Developing and Tuning a Voice Spelling Alphabet for Devices with Small Displays

TR 29.3517
May 15, 2002

James R. Lewis
Patrick M. Commarford

IBM Voice Systems

Boca Raton, Florida

Abstract

This report documents the motivation, method and results of three experiments conducted to develop and tune speech recognition grammars for voice spelling on devices with small displays. The final grammar did not include the names of the letters of the alphabet themselves. The words used to represent letters of the alphabet were small enough to fit on a small display so users would not have to memorize the words used for voice spelling. These words had very high recognition accuracies and high conditional probabilities, giving them the potential for a very effective voice spelling system.

ITIRC Keywords

Recognition accuracy
Recognition errors
Embedded speech recognition
Voice spelling grammar

Contents

Introduction

One of the better-known usability problems associated with devices that have small displays is the input problem (Lewis, Potosnak, & Magyar, 1997). The methods most often used to input data directly into small devices are small keyboards (using physical or virtual keys) and limited handwriting recognition (such as Graffiti or Unistrokes). Speech dictation is a promising alternative, but to date there are no small devices that have sufficient power to allow the use of this technology. The use of voice spelling as an alternative input mechanism is, on the other hand, feasible with current technologies. Once small systems do have sufficient power for limited dictation, voice spelling will still play an important role for the hands-free correction of misrecognitions and inputting words that are out of the dictation system's vocabulary (OOV).

This report describes four experiments conducted for the purpose of developing and tuning a set of words similar to the military alphabet for use in handheld voice spelling. The standard military alphabet (see Appendix A) is not suitable for use in handheld voice spelling for two reasons.

First, some of the words are too long to permit their simultaneous display on a small screen. This is important because one of the difficulties in using a military-style spelling alphabet is remembering the words that represent the letters of the alphabet. We used data from Lewis (2001) to select shorter alternatives for the longer words of the standard military alphabet (see Appendix A).

Second, a comprehensive voice-spelling system requires the recognition of digits, punctuation commands, and commands that control the location of the insertion point. It is possible that words in the standard military alphabet might be acoustically confusable with these commands.

General Method

Participants

The participants were four males and four females (all adult native speakers of American English), all of whom had provided recordings of a test script (using SoundForge[1], 8 kHz, 16-bit mono PCM).

Materials and Procedure

The computer used to record speech and process the recordings had 256 megabytes of memory and several gigabytes of free hard disk space, running Windows[2] 2000. Participants recorded the test script by speaking into an Andrea NC-61. After recording, five seconds of silence were inserted between each token in each recording of the script. We used the IBM WebSphere[3] Voice Server SDK Version 2.0 to decode the recorded scripts.

The recorded test script included the letters of the alphabet, the customized military-style spelling alphabet, digits, punctuation and commands for controlling the location of the insertion point (see Appendix B). The dependent measure in these experiments was the simple command accuracy for each token in the test script, organized by token type.

[1] SoundForge is a trademark of The Sonic Foundry, Inc.
[2] Windows is a registered trademark of Microsoft Corp.
[3] IBM and WebSphere are registered trademarks of International Business Machines Corp.

Experiment 1: The Full Grammar

Motivation

The purpose of the first experiment was to investigate the accuracy of a voice-spelling grammar that included the letters of the alphabet, the first version of a customized military-style spelling alphabet, digits, punctuation and commands for controlling the location of the insertion point.

Method

The method for this experiment was the same as that described in the General Method section. The grammar used in this experiment appears in Appendix C.

Results and Discussion

Accuracy. Table 1 shows the results from this experiment, including the mean accuracy and 95% confidence interval for the mean. The accuracy of the military-style alphabet was significantly (10.1 points) higher than that of the letters of the alphabet ($t(7) = 3.86$, $p = .01$). The accuracy of the other tokens (digits and punctuation/commands) was comparable to the accuracy of the military-style alphabet.

Table 1. Experiment 1: Accuracies

	Accuracy	95% Confidence
Overall Accuracy	91.4	3.5
Accuracy (w/o Letters)	94.3	3.8
Letters	83.2	6.2
Military-Style Alphabet	93.3	4.8
Digits	93.1	6.9
Punctuation/Commands	95.1	3.4

Conditional probabilities. The recognition accuracy for a token is the likelihood of the system correctly recognizing a token when a user speaks it. The conditional probability is similar, but is the likelihood that the token produced by the recognizer is the one that the user actually spoke. Sometimes the accuracy and conditional probability for a token can be very different. For example, in Experiment 1 the accuracy for the letter 'F' was only 25% because the system frequently misrecognized it as 'S'. Its conditional probability, however, was 100% because whenever the system returned an 'F', the spoken letter was 'F'.

The conditional probabilities for tokens that the system did not recognize correctly can be especially useful for designing intelligent correction schemes for voice-spelling systems (Hartley & Lewis, 2001). The following confusion matrix (Table 2) shows the conditional probabilities from this experiment for letters of the alphabet. In all cases but one, the conditional probability for the tokens in the military-style alphabet was 100%. The exception was the word 'jane', for which the system incorrectly produced 'A' and 'G', each with a conditional probability of 10%.

Table 2. Experiment 1: Conditional Probabilities for Letters of the Alphabet

	System Returned:																								
Said	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
A	100																									
B		85.7		20.0																						
C			100																							
D				80.0																						
E					88.9																					
F						100													44.4							
G							85.7			10.0																
H								100																		
I									85.7																	
J										80.0																
K										10.0	100															
L												88.9														
M													100	20.0												
N														80.0												
O															100											
P					11.1											100				12.5						
Q																	100									
R																		100								
S																			55.6					12.5		
T							14.3													87.5						
U																					100					
V		14.3																				100				38.5
W																							100			
X																								87.5		
Y																									88.9	
Z																										61.5
0																										
1																										
2																										
3																										
4																										
5																										
6																										
7																										
8																										
9									14.3																	
alpha												11.1														
bravo																										
cat																										
delta																										
echo																										
frank																										
golf																										
hotel																										
india																										
jane																										
kilo																										
lima																										
mike																									11.1	

Table 2. Experiment 1: Conditional Probabilities for Letters of the Alphabet (cont.)

Said	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
north																										
oscar																										
papa																										
queen																										
romeo																										
sierra																										
tango																										
under																										
victor																										
water																										
x ray																										
yoyo																										
zulu																										
Total	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100

For half of the letters, the observed likelihood that the letter returned by the recognizer was the spoken letter was 100% (A, C, F, H, K, M, O, P, Q, R, U, V, and W). For the other letters, the misrecognitions were systematic rather than random, and were relatively rare. The most frequent substitutions were the letter 'S' for 'F' and 'Z' for 'V' (44.4% and 38.5% respectively).

Specific recognition problems. Table 3 shows the most frequently occurring recognition problems in Experiment 1 (token accuracies at 75% or lower – the number in parentheses indicates the frequency of the misrecognition). A '?' indicates low confidence and 'noReturn' indicates that the recognizer didn't return anything for the token. Of the 17 problem tokens, 10 were letters. For most of the remaining tokens, the majority of misrecognitions involved letters. This indicates that overall recognition accuracy might improve as a result of removing letters from the grammar (see Experiment 2). The other problem tokens were 'papa', 'period', and 'dot'. 'Papa' and 'dot' had fairly high rates of noReturn. 'Period' had fairly high rates of low confidence responses.

Table 3. Experiment 1: Frequent Misrecognitions

Token	Accuracy	Misrecognized as
f	**25.0**	s (4), ?u (1), noReturn (1)
v	**25.0**	z (5), b (1)
papa	**37.5**	NoReturn (4), plus (1)
period	**37.5**	?period (3), ?delete (1), juliet (1)
o	**50.0**	Zero (4)
s	**62.5**	dash (1), at (1), x (1)
dot	**62.5**	NoReturn (2), ?a
b	**75.0**	D (2)
g	**75.0**	j (1), jane (1)
i	**75.0**	five (1), eight (1)
m	**75.0**	N (2)
p	**75.0**	e (1), t(1)
q	**75.0**	Two (2)
alpha	**75.0**	L (2)
delta	**75.0**	L (1), delta o (1)
five	**75.0**	seven (1), five o (1)
nine	**75.0**	I (1), minus (1)

Experiment 2: The Effect of Removing the Letters of the Alphabet

Motivation

The results of the first experiment provided a baseline for evaluating the relative accuracy of a grammar that did not include the letters of the alphabet. The purpose of Experiment 2 was to investigate the accuracy of a grammar that was the same as that used in Experiment 1 except for the absence of the letters of the alphabet. Recognition accuracy of the letters of the alphabet is notoriously poor (as seen in Experiment 1), but unless removing these tokens from the grammar resulted in improved recognition for other tokens, then the best strategy would be to leave them available for use. On the other hand, if removing them led to improved recognition for other tokens, then the best strategy would be to remove the letters of the alphabet from the grammar. To compensate for the difficulty of remembering the words in the military-style alphabet, developers could design a graphical user interface for voice-spelling that displayed the words in alphabetical order (which was the reason for choosing short words for the new 'alphabet').

Method

The method for this experiment was the same as that described in the General Method section, with the exception of using a grammar that did not include the letters of the alphabet and removing the spoken letters of the alphabet from each participants' wave files (see Appendix D).

Results and Discussion

Accuracy. Table 4 shows the results from Experiment 2, including the mean accuracy and 95% confidence interval for the mean. Note that the Overall Accuracy for Experiment 2 (and the following experiments) corresponds to the Accuracy w/o Letters in Experiment 1. The Overall Accuracy in Experiment 2 was significantly better (using $\alpha = .10$) than the Accuracy w/o Letters in Experiment 1 ($t(7) = 2.22$, $p = .06$). The accuracy of the tokens in the military-style alphabet was also significantly better ($t(7) = 2.02$, $p = .08$).

Table 4. Experiment 2: Accuracies

	Accuracy	95% Confidence
Overall Accuracy	95.8	2.9
Military-Style Alphabet	96.2	3.4
Digits	96.3	4.3
Punctuation/Commands	95.4	3.1

Specific recognition problems. Table 5 shows the most frequently occurring recognition problems in Experiment 2 (token accuracies at 75% or lower – the number in parentheses indicates the frequency of the misrecognition). 'Period', 'dot', and 'papa' continued to have problems.

Table 5. Experiment 2: Frequent Misrecognitions

Phrase	Accuracy	Misrecognized as
period	37.5	?period (3), ?delete (1), juliet (1)
dot	62.5	noReturn (1), ?dot (1), ?pound (1)
papa	75.0	noReturn (2)
five	75.0	seven (1), ?seven (1)
plus	75.0	noReturn (1), slash (1)

Experiment 3: Tuning the Grammar – Phase 1

Motivation

Removing the letters of the alphabet from the grammar resulted in significant improvements to recognition accuracy, but some tokens continued to have relatively low accuracy ('period', 'dot', 'papa', 'five', and 'plus'). The purpose of Experiment 3 was to retest the grammar after tuning it (attempting to correct the remaining problems).

Method

The method was the same as that of the second experiment, with the grammar modified in the following ways (see Appendix E):

- Removed 'juliet' from grammar due to acoustic confusability with 'period'.
- Provided additional support for recognition of 'dot' in context of web url and e-mail addresses (see <specialdot> node in Appendix E).
- Added 'paul' to grammar as alternative for 'papa' in potential graphical user interface.

Previous research in the words that people associate most frequently with the letters of the alphabet guided the decision to add 'paul' to the grammar (Lewis, 2001). Note that the addition of the <specialdot> node to the grammar would not improve recognition accuracy of 'dot' in isolation, but should improve its recognition in its typical contexts of use.

Results and Discussion

Accuracy. Table 6 shows the results from Experiment 3, including the mean accuracy and 95% confidence interval for the mean. The overall accuracy for Experiment 3 improved by one point relative to Experiment 2, but this change was not statistically significant ($t(7) = 1.71$, $p = .13$). Compared to the results of Experiment 1, the difference was statistically significant (($t(7) = 2.94$, $p = .02$).

Table 6. Experiment 3: Accuracies

	Accuracy	95% Confidence
Overall Accuracy	96.5	2.6
Military-Style Alphabet	97.1	3.7
Digits	97.5	3.9
Punctuation/Commands	95.7	2.9

Specific recognition problems. Table 7 shows the most frequently occurring recognition problems in Experiment 2 (token accuracies at 75% or lower – the number in parentheses indicates the frequency of the misrecognition). The change to the grammar fixed the 'papa' problem, but 'period' and, to a lesser extent, 'dot' continued to have problems. 'Delta' produced the always-active command 'help' in two instances, indicating that the spelling grammar should include an alternate word for 'D'.

Table 7. Experiment 3: Frequent Misrecognitions

Phrase	Accuracy	Misrecognized as
period	37.5	?period (4), delete
delta	75	help
dot	75	noReturn (2)

Experiment 4: Tuning the Grammar – Phase 2

Motivation

In the final tuning step, we added pronunciations for the words that had continuing recognition problems, and added a word to address the acoustic confusability of 'delta' and 'help'.

Method

The method was identical to that of Experiment 3, with the following exceptions:

- Provided a custom pronunciation for 'period'.
- Provided a custom pronunciation for 'dot'.
- Added 'dog' to the grammar as an alternative for 'delta'.

See Appendix F for the final version of the grammar and the VoiceXML code for the additional pronunciations.

Results and Discussion

Accuracy. Table 8 shows the results from Experiment 4, including the mean accuracy and 95% confidence interval for the mean.

The overall accuracy for Experiment 4 improved by 0.7 point relative to Experiment 2, but this change was not statistically significant ($t(7) = 1.0$, $p = .35$). Compared to the results of Experiment 1, the difference in overall accuracy was statistically significant (($t(7) = 2.97$, $p = .02$). Compared to Experiment 1, the increase in accuracy for the military-style alphabet was statistically significant ($t(7) = 2.35$, $p = .05$).

Table 8. Experiment 4: Accuracies

	Accuracy	95% Confidence
Overall Accuracy	97.5	1.7
Military-Style Alphabet	97.6	4.5
Digits	97.5	3.9
Punctuation/Commands	97.4	1.6

Specific recognition problems. Table 7 shows the most frequently occurring recognition problems in Experiment 2 (token accuracies at 75% or lower – the number in parentheses indicates the frequency of the misrecognition). The changes to the grammar fixed the 'period' and 'delta' problems, and reduced, to some extent, the 'dot' problem.

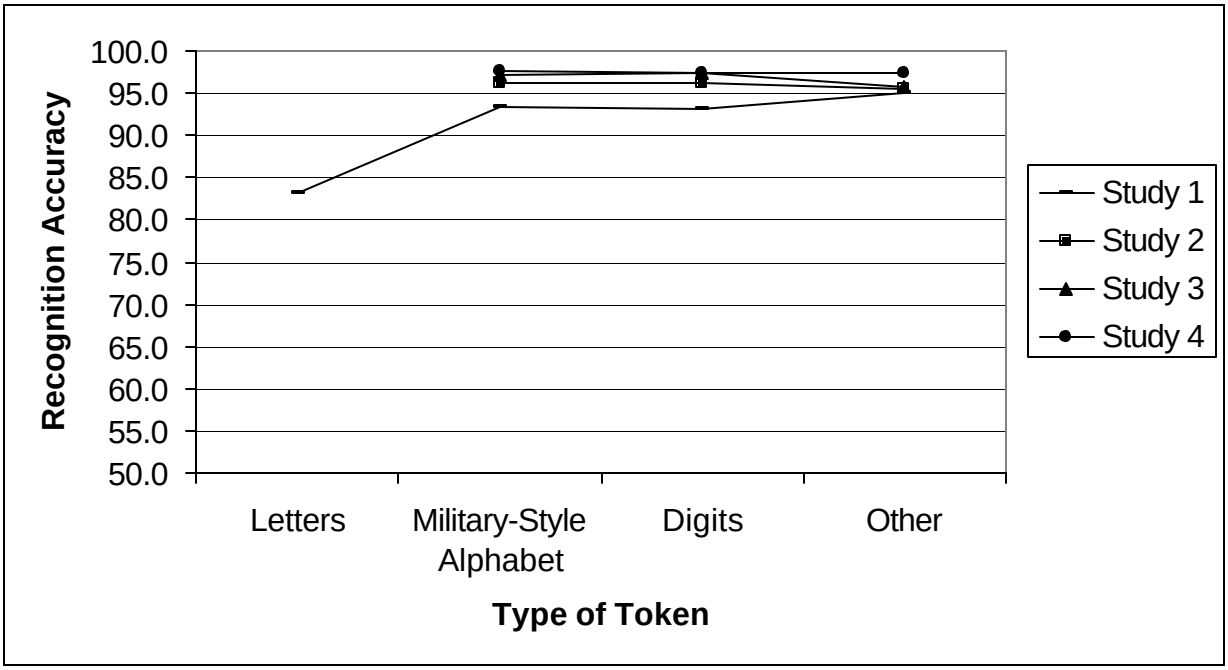
Table 9. Experiment 4: Frequent Misrecognitions

Phrase	Accuracy	Misrecognized as
Dot	75	noReturn, ?dot

General Results and Recommendations

Figure 1 shows the pattern of results across the four experiments.

Figure 1: Summary of Results



Recommendation 1: If the device has sufficient screen area to display the new military-style alphabet during voice spelling, then display this alphabet and do not include letters in the voice spelling grammar.

The bases for this recommendation are (1) the relatively low recognition accuracy for letters seen in Experiment 1 and (2) the improvement seen in recognition accuracy for other classes of tokens after removing letters from the grammar, as seen in Experiments 2, 3, and 4.

Figure 2 shows one way to arrange the new military-style alphabet for presentation on a small-screen device:

Figure 2. New Military-Style Voice Spelling Alphabet

alpha	frank	kilo	paul	under
bravo	golf	lima	queen	victor
cat	hotel	mike	romeo	water
dog	india	north	sierra	x-ray
echo	jane	oscar	tango	yoyo
space				zulu

Recommendation 2: If the device does not have sufficient screen area to display the new military-style alphabet during voice spelling, then make it a user option to include or exclude letters from the grammar.

The basis for this recommendation is the known difficulty of remembering the words in military-style alphabets. Even with reduced recognition accuracy, users who do not know the military-style alphabet will need letters when they begin voice spelling. Users who have memorized the new or standard military-style can exclude letters to obtain the advantage of improved voice spelling accuracy.

References

Hartley, M. W., & Lewis, J. R. (2001). *Conditional probabilities for IBM Voice Browser recognition of letters of the alphabet* (Tech. Report 29.3241). West Palm Beach, FL: International Business Machines Corp.

Lewis, J. R. (2001). *Frequency distributions for names and unconstrained words associated with the letters of the English alphabet* (Tech. Report 29.3437). West Palm Beach, FL: International Business Machines Corp.

Lewis, J. R., Potosnak, K., & Magyar, R. L. (1997). Keys and keyboards. In *The Handbook of Human-Computer Interaction 2nd Ed.* (pp. 1285-1315). Amsterdam: North-Holland.

Appendix A. The Standard and New Military Alphabets

The shorter alternatives used in this series of experiments appear in parentheses.

Alpha
Bravo
Charlie (cat)
Delta (dog)
Echo

Foxtrot (frank)
Golf
Hotel
India
Juliet (jane)

Kilo
Lima
Mike
November (north)
Oscar

Papa (paul)
Quebec
Romeo
Sierra
Tango

Uniform (under)
Victor
Whiskey (water)
X-ray
Yankee (yoyo)
Zulu

Appendix B. The Test Script

This appendix contains the test script used in the experiments, organized by token type.

Letters of the Alphabet

A
B
C
D
E

F
G
H
I
J

K
L
M
N
O

P
Q
R
S
T

U
V
W
X
Y
Z

Military Alphabet (Modified)

Alpha
Bravo
Cat
Delta
Echo

Frank
Golf
Hotel
India
Jane

Kilo
Lima
Mike
North
Oscar

Papa
Queen
Romeo
Sierra
Tango

Under
Victor
Water
X-ray
Yoyo
Zulu

Digits

Zero
One
Two
Three
Four

Five
Six
Seven
Eight
Nine

Punctuation and Insertion Point Commands

Capital
Caps Lock
Lowercase
Tab
Back tab

Enter
Space
Back space
Period
Dot

Comma
Question mark
Exclamation point
At
Dash

Slash
Backslash
Colon
Semicolon
Apostrophe

Quote
Pound
Percent
Ampersand
Asterisk

Open Paren
Close Paren
Greater than
Less than
Plus

Equals
Delete
Move right one character
Move left ten words
Move up three lines
Move down nine lines

Appendix C. Grammar for Experiment 1

```
#JSGF V1.0;

grammar voicespell;

// Copyright (c) 2002 IBM Corp.  All Rights Reserved.

<a> = a | alpha;
<b> = b | bravo;
<c> = c | charlie | cat;
<d> = d | delta;
<e> = e | echo;
<f> = f | foxtrot | frank;
<g> = g | golf;
<h> = h | hotel;
<i> = i | india;
<j> = j | juliet | jane;
<k> = k | kilo;
<l> = l | lima;
<m> = m | mike;
<n> = n | november | north;
<o> = o | oscar;
<p> = p | papa;
<q> = q | quebec | queen;
<r> = r |romeo;
<s> = s |sierra;
<t> = t |tango;
<u> = u |uniform | under;
<v> = v |victor;
<w> = w |water | whiskey;
<x> = x |x ray;
<y> = y |yankee | yoyo;
<z> = z |zulu;

<upper> = capital | uppercase;
<capslock> = caps lock | capital | uppercase | lowercase;
<lower> = lowercase;

<capping> = <upper>
          | <lower>
          | <capslock>
          ;
```

<lowerletter> = [<lower>] <a>
|[<lower>] <b>
|[<lower>] <c>
|[<lower>] <d>
|[<lower>] <e>
|[<lower>] <f>
|[<lower>] <g>
|[<lower>] <h>
|[<lower>] <i>
|[<lower>] <j>
|[<lower>] <k>
|[<lower>] <l>
|[<lower>] <m>
|[<lower>] <n>
|[<lower>] <o>
|[<lower>] <p>
|[<lower>] <q>
|[<lower>] <r>
|[<lower>] <s>
|[<lower>] <t>
|[<lower>] <u>
|[<lower>] <v>
|[<lower>] <w>
|[<lower>] <x>
|[<lower>] <y>
|[<lower>] <z>
;

<upperletter> = <upper> <a>
|<upper> <b>
|<upper> <c>
|<upper> <d>
|<upper> <e>
|<upper> <f>
|<upper> <g>
|<upper> <h>
|<upper> <i>
|<upper> <j>
|<upper> <k>
|<upper> <l>
|<upper> <m>
|<upper> <n>
|<upper> <o>

```
        |<upper> <p>
        |<upper> <q>
        |<upper> <r>
        |<upper> <s>
        |<upper> <t>
        |<upper> <u>
        |<upper> <v>
        |<upper> <w>
        |<upper> <x>
        |<upper> <y>
        |<upper> <z>
        ;

<digit> = zero
        | one
        | two
        | three
        | four
        | five
        | six
        | seven
        | eight
        | nine
        ;

<num2-10> = two
        | three
        | four
        | five
        | six
        | seven
        | eight
        | nine
        | ten
        ;

<punctuation> = tab
        | back tab
        | enter
        | space
        | back space
        | period
        | dot
```

```
        | comma
        | question mark
        | exclamation point
        | at [sign]
        | dash | hyphen
        | slash
        | back slash
        | colon
        | semicolon
        | apostrophe
        | quote
        | pound
        | percent
        | ampersand
        | asterisk
        | open paren
        | close paren
        | greater than
        | less than
        | plus
        | minus
        | equals
        | delete
        ;

<word> = (<lowerletter> | <upperletter> | <punctuation> | <digit> | <capping>)*;

<move> = move (right | left) <num2-10> (characters | words)
        | move (right | left) (one | a) (character | word)
        | move (up | down) <num2-10> lines
        | move (up | down) (one | a) line
        | (next | previous) field
        ;

public <spell> = <word>
        | <move>
        ;
```

Appendix D. Grammar for Experiment 2

```
#JSGF V1.0;

grammar voicespell2;

// Copyright (c) 2002 IBM Corp.  All Rights Reserved.

<a> = alpha;
<b> = bravo;
<c> = charlie | cat;
<d> = delta;
<e> = echo;
<f> = foxtrot | frank;
<g> = golf;
<h> = hotel;
<i> = india;
<j> = juliet | jane;
<k> = kilo;
<l> = lima;
<m> = mike;
<n> = november | north;
<o> = oscar;
<p> = papa;
<q> = quebec | queen;
<r> = romeo;
<s> = sierra;
<t> = tango;
<u> = uniform | under;
<v> = victor;
<w> = water | whiskey;
<x> = x ray;
<y> = yankee | yoyo;
<z> = zulu;

<upper> = capital | uppercase;
<capslock> = caps lock | capital | uppercase | lowercase;
<lower> = lowercase;

<capping> = <upper>
          | <lower>
          | <capslock>
          ;
```

```
<lowerletter> = [<lower>] <a>
        |[<lower>] <b>
        |[<lower>] <c>
        |[<lower>] <d>
        |[<lower>] <e>
        |[<lower>] <f>
        |[<lower>] <g>
        |[<lower>] <h>
        |[<lower>] <i>
        |[<lower>] <j>
        |[<lower>] <k>
        |[<lower>] <l>
        |[<lower>] <m>
        |[<lower>] <n>
        |[<lower>] <o>
        |[<lower>] <p>
        |[<lower>] <q>
        |[<lower>] <r>
        |[<lower>] <s>
        |[<lower>] <t>
        |[<lower>] <u>
        |[<lower>] <v>
        |[<lower>] <w>
        |[<lower>] <x>
        |[<lower>] <y>
        |[<lower>] <z>
        ;

<upperletter> = <upper> <a>
        |<upper> <b>
        |<upper> <c>
        |<upper> <d>
        |<upper> <e>
        |<upper> <f>
        |<upper> <g>
        |<upper> <h>
        |<upper> <i>
        |<upper> <j>
        |<upper> <k>
        |<upper> <l>
        |<upper> <m>
        |<upper> <n>
        |<upper> <o>
```

```
        |<upper> <p>
        |<upper> <q>
        |<upper> <r>
        |<upper> <s>
        |<upper> <t>
        |<upper> <u>
        |<upper> <v>
        |<upper> <w>
        |<upper> <x>
        |<upper> <y>
        |<upper> <z>
        ;

<digit> = zero
        | one
        | two
        | three
        | four
        | five
        | six
        | seven
        | eight
        | nine
        ;

<num2-10> = two
        | three
        | four
        | five
        | six
        | seven
        | eight
        | nine
        | ten
        ;

<punctuation> = tab
        | back tab
        | enter
        | space
        | back space
        | period
        | dot
```

```
        | comma
        | question mark
        | exclamation point
        | at [sign]
        | dash | hyphen
        | slash
        | back slash
        | colon
        | semicolon
        | apostrophe
        | quote
        | pound
        | percent
        | ampersand
        | asterisk
        | open paren
        | close paren
        | greater than
        | less than
        | plus
        | minus
        | equals
        | delete
        ;

<word> = (<lowerletter> | <upperletter> | <punctuation> | <digit> | <capping>)*;

<move> = move (right | left) <num2-10> (characters | words)
        | move (right | left) (one | a) (character | word)
        | move (up | down) <num2-10> lines
        | move (up | down) (one | a) line
        | (next | previous) field
        ;

public <spell> = <word>
        | <move>
        ;
```

Appendix E. Grammar for Experiment 3

```
#JSGF V1.0;

grammar voicespell3;

// Copyright (c) 2002 IBM Corp.  All Rights Reserved.

<a> = alpha;
<b> = bravo;
<c> = charlie | cat;
<d> = delta;
<e> = echo;
<f> = foxtrot | frank;
<g> = golf;
<h> = hotel;
<i> = india;
<j> = jane;
<k> = kilo;
<l> = lima;
<m> = mike;
<n> = november | north;
<o> = oscar;
<p> = paul | papa;
<q> = quebec | queen;
<r> = romeo;
<s> = sierra;
<t> = tango;
<u> = uniform | under;
<v> = victor;
<w> = water | whiskey;
<x> = x ray;
<y> = yankee | yoyo;
<z> = zulu;

<upper> = capital | uppercase;
<capslock> = caps lock;
<lower> = lowercase;

<capping> = <upper>
          | <lower>
          | <capslock>
          ;
```

```
<lowerletter> = [<lower>] <a>
        |[<lower>] <b>
        |[<lower>] <c>
        |[<lower>] <d>
        |[<lower>] <e>
        |[<lower>] <f>
        |[<lower>] <g>
        |[<lower>] <h>
        |[<lower>] <i>
        |[<lower>] <j>
        |[<lower>] <k>
        |[<lower>] <l>
        |[<lower>] <m>
        |[<lower>] <n>
        |[<lower>] <o>
        |[<lower>] <p>
        |[<lower>] <q>
        |[<lower>] <r>
        |[<lower>] <s>
        |[<lower>] <t>
        |[<lower>] <u>
        |[<lower>] <v>
        |[<lower>] <w>
        |[<lower>] <x>
        |[<lower>] <y>
        |[<lower>] <z>
        ;

<upperletter> = <upper> <a>
        |<upper> <b>
        |<upper> <c>
        |<upper> <d>
        |<upper> <e>
        |<upper> <f>
        |<upper> <g>
        |<upper> <h>
        |<upper> <i>
        |<upper> <j>
        |<upper> <k>
        |<upper> <l>
        |<upper> <m>
        |<upper> <n>
        |<upper> <o>
```

```
        |<upper> <p>
        |<upper> <q>
        |<upper> <r>
        |<upper> <s>
        |<upper> <t>
        |<upper> <u>
        |<upper> <v>
        |<upper> <w>
        |<upper> <x>
        |<upper> <y>
        |<upper> <z>
        ;

<digit> = zero
        | one
        | two
        | three
        | four
        | five
        | six
        | seven
        | eight
        | nine
        ;

<num2-10> = two
        | three
        | four
        | five
        | six
        | seven
        | eight
        | nine
        | ten
        ;

<punctuation> = tab
        | back tab
        | enter
        | space
        | back space
        | period
        | dot
```

```
        | <specialdot>
        | comma
        | question mark
        | exclamation point
        | at [sign]
        | dash | hyphen
        | slash
        | back slash
        | colon
        | semicolon
        | apostrophe
        | quote
        | pound
        | percent
        | ampersand
        | asterisk
        | open paren
        | close paren
        | greater than
        | less than
        | plus
        | minus
        | equals
        | delete
        ;

<specialdot> = dot
        | (water dot)
        | (dot (com | edu | net | org | cat | charlie | oscar | echo | north))
        ;

<word> = (<lowerletter> | <upperletter> | <punctuation> | <digit> | <capping>)*;

<move> = move (right | left) <num2-10> (characters | words)
        | move (right | left) (one | a) (character | word)
        | move (up | down) <num2-10> lines
        | move (up | down) (one | a) line
        | (next | previous) field
        ;

public <spell> = <word>
        | <move>
        ;
```

Appendix F. Modifications for Experiment 4

Grammar

```
#JSGF V1.0;

grammar voicespell4;

// Copyright (c) 2002 IBM Corp.  All Rights Reserved.

<a> = alpha;
<b> = bravo;
<c> = charlie | cat;
<d> = delta | dog;
<e> = echo;
<f> = foxtrot | frank;
<g> = golf;
<h> = hotel;
<i> = india;
<j> = jane;
<k> = kilo;
<l> = lima;
<m> = mike;
<n> = november | north;
<o> = oscar;
<p> = paul | papa;
<q> = quebec | queen;
<r> = romeo;
<s> = sierra;
<t> = tango;
<u> = uniform | under;
<v> = victor;
<w> = water | whiskey;
<x> = x ray;
<y> = yankee | yoyo;
<z> = zulu;

<upper> = capital | uppercase;
<capslock> = caps lock;
<lower> = lowercase;

<capping> = <upper>
          | <lower>
          | <capslock>
          ;
```

```
<lowerletter> = [<lower>] <a>
        |[<lower>] <b>
        |[<lower>] <c>
        |[<lower>] <d>
        |[<lower>] <e>
        |[<lower>] <f>
        |[<lower>] <g>
        |[<lower>] <h>
        |[<lower>] <i>
        |[<lower>] <j>
        |[<lower>] <k>
        |[<lower>] <l>
        |[<lower>] <m>
        |[<lower>] <n>
        |[<lower>] <o>
        |[<lower>] <p>
        |[<lower>] <q>
        |[<lower>] <r>
        |[<lower>] <s>
        |[<lower>] <t>
        |[<lower>] <u>
        |[<lower>] <v>
        |[<lower>] <w>
        |[<lower>] <x>
        |[<lower>] <y>
        |[<lower>] <z>
        ;

<upperletter> = <upper> <a>
        |<upper> <b>
        |<upper> <c>
        |<upper> <d>
        |<upper> <e>
        |<upper> <f>
        |<upper> <g>
        |<upper> <h>
        |<upper> <i>
        |<upper> <j>
        |<upper> <k>
        |<upper> <l>
        |<upper> <m>
        |<upper> <n>
        |<upper> <o>
```

```
        |<upper> <p>
        |<upper> <q>
        |<upper> <r>
        |<upper> <s>
        |<upper> <t>
        |<upper> <u>
        |<upper> <v>
        |<upper> <w>
        |<upper> <x>
        |<upper> <y>
        |<upper> <z>
        ;
<digit> = zero
        | one
        | two
        | three
        | four
        | five
        | six
        | seven
        | eight
        | nine
        ;
<num2-10> = two
        | three
        | four
        | five
        | six
        | seven
        | eight
        | nine
        | ten
        ;
<punctuation> = tab
        | back tab
        | enter
        | space
        | back space
        | period
        | dot
        | <specialdot>
```

```
        | comma
        | question mark
        | exclamation point
        | at [sign]
        | dash | hyphen
        | slash
        | back slash
        | colon
        | semicolon
        | apostrophe
        | quote
        | pound
        | percent
        | ampersand
        | asterisk
        | open paren
        | close paren
        | greater than
        | less than
        | plus
        | minus
        | equals
        | delete
        ;

<specialdot> = dot
        | (water dot)
        | (dot (com | edu | net | org | cat | charlie | oscar | echo | north))
        ;

<word> = (<lowerletter> | <upperletter> | <punctuation> | <digit> | <capping>)*;

<move> = move (right | left) <num2-10> (characters | words)
        | move (right | left) (one | a) (character | word)
        | move (up | down) <num2-10> lines
        | move (up | down) (one | a) line
        | (next | previous) field
        ;

public <spell> = <word>
        | <move>
        ;
```

Punctuations

```
<ibmlexicon>
        <word spelling="period" pronunciation="piri&#618;d "/>
        <word spelling="period" pronunciation="&#712;pir.i.&#616;d"/>
        <word spelling="period" pronunciation="piri&#652;d "/>
        <word spelling="dot" pronunciation="d&#593;t"/>
        <word spelling="dot" pronunciation="d&#596;t"/>
</ibmlexicon>
```

Note: You can use the IBM VoiceXML Toolkit to review the sound of these pronunciations or to see them represented in IPA symbols.