Tutorial Day at MobileHCI 2008, Amsterdam

Text input for mobile devices by Scott MacKenzie

Scott will give an overview of different input means (e.g. key-based, stylus, predictive, virtual keyboard), parameters relevant for designing and assessing mobile text input (e.g., writing speed, cognitive load) and issues related to the context of use (e.g., walking/standing).

Mobile GUIs and Mobile Visualization by Patrick Baudisch

Patrick will introduce different approaches for creating mobile graphical user interfaces. He will talk about the design process, prototyping and assessment of user interfaces, trade-offs related to the design of mobile GUIs and different possible interaction styles. As one specific topic in mobile GUIs he will address concept for mobile interactive visualization (e.g. maps).

Understanding Mobile User Experience by Mirjana Spasojevic

Mirjana will discuss different means for studying mobile user needs and evaluating the user experience. This includes explorative studies and formal evaluations (in the lab vs. in the field), including longitudinal pilot deployments. The lecture will discuss traditional HCI methods of user research and how they need to be adapted for different mobile contexts and products.

Context-Aware Communication and Interaction by Albrecht Schmidt

Albrecht will give an overview of work in context-awareness and activity recognition that is related to mobile HCI. He will discuss how sharing of context in communication applications can improve the user experience. The lecture will explain how perception and sensing can be used to acquire context and activity information and show examples how such information can be exploited.

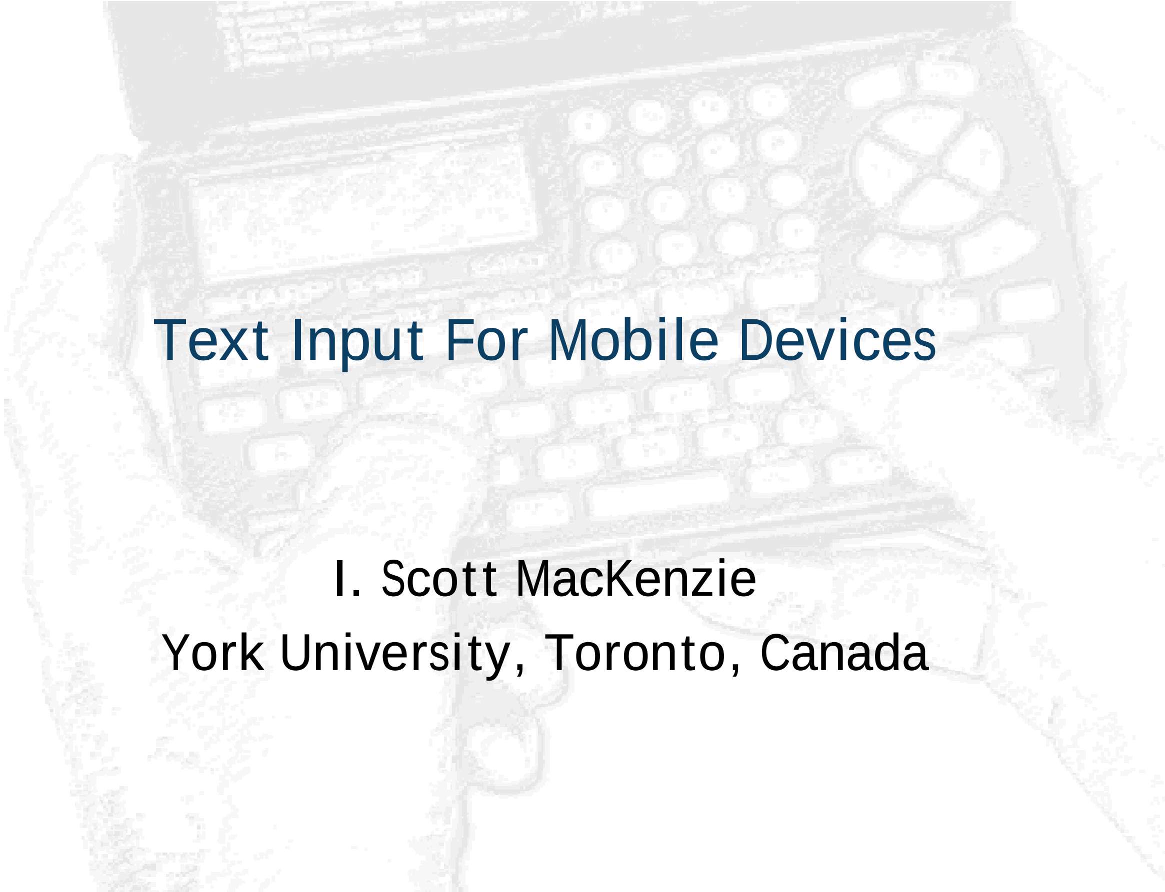
Haptics, audio output and sensor input in mobile HCI by Stephen Brewster

Stephen will discuss the design space for haptics, audio output as well as sensor and gesture input in mobile HCI. Furthermore he will assess resulting interaction methods and implications for the interactive experience.

Camera-based interaction and interaction with public displays by Michael Rohs

Michael will introduce you camera based interaction with mobile devices; this includes a assessment of optical markers, 2D-barcodes and optical flow as well as techniques related to augmented reality. In this context he will address interaction with public displays, too.

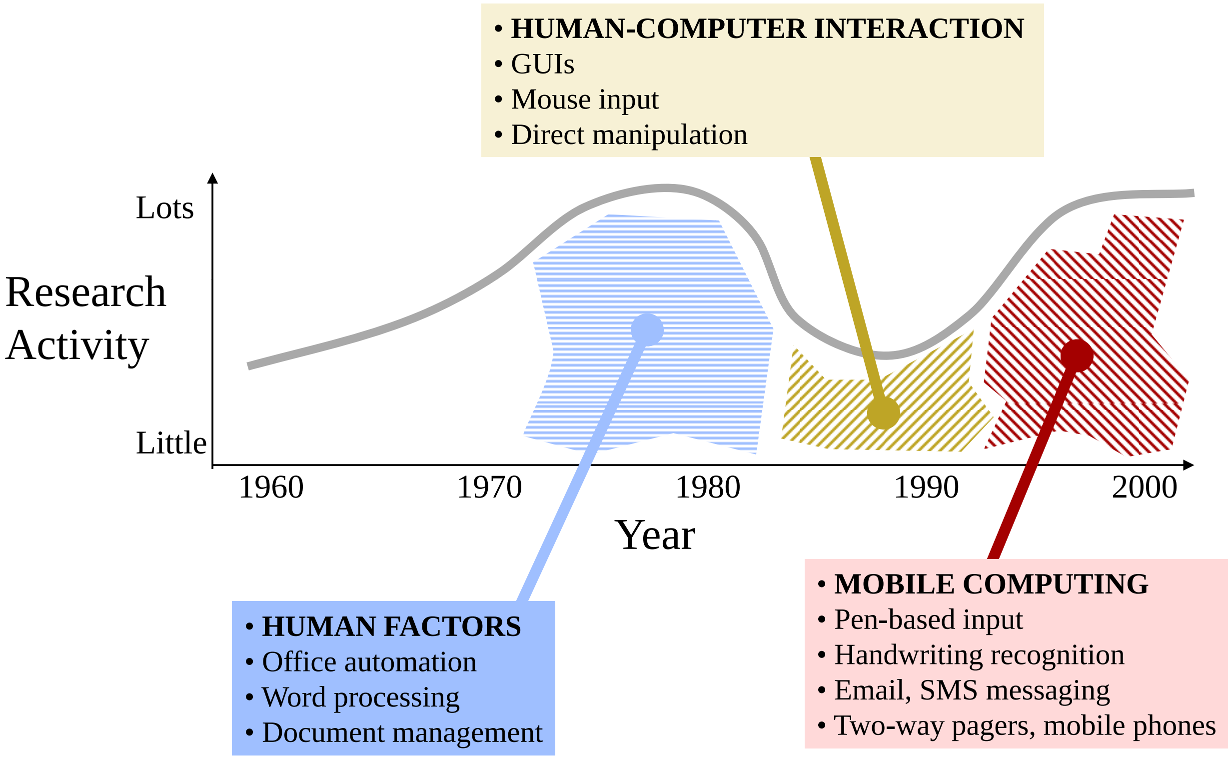
Spetember 2nd 2008



Text Input For Mobile Devices

I. Scott MacKenzie
York University, Toronto, Canada

Text Entry Research - Timeline



Mobile Text Entry

- >1 billion SMS messages sent each day
- Companies are ambitiously searching for improvements to mobile text entry techniques
- Many methods currently exist

Three broad categories

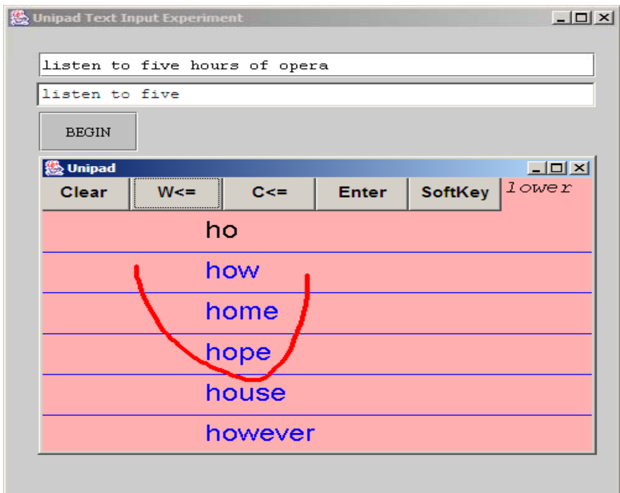
- Key-based
- Finger-based
- Stylus-based

Stylus Based Methods

- Handwriting with automatic recognition
- Tapping on soft or virtual keyboards.

Handwriting Example: Unipad

- Single-stroke handwriting recognition
- Language-based acceleration



KSPC ≈ 0.5
Entry speed > 40 wpm

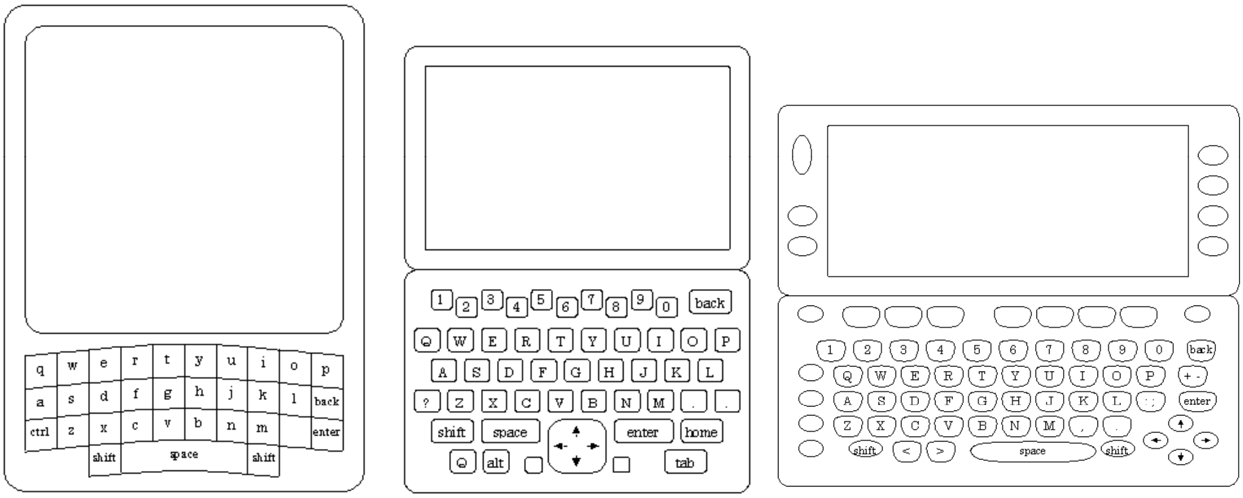
Virtual vs Physical Keyboards

- Virtual keyboards
  - Aka “soft keyboards” or “on-screen keyboards”
  - Similar to clicking buttons in a GUI
  - Typically used with a stylus (but also with a finger, eye tracking, and other input mechanisms)
- Physical keyboards
  - Desktop qwerty, miniature qwerty, mobile phone keypad, 5-button pager, 3-key date stamp, 1-key input, etc.
- Design Issues
  - Key layout, key size, key shape, number of keys, activation force, feedback, disambiguation, language modeling, word prediction, etc.

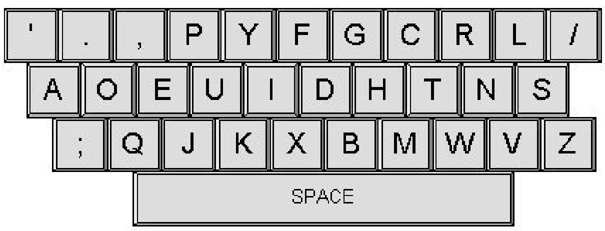
Keyboard Layouts – A Brief Tour

Qwerty

Design issues
- Designed to be slow!
- Prevents typing machines from jamming

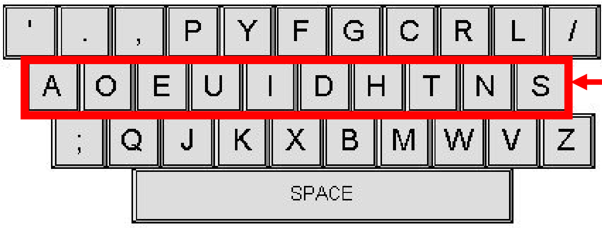


Qwerty variations



Dvorak

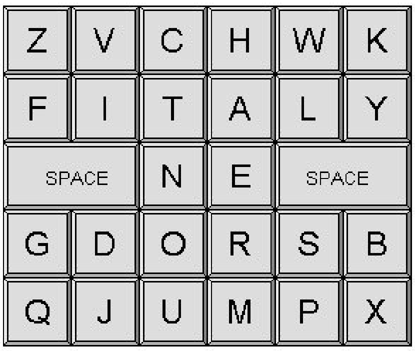
Design issues
- Speed typing by maximizing home row and alternate hand typing



Dvorak

Home row contains 10 of 13 most frequent letters

Letter	Frequency
_	67962112
e	**37907119**
t	**28691274**
a	**24373121**
o	**23215532**
i	**21820970**
n	**21402466**
s	**19059775**
h	**18058207**
r	17897352
l	11730498
d	**10805580**
c	8982417
u	**8022379**
f	7486889
m	7391366
w	6505294
y	5910495
p	5719422
g	5143059
b	4762938
v	2835696
k	1720909
x	562732
j	474021
q	297237
z	93172



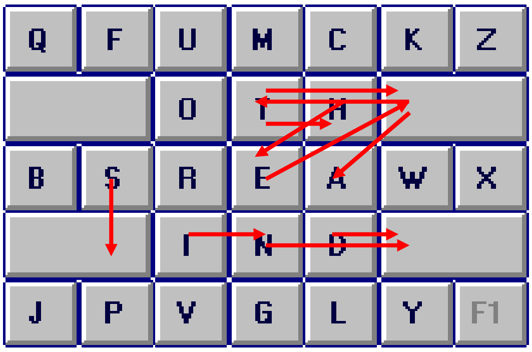
Fitaly

Opti

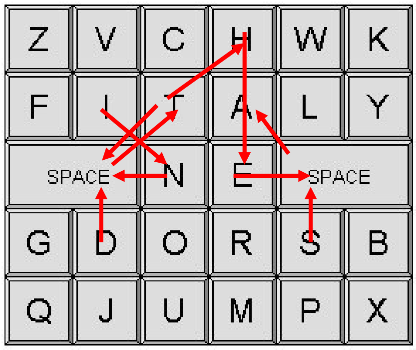
Design issues
- Stylus input on “soft keyboard”
- Speed entry by minimizing stylus movement for English text

Top 10

Digram	Frequency
e_	14788053
_t	11565380
th	10301807
he	9436372
_a	8454634
s_	8359914
n_	6419069
t_	6336756
d_	6235838
in	5534329
...	...
uz	626
zl	626
mh	613
lh	584
cn	567
oq	547
aa	546
xq	543
wm	540
ij	536



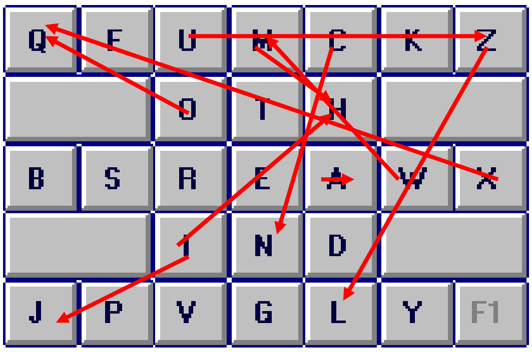
Opti



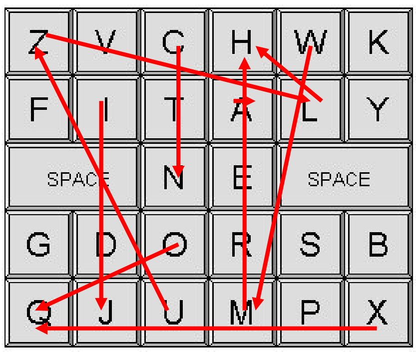
Fitaly

Digram	Frequency
e_	14788053
_t	11565380
th	10301807
he	9436372
_a	8454634
s_	8359914
n_	6419069
t_	6336756
d_	6235838
in	5534329
...	...
uz	626
zl	626
mh	613
lh	584
cn	567
oq	547
aa	546
xq	543
wm	540
ij	536

Bottom 10



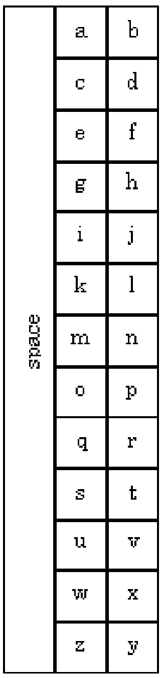
Opti

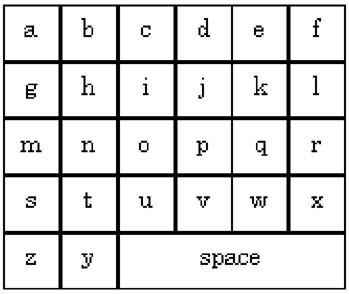


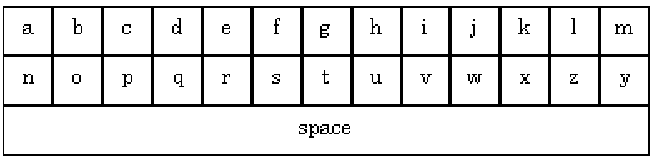
Fitaly

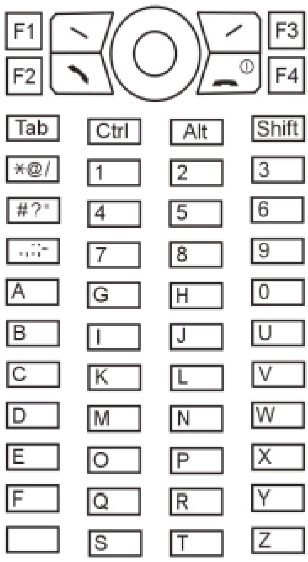
Performance Issues

- *Fitaly* and *Opti* were designed to improve performance by minimizing movement distances (for English text entry)
- Is this a reasonable design goal?
  - Stylus input – YES
    - Movement is slow (Fitts' law)
    - Selection is fast (tap key)
    - Movement time >> selection time
  - Eye gaze input – NO
    - Movement is fast (saccades)
    - Selection is slow (dwell on key)
    - Movement time << selection time



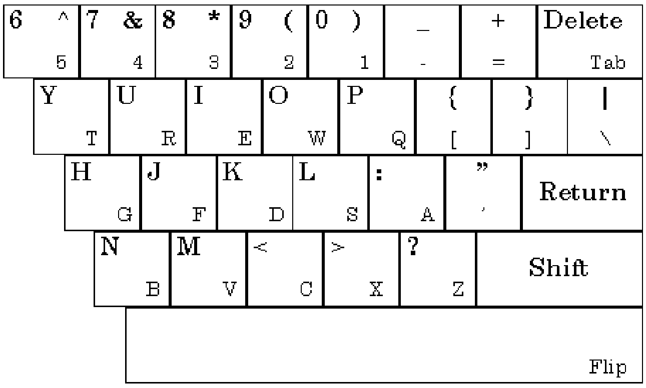






ABC

Design issues
- Non-qwerty shape
- Familiar letter arrangement

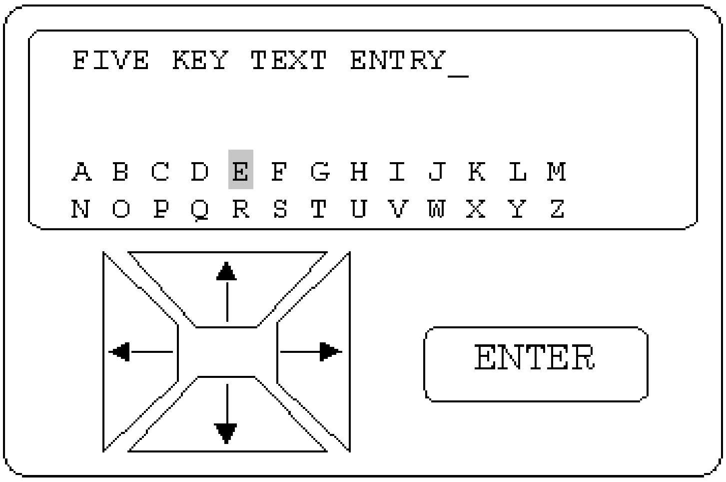


Half Qwerty

Design issues
- One-handed input

Phone

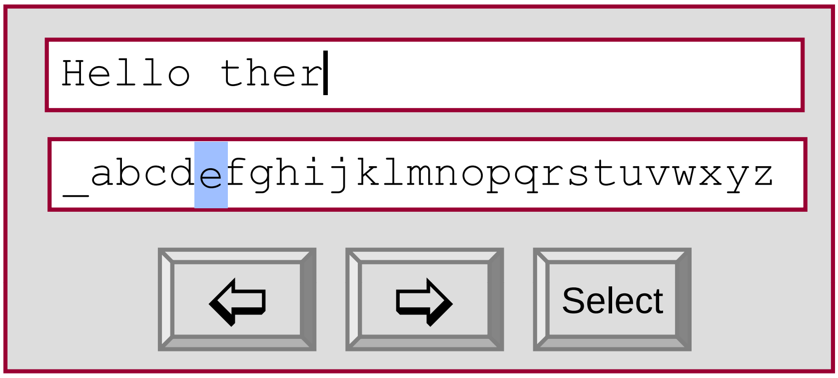
Design issues
- Legacy technology
- Includes letters for telephone exchange names



Five-key pager

Design issues
- Very small device with just 5 buttons

MOBILEHCI 2008

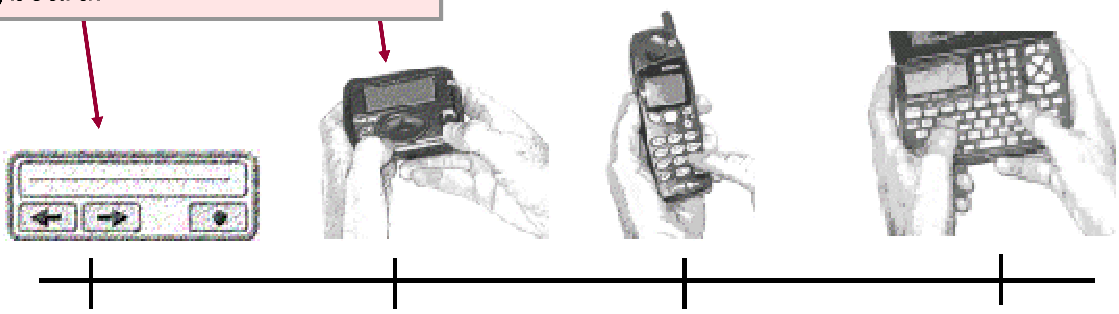


Three-key date stamp

Design issues
- Very small device with just 3 buttons
- Predictive techniques to increase entry speed

Number-of-keys Continuum

But…
The "keyboard" is on the display. The "keys" navigate the keyboard!

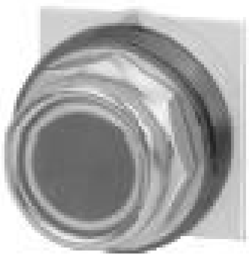


3 5 8-12 26+

Number of Keys

More **Ambiguity Continuum** Less

One-key Input



Is this possible?

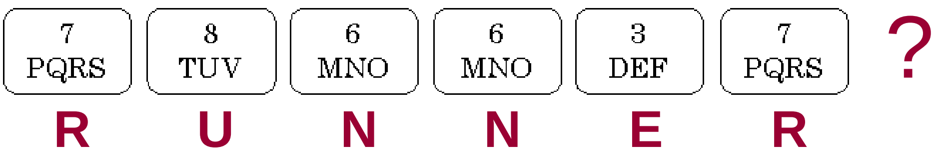
...sure

One-key Text Input

Design Issues
- What moves (a hot spot vs the underlying letters)
- Hot spot speed (places upper limit on entry speed, but faster = more errors)
- Hot spot path (cyclic vs reset on select)
- Letter arrangement (ABC vs. optimized)
- Error correction
- Combine with language acceleration techniques

Ambiguity

- Ambiguity occurs if there are fewer keys than symbols in the language
- Disambiguation is needed to select the intended letter from the possibilities
- Phone keypad is a typical example



Or, is it **SUMMER**, is it **STONES** **?**

Disambiguation Solution #1

- Multi-tap

7	8	6	6	3	7
PQRS	TUV	MNO	MNO	DEF	PQRS

```
RUNNER = 7778866n6633777
           R U N  N E  R

SUMMER = 7777886n633777
            S UM M E  R

STONES = 77778666N66337777
            ST  O  N E   S
```


Disambiguation Solution #2

- Dictionary-based (T9)

7	8	6	6	3	7
PQRS	TUV	MNO	MNO	DEF	PQRS

```
RUNNER = 786637nn
         RUNNE  R

SUMMER = 786637
         SUMMER

STONES = 786637n
         STONE S
```

Demo

`java T9 d1-wordfreq-phoneks.txt -a`

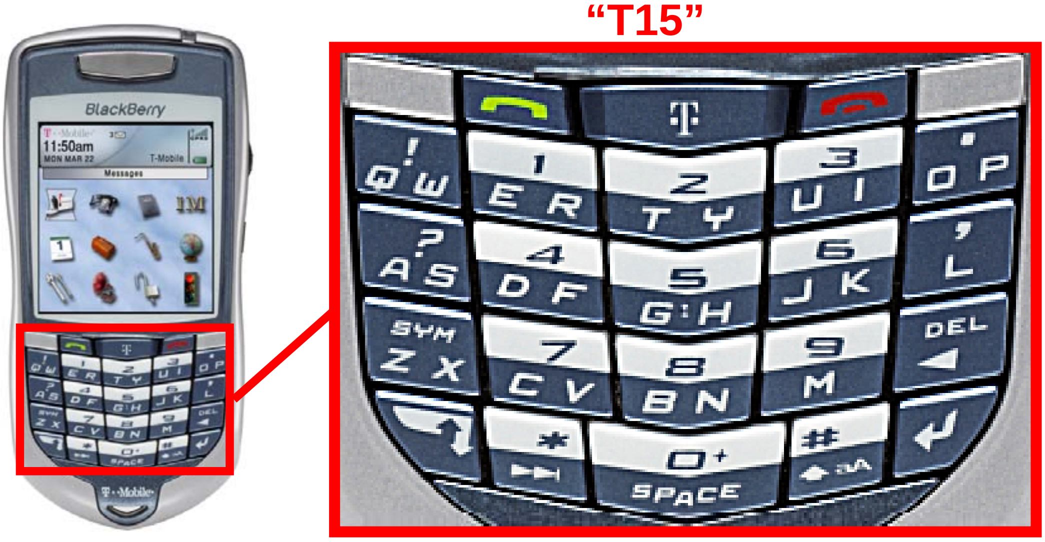
```
DOS
text>java T9 d1-wordfreq-phoneks.txt -a | more
ca ac bc
bbc cab
bad ace
acid cage
able cake bald calf
baker cakes
ball call
calls balls
can ban
came band
car bar cap
case care base card bare cape
based cared
cases cards acres bases
basin cargo
cars bars bass
carries barrier carrier
barriers carriers
carry barry
cast cart
act cat bat
catch batch
acts cats
cause abuse
```


MOBILEHCI 2008

T9-Qwerty Hybrid

- 7100t Blackberry by RIM (Research In Motion)

“T15”



Word Completion

- Basic problem…
  - Given preceding text, predict subsequent text
- Design issues
  - Dynamic vs. static language model
  - Word-level or phrase-level prediction
  - Size of candidate word list
  - Candidate word selection
  - Perceptual cost of attending to predictions
  - Improving performance (audio feedback?)

Demo: `java WordPredict d1-wordfreq.txt 10`

Word Completion Example

- Consider the word “vegetable”
  - How many and what keystrokes are required?

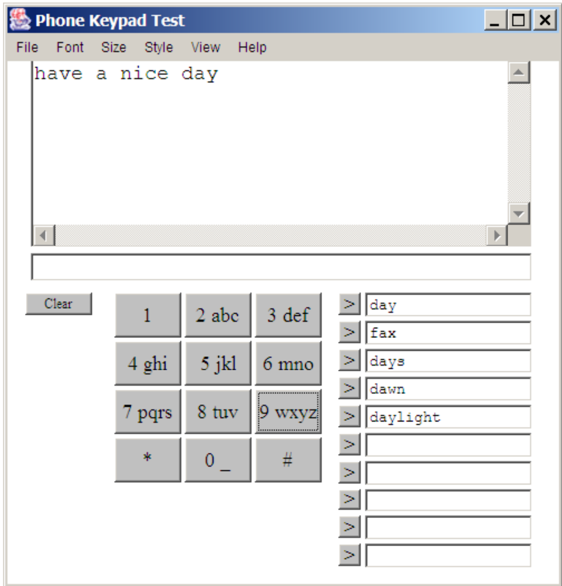
Word Stem	Candidate List
v	very voice view value various
ve	very version vehicle vehicles versions
veg	vegetables vegetation vegetable vegetarian vegetarians

4 keystrokes, KSPC = 0.4

```
vegetable 979 vegS      (stylus input, n = 5)
vegetable 979 vegNNS    (keypad input, n = 5)
```

6 keystrokes, KSPC = 0.6

Disambiguation + Word Completion



Demo: `java PhoneKeypad d1-wordfreq-phoneks.txt`

MOBILEHCI 2008

KSPC[a] Characteristics

Method	KSPC
Date Stamp #1	10.7112
Date Stamp #2	10.5507
Date Stamp #3	9.2143
Date Stamp #4	6.4150
Date Stamp #5	4.9230
Date Stamp #6	4.1032
5-key pager	3.1248
Multitap	2.0242
MessageEase	1.8157
LetterWise	1.1467
T9	1.0064
Qwerty	1.0000
wp-keypad-10	0.7939
wp-keypad-5	0.7293
wp-stylus-1	0.7176
wp-keypad-1	0.7176
wp-keypad-2	0.6867
wp-stylus-2	0.6272
wp-stylus-5	0.5366
wp-stylus-10	0.4896

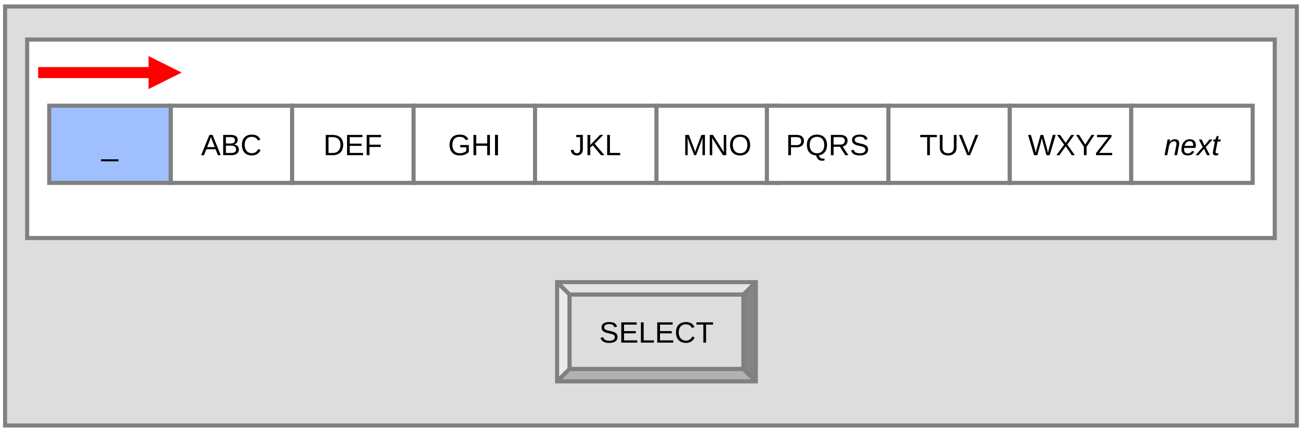
KSPC > 1

KSPC < 1

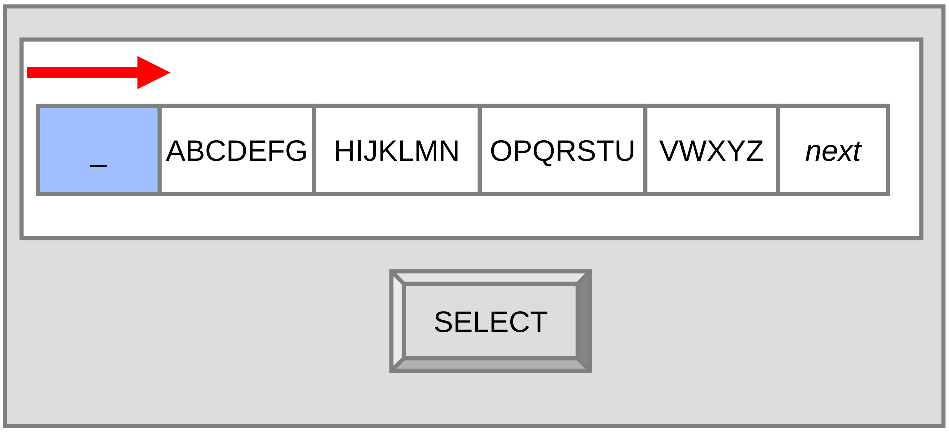
[a] Keystrokes entered per character of text generated

T9 With One Key

- A one-key implementation is possible
  - Advantage: fewer steps to reach desired letter
  - Disadvantage: disambiguation necessary



Fewer Steps



T9 With Eye Gaze

- T9 works remarkably well (KSPC ≈ 1)
- An eye gaze implementation is possible
  - Advantage: large keys
  - Disadvantage: disambiguation necessary

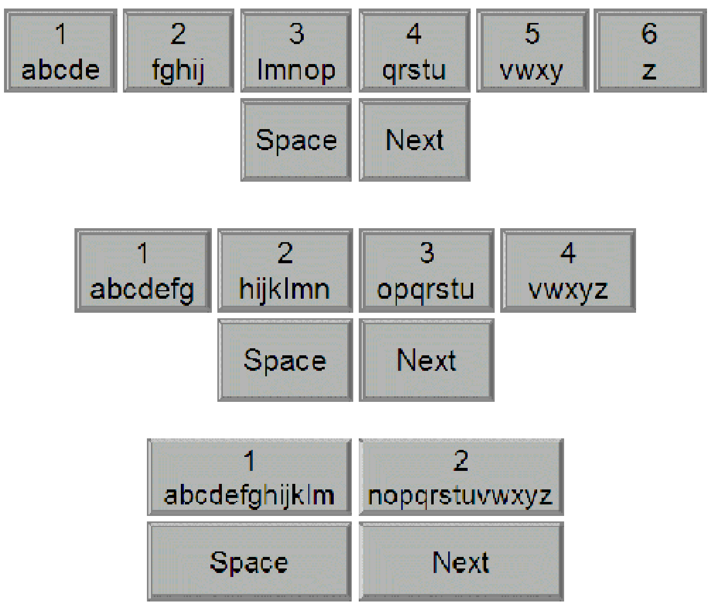
KSPC = 1.0064

	ABC	DEF
GHI	JKL	MNO
PQRS	TUV	WXYZ

KSPC = 1.0670

ABCDEFG	HIJKLMN
OPQRSTU	VWXYZ

KSPC/Ambiguity Demo



Demo: `java T9 d1-wordfreq-k6ks.txt -k`
Demo: `java KSPCWords d1-wordfreq-t6ks.txt`

What's Ahead

- Performance issues
  - Measurement of entry speed
    - Characters per second vs words per minute
  - Measurement of accuracy
    - Measurement of character-level error rates
    - Type of errors
  - Other performance issues (particularly as per eye gaze)
  - Modeling movement, attention, cognition
- Evaluation
  - Research questions and how to answer them
  - Experiment design issues
  - Conducting user studies
  - Gathering and analyzing data
  - Reporting results in research papers

Evaluation Slides | Skip

Evaluation

- Research questions
  - Typically, something like…
    - Is method A as fast/accurate as method B?
    - How much practice to reach, say, 15 wpm?
- Research questions come together in experiments as…
  - Independent variables, and
  - Dependent variables

Independent Variables

- These are the factors and levels in an experiment
- Examples

Factors	Levels
Input technique	Multitap, T9
Keyboard layout	Qwerty, Opti, Fitaly
Key size	small, medium, large
Type of feedback	visual, aural, both, neither
Session	1, 2, 3 ... 10
Word prediction	off, on
Gender	male, female

Dependent Variables

- These are the behaviours measured
- Examples

Variable	Units
Speed	Words per minute (wpm)
Accuracy	Percent errors (%)
Key activity	Keystrokes per character (kspc)
Backspace key events	Count or ratio
"Other" events	Count or ratio

Speed as a Dependent Variable

- Relatively straight forward to measure
- Example…

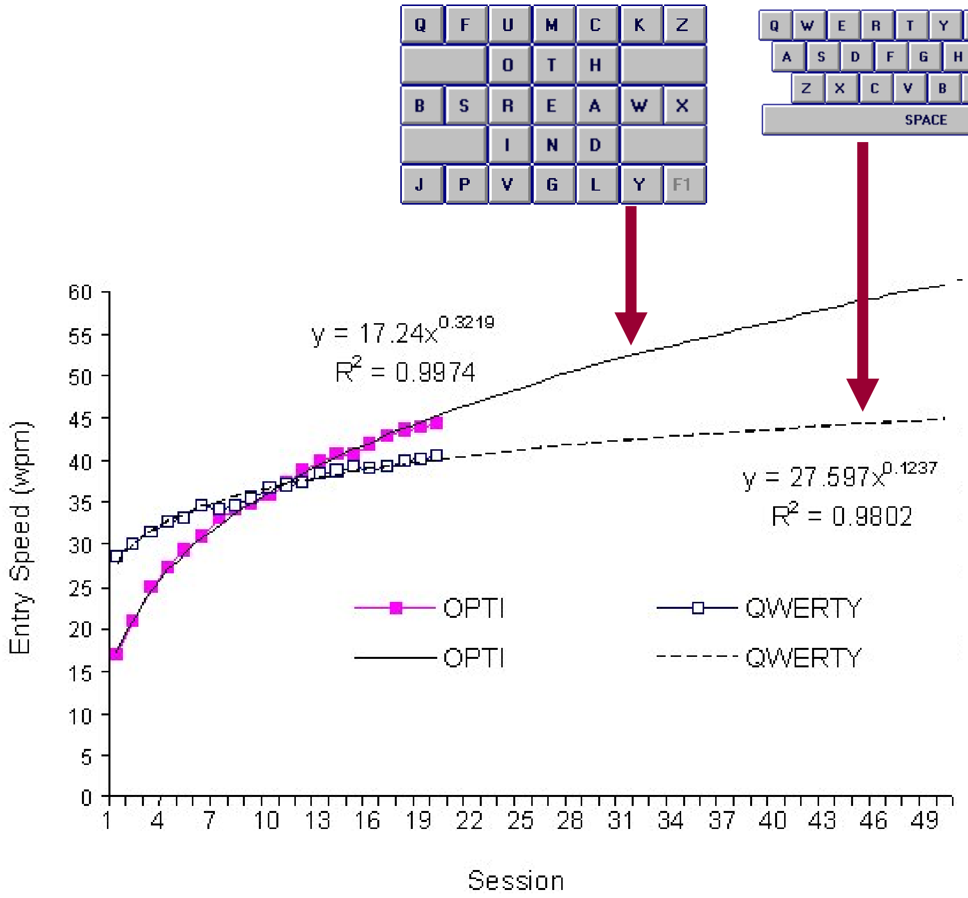
```
          1         2         3         4
1234567890123456789012345678901234567890123
the quick brown fox jumps over the lazy dog
```

t = 60 seconds = 1 minute
Number of characters = 43
Number of words = 43 / 5 = 8.6
Speed = 8.6 / 1 = 8.6 wpm

Note: Definition of a word: "five characters, including spaces, punctuation, etc"

Opti vs. Qwerty Example

- Two Independent variables
  - Keyboard layout with 2 levels: Opti, Qwerty
  - Session with 20 levels: 1, 2, 3, … 20
- Referred to as a 2 x 20 factorial design
- The 40 test conditions were given to all participants, thus we have a 2 x 20 within-subject design (i.e., each subject received all 40 test conditions)
- Note: within-subject design = repeated measures design (cf. between-subjects design)



Accuracy as a Dependent Variable

- A bit trickier
- Example…

Presented text

the quick brown fox
the quixck brwn fox

Transcribed text

1. How many errors?

   2 (gee, that was easy)

2. What are the errors?

   - An "x" was inserted
   - An "o" was omitted

3. What is the error rate (%)?

   ER = 2 / 19 = 0.105 = 10.5%

Minimum String Distance (MSD)

- In the example, there were two errors
- Relative easy for us to spot
- Hard to automate (i.e., use software)
- Can be done using an algorithm from DNA analysis to compute the minimum distance between two strings
- Consider…

computer
coxzter

MOBILEHCI 2008

		-1	0	1	2	3	4	5	6
			c	o	x	z	t	e	r
-1		0	1	2	3	4	5	6	7
0	c	1	0	1	2	3	4	5	6
1	o	2	1	0	1	2	3	4	5
2	m	3	2	1	1	2	3	4	5
3	p	4	3	2	2	2	3	4	5
4	u	5	4	3	3	3	3	4	5
5	t	6	5	4	4	4	3	4	5
6	e	7	6	5	5	5	4	3	4
7	r	8	7	6	6	6	5	4	3

Pseudo Code Algorithm[1]

```
function msd(A, B) {
  for i = 0 to |A|    // |A| = the length of A
     D[i, 0] = i
  for j = 0 to |B|
     D[0, j] = j
  for i = 1 to |A|
     for j = 1 to |B|
                           ( D[i-1, j] +1
        D[i, j] = min      ( D[i, j-1] +1
                           ( D[i-1, j-1] + r(A[i], B[j]) )
  return D[|A|, |B|]
}
function r(a, b) {
  if a = b return 0
  otherwise return 1
}
```

[1]Soukoreff, R. W., & MacKenzie, I. S. (2001). Measuring errors in text entry tasks: An application of the Levenshtein string distance statistic. *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems -- CHI 2001*, 319-320. New York: ACM.

MSD Properties

- Well-defined zero

  `msd(A, B) = 0, iff A = B`

- Bounded

  `0 ≤ msd(A, B) ≤ max(|A|, |B|) where`
  `|N| = length of string N`

- Commutative

  `msd(A, B) = msd(B, A)`

Text Entry Error Rate

$$\text{Error Rate} = \frac{\text{msd(A, B)}}{\text{max(|A|, |B|)}} \times 100\%$$

But, there is a problem… Let's see

Accuracy - Revisted

Presented text

quickly
qucehkly

Transcribed text

1. How many errors?

   3 (that was a bit tricky)

2. What are the errors?

   Hmm, let's see

3. What is the error rate?

   ER = 3 / 8.25 = 0.364 = 36.4%

```
quic--kly
qu-cehkly

quic-kly
qucehkly

qui-ckly
qucehkly

qu-ickly
qucehkly
```

Optimal Alignments

```
quic--kly
qu-cehkly

quic-kly
qucehkly

qui-ckly
qucehkly

qu-ickly
qucehkly
```

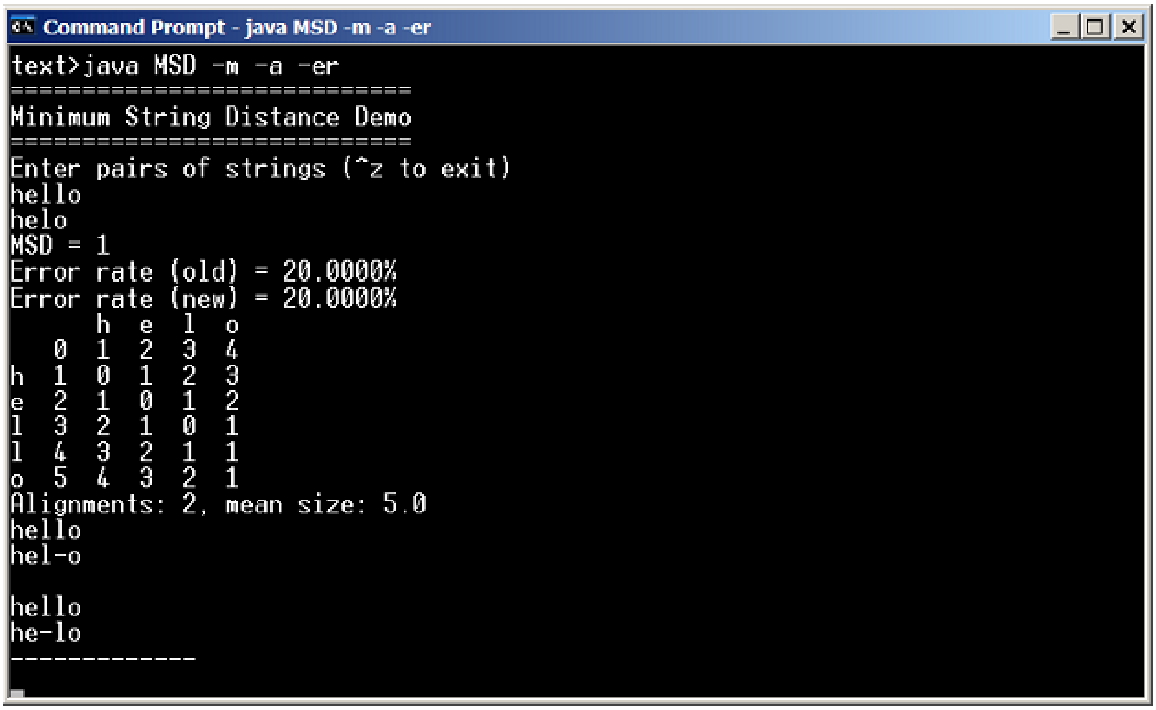
- The answer to the question "What are the errors?" shows the set of "optimal alignments" (the set of string pairings with the computed MSD)
- Properties
  1. The set size is often > 1
  2. The alignments are often of different lengths (in the example, 9, 8, 8, 8)
- Because of #2, the error formula needs to be tweaked…

Text Entry Error Rate (new)

$$\text{Error Rate} = \frac{\text{msd}(A, B)}{\overline{S}_A} \times 100\%$$

Where $\overline{S}_A$ is the mean size of the alignments (in the example, 8.25)

Demo



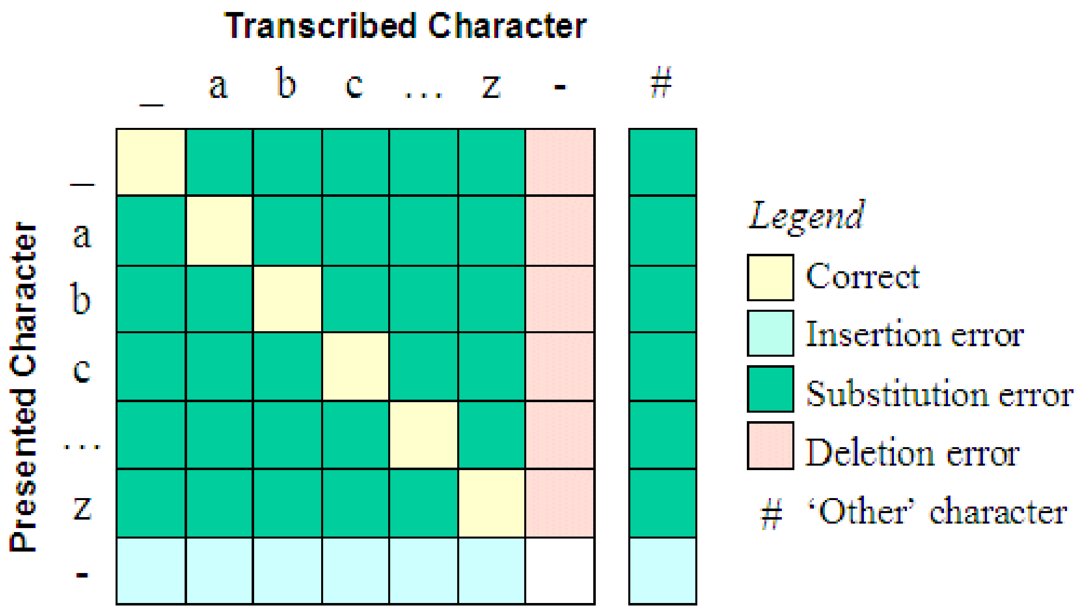
Optimal Alignments - revisited

- Properties (again)
  1. The set size is often > 1
  2. The alignments are often of different lengths (in the example, 9, 8, 8, 8)
- We just dealt with #2
- Implications of #1
  - We don't know what the "user" did
  - This is a problem if we wish to do character-level error analyses

Character-Level Error Analyses

- Reasonable compromise
  - Assume each error possibility occurs with equal probability
  - Instead of adding "1" for each error, add "1 / N", where N is the number of alignments, and repeat N times
  - Table view
    - For each character in the alphabet, tally the weighted occurrences of
      - Deletion errors
      - Substitution errors
      - Insertion errors
  - Matrix view (aka *confusion* matrix)
    - When a substitution error occurs, it is often important to know what was substituted (e.g., handwriting recognition)
    - Next slide...

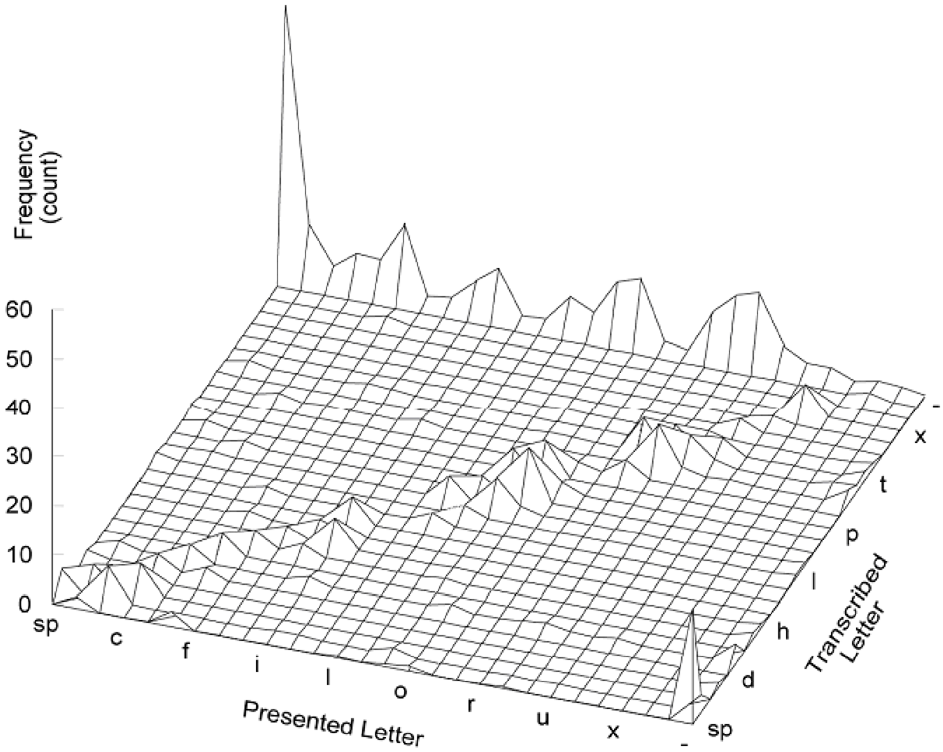
Confusion Matrix



Demo

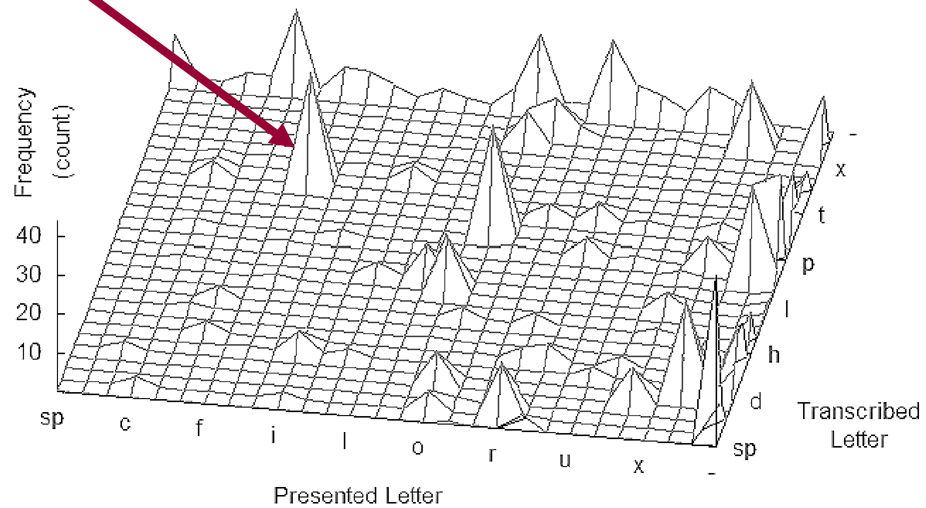
```
Command Prompt
text>java ErrorMatrix ds2-phrases.txt
usage: java ErrorMatrix file [-et] [-em] [-a] [-nd] [-pr] [-co]

where file = a file containing presented/transcribed strings
      -et  = output error table
      -em  = output error matrix
      -a   = output alignments (use for debugging/demo)
      -nd  = null diagonal cell entries in error matrix
      -pr  = use probabilities instead of counts in error matrix
      -co  = console output (looks better on display)
      (Note: default is no output)

text>
```


Three-Key Text Input Example



Handwriting Recognition Example

'g' often recognized as 'q'



Future Challenges

- We have compared "presented" with "transcribed" text
- What about the user's actions that produced the transcribed text?
- The correspondence is inherently one-for-one with a Qwerty keyboard, but this is typically not the case for mobile text entry
- Examples (next slide)

"lazy dog" – Multitap vs T9

Multitap:

lazy dog	← Presented text
55529999↓999036664	← Keystrokes
l az y do g	← Transcribed text

T9:

lazy dog	← Presented text
5299*0364	← Keystrokes
laz y dog	← Transcribed text

Bottom line:

Error analyses must consider the user's input stream, not just the final result.

Thank You

Referernces

1. Majaranta, P., MacKenzie, I. S., Aula, A., & Räihä, K.-J. Auditory and visual feedback during eye typing. *Extended Abstracts of CHI 2003*, New York: ACM, 2003, 766-767.
2. MacKenzie, I. S. KSPC (keystrokes per character) as a characteristic of text entry techniques, *Proceedings of the Fourth International Symposium on Human Computer Interaction with Mobile Devices*. Heidelberg: Springer Verlag, 2002, 195-210.
2. MacKenzie, I. S. Mobile text entry using three keys, *Proceedings of NordiCHI 2002*. New York: ACM, 2002, 27-34.
3. MacKenzie, I. S., Kober, H., Smith, D., Jones, T., and Skepner, E. LetterWise: Prefix-based disambiguation for mobile text input, *Proceedings of UIST 2001*. New York: ACM, 2001, 111-120.
4. MacKenzie, I. S., and Soukoreff, R. W. Character-level error analyses for evaluating text entry methods, *Proceedings of NordiCHI 2002*, New York: ACM, 2002, 241-244.
5. MacKenzie, I. S., and Soukoreff, R. W. Text entry for mobile computing: Models and methods, theory and practice, *Human-Computer Interaction* (2002), 17, 147-198.
6. MacKenzie, I. S., and Zhang, S. X. The design and evaluation of a high-performance soft keyboard, *Proceedings of CHI '99*. New York: ACM, 1999, 25-31.
7. MacKenzie, I. S., and Zhang, S. X. An empirical investigation of the novice experience with soft keyboards, *Behaviour & Information Technology 20* (2001), 411-418.
8. MacKenzie, I. S., Zhang, S. X., and Soukoreff, R. W. Text entry using soft keyboards, *Behaviour & Information Technology 18* (1999), 235-244.