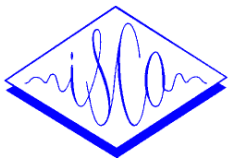


The ESCA Workshop
on Speech Synthesis
Autrans, France
September 25-28, 1990

A New Algorithm for a Concatenative Speech Synthesis System Using an Augmented Acoustic Inventory of Speech Sounds

Joseph P. Olive

AT&T Bell Laboratories
Murray Hill, N.J. 07974

ABSTRACT

Previously we discussed a speech synthesis by rule scheme that consisted of concatenating small elements of analyzed natural speech segements. These segments consisted of transitions between adjacent phonemes and were stored in terms of LPC derived area parameters. Although the speech produced from this scheme was highly intelligible, it did not sound natural or continuous. Investigation showed that most short and reduced vowels, were not described correctly by the previous method, because they depended too much on their environment. Depending on their neighbors, often, these phonemes do not reach their target and thus can not be defined by diphonic units.

Recently, we have introduced a scheme that can access a larger variety of acoustic inventory elements consisting of the previously described transitions as well as longer units. The longer multiphonic units consist of triphones for the short vowels and many common words, especially small function words. Due to the new inventory, the speech produced by the new synthesis scheme has maintained its high intelligibility, but sounds more continuous and pleasant.

1. INTRODUCTION

This paper describes a concatenative synthesis algorithm using variable length units as the basis of the synthesis scheme. The concatenative algorithm is the last stage in our text-to-speech system. which is divided into two major parts; the first part a text analysis part called *express*,[1] and the second, a phonemic synthesis part called *utter*. [2] The text analysis part of the system preprocesses the input text and converts non-alphabetic symbols, abbreviations and acronyms into words. The text is next parsed into a hierarchical structure of phrases and word groups with assigned syntactic stress levels. Next, the pronunciation of each word including lexical stress is established, and the prosodic variables of intonation timing and loudness are computed. Thus, *express* converts the input text into a string of phonemes with their associated stress marking and duration; in addition, *express* computes a pitch and amplitude contours for each phrase in the text. *Express* sends this information to *utter* which in turn converts this information into an audible speech signal. *Utter*, the last stage of the text-to-speech system is the subject of this paper.

Synthesizing speech from phonetic input can be accomplished from rules governed by the phoneme strings[3] [4] or by concatenating short segments of parametrized speech. [5] In the past, our system consisted of concatenating diphone-like units represented by LPC derived parameters. The synthesis was highly intelligible, but the resultant speech lacked in its quality and naturalness.

2. *The Limits of Diphonic Synthesis*

Our Previous synthesis algorithm concatenated diphonic units. The scheme simulated the phonemes by connecting an incoming transition of the phoneme from one diphone and an outgoing transition from another diphone. However, it is well known that the realization of a phoneme depends on its context. When a concatenative scheme relies on diphones as the units for synthesis, it is incapable of simulating the complete contextual variation of many phonemes. The resultant speech sounds overarticulated because of the inability of the system to simulate certain coarticulation effects. Relying on diphones may also introduce some mismatches in spectra at the point in time where two segments are abutted producing clicks and other nonspeech-like artifacts.

Our present synthesis scheme has adapted a more flexible approach to the length of the stored synthesis units. Some units are still diphones while others encompass three or more phonemes, because although the realization of a phoneme depends on its neighbobors, the amount of this dependency is a function of the duration of the phoneme and the strength of the articulatory gesture required to produce the phoneme. Thus, long phonemes such as the vowels *i* and *a*, and highly constrained sounds such as voiceless fricatives or nasals almost always reach their articulatory as well as their acoustic target. The target is defined here as a region somewhere in the middle of a phoneme which is similar in all occurrences of the phoneme regardless of their context. Sounds that reach their targets can be well described by connecting an incoming transition to an outgoing transition. Shorter phonemes that do not require a strong articulatory gesture, are extremely dependent on their context, and might not reach their appropriate target in the neighborhood of some phonemes. For these phonemes there is no guarantee that the end of the incoming transition to the phoneme will match the beginning of the outgoing transition of all possible right hand neighbors; such a match is highly desirable for high quality synthesis. The latter phonemes are embedded in multi-phone units, thus eliminating the over articulated effect and the possibility of generating clicks in the speech signal.

For this reason we have chosen synthesis segments of variable length as the basis of our new synthesis scheme. Because of our departure from the diphonic algorithm, We have named these segments "Acoustic Inventory Elements" (AIEs).

3. *The Structure of the Acoustic Inventory*

We will now describe our acoustic inventory elements. We will discuss the different lengths of these elements; how they are stored and retrieved and their use in synthesizing speech.

3.1 Types of Inventory Elements

Since the length of the acoustic inventory elements (AIEs) vary, their structure depends on their length. All AIEs start at the beginning of the outgoing transition of the first phoneme and terminate at the end of an incoming transition to the last phoneme. This is true for continuants only, stops and affricates are handled differently, and will not be described in this paper. When the element represents two phonemes (diphone) it consists of just the outgoing and incoming transitions. However, when the element represents three or more phonemes, the entire inner phonemes are represented. In the synthesis process, the parameters of the phonemes obtained from the inner section of multi-phone elements are simply adjusted for the desired duration. The parameters of phonemes at the extremities of acoustic inventory elements are obtained by joining together two such elements. To set the proper duration for a spliced phoneme, when the duration resulting from the splicing of two transitions is too short, the parameterS are linearly interpolated between the two transitions; when the duration of the resultant phoneme is too long, the entire phoneme is shortened by linear interpolation.

Only phonemes that reach their target can be separated and their transitions stored at the extremities of the AIEs. Among candidates for separating are most of the long vowels such as *i, a, and* and diphthong such as a^{i} and a^{u}; the nasals, glides, and the fricatives can also be separated and represented by their transitions. On the other hand, all of the short vowels: *I*, ɛ, ʊ, and ʌ, as well as ə are represented as triphones in all of their possible contexts. The semivowel *h,* lacks a unique spectral representation and usually serves as a transition between its two neighbors; it must also be stored in all contexts as the mid phoneme in a triphone. In addition, some special phoneme combinations are stored as triphones. Among these are: *(any phoneme) - (any vowel) - r* because of the large coarticulatory effect of the *r* on the preceding vowel; the combination of *n* - ɜ - *i* to simulate "in/on the (word beginning in a vowel)" because of the nature of the cluster *n* ; and more.

In addition to the triphones there are a few larger units in the acoustic inventory. These units serve to synthesize common occurrences of phoneme sequences such as the ones appearing in function words and numbers. Among these are elements such as *n* - ɜ - ə - *(any consonant)* to synthesize word combination like "in/on the (word beginning with a consonant)," *(any phoneme) - w* - ə - *z* for "(any word) was;" and *r - i - h* - ʌ - *n* for synthesizing "three hundred."

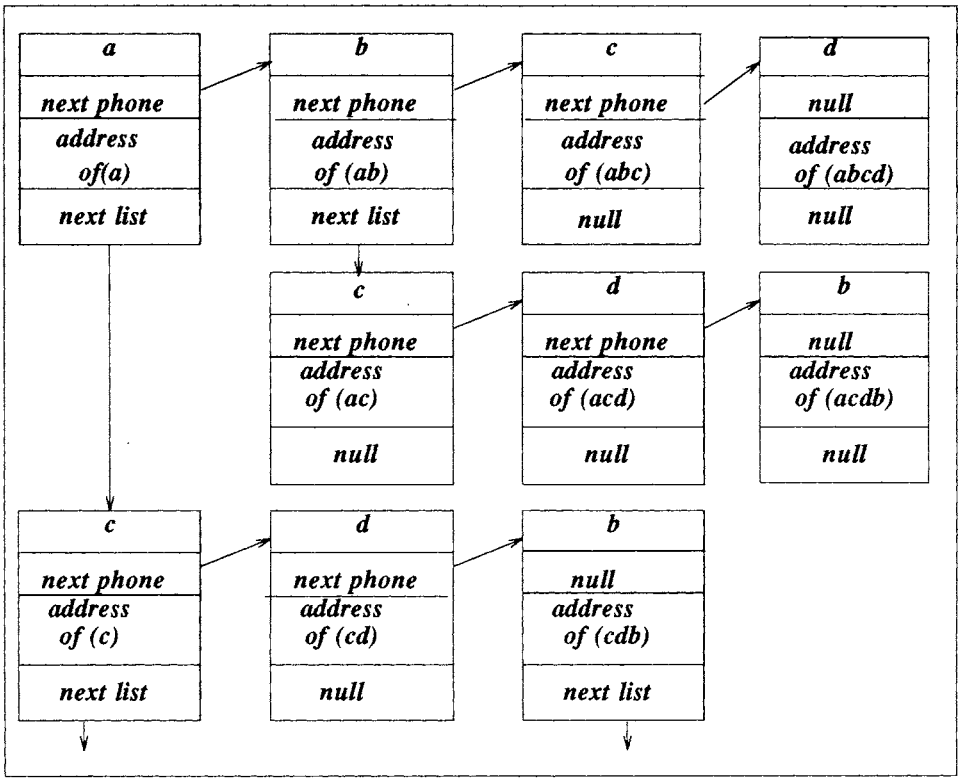
3.2 Storage of the Acoustic Inventory

The inventory elements are stored in terms of LPC derived arcsin reflection coefficients. Each element is stored with markers to indicate the location of the phoneme boundaries, and for the inner phonemes, also the middle of the phoneme is indicated. The different inventory elements are stored sequentially, but the order of storage is not important, because the location of each AIE is stored in an index table. Since the AIEs are identified by the string of phonemes that they describe, this index table must accommodate identifiers defined by an unlimited string of phonemes. To accommodate such unlimited strings, they are stored in a tree structured singly linked list. Figure 1 shows the structure of such a linked list. Each block in the figure represents a node of the tree. Every node of the tree contains the identity of the phoneme represented by the node shown in the top line of the node block; the address of the next phoneme in the string, the address of the inventory element defined by the string thus far, and the address of the next list or next branch of the tree are shown in the next three lines of each node block. The *null* address is used to terminate the nodes of the tree. Thus the addresses of the inventory elements (a), (ab), (abc) and (abcd) are stored in the top row of the figure. Since the "next phone" field of the rightmost node in the first row of the figure contains a *null*, the table does not contain any elements of length greater than four phonemes that start with the sequence (abcd). The elements starting with the string (ac) are found by going to the next list entry in column 2 of the figure, adn elements that start with the phoneme (c) are found by the "next list" entry in column 1.

3.3 Retrieval of the Acoustic Inventory Elements

The synthesis scheme computes the speech parameters by concatenating elements from the acoustic inventory. The synthesis algorithm choses AIEs that match the longest string of phonemes at any point in the input string. The algorithm computes parameters sequentially for each phoneme in the input string starting with the first phoneme; for the following discussion we will define the current phoneme as the phoneme following the last phoneme that has been already synthesized. A match is established as follows: The current input phoneme is compared with the first element in the index table; if the comparison fails, the algorithm tries the phoneme in the next list, repeating the process until a match is established or a *null* is encountered in the "next list" field. If a *null* is encountered, the algorithm terminates the

Figure 1. STRUCTURE OF THE INDEX TABLE



search; however, when a match is established, the input sting is advanced to the next phoneme, and the address of the "next phone" is used to repeat the process. If the address of the next phone is a *null* the process is also terminated. When the match at a previous phoneme produced a multi-phone AIE the algorithm uses the AIE which matches the longest sequence in the input list. If the current phoneme is the first phoneme of the AIE, it consists of an outgoing transition. To simulate the entire phoneme, this outgoing transition will be connected to the incoming transistion left over from previous computation. If the current phoneme is an inner phoneme of an AIE, the entire phoneme is obtained from the AIE. When the current phoneme is the penultimate phoneme of the AIE, the incoming transition of the next phoneme is saved for future computations.

4. Discussion

We have described our new concatenative synthesis scheme. The scheme uses acoustic elements of various lengths depending on the nature of the phonemes and their environment. At present, our inventory consists of 3000 elements; among these we have all the necessary diphonic units and approximately 1900 multi-phone units. The inventory requires approximately 2.5 Mbytes but can be compressed to .5Mbytes without loss of speech quality. Because many branches of the index table terminate at two phonemes, the search algorithm requires less than 70 comparisons for each phoneme. The improvement in the synthetic speech quality produced by this new synthesis scheme is apparent in casual listening; therefore, formal listening experiments were not required to establish its superiority. A demonstration tape will be available for listening to the improved quality.

REFERENCES

1. Buchsbaum, A. L. and Liberman, M. Y., "private communicaitons."
2. Olive, J. P., "Rule Synthesis of Speech from Diadic Units," Proc. Int. Conf. Acoust. Speech Signal Process. ICASSP-77, pp. 568-570.
3. Kelly, J. and Gertsman, L. "Digital Computer Synthesizes Human Speech," Bell-Labs. Rec 40, pp. 216-217.
4. Klatt, D. H., "Synthesis by Rule of Consonant-Vowel Syllables," Speech Communications Group Working Papers 3, MIT, Cambridge, MA, pp. 83-91.
5. Peterson, G., Wang, W. and Sivertsen, E., "Segmentation Techniques in Speech Synthesis," J. Acoust Soc. Am. 30, pp. 739-742.