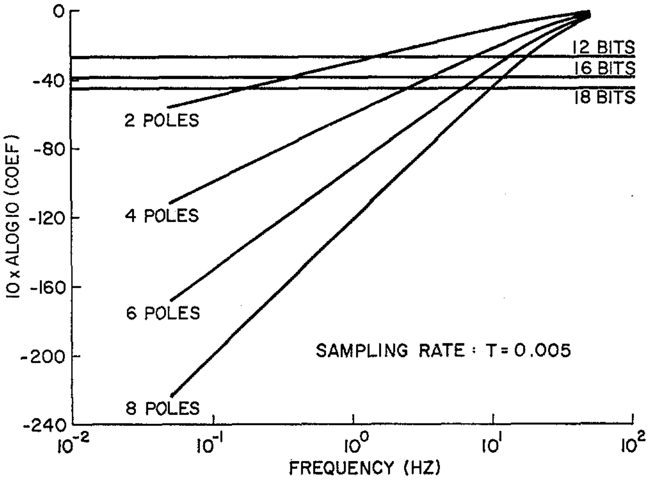


Fig. 1. Nonrecursive filter weight as a function of frequency for the sine lowpass filter.

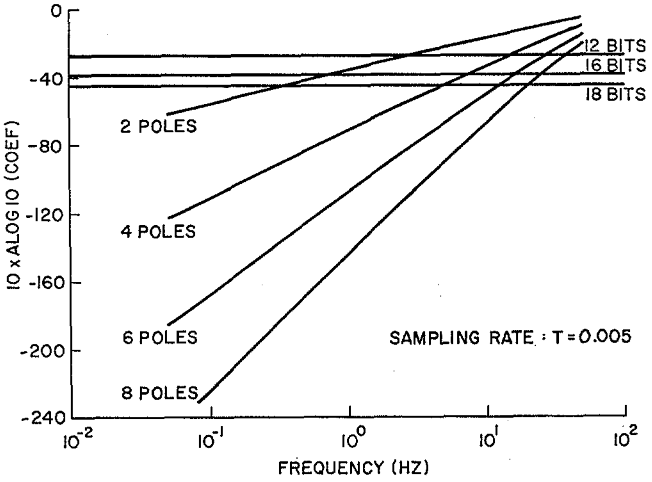


Fig. 2. Nonrecursive filter weight as a function of frequency for the tangent lowpass filter.

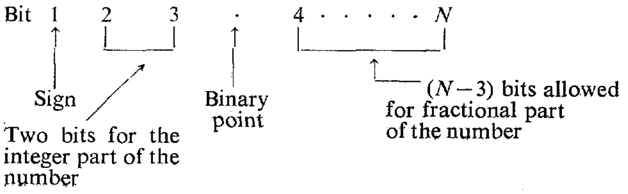
$$c_l = c^{2/M} \tag{7}$$

and

$$d_l = d^{2/M}, \qquad l = 1, \cdots, M/2. \tag{8}$$

This choice has the property of maximizing the smallest of the terms, a desirable characteristic. It should be emphasized that there are many other possible schemes that might be more effective. However, this seems to be a case of major interest.

Using the procedures given in [3] and [4], values of $c^{2/M}$ and $d^{2/M}$ are computed for a number of values of B and for four values of M. These results are shown in Figs. 1 and 2. Note that T has been taken to be 0.05, so that the Nyquist folding frequency is 100 Hz, a convenient figure for working in percentages. Also, lines corresponding to computer word size of several common machines, namely those with 12, 16, and 18 bits per computer word in ordinary single precision, are shown. In all cases, the computations are assumed to be in fixed point, single precision with the following placement of the binary point.


Bit 1 2 3 · 4 · · · · · N
Sign
Two bits for the integer part of the number
Binary point
(N−3) bits allowed for fractional part of the number

Thus the number of bits allowed for the fractional part of the number is as shown in the following.

Size of Word Bits	Number of Bits in Fractional Part
12	9
16	13
18	15

It is possible to argue that the number of bits in the fractional part could be increased by one, at the expense of one of the two bits assigned to the integer part of the word, but the given values are conservative, i.e., not likely to result in overflow. It is assumed that the data are scaled so that they are less than or equal to one in absolute value.

These lines, therefore, indicate the frequency for which a unit step input will completely underflow at the first step of the calculations, and conversely, where the same unit step function will overflow if the coefficients were made larger, in other words, the conditions for which there can never be an output or the output will overflow.

In conclusion, the lower limit for the allowable size of a passband for Butterworth filters implemented on fixed-point digital devices, at least for the above-stated configuration, is fixed by the computer word size and by the number of poles in the filter. This constraint is in addition to the separate problem of stability, which also is a function of computer word length.

References

[1] C. M. Rader and B. Gold, "Digital filter design techniques in the frequency domain," *Proc. IEEE*, vol. 55, pp. 149–171, Feb. 1967.

[2] L. B. Jackson, "Roundoff-noise analysis for fixed-point digital filters realized in cascade or parallel form," *IEEE Trans. Audio Electroacoust.*, vol. AU-18, pp. 107–122, June 1970.

[3] R. K. Otnes, "Recursive bandpass digital filter," *Proc. IEEE* (Lett.), vol. 56, pp. 207–208, Feb. 1968.

[4] ——, "An elementary design procedure for digital filters," *IEEE Trans. Audio Electroacoust.*, vol. AU-16, pp. 330–335, Sept. 1968.

A Real-Time Speech Synthesis System

W. A. AINSWORTH

***Abstract*—A system for synthesizing speech is described. It consists of an electronic synthesizer controlled by a small digital computer. The computer uses stored rules to convert a phonetic input into the analogue voltages required for driving the synthesizer. It has been found possible to make the system operate in real time so that the acoustic output is generated at normal speaking rates.**

I. Introduction

An ideal speech synthesizer should produce natural sounding, intelligible speech; should have a potential vocabulary size equal to the number of words in the English language; and should be capable of producing a continuous stream of speech without delay. Moreover, the data that specify the utterance to be synthesized should be stored in the most economical way, and the whole system should be as inexpensive and reliable as possible.

The simplest kind of talking machine is one in which the waveforms of all the potential utterances are stored, and then played back when requested. This produces natural sounding, intelligible speech; but, if the vocabulary is large, the amount of data becomes prohibitive unless sequential access devices are employed, in which case the response time becomes so great as to render the system unusable. The reason why such systems are not practical is, basically, that about 40 000 bits of information are required to specify the waveform of 1 s of speech.

A great improvement can be achieved by storing the spectrum of the speech instead of its waveform by using a channel vocoder

Manuscript received May 17, 1972. This work was supported in part by the Science Research Council.

The author is with the Department of Communication, University of Keele, Keele, Staffs., England.

type of system [1]. One second of speech can be stored in 2000 bits because the spectrum changes relatively slowly compared with the waveform [2]. A practical speech synthesis system with a small vocabulary employing this method has been developed by Buron [3].

A further improvement can be brought about by the use of a formant synthesizer [4]. In this case, the values of parameters that represent features of the spectrum, such as the frequencies and amplitudes of the major resonances, are stored. With this method of representation, 1 s of speech can be stored in about 1000 bits [5]. The disadvantage of this method is that parameter data cannot be obtained from the original speech by a simple transformation, and consequently, a certain amount of editing is required before high-quality speech is produced.

An alternative approach is to employ speech synthesis by rule [6]. Each utterance of the vocabulary is stored as a binary representation of the phonetic transcript of that utterance. The speech is generated by computing from stored tables the parameter values necessary for synthesizing the utterance, and then using these values to drive a formant synthesizer. The power of this method lies in the economy with which the speech is stored. There are about 40 phonemes (the smallest linguistically relevant elements of speech) in English, so each can be specified by 6 bits. For normal speaking rates, this suggests that 1 s of speech can be represented by only 100 bits.

This method of synthesis is obviously inefficient for systems requiring very small vocabularies, as the storage for the program may exceed the saving in data storage, but this disadvantage is rapidly overcome as the vocabulary is increased. The other disadvantage is that the time required to calculate the parameter values may be too great for systems that require an immediate vocal response. It has been found, however, that by using simple rules to calculate the parameter values, and by interweaving in time the excution of the parameter calculator program with the synthesizer driver program, a system can be built that begins to produce speech almost immediately when it is requested, and that continues as long as a sequence of phonetic symbols is supplied to it.

II. Parametric Speech Synthesizer

The speech synthesis system is based on the method described by Holmes *et al.* [6], but the rules for computing the parameter values are simpler, and the system has been designed to operate in real time. It consists of two parts: a computer program for calculating the parameter values from the phonetic input, and a hardware speech synthesizer. Although software synthesizers are more flexible and have been used for synthesis by rule [7], a hardware synthesizer is mandatory at present if real-time operation is required.

The synthesizer employed is a modified version of the one used by Holmes *et al.* [6]. A pulse generator of variable frequency or a random noise generator, selected by a switch, produces pulses that excite five resonant circuits connected in parallel. The resonant frequencies of four of these circuits are variable, but that of the fifth is fixed. The amplitude of the pulse excitation of each resonant circuit is independently variable. The outputs of these circuits are added together and used to drive a loudspeaker.

The synthesizer is controlled by 11 parameters. The position of an electrical switch (*SW*) determines at each instant whether the sound will be voiced (periodic pulses) or voiceless (random pulses). If it is voiced, the fundamental frequency (*FO*) is controlled by the rate of the pulse generator (50–250 Hz). The frequencies of three of the variable resonant circuits determine the positions of the formants (major resonances of the spectrum) in the ranges 220–1030 Hz (*F*1), 760–2560 Hz (*F*2), and 1540–3340 Hz (*F*3). The fixed resonant circuit has a center frequency of 3500 Hz when the excitation is voiced and a passband of 3600–4000 Hz when random excitation is employed. The other resonant circuit (*FN*) is always excited by periodic pulses. It is used for synthesizing nasals and voiced fricatives, and has a frequency range of 100–400 Hz. The amplitudes (*A*1, *A*2, *A*3, *A*4, *AN*) of the inputs to these resonant circuits may be varied over a range of 50 dB.

III. Parameter Synthesis Program

Speech, unlike printed text or phonetic transcript, does not consist of a sequence of discrete elements, but of a continuous stream of sound. It is often difficult to tell where one phoneme ends and the next begins. One of the simplest ways of transforming a set of discrete points into a continuous function is to draw straight lines between the points. This principle forms the basis of the parameter synthesis program.

The input to the program is a sequence of phonetic elements that correspond approximately to a set of the elements of the International Phonetic Alphabet. A further group of elements (secondary phonemes) is used to control the rhythm and intonation pattern of the spoken utterance.

The program contains a lookup table that has 12 numbers for each of the phonetic elements. Ten of these numbers represent the "steady-stage" values of the following parameters of that element: *SW*, *F*1, *A*1, *F*2, *A*2, *F*3, *A*3, *A*4, *FN*, *AN*. The other two numbers, *T*1 and *T*2, are duration parameters. *T*1 represents the duration of the steady-state part of the element, and *T*2 the duration of the transition from one element to the next. The duration parameters are quantized in 10-ms intervals.

The program works as follows. The first phonetic element is read, and sufficient steady-state values of the parameters are deposited in a buffer, so that when sent to the synthesizer, they will produce an appropriate sound for a time *T*1. The next element is then read, and values are calculated for each parameter, so that it will change in a linear fashion in a time *T*2 from its steady-state value in the first element to its steady-state value in the second element. These values are deposited in the buffer, followed by the steady-state values of the second element for the new *T*1. The third element is read, the transitional values are calculated, and the program proceeds in this manner until the last element in the sequence has been processed. The result is a set of numbers that, when sent via a multiplexer and digital-to-analog converter, produce 11 continuous time-varying voltages that control the speech synthesizer.

It is possible to produce most of the sounds of speech in this simple manner. The continous sounds, vowels and fricatives, and the slowly changing ones, diphthongs and semivowels, are very convincing if appropriate parameter values are stored in the lookup table. Voiced fricatives are produced by using the nasal circuit (*FN*), which is always voiced, in conjunction with the other circuits with switch (*SW*) set so the higher formants are noise excited. Stop consonants, which consist of a silent period followed by a noise burst and then by formant transition, cannot be generated in the simple manner described above. An extra element, corresponding to the silent period, has to be inserted, but this is done automatically by the program.

IV. Rhythm and Intronation

The system, as described so far, is good for producing isolated words, but it generates rather mechanical-sounding sentences. In order to make the speech sound more natural, it is necessary to make the pitch rise and fall, and to modify the duration of some of the elements, depending on the stress in the sentence. An explicit theory of English rhythm and intonation is not available at present, so a number of ad hoc rules have been programmed to try to improve the naturalness of synthetic speech.

There is one theory, the isochronous foot [8], which suggests that the duration of breath groups is constant. An attempt to incorporate this has been made. Stress marks are inserted in the sequence of phonetic elements after each stressed vowel. When the program encounters one of these, it checks the number of time elements since the last stress mark. If this exceeds a threshold value, the program continues; but if the number is less, the duration of the steady part of the stressed vowel is increased, so that the time between stresses is made equal to the threshold.

The intonation pattern is also determined by the stressed marks in the sequence of phonetic elements. It has been arranged so that the fundamental frequency (*FO*) will rise linearly to a maximum during each stressed syllable, and fall to a minimum at a position halfway between the stress marks. Special terminators are employed to determine whether the fundamental will rise or fall at the end of the utterance.

These simple rules do not make the synthetic speech sound like normal spoken English. They do, however, remove the mechanical quality from the speech and make it more pleasant to listen to.

V. Real-Time Synthesis

In some potential applications of speech synthesis, such as producing recorded talking books, a delay between the input of the phonetic elements and the output of the sound can be tolerated. In others, like on-line inquiry systems and vocal alert systems, a delay of more than 1 s would be annoying and possibly dangerous. It was therefore decided to try to make the system operate in real time.

A simple time-sharing system, which had been developed previously for speech perception experiments [9], was modified so that the background program calculated the parameter values from the phonetic elements, while the foreground program converted the parameter values into voltages to control the synthesizer. A delay of

250–500 ms had to be inserted after the parameter calculator program had begun before the synthesizer-driven program started because the fundamental frequency values are not deposited in the buffer until a stress or punctuation mark is encountered. After this starting period, however, the program will run continuous and produce speech in real time.

VI. Discussion

The speech synthesis system has been programmed on a PDP-8 computer. The program occupies about 1000 words and the lookup table about 500. The parameter buffer is six words long for each 10 ms of speech. Thus a complete system of program, lookup table, and a buffer holding sufficient data to generate 3.4 s of speech can be obtained in a single field (4000 words) of computer memory. The storage requirements of the system are thus extremely modest.

There is no doubt that the quality of the speech generated could be improved by using more elaborate rules, especially for the rhythm and intonation and for some of the complex consonants. No provision has yet been made for co-articulation [10] or vowel reduction [11], but development is continuing.

Although the system produces intelligible speech in most instances, it generates more natural-sounding speech with some sentences than it does with others. This is partly due to the placing of stress marks in their optimum positions, and partly because some phonemes lose their identity in some contexts. The intelligibility of each phoneme in every possible context is being examined, and the poor ones are being improved by altering the values in the lookup table. It is probable, however, that there will remain a small residue of bad cases that can only be eliminated by employing more complex rules.

VII. Conclusion

A speech synthesis system has been described that meets many of the requirements of the ideal system. It has a large potential vocabulary, and produces a continuous stream of speech after a very short delay. The input data can be stored about as economically as data representing written language, and the system can be programmed on a small, inexpensive computer. The speech produced is not completely natural sounding or intelligible, but is sufficiently so for many purposes.

References

[1] H. Dudley, "Remaking speech," *J. Acoust. Soc. Amer.*, vol. 11, p. 169, 1939.
[2] W. F. Meeker and A. L. Nelson, "Vocoder evaluation research," Air Force Cambridge Res. Lab., Cambridge, Mass., Rep. AFCRL-64-46, pp. 1–58, 1964.
[3] R. H. Buron, "Generation of a 1000-word vocabulary for a pulse-excited vocoder operating as an audio response unit," *IEEE Trans. Audio. Electroacoust.*, vol. AU-16, pp. 21–25, Mar. 1968.
[4] W. Lawrence, "The synthesis of speech from signals which have a low information rate," in *Proc. Symp. Applications of Communications Theory*, W. Jackson, Ed., 1953, pp. 460–467.
[5] L. G. Stead and R. C. Weston, "Sampling and quantizing the parameters of a formant-tracking vocoder system," presented at the Speech Communication Seminar, Stockholm, Sweden, 1962, Paper G9.
[6] J. N. Holmes, I. G. Mattingly, and J. N. Shearne, "Speech synthesis by rule," *Language and Speech*, vol. 7, pp. 127–143, 1964.
[7] J. L. Kelly and L. J. Gerstman, "An artificial talker driven from a phonetic input," *J. Acoust. Soc. Amer.*, vol. 33, p. 835, 1961.
[8] D. Abercrombie, *Studies in Phonetics and Linguistics*. London: Oxford Univ. Press, 1965.
[9] W. A. Ainsworth and J. B. Millar, "A simple time-sharing system for speech perception experiments," *Behavior Res. Methods and Instrument.*, vol. 3, pp. 21–24, 1971.
[10] S. E. G. Ohman, "Co-articulation in VCV utterances: Spectrographic measurements," *J. Acoust. Soc. Amer.*, vol. 39, pp. 151–168, 1966.
[11] B. Lindblom, "Spectrographic study of vowel reduction," *J. Acoust. Soc. Amer.*, vol. 35, pp. 1773–1781, 1963.

A Flexible Active-Phase Equalizer with Application to Maximally Flat Time-Delay Filters

F. RUSSO and L. VERRAZZANI

***Abstract*—An active *RC* all-pass network has been proposed in which the regulation of only one component allows independent adjustment of each parameter affecting the phase characteristic. This flexibility is very suitable for time-delay equalization. Its use in maximally flat time-delay lines is shown, and tables of normalized components are provided to simplify the design work.**

Manuscript received May 20, 1972. This work was performed as part of the research program of the Centro Studi per i Metodi e i Dispositivi di Radiotrasmissione del Consiglio Nazionale delle Ricerche, Italy.

The authors are with the Dipartimento di Elettrotecnica ed Elettronica dell'Università di Pisa, Pisa, Italy.

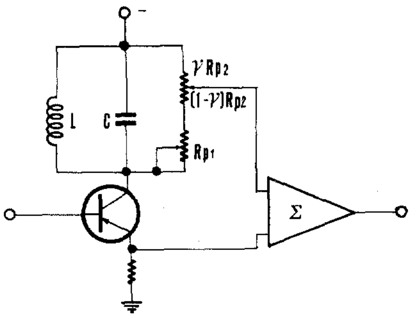


Fig. 1. Schematic diagram of the all-pass network suggested by the authors in a previous paper.

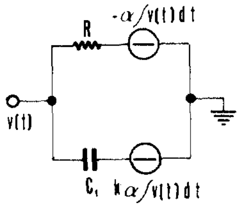


Fig. 2. Logical scheme of an *RLC* one-port simulator

Introduction

All-pass networks are generally used in delay lines and phase equalizers. Equalization can be done both at carrier frequency, as in high-capacity FM radio links, and at baseband, as in videotelephone and high-speed data transmission apparatus using telephone channels where a phase-frequency nonlinear characteristic causes intersymbol interference. It is well known that in the all-pass lattice networks there are some remarkable practical difficulties, whether for impedance matching, proper setup, or because of changes in the amplitude response produced by reactive lossy components.

Therefore, interest has been taken in active equalizers, since, besides removing matching problems, they reduce the number of reactive components that are not necessarily lossless.

Moreover, it is of considerable practical interest to have at one's disposal flexible active all-pass networks in which the regulation of only one component allows independent adjustment of each parameter affecting the phase characteristic. In this correspondence an active *RLC* network [1] fulfilling these statements is recalled, and an *RC* arrangement is suggested as being suitable for low-frequency applications. The use of this circuit in performing maximally flat time-delay lines is shown, and tables of normalized component values are provided to simplify the design work.

A New Active All-Pass Network Configuration

There are several configurations for realizing an active all-pass filter. The one described in this paper is a suitable improvement of an all-pass amplifier suggested by the authors in a previous paper [1] and schematically redrawn in Fig. 1.

As described in detail in [1], the proper setup of this circuit is performed with the following easy procedure.

1) The resonant circuit is tuned by regulating C so that the collector voltage is phase reversed with respect to the emitter.

2) The all-pass state is obtained by regulating R_{p2} so that at the tuning frequency, the collector voltage is twice that of the emitter.

3) The Q factor ($Q=R_p/\omega_0L$) is adjusted by regulating R_{p1} so that at the -3 dB frequencies, the phase of the collector voltage is $\pi\pm\pi/4$ with respect to the emitter.

Unfortunately, the network of Fig. 1 is not suitable for low-frequency applications where *RC* active filters have advantages in size, weight, and cost.

Therefore, it is of practical interest to replace the *RLC* circuit with an equivalent *RC* active one-port, so that the same flexibility is kept.

This can be done, bearing in mind that the current absorbed by the *RLC* one-port is composed of three components: the first is proportional to the input voltage, the second to its integral, and the third to its derivative. With reference to Fig. 2, if a voltage generator proportional to the sign-reversed integral of the input voltage is available, an inductive current and a resistive current are generated by connecting this generator to the input terminal with a resistor.