インターフェイスの街角 (22)
POBox サーバーと漢字絵交じり文

増井 俊之

誰もが、用途にかかわりなく気軽に電子メールを使うようになってきました。手紙や電話などでは、筆の運びや声の調子などから相手の気分を推し量ることができます。しかし、通常の電子メールでは決まった文字しか使えないので、相手に気分を伝えるには工夫が必要です。〝!〟や〝?〟などの記号類もすこしは利用できますが、複雑な感情を伝えるには不十分です。

NetNews や Web の掲示板で口論が絶えないのは、テキストを中心としたメディアでは気分を伝えにくいことが一因かもしれません。これまで、電子メールはおもに仕事や研究の話をするために使われていたので、感情を伝えられなくてもさして問題にはなりませんでした。しかし、一般の人が電話や手紙の代わりに利用するにはなんらかの工夫が必要です。最近、ASCII 文字を並べた〝\^o^/〟や〝^_^;〟などの文字を使った表現をよくみかけますが、感情がうまく伝わるとはかぎりませんし、やや無理な方法をとっているせいか、なんとなく間が抜けて見えます。最近は HTML メールも普及してきたので、いっそのこと本当の絵をメール本文に埋め込めば、このような状況もいくぶんか改善されるのではないでしょうか。文章にさまざまな絵を簡単に入れることができれば新しいメール文化が生まれるかもしれません。今回は、このような〝漢字絵交じり文〟を漢字入力と同様なインターフェイスで作成できるシステムを紹介します。

絵文字の入力と編集

絵を含む文字列の編集には、XEmacs や Tk のテキスト編集ツール、Web ページ作成ツールなどが使えます。さらに、絵文字を含む文書を作るためのメール・ソフトウェアも販売されています。これらのツールを利用すれば、絵を含む文字列を比較的簡単に編集できます。しかし、絵の入力はそれほど容易ではありません。それぞれの絵はたいていは別のファイルになっているので、ユーザーは最初に目的の絵をなんらかの方法で検索し、そのファイル名を指定してテキストに絵を埋め込むのが一般的な方法でしょう。しかし、この手法ではファイルの検索や選択にかなりの手間がかかります。

よく考えれば、絵の検索や入力はかな漢字変換による漢字の入力と同じ方法でもできるはずです。たとえば、〝warai〟という入力をたんに〝笑い〟と変換するだけでなく、〝:-)〟などの文字列や〝  〟といった絵も候補として表示するようにすれば、ユーザーは必要なものを選ぶだけで簡単に目的の絵を入力できます。ワープロを使うと手で書けない漢字でも入力できるのと同様、これらのシステムを利用すれば自分では描けない絵を選んで絵交じりの文章が作れるでしょう。

絵文字入力システム

1998 年 4 月号と 5 月号で紹介した、ペン型計算機用のテキスト入力システム **POBox**[1]を改造して絵文字入力システムを作成できます。POBox では、入力したい単語の最初の何文字かをユーザーが指定すると、その文字で始まる複数の単語が候補として表示され、そのなかから目的の単語を選択して文章を作成していきます。候補として、文字列だけでなく絵も表示すれば、普通の POBox とまったく同じインターフェイスで絵を含む文章が作れます。

POBox は、SKK や Wnn などの日本語入力システム

1 http://www.csl.sony.co.jp/person/masui/POBox/

図 1　UNIX 用 POBox

図 2　初期画面

図 3　`m´を入力

と同様に、指定された読みから単語を検索する部分と、読みの入力や結果の表示などのインターフェイス部分に分けて実装できます。検索サーバーはアプリケーションや OS と独立に作れますし、うまく実装すればペン計算機でもキーボードでも共通に使えるようになります。

絵文字入力の例

Tcl/Tk のアプリケーションをクライアントとして、文字や絵を入力していく様子を以下に示します。Tk のテキスト・ウィジェットでは画像やボタンも文字と同じように扱えるので、このようなクライアントも簡単に実装できます。

図 2 の初期画面で `m´を入力すると、「虫」や「マイク」「増井」など、`m´で始まる絵や単語が候補ボタンとしてリストされます（図 3）。

続いて `a´と `s´を入力すると、`mas´で始まる絵や単語だけがリストされます（図 4）。ここでスペースキーを押すと候補を順番に選択でき（図 5）、リターンキーを押して確定します（図 6）

図 4　`mas´を入力

図 5　候補選択

図 6　候補確定

図 7　`no´の入力/選択

同様にして図 7〜8 のように文字や絵を入力していくと、最終的に図 9 のような絵交じりの文が作成できます。

まず絵の `読み´を入力して候補を表示させ、そのなかから希望のものを選ぶ方法は、従来のかな漢字変換とほとんど変わりません。また、Tk のテキスト・ウィンドウでは、通常の Emacs と同じキー入力でカーソルを移動させることができるので、Emacs でかな漢字変換をおこないながら文章を書くのとほとんど同様な感覚で絵交じり文が

図 8　`ha´を入力

図 9　絵交じり文の完成

図 10　HTML による絵交じり文

作れます。

HTML への変換

このようにして作成した絵入りの文章を HTML に変換すれば、Web ブラウザで参照したり、HTML 対応のメールツールで読んだりすることが簡単にできるようになります。図 10 は、図 9 の絵交じり文を HTML に変換したものを Netscape で表示したところです。それぞれの絵文字を<img>タグで表現した HTML 文（図 11）で、絵を含む文を手軽に表現できます。

この例では、HTML と画像が別々のファイルになっていますが、これらのファイルをまとめて MIME エンコーディングすれば、メールでも絵交じり文を送れるようになります。MIME のエンコーディングには、Bellcore (Bell Communications Research) が開発した Metamail パッケージ[2]に含まれる mimencode コマンドや、mimencode でエンコードしたデータをまとめて 1 つのメール文

2 ftp://ftp.bellcore.com/pub/nsb/

図 11　HTML 文

```
<html>
<body bgcolor=white>
<img src="masui.gif" align=center>
の方法はとっても
<img src="heart.gif" align=center>
です
<img src="exclaim.gif" align=center>
<br>
<br>
</body>
</html>
```

図 12　html2mail スクリプト

```
#!/usr/local/bin/perl

$file = shift;
open(in,$file) || die "Can't open $file";

$tmpfile = "/tmp/msg$$";
open(out,"> $tmpfile") ||
    die "Can't create message file";

$n = 0;

while(<in>){
  chop;
  if(/^(.*img src=")(§+)(".*)$/){
    $head = $1; $tail = $3;
    $image[$n] = $2;
    $_ = "${head}cid:image${n}${tail}";
    $n++;
  }
  print out "$_\n";
}
close(out);

$command = "metasend -b -o mailtext ⇒
            -/ related ";
$command .= "-f $tmpfile -m text/html ⇒
             -e 7bit ";
for($i=0;$i<$n;$i++){
  $command .= "-n -f $image[$i] ⇒
               -m image/gif -i image$i ";
}

system $command;
unlink $tmpfile;
```

（誌面の都合上、⇒ で折り返しています）

書とする metasend コマンドが使えます。図 12 に示した Perl スクリプト **html2mail** のなかで metasend コマンドを呼び出すと、図 11 の HTML ファイルから図 13 のような MIME エンコードされたテキストが作成で

図 13　metasend で生成されたメールテキスト

```
MIME-Version: 1.0
Content-ID: <Thu_Aug_10_20_20_03_JST_1999_4@bird>
Content-type: multipart/related;
boundary="bird.14665.Thu.Aug.10.20:20:03.JST.1999"
                    ……
--bird.14665.Thu.Aug.10.20:20:03.JST.1999
Content-ID: <Thu_Aug_10_20_20_02_JST_1999_0@bird>
Content-type: text/html
Content-Description: An object packed by metasend
Content-Transfer-Encoding: 7bit

<html>
<body bgcolor=white>
<img src="cid:image0" align=center>
の方法はとっても
<img src="cid:image1" align=center>
です
<img src="cid:image2" align=center>
<br>
<br>
</body>
</html>

--bird.14665.Thu.Aug.10.20:20:03.JST.1999
Content-ID: image0
Content-type: image/gif
Content-Description: An object packed by metasend
Content-Transfer-Encoding: base64

R0lGODlhIAAgAPYAAACtAACsAADjAADhAAC5AAC4AADTAADSAADGAADEAADeAADbAAClAACi
AAAtAAAqAAAwAAAuADo6Ojk5OUpKSkhISAsLCwoKCgAcAAAbAEJCQkBAQAAnAAAmAADRAACZ
AACVAABMAABIAADPAADLAAC2AAC+AAC9AADCAADfAOHh4d7e3j8/P+H94d393en+6eX95ezs
                    ……
```

きます。

サーバーとクライアントの実装

POBox サーバー

POBox サーバーは、クライアントから単語の読みが与えられると、その読みに対応する候補単語の集合を返します。SKK サーバーや jserver では単語や文節の完全な読みをもとに候補となる文字列を計算しますが、POBox サーバーは単語の部分的な読みしか与えられていない場合、あるいは読みがまったく与えられていない場合も、コンテクストなどから入力単語を予測して候補単語を返します。たとえば、〝お〟という読みが指定されたとき、POBox サーバーは〝おり〟〝思い〟〝同じ〟などを候補単語としてクライアントに返します。このとき、その直前に〝よろしく〟という文字列があった場合には、〝お願い〟などの単語を候補として返します。このように、コンテクストにもとづいて入力単語を予測するため、クライアントは必要に応じてサーバーにコンテクストを通知します。

POBox サーバーを絵交じり文対応にするのはごく簡単で、候補単語として漢字文字列を返す代わりに画像ファイル名を返すだけです。たとえば、読みとして〝じどうしゃ〟が指定され場合は、〝自動車〟のような文字列のほかに、〝car.gif〟といった画像ファイル名も候補として返します。

POBox クライアント

POBox サーバーを使うクライアントは、ユーザーが単語や絵の読みを入力するたびに、その読みをサーバーに通知して候補となる文字列や画像ファイル名を取得します。候補が画像の場合は、文字列の代わりに画像を表示します。目的の文字列や画像が候補に含まれていれば、ユー

図 14　Emacs をクライアントとした例

ザーはそれを選択してテキストに入力します。

読みの入力や候補選択には、ハードウェアやソフトウェアに合った適当な方法が選べます。さきほど紹介した Tcl/Tk によるクライアントの例では候補をボタンとして表現していました。一方、Emacs をクライアントとする場合には、キーボードから入力した読みをサーバーに通知して候補単語を取得し、カーソルの下に並べて表示して選択できるようにします。こうすれば、通常のかな漢字変換と同様のインターフェイスで漢字の入力ができます（図 14）。

ペン計算機や携帯電話などの場合も、ソフトキーボードやテンキーで読みを指定したり、ペンタップやジョグダイヤルなどで候補を選ぶようにすれば、効率的な文章や絵の入力が可能になります。

おわりに

一般に、文章を書いたり絵を描くこととくらべると、読んだり見たりするのはそれほど難しくはありません。書きたい文章や描きたい絵をすべて自分で作らなくても、辞書や例文集、画像リストなどから選んで文章を作るようにすれば、より簡単に表現力豊かなメールがやりとりできるようになるでしょう。私の周囲では「HTML メールが届いたら捨てる!」と豪語している人も多いのですが、今後どんな端末でも HTML メールが読めるようになり、簡単な操作で絵交じりメールが作れるようになれば、こうした状況も変わるかもしれません。

私は、以前は POBox をもっぱらペン計算機上で使っていましたが、POBox サーバーを実装してからさまざまな環境で予測入力が利用できるようになりました。最近は、Emacs 上の日本語入力にはすべて POBox を使っていますし、電話のようにキーの数が少ない装置への対応も考えています。環境の違いやユーザーの熟練度に応じた柔軟なインターフェイス・デザインを**ユニバーサル・デザイン**といいますが、POBox をさらにこれに近づけるべく工夫を加えていくつもりです。

（ますい・としゆき　ソニー CSL）

付録：Tcl/Tk による絵交じり文入力クライアント

図 2 の絵交じり文入力プログラムを以下に示します。通常モードと漢字/絵入力モードは、Ctrl-J で切り替えることができます。漢字/絵入力モードでは、指定した読みに従って候補が表示され、スペースキーで選択、リターンキーで確定します。このようなプログラムを使えば、Tk のテキスト・ウィンドウでも簡単に漢字入力ができるようになります。

Ctrl-W を押すと HTML ファイルがダンプされるので、html2mail スクリプト（図 12）を使ってメールメッセージとすることもできます。

```
set host localhost
set hostenv [array get env PBSERVER]
if { [llength $hostenv] == 2 } { set host [lindex $hostenv 1] }
set pbserver [socket $host 1178]
set maxcands 10
set candstart 0.0
set candend 0.0
set poboxmode 0
set candindex 0
set cands ""
set pat ""
set exact 0
set sjis [expr [file exists "C:"] ? 1 : 0]

text .text -relief sunken -bd 2 -setgrid 1 -height 20 -width 50
pack .text -expand yes -fill both
```

```
proc euc2sjis { e } {
  set len [string length $e]
  set i 0
  set ret ""
  while { $i < $len } {
    binary scan [string index $e $i] c c1
    if { $c1 < 0 } {
      set c1 [expr $c1 + 0x100]
      incr i
      binary scan [string index $e $i] c c2
      incr i
      set c2 [expr $c2 + 0x100]
      set c2 [expr $c1 & 1 ? $c2 - 0xa1 + 0x40 : $c2 - 0xa1 + 0x9e]
      set c1 [expr ( $c1 - 0xa1 ) / 2 + 0x81]
      set s1 [expr $c1 >= 0xa0 ? $c1 + 0x40 : $c1]
      set s2 [expr $c2 >= 0x7f ? $c2 + 1 : $c2]
      set s1 [expr $s1 - 0x100]
      set sj [binary format cc $s1 $s2]
      set ret $ret$sj
    } {
      set s [string index $e $i]
      set ret $ret$s
      incr i
    }
  }
  return $ret
}

proc normalall { } {
  global cands maxcands
  set candlen [llength $cands]
  for { set i 0 } { $i < $candlen - 1  && $i < $maxcands } { incr i } {
    .text.b$i configure -state normal
  }
}

proc erasecands { } {
  global candstart candend
  .text delete $candstart $candend
  set candstart [.text index insert]
  set candend $candstart
}

proc initcand { context } {
  global cands candindex pat exact
  erasecands
  set candindex 0
  set cands ""
  set pat ""
  set exact 0
}

proc henkan { pat exact } {
  global pbserver cands candindex sjis maxcands candstart candend
  set p 1$pat
  if { $exact == 0 } { set p "$p " }
  puts $pbserver $p
  flush $pbserver
  set r [gets $pbserver ]
  set cands [split $r / ]
  if { [string compare [lindex $cands 0] 1] == 0 } {
    set len [expr [llength $cands] - 1 ]
    set cands [lrange $cands 1 $len ]
    set cands [linsert $cands 0 $pat]
  } {
```

```
    set len 0
    set cands ""
  }
  set candindex 0
  set candstart [.text index insert]

  for { set i 0 } { $i < $maxcands } { incr i } {
    set b .text.b$i
    destroy $b
     button $b -text $b -command "buttonproc $b" \
        -activeforeground #0000ff -activebackground #ffff00
    set t [lindex $cands $i]
    if { $sjis } { set t [euc2sjis $t] }
    if { $i < $len } {
      if { [string last .gif $t] == -1 } {
        $b configure -image ""
        $b configure -text $t -state normal
      } {
        set image [image create photo -file "icons/$t"]
        $b configure -image $image
        $b configure -text ""
      }
      .text window create insert -window $b
      $b configure -state normal
    }
  }
  set candend [.text index insert]
  .text.b0 configure -state active
}

proc buttonproc { b } {
  set image [lindex [$b configure -image] 4]
  if { [string compare $image ""] == 0 } {
    .text insert insert [lindex [$b configure -text] 4]
  } {
    .text image create insert -image $image
  }
  initcand ""
}

proc insert { n } {
  buttonproc .text.b$n
}

proc htmldump { file } {
  set text [.text dump -all 0.0 end]
  set len [llength $text]
  set ch [open $file w]
  puts $ch "<html>"
  puts $ch "<head><title></title></head>"
  puts $ch "<body bgcolor=white>"
  for {set i 0} {$i * 3 < $len } {incr i} {
    set base [expr $i * 3]
    set type [lindex $text $base]
    if {[string compare $type text] == 0} {
      set s [lindex $text [expr $base + 1]]
      regsub "\n" $s "<br>" t
      puts $ch $t
    }
    if {[string compare $type image] == 0} {
      set s [lindex $text [expr $base + 1]]
      set f [lindex [$s configure -file] 4]
      puts $ch "<image src=\"$f\" align=center>"
    }
  }
```

```
  puts $ch "</body>"
  puts $ch "</html>"
  close $ch
}

proc poboxkey { w a } {
  global pat poboxmode candindex exact
  if { $poboxmode &&
     ([string compare $a a] >= 0 && [string compare $a z] <= 0 ||
      [string compare $a -] == 0 || [string compare $a .] == 0)} {
    set exact 0
    if { $candindex > 0 } {
      insert $candindex
      set pat $a
    } {
      if { [string length $pat] > 0 } {
        erasecands
      }
      set pat $pat$a
      henkan $pat $exact
    }
  } {
    tkTextInsert $w $a
  }
}

bind Text <Key> {
  poboxkey %W %A
}

bind Text <Control-j> {
  if { $poboxmode } {
    set poboxmode 0
    initcand ""
  } {
    set poboxmode 1
  }
}

bind Text <space> {
  set candlen [llength $cands]
  if { $poboxmode && $candlen > 0 } {
    set len $maxcands
    if { $candlen < $len } { set len $candlen }
    normalall
    if { $candindex < $len - 1} {
      incr candindex
    }
    .text.b$candindex configure -state active
  } {
    tkTextInsert %W %A
  }
}

bind Text <Return> {
  if { $poboxmode && [llength $cands] > 0 } {
    insert $candindex
  } {
    tkTextInsert %W \n
    tkTextSetCursor %W insert
  }
}

proc bs { w } {
  global poboxmode pat exact
```

```
  set exact 0
  set len [string length $pat]
  if { $poboxmode && $len > 0 } {
    set pat [string range $pat 0 [expr $len - 2]]
    erasecands
    if { [string length $pat] > 0} {
      henkan $pat $exact
    }
  } {
    if {[$w tag nextrange sel 1.0 end] != ""} {
      $w delete sel.first sel.last
      tkTextSetCursor $w insert
    } else {
      tkTextSetCursor $w insert-1c
      $w delete insert
      $w see insert
    }
  }
}

bind Text <BackSpace> "bs %W"
bind Text <Delete> "bs %W"

bind Text <Tab> {
  if { $poboxmode && [string length $pat] > 0} {
    set exact 1
    henkan $pat $exact
  } {
    tkTextInsert %W "\t"
  }
  focus %W
  break
}

bind Text <Key-Left> {
  if { $poboxmode && $candindex > 0 } {
    set candindex [expr $candindex - 1]
    normalall
    .text.b$candindex configure -state active
  } {
    tkTextSetCursor %W insert-1c
  }
}

bind Text <Control-k> {
  if { $poboxmode && [string length $pat] > 0} {
    henkan $pat 1
    insert 2
  } {
    if {!$tk_strictMotif} {
      if {[%W compare insert == {insert lineend}]} {
        %W delete insert
      } else {
        %W delete insert {insert lineend}
      }
    }
  }
}

bind Text <Control-w> {
  htmldump pobox.html
}
```