THE DELTA SYSTEM WITH SYLLT: INCREASED CAPABILITIES FOR TEACHING AND RESEARCH IN PHONETICS

Susan R. Hertz
Eloquent Technology, Inc., 24 Highgate Circle,
Ithaca, New York 14850, U.S.A. and Cornell University

Elizabeth C. Zsiga
Georgetown University, Washington D.C., U.S.A.

ABSTRACT

Syllt is a partial phone-to-speech program designed for use with the Delta System, a sophisticated software tool for teaching and research in phonetics. From a string of phonetic symbols representing a CVC or VCV utterance, Syllt creates a multi-tiered utterance representation (*delta*) from which parameter values for a Klatt synthesizer are automatically derived. The deltas can be modified either interactively with simple commands, or automatically with built-in or user-defined Delta language procedures. Syllt can also quickly implement stepwise changes to a delta to generate stimulus continua or matrices.

INTRODUCTION

The Delta System is a flexible software tool for natural-language processing, with specialized features for speech synthesis. It runs on PCs (Windows or DOS), and Sun and SGI workstations (UNIX). The system includes a linguistically-oriented programming language called Delta [1], and an interactive environment called DeltaTools, both designed for ease in building, manipulating, and synthesizing from nonlinear, "multi-stream" utterance representations (*deltas*). Deltas can be built either with a program expressed in the special Delta programming language, or with interactive DeltaTools commands (or a combination of both). While the system makes manipulating deltas easy, building deltas from scratch can be difficult, especially for users inexperienced in speech synthesis, since they may not know exactly what parameter values to insert. Syllt (derived from Eloquent Technology's more complete text-to-speech program, Eloquence) was designed to alleviate this difficulty [2].

The input to Syllt is a string of phones representing a CVC monosyllable or VCV disyllable in General American English (where C is any stop or fricative and V is any simplex vowel). The output is a delta containing phonological units (e.g., phoneme symbols and associated features), phonetic units (e.g., bursts), and quantitative parameter values (e.g., formant target frequencies) for a Klatt synthesizer [3]. For example, to synthesize the word *toe*, the user would enter it phonetically as:

(1) `to`

A portion of the delta (slightly simplified for space reasons) that the program would construct is shown below:

(2)
```
phone:    |t   |  |o         |  |
trans:    |    |tr|          |tr|
F1:       |300 |  |550 |   |400 |
F2:       |1600|  |1250|   |850 |
AV:       |0   |0 |58        |0 |
AH:       |0   |63|0         |45|
Ms:       |60  |80|0   |210|0|60|
          +----+--+----+---+-+--+
          1    2  3    4   5 6  7
```

This type of representation is called a delta because it consists of "streams" of information of the user's choice. The vertical bars, called *sync marks* (or synchronization marks), coordinate units across streams. All vertical bars in the same column represent the same sync mark. An important property of the delta data structure is that it can combine abstract linguistic information with numeric information in a single integrated representation. The above delta fragment contains abstract linguistic streams representing phone and transition units (see below) as well as streams for first and second formant target values in Hz (`F1` and `F2`), voicing amplitude values in dB (`AV`), aspiration amplitude values in dB (`AH`), and a timing stream called `Ms` that coordinates the units in the other streams. There are also other streams not shown. Although not visible in the above representation, the tokens in the `phone` stream have associated features, such as `consonant`, `vowel`, and, `stop`, which are used by the Syllt program in determining the appropriate acoustic values. From the information in the delta, Syllt generates a set of parameter values for a Klatt synthesizer.

Consider first the `F2` information in the delta. A second formant value of 1600 Hz during the phoneme [t] is followed by an 80 ms transition to an F2 value of 1250 Hz at the beginning of the phone [o]. The values between adjacent sync marks will be interpolated over the specified duration to produce the final synthesizer values. There is no voicing or aspiration (represented by 0 dB for the parameters `AV` and `AH`) during the [t], and there is 63 dB of aspiration amplitude during the transition from the [t] to [o]. At the beginning of [o], voicing comes on and aspiration turns off. The theoretical basis for the phone and transition structure of the deltas constructed by Syllt (and by the more complete program Eloquence) is motivated in [4, 5].

Deltas are constructed by Syllt using context-sensitive rules expressed in the Delta programming language. When Syllt is operating, these rules are visible in one window. In another window, the user can issue interactive commands to trace the operation of the program, to watch the derivation of a delta, to manipulate the delta and synthesize from it, to automatically generate a sequence of deltas (and resulting speech files) differing along one or more dimensions, and much more. Several of these capabilities are illustrated in the sections that follow.

MANIPULATING DELTAS INTERACTIVELY WITH DELTATOOLS

With simple DeltaTools commands, users can easily modify any aspect of the delta, listen to the result, and play selected utterances back to back. For example, a user might modify formant or fundamental frequency trajectories, frequency or duration of burst noise, or the timing of voicing onset relative to stop release. Such manipulations might be used, for example, to illustrate perceptual effects to students, to test perceptual hypotheses, or to test the effect of synthesis rules before incorporating them into a program.

The following commands illustrate one way to make the transition from [t] to [o] in the delta in (2) voiced rather than aspirated:

```
(3) delta insert [AH 0] 2.3
    delta delete AV 2.3
    delta delete AV 3
```

The first command replaces the 63 dB aspiration token between sync marks 2 and 3 (the transition from [t] to [o]) with 0 dB. The second deletes the 0 dB `AV` value in the transition. The final command deletes sync mark 3 from the `AV` stream, to extend the 58 dB voicing value to start at the beginning of the transition. Note that the commands refer to the sync marks by number. Though not shown here, sync marks can also be given mnemonic names. The following delta fragment shows the effect of the above commands. Only the relevant streams are shown:

(4)
```
phone:    |t   |  |o         |  |
trans:    |    |tr|          |tr|
AV:       |0   |58           |0 |
AH:       |0   |0 |0         |45|
Ms:       |60  |80|0   |210|0|60|
          +----+--+----+---+-+--+
          1    2  3    4   5 6  7
```

The user can position values anywhere in the delta, whether there is an existing sync mark at the desired time point or not. For example, consider the following commands, which when applied to the delta in (2) would cause voicing to overlap aspiration, starting 30 ms before the end of the transition from [t] to [o], rather than at the beginning of the [o]:

```
(5) delta delete AV 2.6
    delta insert [AV 58] (3-30).6
    delta delete AV 2
```

The first command deletes the two `AV` values (0 and 58) between sync marks 2 and 6. The second positions the value 58 to start 30 ms before sync mark 3 in the AV stream and to end at sync mark 6. The third command deletes sync mark 2 from the AV stream, extending the 0 dB token from the beginning of the delta to the point where voicing begins in the transition. The following delta fragment shows the effect of these commands when applied to the delta in (2):

(6)

```
phone:  |t   |     |o         |  |
trans:  |    |tr   |          |tr|
AV:     |0      | 58          |0 |
AH:     |0   |63   |0         |45|
Ms:     |60  |50|30|0   |210|0|60|
        +----+--+--+----+---+-+--+
        1    2  3  4    5   6 7  8
```

Note that the 80 ms time token in the transition has automatically been divided into two tokens, 50 and 30, to position the voicing value at the specified point. In general, sync marks are placed in the Ms stream anywhere an acoustic event begins or ends, and the time tokens are divided accordingly.

MANIPULATING DELTAS AUTOMATICALLY WITH PROCEDURES

One of the most onerous tasks in speech perception work can be creating a synthetic stimulus continuum, a series of synthetic stimuli that vary along one or more parameters. With built-in Syllt procedures, users can automatically generate sequences of synthetic utterances that differ systematically along one or more dimensions.

Assume, for example, the user wishes to create a VOT continuum, beginning with a fully aspirated transition, changed to fully voiced in 20 ms steps, starting with the delta shown in (2). Rather than create such a continuum with a large number of interactive commands of the sorts illustrated above, the user can create such a continuum with two simple commands:

```
(7) set_pointers(2,3)
    vot(58,63,80,20)
```

The first command invokes a procedure that delimits the stretch of the delta to be manipulated (the stretch between sync marks 2 and 3). The second command invokes the procedure vot to create the continuum; it specifies the desired voicing value (58), the aspiration value (63), the total duration of the specified stretch (80), and the step size (20). The vot procedure then creates a series of deltas and accompanying parameter and speech files with the /t/ gradually changing to /d/. The first three deltas in the series are shown below. Note the changes in the AV, AH, and Ms streams during the transition between [t] and [o].

(8)

```
phone:  |t   |  |o         |  |
trans:  |    |tr|          |tr|
AV:     |0   |0 |58        |0 |
AH:     |0   |63|0         |45|
Ms:     |60  |80|0   |210|0|60|
        +----+--+----+---+-+--+
        1    2  3    4   5 6  7

phone:  |t   |     |o         |  |
trans:  |    |tr   |          |tr|
AV:     |0   |0 |58|58        |0 |
AH:     |0   |63|0 |0         |45|
Ms:     |60  |60|20|0   |210|0|60|
        +----+--+--+----+---+-+--+
        1    2  3  4    5   6 7  8

phone:  |t   |     |o         |  |
trans:  |    |tr   |          |tr|
AV:     |0   |0 |58|58        |0 |
AH:     |0   |63|0 |0         |45|
Ms:     |60  |40|40|0   |210|0|60|
        +----+--+--+----+---+-+--+
        1    2  3  4    5   6 7  8
```

Each of the deltas and accompanying parameter and speech files is automatically named sequentially and saved. The deltas can be recalled for subsequent manipulation with the Delta System, and the speech files can be played later in an experimental setting. A log file keeps track of the name of each delta, and what it contains.

While the above continuum varies in just one dimension, Syllt can also create a two-dimensional matrix. For example, the user could create a vowel space by varying F1 and F2. The following commands, when applied to the delta in (2), change F1 for the entire vowel from 300 to 800 Hz in 250 Hz increments:

```
(9) set_pointers(3,7)
    matrix(F1,300,800,250)
```

In the next step, a second dimension is created, varying F2 from 1500 to 1800 Hz in 150 Hz steps for each of the stimuli created by the previous commands:

```
(10) dimension 2
     matrix(F2,1500,1800,150)
```

Syllt includes procedures for creating continua for VOT, F1, F2, F3, and duration, but the user can easily use these as models to write procedures for other parameters.

MODIFYING SYLLT

Syllt is structured into three main modules: (1) "abstract linguistic" rules, which insert abstract structure such as phones and transitions, (2) "phone" rules, which fill in the acoustic and durational values specific to particular phones, and (3) "default" rules, which fill in any values that remain constant over the whole utterance (such as values for F4 and F5). The user can quickly learn the structure of these modules by tracing the operation of the rules using DeltaTools commands, and watching how the deltas are changed by them. Syllt is also accompanied by extensive documentation, containing a complete description of the structure of the program, and a number of hands-on tutorials to aid the user in learning how to write rules in Delta, trace Delta programs, manipulate deltas, etc. All source code for Syllt is provided.

Users can modify the structure of Syllt for different needs. For example, for teaching purposes, an instructor might want to suspend the application of the phone rules, so that just the abstract linguistic and default rules apply. For a given input string of phones, the program would then create a template into which students could insert the phone-specific values interactively as they learn about different acoustic cues. Students can also write their own rules for filling in acoustic values, and incorporate them into the program.

Users might also wish to modify the existing Syllt modules—to create deltas for a different language or to add new phone types, for example. They might also wish to add new modules to Syllt—perhaps a filter that modifies the values in the delta to create a different voice quality.

CONCLUSION

The Delta System with Syllt provides increased capabilities for teaching and research in phonetics. Teachers will find the program useful for demonstrating the importance of different acoustic cues and for giving students a head start on their own synthesis projects. Researchers will appreciate the natural-sounding utterances that are automatically generated, the easy and precise control over acoustic parameters, and the speed with which stimulus continua can be created. Syllt gives the Delta System added power and flexibility to meet the needs of a wide variety of users.

ACKNOWLEDGEMENTS

The authors thank Jerry Lame and Katherine Lockwood for useful comments on the manuscript. This research has been supported in part by Grant DC00758-03 from NIDCD to Eloquent Technology, Inc.

REFERENCES

[1] Hertz, S. R. (1990), "The Delta programming language: an integrated approach to non-linear phonology, phonetics, and speech synthesis", *Papers in Laboratory Phonology I: Between the Grammar and the Physics of Speech*, J. Kingston and M. Beckman (eds.), Cambridge University Press, 215-257.

[2] Hertz, S. R., E. C. Zsiga, and M. K. Huffman (1994), "Syllt for building deltas: simple speech synthesis for teaching and research," *J. Acoust. Soc. Amer.* 95, No. 5, Pt. 2, 2815.

[3] Klatt, D. H. and L. C. Klatt (1990), "Analysis, synthesis and perception of voice quality variations among female and male talkers," *J. Acoust. Soc. Amer.* 87, 820-857.

[4] Hertz, S. R. (1991), "Streams, phones, and transitions: toward a phonological and phonetic model of formant timing", *J. Phon.* 19, 91-109.

[5] Hertz, S. R. and M. K. Huffman (1992), "A nucleus-based timing model applied to multi-dialect speech synthesis by rule", *Proc. Int. Conf. Spoken Lang. Proc* 2, 1171-1174.