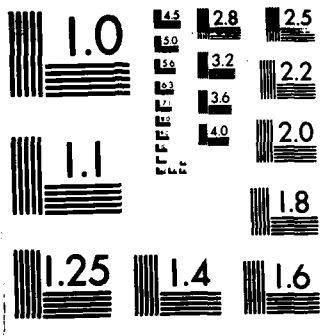
AD-A156 121 L-STEP SEQUENTIAL DECODING(U) ADAPTIVE SENSORS INC 1/1
SANTA MONICA CA I S REED 31 MAR 85 N00014-84-C-0720

UNCLASSIFIED F/G 9/4 NL



(12)

ASI Adaptive Sensors, Incorporated
2nd Floor, Suite 5, Santa Monica, CA 90405 (213) 396-4660

L-STEP SEQUENTIAL DECODING

Irving S. Reed

31 March 1985

Second Quarterly Report

Submitted to

The Office of Naval Research

Arlington, VA 22219

Under Contract #N00014-84-C-0720

by

ADAPTIVE SENSORS, INCORPORATED

L-STEP SEQUENTIAL DECODING

Irving S. Reed

1. INTRODUCTION

In the final report [1] of an earlier yearly contract with NAVAIR a new decoding technique was found for minimum error-path decoding of convolutional codes. This new technique was called piece-wise L-steps minimum-error decoding or more simply L-step decoding. This concept was developed further recently by the author and Mr. Henry Huey.

In a general sense L-step decoding is a method for decoding convolutional codes which appears to bridge the gap between the full implementation of a specific ideal decoder type and its less than ideal version in which memory paths are truncated. Although L-step decoding was introduced ~~in [1]~~ to save steps in an error trellis decoding algorithm, it can be extended to apply to most major types of decoding algorithms for convolutional codes, including Viterbi, stack, Fano, and their associated error-trellis algorithms.

From one point of view, the L-step method as applied to standard Viterbi decoding is simply a systematic method for realizing path memory truncation in L-steps or frames. The rule of thumb commonly used for path memory truncation is 4 to 5 times the constraint length K. Hence an overly conservative value for L or the path memory truncation is between 4 or 5 K.

2. L-STEP STACK SEQUENTIAL DECODING

Stack sequential decoding seems to benefit most from the L-step

decoding approach. In order to present the L-step sequential decoder, let S be a step (frame) counter and let N be the information block length in steps or frames. Then the basic L-step sequential stack decoding algorithm is as follows:

1. Set S = L.
2. Perform stack sequential decoding for L steps or until path in stack with maximum path metric M (the top element in stack) has length S.
3. Extend only those paths in stack to length S with path metrics which equal the maximum path metric M or whose path metric stays between M and M-T where T > 0 is called the path extension threshold.
4. At end of all path extensions in step 3 (above) of algorithm, retain only those paths in stack whose path metrics either tie the top path metric M, or lie between M-R and M where R > 0 is called the path retention threshold.
5. Update counter by setting S = S + L. If S < N, return to step 2 in algorithm. If S ≥ N, stop.

The path extension step 3 in the above algorithm is a vital part of the L-step decoding algorithm. It allows after the L-steps or frames for many of the lower entries in the stack to catch up or tie with the top entry. Without the path extension step in the algorithm the L-step approach would perform well against rather benign channel noise. This is because a severe error pattern which occurs during any L-steps of

such a decoding algorithm would probably cause the decoder to lose itself in the trellis.

The flexibility of the present algorithm over the original algorithm [1] in decoding with noisy channel conditions is due to the many possible choices of the parameters needed for performing the path expansion process and in maintaining the path memories used for back searching. It is also clear for the above L-step algorithm that for a large enough value of L, the primary L-step parameters that

$$\lim_{L \to \infty} [\text{L-step Algorithm}]$$

$$= \text{Stack Sequential Algorithm}$$

However, in reality, as discussed in Sec. 1, L may only have to be from one to 4 or 5 times the constraint length K, depending on the channel conditions.

The classical stack sequential decoding algorithm suffers from the following problems when actually implemented as a decoder:

- Finite buffer memoty overlows cause data erasures.
- The stack memory requirements can be excessive and stack overflow is a certainty over some channels.
- The time needed to sort the entries in an ever increasing stack increases with every step of the algorithm.

By contrast the L-step algorithm allows the decoder a chance to flush out many paths that with high probability will not reach again the top of the stack. Thus, the L-step decoder eliminates many of the classicial stack algorithm problems by never allowing the memory size

requriements to grow past some given limit. This causes the choise of L depend rather importantly upon the channel the decoder is expected to face.

The path extension threshold T is also an important parameter relative to L. If T is too large for a particular L, the path extension part of the algorithm can exceed the L-step part (step 2 of algorithm) in both complexity and stack size. The running of a simulator, developed recently by Mr. Henry Huey for the L-step algorithm has shown it is possible to find experimentally rules for correctly choosing L and T for a given channel.

3. PRELIMINARY SIMULATION RESULTS

Some preliminary performance data of the L-step decoding as compared with the standard stack decoder were obtained using computer simulation. The code and conditions of the test were as follows:

- (3.1) convolutional code with constraint length I=3 and $d_{free} = 7$.
- Generating matrix of code, $G_{(D)} = [1+D, 1+D^2, 1+D+D^2]$
- Initially information sequence has length 12 (recently this was extended to 85).
- L varied from 3 to 6 and threshold T varied from 3 to 9.
- Over channel binary code was corrupted by additive White Gaussian noise (WGN).
- Upon channel reception a hard decision receiver was used to detect the received binary code.

In the simulation the data presented to the decoder is a detected binary stream obtained from the additive WGN channel after a hard limiting receiver. There are two decoder types simulated: The stack sequential and the L-step decoder, both with enough computer memory so that there is no imposition of a buffer constraint.

The results of the initial simulation runs are seen in Fig. 1. These curves indicate that for the simple, rate 1/3, code that there is at most only about a 0.1 - 0.3 dB difference in performance throughout the range graphed.

Statistics on the computational effort involved in the two decoding techniques, the L-step and classical decoding techniques, the L-step and classical stack algorithm, are still in the process of being obtained. For the L-step decoder one statistic is the average size of the stack and the other is the number of expansion steps. These statistics for the L-step decoder will be compared with the average size of the stack for stack sequential decoding and the number of steps involved in the decoding algorithm. Preliminary results show that the rate of growth of the stack for the L-step method with proper threshold settings will be small compared with stack sequential decoding.

4. CONCLUDING REMARKS

Preliminary results from simulation indicate that L-step decoding, when applied to stack sequential decoding, performs almost as well as stack sequential decoding, but for noisy channels L-step decoding uses much less memory and computation time for the decoding process. There still remains a number of research areas for L-step decoding.

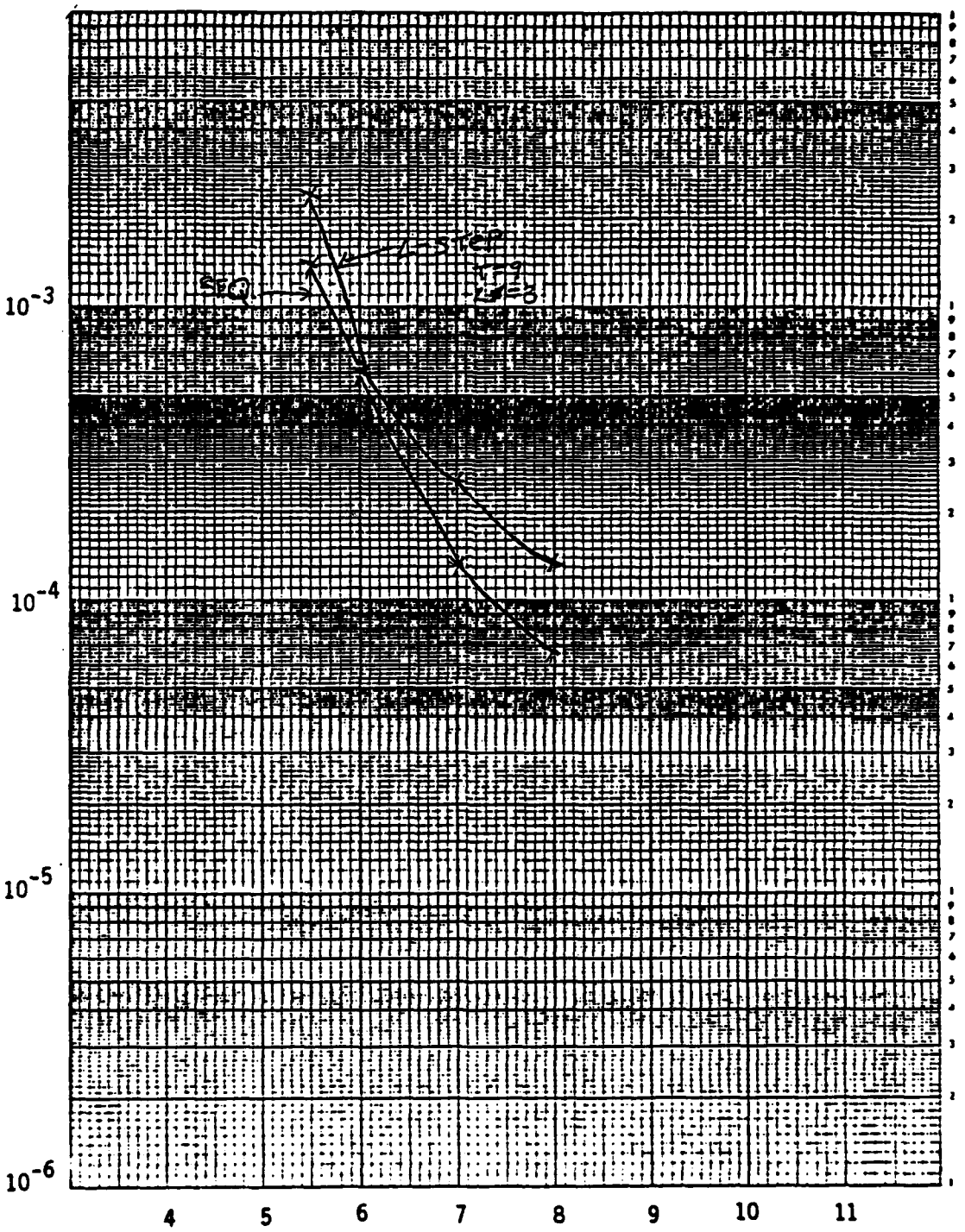


Figure 1 -- E_b / N_o (dB)

1. In the simulation the constraint lengths of the test code should be increased substantially so that more values of the parameters can be tested. These include larger block lengths, increased L, and sensitivity to threshold changes.

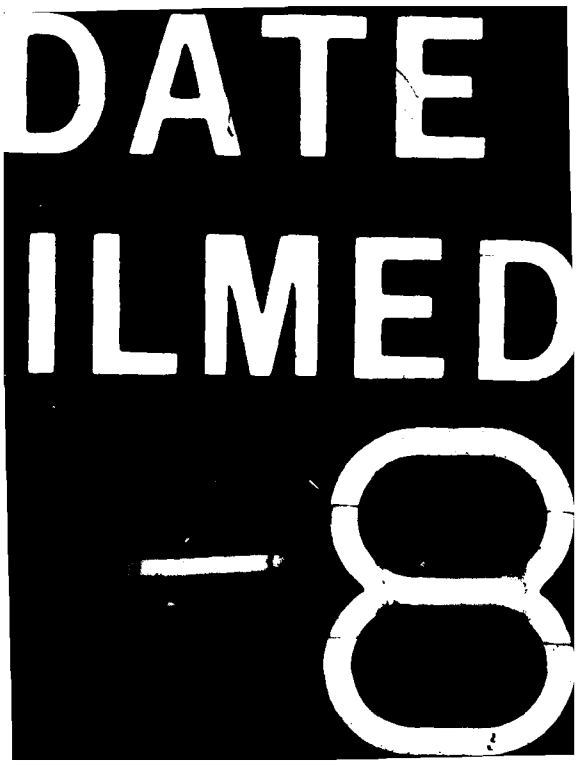
2. There is need to determine the savings in decoding steps, memory and the decoder speed of L-decoding versus the stack sequential algorithms.

3. It would be important to find out whether or not the parameters L, T and R can be determined adaptively.

4. The L-step algorithm should be applied to error trellis decoding with "pruned" error trellis to determine improvements in performance over standard decoding.

REFERENCE

1. Reed, I.S., "Minimum-Error Trellis-Path Decoders for Convolutional Codes," Adaptive Sensors, Inc., Final Report on Contract N00019-83-C-0075, Submitted to the Naval Air Systems Command, March 1984).


DATE
ILMED
-8