DIGITAL SIMULATION IN RESEARCH ON HUMAN COMMUNICATION

BY

EDWARD E. DAVID, JR.

Reprinted from the PROCEEDINGS OF THE IRE
VOL. 49, NO. 1, JANUARY, 1961

PRINTED IN THE U.S.A.

ERRATA:

Figure 2 should be Figure 8B
Figure 8A should be Figure 2
Figure 8B should be Figure 8A

Digital Simulation in Research on Human Communication*

EDWARD E. DAVID, JR.†

Summary—**Digital simulation is a powerful tool in uncovering the basic properties of new or proposed communications principles, particularly those involving coding of visual or auditory information. Operating on digitalized speech or pictorial signals, a stored program computer can perform processing equivalent to any coding. The output signals so produced can then be made available for subjective evaluation, thereby removing the necessity for premature instrumentation to produce samples for viewing or listening. This technique owes its efficacy to 1) the availability of computers fast enough to accomplish the processing in a reasonable time scale, 2) the existence of high quality translators to implement the flow of continuous signals in and out of the computer, and 3) the creation of compiling programs which allow uninitiated investigators almost immediate access to computer facilities, and which keep programming effort low. Simulation is assuming an increasing role in communications research.**

INTRODUCTION

CHEFS, wine tasters, painters, and musicians have long paid tribute to the fine discriminating powers of the human senses. It is an axiom among these artists that sensory appeal is the criterion of success. Engineers, too, have found that subjective factors play a key role in the evaluation of communication systems transmitting visual or auditory information. The physically measurable properties of such a system, such as its bandwidth and signal-to-noise ratio, are often helpful in making gross judgments, but with few exceptions the final test involves listening or viewing the received signal. Only if physical fidelity criteria are based upon subjective measurement can they substitute for the ear or eye. Indeed, it is the perceptual significance of transmission distortions which is the crucial factor in determining the merit of a communication system.

Therefore, it is not unusual for engineers to carry out intelligibility and quality tests on the systems they create, and to rely on the results to specify their performance. To perform these tests, signal samples must be generated to serve as material for subjective evaluation.

Traditionally, system models are built to generate these samples. In some instances, though, the high costs associated with today's instrumentation preclude building either the system or even a simplified version incorporating its basic principles. This constraint often prevents experimental evaluation of speculative communication concepts. In this and other cases, it may pay to depart from tradition and adopt simulation techniques. The digital computer is an apt tool for simulation. Such a facility can be arranged to accept appropriate inputs and to generate output samples for listening or viewing.

By this means, a stored-program computer, whose function can be changed from the simulation of one model to another by altering the instructions, can greatly facilitate evaluation of communication principles and systems. In addition, this type of simulation has the desirable property of separating defects of principle from those associated with a particular instrumentation.

This technique is useful in another context. In this age of advanced communication, transmission systems are tailored to a relevant class of signals. Signal properties determine design and coding procedures. In establishing these properties, digital computers can analyze signals, thereby serving as specialized measurement equipment. Computers can also generate visual or auditory signals directly from a program without any other input. Here the computer acts as a signal source.

Several investigations using simulation and involving pictorial material and speech have been carried out at the Bell Laboratories. Principally, these investigations concern codings aimed at conserving transmission channel capacity. Closely related are investigations aimed both toward automatic recognition of perceptual attributes of signals, and psychological investigation of perceptual mechanisms. This paper describes briefly some of these studies and the techniques developed for attaining realistic, workable simulations.

INPUT-OUTPUT TRANSLATOR

Since computers deal only with numerical data, one prerequisite to digital processing of speech and pictorial signals is a device to translate between analog and digital domains. Such translators commonly utilize a pulse-code (digital) representation.[1–5] The sampling

* Received by the IRE, July 29, 1960.
† Bell Telephone Labs., Inc., Murray Hill, N. J.

[1] G. L. Schultz, "The use of the IBM-704 in the simulation of speech-recognition systems," *Proc. EJCC*, Washington, D. C., pp. 214–218; December 9–13, 1957.
[2] E. E. David, Jr., M. V. Mathews, and H. S. McDonald, "Description and results of experiments with speech using digital computer simulation," *Proc. NEC*, Chicago, Ill., pp. 766–775; October 13–15, 1958.
[3] E. E. David, Jr., M. V. Mathews, and H. S. McDonald, "A high-speed data translator for computer simulation of speech and television devices," *Proc. WJCC*, San Francisco, Calif., 169–175; March 3–5, 1959.
[4] J. W. Forgie and C. D. Forgie, "Results obtained from a vowel recognition computer program," *J. Acoust. Soc. Am.*, vol. 31, pp. 1480–1489; November, 1959.
[5] J. W. Forgie and G. W. Hughes, "A real-time speech input system for a digital computer," *J. Acoust. Soc. Am.*, vol. 30, p. 668A; July, 1958.

theorem[6] prescribes that input analog signals must be time-sampled at a rate equal to or greater than twice their highest frequency component. Each sample is then coded into a number, usually binary, representing its amplitude to the nearest integer. The resulting succession of numbers might then be introduced directly into a computer except for several complicating factors. Computer input circuits usually require that input data conform to a rigid format specifying data rate, error-check coding, and supervisory marks. Furthermore, inputs rarely accept single digits in sequence, but rather take several digits in parallel usually from a magnetic or punched paper tape or from cards. Finally, because the internal computer memory is of limited size, the data on the tape must be grouped into convenient size blocks, usually called "records." Thus, the translator must convert a continuity of pulse-code samples to discontinuous records. Of course, the translator must work from analog to digital and vice versa.

For example, the IBM-704 computer can take its input from a seven-track magnetic tape. Six of these tracks contain binary data, the other an error-check digit. In each track, there are 200 bits/inch. Recording epochs occur simultaneously in all tracks. Each seven-bit line transverse to the long dimension of the tape constitutes one "character" (200 characters per inch). The number of characters to the record can be determined by the computer program, but the end of the record must be marked by a $\frac{3}{4}$-inch gap on the tape. The end of the data must be signified by a special mark, known as an "end-of-file."

All of these specifications are satisfied by the digital recordings made on the Bell Laboratories translator shown in Fig. 1. In the record mode the analog input signal feeds an analog-digital converter and is sampled at the command of an external sampling pulse. The converter's continuous digital output is grouped by the buffer storage into preset-length records which are then read onto magnetic tape by means of a fast start-stop transport. During playback, the translator reads a discontinuously-recorded tape through the buffer into a continuous stream of samples at a rate determined by an external sampling pulse.

By divorcing the input and output sampling times from the instantaneous recording and playback rates on the tape, the arrangement can accommodate a range of analog bandwidths (proportional to sampling rate) while maintaining the 200 character per inch density on the tape. In addition, the fluctuations in character spacing resulting from "jitter and wow" in the low-inertia digital tape drive are removed during playback so that they do not affect the analog output. The translator can operate at sampling rates as high as 10,000 samples per second with each sample quantized into 2048 levels. (Alternatively, it can supply 20,000 64-level samples per second.) This capability is sufficient to handle signals with a bandwidth of 5000 cps and a signal-to-noise ratio of 60 db. This number of quantizing levels insures that quantizing degradation is negligible compared to effects being studied in the simulation.

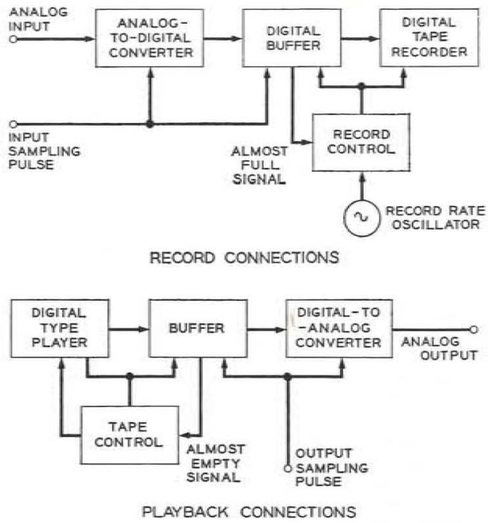


Fig. 1—Block diagram of data translator for computer input-output.

Transducing pictorial material into the computer requires some apparatus in addition to the translator.[7] For this purpose, the translator input is provided by a flying-spot scanner which covers about 1/25 the area of a full television picture in 2.4 seconds. The picture format is then an array of 100×100 picture points. During playback from a digital tape, the picture is displayed on a long-persistence screen, or a 35-mm photograph is made from a short-persistence phosphor. A picture of this size and resolution is sufficient to show those distortions which would be noticeable in a full size picture. Fig. 2 shows a photograph after the material has passed through the computer chain.

There are other pieces of auxiliary equipment which supplement the translator. One is a time-multiplexer which samples up to 24 input signals and presents the samples sequentially to the converter. Alternatively, it can distribute samples from the converter to as many as 20 output channels. This device has been used both to take spectral information from a bank of contiguous band-pass filters and to control a speech synthesizer[8] from computed signals.

[6] B. M. Oliver, J. R. Pierce, and C. E. Shannon, "The philosophy of PCM," Proc. IRE, vol. 36, pp. 1324–1331; November, 1948.

[7] R. E. Graham and J. L. Kelly, Jr., "A computer simulation chain for research on picture coding," 1958 IRE WESCON Convention Record, pt. 4, pp. 41–46.

[8] E. E. David, Jr., "Artificial auditory recognition in telephony," *IBM J. Res. and Dev.*, vol. 2, pp. 294–309; October, 1958.

Fig. 2—Picture consisting of 100×100 points after passing through translator.

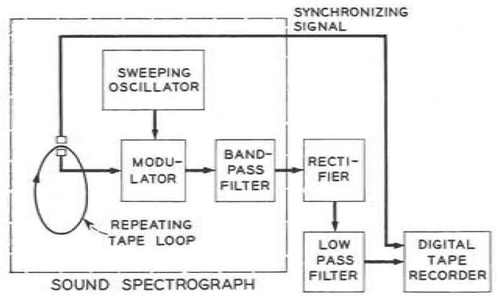


Fig. 3—Spectrograph computer input.

Another useful accessory is the spectrograph[8] input shown in Fig. 3. Here a heterodyne sound analyzer has been adapted for computer input. A repeatedly-scanned loop of tape is successively analyzed at different frequencies by a modulator, oscillator, and band-pass filter. The filter output is rectified, smoothed, and introduced into the converter along with a synchronizing signal which marks the beginning of the loop. Spectral inputs are sometimes preferable to computing the spectrum from the time waveform. Such computations can be overly time consuming.

Since the translator's record and playback sampling rates are independent, any signal may be reproduced slowly enough so that it can be recorded by a pen oscillograph. The translator can in addition supply a timing track which specifies the sample number of each output sample. Thus, the data may be subjected to detailed visual analysis.

These "peripheral" devices, along with other special apparatus built for particular tasks, greatly increase the versatility of the translator.

Sampled-Data Approximations to Continuous Systems

The translator described in the preceding section supplies a quantized and sampled computer input. Thus, any simulated communication system must necessarily operate on a sampled signal. This fact is no constraint on the simulation of systems which themselves process sampled data. The operations incorporated in such systems can be duplicated exactly with a computer. However, this statement is not generally true for systems operating on continuous signals. Fortunately, there are sampled-data near equivalents to continuous systems. Consequently, one important facet of simulation involves formulating such approximations.

Many operations, such as addition, give equivalent results whether applied to continuous signals or their sampled versions. Other operations must be suitably tailored for digital simulation. An example is linear filtering.

Filtering in a continuous system may be expressed by the convolution integral which relates the filter output waveform, $o(t)$, to the input waveform, $i(t)$, and the filter impulse response, $h(t)$. Since in the simulation both input and output must consist of samples, the impulse response, too, must be sampled at the same rate, thereby reducing the convolution integral to a summation. Many impulse responses are not bandlimited and, therefore, cannot be represented in sampled form without error. However, the effect of this error in the convolution computation can be specified. Let $h(t)$ be the impulse response, $H(j\omega)$ its Fourier transform, and W_s the sampli n rate of the input; then the mean-square error in the output band $o \leq \omega \leq W_{s/2}$ is

$$\sum = 2\int_{W_{s/2}}^{\infty} |H(j\omega)I_s(j\omega)|^2 d\omega$$

where $I_s(j\omega)$ is the spectrum of the sampled input. Note that this error is zero if $H(j\omega)$ is limited to the band $o \leq \omega \leq W_{s/2}$. If not, one means for restricting the error substitutes for $H(j\omega)$ another response $H'(j\omega)$ which is nearly equal to $H(j\omega)$ in the interval $o \leq \omega \leq W_{s/2}$, but is smaller outside. For instance, H' might be formed by adding several low-pass sections to H. The procedure carries the disadvantage that it may increase the duration of the impulse response. Another method is to increase the sampling rate until the error falls within the required bound. In this case, also, the amount of data which must be handled is correspondingly increased. However, often the input signal is only nominally bandlimited to W cps and so is sampled at, perhaps, $4W$ times per second. This rate is high enough to accommodate many filters of interest. Nonetheless, care must be taken to avoid the cumulative effects of sampling error in simulating multiblock systems.

Once a sampled-date filter approximation is established, it can be simulated in two ways. One is the "transversal filter," a tapped delay line whose outputs are weighted and summed according to

$$o(t_i) = \sum_{j=-\infty}^{t_i} h'(t_i - t_j)i(t_j)$$

where $o(t_i)$ and $i(t_i)$ are the sampled input and output, and $h'(t_i)$ is the sampled impulse response. Such a filter is shown in Fig. 4(a). One difficulty here is that the impulse response must be truncated to some finite number of values, requiring an additional approximation. In the second method, this is avoided by expressing each

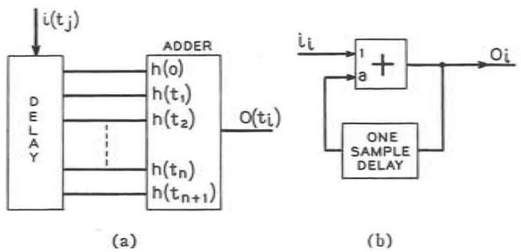


Fig. 4—Two methods of filter simulation.
(a) Transversal filter. (b) Accumulator.

sample as a linear difference equation involving the input and previous outputs. The coefficients of this equation must be determined from the desired filter characteristic. An example of this realization for a "leaky" accumulator is shown in Fig. 4(b). The governing difference equation is

$$o_{t_i} = i_{t_i} + a o_{t_{i-1}},$$

where a is a constant less than 1 specifying the rate of decay of the impulse response. Since the quantities making up this equation are quantized (expressed as a finite number of digits), the indicated computation must be carried out with sufficient accuracy to keep the round-off error within bound. For instance, if a were almost 1, round-off errors could cause to output to diverge, yielding an unstable system.

From this discussion, perhaps the reader might guess that the engineer, rather than the professional programmer, is better suited to formulate such problems for simulation. Indeed, this impression has been confirmed in much of our work. Actually, only the use of engineering and programming skills in combination can avoid studies which are either computationally or physically nonsensical.

Computer Requirements and Estimating Running Time

Simulation of communication systems requires processing of large amounts of data. For instance, ten seconds of speech may be represented by 10^5 samples or about 10^6 bits, and all of these samples must be processed. This situation demands a fast computer with a considerable memory capacity if the processing is to be carried out expeditiously. The IBM-704 is such a machine. Its 32,000-word, random-access memory is organized into 36-bit words. Each elementary operation (addition, for instance) requires 12 μsec, while a complex one such as multiplication takes about 240 μsec. Typically, there may be 20 to 100 operations per sample point in a simulation. Assuming an average time of, say, 50 μsec per operation, it takes between 10 and 50 times as long to process a speech sample as it does to speak it. Such time scales are feasible from the economic point of view. Scales of 500–1000 to 1 are usually prohibitively expensive.

Of course, the exact time scale depends upon the complexity of the system to be simulated. The running time can usually be estimated accurately before writing the program by locating those operations which are most time consuming and which must be performed on every data sample.[9] The total number of these operations for each second of processed speech can be found by multiplying the number of signal samples per second by the number of operations per sample. The time scale is then the number of operations multiplied by the time per operation. For example, the number of multiplications per signal sample for a difference-equation filter simulation is one plus the number of poles and zeros in the analog filter characteristic. For a 4-pole filter operating on 10,000 signal samples per second, the time scale on the IBM-704 would be about 12/1.

The total computation time depends, of course, not only on the time scale but also on the duration of the speech sample. Preliminary listening evaluations can be obtained from as little as 10 to 15 seconds of processed speech. This time is sufficient for four or five short sentences each by a different speaker. Such an amount of data permits qualitative judgments of general fidelity and intelligibility. Formal quantitative measurements may require thirty seconds to a minute or more of material. Comparative evaluations of different systems, or the same system with different parameters, is facilitated by the identical digital speech samples which can serve as inputs to all versions.

The computation time and storage requirements for single pictorials can be estimated from the component 10^4 picture points and the operation time per point. Such preliminary estimates of running times can prevent inappropriate uses of simulation, uses in which the results may not be commensurate with the time and effort expended.

Programming

Programming for simulating signal processing systems is not basically different from programming for arithmetic tasks with, perhaps, two exceptions. Simulation often requires a greater number of logical operations. In addition, the amount of processed data is typically very much greater than both the number of computer operations per data point and the computer memory capacity. Therefore, the lion's share of the programming effort lies in shifting data sequentially from one location to another in the machine's memory—a sort of tedious bookkeeping discouraging to all except the most dedicated, and conducive to a high error rate for all except the most meticulous. The problem is most severe when programming in basic machine language. Here, each operation which the machine is to execute must be individually specified. This exacting requirement implies that basic is the most general machine language. Using it, the expert programmer can tailor the machine's operation precisely to the task at hand, thereby insuring an efficient computation.

Other programming languages, less demanding but also less efficient than basic, have been devised. They permit the programmer to utilize instructions each of which may specify many machine operations. Typically, these instructions are coded in mnemonic form to minimize errors and to facilitate rapid transcription. A program written in these terms is referred to as a "source" program. A compiling program or "compiler" translates the source program into an "object" program in basic machine language. The object program can then be used to process the data in the desired fashion. Compilers do not usually produce a basic program as efficient as an expert programmer might using ingenious short cuts in basic language. The programming economy achieved, however, often offsets this factor. In addition, a compiler catalyzes access to computational facilities. The aspiring user may require as little as a few hours training to use one effectively.

Fortran, the IBM algebraic compiler, has proved useful for certain speech-recognition studies.[10] A compiler conceived especially for signal processing (using the IBM-704) has been written by J. L. Kelly, Jr., C. C. Lochbaum, and V. A. Vyssotsky.[11] A source program in this language is a sampled-data block diagram description. For this reason, the compiler is called BLODI (BLOck DIagram). Its use is best described by an example. The feedback system in Fig. 5(a) is redrawn in a form compatible with BLODI in Fig. 5(b). Input and output boxes have been added and each block has been labeled with an arbitrary letter. In general, any processing system must be converted to a sampled-data equivalent before BLODI can be used. The source program consists of a three-column listing of the blocks. Column 1 contains the block label, Column 2 is a mnemonic specifying the block function, and Column 3 lists the block parameters and the destinations of the block's "output" connections. The order of the listing is immaterial. A BLODI program for Fig. 5(b) might begin with the input:

A INP B/1

The input block is designated A, its function is denoted by INP, and its output goes to input 1 of block B. The code for the subtractor, block B, is

B SUB C.

The quantizer is assumed to have four representative (output) levels and three input decision (transition) levels [see Fig. 5(a)]. These must be listed in Column 3

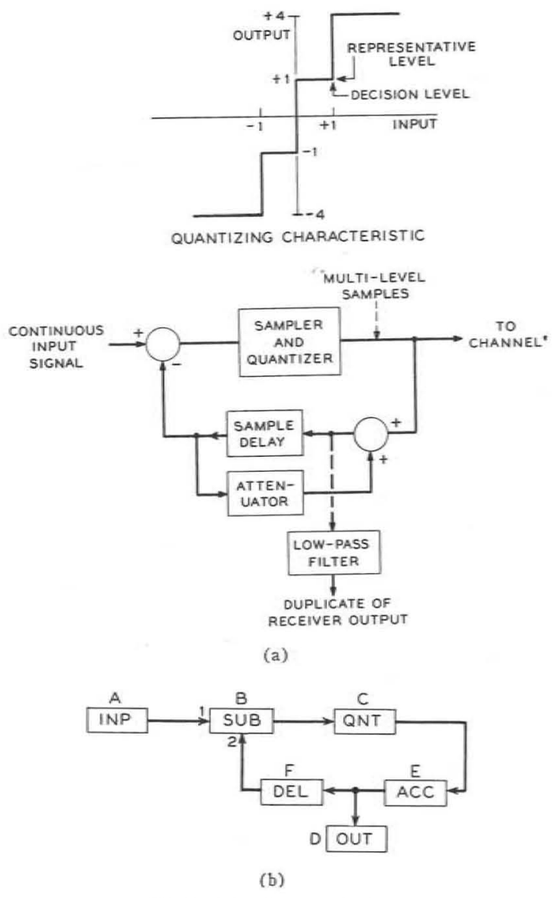


Fig. 5—(a) A differential coder.
(b) Equivalent BLODI diagram.

beginning with the lowest representative level and progressing alternately through all the decision and representative levels. For the quantizing characteristic of Fig. 5,

C QNT −4, −1, −1, 0, +1, +1, +4, E.

Block E is an accumulator with a decay constant, a, and is coded

E ACC a, D, F.

Block F is a delay line with 1 sample delay,

F DEL 1, B/2,

and finally the output is specified,

D OUT −.

These six instructions constitute the source program. From it the compiler prepares a longer object program in which all of the detailed operations and data shuffling

[9] M. V. Mathews, "The effective use of digital simulation for speech processing," *Proc. Sem. on Speech Compressing and Processing*, AF Cambridge Res. Ctr., vol. 1; September, 1959.

[10] P. B. Denes and M. V. Mathews, "Spoken digit recognition using time-frequency pattern matching," *J. Acoust. Soc. Am.*, vol. 32, pp. 1450–1455; November, 1960.

[11] J. L. Kelly, C. C. Lochbaum, and V. A. Vyssotsky, "A block diagram compiler," *Bell Sys. Tech. J.*, in press.

are automatically programmed.

The internals of BLODI are too extensive to recount here. Clearly, it has available or can compile[12] programs to perform on the signal samples each of the operations specified in Column 2 of the source program. In addition, the compiler establishes the sequence of block operations so that only one output sample from each block need be retained in the computer memory at any time.

BLODI programs generally approach the efficiency of basic language programs, though in some instances they may be only about half as fast. The compiler encompasses a large number of block functions including noise generators, peak and low-level clippers, amplifiers, transversal and difference-equation filters, etc. Other functions can be added by the programmer at will. As many as 800 simple blocks in a single diagram can be compiled. An engineer can learn the formalities of BLODI programming in about two hours. Thus, it affords almost immediate access to a powerful tool for communication research.

To aid in using a computer as a signal generator, Mathews has written a compiler (for the IBM-704) which will produce on command quasi-periodic functions of time of arbitrary waveshape.[13] The basic unit of this compiler is a canonical function generator as shown in Fig. 6. The waveshape is specified by the

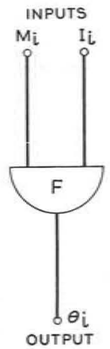


Fig. 6—Canonical generator used in Music Compiler.

function $F(X)$. Its samples, taken at 512 equispaced points (X values), are stored in the computer memory. In generating a wave, samples from this description are printed repetitively on the output tape. The repetition rate is specified by the "scan" parameter, I_i. M_i is a variable scale factor which is used to specify the attack, decay, duration, and location of the wave on the output tape. The ith output sample, θ_i, is related to these parameters by

$$\theta_i = M_i F([X_i] \text{ modulo } 512].$$

Values of X_s are generated according to

$$X_{i+1} = X_i + I_i.$$

Thus X increases progressively in steps of size I_i until it reaches the value 512, then it returns to zero (modulo 512). Thus, if I_i is constant, F is in effect scanned by a linear sweep; each period generated will consist of $512/I$ samples, which at a 10 kc output rate corresponds to $512/I \times 10^{-4}$ seconds.

The output of a basic generator may be printed directly on the computer output tape, or it may supply the input to other unit generators. In this way, groups of generators may be used to form more complex units.

A source program for this compiler begins with a definition of all the unit generators in terms of their F functions and their interconnections. The subsequent portion specifies the intensities, time origins, and repetition frequencies of each waveform segment. This program is well suited for producing musical sounds, and is known as the Music Compiler. It has been used, in addition, for speech production and for the generation of low-frequency control signals.

Compilers, in effect, adapt a general-purpose computer for the solution of particular problem classes. Once such a problem class is defined, it is usually possible to formulate a compiling language which will facilitate programming as well as produce reasonably efficient object programs. Where large numbers of generically similar problems must be solved, such a compiler is practically a necessity.

Studies Utilizing Digital Simulation[14]

Digital simulation has been used for a wide variety of investigations, but perhaps the most interesting concern digital coding of information-bearing signals and duplication of human recognition functions. Also, the computer has been used for generating precisely controlled psychological stimuli. Examples of these studies will be presented in this section.

Digital Coding of Information-Bearing Signals

Digital, or discrete, signals can be transmitted through noisy channels with little or no impairment since such signals can be regenerated exactly, provided the signal-to-noise ratio never falls below a certain threshold value.[6] Information-bearing signals such as speech or the output from a television camera are continuous functions of time. Hence, certain transmission situations require that these be digitally coded. The method most often employed is pulse-code-modulation (PCM). Here, instantaneous amplitude samples of the continuous signal are each represented by a number of binary digits. In order to insure adequate reproduction of the audio or video, the number of digits must often be inconveniently large. This situation arises because PCM codes each sample independently. Thus, novel coding methods tailored to take advantage of dependencies between samples may permit economies.

An important example is the feedback coder in which the difference between the continuous input signal and a replica of the received signal delayed by one sample time is converted into a digital code. A typical coder is shown in Fig. 5(a). The input is bandlimited to W cps and is compared with a reference signal by subtraction. The resulting difference signal is sampled and quantized to produce the discrete code for transmission. The feedback loop accumulates the quantized samples, delays them one sample time, and provides the reference signal. The receiver is an identical accumulator plus a low-pass filter. Thus, the original signal is represented in terms of the difference between its present value and the *previously received* signal value. The terms "delta modulation"[15] and "differential quantization"[16] have been applied to this process. Digital simulation has greatly facilitated comparisons of signals so coded with identical signals pulse-code modulated.

Programming the computer to simulate a differential coder can be done with dispatch using the BLODI compiler. The resulting object program will process a picture in the 100×100 format in about 30 seconds. For speech, the time scale is about 20/1 for those cases in which the computer input sampling rate is sufficiently high. In some delta-modulation systems, the sampling rate may be an important variable. One example is the simple "one-digit" system. Here, the quantizer has only two representative levels; one calling for an increase in the receiver output, the other calling for a decrease. Since the output can change only one "step" per sample, rates much higher than $2W$ may be required to achieve close correspondence between the input and the receiver output. In such cases, the maximum sampling rate afforded by the input-output translator may be insufficient. If so, additional samples may be interpolated by the computer itself. Alternatively, the translator input may be introduced in reduced time scale, achieved, for instance, by a reduced-speed playback of a recording.[2] The effective sampling rate is, thereby, multiplied by the reduction factor. Of course, both methods increase the computing time scale.

The results of this differential coding study illustrate the scope which can be attained with the aid of simulation.[16] Differential quantization has a definite advantage over PCM for transmission of pictorial material. Using the tapered eight-level quantizer (Fig. 7) in the differential coder, pictures such as that in Fig. 8(a) are

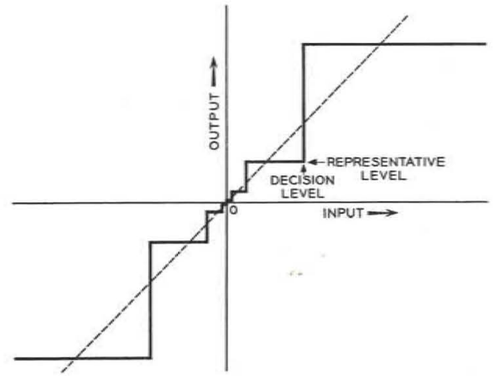


Fig. 7—Quantizing staircase used in differential coding of pictures.

(a) (b)

Fig. 8—Comparison of picture codings. (a) 3-bits/picture point, differential quantizing. (b) 3-bits/picture point, pulse-code modulation.

achieved. Compared to the original, Fig. 2, little difference may be noted except for a slightly ragged contour along the left edge of the face. Since this version of delta-modulation requires three binary digits per picture point, a three-digit PCM representation requires the same total number of pulses per second for transmission. Note the objectionable contouring in the picture so coded, Fig. 8(b).

An analysis of this differential coder indicates that the error (difference between the original and the reproduced picture) is concentrated in areas where the brightness is changing drastically from point-to-point; that is, in areas of fine detail. In regions of constant brightness, the picture is rendered quite faithfully. Subjectively, this distribution of the error is much preferable to the more uniform distribution typical of PCM.

Several refinements of differential coding have been investigated.[16] One makes use of two-dimensional rather than one-dimension differences. It can use as the feedback reference signal either the receiver output delayed by one sample or by one scanning line. A logical test on past picture points determines which interpolation mode gives the best fit at each point of the picture. When coupled with a quantizing staircase which varies according to the instantaneous value of the reference

[12] The term "compile" as used here refers to the assembling of a sequence of computer instructions to perform a given function.

[13] M. V. Mathews, "An acoustic compiler for music and psychological stimuli," *Bell Sys. Tech. J.*, in press.

[14] Studies described in this section were carried out on the IBM-704 computer.

[15] F. de Jager, "Deltamodulation," *Philips Res. Repts.*, vol. 7, pp. 442–466; 1952.

[16] R. E. Graham, "Predictive quantizing of television signals," 1958 WESCON Convention Record, pt. 4, pp. 147–157.

signal, this scheme considerably improves the reproduced picture.

Another method of digital encoding represents the continuous signal in terms of its extremes (maximums and minimums) and the time between them. The value of such a code for speech and picture transmission has been studied by stimulation. While the results are of interest in themselves, they will not be reviewed here. Interested readers are invited to refer to the appropriate publications.[17,18] Suffice it to say, the simulation technique enables an economic study of this coding, which, while conceptually and computationally simple, would have called for extensive special-purpose equipment.

Recognition of Signal Properties

Many different physical events are regarded as equivalent by the human observer. For instance, specimens of the word "one" as spoken by ten different people will evidence gross acoustical disparities; yet a listener can recognize all versions as symbolizing the intended numeral. Such categorization depends upon abstract properties of the acoustic event, and is closely related to the problems of character recognition, and reduced bandwidth transmission of continuous signals.[4] Paradigms aimed toward machine duplication of human recognition behavior usually involve a group of measurements on the event to be categorized. These are then subjected to appropriate decision logic.

Even such simple distinctions as speech vs silence involve elements of recognition since the noise energy in "silence" is often greater than in some speech sounds.[8] Another significant distinction is that between classes of speech sounds. For instance, a person ordinarily uses two different mechanisms to generate the necessary acoustical energy for speaking. The vocal cords with their quasi-periodic vibrations supply the vowel-like sounds (*voiced* sounds). Turbulent airflow at a constriction appropriately formed in the vocal tract provides the random noise energy associated with sounds such as *s* and *sh* (*unvoiced* sounds). The distinction between these two modes of generation, as well as the identification of silent intervals, is fundamental to both speech bandwidth reduction and speech recognition. For these purposes, these distinctions must be made automatically by a device operating solely on the speech signal itself.

Systematic investigation of such devices using conventional equipment is tedious because observation of the nonrepetitive dynamics is difficult, and input conditions can be reproduced only with great care. Simulation alleviates both of these problems. Computer printouts and tabulations can provide an accurate log of the processed results, while the input speech can be reproduced precisely from a digital recording.

[17] M. V. Mathews, "Extremal coding for speech transmission," IRE Trans. on Information Theory, vol. IT-5, pp. 129–136; September, 1959.

[18] B. Julesz, "A method of coding television signals based on edge detection," *Bell Sys. Tech. J.*, vol. 38, pp. 1001–1020; July, 1959.

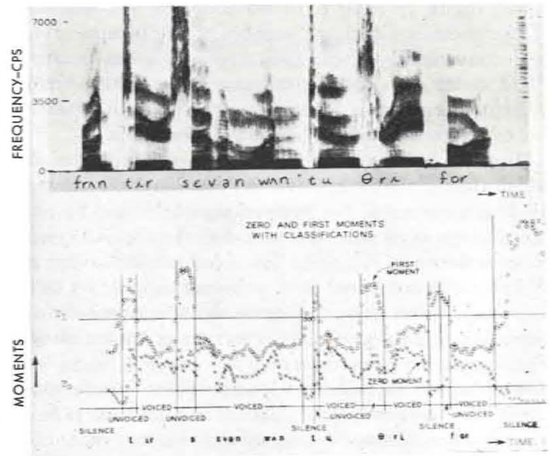


Fig. 9—Operation of voiced, unvoiced, silence recognition on speech spectra data

One investigation of voiced-unvoiced-silence discrimination utilizes measurements of the normalized zeroth and first moments of the speech wave followed by decision logic.[19] The computer input is taken from the sound spectograph (Fig. 3) which produces a running amplitude spectrum display as shown in Fig. 9. This display specifies the approximate rms value of the speech (indicated by increasing density of marking) at each time-frequency intercept.[8] As this spectral information is taken into the computer, it sums and groups the data into 70 cps by 7-msec blocks. The moments, M_0 and M_1, are computed by:

$$M_0 = \sum_i X_i \qquad M_1 = \frac{\sum_i iX_i}{\sum_i X_i},$$

where the X_i's are the rms speech amplitude values in each block, and the index i runs over the frequency axis. The average moments are computed by summing over the time index j and averaging

$$\overline{M_0} = \frac{1}{n}\sum_j M_0 \qquad \overline{M_1} = \frac{1}{n}\sum_j M_1$$

where the summations include only those n values of M which exceed an arbitrary threshold value. The normalized moments are:

$$M_0' = \frac{M_0}{\overline{M_0}} \qquad M_1' = M_1 - \overline{M_1'}$$

Values of M_0' and M_1' are plotted beneath the spectrographic display of Fig. 9. One simple decision logic

[19] C. C. Lochbaum, "Segmentation of speech into voiced sounds, unvoiced sounds, and silence," *J. Acoust. Soc. Am.*, vol. 32, p. 914; July, 1960.

examines each moment with respect to a threshold value. Separation of the speech into voiced, unvoiced, and silent intervals is accomplished according to Table I. There is in addition a continuity requirement, namely, intervals must be at least 28 milliseconds in duration, otherwise they are assigned to the same category as the adjacent left-hand segment. The decisions corresponding to this process are marked in Fig. 9. They define quite accurately the appropriate regions of the speech.

TABLE I

	M_0 Below T_0	M_0 Above T_0
M_1 Below T_1	SILENCE	VOICED
M_1 Above T_1	SILENCE	UNVOICED

Programming for this simulation was done in Fortran and the time scale was about 20/1. This study is typical of several speech recognition studies aided by simulation.[20] Others have, however, included spectral analyses of waveform inputs as well as operations directly on spectral input data.[21–23]

In the pictorial realm, the recognition approach has been used to remove additive noise selectively from a picture. This process depends upon analysis of local picture properties, and is called "noise-stripping."[24] Its basic properties are easily established by digital simulation. One basic scheme considers a 3×3 matrix of picture samples spaced at the Nyquist interval ($1/2W$ seconds) along the scanning (horizontal) direction and at the line spacing in the vertical. As shown in Fig. 10, the nine samples can be formed into six subgroups, designated by Roman numerals each of which represents the sum of its three sample values. The "flexure" of the region covered by the matrix is:

$$\Delta_x = \tfrac{1}{6}(\mathrm{I} + \mathrm{III} - 2\mathrm{II})$$

$$\Delta_y = \tfrac{1}{6}(\mathrm{IV} + \mathrm{VI} - 2\mathrm{V}),$$

where Δ_x and Δ_y are the second finite-differences of the picture brightness surface in the x and y directions. These quantities can be compared with a threshold value δ, which is typically a small fraction of the picture brightness range. The outcome of this comparison

[20] E. E. David, Jr., M. V. Mathews, and V. A. Vyssotsky, "Recognition of voicing, voice pitch, and formant frequencies with a digital computer," *Proc. Third Internatl. Congress on Acoustics*, Elsevier Publishing Co., Amsterdam, The Netherlands; 1960.

[21] M. V. Mathews, J. E. Miller, and E. E. David, Jr., "Monaural phase effects in speech perception," *Proc. Third Internatl. Congress on Acoustics*, Elsevier Publishing Co., Amsterdam, The Netherlands; 1960.

[22] J. E. Miller, M. V. Mathews, and E. E. David, Jr., "Pitch synchronous analysis of vowel sounds," *J. Acoust. Soc. Am.*, vol. 31, p. 1564A; September, 1959.

[23] J. E. Miller, M. V. Mathews, and E. E. David, Jr., "Pitch synchronous spectral representations of voiced speech," *J. Acoust. Soc. Am.*, vol. 32, p. 913; July, 1960.

[24] R. E. Graham, "A noise-stripping process for picture signals," presented at the SMPTE Conv., New York, N. Y.; October 7, 1959.

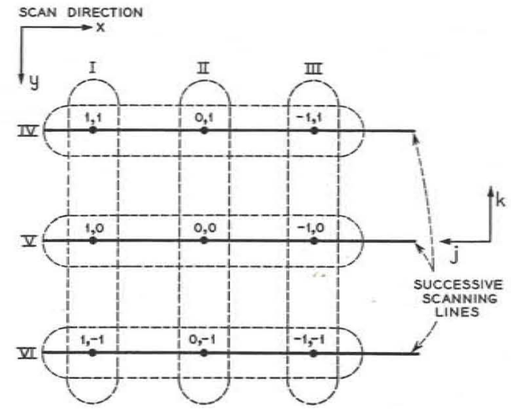


Fig. 10—Picture-point matrix for noise stripping studies.

specifies one of three smoothing modes to be applied to the point in the matrix center. Table II summarizes this process.

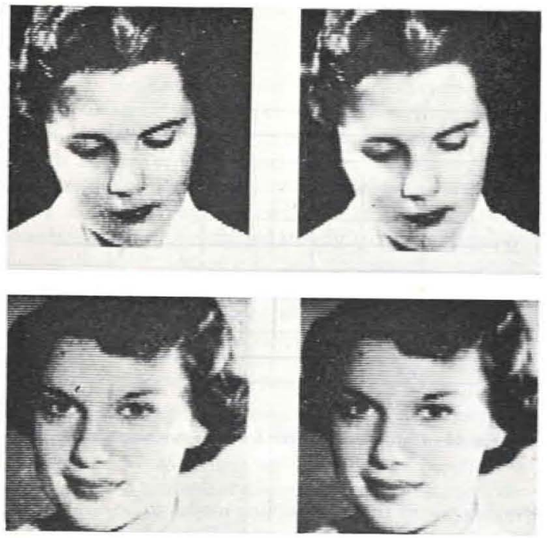
TABLE II

Comparison Result	Local Surface Property	Value Substituted for Center Point	Noise Reduction
$\Delta_x < \delta$ $\Delta_y < \delta$	Planar	$\frac{\mathrm{I} + \mathrm{II} + \mathrm{III}}{9}$	10 db
$\Delta_x > \delta$ $\Delta_y < \delta$	Flexure in x Direction	$\frac{\mathrm{II}}{3}$	5 db
$\Delta_x < \delta$ $\Delta_y > \delta$	Flexure in y Direction	$\frac{\mathrm{V}}{3}$	5 db
$\Delta_x > \delta$ $\Delta_y > \delta$	Flexure in Both Directions	Same as original	0 db

For a planar surface, two-dimensional low-pass filtering is used. For a surface flexed in one direction, low-pass filtering in the opposite direction is appropriate. No filtering is applied to a surface flexed in both directions. This and similar transformations on pictures accomplish considerable noise reduction. Typical results from a simulation study are shown in Fig. 11. Simulation is at its best in a study of this kind, a study which is computationally simple with many variations on the original theme to be examined. Each picture requires about 30 seconds of computer time for processing. Thus, many different versions can be run economically.

The Computer as a Signal Generator

Through the medium of the data translator, a computer can be used effectively as a signal generator. From appropriately written output tapes, the translator can produce visual and auditory signals for subjective con-



(a) (b)

Fig. 11—Typical results from simulation of noise stripping operation. (a) Unprocessed. (b) Processed.

sumption. As reviewed in an earlier section of this paper, the Music Compiler is an effective aid in writing such tapes when quasi-periodic, music-like sounds are desired. Another application is the generation of stimuli for psychological exploration of perceptual mechanisms.

Julesz found this technique quite effective in his investigation of depth perception in vision.[25] The human observer uses many factors, or cues, in his environment to deduce the relative and absolute distance of objects in his field of view. Among them are perspective, familiarity with the known size of objects, interposition,[26] monocular parallax,[27] and binocular parallax.[28] One significant distinction opposes the monocular cues with the binocular of which there is only one. In studying the latter, stimuli with no monocular cues and controlled binocular parallax are required. Of course, natural scenes contain all the cues in varying proportions. In his study of binocular parallax, Julesz solved this problem by having the computer generate his stimuli.

He programmed it to generate rasters of dots whose brightness was a random function of the dot position. In order to see a figure in depth, two such patterns must be presented; one to each eye. If the two are identical, the raster will appear flat when the two fields are fused to a single image in the brain of the viewer. If some sector of the dots in one pattern is shifted horizontally a small distance with respect to the same dots in the other, then the shifted sector will appear to be in a different plane from the background. The separation between this plane and the background depends upon the distance shifted. These computer-generated fields permit the amount of binocular parallax to be controlled accurately. Furthermore, other cues are absent. This convenience permits a systematic investigation of binocular information in depth perception. Stimuli of this type can be generated in less than 30 seconds computer time.

[25] B. Julesz, "Binocular depth perception of computer-generated patterns," *Bell Sys. Tech. J.*, vol. 39, p. 1125; September, 1960.
[26] Objects in front often obscure parts of objects behind.
[27] Near objects seem to move more rapidly than far objects when the observer himself is in motion.
[28] The two eyes see slightly different views of the world.

Concluding Remarks

Computers are proving to be flexible tools in research on human communication. They provide a ready means for uncovering the fundamental properties of any well-thought-out signal processing or coding scheme. This approach owes its efficacy, first, to the high-quality computer input-output translators now available for a range of signals, and, second, to high-speed digital computers which bring significant signal processing tasks within a reasonable time scale. Of equal significance is the creation of compiling programs. These are the "open sesame" to computers for any interested engineer. A well-written compiler requires little training to use effectively, and programming time is often negligible.

However, enthusiasms for simulation studies of communication systems must be tempered by a few caveats. Regardless of the rapid strides toward increased computer speeds and availability, the economics of simulations should not be taken for granted. Some problems will undoubtedly prove to be unsuitable. For instance, the study of apparent (subjective) motion in a sequence of rapidly projected visual images is a strain on present capabilities. Though suitable sequences of picture frames could be processed, the resulting time scale seems excessive. On the assumption of 25 frames per second of viewing time, and one minute of computer processing on each frame, the time scale would be 1500/1. Such an extensive problem should not be undertaken lightly.

A lack of intimacy is voiced by those engineers who like to "feel the knobs" on their experimental equipment. This objection is a very real one since in simulations it may be difficult to gain insight into the workings of a signal processing scheme. Often, a number of hours elapse between the adjustment of a processing parameter and the observation of the corresponding effect. Thus, the valuable interplay between the engineer and his equipment is difficult to establish. This situation promises to be remedied by the current trend toward real-time computer input-output. When generally accessible, this closer coupling between the computer and the engineer holds promise of a near-revolution in experimental communication research.

Problems sometimes arise because computationally simple operations may defy duplication outside the digital realm. If a coding scheme is shown by simulation to possess desirable properties, then construction of efficient and economic equipment incorporating that coding may be appropriate. If so, the computer program or the principles incorporated in it must be converted into a practical system. Unfortunately, not all programs yield easily to such metamorphosis. Thus, an additional nontrivial step beyond simulation may be required. On the other hand, many simulations concern basic principles drawn from analyses of an existing or proposed system. Here simulation results are often directly applicable without further elaboration.

The judicious use of computers to simulate systems and generate signals which can be evaluated subjectively is catalyzing many areas of research—areas previously hampered by the necessity for excessive amounts of instrumentation. Perceptual research in pattern recognition, signal coding, and sensory psychology are benefiting from a closer association of theory and experiment than heretofore feasible. At some distant time, the terminals of information transmission systems may well turn out to be special-purpose digital computers incorporating the principles established by experiments on general-purpose computers. The speed of digital circuitry within the available technology is increasing at a startling rate. We can confidently look forward to real-time digital processing of speech and pictorial signals.

Acknowledgment

In the preparation of this paper, the author has drawn freely on the work of his colleagues in the Visual and Acoustics Research Department of the Bell Telephone Laboratories. Only through their industry and ingenuity could the simulation techniques described above have come into being. In particular, the contributions of Miss J. E. Miller and Mrs. C. C. Lochbaum, as well as R. E. Graham, B. Julesz, J. L. Kelly, Jr., M. V. Mathews, H. S. McDonald, and V. A. Vyssotsky, are noteworthy. To these and the many other contributors, the author expresses his gratitude.