shown at the bottom in Fig. 28. They were quite successful for this vowel, which is fortunate, because the general technique is essentially identical to the synthesis strategies used in Klattalk and other phonemic rule programs to generate consonant–vowel–consonant formant patterns.

These data should be augmented for other vowels, but the analysis task is formidable, so only partial data are available on some vowels in symmetrical CVC contexts (Stevens and House, 1963). Similarly, the same kinds of studies should be performed for other speakers, and at several syllable durations and degrees of stress. Of particular interest are rules that modify vowels in sentence contexts depending on consonantal context, stress, and duration. From what little is known, it appears that the vowel space shrinks when one goes from words spoken in isolation to sentences (Fant *et al.*, 1974; Shearme and Holmes, 1961), but it is not clear whether vowels tend to neutralize toward schwa, or simply accede to articulatory demands of adjacent consonants (Lindblom, 1963). It is possible that some of the subjective impression of unnaturalness and "foreign dialect" of synthesis-by-rule systems can be attributed to insufficient attention to details of this sort, both known and those yet to be discovered.

The formant transitions for a CV syllable depend to some extent on the nature of the phonetic segment that precedes the consonant. Öhman (1966) has published data on formant motions for [b,d,g] in different VCV environments that demonstrate significant interactions (Fig. 29), and Martin and Bunnell (1982) have shown that listeners expect these coarticulatory shifts—subjects show reaction time deficits when the formant shifts are not present.

Text-to-speech systems have only begun to simulate the details of the phenomena noted in this section (Coker *et al.*, 1973; Klatt, 1976b). Klattalk now contains a separate subroutine for allophonic substitution rules, as well as many detailed parameter adjustment rules in the part of the program concerned with drawing parameter values for phonetic sequences. Taken in total, these rules characterize all of the allophonic variants and word-boundary cues described here, although the rules are simplified and generalized beyond the available data in such a way that they probably do not adequately represent the environments where the rule should be

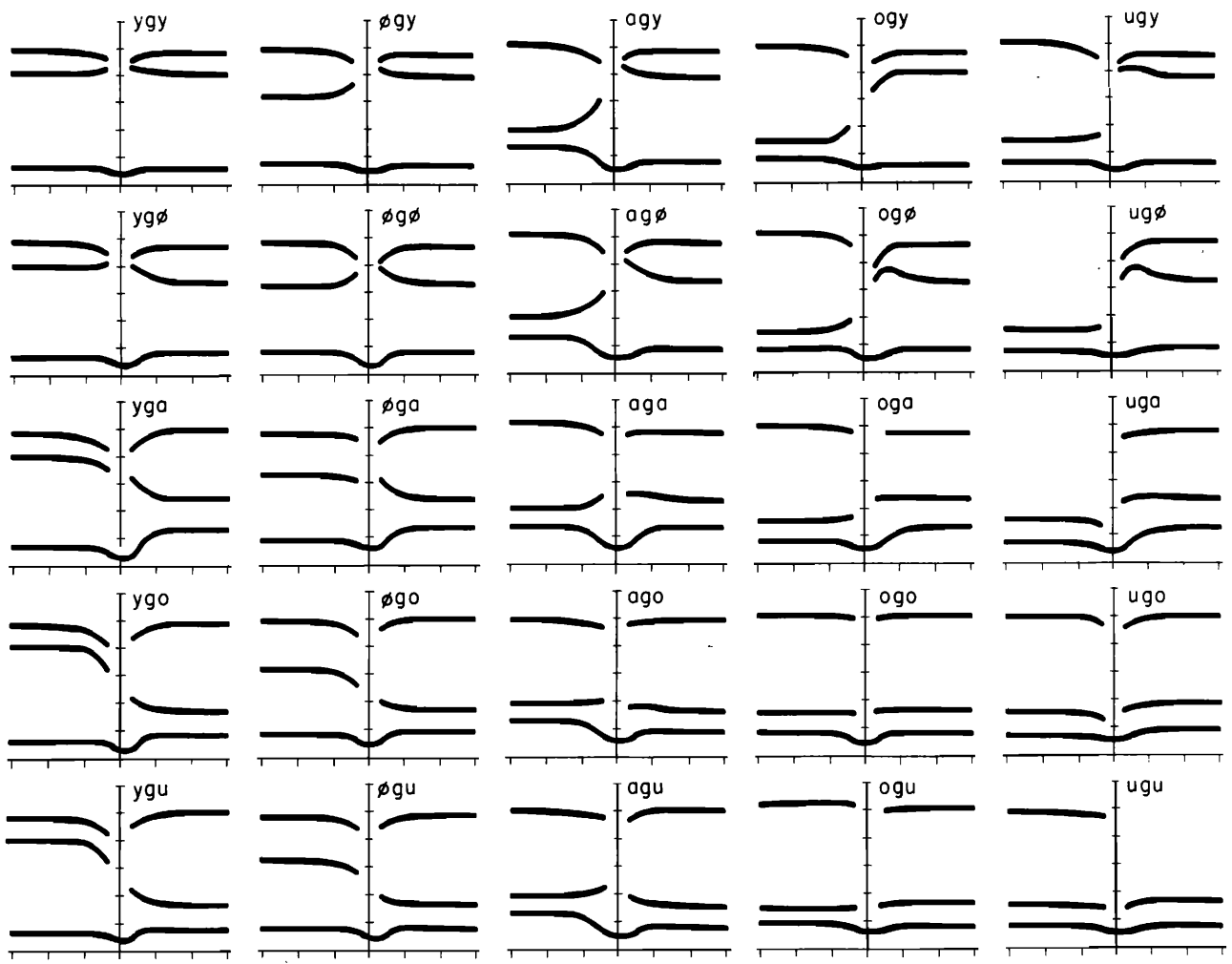

FIG. 29. Formant transitions for [g] as a function of preceding and following vowels, after Öhman (1966). Note that the formant motions for, e.g., [ga] release (middle horizontal panels) change as the preceding vowel changes.

evoked or the detailed acoustic effects of the transformation. For example, only the fronting/backing anticipation of the interacting vowels in a VCV sequence described by Öhman has been implemented as a change to the $F2$ trajectory. The list of discrete allophones inserted and manipulated by internal Klattalk rules, shown in Table III, is rather small. All other allophonic variation is created by modifying synthesizer control parameter data directly rather than by defining a discrete symbol.

In summary, it is likely that this area of allophonic detail and prosodic specification is one of the weaker aspects of rule systems, and contributes significantly to the perception of unnaturalness attributed to synthetic speech. Incremental improvements that are made to these rules on the basis of comparisons between rule output and natural speech cannot help but lead to improved performance of text-to-speech systems.

II. TEXT-TO-PHONEMES CONVERSION

Having considered the steps required to go from an abstract linguistic description to synthetic speech, we now turn to the problem of deriving this description from text. The recognition of printed characters, as required in, e.g., a reading machine for the blind, is beyond the scope of this review. We will assume that an ASCII representation of each input sentence is available as input to the text analysis module of a text-to-speech system. From considerations outlined in the previous section, it is clear that the text analysis routines have a formidable task. Ideally, the input is to be analyzed in such a way as to:

- reformat everything encountered (e.g., digits, abbreviations) into words and punctuation,
- parse the sentence to establish the surface syntactic structure,
- find the semantically determined locations of contrastive and emphatic stress,
- derive a phonemic representation for each word,
- assign a (lexical) stress pattern to each word.

For example, the input ASCII string for a typical input sentence, shown below, was processed by rules of Klattalk to derive an abstract linguistic representation consisting of phonemes, stress, and syntactic symbols. First, the word-formatting module transformed the numerals "*23*" into the words "*twenty-three.*"

INPUT TEXT:
The 23 protesters were arrested.
REFORMATTED INTO WORDS:
The twenty-three protesters were arrested.
(PARTIAL) SYNTACTIC ANALYSIS:
The twenty-three protesters) were arrested.
SEMANTIC ANALYSIS:
None.
(PARTIAL) MORPHEMIC ANALYSIS:
The twenty-three protest-er-s) were arrest-ed.
PHONEMIC CONVERSION AND LEXICAL STRESS ASSIGNMENT:
/ðə tw´ɛnti θr´i pr´otɛstɝz) wɝ ər´ɛstɨd./

A crude syntactic analysis of the sentence is then performed based on locations of any orthographic commas, as well as the syntactic role of function words and verbs that are detected during the dictionary matching process. In the sample text just above, the verb "*were*" is detected and marked as the beginning of the verb phrase through use of the [ )] symbol. The end of a declarative sentence is indicated by the period symbol. The most important aspects of syntactic structure are the locations of clause boundaries, and the location of the boundary between the noun phrase and the verb phrase, although there are other syntactic factors that affect the rhythm and intonation of longer sentences. Liberal use of commas in text would help a great deal in formulating natural phrasing; their presence is generally a reliable cue, but unfortunately their absence does not indicate the absence of an intonational phrase boundary.

There is no semantic analysis in Klattalk or any other present-day text-to-speech system. Every sentence is spoken in a sort of semantically "*neutral*" way, i.e., without emphatic or contrastive stress, unless the user indicates an important word by placing the phonemic ["] symbol before it in the orthography.

Next, a phonemic representation is obtained for the words in the manner shown in Fig. 30. Each word is compared with entries in a small pronunciation dictionary. If no match is found, the word is broken into smaller pieces (morphemes) by attempting to remove common suffixes such as "-ed," "-ing," etc. It may be necessary to add a silent "e" or to reconstitute the "y" in order to recover the true form of the root, as in "biting = bite + ing." Then the remaining root is again compared with entries in the phonemic dictionary. If there is still no match, a set of letter-to-phoneme rules are invoked to predict the pronunciation. In this sentence, two words had affixes removed, five words/roots were found in the dictionary, and the remaining one was processed by letter-to-sound rules. No errors were made. The morpheme "protest" was found to have two alternative pronunciations in the dictionary, one with primary stress on the first syllable and the other with primary stress on the second syllable, but a selectional restriction associated with the "-er" suffix caused correct selection of the noun form.

TABLE III. Two-character representations for selected allophones in Klattalk.

Allophone	Two characters	Description
ɾ	DX	flap
ʔ	Q	glottal stop
tˀ	TX	glottalized t
ɝ	RX	postvocalic r
ɬ	LX	postvocalic l
iɝ	IR	vowel in "beer"
eɝ	ER	vowel in "bear"
ɑɝ	AR	vowel in "bar"
oɝ	OR	vowel in "boar"
uɝ	UR	vowel in "poor"

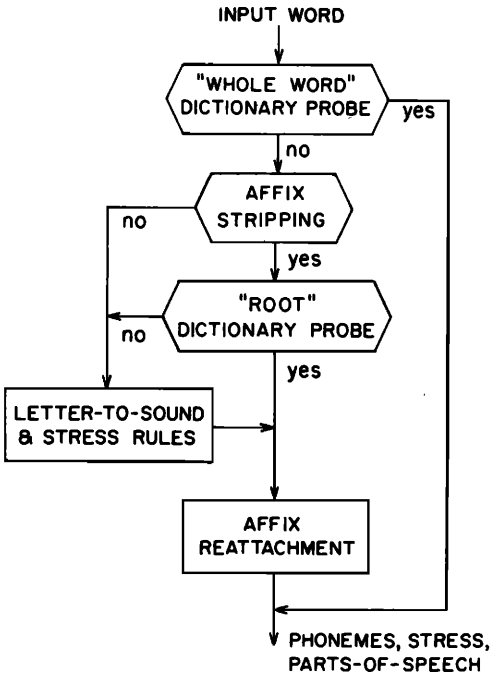

FIG. 30. The steps involved in converting an ASCII orthographic representation for a word into phonemes, stress, and parts-of-speech information. If the word or its base root is not in the dictionary, letter-to-sound rules guess at the proper pronunciation.

A part of the phonemic conversion process concerns the derivation of a stress pattern for the syllables of a word. Stress must be predicted if the word is not in the system vocabulary, or if the orthographic word is broken down into root plus affixes and an affix changes the stress pattern given for the root. The stress level of a syllable will be indicated by inserting a stress symbol just prior to the vowel in the phonemic representation. Absence of a stress symbol means that the syllable is unstressed.

A. Text formatting

A practical text-to-speech system has to be prepared to encounter words containing nonalphabetic characters, digit strings and unpronounceable ASCII characters. MITalk was one of the first systems to include algorithms for handling special cases such as how to speak digits in different formats (Allen *et al.*, 1979), e.g., "$35.61, 35.61, 2000, the year 1971, 10:15 p.m." This system also expanded many common abbreviations into full word equivalents. Commercial systems, which must be prepared to deal with more exotic material such as embedded escape sequences and other nonalphabetic characters, have adopted two general strategies. The Infovox SA-101 and the Prose-2000 provide the user with a set of logical switches which determine what to do with certain types of nonalphabetic strings. For example, " – " is translated to either "*dash*" or "*minus*" depending on the state of a switch. DECtalk, on the other hand, ignores escape characters, and usually spells out words containing nonalphabetic characters. The reasoning is that it is impossible to do the right thing in general, and the correct option for a particular application should be determined by a host computer. Even a simple strategy, such as interpreting a tab as an indicator of a new paragraph that should begin with a higher fundamental frequency, is not a safe assumption in arbitrary text; DECtalk therefore requires that a host computer insert a special "*new paragraph*" symbol in the text instead whenever tabs can be interpreted as new paragraphs. O'Malley *et al.* (1986) point out that many abbreviations are ambiguous, but can be disambiguated in particular applications. For example "N." is spoken as a letter in a name, as "North" in a street address, and as "New" in a state abbreviation, but these are easy fields to distinguish in a properly structured data base.

B. Letter-to-phoneme conversion

One issue in the preparation of rules and data structures for synthesis is how to best represent phonemes, allophones, stress, and syntactic symbols. Dictionaries generally do not agree on a standard representation, although the International Phonetic Association publishes one standard, and *The Journal of the Acoustical Society of America* employs a similar standard set of phonemic symbols that are used here in the examples. However, computers often require a representation that can be printed within the limitations of the ASCII character set. There is no agreement on either the set of phonetic symbols to be represented or the phonetic/alphabetic correspondences in this situation. The problem does not really require solution until such time as researchers wish to share data bases consisting of dictionaries or rules, and even then the most important issue is clear definition since computers are very good at symbol translation if they know what each symbol is intended to mean.

In my research, I have found it convenient to work with two different computer representations. One is case insensitive (upper case and lower case letters are equivalent) and requires two letters to represent vowels and some consonants. It is easy to type and easy to learn, so it is the way that words are input to Klattalk in phonemic form. The representation is nearly identical to the ARPAbet (Shoup, 1980). The second representation consists of a single ASCII character per phonetic symbol and so is an efficient way to store dictionaries and compare strings. Both representations can be parsed without the need for spaces between phonetic elements—in fact, "space" is the symbol used to indicate a word boundary. The two-character representation is defined and explained in Conroy *et al.* (1986, pp. 79–97), while the one-character set is described in Minow and Klatt (1983, Chap. 4). They are reprinted in Tables IV and V. The following are the only somewhat nonstandard symbols allowed in the abstract representation for a sentence: (1) there are two variants of schwa, /ə/ and /ɨ/, although the one to be used in any context is largely determined by the adjacent phonetic segments, (2) there is a separate symbol /yu/ for the more usual /y/ + /u^{w}/ because the fronting of /u^{w}/ in this environment would otherwise have to be done by a special rule, (3) there is a mapping of stressed /ɝ/ and unstressed /ɚ/ onto the single symbol /ɝ/, since this will cause no confusion and will make possible a slight saving in table space, (4) a silence phoneme is defined which is inserted by rule at cer-

TABLE IV. Two-character and one-character representations for phonemes in DECtalk.

Phoneme	Two characters	One character	Example
i	IY	i	beet
ɪ	IH	I	bit
e^{y}	EY	e	bait
ɛ	EH	E	bet
æ	AE	@	bat
ɑ	AA	a	pot
ɔ	AO	c	bought
ʌ	AH	^	but
o^{w}	OW	o	boat
ʊ	UH	U	book
u	UW	u	boot
ɝ	RR	R	Bert
$ɑ^{y}$	AY	A	bite
o^{y}	OY	O	boy
$ɑ^{w}$	AW	W	bout
yu	YU	Y	Butte
ə	AX	x	about
ɨ	IX	|	nieces
p	P	p	pet
b	B	b	bet
t	T	t	tet
d	D	d	debt
k	K	k	kit
g	G	g	get
č	CH	C	Chet
ǰ	JH	J	jet
m	M	m	met
n	N	n	net
ŋ	NX	G	sang
f	F	f	fed
v	V	v	vet
θ	TH	T	thin
ð	DH	D	this
s	S	s	set
z	Z	z	zero
š	SH	S	shed
ž	ZH	Z	azure
w	W	w	wet
y	YX	y	yet
r	R	r	red
l	L	l	let
h	HX	h	head
n̩	EN	N	button
l̩	EL	L	bottle
_	_	_	silence "phoneme"

TABLE V. Two-character and one-character representations for stress and syntactic symbols in DECtalk.

Two characters	One character	Example
´	´	primary stress
`	`	secondary stress
"	"	emphatic stress
-	-	syllable boundary
*	*	morpheme boundary
#	#	compound noun
⟨space⟩	⟨space⟩	word boundary
(	(	begin preposition
)	)	begin verb phrase
,	,	clause boundary
.	.	end of sentence
?	?	question intonation
!	!	end of exclamation
+	+	new paragraph

tain syntactic boundaries, but can also be specified by the user, (5) one permitted special level of phrasal emphasis and two levels of lexical stress are introduced (plus the additional alternatives unstressed and reduced), (6) syllable boundary and morpheme boundary symbols are provided, but are usually not required since rules assign consonants to syllables correctly in most cases anyway, (7) the compound noun symbol is introduced in order to be able to force stress reduction in the second element of the compound, (8) a very limited inventory of syntactic symbols is provided, and (9) the new paragraph symbol is defined and used to realize prosodic marking of new paragraphs. A clear deficiency of the present symbol inventory is the lack of any ability to approximate non-English sounds in foreign words, although this limitation is shared by many English speaking individuals.

Historically, English and most other languages employing an alphabetical spelling representation began with a system that was close to the way the word was pronounced (Venezky, 1965, 1970; Chomsky and Halle, 1968; Henderson, 1982). Over time, pronunciation habits changed, sometimes dramatically, so that the spelling reflects more nearly an underlying historical antecedent of current pronunciation instead of the synchronic phonemes. Thus rules for pronunciation of English words depend on complex conventions involving, e.g., remote silent "e," the number of consonants following a vowel, the grouping together of special letter pairs, such as "ch" and "gh," which normally function like a single letter (Wijk, 1969), but not if in separate morphemes, etc. English has also borrowed words from other languages, so that Latin, French, German, and other patterns, somewhat Anglicized, are fairly common.

A selected survey of the literature on derivation of phonemes and stress from orthography is presented in Fig. 31 as a block diagram. Interconnections indicate how fundamental theoretical analyses of English have been incorporated in laboratory programs for text analysis and, finally, in commercial text-to-speech systems. As indicated in the figure, methods used in most commercial systems for deriving a phonemic representation of a word involve the use of letter-to-sound rules and an exceptions dictionary. An attractive alternative, as we will see, is to develop a large morpheme dictionary and try to decompose each input word into its constituent morphemes (where morphemes are the minimal meaningful subparts of words).

Several initial attempts to predict word pronunciation just from the spelling (Ainsworth, 1973; McIlroy, 1974; Hunnicutt, 1976; Elovitz *et al.*, 1976; Carlson and Granström, 1976) started from the assumption that a letter or letter pair could be converted to the appropriate phoneme if just the right amount of adjacent letter context was examined. Based on this view, a set of conversion rules was devised to take care of letter pairs such as "ch" and "ea," and

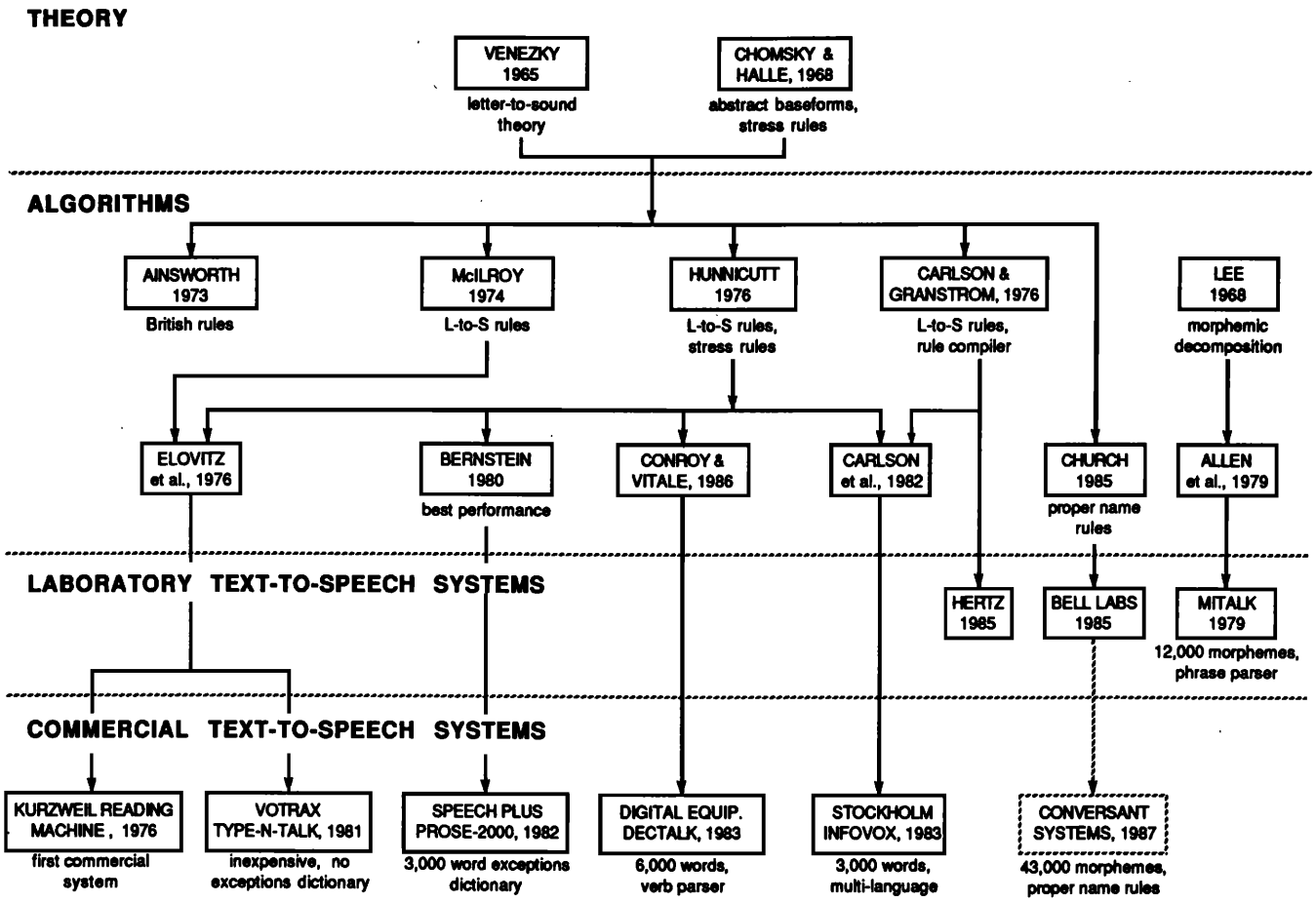

FIG. 31. Historical antecedents of the text-to-phoneme algorithms used in several laboratory and commercial text-to-speech systems.

then single letters were converted to phonemic form. For each letter, rules were ordered so that the first rules treated special cases of complex environmental specification, and the last case was always a default phonemic correspondence. For example, a rule might say that the letter A$\Rightarrow$/e/ if followed by VE. The rule treats correctly words like "behave," but not "have." A slightly more complicated variant of the same idea was to convert consonants first (Hunnicutt, 1976). This permitted the phonemic representations of consonants to be used in the context specifications for the more difficult conversion of vowels. Systems of this kind may have more than 500 such rules for the interpretation of letter strings.

Several major problems were immediately apparent: (1) vowel conversion depended in part on stress pattern, (2) correct analysis often required detection of morpheme boundaries, and (3) letter contexts had structural properties such as VC vs VCC that one would rather refer to instead of enumerating all possible letter sequences. Before discussing how these and the next generation of spelling conversion programs dealt with these issues, we consider a novel approach to the problem that has received considerable attention in the artificial intelligence community.

The conversion of letters to phonemes might appear to be a pattern matching problem amenable to statistical learning strategies. For example, Sejnowski and Rosenberg (1986) considered the problem of creating a network, which they called NETtalk, that takes a seven-letter window as input and outputs the phoneme corresponding to the middle letter. A set of 120 "hidden" neuron-like threshold elements mediated between input neurons corresponding to 29 possible letters at each of seven positions and an output set of neurons representing about 40 phonemes and two degrees of stress. The weighting of input connections and output connections of the hidden units was initially random, but was adjusted through incremental training on a 20 000 word phonemic dictionary. When evaluated on the words of this training set, the network was correct for about 90% of the phonemes and stress patterns. In some sense, this is a surprisingly good result in that so much knowledge could be embedded in a moderate number of about 25 000 weights, but the performance is not nearly as accurate as that of a good set of letter-to-sound rules (performing without use of an exceptions dictionary, but with rules for recognizing common affixes). A typical knowledge-based rule system (Bernstein and Pisoni, 1980) is claimed to perform at about

85% correct at a word level (all phonemes correct and stress pattern correct) in a random sampling of a very large dictionary, which implies a phoneme correct rate of better than 97%.[10]

NETtalk is related to a letter-pattern learning program described earlier by Lucassen and Mercer (1984). They defined a set of features corresponding to "random" sets of letters, used the forward–backward algorithm of the IBM speech recognition strategy (analogous to incremental training) on a 50 000 word lexicon to find the best feature sets for predicting individual phonemes, and established a set of probabilities (analogous to weights) for a search tree recognition model, based again on a seven-letter input window. They obtained correct letter-to-phoneme correspondences for 94% of the letters in words in a random sample from a 5000 word office-correspondence lexicon. In terms of error rate, this is slightly better than NETtalk, especially considering that some fraction of the test words was probably not in the training set, but the Lucassen and Mercer approach still results in an inferior words-correct error rate compared with traditional rule systems. Even a very powerful statistical package cannot yet discover much of the underlying structure in a process as complex as natural language.

A proposal in the psychological literature related to these pattern learning programs is that readers learn the letter-to-phoneme conversion rules not as explicit rules, but by analogy with similar local letter patterns in words that they already know how to pronounce (Glushko, 1981; Dedina and Nusbaum, 1986). For example, a novel word might be compared with all words in the lexicon, and the word sharing the largest number of letters with the unknown word would get to determine the pronunciation of that local substring. Glushko showed that subjects were slower to pronounce pseudowords that would have two equally likely alternative pronunciations if this strategy were followed. A computer implementation of a slightly more complicated version of this strategy (taking into account frequency of occurrence of analogous words) agreed with one of the pronunciations furnished by human subjects 91% of the time when tested on 70 simple pseudowords (Dedina and Nusbaum, 1986), while DECtalk pronunciations agreed with the response from at least one of seven human subjects 97% of the time. Klatt and Shipman (1982) defined a way in which the substring comparison strategy might be performed optimally and rapidly, one letter at a time, by creating a moderate-sized decision tree. They examined the performance when a 20 000 word phonemic dictionary was divided in half randomly such that the first half was used to create the tree, and the second half used to test it. The error rate for individual letters was 7%, which is not bad considering that test and training data were different, but this performance is still not nearly good enough to compete with conventional rule systems. Consonantal letters were found to be quite regular and amenable to translation with low error rates by this approach. However, the five vowel letters and "Y" accounted for four-fifths of the errors. In summary, given the attention that NETtalk and other neuron-like devices have received recently, it is disturbing that NETtalk does not learn training set data perfectly, appears to make generalizations suboptimally, and has an overall performance that is not acceptable for a practical system. Furthermore, it is unlikely that larger training lexicons would converge to a more acceptable performance. Aside from limitations imposed by the network model, problems inherent in all these approaches are (1) the considerable extent of letter context that can influence stress patterns in a long word (and hence affect vowel quality in words like "photograph/photography"), (2) the confusion caused by some letter pairs, like CH, which function as a single letter in a deep sense, and thus misalign any relevant letters occurring further from the vowel, and (3) the difficulty of dealing with compound words (such as "houseboat" with its silent "e"), i.e., compounds act as if a space were hidden between two of the letters inside the word. The necessity of morphemic analysis is supported by data indicating that good spellers look for morphemes inside letter strings (Fischer *et al.*, 1985), whereas to date these learning models seek regularities in letter patterns without recourse to a lexicon of any sort. On the other hand, efforts to find clear psychological evidence for morphological analysis of complex related forms (as opposed to rote learning of each) for word pairs such as "heal-/health," "original/originality," "magic/magician," and "sign/signal" have generally failed (Carlisle, 1985).

1. Prediction of lexical stress from orthography

The Hunnicutt (1976) rule system included the improved version of Chomsky–Halle stress rules (Halle and Keyser, 1971) consisting of eight general rules, the most well-known of which are the main and alternating stress rules for predicting which syllable receives primary stress as a function of the "strong/weak" syllable pattern of the word. Also included were rules for decomposing words by stripping off affixes to recover the root. About 15 different prefixes and 50 suffixes were detected. Grammatical constraints were invoked to prevent incompatible suffix sequences from being removed. Orthographic features permitted rules to refer to concepts such as "true consonant" and "vowel-like letter." While the best performing algorithm of its time, this system was completely correct for only about 65% of a random selection of words (Hunnicutt, 1980).[11] A good fraction of the errors made by this letter-to-phoneme system were stress errors. In fact, Bernstein and Nessly (1981) showed that a much simpler set of stress rules described by Hill and Nessly (1973) performed about as well as the Chomsky–Halle implementation. More recent high-performance letter-to-phoneme rule systems (Bernstein and Nessly, 1981; Hunnicutt, 1980; Hertz, 1982; Carlson *et al.*, 1982a; Church, 1985, Conroy and Vitale, 1986) include improved attempts at morphemic decomposition and stress prediction. Stress assignment is perhaps the weakest link in all systems because an incorrect stress pattern, while perceptually disruptive in and of itself, usually also triggers mis-selection of vowel qualities. The newer systems not only base stress assignment on factors such as morphological structure and the distinction between strong and weak syllables (Chomsky and Halle, 1968), but also on presumed part of speech, and in some cases, etymology (for a good review, see Church, 1985). The importance of syntactic categorization is sug-

gested by statistics indicating that over 90% of bisyllabic nouns have stress on the first syllable, while only about 15% of bisyllabic verbs are stressed on the first syllable (Francis and Kučera, 1982).

One issue faced by designers of systems is which to do first, stress prediction or phoneme prediction. Another issue is whether to essentially work forward or backward through the letter string for a word. While no system has to go only left to right, or completely settle stress prediction prior to phonemic analysis, there seem to be clear advantages to working backwards through the letter string, and to having stress information prior to making vowel decisions (Bernstein and Nessly, 1981).

2. Exceptions to the rules

When evaluating a set of letter-to-phoneme rules, it is easy to make up lists of words that fail to be pronounced properly. Systematic comparison of the rules against a list of frequent words can produce a dictionary of exceptions that, if added to the system, will make overall pronunciation performance much better than for a system that only uses rules. The utility of a small exceptions dictionary can be appreciated by observing the ability of a small number of most frequent words to account for a given fraction of words in running text (Hunnicutt, 1980). The data are reproduced in Fig. 32. They indicate that a small number of words, about 200, are required to cover half the words occurring in a random text. With a dictionary of 2000 words, over 70% of the words in text will be matched and not have to go through letter-to-sound rules. However, the law of diminishing returns begins to take over shortly after this point—if one extrapolates from the slope of the curve prior to 10 000 words,[12] as indicated by the dashed line in Fig. 32, it appears that to go from 90% to 93% coverage would require about an additional 60 000 words!

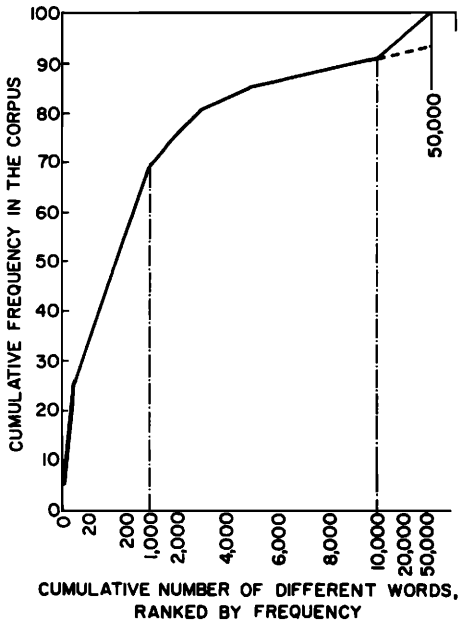

FIG. 32. A one million word corpus (Kučera and Francis, 1967), containing about 50 000 different words, can be used to estimate the number of entries in a lexicon necessary to match a given percent of words in a new text, after Hunnicutt (1980).

Elovitz *et al.* (1976) and Hertz (1982) embed lists of exceptions inside the letter-to-sound rules of their systems (such as the observation that the letter "f" is pronounced with a voiceless/f/ phoneme in all words except "of") so as to ensure getting common words correct, whereas others tend to segregate out exceptions as a separate dictionary. The best performance for a rule system without exceptions dictionary, better than 85% correct when tested on a random sample from a large dictionary, has been obtained by the Bernstein rules that are a part of the Speech Plus, Inc. Prose-2000 (Groner *et al.*, 1982). Bernstein argues that it is possible to design a letter-to-sound algorithm with a very simple structure—consisting of one right-to-left pass through the letters, starting inside all stress-neutral suffixes.

A moderate-sized exceptions dictionary can hide the deficiencies of a weak set of letter-to-sound rules, but at a high cost in terms of storage requirements. Based on data shown in Fig. 32, Hunnicutt (1980) showed that the size of an exceptions dictionary required to get a target fraction of input words pronounced correctly in a typical running text is a strong function of letter-to-sound rule performance. For example, the 3000-word exceptions dictionary in the Speech Plus Prose-2000, coupled with rules that are correct 85% of the time, results in an overall system performance of better than 97% correct (only 1 word in 33 in a typical text contains a noticeable phoneme or stress error). On the other hand, the first version of DECtalk, employing the Hunnicutt (1980) rules with 65% accuracy and a larger 6000-word exceptions dictionary, barely reached 95% correct (1 error every 20 words). Independent confirmation of this accuracy comparison comes from Huggins *et al.* (1986), who examined over 1600 low-frequency polysyllabic words and found phonemic mispronunciations in 8.3% for the new Speech Plus Calltext system, compared with 12.9% errors for Version 1 of DECtalk. The current DECtalk, Version 3.0, uses a new letter-to-sound rule system (Conroy and Vitale, 1986) to achieve performance of fewer than 6% errors for this data set, according to my evaluation.

In the future, it is expected that morpheme-based algorithms (see below) will replace exceptions dictionaries in commercial systems because the cost of memory is such that the added performance is well worth the expense. Similarly, special algorithms for pronunciation of names are likely to be incorporated in commercial systems in the near future. Special purpose vocabularies, such as a dictionary of medical terms, will probably also become available in response to market pressures.

3. Morphemic decomposition

Problems with the pronunciation of compounds such as the "*th*" in "*hothouse*" and the silent "*e*" in "*houseboat*" led Lee (1969) to attempt to break each word into morphemes, the minimal meaningful unit of language (see, e.g., Bloomfield, 1933, Chaps. 10, 13–14). Using a dictionary of about 3000 morphemes, Lee was able to split a word such as "*houseboats*" into "*house*" plus "*boat*" plus the plural "*-s*," and to retrieve from storage or predict the pronunciation of each piece. Lee developed techniques for recovering the proper base form after an affix was removed. The three most common problems, which could be handled correctly most

of the time using morphological decomposition, involved situations when the surface form did not contain a silent "e" (choking→choke + ing), there had been consonant doubling (omitted→omit + ed) or a final "y" had been modified (cities→city + s). Jonathan Allen and Deborah Finkel extended these techniques by increasing the morpheme dictionary to 12 000 items (Allen *et al.*, 1979; Allen *et al.*, 1987). Morphemes were selected by interactive examination of the approximately 50 000 unique words in the Brown corpus, a sampling of one million words of text (Kučera and Francis, 1967).[13]

Allen *et al.* (1979) also developed rules for handling cases where a word has multiple parses (e.g., "*scarcity*" = "*scarce* + *ity*" or "*scar* + *city*"). One rule, illustrated by this example, is that affixing is more likely than compounding. None of these guidelines is absolute, so in comparing two alternative morphemic decompositions, the authors invoked a set of heuristic scoring procedures whereby a given morphemic division incurs a scoring penalty depending on what has happened so far. This scoring algorithm picks the correct decomposition for "formally" from among the set ⟨form + all + y, for + mall + y, form + ally, form + al + ly⟩. If, after all of this computation, the word was found to be an exception to the parsing heuristics (e.g., "*been*" not pronounced as "*be*" + "*-en*"), the whole word was added to the morpheme lexicon in unparsed form.[14] An alternative method for dealing with inflectional suffixes, derivational affixes, and compounding is discussed in Church (1985, p. 251).

Some morphemes are pronounced differently depending on the stress pattern of the word and the nature of the other morphemes present (note the second "*o*" of "*photo*" is realized phonemically as /o,ə,ɑ/ in "*photo*," "*photograph*," "*photography*," respectively). The MITalk group developed rules to handle some of these cases, and simply added whole multimorphemic words to the lexicon if the rule was too complex or not sufficiently productive. The morpheme decomposition algorithm is able to parse about 98% of the words in a typical text, and should have greater accuracy than letter-to-phoneme rules. The exact accuracy of the MITalk morpheme decomposition algorithm was never measured, although a cursory glance at a three-paragraph text (Allen *et al.*, 1987, pp. 89–92) indicates a few (easily correctable) errors and a words-correct rate of only about 95%.

One of the advantages of a morpheme lexicon, aside from an ability to divide compound words properly, is that a set of 12 000 morphemes can represent well over 100 000 English words. Thus a very large vocabulary is achieved at moderate storage cost. However, the greatest advantage of the morpheme lexicon may turn out to be its ability to specify parts of speech information to a syntactic analyzer in order to improve the prosody of sentences, see below.

Recent work at Bell Laboratories (Coker, 1985) has extended this approach by augmenting the morpheme lexicon to 43 000 morphemes, and adding to the rules for suffix and prefix analysis and stress reassignment for the stress-shifting suffixes. The algorithm and morpheme lexicon occupy about 900 kbytes on a developmental real-time text-to-speech board (Olive and Liberman, 1985).

4. Proper names

Proper names are a special problem because the rules for their pronunciation often depend on which language is assumed as the underlying origin of the spelling (Liberman, 1979). The commercial system that performs best at pronouncing proper names, the newest Speech Plus Calltext board, still has an error rate of about 20% in its rule component when confronted with random proper names (Wright *et al.*, 1986). Church (1985) has recently proposed a solution to this problem that involves statistics on the frequency of occurrence of three-letter sequences in each of several languages. The first step is to use these statistics to estimate the language family of the unknown word. For words of moderate length, he finds that frequently one or another letter triple in the word essentially rules out all but the correct language. The second step is to apply stress and letter-to-phoneme rules for the language in question. Performance is claimed to be far superior to that of any system restricted to a single set of rules for all proper names. The importance of doing proper names by rule is brought out by statistical analyses showing that large name dictionaries do not solve the problem. An exceptions dictionary containing 2000 proper names will cover about 50% of the names in a random telephone directory, and 6000 proper names will cover about 60%. However, adding to the exceptions dictionary beyond 6000 names is essentially fruitless in that one is unable to get beyond an asymptote of about 62% of the names in one telephone directory, no matter how many names are obtained from another directory (Church, 1985).

C. Syntactic analysis

Imposition of an appropriate prosodic contour on a sentence requires at least a partial syntactic analysis. Furthermore, some pronunciation ambiguities can be resolved from syntactic information. For example, there are more than 50 noun/verb ambiguous words such as "*permit*" that are pronounced with stress on the first syllable if a noun, and with stress on the second syllable if a verb (see Appendix D in Conroy *et al.*, 1986). The only way to pronounce these words correctly is to figure out the syntactic structure of an input sentence, including the location of the verbs. Proper phrasing of moderately long clauses also requires knowledge of the locations of phrase boundaries. Thus it would be highly desirable to include a parser in a text-to-speech system.

While powerful parsing strategies exist (see, e.g., Woods, 1970; Aho and Ullman, 1972; Marcus, 1980; Kaplan and Bresnan, 1982), they tend to produce many alternative parses, even for sentences that seem simple and unambiguous. For example, "Time flies like an arrow" is multiply ambiguous at a *syntactic* level; a syntactic analysis system would require an immense store of world knowledge (semantics/pragmatics) to behave as we do and focus immediately on the only sensible structural interpretation of the sentence. Allen (1976) foresaw this problem and restricted himself to the goal of selecting the most probable local phrase parse of an arbitrary English sentence. Using the morpheme decomposition algorithm just described, he and Calvin Drake were able to obtain reasonably accurate

part-of-speech alternatives for most words of the sentence from the morpheme decomposition routine, and assumed tentatively that all unanalyzable words were nouns. The syntactic analysis proceeded left-to-right, attempting to add as many words as possible to each phrasal constituent. A backup algorithm suggested by Lorinda Cherry at Bell Laboratories sought possible verbs if it turned out that this process failed to recover a verb, as would be the case when a noun/verb ambiguity like "*permit*" was present in a sentence such as "Police permit mopeds." While the performance of this parser was never extensively tested, examination of some sample texts (Allen *et al.*, 1987, pp. 89–92) suggests that it works reasonably well, but produces several inappropriate pauses and pseudopauses at falsely detected boundaries.

If a parts-of-speech categorization is not available for most words, the simplest parsing strategy would be to use function words such as prepositions, conjunctions, and articles to find obvious phrase boundaries, leaving the remaining boundaries undetected. This is the strategy employed in the Prose-2000 and in the Infovox SA-101. The Votrax Type-n-Talk appears to use only punctuation marks as parsing cues.

DECtalk employs not only function words, but also a moderate-sized dictionary of verbs that unambiguously indicates the beginning of a verb phrase (Klatt, 1975a). Detection of the beginning of a verb phrase in a long clause permits DECtalk to break the intonation contour into two rise–fall "*hat-pattern*" units that help the listener parse the sentence. However, it is better to miss a noun-phrase/verb-phrase boundary than to insert prosodic boundary gestures (fall–rise intonation contour and lengthening of a phrase-final syllable) at locations where they do not belong. In an earlier experimental system that assumed that any word that could be a verb was a verb, listeners were distracted and often confused by extra prosodic boundaries, while the absence of a prosodic gesture just sounded like the speaker was talking too fast. DECtalk also provides a simple mechanism for a user to indicate a phrase boundary when one is missed—the [)] symbol can be inserted between the words in question. DECtalk does not try to disambiguate noun/verb ambiguities; the most frequent pronunciation is given unless the user requests the second most frequent pronunciation by attaching a special symbol to the front of the orthography.

DECtalk and other text-to-speech systems make a large number of syntactic errors that lead to noticeable misphrasings. In the future, syntactic routines will be expected to provide better detection of the following:

- phrasal constituency—particularly the locations of left-branching constituents and non-adjacent sister constituents that should probably be marked by prosodic gestures,
- internal structure and compounding relations within long noun/adjective strings,
- when to "pop" from an embedded clause that is not terminated by a comma,
- how to determine the nature of conjoined units on either side of a conjunction so as to be able to insert a syntactic break when appropriate,
- syntactic deletion sites where some sort of prosodic gesture should be synthesized to indicate the location of the missing material (Cooper *et al.*, 1978),
- how to detect tags and parenthetical material such as "This is the answer, he told us," that are usually said in a noninflected way,
- resolution of part-of-speech ambiguity, for (1) words that can be either an unstressed preposition or a stressed verbal particle such as "*on*" in "*He takes on hard jobs,*" (2) instances where "*that*" is functioning as a (stressed) demonstrative, e.g., "*I know (that) THAT book is red*" rather than as an unstressed clause introducer, as in "*I know that books are red,*" and (3) instances of compounds that are pronounced with reduced stress on the second word, such as "*He lived in Baker House* (this is largely a lexical/semantics problem).

D. Semantic analysis

Semantic and pragmatic knowledge is needed to disambiguate sentences like the ones the New Yorker is fond of reprinting. For example, in a sentence such as "*She hit the old man with the umbrella,*" there may be a pseudopause (a slowing down of speaking rate and a fall–rise in pitch) between the words "*man*" and "*with*" if the woman held the umbrella, but not if the old man did. Similarly, a "*rocking chair*" will have the word "*chair*" destressed if the combination of adjective and noun has been associated by frequent use into a single compound-noun entity. Emphasis or contrastive stress may be applied to an important word depending on the meaning: "*The OLD man sat in a rocker*" (not the younger man). Finally, words that have lost their importance in a dialog, either because of prior occurrence of the word or by anaphoric reference, should be destressed.

No text-to-speech system is capable of dealing automatically with any of these issues. DECtalk employs the simplest possible solution by providing the user with an input inventory of symbols to facilitate user specification of the locations of missing pseudopauses (the [)] symbol), unmarked compound words (spell as "*rocking-chair*"), and emphasis (precede the emphasized word by an emphasis symbol ["]).

It is possible to think of applications where the computer is not simply attempting to speak ASCII text, but may know a great deal about the meaning of the message, perhaps having formulated the text from a deep-structure semantic representation in, e.g., a data base information retrieval application (Young and Fallside, 1979). In such cases, one would want to take advantage of the ability to mark for emphasis important words when forming the input to the text-to-speech system. Hirshberg and Pierrehumbert (1986) provide an excellent review of the factors influencing the intonational structuring of discourse.

In the future, systems that have available parts-of-speech information from a large morpheme lexicon can be expected to develop better syntactic analysis routines that are particularly suited to the problems of text synthesis. Perhaps computer science efforts to produce expert systems will

lead to advances in semantic representation that can be adapted to text synthesis as well.

III. HARDWARE IMPLEMENTATION

A laboratory text-to-speech system, or a development system, is best implemented on a large general-purpose digital computer. The flexibility and nearly unlimited computational resources outweigh disadvantages of non-real-time output. However, practical commercial systems must realize real-time operation at a reasonable cost/performance tradeoff, while simultaneously providing additional features such as a flexible user interface and telephonics for many commercial applications. Solutions may require specially designed chip sets (Gagnon, 1978; Goldhor and Lund, 1983) or circuit boards containing off-the-shelf components rich in computer power and memory (Groner *et al.*, 1982; Bruckert *et al.*, 1983).

One important design consideration is the sampling rate and resultant high-frequency cutoff of the output speech. Since many business applications require the telephone, some systems limit the frequency response to that of telephone bandwidth—3.4 kHz, or the 4.0-kHz limit imposed by the 8-kHz sampling rate of standard codec digital transmission of speech (Groner, 1982; Olive and Liberman, 1985). DECtalk, on the other hand, produces information at frequencies up to 5 kHz in order to maximize intelligibility over a loudspeaker in, e.g., handicapped applications, such as a reading machine for the blind.

My own experiences may help illustrate hardware issues. In order to transform the Klattalk software into a real-time device, it was necessary for me to find a commercial partner with the appropriate skills and deep pockets. Fortunately, Digital Equipment Corporation was willing to underwrite the development costs. We signed a license agreement in 1982 (Klatt, 1987), and a product, DECtalk, was announced some 18 months later (Bruckert *et al.*, 1983).

The DECtalk hardware, Fig. 33, was capable of implementing the complete existing Klattalk software; no engineering compromises were necessary. Software added by Digital engineers controlled the user interface to a host computer. Host computer commands were defined to permit initiation or reception of telephone calls, and to permit the host to suddenly halt speaking, or to monitor the instant when a particular word in a sentence has been spoken.

The hardware shown in Fig. 33 includes (1) a Motorola MC68000 general purpose digital computer that processes text corresponding to one clause at a time, producing a set of synthesizer control parameters every 6.4 ms, and (2) a Texas Instruments TMS-32010 signal processing chip that converts control parameters to difference equation constants, and simulates the digital formant synthesizer in order to produce 10 000 12-bit waveform samples per second. Memory requirements are modest. The 6000-word exceptions dictionary places the greatest demands on memory; it occupies about half of the read-only memory shown in the figure. DECtalk can be controlled by any computer or by an ordinary computer terminal since the communication link is via a standard RS-232 port.

The only disappointment was that the price of the original DECtalk system turned out to be about four times our early estimate of $1000, and this placed the device outside the reach of many potential handicapped users. A recent redesign of the main DECtalk board to contain less "integrated circuit glue" has resulted in the DECtalk 3.0 system that is improved in several performance areas and is less expensive to manufacture, so there is still hope that an acceptable price might be achieved. Board size, about $8 \times 10 \times 0.7$-in. sans power and loudspeaker, is now satisfactory for portability, but lower power consumption is a goal that will have to be met in the future.

Today's technology is such that, I am told, it would be possible to put the entire text-to-speech algorithm on a single wafer-sized integrated circuit chip. However, this is not likely to happen until the demand is sufficient to justify chip design costs. Instead, it appears that future versions of the hardware may move toward greater flexibility by replacing all of the read-only memory by RAM that can be down loaded with new code as algorithms are improved.

IV. PERCEPTUAL EVALUATION OF TEXT-TO-SPEECH SYSTEMS

Text-to-speech systems can be evaluated and compared with respect to intelligibility, naturalness, and suitability for particular applications. One can measure the intelligibility of individual phonemes, words, or words in sentence context, and one can even estimate listening comprehension and

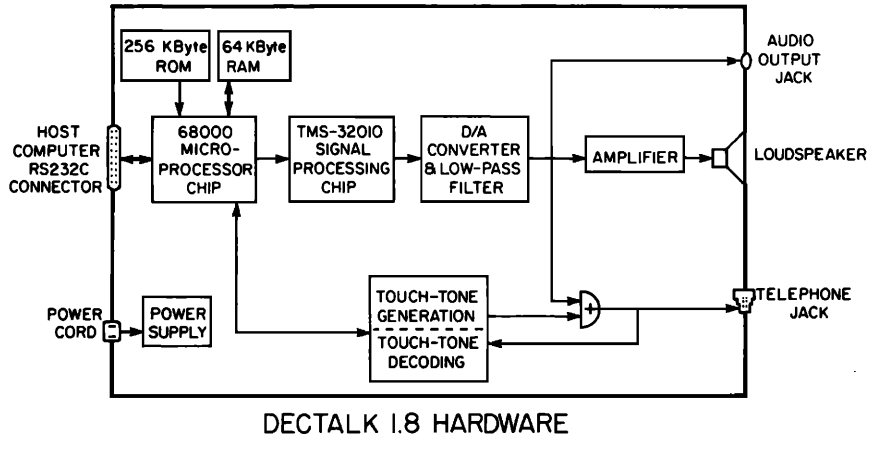

FIG. 33. Electronic hardware used in the first DECtalk implementation of Klattalk.