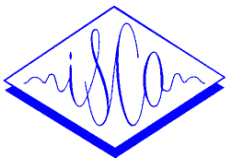

2nd International Conference on
Spoken Language Processing (ICSLP 92)
Banff, Alberta, Canada
October 12-16, 1992

# DEGAS: A SYSTEM FOR RULE-BASED DIPHONE SPEECH SYNTHESIS

Leonard C. Manzara and David R. Hill

Computer-Human Systems Lab, Department of Computer Science
University of Calgary, Calgary, Alberta, Canada T2N 1N4

## ABSTRACT

This paper describes DEGAS, a Diphone Editor and Generator for Animation and Speech. DEGAS is a simple to use, yet powerful computer program designed to give researchers the means to develop speech synthesis rulebases in a diphone framework. It is an open system, allowing an unlimited number of arbitrary symbols for phones, synthesizer parameters, and categories. An arbitrary number of rules can be formed by combining these symbols with boolean operators. Target values for every parameter are prescribed by the user to define postures for each phone. A transition profile, zero or more special event profiles, and duration rules are specified by the user to define the interpolation between the two sets of targets in a diphone pair. DEGAS provides immediate audio and visual feedback, as well as a means to create a complete diphone inventory.

10.21437/ICSLP.1992-32

## 1 INTRODUCTION

Ever since the advent of electronic speech synthesizers, researchers have been searching for an understanding of how to control these devices to produce intelligible and natural sounding speech. Good results are scarce not because the devices are incapable of producing lifelike speech. In fact, initial experiments have shown that if the controls are copied from real utterances, a moderately complex synthesizer can produce speech nearly identical to that of a human [3]. Indeed, what contributes most to the degradation of the resulting speech when doing synthesis-by-rule is our inability to prescribe perceptually significant speech cues adequately, with a limited set of rules.

What has often impeded researchers the most is the lack of a convenient framework within which to postulate and then evaluate these rules quickly and easily. In the past, the rulebase has been formulated using conventional or specially developed programming languages. While very flexible, using these languages means that changes are usually accomplished by recoding the rules, recompiling the code, and then evaluating the results. This can be very tedious. Another problem is that such schemes are usually limited to one synthesizer and to one phonetic representation.

It is with these problems in mind that a new utility was conceived. DEGAS, a Diphone Editor and Generator for Animation and Speech, is a system which is simple to use, yet with enough generality and power to allow the researcher to develop sophisticated rulebases of unlimited size and detail. An arbitrary number of rules can be formed with user-specified symbols and with familiar boolean operators. Conflicts in scope are mediated by simple order precedence. The user can also specify an unlimited number of arbitrary symbols for phones and synthesizer parameters. In practice, this means that the system is not tied to one particular phonetic representation or to one synthesizer. This also allows devices other than a speech synthesizer to be driven from the same rulebase. DEGAS features immediate audio and graphical feedback, and the user interacts with the application using the intuitive NeXTSTEP graphical user interface running on a NeXT computer workstation.

## 2 BACKGROUND

### 2.1 Speech Synthesis-by-Rule

When discussing the artificial production of speech, it is customary to distinguish between speech synthesis-by-rule, speech synthesis-by-copying, and speech production from recordings. Strictly speaking, speech production from recordings should not be considered synthesis since the process depends primarily upon the retrieval of pre-recorded utterances. Speech synthesis-by-copying, in contrast, uses some sort of hardware or software synthesizer. The parameter tracks used to drive the synthesizer are derived from analyses of actual recorded utterances. Usually these are from spectrograms, and are copied by hand, but automatically derived LPC coefficients can also be used. Speech synthesis-by-rule involves the automatic creation of parameter tracks to drive a synthesizer using a small set of formal rules. The nature of such rules obviously depends upon the type of input into the system, and upon the parameters needed to drive the synthesizer.

Klatt [3] divides techniques for doing segmental synthesis-by-rule into three broad categories:

- heuristic acoustic-domain rules to control a formant synthesizer;
- articulatory rules to control a model of the larynx and vocal tract; and
- strategies for concatenating pieces of encoded natural speech.

To these categories we can add:

- rules to control acoustic-domain synthesizers (other than formant synthesizers); and
- rules to control event-based synthesizers.

Note that these categories are based primarily upon the type of speech synthesizer to be driven by the rules. It *is* possible to create a common framework for rule specification to drive all the above types of synthesizer except, perhaps, for the synthesizer based upon the concatenation of natural speech. This is, in fact, one goal of our research. The generalization of a rule framework to encompass most types of speech synthesizer is a powerful tool. It allows rules developed for one system to be easily transferred to another system, and provides a convenient means to evaluate different speech synthesizers.

Implicit in the description of rule frameworks is the role of the computer. It is usually understood that the rules are to be automatically applied by the computer to synthesize the desired speech. This also implies that the rules themselves can be formalized, and thus programmed. In most speech-by-rule systems the rules are embodied in a computer program as as a set of logical or mathematical statements, with possible reference to one or more tables of values. Since the rules are "hard coded" using a conventional programming language, it is quite tedious to make changes to the rules because this involves editing a source program, and then recompiling it. Evalu-

ating the effect of these modifications is laborious, and is a powerful disincentive to making any changes at all.

Carlson and Granström [1] have developed a special programming language to allow synthesis rules to be formulated similarly to the Chomsky and Halle formalism. Rule statements are easier to understand and change in this format, and also permit a number of different natural languages to be synthesized. Unfortunately, the rules must still be compiled into a synthesis-by-rule program before the results of these changes can be evaluated. Nevertheless, such a rule compiler encourages a more natural formulation of the synthesis rules than would be attained with a conventional programming language, and also discourages language-specific "ad hoc" fixes, many of which have unpredictable side effects.

The DEGAS system attempts to capture the power of the rule compiler without the need to iterate the edit-compile-evaluate cycle. The rules themselves are given graphic form wherever possible, making rule specification more directly relatable to common speech analysis output such as the spectrogram, and thus more intuitive to use. Just as importantly, the system can be used with almost any speech synthesizer desired since the number and type of parameters are not preconfigured.

It is important to remember that speech is not just a series of interchangeable segments, but a dynamic progression of target postures and intervening articulatory gestures. These vary according to context, and interact in subtle ways, generally classified under the rather broad heading of coarticulation. This is a view of speech at the segmental level. In addition, there are suprasegmental effects—patterns of rhythm and pitch movement that extend over many segments. Some effects (such as micro intonation) involve disturbances at one level (segmental events) affecting parameters associated with the other level (pitch value).

Any satisfactory framework for a synthesis-by-rule system must allow the specification of:

1. quasi-steady-state sounds (related to target postures);
2. transient sounds (related to articulatory gestures and the dynamics of the changing acoustic system);
3. coarticulation effects; and
4. suprasegmental effects (or at least provide means of integrating control of suprasegmentals into the segmental level description).

### 2.2 Diphones

Klatt [3] defines the diphone as ". . . the acoustic chunk from the middle of one phoneme to the middle of the next phoneme." The diphone is an attractive segmental level unit to work with because it captures most of the essential allophonic variations in the transition of any one phoneme to any other explicitly. Also, since the "steady-state" part of most phonemes is relatively steady, it is possible to synthesize unlimited continuous speech by concatenating units from a small set of source diphones. Dixon and Maxey [2] posed the question: "Can a library of synthetic segments be devised which will be acceptable to listeners if the segments are forced to fulfill multiallophonic roles in continuous speech?" Their answer is yes, given that the library is large enough, and thus contains enough phonetic variants. Since there are about 40 phonemes in English, the total number of diphones is that number squared, or about 1600. Stressed versions of these diphones are also needed, bringing the typical number of diphones in a library to about 3200. However, not all diphones will be needed when synthesizing typical speech because some sequences of phonemes never occur.

Diphone concatenation was first applied to recorded segments of live speech. Such systems did not work very well because of difficulties in smoothly joining segments in the time domain. If the diphone boundaries do not exactly match, an audible click will be produced due to a waveform discontinuity. Smoothing at the boundary can reduce this effect, but any mismatch of vowel quality on either side of diphone boundary ruins the illusion of connected speech. Also, the concatenation process cannot take into account the effect of higher level features such as stress, rhythm, and intonation on the segmental level assembly. The approach merits more discussion but space is limited. The important point is that this simple approach does not do justice to the interactions previously mentioned unless the diphone database is enormously expanded. DEGAS attempts to circumvent problems associated with this expansion by rules which capture the interactions economically.

## 3 THE DEGAS SYSTEM

DEGAS is currently in use at the Computer-Human Systems Lab at the University of Calgary. Although it allows one to see and hear the results of one's work, DEGAS is not a complete speech and animation facility in itself. Rather, it was designed to create and edit diphone generating databases used as part of other speech and animation systems. DEGAS has been in use for nearly a year, and has been used to create a successful commercial text-to-speech system on the NeXT computer.

### 3.1 General Design Principles

The DEGAS system was designed with the following principles in mind:

- Generality and expandability. The utility should not restrict the number and type of phones and parameters which may be specified by the user.
- Rules should be intuitive and natural using familiar logical operators and user-specifiable operands.
- Visualization aids should be used wherever possible.
- Manipulations in the system should be achieved with a small number of abstractions that are general and powerful enough accomplish all needed operations.
- Ease of use. The utility will dispense with the code/compile/evaluate cycle, thus providing a tight feedback loop for evaluation. The effect of changes should be heard immediately.
- The user should be able to compare an unlimited number of different versions of his or her data.
- The utility should provide a uniform user interface across all levels of control.

### 3.2 Program Structure

DEGAS provides a graphical user interace constructed according to the NeXTSTEP operating system and Interface Builder conventions. The functions are modularized within menu-selectable windows.

#### 3.2.1 Template

The Template window allows the user to define the number and symbolization of phones and parameters, together with parameter ranges and defaults. Durations of phones are also specified. Symbols are arbitrary, within the ASCII character set, and values are all floating point.

#### 3.2.2 Categories

When formulating rules the user needs to be able to classify phones into arbitrary groups. The Categories window allows the user to specify an unlimited number of categories. A phone necessarily belongs to the super-category *phone* as well as the minimal category represented by its unique name symbol. It may belong to an arbitrary number of other categories as defined by the user.

### 3.2.3 Phone Description

The Phone Description window allows phone-specific information to be defined, namely:

- the target value for each parameter;
- the total duration of the phone in milliseconds;
- the transition duration of the phone in milliseconds, or as a percentage of the total duration; and
- the categories in which the phone belongs.

### 3.2.4 Rule Specification

Once the template, category, and phone descriptions are complete, the user is ready to specify segmental synthesis rules. These rules determine how the transition is to made between one posture (phone) and another, creating arbitrary diphones as needed. The form of these rules is very simple: given a particular *transition specifier*, create the transition using a user-specified *transition profile*, zero or more *special event profiles*, and a *duration rule*.

**Transition Specifiers.** The transition specifier has the form:

$$[category1] \rightarrow [category2]\,.$$

The user can specify any valid category for *category1* and *category2*. For instance, the transition specifier

$$[\text{t}] \rightarrow [\text{vowel}]$$

says that for all diphones which consist of the transition from the phone "t" to any phone categorized as a "vowel", do something particular (as described below).

The transition specifier also allows "compound categories" composed of any number of category names plus the logical operators *and*, *or*, *xor*, *not*, and the grouping symbols "(" and ")". For example, the transition specifier

$$[\text{vowel and not o}] \rightarrow [\text{consonant}]$$

says that for all diphones which consist of the transition from all "vowels" except "o", to all "consonants", perform a yet to be determined transition. A compound category can be arbitrarily complex, and is evaluated with normal boolean logic.

**Precedence of Transition Specifiers.** The user can create as many rules as needed. Each rule will, of course, have a transition specifier which specifies the scope over which the particular rule has effect. It may be possible that some rules overlap in their scope. The DEGAS system uses a very simple scheme to arbitrate which rule has precedence whenever there is a conflict. The user simply lists the transition specifiers from high to low precedence. When the complete inventory of diphones is calculated, each particular diphone is forced to use the first rule in this list under whose scope it falls. In practice, the user would list the rules in order from the particular to the general, so as to ensure that general principles were overridden by exceptions. The user can create as many or as few rules as needed. However, there must always be at least one transition specifier, namely:

$$[\text{phone}] \rightarrow [\text{phone}]\,.$$

The guaranteed existence of this transition specifier ensures that there will always be at least one rule to cover every single diphone. In effect, this becomes a "catch-all" rule and, because of its generality, is normally listed last in the precedence list.

**Transition Profiles.** Each phone or posture is described by a number of target values. The transition profile describes how the transition is to be made from the target values for one phone to those for the next. The profile is applied equally to all parameters.

The transition profile is divided into a number of *intervals*. The user can choose to have as many or as few intervals as desired, although there must be at least one. Each interval can either be fixed in length (in milliseconds), or can be a proportion of the overall length of the diphone (a percentage). At least one of the intervals must be proportional because the computed total length of the diphone may vary, and this proportional interval is needed to "fill up" any excess time. The number and length of intervals can also be made a function of the duration rule (see below). In this case, the user cannot change the number and length of intervals, but may alter the values of the transition profile in the parameter value dimension.

Once the length type (fixed or proportional) and number of intervals have been specified, the user can manipulate the variations in parameter value at each interval boundary, either directly (as a parameter value) or indirectly (by specifying the relative slopes of successive intervals). The former is called *fixed rise mode*, and the latter *slope ratio mode*. Since some intervals may be fixed in duration, while others are proportional, the shape of the transition profile may be distorted for diphones of different length in fixed rise mode. In contrast, the slope ratio mode preserves the overall shape of the transition profile, regardless of the length of the diphone. In this mode, the user specifies the slopes of each interval as a ratio (for example, 1:6:1 for a profile with three intervals). No matter what the length of each interval, the slopes will always maintain the same ratio, and thus preserve the same shape. DEGAS allows the user to switch between these modes at any time.

Any interval can be designated as "stretchable". If the speed of a synthesized utterance is changed (i.e. the length of the diphone is altered), stretchable intervals are expanded or contracted while the other intervals remain fixed in length. This allows more natural sounding changes of speed.

**Special Event Profiles.** A special event profile may be created for any parameter. This allows the user to specify parameter movements which deviate in some way from the transition profile applied generally to all parameters. The special event profile inherits its timings (i.e. the interval structure, including number and length type) from the transition profile created for the rule under consideration. In the simplest case, the special event profile may be used to alter the $y$ values of the transition profile to modify its shape. This merely results in changes of slope of one or more intervals. In other cases, the special event profile may be used to split the intervals inherited from the transition profile, creating an arbitrary number of sub-intervals. This is useful when "special events" such as noise bursts need to be prescribed.

**Duration Rules.** The duration of the diphone is calculated using one of a number of duration rules such as: $P_1$, or $\frac{P_1}{2} + T_1 + \frac{P_2 - T_2}{2}$, or $\frac{P_1 - fixed}{2} + fixed + \frac{P_2 - fixed}{2}$, where $P_1$ is the length of the first phone in milliseconds, $P_2$ the length of the second phone, $T_1'$ the transition duration of the first phone, $T_2$ the transition duration of the second phone, and $fixed$ an arbitrary length supplied by the user in milliseconds.

### 3.2.5 Synthesis and Display

The user can synthesize and display any arbitrary sequence of diphones at any time using the Synthesize Utterance window. The parameter tracks are created using the current values in the Template, Categories, Phone Description, and Rule Specifications windows, and thus the results of any changes can be heard and seen immediately. The user can enter either phones or diphones, as long as the phones have been pre-defined in the Template window. Options to vary pitch, speed, and graphic display are also provided.

## 4 PRACTICAL USAGE

DEGAS has been in practical use for nearly a year. We found it easy and convenient to use, both as an experimental tool, and as a utility to create databases for a text-to-speech system. In two weeks, we

created a complete diphone generating database from scratch containing 118 phones and 12 parameters. What is most remarkable is that only 57 rules were developed, and these rules specify 13,924 diphones. To edit each diphone individually would have been an extremely laborious task, and the quality of a database so created would consequently suffer.

Altering or adding to the rulebase is extremely easy to do. We would often discover a particular diphone pair that was not rendered particularly well with a general rule. DEGAS makes it easy to augment the rulebase to deal with such exceptions, since an individual diphone can be edited in the same manner as a class of diphones.

We also found that adding new phones was necessary. For example, allophones of "h" were easily created to cope with the fact that this phone is not effectively rendered as a single set of fixed targets. The fifteen allophones were added to the database in less than half an hour, and no new rules were needed since the rules for phones we classified as "aspirates" were already in place.

Some experimentation with the target values for particular phones was necessary. DEGAS allows quick changes to these, and the results could be auditioned immediately. Since subtle adjustments to factors such as relative balance between frication and voicing could be heard and evaluated so easily, we were able to refine the database in very short order.

An important feature of DEGAS is the self-documenting nature of its structure. Since the rules are formulated in a natural and intuitive manner, we found that the rationale for previously developed rules was easy to grasp, so that editing or extending the rulebase required little mental effort. In fact, the ease with which new rules could be developed and discarded meant that we often explored several new ways of formalizing a speech synthesis process before settling on what we thought was the best. Another important characteristic of DEGAS is the clear structure of the rulebase. This helped us to avoid unintended side effects, a problem we frequently encountered in older, compiled-language systems. Since the transition specifiers can be tailored to have a precisely defined scope, a newly developed rule almost never has an unintended effect, nor undoes the effort put into rules already developed.

Perhaps the most important problem solved by DEGAS is the logistics of keeping track of the massive amounts of detail needed to reproduce allophonic variation. An earlier attempt to incorporate such detail into a diphone database raised the spectre of hand-editing a graphical representation of between 8 and 20 parameter profiles for over 3000 diphones. Not only is this task impossible in practice, but the data is only defined implicitly in the stored parameter traces. DEGAS has not only reduced the problem to manageable proportions, accessible to incremental attack, but has provided an explicit analytical representation of every detail involved. This is a major step forward in our program towards understanding the structure and variability of speech.

## 5 POSSIBLE IMPROVEMENTS

DEGAS is not without its flaws and several improvements could be made. First there is a need to generalize the method of specifying duration rules. Although the eight rules hard coded into the program were adequate for our immediate purposes, we realized very quickly that the user should be able to formulate these rules in a much more flexible manner. This would involve writing a parser which could handle simple mathematical expressions and recognize user-specified constants and variables. The Phone Description window would also have to be changed so that arbitrary symbols and values could be associated with each phone. These would be used in the duration rule formula created by the user.

Another problem is the fact that only one set of targets is allowed for each phone. Quite often we found it desirable to be able to specify an alternate set of targets, or to interpolate between more than the usual two targets. One can create special allophones to work around this problem, but it is not an entirely satisfactory solution.

Better control over the Special Event profiles is also needed. In some cases, the absolute deviation mode of the current system was hard pushed to create the effects we wanted on particular parameters. A mode is needed where the profile is not added to the inherited transition profile, but substitutes for it.

## 6 CONCLUSION

The DEGAS system is intended to provide a simple framework in which to formulate segmental level synthesis rules. Since only a few universally applied abstractions are used, the user can formulate rules to any degree of complexity at all levels of description. This simplifies the conceptual task of the researcher, and reduces the manipulations within the system to a limited set of procedures. The ease with which the system can be navigated also allows the researcher to experiment with a considerable amount of freedom. The ability to create and discard new rules will, we are sure, lead to new theories and formalizations with regard to speech processes. The fact that that DEGAS can be used with any type of parametric speech synthesizer also suggests that previously distinct formalizations will be brought into closer juxtaposition, resulting in a new understanding of the production of speech. Most importantly, everything we learn is now explicitly represented in the databases used to generate diphones. This is of considerable consequence not only for for speech synthesis, but also for speech recognition.

## 7 ACKNOWLEDGEMENTS

This work is supported by the Natural Sciences and Engineering Research Council of Canada, grant no. 0GP0005261.

## References

[1] R. Carlson, and B. Granström. "A Phonetically Oriented Programming Language for Rule Description of Speech", *Speech Communication* (Ed. G. Fant), Volume 2, 245–253. Uppsala, Sweden: Almqvist and Wiksell, 1975.

[2] N. Rex Dixon, and H. David Maxey. "Terminal Analog Synthesis of Continuous Speech Using the Diphone Method of Segment Assembly", *IEEE Transactions on Audio and Electroacoustics.*, AU-16, 1 (1968), 40–50.

[3] Dennis H. Klatt. "Review of text-to-speech conversion for English", *The Journal of the Acoustical Society of America*, 82, 3 (1987), 737–793.