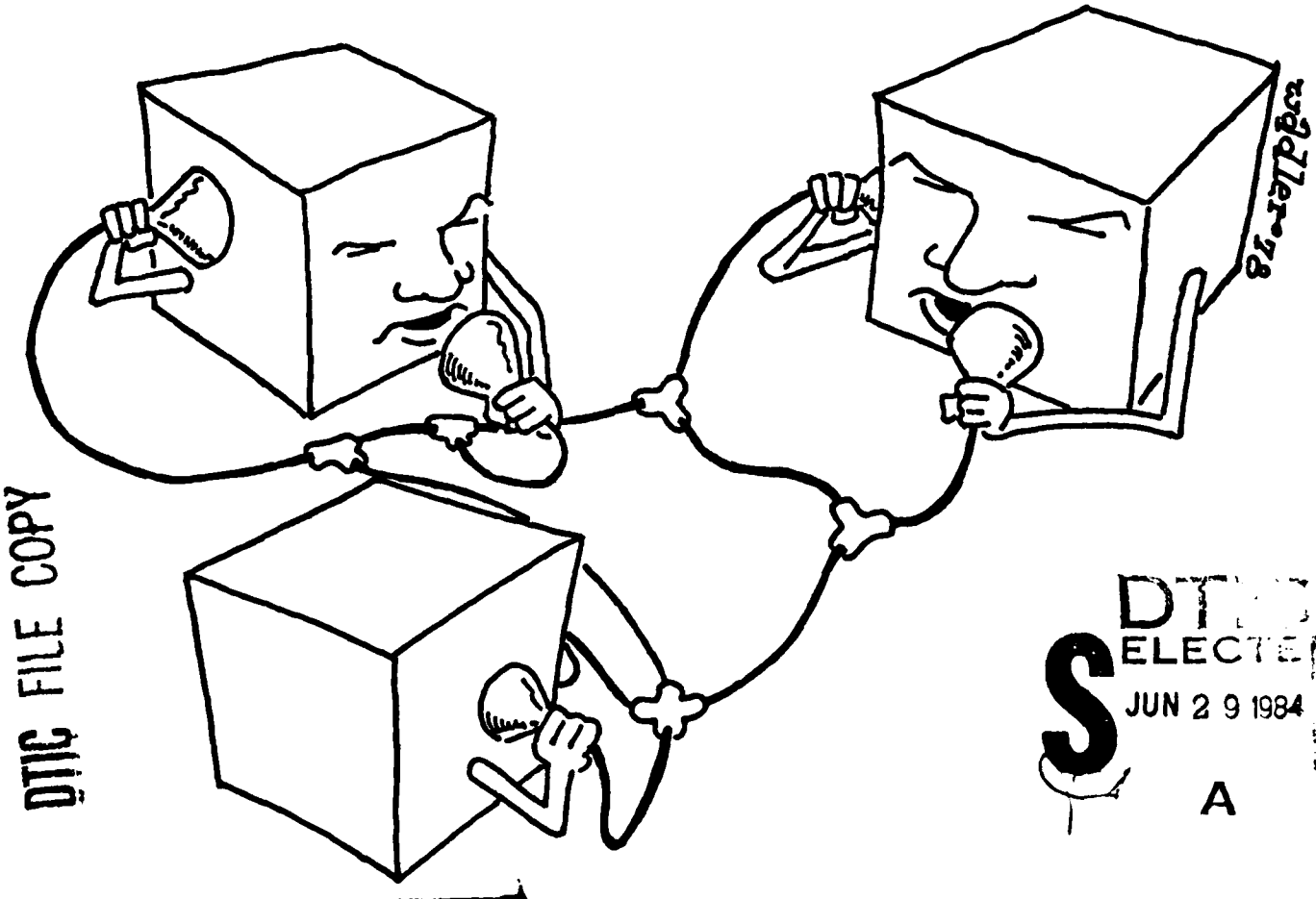
DISTRIBUTED SENSOR NETS

Sponsored By:
Information Processing Techniques Office
Defense Advanced Research Projects Agency

Hosted by:
Carnegie-Mellon University
Pittsburgh, Pennsylvania
December, 1978



MACHINE RECOGNITION AND UNDERSTANDING OF MANUAL MORSE

Albert Vezza, P. David Lebling,
Edward H. Black*, Timothy A. Andersen*, John F. Haverty*,
David Sherry*, and Gail E. Kaiser

Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

A. INTRODUCTION

"Morse code (as sent by hand) is one of the simplest of aural languages, yet it has features common to all spoken languages." That was written nearly 20 years ago by M. Freimer et al [1]. Until recently machine recognition of manual Morse code (its translation to printed text) eluded acceptable solutions, albeit not from lack of attention. A project in the mid-fifties at Massachusetts Institute of Technology's Lincoln Laboratories resulted in MAUDE (for Morse Automatic Decoder), one of the earliest machine translators of manual Morse code [2,3,4,5]. MAUDE and other early machine translators [6,7,8,9] were based on a small set of simple rules, both statistical and linguistic. The rules did not in any sense comprise a model of the language -- for instance English -- to be translated. These early attempts were restricted solely to translation. No attempt was made to incorporate into the systems an understanding of the information being sent, nor were aspects of the radio domain taken into account. A main premise of MAUDE and other early Morse code systems -- explicitly stated or implicitly assumed -- was that Morse code could be decomposed into a sequence of symbols comprised of the alphabet, the numerals 0 through 9 and punctuation marks. Since such a set is complete any sequence of Morse code could in principle be expressed by a sequence of these symbols.

Recently, a project at Massachusetts Institute of Technology's Laboratory for Computer Science has developed a new perspective on the manual Morse code problem. We have taken quite seriously the character of manual Morse code as expressed by the first sentence of this paper. Indeed our approach has been to include in a manual Morse code system extensive models or knowledge of the Morse, radio and natural language domains. Further, understanding of parts of a Morse code conversation is attempted, and it appears that it is necessary to understand some parts of what is transmitted in order to translate it correctly.

B. THE MANUAL MORSE CODE PROBLEM

The problem domain is that of hand-sent Morse code in an amateur radio network environment. To provide a global perspective of the Morse problem, a scenario of a typical amateur Morse-code network may prove helpful. A Morse-code network can be thought of as being composed of a hierarchical collection of sub-networks. A message in one sub-network may be destined ultimately for another sub-network in the hierarchy. A sub-network is typically composed of a group of operators and a controller (who is also an operator). The network controller's function is that of a traffic manager and general overseer to insure proper message flow within his sub-network. The network (we will henceforth drop the prefix "sub") becomes operational when the controller calls it to order. Each operator, on joining the network, communicates to the controller the number of messages the operator has to transmit, and to whom. The controller matches up operators and assigns them to a nearby (usually within 10 or 15 kilohertz) frequency to conduct their business. After the two members complete their communication, each reports back to the controller, indicating what traffic was transmitted as a check of completeness. At any time after being dispatched, a pair of operators may select a new frequency in a clearer area of the radio spectrum.

All negotiation between operators and controller is accomplished with the aid of a shorthand network protocol. Often the protocol is not followed precisely [10]. The protocol language is called "chatter," and it is composed of words whose generic type is "Q-sign," "Pro-

*Mr. Edward Black is currently employed by N.E. Berg Company in Bedford, New Hampshire; Mr. Timothy Anderson by Computer Corporation of America in Cambridge, Massachusetts; Mr. John Haverty by Bolt Beranek and Newman in Cambridge, Massachusetts; and Mr. David Sherry by the Xerox Palo Alto Research Center at Xerox Parc, Palo Alto, California

sign" or "Call-sign." The vocabulary of chatter includes about 100 essential words and as many as 900 others. The chatter language allows an operator to make statements, ask questions, and give orders. It provides a means for statement and query of operator identification, signal characteristics, rendezvous information, message traffic information, and so forth. Chatter contains an error recovery procedure for obtaining retransmission of either a word, the first character of each word in the message, everything after or before a particular word or phrase in the message, etc. The traffic that is communicated between Morse-code operators consists of messages, each comprised of a header, body, and signature. Header information typically contains from whom, to whom, handling instructions, precedence, number of words in the message, and other items.

1. The Radio Domain

There are a number of attributes of Morse code signals which we call radio domain attributes. Some of these attributes help but many hinder an operator's ability to translate Morse code. Unfortunately, even those attributes which helped operators perform translation heretofore were a bane to machine translators. Transmitter characteristics such as chirp are used to good advantage by an operator to separate a wanted signal from interference. Not only did machine translators not take advantage of such transmitter characteristics but such characteristics hindered the demodulation process to the extent that errors were introduced into the translation because of the characteristics. In addition to errors introduced by the demodulator's inability to handle transmitter idiosyncrasies the atmosphere adds its own characteristics which interfere with the Morse code signal; for example shot noise, fading, multipath, etc. make translation difficult for operators and machines alike.

(The current Morse code project did consider the problem of transmitter characteristics but unfortunately, because of some contractual restrictions, was prevented from dealing with atmospheric characteristics.)

2. Sender Induced Irregularities

Morse code is composed of a sequence of alternating marks and spaces. In on-off keyed Morse code, a mark is characterized by the presence of an audible tone for some interval of time and a space by its absence for some interval of time. There are two types of marks: dots and dashes. Ideally, the duration of a dash is three times that of a dot. There are three types of spaces. They are given various names. Here we shall call them mark-space, letter-space and word-space. Mark-spaces are used to separate marks of a single character, and their ideal duration is that of a dot. Letter-spaces are used to separate adjacent letters, and their ideal duration is three times that of a dot. Word-spaces are used to separate adjacent words, and their ideal duration is seven times that of a dot.

The precision and correctness of manual Morse code is far from ideal, even if problems associated with the radio domain are omitted. Senders themselves induce three types of irregularities: (i) spacing-errors, (ii) mark-errors and (iii) spelling-errors. More than ninety percent of all sender induced irregularities in hand-sent Morse code are spacing-errors. Such irregularities occur when a sender does not keep the proper ratios between mark-spaces, letter-spaces, and word-spaces. The result is analogous to spoken language that is slurred or broken by arbitrary pauses. The segmentation problem in manual Morse code is analogous to the segmentation problem in continuous speech. Irregular spacing makes it a difficult problem. Hand-sent Morse code of plain English text often has a spacing error in each word of the message and some words (typically the long ones) may contain several spacing errors. The next most frequent sender induced irregularity is a mark-error. A mark-error occurs when a sender omits, adds or changes the sense of one or more of the marks (dots and dashes) making up a word. Our experience indicates that as many as twelve percent of the words in a Morse code message may contain mark-errors. Spelling-errors which can be so classified occur most infrequently. Certainly less than one percent of the errors are classified as belonging to this type, mainly because senders know the chatter language very well and because many common English spelling errors map directly onto a spacing-error or a simple mark-error, making spelling-errors indistinguishable from these other errors.

Needless to say, sender induced irregularities in manual Morse code proliferate to the extent that early machine algorithms such as MAUDE often produced translations that could be read only with great difficulty, especially if the reader had no knowledge of Morse code. Yet operators have little or no difficulty coping with the irregularities resulting from operator lapses, radio noise and interference and can translate a manual Morse code signal on which moving-threshold translators such as MAUDE would produce hopelessly garbled results.

C. DOMAIN MODELS

It is clear that good Morse operators have conceptual models of the Morse code environment that they use to help them perform their task. They have models of Morse "sounds" -- sequences of dots and dashes with rhythm and timing information -- and map these sounds into the letters and words. They have models of the language constructs that are used, be they English, another natural language, or the chatter language. Operators form models of other operators' idiosyncratic

mannerisms and use these models in the translation and understanding processes and in identifying other operators. Operators also have models of the Morse code and radio domains. It is common knowledge that "TH" is often sent with a short space between the letters, so that a machine often interprets it as "6" (thus "6E" is really "THE"). Similarly, "AN" is often interpreted as "P" (thus "PO" is really "AND"). In the radio domain, knowledge about where relevant operators are in the frequency spectrum, what a particular transmitter sounds like, how a signal fades and returns -- all these form the models that help operators identify, track, transcribe and understand one another. It is the human beings ability to interpret Morse sounds in the context of such mental models that allows her or him to perform so well.

At this point a slight digression is in order. Listening to a Morse code conversation among a group of operators, one notices three distinct aspects of the conversation. These correspond to (i) network chatter, (ii) message headers, and (iii) message bodies. The chatter section is often very poorly sent. Characteristically, many letters and words are slurred or separated and corrupted by other operator lapses. Yet receiving operators have little difficulty understanding chatter, because they have a model of the global situation: what question was asked by whom, who is currently waiting on the network, who has message traffic for whom, and so forth. This model and the ability to understand the conversation is vitally important to translation.

Headers of messages are structured but, unfortunately, not rigidly. Again, to translate them correctly one must have some understanding of what headers are about. For instance, dates may be sent as "8 Dec 78" or "8 12 78" or "81278" and times may be sent as "1000Z" or "1000". We have written the dates and times in an ideal manner, but, in fact, they might be -- as a result of operator lapses -- segmented quite differently, so that parts of numbers are run together or a number is split apart, and one or more numbers might contain a mark-error. One other aspect of numbers is very important. All numbers in Morse code are five marks long -- see [11] under Morse code -- yet they are often abbreviated and sent as what are called cut-numbers. Again, context is often required to perform translation correctly.

The body of amateur radio message traffic is typically English with some abbreviations. To attempt to understand all of the English language would be far beyond the scope of this research. We have built into the system just enough knowledge to let it perform in a creditable fashion. Our experience indicates that a vocabulary, some rules about where numerals can occur in text, rules about how to handle error signs, and a measure of closeness in a Hamming-like space (for correcting operator induced irregularities) are absolutely essential to the correct translation of plain text. Given

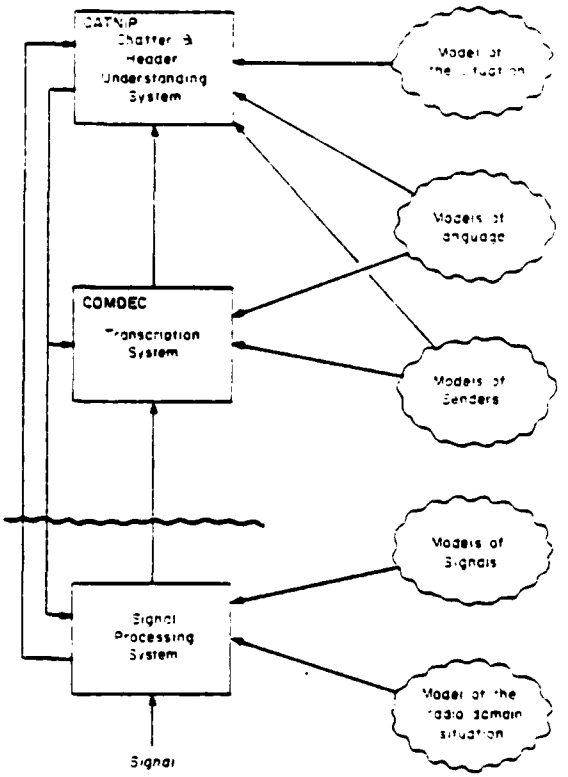


Figure 1. The Three Major Modules of the Morse Code System and the Domain Models They Use.

those sine qua nons, our experience suggests that knowledge about idiosyncratic and irregular behavior of individual operators facilitates translation.

Military message traffic differs slightly because the body may consist either of plain text or of cipher groups. A message sent in cipher has no language context, but knowledge about the number of groups in a message, the number of characters in a group, and whether groups are alphabetic, numeric, or mixed is necessary to translate such a message.

Figure 1 shows a block diagram of the three major modules of the Morse code system under development by the authors. Also shown are the necessary domain models required by each module in order for it to perform its task properly. The wavy line in the diagram indicates that the signal processing system, which is composed of special hardware and a PDP-11 computer, is not integrated with the other major modules which are COMDEC, the transcription (or translation) module, and CATNIP, the chatter and header understanding module. The last two are software modules written in MDL (a LISP-like language) [12] and running under TOPS-20 [13] and ITS [14]. Experiments are conducted independently

for the signal processing system, and human intervention is required to transfer the results to the other two modules. COMDEC and CATNIP are well integrated, with appropriate feedback, and externally they appear to behave as one system.

A few phrases about each domain model may prove helpful:

1. Model of the radio domain situation -- how the individual transmitters of interest sound, i.e., whether a transmitter has any characteristic envelope or carrier distortions, and if so what kind and a measure of the amounts.

2. Model of the Network situation -- which operators are logged into the network, which are off control frequency, which are on control frequency and where each operator's transmitter is tuned relative to those of the other operators on his frequency. This last bit of information turns out to be quite important, as we will show later, even though all the operators are working in a thirty to fifty Hertz band.

3. Models of senders -- the irregularities a particular sender may introduce, such as his or her idiosyncrasies of language, a proclivity to introduce extraneous dots or omit dots, etc.

4. Models of language -- the full gamut of possibilities are required for parsing and understanding chatter, but we have rather simple models for handling message bodies such as a vocabulary and some simple rules for handling some special constructs and numbers.

5. Models of the situation -- the system must know when a question is asked and the possible range of expected answers; it must know that a frequency change has been ordered or negotiated and how to respond appropriately; and so forth.

D. THE MORSE CODE SYSTEM

As mentioned, the Morse code system is composed of three major modules and some domain models. We will attempt in this section to present a short explanation of each of the major modules.

1. Signal Processing System

We believe that an interesting and important development in the signal processing area of the Morse code project was the implementation of a novel tandem phase-lock-loop filter that utilizes time reversal of the input signal. Despite the use of time reversal, the output can be obtained in real time, albeit with a constant delay. The nature of Morse-code signals -- the fact that they are on-off keying or frequency-shift keying -- and the fact that initial experiments indicated that the transient response of the phase-lock-loop filter interfered with the measurement of important signal parameters, led to the development of the novel filter.

There is a great deal of information contained in the audio sound of a Morse-code signal -- the signal characteristics per se -- besides the timing information of the marks and spaces. It became clear while running some experiments in understanding Morse-code network conversations that the signal characteristics contained information that was an important part of the context of the situation; it is necessary to extract this information in order to understand the network conversations (q.v.).

A small digression is required to explain what a human operator hears in the sound quality of the signal in order to understand what must be extracted from Morse code signals. Briefly, a human operator is capable of detecting and tracking certain signals in a crowded spectrum of similar competing signals. Note how one can follow a particular conversation at a crowded cocktail party. One can do fairly well even with one ear. There is no binaural effect in the Morse-code domain. This discriminating ability of human beings is evidently knowledge-based. To discriminate signals, an operator uses information about how the signal sounds: (a) its frequency; (b) its anticipated frequency drift; (c) its amplitude and rate of chirp, if any; and (d) the amount of envelope distortion such as hum, clicks, yoop and whatever other characteristics of the waveform can be characterized. A good signal-processing front end should be capable of measuring some, if not all, of the above signal characteristics and of using the measured characteristics for signal discrimination.

a. Tandem Phase-Lock-Loop

The general requirements can be translated into specific requirements of a receiving filter process for the Morse-code application. (The specific filter design is for an on-off keyed signal, and experiments were conducted only with such a signal. Therefore, the discussion that follows is in the context of on-off keyed signals. However, it should be pointed out that similar arguments can be made, and similar results can surely be obtained, for the case of frequency-shift keyed signals.) Extracting the on-off timing information for marks and spaces as well as signal quality information requires determination of the transitions of the signal as well as continuous estimation of the amplitude and frequency of the signal. The latter information serves a dual purpose. First, it is used to characterize transmitter signals for use in transmitter recognition. In addition, the frequency on which a station is transmitting is part of the situation model, and an uncharacteristic frequency shift of ten or several tens of hertz often indicates a change of

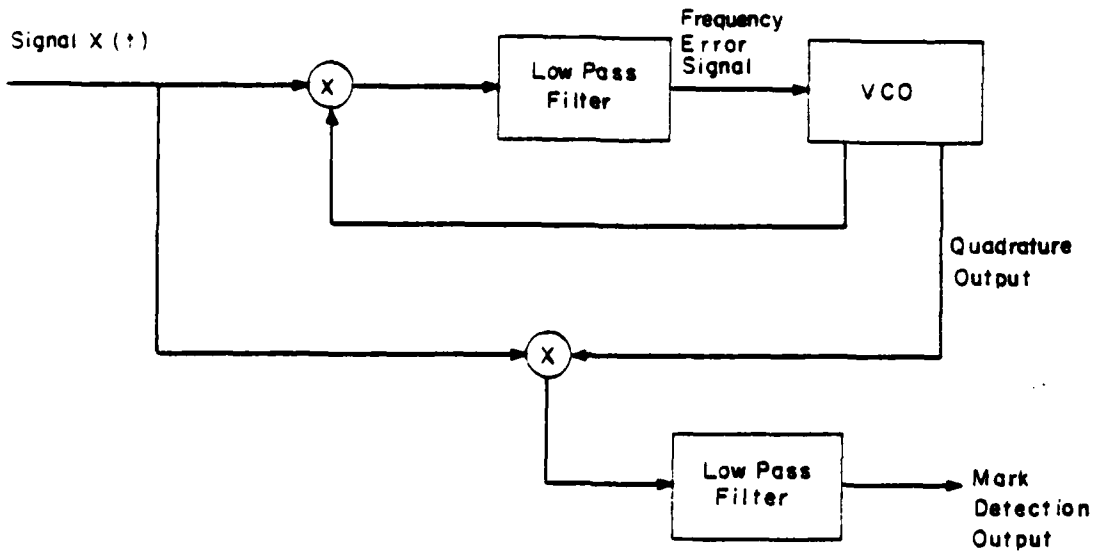


Figure 2. Phase-Lock-Loop Detector.

"sender" during network conversation. This type of cue is extremely useful, because, after contact has been well established, operators often do not re-identify themselves. Furthermore, without the change of sender context information, some chatter constructs are ambiguous.

The tracking-filter model of the phase-locked loop (PLL), Figure 2 [15, 16], is well suited to extracting the information indicated above from a signal in that it gives continuous frequency and amplitude estimates, and presents a relatively narrow-band filter to a frequency-modulated carrier.

However, before a PLL can give accurate demodulation, it must achieve lock. (Lock is the state of the PLL when the voltage controlled oscillator (VCO) tracks the incoming signal with a constant phase lag.) The time to achieve lock is inversely proportional to the natural frequency of the loop, and is affected by such factors as the initial frequency error (difference in frequency between the VCO and the input) and noise in the loop.

Chirp is frequency modulation which frequently occurs in low quality transmitters. It is often caused by inadequate filtering of the power supply, which causes the oscillator to change frequency when the power stage is turned on. Thus, the frequency modulation, or chirp, exists where the signal makes a transition from off to on, or vice versa. Most often, it exists only at the beginning of the "on" period or "mark" of the Morse-code signal. Unfortunately, in a traditional PLL arrangement, or for that matter any type of traditional filtering, the transient response of the filter is superimposed on the signal and is largest at the signal transition points. The problem is exacerbated when interference is considered; as one narrows the bandwidth of the filter to eliminate the interfering signals, the period for which the filter transient response is a significant factor in the output is lengthened. In the case of the PLL, the transient between acquisition and lock at the beginning of the signal is the major one, because a PLL will track the signal during the on-to-off transition until it reaches a signal-to-noise ratio at which the signal is lost. Thus, because the frequency and amplitude estimate of the signal prior to lock contains important information, i.e., the chirp information and the time at which the mark began, it is desirable to reconstruct that portion of the signal.

A number of ways of recovering the pre-lock information can be conceived. The method settled upon is simple. It involves sampling and storing the input to the PLL, and then, after the PLL has completed processing the mark in the forward direction, sending the stored samples in reverse order through the loop. The loop then demodulates a time-reversed replica of the original signal, and the original leading-edge information is reliably obtained from the trailing edge of the reversed signal.

Because it was desirable to run the process in real time, only the beginning portion of the signal is reversed and a second PLL is used to demodulate it so

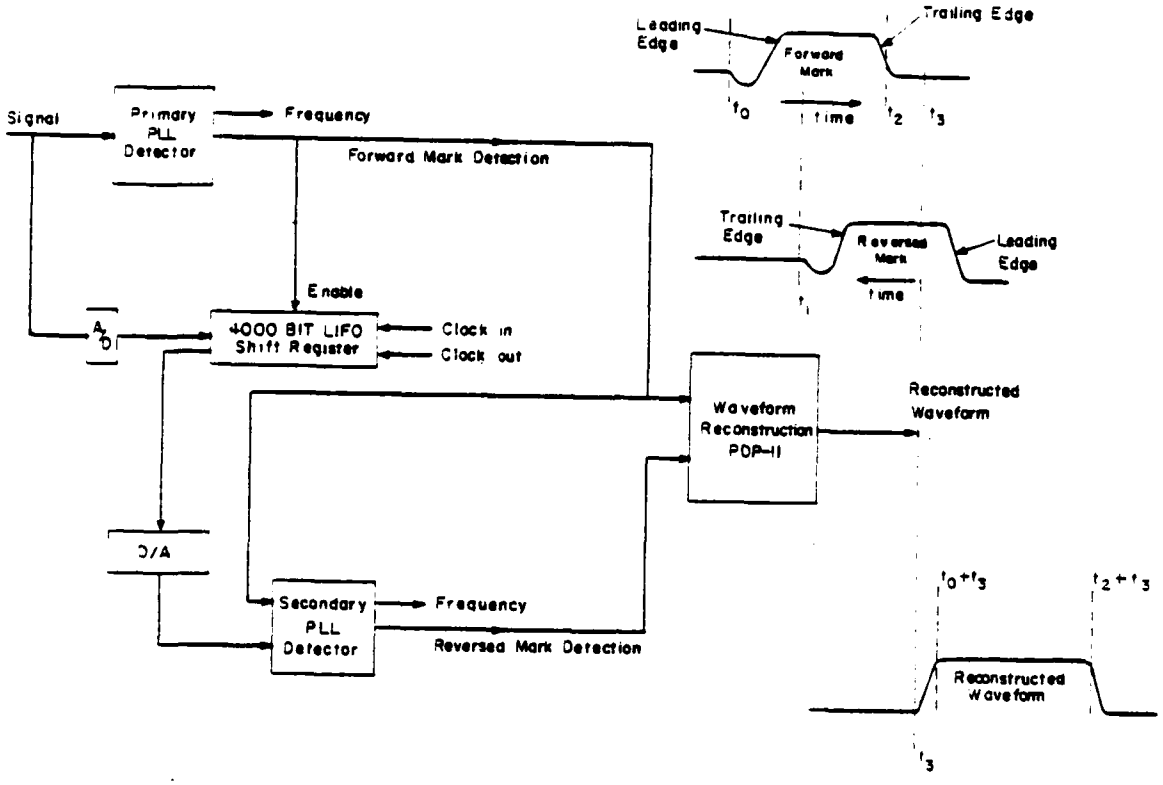


Figure 3. Tandem Phase-Lock-Loop Detection System with Wave Form Reconstruction.

that the first PLL can continue to demodulate the forward signal. In addition, the reversed signal can be compressed in time by sending the reverse-order samples through the secondary PLL at a rate faster than they were collected. Of course, the secondary PLL needs to run at a higher frequency. Thus, it is possible to have reconstructed the mark before the next mark begins.

The tandem phase-lock-loop system is shown in Figure 3.

The input is sampled and stored in a last-in-first-out (LIFO) memory. Meanwhile the input to the secondary PLL is taken from the quadrature phase of the primary PLL. When lock is indicated (by the quadrature, or correlation output of the primary PLL) the input of the secondary PLL is switched to the digital-to-analog converter (DAC), and the stored samples are read out in reverse order. The outputs of the secondary PLL are taken as the demodulated signal for the time prior to lock. The outputs of both PLL's are sampled and a PDP-11 program reconstructs the waveform (mark).

2. COMDEC: The Morse code Translation System

The COMDEC (computerized Morse Decoder) control structure was designed to consist of any number of modules, each of which would be an "expert" on one aspect of translation. Each module can add suggested translations to a lattice of possible translations. Successive modules decide whether further error correction is necessary by examining the quality of existing suggested translations in the lattice, then pass that section of code to the next module in the chain, and so on, until each translator module has examined the entire message. A major part of the design is an N-dimensional metric for measuring trial translations and a heuristic algorithm for comparing the measures.

Translator modules are ordered approximately in terms of increasing severity of the sending errors they are able to correct. Thus, spacing-errors are corrected first. Later, mark-errors are corrected.

The first module to process a code sample is a moving-average translator, a hybrid of MAUDE [2] and FRAUD [7]. It classifies marks and spaces into the

appropriate types by comparison with continuously updated averages for each type and thresholds between types. More importantly, it associates with each mark and space a number which represents its confidence that the classification was correct. These assignments and confidences are used by later modules to select areas of the sample where errors appear to have occurred.

a. Handling the Segmentation Problem

The performance of moving threshold Morse code translation systems suffered quite severely from the effects of improper segmentation of Morse code sequences. Morse code translators suffer from two types of segmentation difficulties: first, character-spaces and mark-spaces are confused such that two characters are sometimes decoded as one, e.g., 'AN' as 'P', and vice versa; second, word-spaces and character-spaces are confused such that 'THEY ARE' is translated as 'THEYARE' and 'HOLD' is translated as 'H OL D'. The DEGARBLER [17], a program based on the use of digraphs and trigraphs found in English words and a dictionary of words, was capable of correcting many of the word-segmentation errors that occurred in MAUDE output, but it could not obtain the proper transcription of Morse sequences that had character-segmentation errors present.

Because words corrupted only by segmentation errors contain a correct sequence of marks, we realized that words can be represented by the "run-length sequence" (RLS) of the marks. The run-length sequence is a string of numbers, composed of the number of consecutive dots which begin the word, followed by the number of consecutive dashes which follow those dots, and so on. (By convention, a run-length sequence always begins with the number of dots, even if the number is 0.) The principle is simple. All mark and character spaces are removed from the representation. Because spacing errors occur frequently -- one or more per word -- a major source of difficulty is removed. Basically, the RLS is a mapping of words in the Morse-code domain onto some space. Each word along with its variants, obtained by taking all possible combinations of character segmentation errors, are mapped to the same point in that space.

The reverse mapping of run-length sequence onto legal words is not necessarily unique. However, better than 85% of the RLS for a dictionary of over 10,000 words are unique and most of those that are ambiguous have but a few members in their ambiguity sets. The ambiguity resulting from representing words by their RLS is not a significant problem, because often the simple heuristic of comparing all the words with a specific RLS to the MAUDE transcription of that segment of Morse code is sufficient to determine the one the sender intended.

To understand how the RLS is used, consider the following. The sequence of dots and dashes for the word "the" was sent with spaces such that a context-insensitive decoder could not transcribe it into a proper word, i.e., the sequence sounds like '6E'. A general heuristic that one might think of adopting is this: find all the possible words which that sequence of marks could represent and pick the one that seems to fit best according to some criterion. If one tried to implement such a heuristic in a brute-force manner, 2**5 or 32 possibilities would need to be considered, as the word "the" (- .... .) has three mark spaces and two character spaces for a total of five spaces. Thus 32 permutations and look-ups would be required to find "the" and any other words with that mark sequence. On the other hand, using the RLS representation with no permutations and one look-up, "the" and all other legal words with that RLS are returned. Consider the word "complex", not an unusual or overly long English word. If it were sent with a spacing error, and a brute-force method were used to obtain possible candidate words, the possible spaces associated with the sequence of dots and dashes would require more than two million (2**21) permutations and look-ups, only to discover that in the dictionary currently in use only the word "complex" could be represented by such a sequence of marks. Again, the RLS representation would return the word "complex" in one look-up.

The use of the RLS solves only the character segmentation problem. Word segmentation is handled by a heuristic that splits or joins character sequences. The heuristic uses the confidence levels returned by the MAUDE decoder to determine how to proceed. (For more details on RLS translators, see [18, 19, 20]).

b. Handling Mark Errors

The mark-error correction modules are able to correct eight different classes of mark-error in words which contain at most one mark error. Statistically, the vast majority of mark errors are of the types COMDEC can correct, and occur only one to a word. The types of mark errors corrected by COMDEC are (i) sending an extra dot, (ii) sending an extra dash, (iii) sending two extra dots, (iv) running two dots together as a dash, (v) splitting a dash into two dots, (vi) dropping a dot, (vii) dropping a dash, and (vii) dropping two dots.

These mark-errors are easily correctable because words with these classes of mark-errors map to nearby points in the RLS space and the space is very sparsely populated for containing eleven marks or more.

c. Handling Error Signs and Number Constructs

When a sender recognizes that he or she has made an error, he or she will resend the erroneous word, phrase, or sentence, signalling with an error sign that

retransmission is about to occur. This behavior is somewhat analogous to a typist who spaces back over an error and overstrikes it with X's. An error sign is usually a sequence of dots sent rapidly, rarely fewer than six, and rarely more than twenty. The number of dots sent varies even within a single transmission, as does the separation of the dot-sequence from the erroneous code preceding it and the "correct" code following it. More importantly, the semantics of an error sign vary even more widely. Thus it is necessary to locate and to ascertain the meaning of an error sign. An error sign may mean to ignore the previous word or characters, or it may mean that the previous word or phrase will be resent, and so on. Some examples from actual code (with the symbol "@" used to represent an error sign) follow:

ANY B OY OR GIRL 13 TO 18 @ 19 WHO

The correct translation:

ANY BOY OR GIRL 13 TO 19 WHO

This is the most typical use of an error sign. It signals that the previous word or object was in error, and the sender resends the word correctly. The error sign in this example contained thirteen dots.

PAGE B 23 OF TODAYS PE TPERS @ TODAYS PAPER

This is similar to the previous example, but two words are erased and resent. The error sign contained eleven dots.

THE CORNER OF WASHINGTON
BLVD @ AND SCHOOL STREETS

In this example, the word "BLVD" is erased — it should not have been sent at all. The error sign contained seventeen dots.

d. Handling Numbers in Plain Text

COMDEC recognizes arbitrarily long sequences of digits as "numbers".

The problem of transcribing numbers is analogous in some ways to that of transcribing error signs, and it arises from the fact that most of COMDEC's transcribing is vocabulary-based. Since it is theoretically possible to send a number containing an arbitrary number of digits, it is impractical to use a "dictionary" of numbers. Instead, COMDEC takes advantage of the properties of the Morse code used to represent the digits: (i) All digits consist of five marks (except cut numbers which are also handled). (ii) Every digits contains a sequence of dots followed by a sequence of dashes, or vice versa. (iii) A number very often appears in context, for example, as a part of a date, time, address, page number, or age specification. This context is used to reinforce the probability that the item is a number. A number appearing out of context must be allowed, as all possible contexts have not been or cannot practically be implemented. If a number appears out of the contexts in which a number is expected, it is looked upon by COMDEC with suspicion, and it will be allowed to remain a number only if it is relatively well sent.

COMDEC searches for mark sequences that fit these criteria and then attempts to "expand" them on either side (to produce complete numbers). The only limitation on this algorithm is that at least one digit of an n-digit number must be sent correctly.

e. The COMDEC Dictionaries

COMDEC's dictionary for plain text messages is comprised of 4200 root words. A morphology program embedded in the translator handles a large but not complete set of word endings. Thus, a word usually appears in the dictionary only in its root form. The current morphology program understands the endings "-s," "-ed," "-ing," "-er," "-est," "-ly," "-tion," "-ment," and combinations of the preceding, such as "-ers." The dictionary specifies the endings taken by each word. When endings are considered, the effective size (about 18,000 words) of the dictionary is several times the number of its roots and it approaches the size of the conversational vocabulary of the average English speaker.

3. CATNIP Chatter and Header Understanding System

CATNIP is a semantic-syntactic augmented-transition-network (ATN) parser that chooses a path through the lattice of possible translations created by COMDEC.

CATNIP uses ATN diagrams to choose the correct word from a lattice of possible translations. It starts in a certain state of the transition network, and progresses from one state to another, depending on the next word or words in the lattice. With each state is associated a list of words, and with each word a new state. CATNIP matches the list of words from the state with the list of words possible at that point in the translation lattice, matches yield valid new states.

If that were all, the network would simply be an unaugmented transition network. However, CATNIP retains a context, which it changes (usually with every word) and which can be tested when it is trying to match the words. The context includes such things as who is the sender of the current transmission, who is the receiver, who is the net controller, and so on. ATN's, as opposed to unaugmented transition networks, are good for parsing grammars that are dependent on the context and on past occurrences [21].

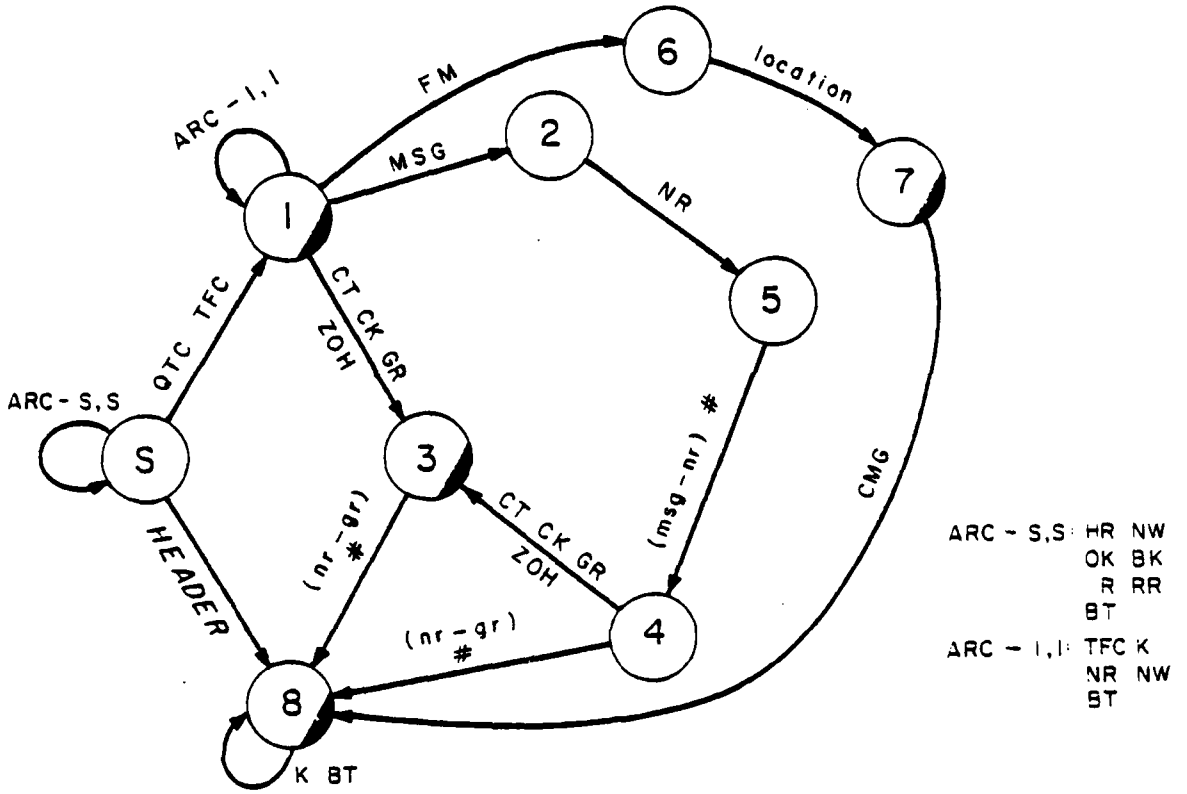


Figure 4. Augmented Transition Network Diagram Called Traffic Header.

Naturally, ambiguities creep in. Sometimes more than one match is possible; CATNIP allows for this by processing one of the new valid states and saving all the others. The context at that point is saved with the states that were saved. CATNIP has the ability to return to the saved states and try those alternate paths.

Finally, CATNIP also has a limited understanding of the events on the net. Understanding these events is important in understanding the state of the net at any point (how many operators are working, who they are, who is talking, etc.) and it is important in choosing the correct word at a particular point in the translation.

The context is used as the "understanding" part of CATNIP. Take the following transmission as an example: "ROCK ROCK ROCK DE SALT SALT QSA ? QRK ? K". Upon completion of the parse, the parser would retain a context that contained the information that the receiver was ROCK, the sender was SALT, and SALT had asked ROCK two questions: "What is my signal strength?" and "What is my intelligibility?"

Retaining this kind of context helps find the right translation and decide later ambiguities (such as who is the receiver at a certain point, if he or she was not explicitly named). The successive contexts also furnish a synopsis of the entire session after the parser is finished.

CATNIP is a recursive procedure that allows one to name ATN diagrams of simple structures (such as Q-signs that are often used), and to use those as parts of other diagrams without actually duplicating the simple diagrams. Thus a more structured "grammar" can be created without over complicating the data base.

Figure 4 shows a typical ATN diagram. Each italicized ARC label such as "Header" indicates a call to another diagram; in lower case are labels such as "location", which indicate that the labeled input that will parse is a location (such as "BOSTON" or "BOS"); arcs labeled with a number sign "#" mean that a number is acceptable as input (with the parenthesized statement indicating the meaning of the number, e.g., (nr - gr) means number of groups); and arcs labeled in upper case mean that literal input of one of the specified labels is acceptable. The system currently contains about 25 diagrams with an average complexity of the one shown in Figure 4.

E. RESULTS

Two sources of test data were used. We obtained approximately 30 thousand characters worth of data from amateur operators and others who could send and receive Morse code. This data was hand-keyed in our laboratory. Some of this data was obtained using actual transmitters connected to our cable network [22]. This data was composed primarily of plain text with a small admixture of network chatter. Chief Warrant Officer Robert Bolling at the Army's Morse code school at Fort Devens arranged to have several instructors at that school simulate an actual network environment. From that simulated environment we received audio tapes containing approximately 20 thousand characters of hand-sent Morse code. This data was composed of chatter, headers, plain text messages and cipher group messages. It was the most useful data in that it forced us to face many problems and constructs we had not encountered previously: error-signs, numbers in plain text and many different types of chatter constructs.

We measure the performance of our Morse code system in two ways. First we compare the final results with the results produced by the first stage of our translator, which is a MAUDE-like moving threshold translator. Second we compare the final results with an experienced operator's translation. The experienced operator had the use of a variable speed tape recorder. He at times found it necessary to replay parts of the tape, occasionally at slower speeds.

Figure 5a shows a translation by a MAUDE-like translator of one of the plain text messages contained in the Fort Devens data. While it can be read, especially by a person who knows Morse code, a considerable effort is required. Figure 6a shows the same message as translated by COMDEC, our Morse code system's

```
IN ANATT E MPT TOALLEVE EE{.........}ALLE4IATE YOUTV UN E MPLOYMENT I N THE CI
TY MIM T {.................} CI TY. TSE SUNDAYGLOBE ON MAY30 WILL PUBLISS FRE E
AOVERTIS E MENTS FOR 6OSTON T E E NAGERS S E E KING SUMMER JOBS. ANY B OYORGIRL
13TO 10 {............}19 WSOLIVES IN BOSTONCANP LACE AJO6 WANTED AO WI THOUT
CAAEEEEEEEEEEEEEEE CH#GE 6YFILLING OU T THE C{---..-}PON ON PAGE 8 23 OF TODAYS
PETPER{.........}{.......} TODAYSPAPER ANDMAILING I T TO SUMMER JOBS {--..-}
TSE BOS{----}N GLOBE . BOSTON . MAS S AC S EEEEEEEEEEE MAS SACSUSE T TS 02107.
TE E N{.---.}ERS MAY ALSO TAKE T5 E COUPONS T O TSE GLOBES OOWNTOWN OFFICE . A
T TSE C ORNER OF WASHINGTON 6LVO EEEEEEEEEEEEEEEE AND SCH OOL STRE E
TS.ORATITS MAI N OF FICE . 13 {.......} MORRISS E Y BLVO. OORC4 E STER A{.-.-}
COUPTANS {--..-}S T B E RE CEIVED {-.....}Y EEEI P M .WEONESOAY M{..--}
MAY{..--} T EEEEEEEEEEE . MAY2T5.JO8SE E KI EEEEEEEEEEE JO8SEEKERS MAY 8E
&SPECIFICAS TH E Y L IKE I N ME'NTIONING TSE SO{..-.-.}S OR OAYS TSEY AR E
AVAILA8TIIE{........} AVAILABLE FOR E MPLOYMENT. THE TYPE OF WTMRK 6 E YOES IRE
{---.-.}CANOO{---.-.}WV AT WI.---.} E S TH E Y E XPECT. THE
{.--..}SWILLAPPE#INTSE MAY30CLASS IFII EEEEEEEEEEEEEEEEET CL&SISE I E O SECTION
UNOER EH E EHE {.--..}ING SI EEEEEEEEE HEADING S IRE ABOSTON T E E NAGE RFOR TH
E SUMMER .
```

Figure 5a. MAUDE Translation of Plain Text.

```
IN AN ATTEMPT TO [xxxxx e] <ALLEVIATE> <YOUTH> UNEMPLOYMENT IN THE [xxxxx e]
CITY . <THE> SUNDAY GLOBE ON MAY 30 WILL <PUBLISH> FREE ADVERTISEMENTS FOR
<BOSTON> TEENAGERS SEEKING SUMMER JOBS . ANY BOY OR GIRL 13 TO [xxxxx e] 19
<WHO> LIVES IN BOSTON CAN PLACE A <JOB> WANTED AD WITHOUT [xxxxx e] CHARGE <BY>
FILLING OUT THE COUPON ON PAGE B 23 OF [xxxxx e] TODAYS PAPER AND MAILING IT TO
SUMMER JOBS <.> <THE> BOSTON GLOBE . BOSTON . [xxxxx e] <MASSACHUSETTS> 02107 .
TEENAGERS MAY ALSO TAKE <THE> COUPONS TO <THE> GLOBES DOWNTOWN OFFICE . AT
<THE> CORNER OF WASHINGTON <BLVD> [e] AND SCHOOL STREETS . OR AT ITS MAIN
OFFICE . 135 MORRISSEY BLVD . <DORCHESTER> . <COUPONS> MUST BE RECEIVED THEY S
PM . WEDNESDAY [xxxxx e] . MAY <26> . [xxxxx e] JOB SEEKERS MAY BE AS SPECIFIC
AS THEY LIKE IN MENTIONING <THE> <HOURS> OR DAYS <THEY> ARE [xxxxx e] AVAILABLE
FOR EMPLOYMENT . THE TYPE OF WORK <BE> (Y) DESIRE OR CAN DO OR <WHAT> WAGES
THEY EXPECT . THE ADS WILL APPEAR IN <THE> MAY 30 [xxxxx e] <CLASSIFIED>
SECTION UNDER <HE> [xxxxx e] HEADING <HIRE> A BOSTON TEENAGER FOR THE SUMMER .
```

Figure 6a. COMDEC Translation of Plain Text.

translator. An "@" in square brackets indicates an error-sign; "XXXXX" preceding the "@" indicates that a portion of the message, believed to be the part in error, was suppressed; and "< >" indicates a word obtained from the dictionary by assuming the sender made a mark-error. The COMDEC translator made four errors: transcribing "BY" as "THEY", "THEY as "<BE>", "THE" as "HE"; and not suppressing "BLVD" after the word "WASHINGTON". To handle the last error correctly the translator would need to know that in Boston, Washington is a street and not a boulevard. Less than three percent of the words in the message were in error. The operator made 33 errors in translating this message: a 20% error rate. Both COMDEC's result and the operator's result are quite readable.

Figure 5b shows a translation by a MAUDE-like translator of some chatter, a header and a portion of a ciphered message. Figure 6b shows the identical portion of Morse code as translated by the cooperative efforts of COMDEC and CATNIP. Both the operator and the COMDEC-CATNIP translations of the chatter and header were perfect. (The "{ }" at the beginning indicates that the enclosed sequence cannot be translated. Consequently the translation obtained from the moving threshold first stage translator was used. (This construct resulted from a splice in the audio tape.)

The COMDEC-CATNIP tandem translated 19 groups incorrectly and the operator 32. It is interesting that both the operator's translation and and the machine's translation had 13 incorrectly translated groups in common. In addition to translating groups incorrectly both the operator and the machine added 9 extra groups because the semantics of the error-signs were ambiguous.

One very encouraging aspect of the system is that on a DEC 20 it is 2 to 10 times faster than real-time, depending upon the number of irregularities and errors in the Morse code.

One should not draw hasty conclusions from these results. We have worked very hard to obtain the results we have but the job is not yet complete. For instance, the chatter and header constructs preceding the plain text message of Figure 6a are not yet understood by CATNIP. Furthermore, a different set of operators using

```
"MAUDE" decoding:

SALT:   {..-VVVV} {VVV}QSA? K

ROCK:   RRRRQSASQSA5 K

SALT:   RAE {..-.-.} METSA 5 EAEN METS AS UEN METENK 5 GTTC MATC TN A? K

ROCK:   RRRQRVK

SALT:   RRR {-..--} NWHR TFC HR TFC = = = NR 1 GR 200 044e = VOLVR QZQNT
        PQTQR TCY{..-....} VLHAP XGJIN NVRLC TJOMM {V.--.-..}AG QTZTZ
        PQQIH QRCNAGZQC E LYQE HSS GQQTR# H L{-.---..}QE B TXTB 01e 32
```

Figure 5b. MAUDE Translation of Chatter, Header, and Parts of a Ciphered Message.

```
COMDEC decoding:

SALT:   {{..-VVVV}} VVV QSA ? K

ROCK:   RR RR QSA 5 QSA 5 K

SALT:   RR UR QSA 5 UR QSA 5 UR QRK 5 QTC QTC GA ? K

ROCK:   R RR QRV K

SALT:   RR R NW NW HR TFC HR TFC [BREAK]
        [BREAK]
        [BREAK]
        NR 1 GR 200 <0445> [BREAK]

        VOLVR QZQNT PQTQR TCYUH VLHAP
        XGJIN NVRLC TJOMM VWLAG QTZTZ
        PQQIH QRCNA GZQCE [xxxxx e] GQQTR [xxxxx e] LYQQE BTXTB 01532
```

Figure 6b. COMDEC Translation of Chatter, Header and Parts of a Ciphered Message.

a different set of chatter conventions might require a significant amount of effort to augment the domain models to understand and translate the new chatter. Yet a human operator could adapt to the new operator's style and acquire knowledge about the new chatter conventions by listening to the conversations for a few hours or at most a few days.

F. ACKNOWLEDGMENT

This research was supported by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under contract number N00014-75-C-0661. We are grateful to J.C.R. Licklider for drawing our attention to the Morse code problem. We also wish to thank Stuart W. Galley and Janet L. Schoof for their editorial assistance.

G. REFERENCES

1. Freimer, M., B. Gold and A.L. Tritter. The Morse Distribution. Cambridge, Ma.: M.I.T. Lincoln Laboratory, January 1959.

2. Eisenstadt; Gold; Nelson; Pitcher; and Selfridge. "MAUDE (Morse Code Decoder)." Cambridge, Ma.: M.I.T. Lincoln Laboratory Group Report 34-57.

3. Gold, Bernard. Machine Recognition of Hand-Sent Morse Code. Cambridge, Ma.: M.I.T. Lincoln Laboratory.

4. Byrnes; Gold; Nelson; Pitcher; and Selfridge. "Some Results of the MAUDE Program." Cambridge, Ma.: M.I.T. Lincoln Laboratory Group Report 34-72.

5. Gold, B. "Machine Recognition of Hand-Sent Morse Code." Transactions of IRE, PGIT, IT-5, 17, (1959).

6. Smutek, John M., and Richard S. Gawlick. A Digital System for Recognising and Translating Morse Code. May 1968.

7. Poehler, P.L. "Computer Recognition of Hand-Sent Morse Code." S.M. Thesis, M.I.T. Department of Electrical Engineering and Computer Science, July, 1968.

8. Petit, R.C. "The Morse-A-Verter -- Copying Morse Code by Machine." QST, Vol. IV, No. 1:30-35, January, 1971.

9. Guenther, J.A. "Machine Recognition of Hand-Sent Morse Code Using the PDP-12 Computer." Air Force Institute of Technology, Wright-Patterson Air Force Base, December, 1973.

10. American Radio Relay League. The Radio Amateur's Operating Manual. ARRL Publication #24, Newington, Conn., 1972.

11. Webster's New Collegiate Dictionary. Springfield, Ma.: G. & C. Merriam Company, 1974.

12. Galley, S.W. and Pfister, Greg. MDL Primer and Manual. Cambridge, Ma.: Massachusetts Institute of Technology, 1977.

13. DECSYSTEM-20 User's Guide. Maynard, Ma.: Digital Equipment Corporation, 1978.

14. Eastlake, D.; Greenblatt, J.; Holloway, J.; Knight, T.; and Nelson, S. ITS 1.5 Reference Manual. Cambridge, Ma.: Massachusetts Institute of Technology, 1969.

15. Van Trees, Harry L. Detection, Estimation, and Modulation Theory; Pare II: Nonlinear Modulation Theory. New York: John Wiley and Sons, Inc., 1971.

16. Gardner, Floyd M. Phase Lock Techniques. New York: John Wiley and Sons, Inc., 1966.

17. McElwain, C.K., and M.B. Evens, "The Degarbler -- A Program for Correcting Machine-Read Morse Code," Information and Control. Vol. 5 (1962), pp. 368-384.

18. Anderson, T.A., "Machine Recognition of Hand-Sent Morse Code," M.I.T. Laboratory for Computer Science, Programming technology Division Document, SYS. 17.00, July, 1975.

19. Lebling, P.D., and Haverty, J. "Improvements to COMDEC." M.I.T. Laboratory for Computer Science, Programming Technology Division Document, SYS. 17.01, July 30, 1975.

20. Massachusetts Institute of Technology's Laboratory for Computer Science, Progress Report XII, July 1974 - July 1975, Cambridge, Massachusetts.

21. Woods, William A. "Transition Network Grammers for Natural Language Analysis." Communications of the Association for Computing Machinery. Vol. 13 No. 10, 1970.

22. Massachusetts Institute of Technology's Laboratory for Computer Science, Progress Report XIII, July 1975-1976, Cambridge, Massachusetts, p. 185.