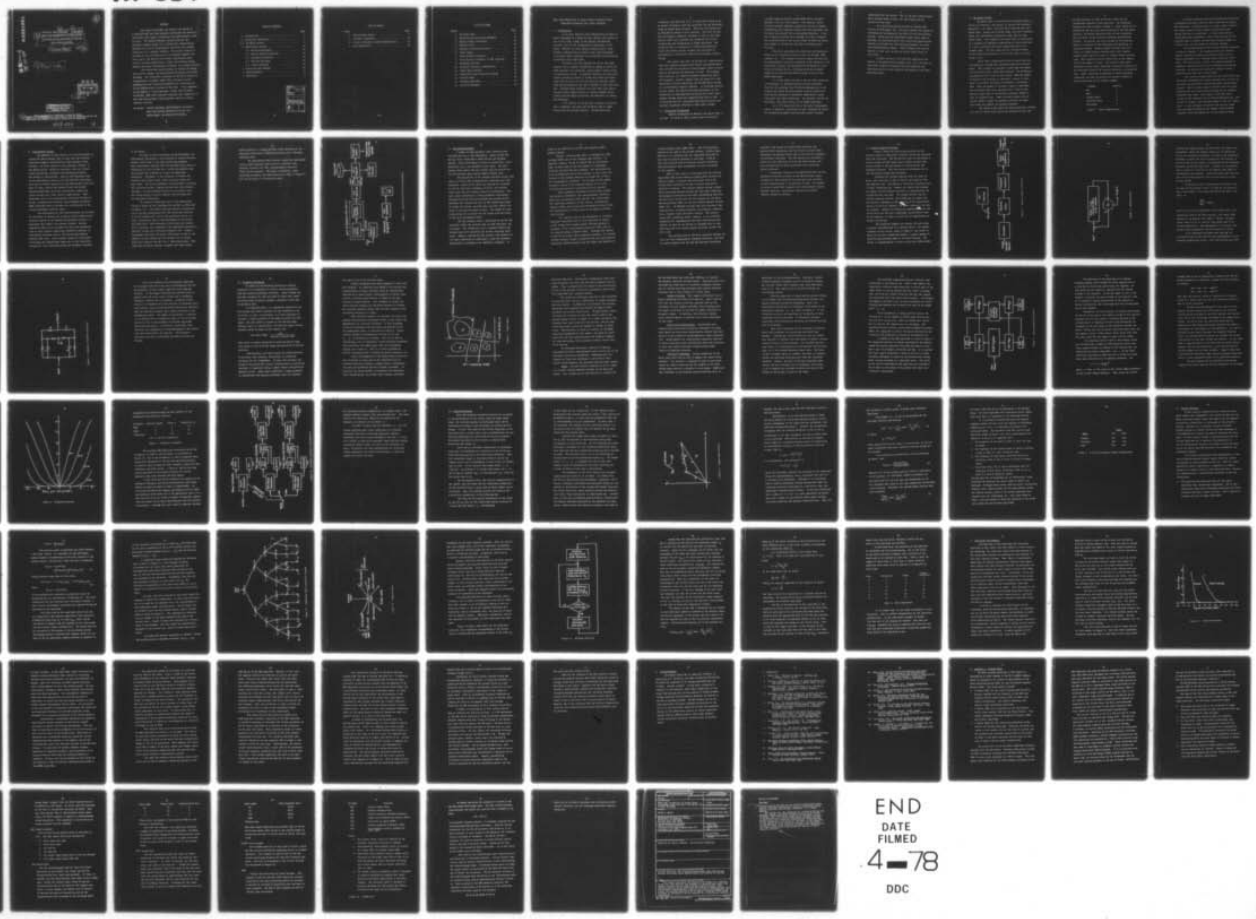
AD-A051 001 MARYLAND UNIV COLLEGE PARK DEPT OF COMPUTER SCIENCE F/G 9/4
REAL-TIME RECOGNITION OF MANUAL MORSE TELEGRAPHY USING NONLINEA--ETC(U)
JUL 77 D L MILLS N00014-67-A-0239-0032
UNCLASSIFIED TR-554 NL

1 OF 1
AD A051 001



END
DATE FILMED
4-78
DDC

(2)

B S.

(14) Technical Report TR-554 (11) July 1977

(6) REAL-TIME RECOGNITION OF MANUAL MORSE TELEGRAPHY USING NONLINEAR ESTIMATION AND VITERBI DECODING.

(10) by David L. Mills

(12) 92 p.

(9) Final rept.,

(15) This research was supported in part by Grant N0014-67-A-0239-0032 from the Office of Naval Research and by the Department of Computer Sciences, University of Maryland.

409 022 LB

ABSTRACT

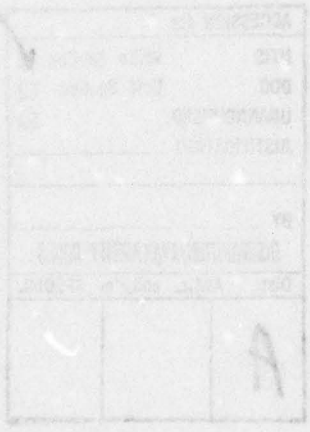
This report describes the analysis and design of a sophisticated real-time detector and decoder for manually-sent Morse code signals transmitted via high-frequency radio circuits. The detector/decoder system processes the digitally-sampled output of a conventional communications receiver in a LSI-11 microcomputer and prints the decoded characters on a computer terminal. The system software is designed to run if required in a distributed computer network such as the DCN and can provide simultaneous multi-circuit decoding capabilities, using a multiplexed analog-digital converter and multiple terminals, in a single machine.

The system uses the principles of optimum filtering and receiver design, including matched filtering and Viterbi decoding. The statistical structure of the Morse process is encapsulated in a Markov model which includes a-priori probabilities and transition probabilities reflecting the entire alphabetical structure of the code. A key component in the design is a new algorithm, called ratio-weighted estimation (RWE) which provides extremely fast adaptation to the time-varying Morse code parameters typical of multi-operator circuits.

Key Words: Viterbi Decoding, Ratio-Weighted Estimation, Real-Time System, Matched Filtering, Morse Markov Model, Distributed Processing.

TABLE OF CONTENTS

LIST OF TABLES



LIST OF FIGURES

REAL-TIME RECOGNITION OF MANUAL MORSE TELEGRAPHY USING NONLINEAR ESTIMATION AND VITERBI DECODING

1. Introduction

It is often observed that communication by means of Morse telegraphy is a dying art and, indeed, the commercial use of this code, except in the Amateur and Maritime services, has all but disappeared in many parts of the world. However, as every Amateur operator knows, long-distance communication with Morse code radio-telegraphy is possible under conditions of interference and distortion intolerable with other modes.

The Morse code was designed for use by the human operator and is usually transmitted at speeds at which the human operator can transcribe, using either a hand-written or typewritten record. A competent operator can transcribe a Morse message at speeds of twenty-five words-per-minute (wpm) with an error rate of less than five percent, while some skilled operators can copy at speeds to forty and sixty wpm. Although these operators can cope with truly minute received radio signals buried in strong cochannel interference and multipath distortion, the operation is tedious and fatiguing.

It is natural to ask how well a machine (in particular a computer) can cope with the Morse code in competition with the skilled operator. Having asked such

a question, one then asks if it is worth the trouble to get an answer, bearing in mind the relatively low and declining usage of manual Morse circuits. There are two reasons for an affirmative answer to this question. The first is that the intelligence community continues to rely on this mode when no others are available and automation of circuit surveillance and watchkeeping is highly desirable. The second is that it represents an interesting problem in itself and reveals useful insights into questions of psychophysics, statistical inference and optimum receiver design.

This report describes the design of a sophisticated real-time computer program which detects and decodes Morse code received via off-the-air high-frequency radio circuits used by the Amateur and Maritime services. The program, which operates on a Digital Equipment Corporation LSI-11 microcomputer, receives signals from the output of a communications receiver and prints the decoded Morse message on a conventional computer terminal. In following sections of this report the construction and performance of this system will be described and contrasted with other experimental systems which have appeared from time to time. It is believed that the system described here possesses significant advantages compared to these other systems.

1.1 Historical Perspective

Machine recognition of manually sent Morse code is not new. As early as 1959 a group at MIT constructed

a rather ungainly machine called MAUDE (Morse AUtomatic DEcoder) [2,3] for this purpose. This machine, really a special-purpose computer, processed a series of numbers representing the consecutive Morse code mark and space time intervals and decoded the corresponding character. A number of special features were included in order to improve the classification of the various kinds of marks and spaces and attempt to correct the text when a decoding error occurred.

The first experience the author had with these problems was in a program written in 1971 for the DEC PDP8 computer [5]. This program processed samples derived from an analog/digital converter connected to the output of a communications receiver and performed the functions of a predetection matched filter, following which the filtered samples were detected and decoded using certain of the MAUDE algorithms.

In a 1973 thesis written at the Air Force Institute of Technology [4] Guenther applied certain principals from the fast-growing field of Pattern Recognition to the problem and showed that performance could be improved by considering sequences of mark/space combinations rather than marks and spaces separately. His program, written for the PDP12, also used certain of the MAUDE algorithms.

In a 1975 thesis written at the Naval Postgraduate School at Monterey, California [6] Bell examined the effect of incorporating Kalman filtering and Viterbi decoding

algorithms into the system. This is the most sophisitcated Morse decoder known to date, but the program did not operate in real time.

In a 1976 dissertation written at Catholic University in Washington, D.C. [14] Howe studied the problem of estimating the mathematical statistics of the manual Morse process. Using the two-dimensional signal space model of Guenther, he used quasi-Bayes and statistical approximation methods to estimate the time-varying parameters of actual and simulated Morse signals. Howe's program also did not operate in real time, as much as thirty times slower in some cases.

A number of how-to articles have appeared on the subject of automatic Morse decoders (e.g. [1]), but none of these have studied the problem from the technical and analytical point of view taken by the authors of the works described above.

1.2 The Morse Process

The Morse code is constructed of alternate marks, or carrier-on intervals, and spaces, or carrier-off intervals. Marks are of two types, the short (dot) mark and the long (dash) mark. Spaces are of three types, the short (element) space, the intermediate (character) space and the long (word) space. Each symbol of the Morse code is constructed of a sequence of mark/space combinations in which each mark is a dot or dash and each space except the last is an element space. The last space can be a character space, if the following character is in the same word, or a word space if not.

Morse code is generated by one of several devices, including the straight key, mechanical bug and electronic keyer. The straight key generates a mark when the handle is pressed down and a space when released. The mechanical bug operates with a side-to-side motion. When the paddle is moved in one direction it generates a series of dots, while in the other direction it generates a continuous mark. When the paddle is released a space is generated. The electronic keyer also operates with a side-to-side motion, except that it can generate automatic dashes as well as automatic dots. Some electronic keyers (the so-called "iambic" types) can generate alternate dot/dash pairs when the paddles are squeezed together.

The timing of the marks and spaces generally conforms to a set of ratios which relate the duration of each type

of mark and space to that of the dot, which has the (normalized) duration of one time unit. The remaining elements are then as shown in Table 1. These ratios can be affected by the key used, the preference of the individual operator, the keying characteristics of the transmitter and the nature of the receiver filtering and detection process. In general, the ratio of dash to dot duration, which is a function of key adjustment and operation, is called the weight. Heavier weights (greater than 3:1) result in clumsy heavier-sounding code, while lighter weights (less than 3:1) result in choppy lighter-sounding code. Most operators tend to prefer code on the lighter side of the ideal 3:1 ratio. The ratio of short mark durations to short space durations, primarily a function of the threshold level of the receiver detector, is called the bias. Under conditions of moderate to severe multipath conditoons the bias can become very severe, leading to severe distortion of the element lengths.

Element	Duration
dot	1
dash	3
element space	1
character space	3
word space	7

Table 1. Ideal Element Ratios

In actual experience the most troublesome distortions in manually sent Morse code are those involving the spaces. Most operators, especially when in a hurry, tend to shorten the longer spaces so that correct classification can become difficult or impossible. In some cases only by using additional information, such as the duration of the previous mark or whether the partially detected symbol is valid or not, can the symbol be correctly decoded.

The interpretation of the various dot/dash sequences is shown in Figure 1. The sequences shown for letters, digits, punctuation marks and procedure signals are in general agreement [11], although some of them are rarely used. The error signal, used by the sending operator to indicate a formation error in the immediately preceeding character, is actually an indefinite sequence of dots, generally more than the six shown here.

The a-priori probabilities for the letters have been previously determined for typical English text [13] but are not known for the digits, punctuation marks and procedure signals. There is good reason to believe that the letter probabilities for typical Morse messages may be somewhat different than these. One reason is that most Morse operators make extensive use of service abbreviations, such as the "Q" signals [10] and patois such as "TU" for "thank you." A more reasonable choice for the a-priori probabilities might be based on the length of the dot/dash sequence, with the probability of each sequency being

Symbol	Freq Occur	Lng	Code	Symbol	Freq Occur	Lng	Code
A	.0642	.0336	·-	1		.0134	·----
B	.0127	.0168	-···	2		.0134	··---
C	.0218	.0168	-·-·	3		.0134	···--
D	.0317	.0224	-··	4		.0134	····-
E	.1031	.0671	·	5		.0134	·····
F	.0208	.0168	··-·	6		.0134	-····
G	.0152	.0224	--·	7		.0134	--···
H	.0467	.0168	····	8		.0134	---··
I	.0575	.0336	··	9		.0134	----·
J	.0008	.0168	·---	0		.0134	-----
K	.0049	.0224	-·-	= (break)		.0134	-···-
L	.0321	.0168	·-··	/ (stroke)		.0134	-··-·
M	.0198	.0336	--	? (question)		.0112	··--··
N	.0574	.0336	-·	" (quotes)		.0112	·-··-·
O	.0632	.0224	---	. (period)		.0112	·-·-·-
P	.0152	.0168	·--·	' (apostrophe)		.0112	·----·
Q	.0008	.0168	--·-	- (dash)		.0112	-····-
R	.0484	.0224	·-·	; (semicolon)		.0112	-·-·-·
S	.0514	.0224	···	() (parens)		.0112	-·--·-
T	.0796	.0671	-	, (comma)		.0112	--··--
U	.0228	.0224	··-	: (colon)		.0112	---···
V	.0083	.0168	···-	$\overline{SN}$ (understand)		.0134	···-·
W	.0175	.0224	·--	$\overline{AS}$ (wait)		.0134	·-···
X	.0013	.0168	-··-	$\overline{AR}$ (end msg)		.0134	·-·-·
Y	.0164	.0168	-·--	$\overline{CT}$ (attention		.0134	-·-·-
Z	.0005	.0168	--··	$\overline{KN}$ (go ahead)		.0134	-·--·
				$\overline{III}$ (error)		.0112	·····
SP (space)	.1859			$\overline{SK}$ (end work)		.0112	···-·-

Figure 1. The Morse Code

inversely proportional to its length. Both measures are shown in Figure 1. Provisions have been made to use either of these measures or to omit this information entirely in the experimental system.

2. Experimental System

The emphasis in this report is on the development of a practical Morse decoder able to cope with the difficult conditions commonly encountered on high-frequency radio circuits. Of major concern in this approach, however, is the demonstration, insofar as possible, that the various signal processing, estimation and decoding procedures are optimal or near-optimal. Many of the techniques used in the experimental system to be described have previously been used in similar applications and demonstrated effective. Some of these, such as the matched filter and Viterbi decoding algorithms, have been extensively studied and proven optimal for a wide class of applications. Other techniques used in the experimental system are believed to be new, including the nonlinear ratio-weighted estimation (RWE) algorithm used to estimate the distributions of the various mark/space interval types.

The requirement for real-time operation was dictated by the desire to operate the system on actual communication circuits typical of the Amateur and Maritime services, in particular the author's own Amateur station. Another important reason was to evaluate how real-time processing such as this could be effectively handled in networks of distributed resources such as the Distributed Computer Network (DCN) [18]. Still another reason was the problem of storing and transmitting large sets of real-time observations for later processing using the equipment available

to the author.

In order to allow convenient system development and experimental evaluation it was necessary to locate the Morse decoder itself near to the radio receiving equipment. Early experiments revealed that the man-made noise levels at frequencies in the 3-30 MHz were totally unacceptable when the receiver was situated in or near the Department Laboratory. Therefore, it was necessary to install the system at the author's home, where the noise levels are much lower. All of the equipment installed there, including the receiving station, computer system, test and interface equipment, are the personal property of the author; only the analog-digital converter modules were borrowed from the Department Laboratory.

The software technology used in the experimental system is based on that developed for the DCN, as described elsewhere [17]. The modular multiple-process architecture and re-entrant programming allow the processing functions to be distributed over two or more processors as was, in fact, occasionally done in the development of the system described here. When operating in this mode the resident LSI-11 processor was connected to the Department Laboratory's PDP11/45 systems via a 300-baud telephone line. Program development used both the Univac 1100-series systems at the Computer Science Center and the resident LSI-11 system, which also supports the DEC RT-11 Operating System. When used with the Univac systems, programs were down-loaded

either directly or through the DCN system operating in the Department Laboratory's PDP11/45 processors via a 300-baud telephone line.

The experimental Morse decoder system was implemented using a high-frequency radio receiver, modified radio teletype terminal unit (TU), analog/digital converter (ADC) and minicomputer. The types of equipment, their operating parameters and interconnections, shown in Figure 2, will be described in following sections.

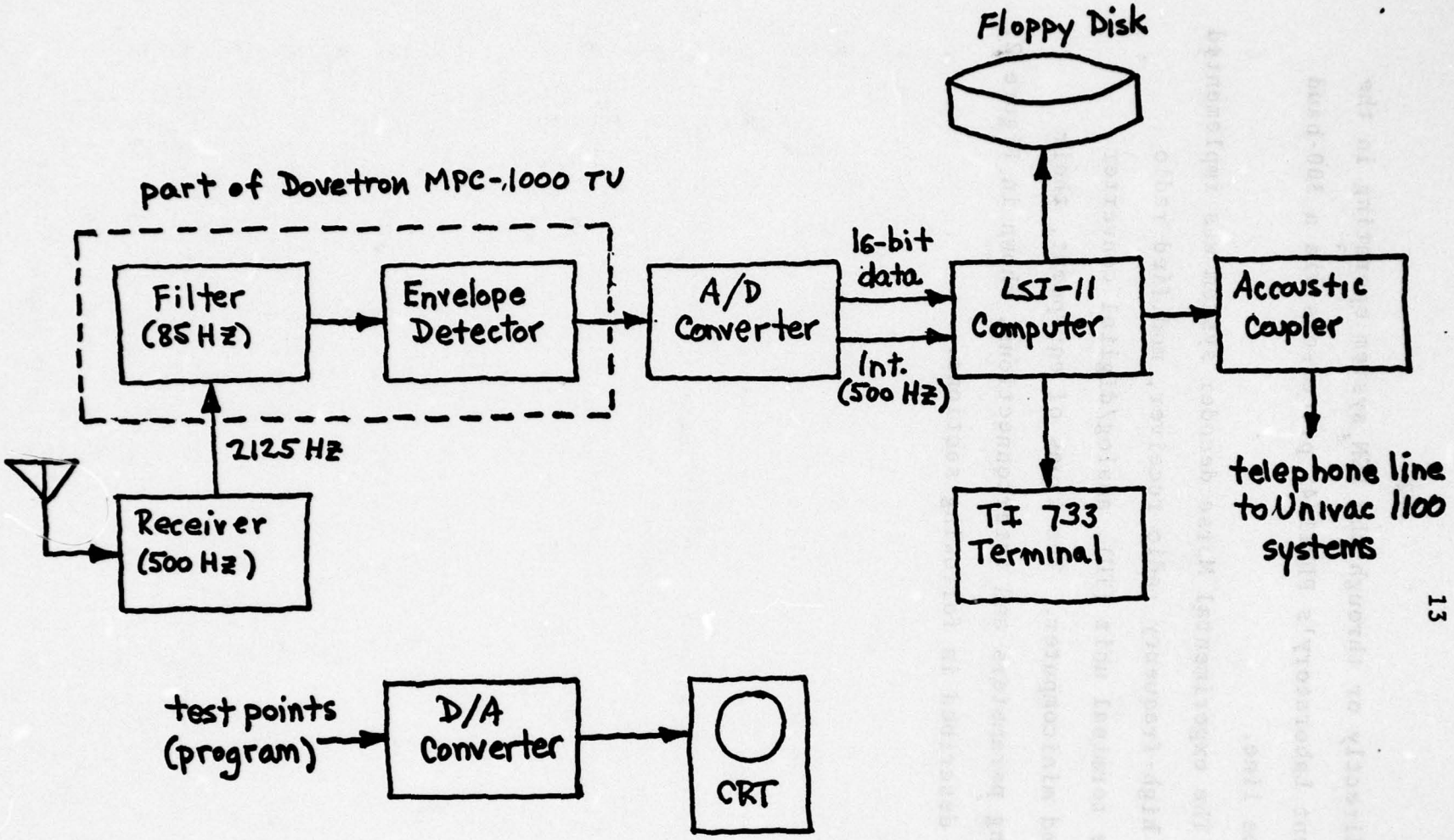


Figure 2. Receiving and Processing Equipment

2.1 Receiving Equipment

A number of high-frequency radio receivers were available for use in the experiments. These include the Collins 75S-3, Drake SPR-4, Heath SB-104 and National HRO-500. Their performance in terms of sensitivity, intermodulation, etc., was adequate in all tests. The only factors which affect the test results are the selectivity and the automatic gain control (AGC) recovery time.

The selectivity of the various receivers varies from 250 Hz to 500 Hz when configured for continuous-wave (CW) operation. Since in the tests the receiver was followed by the TU, which has a bandpass of 85 Hz, the receiver selectivity was not usually a factor in performance. However, under conditions of strong cochannel interference falling within the receiver passband but outside the TU passband, the receiver AGC acts on the interfering signal to reduce the overall receiver gain, and this reduces the level of the desired signal as well. One remedy for this problem would be to derive the AGC signal from the output of the 85 Hz narrowband filter.

The AGC recovery time is controlled by the RC time constants within the receiver and, on some receivers, can be varied. This turned out to be a valuable feature and allowed copy to be optimized for various receiving conditions. However, setting the AGC recovery time manually to fit these conditions is undesirable, since the equipment design here is intended to be completely automatic. It

would not be difficult to perform this function under program control.

The TU, a Dovetron MPC-1000, is designed for radioteletype (RTTY) use on two-frequency FSK circuits. It operates, however, in the so-called "AM" or limiterless mode, in which each of the two channels is separately filtered, detected and then combined. For the purpose of Morse decoding only one of the channels is used, and a connection to the envelope detector is brought out to drive the analog/digital converter. The TU used in this manner provides a narrow 85 Hz passband together with a soft limiting characteristic controllable by the operator.

Other equipment interfaced to the system includes one-kilowatt transmitters for radiotelegraph, radiotelephone and radioteletype modes and VHF FM transceivers. The remaining parameters of the station, including the antenna system, feed system and environmental noise levels, are representative of those used in the Amateur and Maritime services.

The computer used in the experiments is a Digital Equipment Corporation LSI-11, with 20K words of memory, a dual floppy disk storage system, 300-baud operator's terminal, 12-bit analog-digital converter (ADC), and 10-bit digital-analog converter (DAC). Although this machine is one of the slower PDP11-compatible machines, its features include firmware integer multiply/divide and floating-point operations (EIS/FIS option) and the floppy disk operates on

a direct-memory access (DMA) basis. (The floating-point feature is not used in the present system.) Without these features it is unlikely that the algorithms described in this report could be operated in real-time. A 500 Hz time base (derived from a precision crystal oscillator) and miscellaneous other peripherals are also interfaced to the computer.

Most of this report is concerned with the software algorithms used in the computer program which receives the digital samples, processes them and either stores the decoded Morse characters on disk or prints them on the operator's terminal. The software support system in which these algorithms operate handles the real-time scheduling, interprogram communication and other housekeeping functions. The particular system used for this purpose was the Basic Operating System (BOS) for the Distributed Computer Network (DCN) [17]. The BOS/DCN performs many more functions than required here, including automatic intercomputer communication, resource access and fault recovery. The primary reason this system was chosen, however, is its extensive capability for comprehensive real-time communication and control, as well as its ability to exchange data in standard form with other multi-purpose operating systems like RT-11 [19].

The software used in the Morse algorithm package consists of three independently scheduled processes, the first to receive samples from the ADC and determine mark/space

intervals, the second to receive these intervals and generate ASCII characters, and the third to receive ASCII characters and generate Morse code in the form of a sequence of mark/space intervals. The last process was included to serve as a test generator to drive the Morse decoder and also to allow two-way Morse operation with cooperating Amateur stations.

A noteworthy feature of the BOS/DCN software and the sharable re-entrant structure of the Morse decoder implementation is that extensions to serve more than one Morse circuit simultaneously is essentially trivial. For example, by using a faster PDP11 computer and a multichannel analog-digital converter, simultaneous operation of several channels should be possible.

2.2 Digital Signal Processing

Figure 3 shows the processing performed as the receiver output is digitized, filtered, sliced and converted into time intervals. The CRT monitor shown in the Figure is driven through the DAC using signal samples derived from various points in the processing, as determined by the equipment operator. This has proven an invaluable aid in the algorithmic design and evaluation.

Primarily for reasons separate from the needs of Morse code detection, a rate of 500 Hz was chosen as the ADC sampling rate. The ADC has a twelve-bit resolution and a linear companding law. Samples are read by the program at nominal two-millisecond intervals, as determined by the crystal-derived time base. The nominal dither from the uniform sampling interval varies from zero to almost one millisecond, depending on the activity of the input/output devices of the host computer, but never results in the loss of a sample. Samples are normalized immediately to fit the 16-bit word size and two's complement representation used by the computer, in order to avoid loss of significance in further processing.

The normalized samples are further filtered using a digital implementation of a matched filter. The impulse response of this filter, shown in Figure 4, was chosen to match the shortest signalling element, a square impluse of duration approximately equal to the baud interval. The filter is implemented as a first-in-first-out (FIFO) buffer

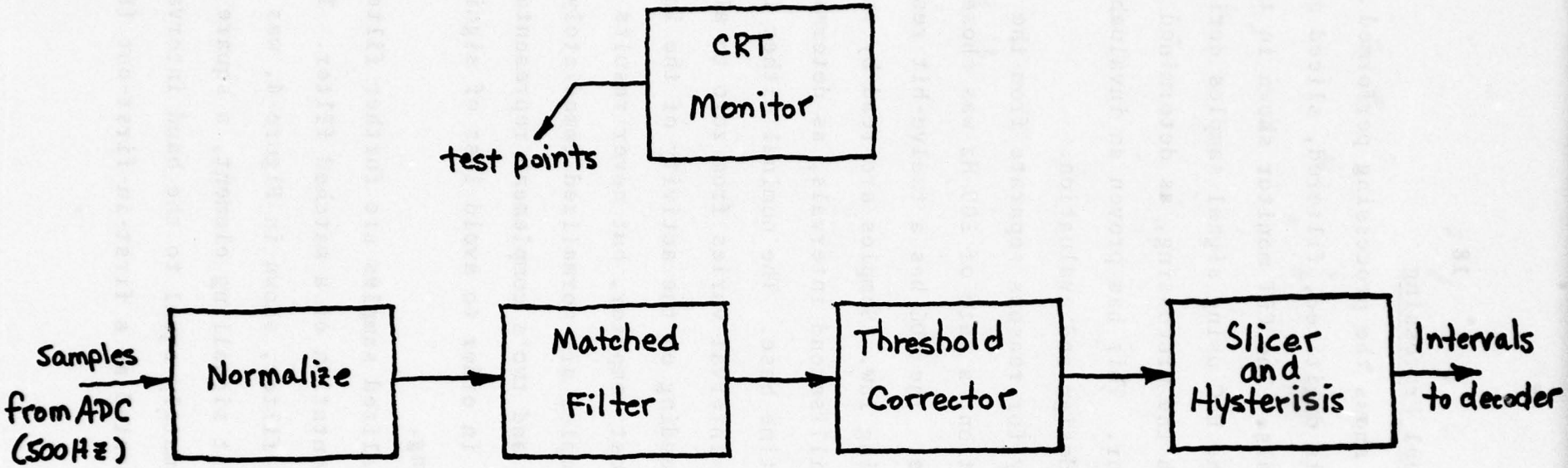


Figure 3. Digital Signal Processing

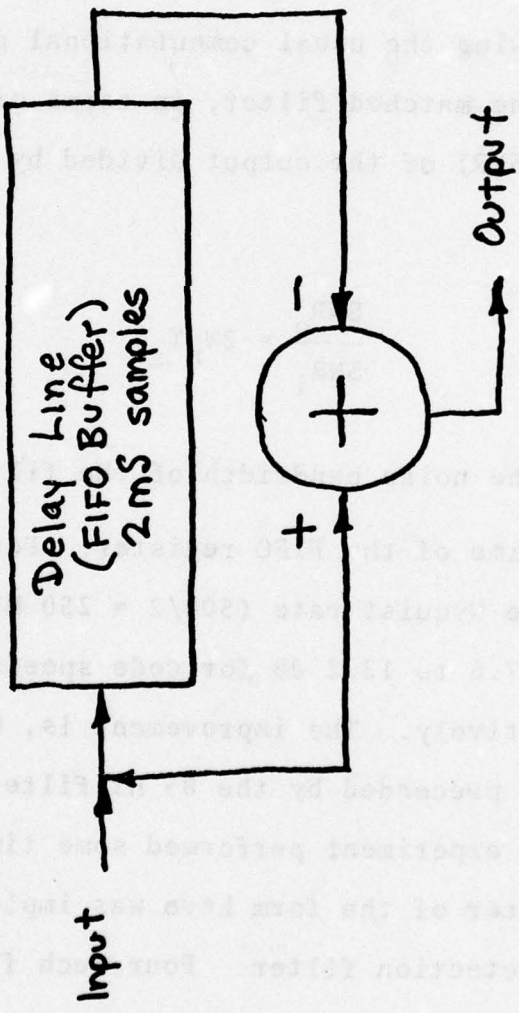


Figure 4. Matched Filter

in which the samples going into the filter are added to an accumulator, while the samples coming out are subtracted. It was not considered worthwhile, in view of the complexity required, to match the filter integration time exactly to the baud rate of the signal, as determined by the following parameter estimation algorithms. Instead, the integration time is fixed as a function of the nominal code speed, as determined by the operator. Experience shows that this portion of the system performs well over at least an octave in code speeds.

Following the usual computational procedures [16], the gain of the matched filter, in terms of the signal-to-noise ratio (SNR) of the output divided by the SNR at the input, is

$$\frac{SNR_o}{SNR_i} = 2W_n T_s,$$

where W_n is the noise bandwidth of the filter and T_s the integration time of the FIFO register. For inputs band-limited at the Nyquist rate (500/2 = 250 Hz), the gain varies from 17.6 to 12.2 dB for code speeds from 10 to 35 wpm respectively. The improvement is, of course, less dramatic when preceeded by the 85 Hz filter of the TU.

In an experiment performed some time age [5], a matched filter of the form here was implemented as a synchronous predetection filter. Four such filters were used

in opposing quadratures and vector-combined at their outputs to a square-law detector. In another experiment a synchronous digital phase-lock filter/detector was implemented. Although delivering performance superior to the postdetection filter described here, these devices were very hard to tune (the bandwidth was less than 20 Hz) and consumed virtually all the available computer cycles.

As will be seen, the intent of the digital signal processing is to derive successive time intervals during which the transmitted carrier signal is on or off. Normally, this would be done using a slicer, so that signals above the slicer threshold would cause one of two output signals, while signals below the threshold would cause the other. The problem here is that the optimum slicer threshold varies continuously with time, due to signal amplitude variations and noise. Although various methods have been used to estimate the optimum slicer threshold following a matched filter [16], most require considerable computation, which is impractical in the equipment here. The one described below is simple and effective, although little can be said about its theoretical performance.

Under multipath propagation conditions the received signal amplitude is distributed according to the classic Rayleigh or Rice distributions, and variations exceeding the system dynamic range can be expected to occur. The receiver AGC circuitry compensates for these variations to some extent; however, the variations are still too great to permit

a simple fixed threshold to be used. A device which has proved very effective for this purpose in conventional analog-type RTTY equipment is called an automatic threshold corrector (ATC).

The ATC is usually implemented in an analog fashion using a network of resistors, capacitors and diodes, as shown in Figure 5. In the system described here this device is implemented in a digital algorithm having similar characteristics. The RC time constants indicated are implemented in the usual way where, if e_k represents a voltage across a capacitor at sample (time) k, then the voltage at the next sample is

$$e_{k+1} = e_k + Ci\Delta t$$

where i is the current into the capacitor, C the value of the capacitor and Δt is the time between samples (2 mS). A time constant of about two seconds is used in the algorithm, representing a compromise effective under the majority of observed conditions.

It should be noted in passing that the design of the ATC could probably be improved to good effect. The conditions under which this circuit operates vary dramatically in the face of occasional ignition and atmosphereic-discharge impulses, Gaussian-distributed random noise, biases due to inband interference and deep signal fades and flutter.

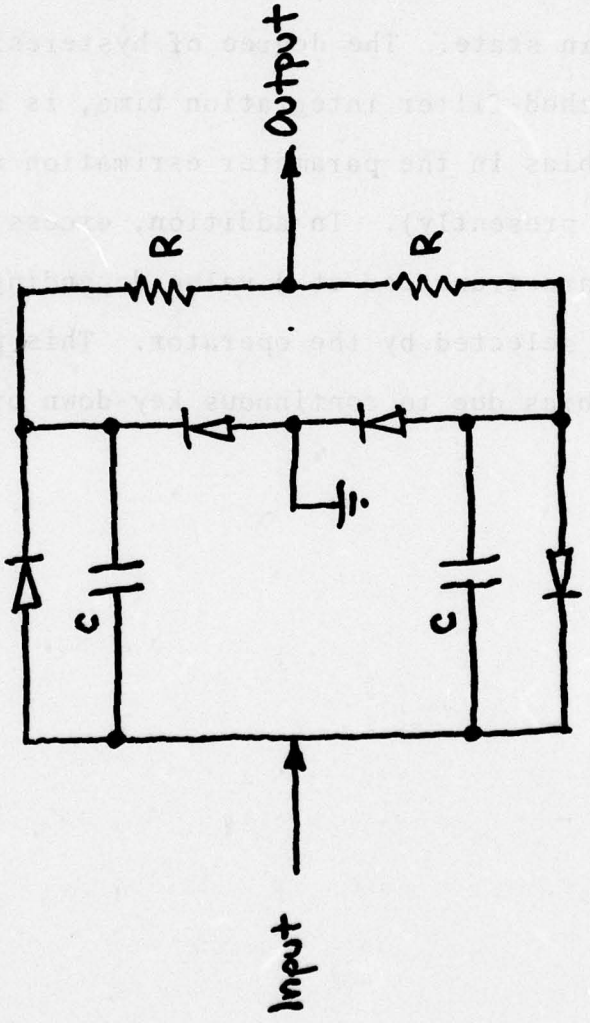


Figure 5. Automatic Threshold Corrector

The slicer compares the instantaneous amplitude of the signal with the slice level established by the ATC and produces a binary output signal for each sampling interval. If the signal following the matched filter is greater than the slice level a binary one is produced; otherwise a binary zero is produced. A measure of hysteresis is introduced into this decision to attenuate short noise bursts, such as produced during long key-up intervals when the receiver automatic gain control recovers to the maximum-gain state. The degree of hysteresis, a function of the matched-filter integration time, is set to prevent excessive bias in the parameter estimation algorithms (described presently). In addition, excessively long intervals are truncated at a value depending on the nominal code speed selected by the operator. This prevents excessive bias due to continuous key-down or key-up conditions.

2.3 Parameter Estimation

In order for the decoding processes to operate correctly, it is necessary to extract certain parameters from the received mark/space intervals. These include the interval estimates for the two kinds of marks (dot, dash) and three kinds of spaces (element, character, word) used in the Morse code.

As explained above, it is generally agreed that for well-formed code these intervals are all related by a set of ratios, so that, in principle, it is only necessary to estimate a single parameter which might be called the code speed. The code speed is related to the shortest mark interval (dot) or, equivalently, the shortest space interval (element space) by means of the formula [12]

$$\text{code speed (wpm)} = \frac{1154}{\text{shortest interval (mS)}} .$$

Code speed is usually expressed in words-per-minute (wpm) with values in the 15-30 wpm range representative of skilled operators.

Unfortunately, the ideal ratios are seldom observed in practice; and, moreover, distortions in successive intervals are not independent. In fact, the intervals can become so distorted that correct classification of successive intervals is impossible using a simple linear classification approach [4,14]. Under these conditions a single parameter is insufficient and separate parameters must be estimated

for some or all of the interval types.

Several techniques have been suggested to deal with this problem. In connection with MAUDE it was noticed that classification of space intervals could be significantly improved by a simple linear transformation in which a fraction of a previous mark duration is added to the next successive space duration. This attempts to correct the often observed tendency for the Morse operator to shorten the word and character spaces when the last element of the preceeding character is a dash.

More recently Guenther [4] and Howe [14] have used pattern-classification techniques to show in more detail the dependencies of successive elements. In this technique successive mark/space interval pairs are plotted on the two-dimensional plane, as indicated in Figure 6. Classification amounts to determining to which of the sets s_1 - s_6 an observed pair belongs. This can be done by determining a set of boundaries, called decision thresholds, which partition the signal space into non-overlapping regions (see Figure 6). This is a classic problem in pattern recognition and is studied in detail elsewhere [15].

Since the parameters of the manual Morse process are not stationary in time, due to the characteristics of different operators sharing a common circuit, determining the decision thresholds must be a dynamic procedure. In practice, the Morse decoder is presented with relatively short transmissions, of no more than a minute, and often

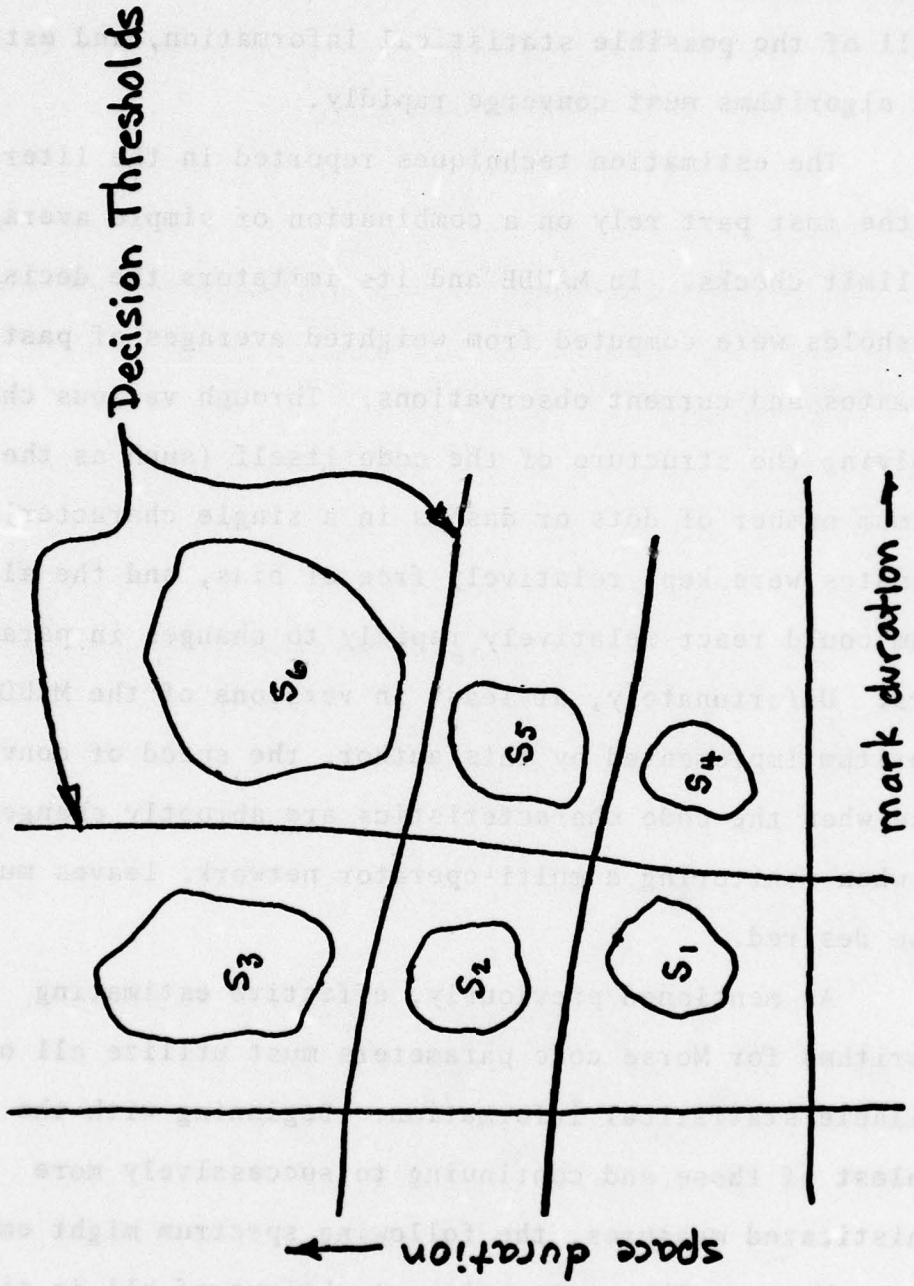


Figure 6. Signal Space

much less than this. During such transmissions often only a few characters are sent, and the code parameters can vary widely. Thus, a premium must be placed on the utilitzation of all of the possible statistical information, and estimation algorithms must converge rapidly.

The estimation techniques reported in the literature for the most part rely on a combination of simple averaging and limit checks. In MAUDE and its imitators the decision thresholds were computed from weighted averages of past estimates and current observations. Through various checks involving the structure of the code itself (such as the maximum number of dots or dashes in a single character) the estimates were kept relatively free of bias, and the algorithm could react relatively rapidly to changes in parameters. Unfortunately, at least in versions of the MAUDE algorithm implemented by this author, the speed of convergence when the code characteristics are abruptly changed, say when monitoring a multi-operator network, leaves much to be desired.

As mentioned previously, effective estimating algorithms for Morse code parameters must utilize all of the available statistical information. Beginning with the simplest of these and continuing to successively more sophisticated measures, the following spectrum might emerge:

Means. The most obvious technique of all is simply to calculate running weighted averages of the marks and spaces. This average can be used directly to classify the

dot and dash marks and, with some fiddling, to classify the element, character and word spaces. Experience with this technique has shown that it is grossly inadequate and subject to serious bias when presented with certain character sequences like 0 (five dashes) or 5 (five dots).

Linear Filtering. This suggests techniques such as Weiner smoothing and Kalman filtering [6]. Apart from the fact that the various statistics required (such as the variances) are not readily available, this technique does not appear to do well in the face of sudden transients in parameter values. In addition, the extensive computations required appear to preclude use of these techniques in real time.

Statistical Approximation. Considerably more sophisticated statistical estimation techniques including quasi-Bayes and statistical approximation methods have been applied to the problem of estimating Morse signal parameters [14]. Unfortunately the computational complexity required of these techniques leaves considerable doubt that operation in real time would be possible without prohibitively expensive computing resources.

Nonlinear Techniques. Various properties of the Morse code itself can be used to augment the linear filter approach. For instance, a sequence of seven or more marks cannot be a Morse character and the longest of the intervening spaces must be a character or word space. MAUDE used this technique to pre-classify certain intervals so as to

avoid bias in the averaging process. Similarly, certain invalid mark/space sequences can be corrected by looking for the most likely valid sequence that could have given rise to it. This aspect will be discussed further in a following section.

From the experience of these and the present authors it is evident that the advanced-design Morse decoder will require nonlinear estimation techniques. Here, the requirements extend only to estimation of the means of the mark/space intervals s_1 - s_6 shown in Figure 6. In the following section a new algorithm for estimation of these parameters will be presented. In subsequent sections algorithms involving Bayesian decoding and Viterbi decoding will be presented.

Upon close inspection of the statistics of typical Morse code intervals, an interesting and useful fact emerges. Although the various statistics can vary widely from operator to operator and even moment to moment, certain ratios are more or less invariant. For instance the dot/dash ratio is a characteristic of the operator and the keyer type, as is the element/character space ratio. The ratio of dot to element space is, however, due more to equipment design and operating characteristics than the individual operator. Finally, and this is the most important observation, the best estimate for all parameters occurs when a set of samples are available in which the ratio of the maxima to the minima is close to the ideal.

The rationale supporting these as "natural" characteristics is not hard to see. Given a few samples concentrated in clusters whose means are in ratios near to the ideal value, it can be assumed that the sender is taking some care with his element formation and that the samples are relatively free from noise and distortion. Accordingly, the samples from such a set are good candidates for classification and updating the estimates for each of the parameters $s_1 - s_6$.

Early experiments in using measures such as the variance of a set of samples to detect good clusters did not work well in practice. What was needed was a better way to classify the samples prior to averaging. The algorithm finally evolved, called Ratio-Weighted Estimation (RWE), was suggested in part by the above observations and in part by the properties of the Morse distribution [3].

A diagram of the RWE algorithm is shown in Figure 7. The algorithm operates on samples developed by the digital signal processing components described previously to produce a pair of estimates called max and min. It is assumed that each sample represents a value of one of two random variables belonging to either of two distributions having means related according to a specified ratio. The values of max and min developed by the algorithm are intended as the estimate of the means of the greater and lesser distributions respectively.

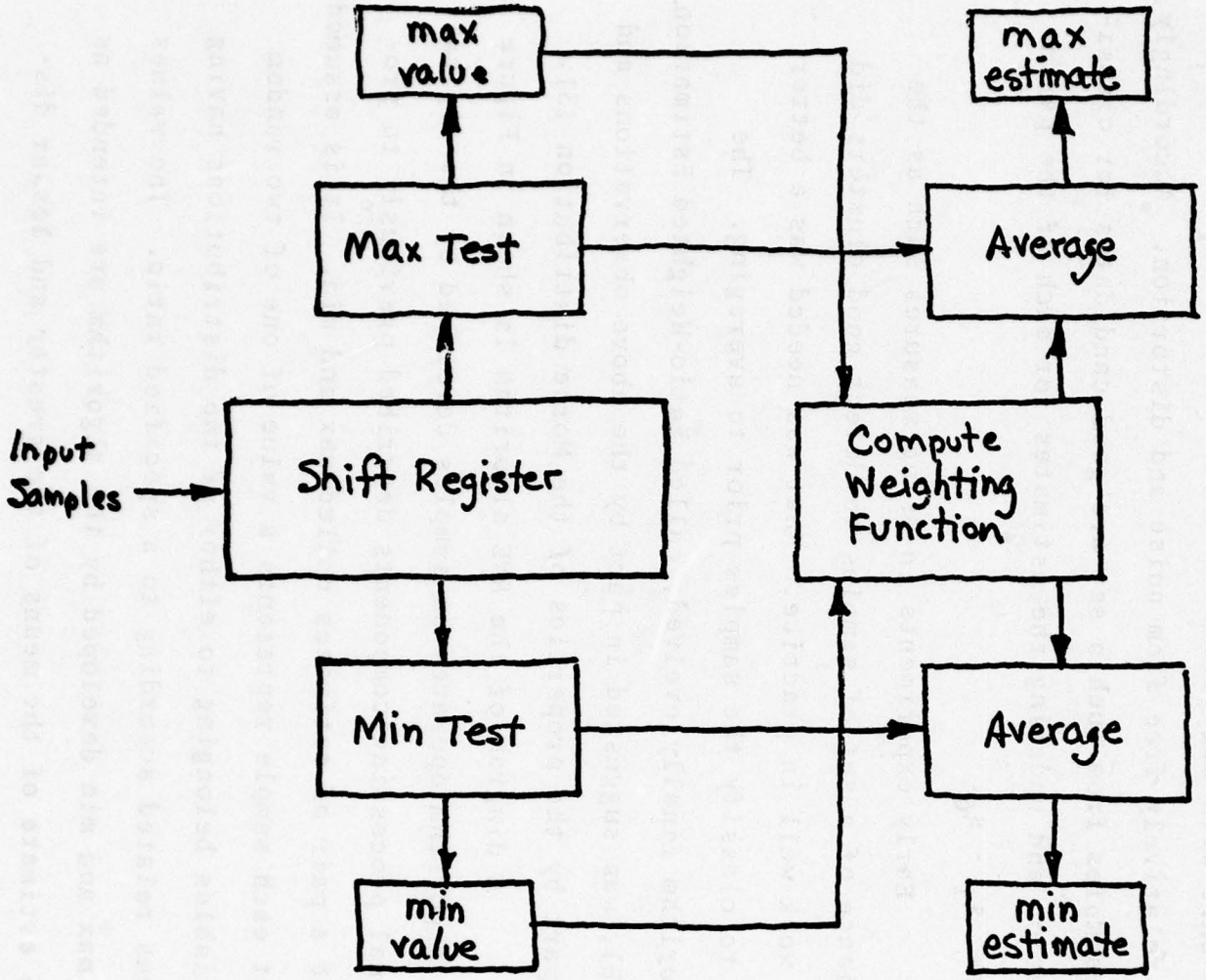


Figure 7. Ratio-Weighted Estimator

The operation of the algorithm is as follows. Incoming samples shift through the shift register one at a time and are discarded upon leaving the register. At each shift the new sample is compared with the samples already in the register. If the new sample is one of the largest or smallest of those already in the register, then it is assumed to belong respectively to the greater or lesser distributions and is averaged into the corresponding max or min estimate.

Meanwhile, the ratio between the maximum and minimum in the set consisting of the samples already in the register, together with the new sample, is computed and used in the calculation of a weighting function which determines the influence of the new sample on the estimates already computed. Ratios near the previously specified ideal cause the new sample to have a large affect, while ratios away from the ideal cause the new sample to have a lesser affect.

The weighting function used in the present system was selected from a number of substantially similar functions largely because it behaved in the required way, was simple and was easily computable. Let μ be the ratio previously specified as ideal and let k, p and q be parameters whose function will be apparent presently. Let

$$w = k + q \left[\frac{r - \mu}{r + \mu}\right]^p$$

where $r = M/m$ is the ratio of the current sample maximum M to the current sample minimum m. Then, using the current

estimate max or min as appropriate, together with the new sample maximum M' or minimum m', compute the new estimates as follows:

$$\max = \max + (M' - \max)2^{-w}$$

$$\min = \min + (m' - \min)2^{-w}$$

Note that the factor w, which is truncated to a integer, serves as a shift count and that the calculations must be done to 32-bit precision.

The RWE algorithm depends for its success on the balance achieved between the efficiency of the discrete classification performed by the shift register and max/min tests and on the "filtering" performed by the weight calculation. As can be seen, the real-time efficiency of separation of new maxima and minima determine the speed of convergence to step changes in code parameters. In order to provide the most rapid response, the register should be short, as isolated outlier samples can "deaden" the estimation process as they pass through the register. On the other hand, a register which is too short will not often contain samples from both distributions, so that the new max and new min samples will be less reliable. Figure 8 shows the classification efficiency under typical operating conditions with k, p and q parameter values to be specified presently.

The best balance can be achieved by varying the length of the shift register and the parameters of the weight

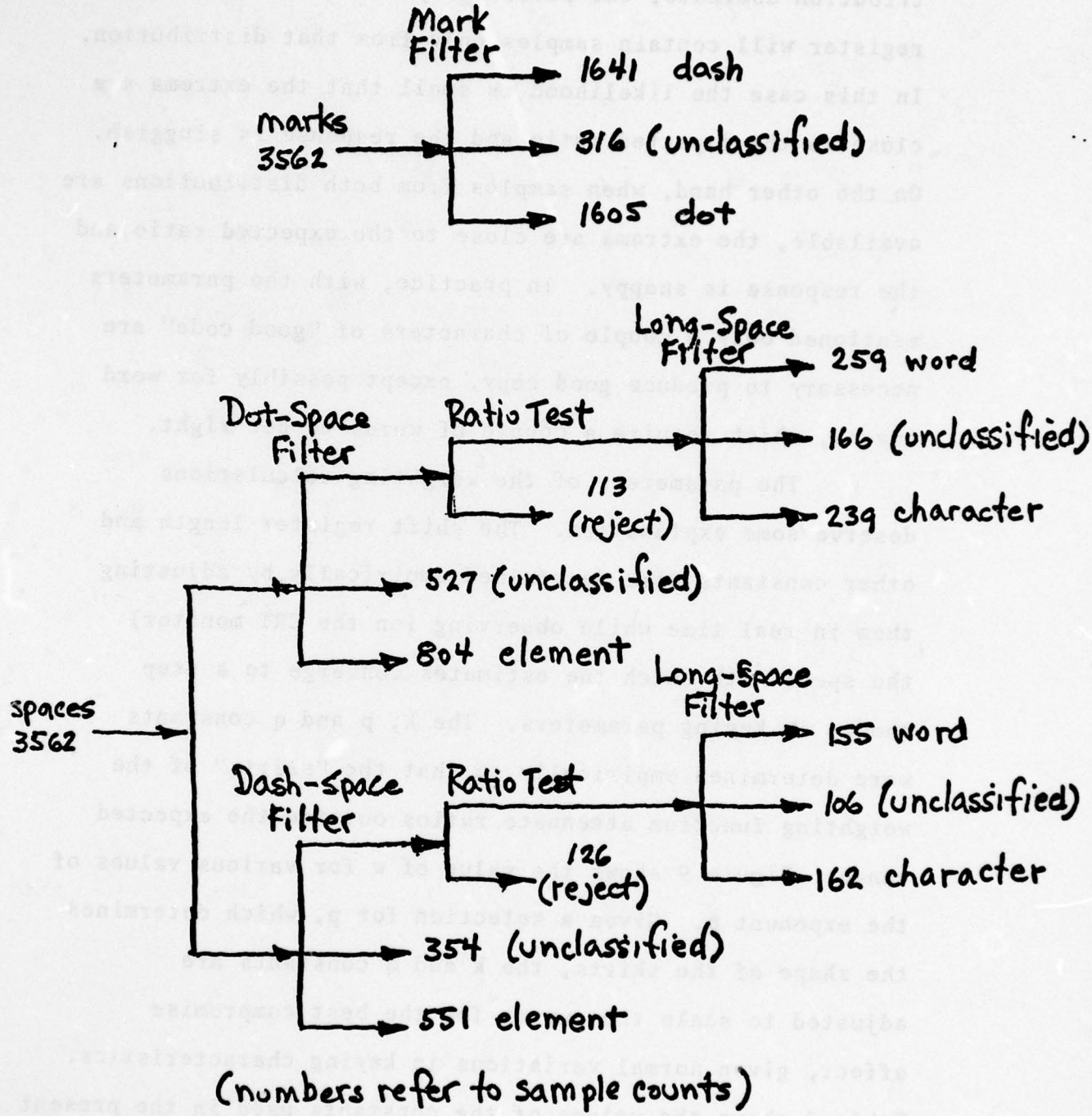


Figure 8. Classification Schematic of RWE Algorithm

calculations. Under conditions where samples from one distribution dominate, the probability increases that the register will contain samples only from that distribution. In this case the likelihood is small that the extrema are close to the expected ratio and the response is sluggish. On the other hand, when samples from both distributions are available, the extrema are close to the expected ratio and the response is snappy. In practice, with the parameters mentioned only a couple of characters of "good code" are necessary to produce good copy, except possibly for word spaces, which require a couple of words to get right.

The parameters of the weighting calculations deserve some explanation. The shift register length and other constants were determined empirically by adjusting them in real time while observing (on the CRT monitor) the speed with which the estimates converge to a step change in keying parameters. The k, p and q constants were determined empirically so that the "skirts" of the weighting function attenuate ratios outside the expected range. Figure 9 shows the value of w for various values of the exponent p. Given a selection for p, which determines the shape of the skirts, the k and q constants are adjusted to scale the curves for the best compromise effect, given normal variations in keying characteristics. Table 2 shows the values of the constants used in the present algorithm. These empirically determined constants could

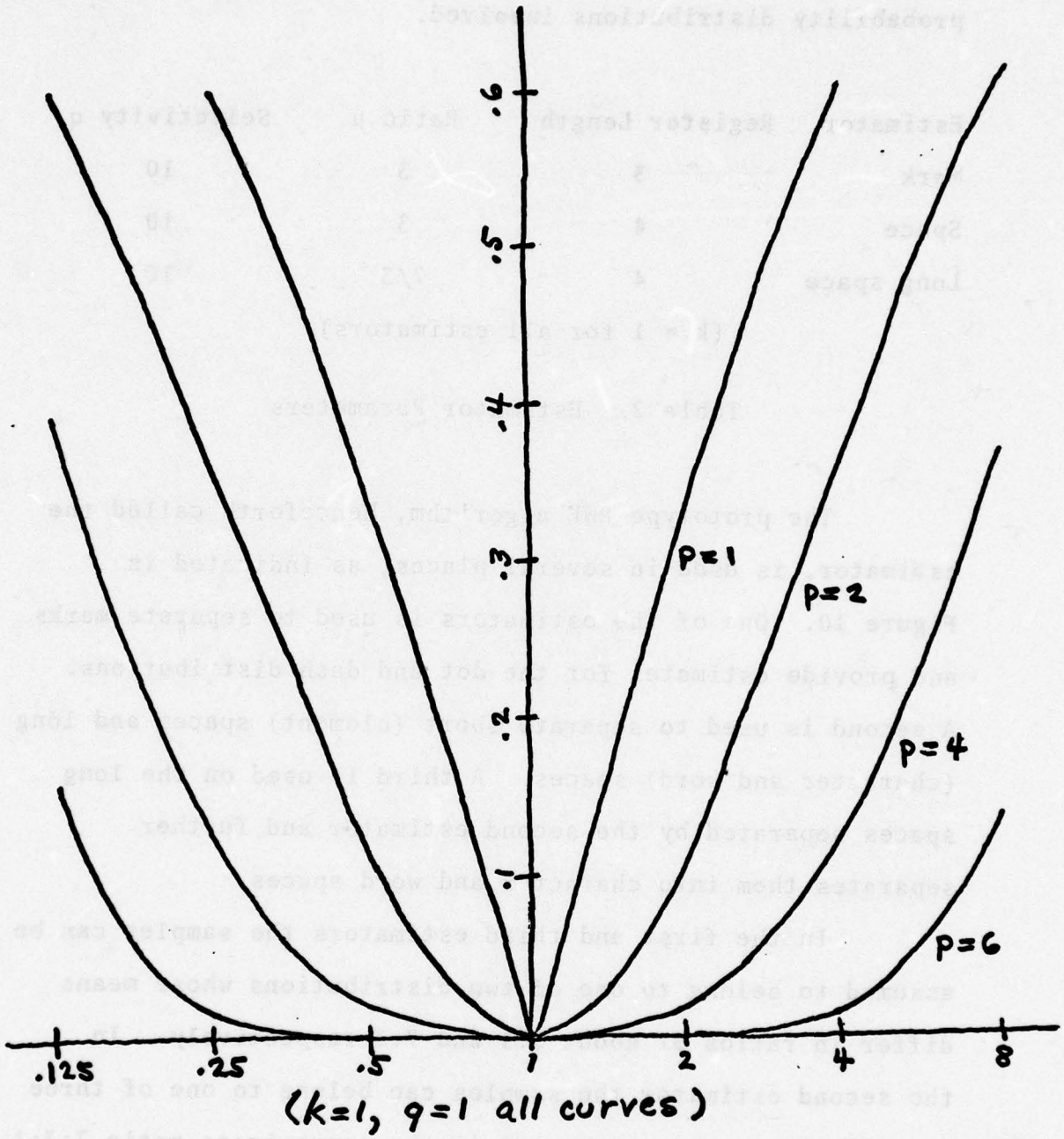


Figure 9. Weighting Function

undoubtably be refined through further analysis of the probability distributions involved.

Estimator	Register Length	Ratio μ	Selectivity q
Mark	3	3	10
Space	4	3	10
Long space	4	7/3	10

(k = 1 for all estimators)

Table 2. Estimator Parameters

The prototype RWE algorithm, henceforth called the estimator, is used in several places, as indicated in Figure 10. One of the estimators is used to separate marks and provide estimates for the dot and dash distributions. A second is used to separate short (element) spaces and long (character and word) spaces. A third is used on the long spaces separated by the second estimator and further separates them into character and word spaces.

In the first and third estimators the samples can be assumed to belong to one of two distributions whose means differ in ratios of about 3:1 and 7:3 respectively. In the second estimator the samples can belong to one of three distributions whose means are in the approximate ratio 7:3:1. The discrete ratio test shown in Figure 10 separates samples from the larger two of these distributions from the smallest distribution. Although this test could be improved, perhaps

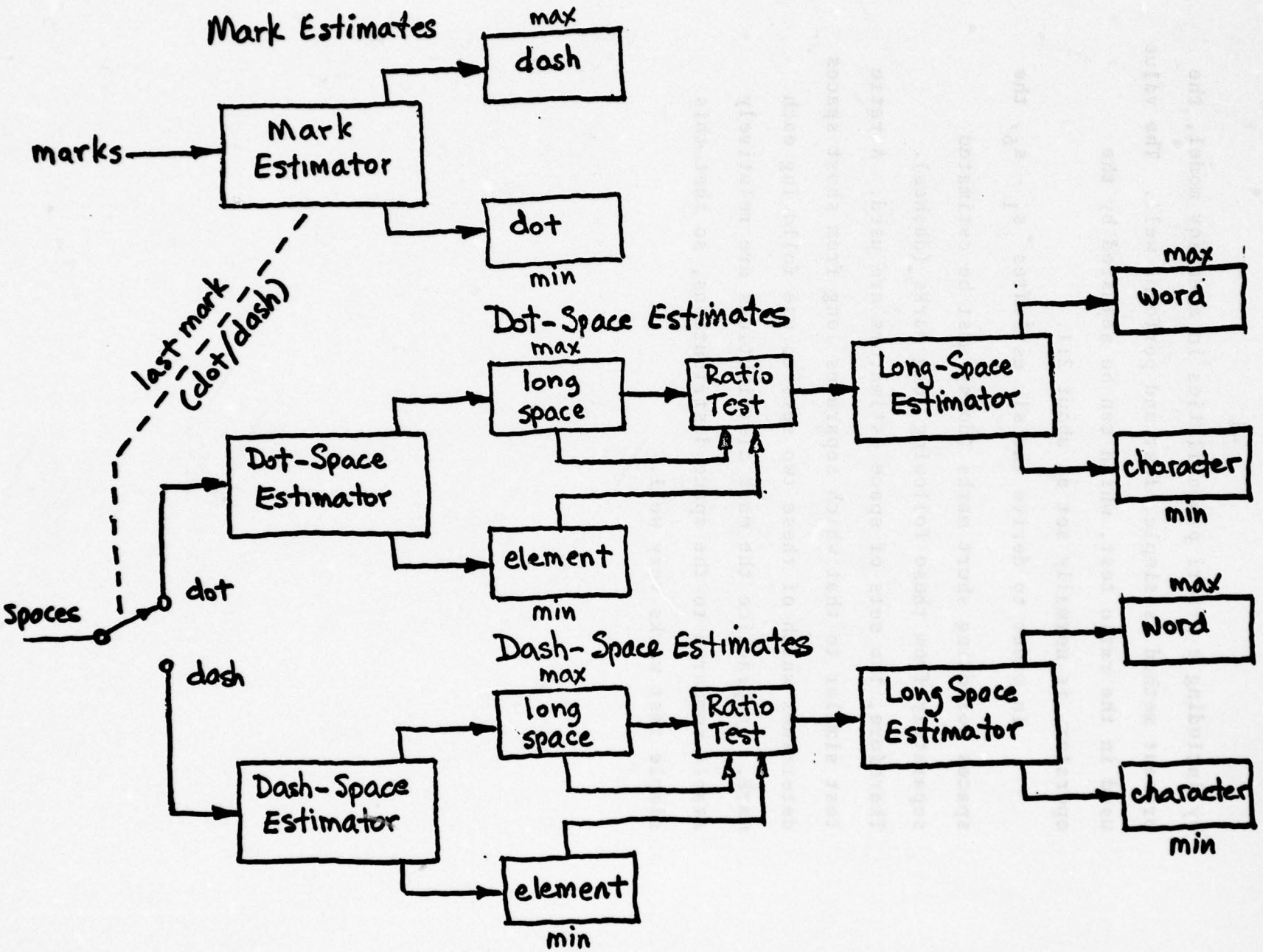


Figure 10. Parameter Estimator Configuration

by including a-priori probabilities in a Markov model, the present method is simple, fast and performs well. The value used in the ratio test, which can be adjusted by the operator, is normally set at about 2:1.

In order to derive the six estimates s_1 - s_6, the spaces following short marks (dots) must be estimated separately from those following long marks (dashes). Therefore, two sets of space estimators are used. A ratio test similar to that which separates long from short spaces determines which of these two sets to use following each mark. In practice the mark distributions are relatively stable compared to the space distributions, so that this simple test works very well.

2.4 Bayesian Decoding

Once the parameter-estimation process has converged to an approximation of the various mark and space parameters, the received sequence of mark/space pairs can be classified according to the principles of optimum receiver design. In fact, the needs of the Viterbi algorithm, to be described later, require only a set of probabilities of each received mark/space pair given each of the six estimates. The method described in this section is adapted from Wozencraft and Jacobs [7].

There are six allowable combinations of mark/space pairs $\{m_i\}$ corresponding to a selection of one of the two mark durations and one of the three space durations. Thus, the transmitted signal is characterized by a set of six signal vectors $\{s_i\}$ corresponding to each of the six symbols $\{m_i\}$ in turn. At the receiver the random vector r is received. Its problem is to assign each value ρ of r to one of the six symbols $\{m_i\}$, or equivalently $\{s_i\}$, which may have given rise to it.

For the purposes here, the a-priori probabilities of the symbols $\{m_i\}$ are known and the conditional probability of the received vector ρ given each of these symbols can be calculated. Thus, the optimum receiver can maximize the a-porteriori probability in the following way:

Let $P[m_i]$ be the a-priori probability of the symbol m_i and let $p_r(\cdot|m_i)$ be the conditional density function of r given that the vector s_i corresponding

to the symbol m_i was transmitted. If the random vector r extracted by the receiver takes the value ρ, the a-posteriori probability $P[m_i | r = \rho]$ specifies the probability that the s_i corresponding to m_i was transmitted. As each value ρ is extracted, the optimum receiver calculates $P[m_i | r = \rho]$ for each of the six symbols $\{m_i\}$ and chooses the m_i associated with the maximum value.

Consider the signal space shown in Figure 11, where the ϕ_1 axis corresponds to mark durations and ϕ_2 axis corresponds to space durations. By the technique described previously, estimates for the transmitted mark/space pairs have been made and the transmitted vectors $\{s_i\}$ can be assumed as shown. The received vector ρ is to be classified as arising from one of the transmitted vectors s_i. The random vector $n = \rho - s_i$ is assumed to be due to the noise process, which is determined by the local receiving conditions in effect at the time of observation.

The parameters of the noise process are extremely complicated and difficult to estimate. As evident from off-the-air observation and measurement, atmospheric noise bursts, cochannel interference and multipath fading all contribute to distortions of the transmitted element durations. The filter and slicer, which normally precede the detector, also affect these distortions in complicated ways. Furthermore, in the case of manually-generated Morse transmissions, the element durations generated by the operators are themselves random vectors distributed according to the speed of

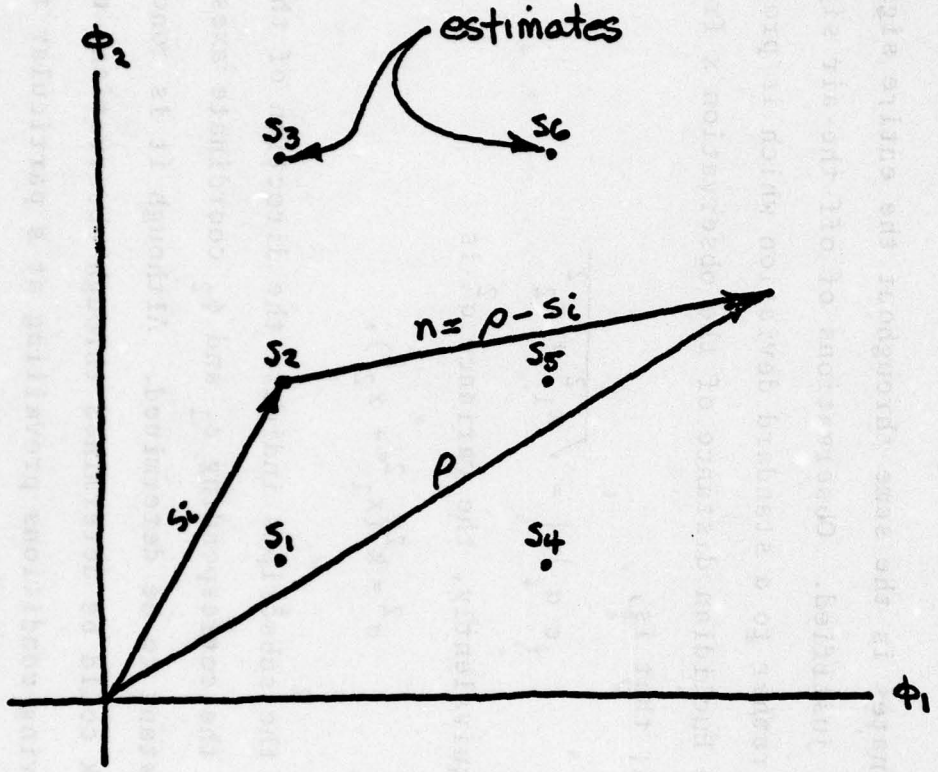


Figure 11. Signal Space Vectors

sending, the type of key used and the individual operator's characteristics.

Nevertheless, it is inviting and useful to model the noise process as a zero-mean Gaussian process, statistically independent of the signal and the coordinate, with specified standard deviation σ. However, the conventional assumption that the standard deviation in the ϕ_1 and ϕ_2 coordinates is the same throughout the entire signal space is not justified. Observations of off-the-air signals point rather to a standard deviation which is proportional to the Euclidian distance of the observation x from the origin; that is,

$$\sigma \alpha |x| = \sqrt{x_1^2 + x_2^2}$$

or, equivalently, the variance σ^2 is

$$\sigma^2 = k^2(x_1^2 + x_2^2),$$

where the subscripts indicate the direction of the components along the corresponding ϕ_1 and ϕ_2 coordinate axes and k is a constant to be determined. Although it is conceivable that k could be determined through observations under the receiving conditions prevailing at a particular time, the improvement obtained would be minor. It has been determined that a value of $k = 0.15$ is a good compromise and agrees well with a number of observations under widely varying receiving conditions and operator characteristics. Howe [14]

has estimated a similar value, although under different conditions.

Accordingly, $|\rho - s|$ can be approximated by the zero-mean Gaussian distribution:

$$p_n[\rho - s_i] = \frac{1}{2\pi\sigma_i^2} \exp\left(-\frac{|\rho - s_i|^2}{2\sigma_i^2}\right), \qquad (1)$$

in which

$$\sigma_i = k^2|s_i|^2.$$

These expressions are not exact, of course; but, in view of other assumptions made here, should be accurate enough for the purpose.

The a-posteriori probabilities can be calculated by Bayes' rule:

$$P[m_i|\rho] = \frac{P[m_i]p_r(\rho|m_i)}{\sum_j P[m_j]p_r(\rho|m_j)}. \qquad (2)$$

Equation (1) in fact yields $p_r(\rho|s_i)$, which is equivalent to the $p_r(\rho|m_i)$, or likelihood, needed in Equation (2). For each value of i we note that the denominator of (2) is a constant and, since we are only looking for the maximum, can be deleted. Therefore, the optimum Bayes receiver need only calculate

$$\frac{P[m_i]}{2\pi k^2 s_i^2} \exp\left(-\frac{|\rho - s_i|^2}{2k^2 s_i^2}\right) \qquad (3)$$

for each i and pick the m_i corresponding to the maximum value. By using logarithms this expression can be simplified still more, as explained in the following section.

It is possible to decode and print the English alphabet symbols directly from the sequence of mark/space symbols m_k (k = 0,1,...) classified according to the principles above. For the Morse code the a-priori probabilities needed by Bayes' rule can be approximited from Figure 8, in which it is apparent that:

1. The frequency of occurance of dots is about the same as dashes.
2. The frequency of the various types of spaces following a dash is about 0.7 that following a dot.
3. The frequency of long spaces (word and character) following a dash or dot is about 0.8 that of element spaces.
4. From other data [13] it can be determined that the freqneucy of word spaces following a dash or dot is about 0.2 that of character spaces.

Writing down the corresponding set of simultaneous linear equations and solving, the values shown in Table 3 are obtained. Although these values could be easily incorporated directly into a Bayes decoder for the experimental system, the Viterbi decoder, which is the normal algorithm used, requires only the likelihoods (3). The values shown in Table 3 are more useful to check the operation of the parameter estimation and decoding algorithms.

	Space	
Mark	dot	dash
element	.327	.229
character	.218	.153
word	.044	.031

Table 3. A-Priori Mark/Space Symbol Probabilities

2.5 Viterbi Decoding

If the a-priori probabilities of successive mark/space symbols were independent, the optimal decoder for the Morse code would simply choose the symbol with the greatest a-posteriori probability, as calculated in the previous section. However, it is possible to model the Morse decoder process as a Markov process and take advantage of its probabilistic structure in order to decrease the error rate. In this section a model is described which uses the Viterbi algorithm [9] to maximize the probability of correct decoding for sequences of symbols in much the same way as the Bayes algorithm maximizes this probability for individual symbols.

The Viterbi algorithm has been implemented in a number of high-performance digital signal detection systems and applied to other problems as well [8]. Bell [6] implemented the Viterbi algorithm as a component in a Morse decoder in a manner similar to that described here. However, his implementation is substantially different in the following ways:

1. The algorithm described here does not use signal amplitudes when calculating likelihoods. Use of this information under severe multipath conditions has proven worthless and even a source of error. Bell's algorithm includes the use of signal amplitudes.

2. The algorithm described here operates in real time and does not require recording and postprocessing. Bell's algorithm does not operate in real time.
3. The likelihoods of the input symbols are computed in a two-dimensional (mark/space) signal space using the parameter estimating procedures described in the previous section. Bell's algorithm uses rather arbitrary "figures of merit" derived using smoothed values of preceeding samples.
4. The full probabilistic structure of the Morse process is incorporated, including a-priori transition probabilities for all valid received sequences.

The Viterbi algorithm described here operates in the manner described by Forney [8] and his notation will be used. The algorithm involves three components: a Markov model in the form of a graph, from which the a-posteriori probabilities of each sequence can be calculated, a set of a-priori probabilities and state-transitions probabilities, and a statement of the mechanisms to construct and delete candidate sequences as input mark/space symbols arrive.

A discrete Markov model of the Morse process can be constructed in the following way: Let m designate one of the M states $1 \leq m \leq M$ of the state space. The state of the process at time k is designated x_k, with the initial state at time zero designated x_0 and the final state at time k designated x_k. In the cases of interest here, the

final state will be the initial state, so that $x_k = x_0$. Starting at x_0, presumably just following a character or word space, the decoder is presented with a sequence $z = z_0, z_1, \ldots$ of symbols. Each of these symbols is represented as the mark/space coordinates of the received vector ρ (see preceeding section). We use $x = (x_0, x_1, \ldots, x_0)$ to designate a minimal circuit or cycle of states of the Markov process and $z = (z_0, z_1, \ldots)$ be the potentially infinite received sequence of mark/space symbols. The problem addressed by the algorithm described here is to map the sequence z onto a concatenation of cycles x so as to maximize the a-posteriori probabilities.

Following conventional dogma [8], we assume a first-order Markov process, where the probability that the state of the process at time k is

$$P[x_{k+1} | x_0, x_1, \ldots, x_k] = P[x_{k+1} | x_k];$$

that is, the process is memoryless. In the case here, a received mark/space symbol z_k is assumed dependent only on the state of the transmitted x_k; that is, subsequent symbols are independent. This assumption can be questioned, since the influence of the matched filter and threshold corrector could be expected to affect subsequent symbols in a complicated way. However, in view of the robustness usually associated with assumptions of this kind, we set

$$P[z|x] = \prod_k P[z_k|x_k].$$

The receiver needs to determine the state sequence x for which $P(x|z)$ is a maximum, for the mark/space symbol sequence corresponding to this state sequence is the optimum choice. Noting that, under the above assumptions,

$$P[x,z] = P[x]P[z|x]$$
$$= \prod_k P[x_{k+1}|x_k] \prod_k P[z_k|x_{k+1},x_k]. \quad (4)$$

Taking negative logarithms of both sides,

$$-\ln P[x,z] = -\ln P[x_{k+1}|x_k] - \ln P[z_k|x_{k+1},x_k]. \quad (5)$$

Since

$$P(x,z) = P(x|z)P(z),$$

we can maximize the a-posteriori probability $P(x|z)$ by maximizing (4) or, equivalently, minimizing (5). The later formulation has the advantage that "distances" between x and z can be conveniently calculated and compared during the operation of the algorithm.

The first term on the right-hand side of (5) can be recognized as the negative logarithm of the transition probability from state x_k to state x_{k+1}, which can be calculated in advance from the topology of the state graph and the a-priori symbol probabilities. The second term can be calculated by the methods of the preceeding section. The optimum decoder algorithm then computes values (5) for each of the six mark/space symbols possible to generate in

in the transition from state x_k to state x_{k+1} and adds these to the total accumulated so far as the distance between the particular assumed sequence $(x_0,x_1,\ldots,x_k)$ and the observed sequence $(z_0,z_1,\ldots z_k)$.

A simplified state transition diagram for the Morse code is shown in Figure 12. This diagram, actually a decoding tree [9] shows that state transitions for decoding each valid Morse character when started at the apex of the tree and presented with the dot/dash sequence corresponding to the character. Presumably, when the dot/dash sequence terminates at some state, say x_k, the character corresponding to the final mark element at x_k in Figure 12 is printed and the process returns to the initial state.

Actually, the full structure of the state transition diagram is rather more complicated. At each state provision has to be made for six emergent arcs, one corresponding to each mark/space symbol that may occur. Two of these, the dot-element space and dash-element space, lead to further nodes as shown in Figure 12. The remainder all represent the last symbol in the Morse character and lead to the initial state. Figure 13 shows the structure of a typical state in the diagram. Note the transition probabilities assigned to the various arcs, which of course must sum to unity.

The algorithm operates generally as follows. During the decoding process candidate sequences $(x_0,x_1,\ldots x_k)$

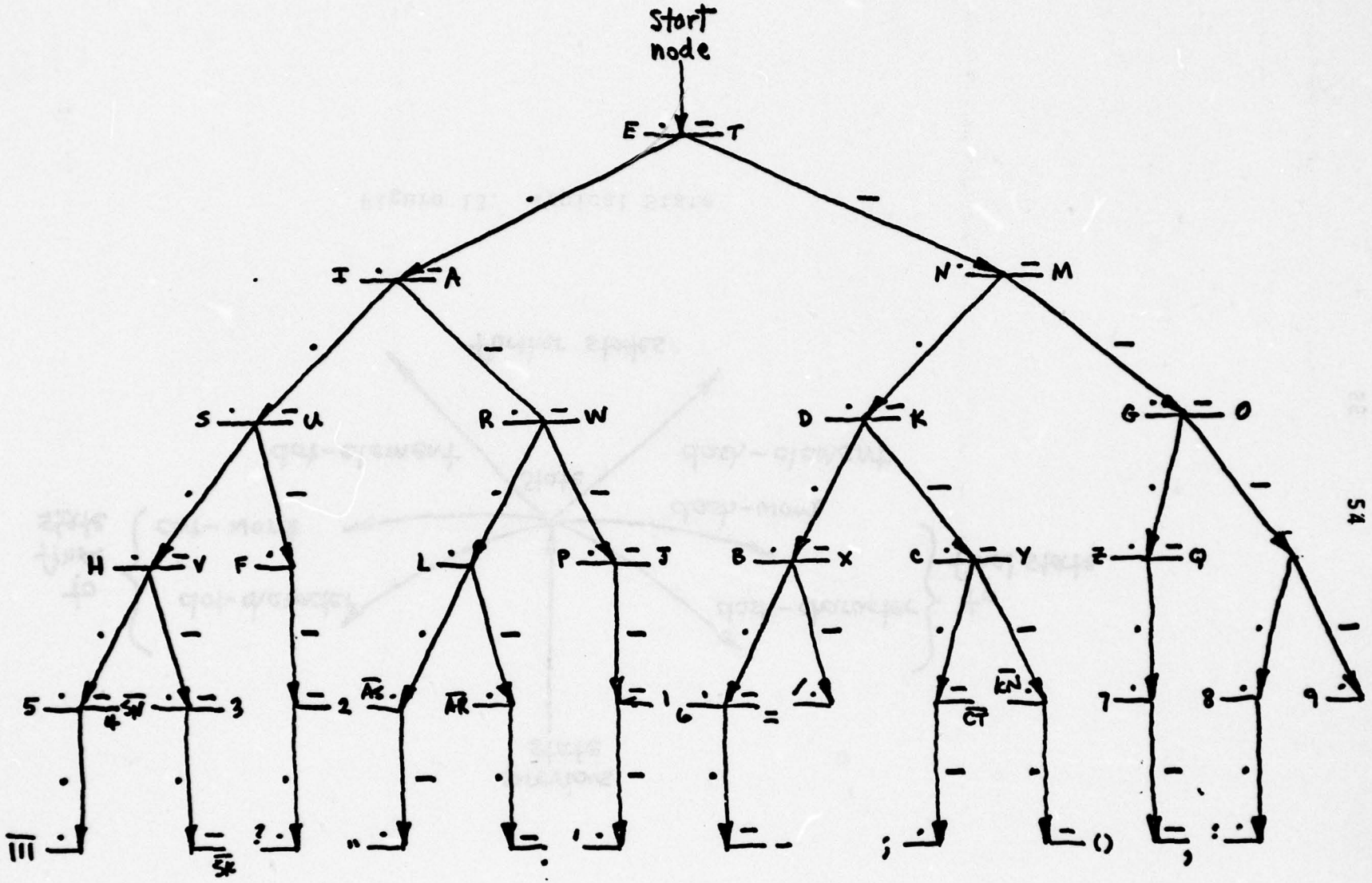


Figure 12. Simplified State Transition Diagram

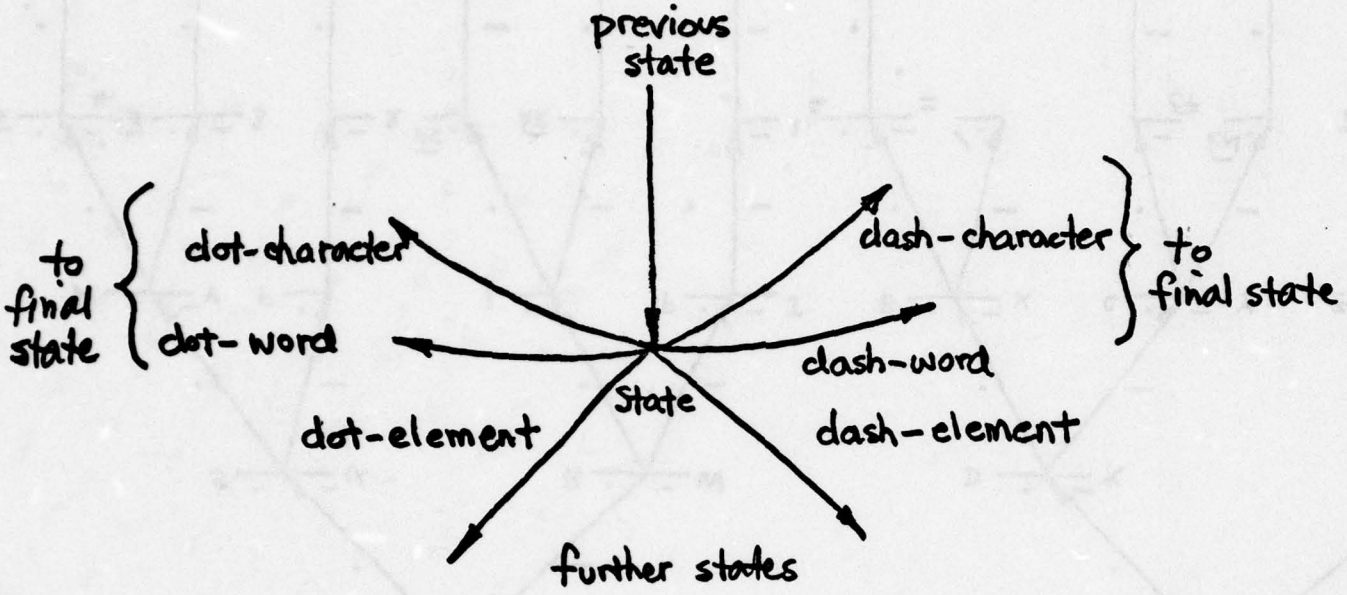


Figure 13. Typical State

accumulate as the state sequence proceeds. When two arcs in the state diagram join, all of the candidates accumulated on each path are deleted except the one of minimum distance, which is called the survivor. In general, there can be a survivor corresponding to each state.

However, there are two features of the Morse process which distinguish the algorithm described here from the usual Viterbi algorithm. First, there is only a single remergent state; that is, a state with more than one terminating arc, and this is simultaneously the initial and final state. Second, Morse code sequences have variable length, so that certain survivors (those terminating in the final state) can preclude further consideration of survivors elsewhere. These two features combine to considerably simplify the complexity of the algorithm.

The algorithm operates on a data structure which includes a set of nodes corresponding to the set of states. Each node contains a set of pointers linking it with its successor states, a set of pre-computed transition probabilities, and an accumulator cell, the function of which will be apparent shortly. Additional information, such as the character to be printed, is also associated with each node.

Figure 14 shows a flow chart for the algorithm. Initially, the accumulator corresponding to the initial state is cleared and the algorithm started in the state x_0.

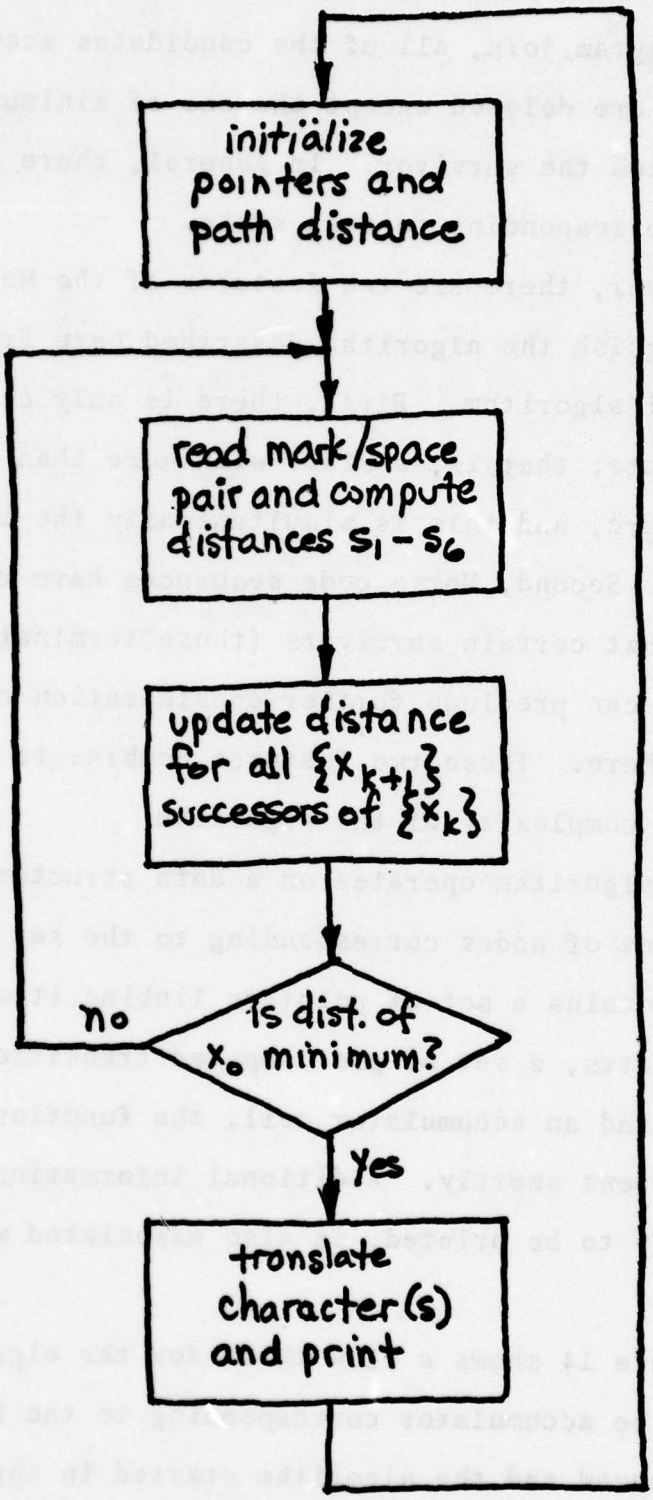


Figure 14. Decoding Algorithm

Assume that the algorithm has operated for some time and is currently processing the kth mark/space symbol z_k, as received via the operations described in previous sections. There will be a nonempty set of states $\{x_k\}$ corresponding to the nodes that have already been computed. Consider the set $\{x_{k+1}\}$ of states which will be computed at the kth symbol. For each of the states of $\{x_k\}$ the associated conditional probabilities $P[z_k|x_k]$ are computed for each of the six possible transmitted mark/space symbols. Then the corresponding distance functions (5) are calculated using these values along with the precomputed transition probabilities. Next the accumulators at the nodes corresponding to the (k+1)th states are set as the sum of the accumulator value for the kth state and the corresponding distance function. If the arcs leading to the next state are remergent (as they can be only for the final state) then only the smallest of the accumulator values computed so far is retained, as well as the character corresponding to the node at which this computation occured.

The details of the node calculations are as follows: as each mark/space symbol is produced by the digital signal processing components of the system, the first term in (5) must be computed for each of the six possible transmitted symbols. Recalling from Equation (1) in the previous section,

$$P[z_k|x_{k+1},x_k] = \frac{1}{2\pi\sigma_i^2}\left(\exp - \frac{|\rho_k - s_i|^2}{2\sigma_i^2}\right),$$

where ρ_k is the vector received at the kth position in the input sequence and s_i is the vector estimate corresponding to the transmitted symbol m_i.

In a preceeding section it was argued that $\sigma_i = k|s_i|$. Thus, if we make the transformation of variables

$$n = \frac{1}{k} \frac{|\rho_k - s_i|}{|s_i|}$$

on the right-hand side, we obtain

$$\frac{1}{2\pi} \exp(- \frac{n^2}{2}).$$

Taking the negative logarithm of this quantity we obtain

$$\ln 2\pi + \frac{n^2}{2}$$

The first term of this expression is a constant and can be discarded. The second can be easily calculated without the requirement to calculate logarithms.

The key to the operation of this algorithm is the mechanism by which survivors are selected at the final node. While at each step in the operations described above the survivor with the minimum distance is the only one retained, it is still necessary to determine whether or not to stop, print the character and resume at the initial state. This is done by the simple rule that, if the distance of the survivor at the final node after the kth step is less than the distance of the survivor at each of the $\{x_{k+1}\}$ successor

nodes, then stop and print; otherwise, delete the survivor at the final node and continue.

As described here, the operation of the algorithm is continuous and self-synchronizing. Due to the nature of the state transition diagram, only a fraction of the nodes need to be computed at each step. Table 4 shows the number of these nodes at each step; to these should be added the final node, which in general is recomputed for every step.

Stop	Successors	Nodes	Morse Characters
1	2	1	2
2	4	2	4
3	8	4	8
4	13	8	12
5	11	13	17
6	0	11	11

Table 4. Node Computation

It is evident that not all nodes correspond to valid characters. This is handled automatically by the transition probabilities. In the same manner attempts to decode "off the end" of the diagram are handled. Note that the $P[z_k|x_k]$ probabilities need be computed only once for each step and that disallowed transitions (transition probability zero) need not be considered at all.

3. Discussion and Comments

Although many of the algorithms and techniques described in this report have been subject to prior analysis and modelling, the conditions under which they operate in the actual Morse decoder are anything but well-behaved, in the sense commonly assumed. Thus, there is considerable doubt that the behavior predicted by analysis would agree very closely with that actually observed, in particular the relationship between the input signal-to-noise ratio (SNR) and the corresponding error rate. Accordingly, it was concluded that the most useful characterization of the system described here would be obtained through direct measurement of its behavior under certain experimental conditions. These experiments were of two types: those designed to test the filtering, detection and decoding algorithms using perfect code and bandlimited white Gaussian noise, and those designed to test the parameter estimation algorithms using off-the-air signals.

In order to evaluate the effectiveness of the filtering, detection and decoding algorithms, an experiment was set up where the Morse signals corresponding to random five-letter code groups was mixed with white Gaussian noise bandlimited to 500 Hz. The random sequence consisted of a potentially infinite sequence of groups of five letters followed by a space, where the letters were generated randomly with equal probability. A sample sequence of 108 groups was produced at 20 wpm, using the Morse code

generator which is part of the system, and recorded at 2125 Hz on analog magnetic tape. When the tape was played back the signal was added to the noise signal produced by bandlimiting a wideband noise source to 500 Hz centered on 2125 Hz.

The resulting signal was used to drive the system in place of the normal receiver output (see Figure 2). The system was then run on the sample sequence and the error rate recorded. The various level controls were set so that limiting in the TU did not occur and so that the signal remained in the TU passband at all times. For these tests the parameter estimating algorithm was disabled and the decoding algorithms set for 20 wpm and equi-probable symbol probabilities.

Two sets of tests were run, one with the system described here and another where the Viterbi algorithm was replaced by a simple fixed-interval classifier of the kind used in several previous systems (e.g. [2]). The fixed-interval classifier assigned a mark to the dot class if it is shorter than the average of the dot and dash estimate and the dash class otherwise. In a similar fashion spaces are classed as element, character and word spaces. No post-decoding correction algorithm, such as the alphabet test [2], was used in either system.

The error rates observed in each of these two sets of tests appear in Figure 15. Note the rather pronounced threshold, also observed in other data [4,14], below which

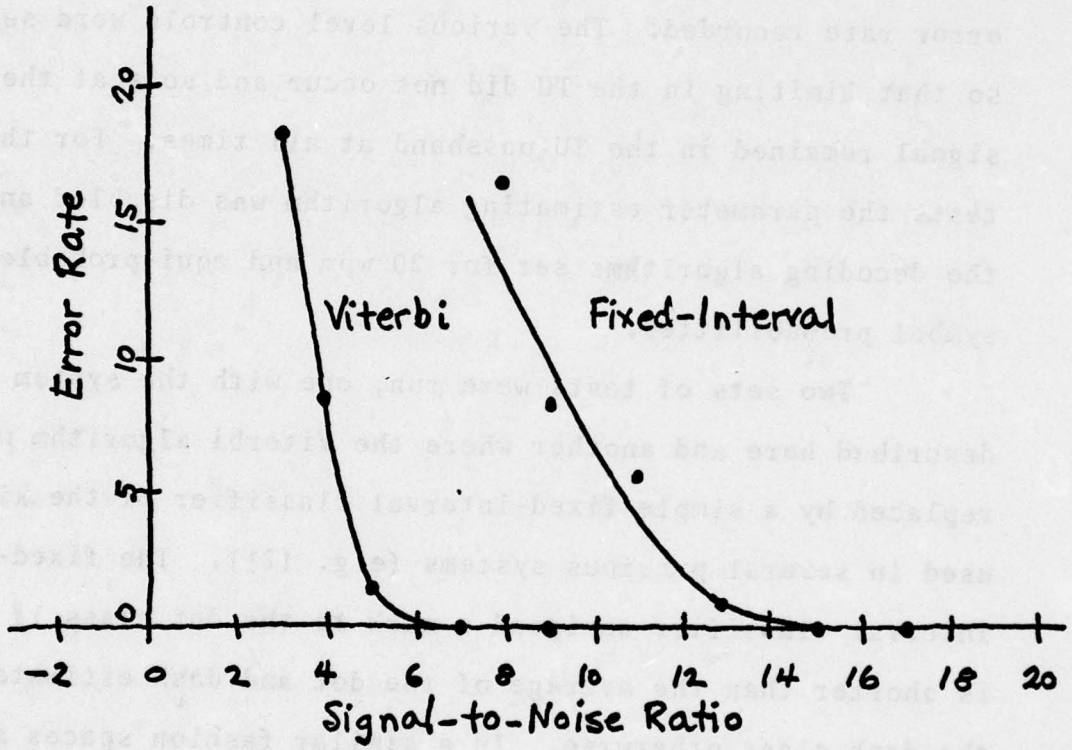


Figure 15. System Performance

copy is relatively good and above which copy degenerates badly. It is difficult to compare the system performance illustrated in the Figure with that of other systems such as Gold [2], Guenther [4], Bell [6] and Howe [14] because of the different methodology and experimental design used. Nevertheless, it is possible to compare these systems using as a criterion the SNR in an equivalent 2 kHz bandwidth required for a ten-percent error rate. Subtracting six decibels to account for the different noise bandwidth, the Viterbi system requires a SNR of about -2 dB, while the fixed-interval system requires about 3 dB. Data from other systems seem to indicate that these figures are too high by a few decibels. The difference is probably due to the crude equipment used to generate and filter the noise signal. (The generator was shot noise from an idling transmitter amplifier and the filter the intermediate-frequency filter of a receiver.)

In spite of the likely inaccuracies in the base level of the noise source, differences due to processing gains could be accurately measured. The approximate 5 dB gain of the Viterbi algorithm agrees quite well with that measured by Bell, as does the shape of both curves.

In a related series of tests the performance of the Viterbi system with and without the matched filter-threshold corrector and with and without the use of accumulated path distances was compared. In the case without the use of accumulated path distances the algorithm becomes simply

a Bayes decoder and the Markov structure of the Morse code is suppressed. As expected, the matched filter-threshold corrector gave a slight advantage, as did the Viterbi decoder, but not enough to be statistically significant, considering the test equipment and methodology used. A relatively small advantage when using a-priori probabilities was also noticed by Howe.

That the matched filter-threshold corrector did not markedly improve the performance could be expected, since the TU bandwidth is already quite narrow (85 Hz) and the theoretical SNR gain of the matched filter cannot be realized. However, the threshold corrector gains essentially nothing with the stationary sources used in the test, but greatly improves copy under actual conditions where varying signal levels due to multipath propagation are common.

The lack of significant advantage of the Viterbi over the Bayes algorithms is more significant. It is possible that, upon further and closer analysis, the considerable processing complexity of the Viterbi algorithm may not be justified, in view of the limited improvement possible. It is, however, clear that the use of probabalistic decoding and the use of a higher-dimensional state space does significantly improve the performance.

Tests such as the above, which are designed to measure the system performance under steady-state conditions with no distortion and well-behaved noise sources, are relatively easy to perform and yield data readily comparable

to other systems. On the other hand, tests to measure the dynamic system performance under conditions of varying operator and circuit parameters are much more difficult to design. Howe, for instance, recorded typical operators with varying degrees of skill and sending speeds. He also constructed a synthetic source which behaved statistically much like these operators. The experimental framework in which the work reported here was carried out made it impossible to invest the time and energy necessary to make proper dynamic measurements such as these. In particular, it was not possible to adequately investigate the analytical and experimental behavior of the RWE algorithm, which, obviously is a critical determinant of the system dynamics.

Nevertheless, considerable experience was gained by observing the system response to off-the-air signals prevalent on Amateur and Maritime frequencies. This was done by observing critical parameters on the CRT monitor, as well as error rates in the decoded text. Many hours of operation were accumulated using the system for two-way communication with stations in the Amateur service and receiving-only with coast and ship stations in the Maritime service. As used with the author's Amateur station, operation is completely automatic and manipulation of no station controls other than the terminal keyboard is required. In every case the performance of the system was far superior to that of previous implementations based on the MAUDE algorithms.

The experience gained so far points to a critical need for data in two areas: first, a study of the statistical parameters of the "typical" human operator using any of several kinds of keying devices. There appears to be several more-or-less distinct styles or dialects of Morse code, all using the same code but with different emphasis patterns or "swings." For instance a "bug" operator often falls into a sending pattern in which the dot frequency is rather too high for the dash interval. The variance of the dash and long spaces in such a pattern is rather high, yet the variance of the dot and element spaces, being mechanically generated, is very small. As Howe points out, it would be possible to significantly improve the speed of convergence of any parameter-estimation algorithm by using such information to update all the parameters as a function of changes in one of them.

This could be done in an interesting way by using the RWE algorithm "in reverse," so to speak. The idea is to allow the various parameters to float at their current values unless new data arrive which indicate a substantive change in all of the parameters. Most commonly this would occur when a number of new dots, dashes and element spaces arrive with intervals indicating that the word and character spaces should be significantly altered.

The ideal and observed ratios between these parameters can be used to compute a weighting function in the

same way as in the RWE algorithm. However, in this case the negative value of the function is used as the shift count. This has the effect that ratios near the ideal cause very little coupling between the new observation the remaining parameters, while ratios far from the ideal cause a more direct effect. The net effect is that a "good" estimate for a new dot or dash, for example, will tend to override old estimates for the long spaces only if these old estimates depart significantly from those indicated by the new estimate. Thus, the system has a "slop" in which the parameters are free to float within predefined ranges according to the confidence of new information.

The second area in which data are needed is in modelling and, possibly, simulation of the RWE algorithm. It should be possible to determine the various parameters of the estimator algorithms as a function of the statistics of the code variations. These parameters would, of course, depend on the statistical structure of the code elements encountered in actual use. For instance, Maritime coast stations frequently use the sequence "DE" to prompt a ship station to send its call sign. Unfortunately, the letters D and E both end with a dot, so that some of the parameters do not get properly estimated until the coast station sends more text. This further emphasizes the need to use additional information relating the behavior of each parameter to changes in the other.

The statistical structure of the Morse code was studied some time ago by Freimer and Gold [3]. It should be possible to apply the considerable body of work on order statistics to gain further insight into the behavior of the RWE algorithm, especially as to the factors which affect the classification efficiency and convergence rates. As these factors are in turn affected by the distribution of the various Morse intervals in the actual text, more work is indicated to more closely estimate the a-priori character probabilities and thus the various Morse-interval a-priori probabilities. The work in [14], which develops a Markovian model for the Morse process based on a Markovian model for English letters, is a step in this direction.

A number of observations can be made about the modifications which might be made in this system if adopted for general use. First, the sampling rate could be made quite a bit slower. If a narrowband predetection filter of bandwidth in the order of 85 Hz is used, the sampling rate after detection need by no higher than 200 Hz or so. Second, from the experience with the present system, it appears possible to delete the matched filter, and implement the threshold corrector and slicer in analog form prior to digitization. The analog-digital converter would not be required, and the binary signal from the slicer could be read by the computer as a single bit. Both of these factors would considerably decrease the CPU processing required and

should allow up to several Morse circuits to be multiplexed through a single LSI-11.

Conversely, by using careful software coding and eliminating unessential features, it should be possible to delete the TU and implement the matched filter as a pre-detection filter, saving a considerable system cost as well as allowing a substantial improvement in the signal-to-noise ratio. With an intermediate-frequency bandwidth of 250 Hz centered on 500 Hz, a sampling rate of 2000 Hz (four samples per cycle at the Nyquist rate) and envelope detection, an SNR improvement of 23.2 decibels is possible.

A number of improvements in this system could be made by refining the detection process. A 3 dB improvement in the SNR can be obtained by using synchronous demodulation. Guenther demonstrated that linear filtering of the Kalman-Bucy type [20] can improve the SNR about 2 dB when used as a postdetection filter and about 14 dB when used as a pre-detection filter. He also showed that smoothing following the filter can improve the SNR another 1 dB. Perhaps the greatest gains, however, would seem to be achieved by incorporating a richer structure to the Markov model used by the Viterbi decoder. Use of digram probabilities, when justified, could markedly reduce the common kinds of errors where Morse characters get concatenated or amputated due to mangled element/word spaces. Finally, postdecoding dictionary-directed correction algorithms based on the Viterbi algorithm [8] and the Levinshtein metric [21] may

well prove the most useful of all.

With all the overhead of the DCN software included, the present program requires about 6000 sixteen-bit words of storage, about half of which is dedicated for the algorithms described in this report, together with the Morse output routines, radioteletype routines and transmitter interface. A program designed specifically for the Morse decoding process with only the necessary features could easily be coded to run in the basic LSI-11 system with 4K of memory. However, due to the extensive and time-critical computations required, the firmward EIS/FIS option would probably still be necessary.

4. Acknowledgments

The author would like to thank his students, in particular R.J. Wetherell, for many stimulating discussions in which the framework of the design described here was evolved. Several students, sensing their instructor's weakness in such matters, spent many a happy hour of computer time playing with various aspects of formal-grammar, coding, detection and information theories, as applied to the Morse decoding problem. Besides being a lot of fun, these activities certainly expanded the horizons of the lot of us and profoundly impressed us with the remarkable ability of the human Morse copyist to easily outperform systems even as sophisticated as this under marginal circuit conditions and gross eccentricities in operator style.

5. References

1. Petit, R.C. The Morse-A-Verter. QST Vol. LV, 1 (January 1971), 30-39.

2. Gold, B. Machine recognition of hand-sent Morse code. IRE Trans. Inform. Theory 1 (March 1959), 17-24.

3. Freimer, M., Gold, B., and Tritter, A.L. The Morse distribution. IRE Trans. Inform. Theory 1 (March 1959), 25-31.

4. Guenther, J.A. Machine generation of hand-sent Morse code using the PDP-12 computer (M.Sc. Thesis). Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1973.

5. Mills, D.L. On teaching Morse to a computer (limited distribution technical memorandum). Department of Computer Science, University of Edinburgh, Scotland, June 1971.

6. Bell, E.L. Processing of the manual Morse signal using optimal linear filtering, smoothing and decoding (E.E. Thesis). Naval Postgraduate School, Monterey, California, September 1975.

7. Wozencraft, J.M., and Jacobs, I.M. Principles of Communication Engineering. Wiley and Sons, New York, 1965.

8. Forney, G.D., Jr. The Viterbi algorithm. Proc. IEEE 61, 3 (March 1973), 268-278.

9. Viterbi, A.J. Convolutional codes and their performance in communication systems. IEEE Trans. Commun. Technol. COM-19 (October 1971), 751-772.

10. The Radio Amateur's Handbook, Fifty-fourth Edition. American Radio Relay League, Newington, Connecticut, 1977.

11. Reference Data for Radio Engineers, Sixth Edition. Sams and Co., New York, 1975.

12. Radio Communication Handbook, Fourth Edition. Radio Society of Great Britain, London, 1971.

13. Reza, F.M. An introduction to Information Theory. McGraw-Hill, New York, 1961.

14. Howe, D.M. Decision-directed modified quasi-Bayes estimation and tracking of the *nonstationary* stochastic parameters associated with a communication signal. (Ph.D. Dissertation). Department of Electrical Engineering, Catholic University, Washington, D.C.

15. Tou, J.T., and Gonzalez, R.C. Pattern Recognition Principles. Addison-Wesley, 1974.

16. Turin, G. An introduction to digital matched filters. Proc. IEEE 64, 7 (July 1976), 1092.

17. Mills, D.L. The Basic Operating System for the Distributed Computer Network. Computer Science Technical Report TR-416, University of Maryland, January 1976.

18. Mills, D.L. An overview of the Distributed Computer Network. Proc. AFIPS 1976 NCC. New York. June 1976.

19. RT11 system reference manual. Order number DEC-11-ORUGA-C-DN2. Digital Equipment Corporation, Maynard, Massachusetts, 1976.

20. Van Trees, H.L. Detection, Estimation and Modulation Theory, Part I. Wiley and Sons, New York, 1968.

21. Okuda, T., Tanaka, E., and Kasai, T. A method for the correction of garbled words based on the Levenshtein metric. IEEE Trans. on Computers C-25, 2 (February 1976), 172-177.

6. Appendix A. Program Notes

The Morse decoder described in this report is implemented as three modules in the RT-11 MACRO language which are designed to be linked to other modules of the Basic Operating System (BOS) for the Distributed Computer Network (DCN). The three modules include:

1. A module (CWV) for the input filtering and detection process together with an extension to the BOS asynchronous device output process (TTX) for Morse and Baudot transmission. This module implements the digital signal processing algorithms.
2. A module (CWM) for an extension to the asynchronous device input process (TTR) for the Morse decoder. This module implements the Ratio-Weighted Estimation (RWE) and Viterbi algorithms.
3. A module (MPB) for the transition-probability tables used by the Viterbi algorithm. There are currently two tables in this module, one for the case of a-priori symbol probabilities inversely proportional to the number of marks in the Morse character and the other for the case of equal a-priori symbol probabilities.

The system will run on any PDP11-compatible processor equipped with the EIS option. Hardware requirements include the DRV-11 Parallel Interface Unit and Real-Time Clock (LSI-11 event line) connected to a 500 Hz source. The total memory size required for the three modules (exclusive of the

BOS supervisor and other peripheral support) is a little less than 2000 words (decimal) for re-entrant code and about 300 words (decimal) for the impure data. The 12-bit analog-digital converter (ADC) is connected to the input section of the DRV-11 in two's complement form, right-justified in the 16-bit word. The 10-bit digital-analog converter (DAC) is connected to the output section of the DRV-11, also in the two's complement right-justified mode. In addition one of the control bits of the DRV-11 status/control register is connected to the station keyer for Morse radiotelegraph transmissions and the other to the audio frequency-shift keyer (AFSK) for Baudot radioteletype transmissions.

The current software is run in a 20K-word LSI-11 system which includes an AED 6200 Dual Floppy Disk. The BOS configuration includes all of this standard network software for intercomputer communication and resource sharing, a resident, re-entrant BASIC interpreter and two independent user processes. Operation of all DCN functions, including BASIC program interface, command language interpretation and file operations are possible by a remote radio station using the Morse/Baudot input/output system. These mechanisms have been used to experiment in automatic station control and logging using small BASIC programs. It is in principle even possible to write and debug a BASIC program entirely in Morse code, an exercise which can be recommended only for one with infinite patience or one who is blind. Nevertheless,

the use of the Morse output facility, when connected to a code-practice oscillator, has proved exceptionally useful as an auxilliary operator "terminal."

The Morse/Baudot system operates as a device-service routine (DSR) in the BOS architecture and obeys the standard asynchronous device control and protocol procedures.

The name of the first Morse/Baudot device is CWO; subsequent devices are identified by a unique digit as the final character. The following restrictions apply:

1. The escape character ESC is not recognized on input. Therefore such parameters as mode and modifier bits cannot be controlled by input messages.
2. Only the ASCII and cleartext modes are operable.
3. No characters, other than those listed below, are recognized on input. Printing ASCII characters for which there is no Morse equivalent are transmitted as question marks. Printing ASCII characters for which there is no Baudot equivalent are transmitted as spaces. Nonprinting ASCII for which there is no Morse or Baudot equivalent are ignored on output.
4. Only those Morse character codes defined in Figure 1 may be transmitted or received. Figure 16 shows the correspondence between the service signals of the Morse code and their ASCII counterparts.

Morse Code		ASCII Code	Interpretation
SN	···-·	ACK	understand
AS	·-···	DC3	wait
AR	·-·-·	CR/ETB*	end message
CT	-·-·-	BEL	attention
KN	-·--·	DC1	go ahead
III	·····	CAN**	error
SK	···-·-	ETX	end work

*ASCII code CR is produced by Morse code AR. Morse code AR is produced by ASCII code ETB.

**ASCII code CAN is produced by a string of Morse dots longer than 5. A string of eight dots is produced by ASCII code CAN.

Figure 16. Morse/ASCII Code Correspondence

5. The Baudot character codes defined for the upper (FIG) and lower (LTR) case-shift correspond closely to the Teletype-A keyboard as used by most 60 and 66 wpm radio-teletype equipment available to the Amateur in this country. Figure 17 shows the correspondence between the Baudot and ASCII codes.

The following control commands are recognized:

RM <margin>

Set the right margin to the value <margin> (octal). An automatic carriage-return/line-feed sequence (CR/LF) will be transmitted at the first space following column number <margin> - 8. If no space occurs at or before

00	NUL	NUL
01	E	3
02	LF	LF
03	A	-
04	SP	SP
05	S	' (BEL)*
06	I	8
07	U	7
10	CR	CR
11	D	$
12	R	4
13	J	'
14	N	,
15	F	!
16	C	:
17	K	(
20	T	5
21	Z	"
22	L	)
23	W	2
24	H	# (pound sterling)
25	Y	6
26	P	0
27	Q	1
30	O	9
31	B	?
32	G	&
33	FIG	FIG**
34	M	.
35	X	/
36	V	;
37	LTR	LTR**

*Baudot code FIG-05 is produced by ASCII BEL; ASCII code ' (apostrophe) is produced by Baudot code FIG-05.

**ASCII code NUL is produced by these Baudot codes.

Figure 17. Baudot/ASCII Code Correspondence

column number <margin> then the CR/LF sequence will be transmitted at that point. In either case the remainder of the line is transmitted following the CR/LF. Note that this applies only in radioteletype output operations; the CR/LF sequence is ignored in radiotelegraph output operations. This operation is controlled by an escape sequence (see ESC command).

ESC <digit sequence>

Set or clear device modifier bits as described in [17]. The only digits which have meaning are:

0 enter clear-text mode

1 enter ASCII mode

4 set echo on

5 set echo off

6 set output right-margin edit on (see RM command)

7 set output right-margin edit off

RTG <baud><char>

Set the radiotelegraph mode for input and output operations at the element rate <baud> wpm and the character/word rate <char> wpm (decimal). If <char> is omitted, set the character/word rate equal to the element rate. Except for unusual cases, above 10 wpm the character/word rate is set equal to the element rate; while, at lower speeds, the element rate is fixed at 13 wpm and the speed of transmission set by the character/word rate according to the following table:

Speed (wpm)	Element Rate	Character/Word Rate
10	13	6
7	13	4
5	13	3

(These rates correspond to those used by W1AW in code-practice transmission).

Note that the <element> rate implicitly specifies a number of parameters in the Morse decoder, including the matched filter integration time and hysterisis level. In general, the system is tolerant to speed variations of plus-or-minus fifty percent or more of the nominal value.

RTTY <baud><char>

Set the radioteletype mode for input and output operations at the baud rate <baud> and character rate <char> (decimal). If <char> is omitted, set the character rate equal to the baud rate. Except for unusual cases, the character rate is set equal to the baud rate. When transmitting at a character rate less than the baud rate, the proper delay is added between the stop interval for each character and the start interval for the next succeeding character. Following are the common rates likely to occur on Amateur and commercial services:

Speed (wpm)	Baud Frequency (b/s)
60	45.45
66	50.0
75*	66.67
100*	75.0

*Receive-only

Note that output operations are possible only at the 60 and 66 wpm speeds; this is due to the limited number of frequencies possible to derive from the 500 Hz real-time clock.

FIDDLE <bits><weight>

Set extended mode bits as the value of <bits> (octal) and the a-priori probability weighting factor to <weight> (decimal). This command is used to turn on and off various processing features for test and evaluation purposes. The bits corresponding to the various features are documented in Figure 18.

STOP

Disable the processing for input messages. This command is used to stop the timer operations normally requested by the input processing module and provides a substantial increase in processing time available to other programs. The RTG or RTTY commands are used to restart input processing.

Bit Mask	Function
001	reverse signal sense
002	disable matched filter
004	disable automatic threshold detector
010	enable auto-downshift on receive (RTTY)
020	send shift on space (RTTY)
040	disable parameter estimation (RTG)
100	use alternate a-priori probability table (RTG)

Notes:

1. The matched filter cannot be disabled if the automatic threshold corrector is enabled.
2. Auto-downshift, when enabled, occurs on receive at a space (SP) or carriage return (CR).
3. Send shift, when enabled, causes a Baudot shift character of the proper type (LTR or FIG) to be sent preceeding the first character following one or more spaces (SP) or control characters (CR, LF, NUL).
4. The normal a-priori probability table is designed to optimize decoding for English text (probabilities inversely proportional to character length). The alternate table is designed to optimize decoding for code groups and cyphers (probabilities equal for all characters).

Figure 18. Fiddle Bits

In normal operation the equipment is turned on and the BOS loaded from floppy disk. The mode (radiotelegraph/radioteletype) and speeds are selected with a command of the form:

CTL 1 RTG 25

or equivalent (logical channel 1 is normally reserved for the radiotelegraph/radioteletype interface). With the station transmitter set for CW and automatic semi-break in, it is only necessary to start typing on the keyboard; the transmit/receive switching is automatic. The ESC-0 and ESC-1 sequences are used as necessary to switch between control (ASCII) and data (cleartext) modes. Typing can be well ahead of the transmitted Morse characters - up to the limits of the buffer space available.

When used in the radioteletype mode transmit/receive switching must be performed manually. Current Federal regulations require station identification in both radioteletype and radiotelegraph modes. Switching between these two modes for output (only) is controlled by the ASCII SO (Shift Out) and SI (Shift In) characters. The SO character switches to the radiotelegraph mode and the SI to the radioteletype mode. An automatic Morse station-identification feature is built in. When actuated by the ENQ (Enquiry) character the sequence of operations occur exactly as if the following characters had been typed at the keyboard:

CR LF SO DE W3HCF SI CR LF

where CR, LF, SO and SI represent the corresponding ASCII control characters and the remaining characters stand for themselves.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER TR-554	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Real-time recognition of manual Morse telegraphy using nonlinear estimation and Viterbi decoding.		5. TYPE OF REPORT & PERIOD COVERED Final 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) David L. Mills		8. CONTRACT OR GRANT NUMBER(s) N00014-67-A-0239-0032
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Computer Science University of Maryland College Park, Md. 20742		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Information Systems Program Code 437 Arlington, Va. 22217		12. REPORT DATE July 1977 13. NUMBER OF PAGES 85
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release. Distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Viterbi Decoding, Ratio-Weighted Estimation, Real-Time System, Matched Filtering, Morse Markov Model, Distributed Processing.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the analysis and design of a sophisticated real-time detector and decoder for manually-sent Morse code signals transmitted via high-frequency radio circuits. The detector/decoder system processes the digitally-sampled output of a conventional communications receiver in a LSI-11 microcomputer and prints the decoded characters on a computer terminal. The system software is designed to run if required in a distributed computer (continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified 7-14-77

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TR-554 (continued)

Abstract

network such as the DCN and can provide simultaneous multi-circuit decoding capabilities, using a multiplexed analog-digital converter and multiple terminals, in a single machine.

The system uses the principles of optimum filtering and receiver design, including matched filtering and Viterbi decoding. The statistical structure of the Morse process is encapsulated in a Markov model which includes a-priori probabilities and transition probabilities reflecting the entire alphabetical structure of the code. A key component in the design is a new algorithm, called ratio-weighted estimation (RWE) which provides extremely fast adaptation to the time-varying Morse code parameters typical of multi-operator circuits.