機能実行後から文章入力に戻るまでに要した時間を評価する．また，評価実験後に被験者にアンケートを実施し，ショートカットキーと PICo のコマンドの覚えやすさ，被験者の体感入力速度を評価する．

5.2 入力速度

各入力手法を用いた際の，文章入力後から機能実行までに要した時間 t_1，機能実行後から文章入力に戻るまでに要した時間 t_2 を**表 4** に示す．全ての被験者と入力手法において，t_2 よりも t_1 の方が大きくなった．全ての被験者で t_1 はショートカットキー，PICo，マウス・タッチパッドの順で短くなり，ショートカットキーと PICo の間には大きな差はみられなかった．t_2 の全体の平均はマウス・タッチパッド，PICo，ショートカットキーの順で短くなったが，3 つの入力手法において大きな差はみられなかった．PICo は馴染みのない入力にも関わらず，被験者が普段から使用しているショートカットキーと同等の入力速度を実現した．以上から，PICo を習熟することでショートカットキーよりも早い入力速度を実現できる可能性があると言える．

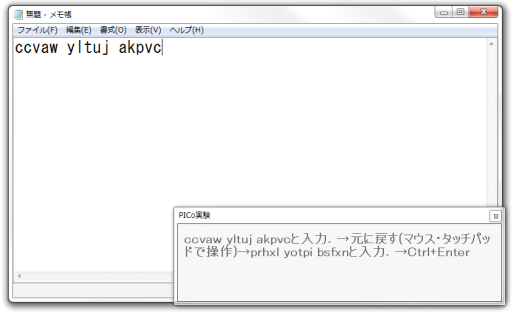

図 3 タスク指示の画面例
Fig. 3 The screenshot of task directions

表 4 各入力手法における所要時間
Table 4 The elapsed time of each input method

入力手法	被験者	文章入力後から機能実行までに要した時間 t_1(ms)		機能実行後から文書入力に戻るまでに要した時間 t_2(ms)	
		平均	標準偏差	平均	標準偏差
マウス・タッチパッド	A	5775.9	2055.5	3260.6	1485.5
	B	8332.1	4342.5	2868.2	1475.8
	C	3103.3	534.6	1117.0	400.1
	D	6543.0	2969.1	2889.8	665.1
	全体	5938.6	3171.0	2533.9	1364.6
ショートカットキー	A	3737.4	2426.1	2555.6	1151.6
	B	5004.0	2963.3	2952.7	1455.4
	C	801.2	304.1	1489.7	1025.7
	D	3918.1	1651.8	3962.8	1594.4
	全体	3365.2	2537.5	2740.2	1558.5
PICo	A	4181.3	881.2	2820.2	2297.5
	B	5207.7	1830.6	3063.2	1168.6
	C	843.9	208.3	962.8	264.3
	D	4299.2	1463.8	3816.6	972.6
	全体	3633.0	2034.2	2665.7	1706.8

5.3 アンケート

実験終了後に行った各入力手法のコマンドの覚えやすさ，被験者の体感入力速度のアン

表 5 アンケート結果
Table 5 The questionnaire result

被験者	覚えやすさ		体感入力速度		
	ショートカットキー	PICo	マウス・タッチパッド	ショートカットキー	PICo
A	4	4	3	4	4
B	3	4	4	2	4
C	3	4	2	4	4
D	4	4	2	5	4

ケートの結果を**表 5** に示す．

コマンドの覚えやすさは全ての被験者において，PICo がショートカットキーと同等かそれ以上になった．その理由として，割り当てキーの競合が考えられる．被験者はショートカットキーの割り当ての際に，既知のショートカットキー（“Ctrl+C”，“Ctrl+V” など）はそのまま用いて，ショートカットキーを知らない機能には Ctrl キーと各機能の頭文字のアルファベットキーの組合せを割り当てていた．そのため，“保存” と “ステータスバー” や，“貼り付け” と “バージョン確認”，“検索” と “フォント” でアルファベットキーの競合が起こり，修飾キーの組合せの変更や他のキーを割り当てることで競合を回避していた．一方，PICo の割り当ては各機能の初めの二文字を割り当てられるため，“上書き保存” と “名前を付けて保存” 以外ではキーの競合は発生しない．

次に，体感入力速度を見ると被験者 A，C，D はマウス・タッチパッドが最も遅いと答え，実際の入力速度とも一致している．被験者 A，C はショートカットキーと PICo の入力速度を同等と判断したが，被験者 B は実際の入力速度とは異なりショートカットキーよりもマウス・タッチパッドや PICo の方が早く入力できると感じた．体感速度が実際の入力速度と逆転した理由として，マウス・タッチパッドの入力動作開始が他よりも早いことや，PICo はホームポジションを崩さずに入力動作を開始できることなどが考えられる．

5.4 PICo の割り当て

各被験者の PICo の割り当てを**表 6**，**表 7**，**表 8**，**表 9** に示す．被験者 A と C は 18 種類，被験者 D は 13 種類のコマンドをタイミング C に割り当て，被験者 B は 17 種類のコマンドをタイミング F に割り当てている．割り当てるタイミングは個人ごとに嗜好があるが，割り当てられるキーは “新規” の ‘N’ と ‘E’ や，“開く” の ‘O’ と ‘P’ など，全ての被験者に対してある程度の共通点がみられた．PICo の覚えやすさは 2 つのアルファベットキーにあり，タイミングに直観性はないため，タイミングを統一してコマンドを記憶しやすくしたためだと考えられる．被験者 B，C は全ての PICo を機能から連想される単語中から 2 文字を割り当てている一方で，被験者 A は F キーと数字キーの組み合わせでファンクションキーのように用いたり，被験者 D はショートカットキーのコマンドを知っている機能についてはショートカットキーに使われているキーと別のキーの組合わせを用いていた．

表 6 PICo の割り当て（被験者 A）
Table 6 The allocation of PICo (Participant A)

機能	PICo	機能	PICo	機能	PICo
新規	N E	開く	O P	上書き保存	S V
名前を付けて保存	S V	印刷	P R	メモ帳の終了	Q W
元に戻す	U D	切り取り	C U	コピー	C P
貼り付け	P A	検索	F I	置換	R P
行へ移動	G I	すべて選択	S A	日付と時刻	D T
右端で折り返す	F 8	フォント	F O	ステータスバー	F 9
バージョン情報	F 0	———			

表 7 PICo の割り当て（被験者 B）
Table 7 The allocation of PICo (Participant B)

機能	PICo	機能	PICo	機能	PICo
新規	N E	開く	O P	上書き保存	S A
名前を付けて保存	S A	印刷	P T	メモ帳の終了	E D
元に戻す	R E	切り取り	C U	コピー	C O
貼り付け	P A	検索	F I	置換	P R
行へ移動	M O	すべて選択	A E	日付と時刻	T I
右端で折り返す	R E	フォント	F S	ステータスバー	S B
バージョン情報	V K	———			

表 8 PICo の割り当て（被験者 C）
Table 8 The allocation of PICo (Participant C)

機能	PICo	機能	PICo	機能	PICo
新規	N E	開く	O P	上書き保存	S A
名前を付けて保存	S A	印刷	P R	メモ帳の終了	Q U
元に戻す	U N	切り取り	C U	コピー	C O
貼り付け	P A	検索	F I	置換	R E
行へ移動	M O	すべて選択	A L	日付と時刻	D A
右端で折り返す	R I	フォント	F O	ステータスバー	S T
バージョン情報	V E	———			

表 9 PICo の割り当て（被験者 D）
Table 9 The allocation of PICo (Participant D)

機能	PICo	機能	PICo	機能	PICo
新規	N E	開く	O P	上書き保存	S A
名前を付けて保存	S A	印刷	P R	メモ帳の終了	E N
元に戻す	0 Z	切り取り	0 X	コピー	0 C
貼り付け	0 V	検索	0 F	置換	C H
行へ移動	M O	すべて選択	A L	日付と時刻	T I
右端で折り返す	T U	フォント	F O	ステータスバー	S T
バージョン情報	V A	———			

6. 考　　察

PICo で実現されるコマンド数はアルファベットキーのみを用いた場合で 2925 通り，アルファベットキーと数字キーを用いた場合で 5670 通りと膨大である．そこで，その膨大なコマンドの扱い方を以下で考察する．以下を組み合わせて使用することで，初心者から熟練者まで様々なユーザに対応したコマンド入力機構を実現できる．

一つ目は，高機能エディタに組み込む使い方である．PICo はキーボードのホームポジションから手を移動させずに入力できるので，文字入力が主な作業となるアプリケーションに適用すると利点を最大限に活かせる．エディタを使用中にユーザは膨大な数のコマンド中から自分が覚えられる範囲で機能を関連付けて扱う．カーソル移動などや，複数の操作を一つにまとめたマクロを PICo に関連付けることで，操作量を大幅に減らせる．これは，PICo のコマンド数によって実現される利点である．

次に，記憶しやすさを向上させるためにタイミングの種類を減らす使い方である．前述した通り，PICo の記憶しやすさは割り当てる 2 つのキーにあり，タイミングは個人ごとに偏る傾向がある．PICo として扱うタイミングを 1 種類に限定しても，アルファベットキーだけを用いる場合はタイミング C，D，E，G が 650 種類，タイミング F が 325 種類のコマンドを扱える．タイミングの数を限定することで記憶しやすくなるだけでなく，誤認識数を減らす効果も期待できる．

最後に，タイミングの種類をコマンドとして扱う使い方である．この使い方は，入力に用いたキーに関わらず，2 つのキーが特定のタイミングで入力された際には PICo として扱うというものである．入力キーの位置を把握する必要がないので，キー配置を正確に把握していない初心者に向いている使い方だと言える．また，この使い方では，同じアルファベットキーで複数の機能を実行できるので，アルファベットの入力を反映しながら機能を実行するという使い方（例えば，半角全角の決定を入力単語の最後の 2 キーの入力タイミングで決定）もできる．

7. おわりに

本研究では，キーボードのショートカットキーやファンクションキーの入力時にホームポジションから手が離れてしまうという問題点を解決し，コマンドを記憶しやすくするために，2 つキーの入力タイミングをコマンドとして扱う入力方式 PICo を提案した．提案システムは PICo と文章入力を自動で識別する機構をもつので，提案システムを用いることで従来の文章入力に支障をきたすことはない．本稿では，PICo の識別機構を構築し，システムのプロトタイプを作成し，ユーザ評価から PICo の認識精度と入力速度，覚えやすさを調査した．結果，提案システムを用いた時の PICo と文章入力の認識精度はそれぞれ 95.6%，99.4%となった．これらの認識精度は PICo の入力動作に慣れることでさらに向上すると思

われる．また，PICo は，ユーザが使い慣れたショートカットキーと同等の入力速度を実現でき，ショートカットキーと同等以上にコマンドを記憶しやすいということが分かった．今後は PICo を継続的に使用した際に入力速度がどの程度向上するか，PICo の活用方法を調査していく予定である．そして，提案システムのプロトタイプを公開して幅広いユーザからフィードバックを収集する予定である．

参 考 文 献

1) J. Rekimoto: ThumbSense: Automatic Mode Sensing for Touchpad-based Interactions, CHI 2003 extended abstracts on Human factors in computing systems, pp. 852–853 (2003).
2) スピードワープロ研究所, ステノワード, http://www.speed-wp.co.jp/laboratory/index.html/.
3) 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).
4) FingerWorks, Inc: TouchStream Stealth, http://www.fingerworks.com/.
5) R. Joyce, G. Gupta: Identity Authentication Based on Keystroke Latencies, Communications of the ACM, Vol. 33, Issue 2, pp. 168–176 (1990).
6) 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).
7) P. O. Kristensson and S. Zhai: SHARK^2: A Large Vocabulary Shorthand Writing System for Pen-Based Computers, Proceedings of the 17th annual ACM symposium on User interface software and technology, pp. 43–52 (2004).
8) 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).