空中手書き文字入力システムの実用化に向けた検討*

西田 好宏[*1]，　野口 祥子[*2]

A Study for Practical Use of Aerial Handwritten Character Recognition

Yoshihiro NISHIDA[*1]　and　Shoko NOGUCHI[*2]

[*1] Department of Electrical and Electronic Engineering

This paper describes a method to recognize a character handwritten in the air. This recognition method uses the motion direction instead of positions of the device mounted on the hand. It also doesn't use the information of pen-up and pen-down. It selects and orders character candidates with DP (dynamic programming) matching. We prototyped a handwritten character recognition system that detect single character period by hovering in the air instead of adding some behavior for example, hand-open and hand-close. But this system is not enough to put it to practical use because it does not have easy functions to use. We intend to improve usability and work toward practical use of aerial handwritten character recognition by additional input method functions.

Key Words : Character Recognition, Handwriting in the Air, DP Matching and Input Method Editor

1. 緒　　言

インターネットの利用目的の中で常に上位を占めているのは，友人とのコミュニケーションや色々な情報を得るための検索である．そのためにはテキストデータの入力が必要不可欠で，片手しか使えない状況でも簡単に文字を入力できれば，より便利になると考えられる．

同様のモバイルやウェアラブルコンピュータに適したオンライン手書き文字入力の研究として，嵯峨山らによるストローク HMM を用いた手法[(1)]や園田らによる装着型ビデオカメラで操作者の手を撮影して空中に書いた文字を認識するシステム[(2)]などが提案されている．

空中手書き文字入力の実現には，空中での手指の移動方向を検出するセンサ処理，移動方向から文字を認識する文字認識エンジン処理，認識した文字を基に変換や確定等で文章にする文章編集処理の３つの課題が存在するため，我々は今までこれらの課題の要素検討を中心に行ってきた[(3)(4)]．その結果，昨年には端点フリーDP マッチングにより，一文字単位の文字区切りの操作を無くして文字を書き終えて止めるだけでその文字を認識できるようにすることができた[(5)]．

しかし，従来の空中手書き文字入力システムでは，実際に修正や推敲を含めて色々な文章を入力出来るようにして実用化するには，まだ十分とは言えない．そのため今回は，実用化に向けた検討として認識アプリケーションにいくつかの機能を追加して使い勝手の向上を図った．

アプリケーションに機能追加を行うにあたって，昨年検討した端点フリーDP マッチングを実装して機能追加前の基準となるアプリケーションを作成した．これによって，文字区切りの際に動きを停止するだけで文字の認識が可能になった．これを基準として機能の追加を行った．

まず，句読点を１回の操作で入力できなかったため，句読点に特殊キーを割り当てた．次に，文字を訂正する際に，従来は訂正したい箇所まで文章を消去することでしか戻れなかったため，矢印キーを導入することで文章を消さなくても訂正箇所まで移動できるようにした．さらに，濁点・半濁点キーを導入し，スマートフォンでの

[*] 原稿受付　2016年2月29日
[*1] 電気電子工学科
[*2] 電気電子情報工学科学生
E-mail: nishida@fukui-ut.ac.jp

濁音，半濁音の入力と近いものにした．このことにより，辞書データの削減や，個人の書き癖の問題を解消することができた．また，従来のアプリケーションは日本語入力のみであったが，今回はアルファベット入力についても検討した．そのため，アルファベットの辞書データの作成を行った．これには一筆書き入力に特化した Palm 社の Graffiti を採用し，既存の文字と似た形の文字は変更したものを辞書データとして作成した．さらに，半角入力と全角入力の切り替え操作を判別し，辞書データの切り替えを行う．半角入力の場合はアルファベット用の辞書データ，全角入力の場合は日本語入力用の辞書データを使用する．

機能追加の効果を確認するにあたって，アプリケーション化を行った状態のアプリケーション，機能追加後のアプリケーション，そして iPhone の標準キーボードの 3 つのアプリケーションで例文を入力し，その入力時間と入力回数とを比較した．その結果，今回の機能追加により，入力時間，入力回数共に大幅に短縮することが出来た．また，入力回数に関しては iPhone の標準フリック入力と同等にすることが出来た．

2.　基準アプリケーションの作成

昨年検討した端点フリーDP マッチングにより，Fig.1 の文字区切りのイメージに示すように文字区切りの際に何か動作を加えなくても，書き終わりの位置で一瞬停止するだけで文字を認識することが可能になることが分かった．この一文字単位の文字区切り操作を不要にしたことで，実用化に向かって大きく進歩したため，端点フリーDP マッチングをアプリケーションに実装し，改善のための基準アプリケーションとした．

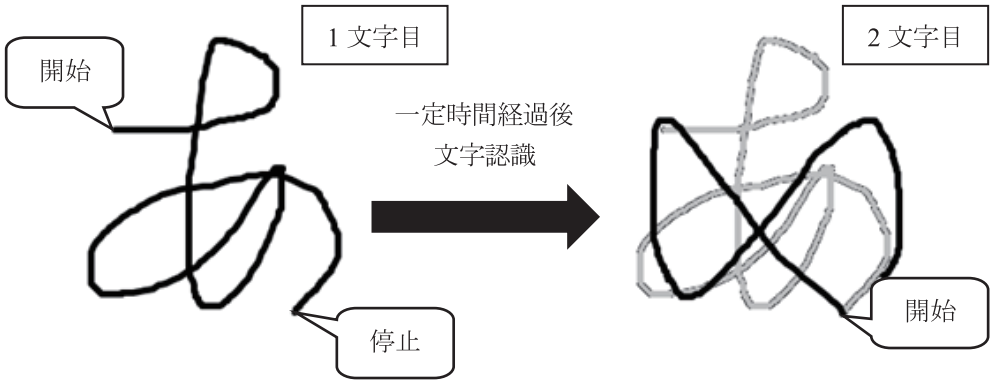

Fig. 1 Image of detecting handwriting period

2.1　端点フリーDP マッチングの実装

まずマッチングの最後にノイズが含まれる部分が来るように，辞書データと筆跡データの文字列を反転する．

従来の一般的な DP マッチングの場合，各要素での累積ペナルティ値がすべて算出された後，Fig. 2 の左にあるように一番右下の累積ペナルティ値のみを参照していたが，端点フリーDP マッチングの場合は，Fig. 2 の右にあるように，右下のものだけでなく，同じ列のすべての要素に対して，累積ペナルティ値をさらに比較し，最も小さい累積ペナルティ値を返すようにする．

この端点フリーDP マッチングを実装するためには，筆跡データ数を a_num，辞書データ数を b_num，マッチングのエラー値を累積加算する配列を g[i][j], 筆跡データの 60%の位置以降において累積ペナルティの最小 min を求めるため，従来の DP マッチングに Fig. 3 のコードを追加した．

この基準となる端点フリーDP マッチングを実装した段階のアプリケーションで使用できる空中操作（動作）と特殊キーを Table 1 に示す．

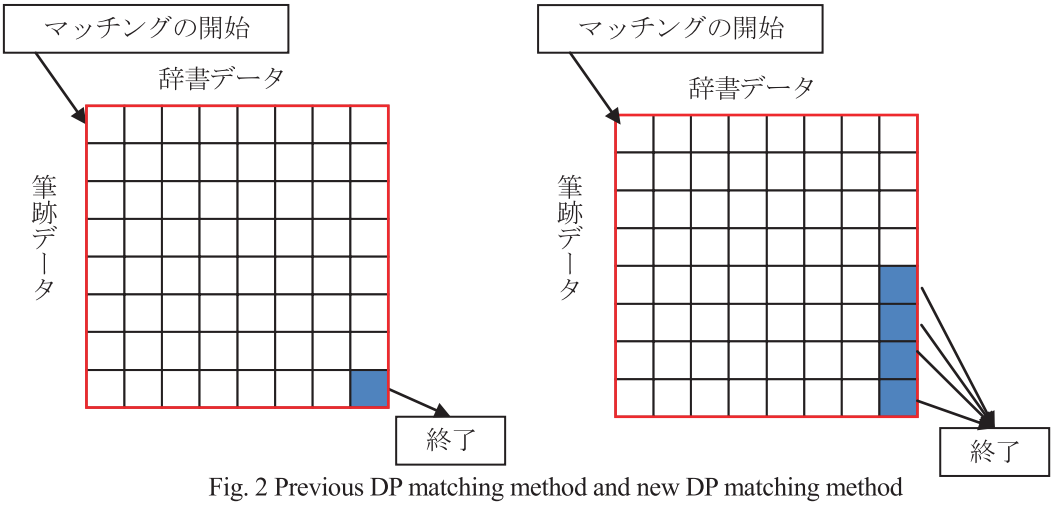

Fig. 2 Previous DP matching method and new DP matching method

```
int min = g[a_num-1][b_num-1];
int tmpnum_a = a_num * 0.6;
for (i = a_num-1; i >= tmpnum_a; i--) {
        if (g[i][b_num-1] < min) min = g[i][b_num-1];
}
return min;
```

Fig.3　Additional code of new DP matching

Table 1　Special function keys and Operations

キー	動作	プログラム内での記号	キー	動作	プログラム内での記号
消去(Back Space)		BS	半角/全角		HZ
変換(Space)		Sp	小文字		Sm
予測変換(Tab)		Tb	第二候補		P2
改行(Enter)		En	第三候補		P3

3.　機能追加の検討

上述の基準アプリケーションにより，日本語入力である程度の文章を入力できるようになった．しかし，実用的に文章を入力するには下記のような問題があった．

（1）句読点を 1 回のキー入力で入力できない．具体的には句点を入力する場合は「まる」の 2 文字，読点を入力する場合は「てん」の 2 文字を入力して変換する必要があった．

（2）文章の途中を訂正する場合に，その訂正したい場所まで簡単に戻れない．マウスを使用しての手書き文字入力の場合はマウスのクリックで訂正したい場所を指定することができるが，実際に空中で文字入力を行う場合はボタンの入力ができない．そのため，訂正したい場所まで戻るためには今まで書いてきた文字を消去する必要があった．

（3）基準アプリケーションでは清音と濁音を区別して入力しているため，辞書データには清音の他に，濁音・半濁音のが，ざ，だ，ば，ぱ行の計 25 個の辞書データが必要になる．しかし，辞書データの量が多いとそれだけ文字の認識率が下がる問題があった．

（4）日本語入力にしか対応してない．このアプリケーションは全角入力を前提としており，半角入力でのアルファベット入力には対応出来ていない．そのため，アルファベットを入力する場合には日本語入力で入力しなければならない．例として，「abc」と入力する場合は「えーびーしー」と入力し，アルファベットに変換する必要があった．

3.1　特殊キーの追加

（1）句読点キー

1 つ目の問題である，句読点を 1 回のキー入力で入力できない点を解決するために，句点と読点それぞれに特殊キーを割り当てることにした．その動作を Table 2 に示す．句点は時計回り，読点は反時計回りに正三角形を書くことで入力することが出来る．句点は VK_OEM_PERIOD, 読点は VK_OEM_COMMA の仮想キーコードを使用した．

Table 2 Comma and Period keys

キー	句点（ピリオド）	読点（コンマ）
動作		

（2）矢印キー

2 つ目の問題である，文章を遡って訂正しにくいという点を解決するために矢印キーの導入を検討し，上下左右の矢印キーに特殊キーを割り当てた．その動作を Table 3 に示す．右矢印は VK_RIGHT，左矢印は VK_LEFT，上矢印は VK_UP，下矢印は VK_DOWN の仮想キーコードを使用した．

Table 3 Arrow keys

キー	右矢印		左矢印		上矢印		下矢印	
動作								

（3）濁点・半濁点キー

濁点・半濁点キーの動作を Table 4 に示す．濁点キーの使い方としては，清音（か，さ，た，は行）を入力後，濁点キーを入力する．すると入力されていた清音を消去し，代わりに濁音が入力される．同様に半濁点キーも，は行の入力後に半濁点キーを入力することで半濁音を入力することが出来る．

Table 4 Sonant mark keys

キー	濁点	半濁点
動作		

3.2　アルファベット入力の追加

アルファベットやひらがなを入力モードの変更なく自由に入力できることが理想であるが，日本語入力用の辞書データとアルファベット入力用の辞書データとを一緒にすると，ひらがなの「こ」とアルファベットの「Z」のように類似の文字が増えて認識率が下がる問題があった．

今回はその問題を解決するために，半角入力の際はアルファベット用の辞書データを，全角入力の際はひらがな用の辞書データを使用し，半角入力と全角入力の切り替えとともに辞書データも切り替えることにした．

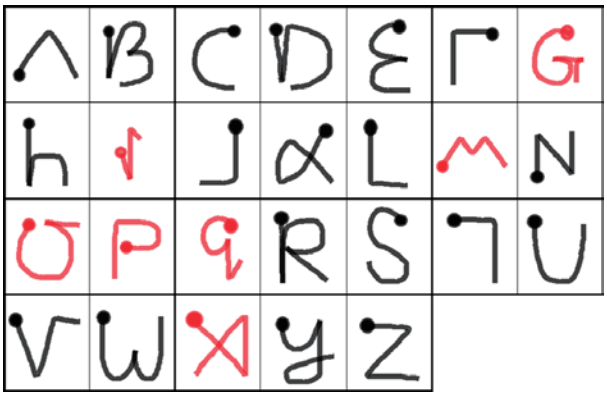

Fig. 4 New alphabet input pattern

アルファベットの辞書データには Palm 社が開発した Graffiti（グラフィティ）をベースにして，既存の数字や特殊キーと似ている文字，一筆では書けない文字については変更を加え新たなアルファベットの辞書データを作成した．Fig. 4 に作成したアルファベットの一覧を示す．なお，Graffiti から変更したアルファベットを赤色で示す．

アルファベットとひらがなの辞書データの切り替えには，半角入力と全角入力の切り替えを利用する．そのために，GetKeyState 関数を使用する．この関数は指定された仮想キーの状態を取得する関数で，戻り値はトグル動作になり，OS や他のアプリケーションの起動により今回対象とするメモ帳起動時の初期値が変化する．

そのため，本アプリケーションの起動時に半角/全角キーの初期状態を保存しておいて，常に状態を判別する必要がある．具体的には，初期化処理において，半角/全角キーの仮想キーである VK_KANJI の状態を取得し，その戻り値を変数 initgetkeystate に格納する．その後の認識処理において，入力した文字の出力先であるメモ帳の半角/全角キーの状態を判別し，先ほど initgetkeystate に格納した初期状態と比較する．値が同じであれば半角，違っていれば全角と判別し，半角の場合は key_fl に「0」を，全角の場合は key_fl に「1」を代入する．その半角/全角切り替えのコードを Fig.5 に示す．

```
BOOL CMousePosCDlg::OnInitDialog()
・・・
        //TODO: 初期化をここに追加します.
        Sleep(100);
        initgetkeystate = GetKeyState(VK_KANJI);
・・・

void CMousePosCDlg::OnTimer(UINT_PTR nIDEvent)
・・・
        if (GetKeyState(VK_KANJI) == initgetkeystate) {     //メモ帳の状態と初期状態を比較
                key_fl = 0;                                 //半角と判別
        } else {
                key_fl = 1;                                  //全角と判別
        }
・・・
```

Fig.5　Code for Alphabet/Hiragana mode change

4. 機能追加の効果比較実験

今回の機能追加の効果を確認するため，第 2 章でアプリケーション化したアプリケーション，第 3 章の機能を追加したアプリケーション，そして iPhone の標準キーボードの 3 つのアプリケーションの入力時間とキーの入力回数を比較した．今回，入力する例文は，「fut タワーで集まろう。」という文で，IME の履歴をリセットした状態で入力した．第 2 章でアプリケーション化したアプリケーションを A，第 3 章の機能を追加したアプリケーションを B，iPhone の標準フリック入力を C とし，その比較結果を Table 5 に示す．

ログ中の記号はそれぞれ，HZ は半角/全角，Sp は変換，Sm は小文字，Tb は予測変換，En は改行の特殊キーの入力を表している．なお，C については，日本語とアルファベットの切り替えキーを HZ，確定キーを En，濁点半濁点小文字キーを Dt，予測変換から選択することを Tb と表記している．

Table 5 Input period and times of each input method

使用アプリ	入力時間	入力回数	入力ログ
A	1 分 53 秒	29 回	HZ えふゆーSpSp てい Smー SpSp 6 たわーTb で En あつまろ Tb まる TbEn
B	49 秒	18 回	ｆｕｔHZ たわーTb て DtEn あつまろ TbPeEn
C	7 秒	21 回	HZ ｆｕｔEnHZ たわーTb て DtEn あつまろう Tb．En

結果として，今回の機能追加により入力時間は 1 分 4 秒，入力回数は 11 回短縮することが出来た．また，入力回数においては，iPhone の標準フリック入力と同等であり，切り替え操作や予測変換の条件により 3 回少ない回数で入力することができた．

5. 結　　言

アプリケーションの使い勝手向上に向けて機能追加の検討を行った．これに伴い，従来からの改行や変換などに加えて，句読点や矢印キーや濁点，半濁点にも特殊キーを割り当て，アルファベット入力も可能にした．このことにより，幅広い文章の入力が可能になり，入力時間と入力回数を短縮することが出来た．これらのことから，空中手書き文字入力の実用化が一歩近づいたと考えられる．

しかし，例えば第一候補として BS（消去）が出力された場合は，第一候補であった BS によって誤って消去された文字は第二候補，第三候補を選択しても戻せない問題がある．これを解決するためには，前の状態の戻す操作を導入することが必要である．また，矢印キーを連続して入力してカーソルを沢山移動させるのは効率が悪いため，指定したキーを指定した回数だけ入力できる繰り返しキーの導入も検討するべきである．

これらの検討，改良を含めて，さらに使い勝手の向上を図りたい．

文　　献

(1) 嵯峨山茂樹，中井満，下平博，“ストローク HMM に基づくオンライン手書き文字認識”，信学技報，PRMU2000-35(2000),pp.1-8.
(2) 園田智也， 岡村洋一，“ 空中での手書き文字認識システム”，信学論（D-II）,Vol.J86-D-II,No.7(2003),pp.1015-1025.
(3) 西田好宏，小倉一孝，三浦浩一，松田憲幸，瀧寛和，安部憲広，“移動方向情報のみを利用した空中手書き文字認識”，ヒューマンインタフェース学会誌，Vol. 12, No.3 (2010), pp. 289-296.
(4) 西田好宏，吉田大志，“「Kinect Interaction」を利用した空中手書き文字入力の検討”，ヒューマンインタフェースシンポジウム 2013 (2013), pp.275-276.
(5) 西田好宏，野口祥子，“一文字単位の区切り操作を必要としない二次元移動方向による手書き文字入力”，ヒューマンインタフェースシンポジウム 2015 (2015), pp.349-352.

（平成 28 年 3 月 31 日受理）