

A STUDY OF A SHORT TERM CORRELATION ALGORITHM

by

Siravara Vijayendra
B.E., Bangalore University, India 1979

A MASTER'S report

submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

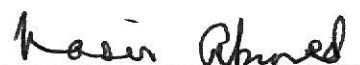
ELECTRICAL ENGINEERING

COLLEGE OF ENGINEERING

KANSAS STATE UNIVERSITY, MANHATTAN, KANSAS

1982

Approved by


Major Professor

Spec
Coll.
LD
2668
R4
1982
V54
C.2

A11202 246938

TABLE OF CONTENTS

	Page
Introduction	1
STCOR Algorithm	2
Statistical Considerations	7
Applications	14
Conclusions	29
Appendix I	31
Appendix II	39
References	48
Acknowledgements	49

**THIS BOOK
CONTAINS
NUMEROUS PAGES
WITH DIAGRAMS
THAT ARE CROOKED
COMPARED TO THE
REST OF THE
INFORMATION ON
THE PAGE.**

**THIS IS AS
RECEIVED FROM
CUSTOMER.**

INTRODUCTION

The crosscorrelation function of two sets of random data sequence is a measure of the linear dependence of one set of data on the other. For an exact estimate of the crosscorrelation function we need infinite length of data sequence. In practice, only finite lengths of data sequences are available. In such cases we can obtain only an estimate of the crosscorrelation function.

Consider two real-valued data sequences $X(n)$ and $Y(n)$, where n is the time index. Suppose these sequences are known over a finite length of time. Then an estimate of their crosscorrelation is given by

$$R_{XY}(\tau) = \frac{1}{(N-\tau)} \sum_{n=0}^{N-\tau} X(n) Y(n+\tau) \quad |\tau| \leq L \quad (1)$$

where

τ is the shift or lag index, L is the maximum shift and N is the total number of points in each sequence. In the interest of reducing the variance of the estimate, L is typically chosen to be less than 10% of N .

The crosscorrelation function $R_{XY}(\tau)$, in (1) is usually estimated via Fast Fourier Transform (FFT) techniques, which rely on the Wiener-Khinchin theorem, according to which

$$\hat{R}_{XY}(\tau) = FT^{-1} (G(w)) \quad (2)$$

where FT^{-1} denotes the inverse Fourier transform, $G(w)$ is an estimate of the cross Power Density Spectrum (PDS), which is usually computed using overlapping FFT's. This method is discussed in great detail and literature pertaining to it is readily available, e.g. see [1-3].

This frequency domain method is commonly employed to compute the crosscorrelation function in order to estimate the time delay between the sequences $X(n)$ and $Y(n)$. The estimate of time delay is given by that value of lag index τ at which $R_{XY}(\tau)$ attains its peak value.

In this report we restrict our discussion to a time domain approach, which we refer to as the Short term correlator (STCOR), for estimating the crosscorrelation sequence. The motivation for doing so is its relevance to two possible applications which are discussed in section IV.

The computational simplicity of this method makes it amenable to implementation in a microcomputer system. In addition, it has the capability of adapting to variations in the crosscorrelation function, in the event the data sequences $X(n)$ and $Y(n)$ are nonstationary.

II. STCOR ALGORITHM

The basic idea of this algorithm is to estimate the crosscorrelation $R_{XY}(\tau)$ at each lag value τ , using the first order recurrence relation

$$\hat{R}_{XY}(n, \tau) = \beta_1 \hat{R}_{XY}(n-1, \tau) + \frac{(1-\beta_1) X(n) Y(n+\tau)}{S_X(n-D) S_Y(n-D)}, \quad |\tau| \leq L \quad (3)$$

where $R_{XY}(n, \tau)$ is an estimate of the crosscorrelation at the time n and lag τ , $0 < \beta_1 < 1$ is a smoothing parameter, and D is a delay parameter; S_X and S_Y are estimates of the signal strengths of the sequences X and Y respectively, and are estimated recursively as

$$S_X(n) = \beta_2 S_X(n-1) + (1-\beta_2) |X(n)| \quad (4a)$$

$$S_Y(n) = \beta_2 S_Y(n-1) + (1-\beta_2) |Y(n)|, \quad 0 < \beta_2 < 1. \quad (4b)$$

It is apparent that (3) represents the process of shifting the sequence $Y(n)$ respect to $X(n)$ by τ time steps, and then averaging the related products. A total of $(2L+1)$ first-order computations are involved at each time step n , since the lag index take on values $-L, -L+1, \dots, -1, 0, 1, \dots, L-1, L$. This aspect is summarized in Fig. 1. The denominator terms S_X and S_Y causes $R_{XY}(n, \tau)$ to be a dimensionless quantity.

Consider the equation

$$R_{XY}(n, \tau) = \beta_1 R_{XY}(n-1, \tau) + (1-\beta_1) X(n) Y(n+\tau).$$

For a particular lag index τ , the above equation implements a first order filter whose transfer function is

$$H(z) = \frac{(1-\beta_1)}{(1-\beta_1 z^{-1})} \quad (5)$$

The input to this filter is the product $X(n)Y(n+\tau)$. Since at every time index n we require a total of $(2L+1)$ crosscorrelation values, each corresponding to a particular lag value, we can implement equation (3) using a bank of first order filters, as shown in Fig. 2.

To illustrate, the autocorrelation function estimate of a 45 Hz sinusoid sampled at 256 Hz per second is displayed in Fig. 3(a). This estimate was obtained using the overlapping FFT technique [1], with a 256 point FFT's and a 50% overlap for successive segments. The corresponding autocorrelation function estimate was obtained using STCOR algorithm and are displayed in Fig. 3(b) to 3(d). These were the outputs at 2 second intervals, which implies that Figs. 3(b)

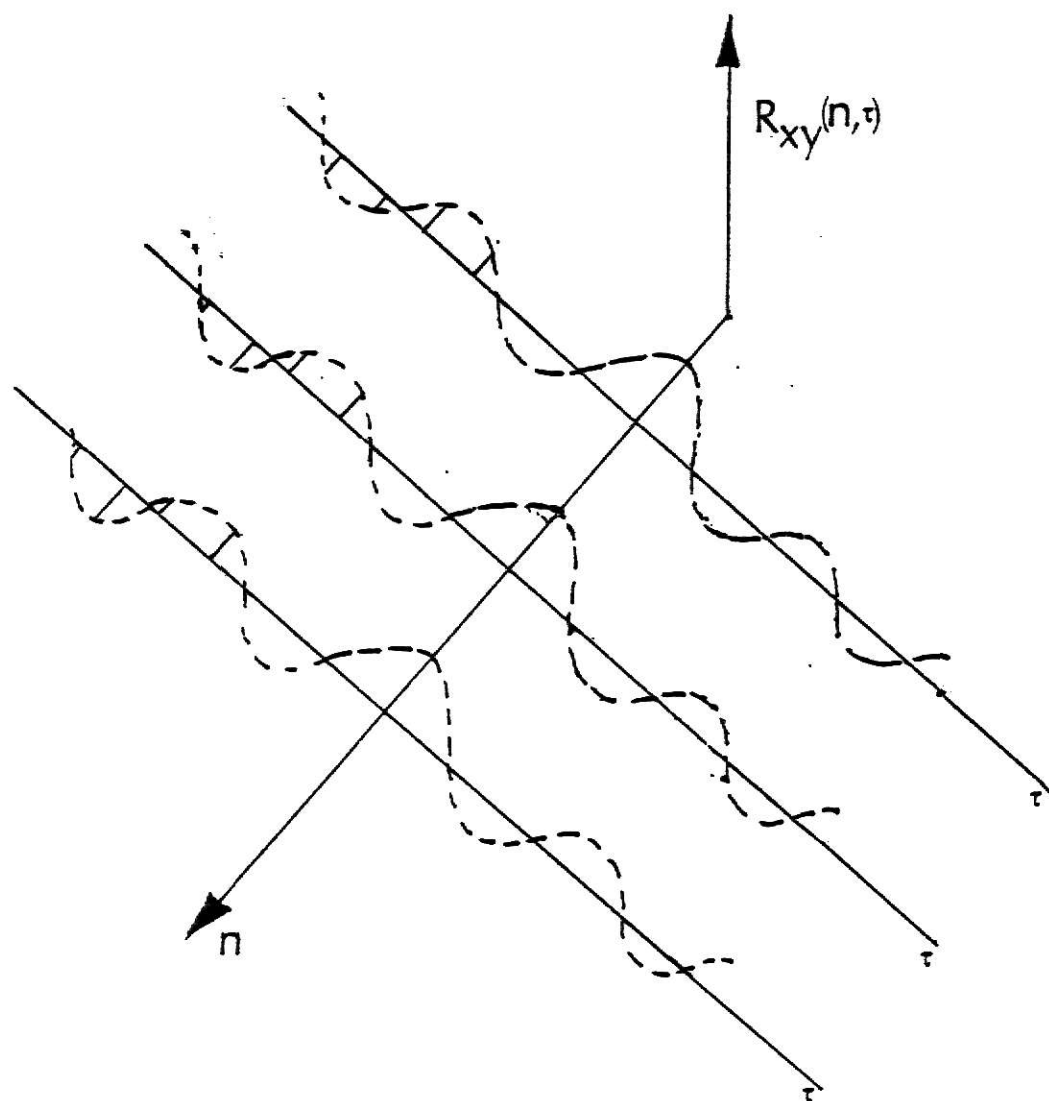


Figure 1. Summary of equation (3)

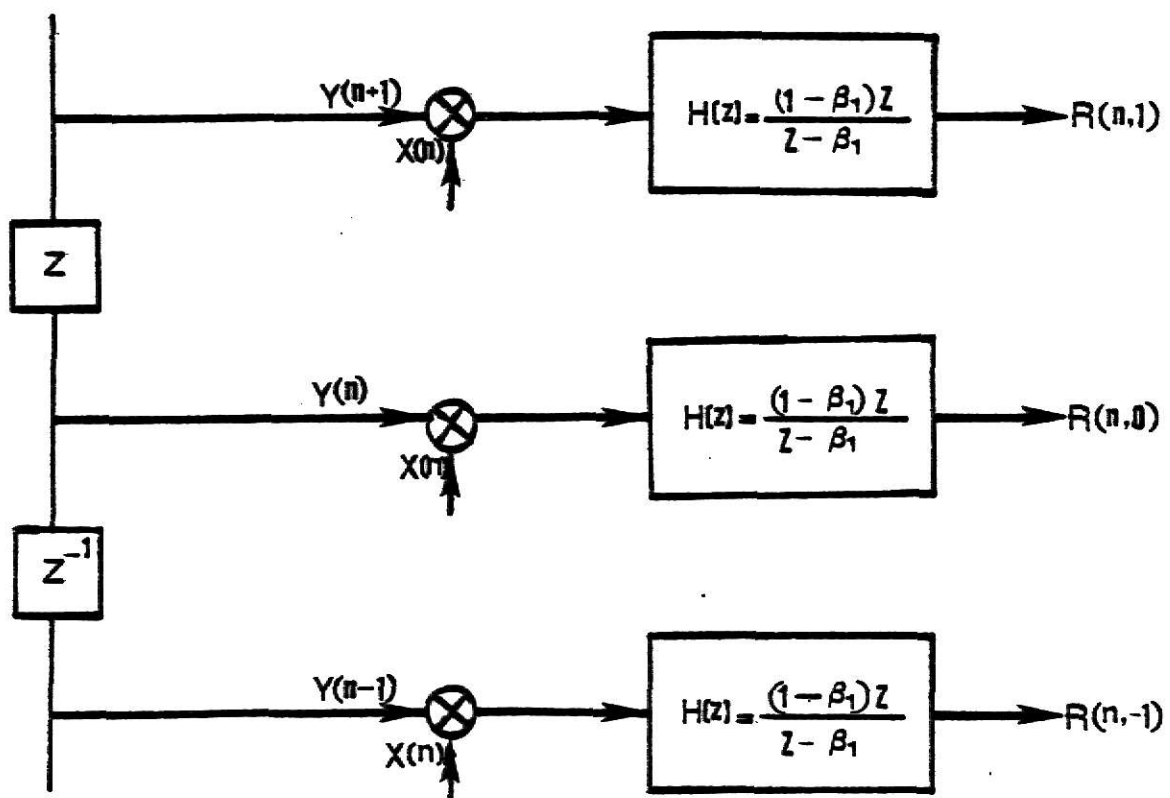


Figure 2. Bank of first-order filters.
For simplicity the structure is shown for lag values -1,
0 and 1 only.

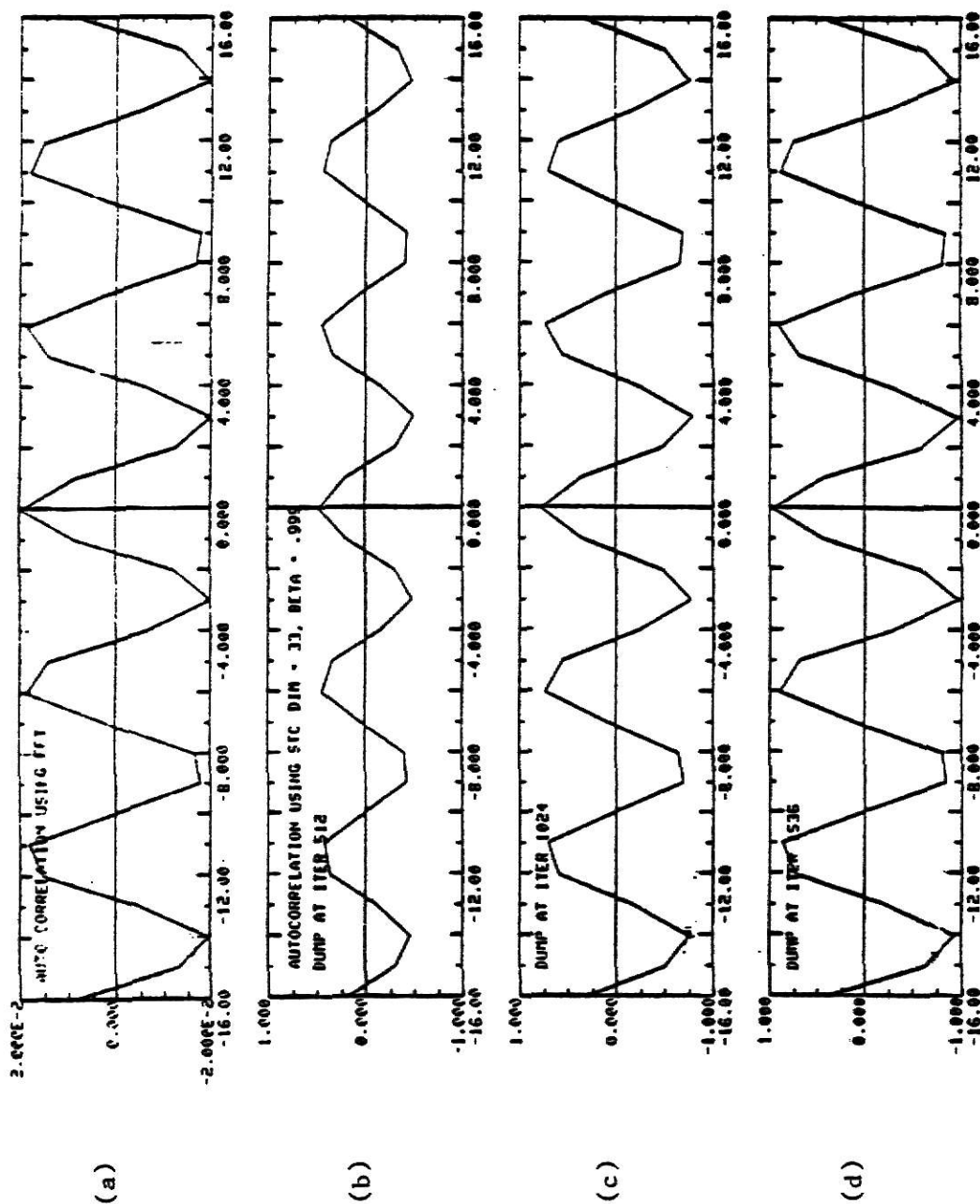


Figure 3. Autocorrelation function of 45 Hz Sinusoid.

through 3(d) represent the results of a 6 second data sequence. It can be observed that the estimates are also sinusoids of 45 Hz as they should be. We note that the shapes of the autocorrelation functions obtained by the FFT technique (Fig. 3(a)) and the STCOR algorithm (Figs. 3(b)-3(d)) are essentially the same. However, their amplitudes are different. This is because the estimates obtained via the FFT's are not normalized, while the estimates obtained via the STCOR algorithm are normalized by the signal strength in the sequences. Hence the estimate obtained via the STCOR satisfies the condition

$$-1 \leq \hat{R}_{XY}(n, \tau) \leq +1.$$

Similar autocorrelation estimates were obtained for 35 Hz - 45 Hz bandpassed white noise sampled at 256 sps. These results are displayed in Fig. 4. It's apparent that the FFT and STCOR methods yield the same crosscorrelation information. The parameters used for the STCOR algorithm in both the above experiments were as follows.

$$\begin{aligned} L &= 16 \\ \beta_1 &= 0.999 \\ D &= 0 \\ \beta_2 &= 0.98 \end{aligned} \tag{6}$$

III. STATISTICAL CONSIDERATIONS [4]

A statistical analysis for the estimates obtained from the STCOR algorithm can be carried out assuming that the input sequences $X(n)$ and $Y(n)$ are stationary Gaussian process with zero means. Since the denominator in (3) is merely a normalizing constant, it is ignored for convenience. Hence our discussion is restricted to the cross-correlation function defined as

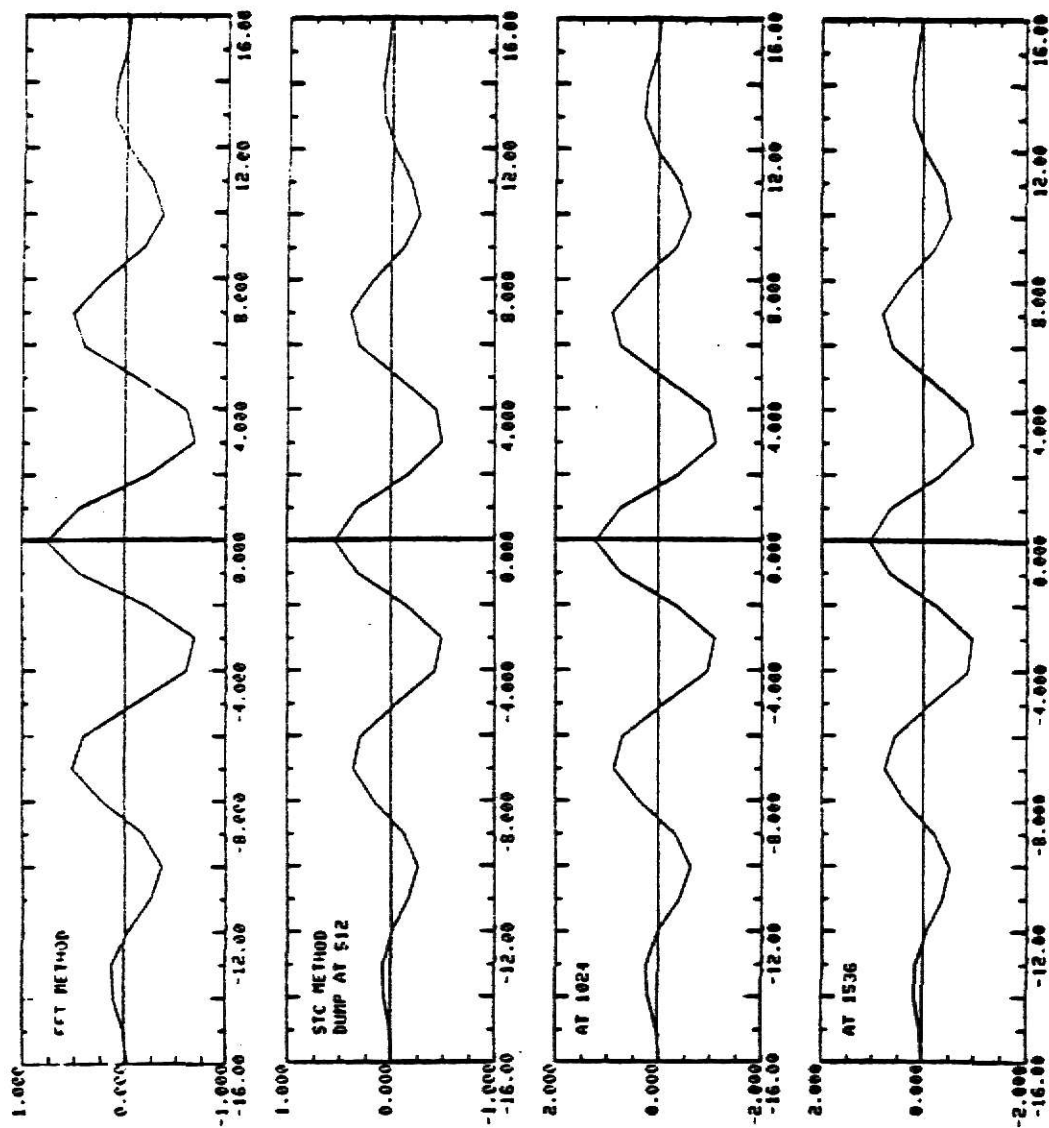


Figure 4. Autocorrelation function for bandpassed white noise.

$$\hat{R}_{XY}(n, \tau) = \beta_1 \hat{R}_{XY}(n-1, \tau) + (1-\beta_1) X(n) Y(n+\tau) \quad (7)$$

In (7), n is the time index and τ is the lag or shift index.

With some manipulations the recursive relation in (7) can be rewritten as

$$\hat{R}_{XY}(n, \tau) = (1-\beta_1) \sum_{i=1}^n \beta_1^{i-1} X(n-i) Y(n-i+\tau) + \beta_1^n \hat{R}_{XY}(\tau, 0) \quad |\tau| \leq L \quad (8)$$

for a fixed lag value τ .

We now define the true (exact) crosscorrelation function of the sequences $X(n)$ and $Y(n)$ as

$$R_{XY}(\tau) = E\{X(n-i) Y(n-i+\tau)\} \quad (8a)$$

with

$$R_{XX}(0) = E\{X^2(n-i)\} \quad (8b)$$

and

$$R_{YY}(0) = E\{Y^2(n-i)\} \quad (8c)$$

where the operator $E(\cdot)$ denotes the statistical expectation of the quantity within the brackets. Taking the expected value of (8), we get

$$E\{\hat{R}_{XY}(n, \tau)\} = (1-\beta_1) \sum_{L=1}^n \beta_1^{i-1} R_{XY}(n, \tau) + \beta_1^n \hat{R}_{XY}(0, \tau) \quad (9)$$

The limiting value of (9) as the time index, n , tends to infinity is given by

$$\lim_{n \rightarrow \infty} E\{\hat{R}_{XY}(n, \tau)\} = R_{XY}(n, \tau) = R_{XY}(\tau); \quad |\tau| \leq L \quad (10)$$

since,

$$\begin{aligned} \sum_{i=1}^n \beta_1^{i-1} &= 1 + \beta_1 + \beta_1^2 + \dots \\ &= \frac{1}{1-\beta_1}, \quad \text{for } |\beta_1| < 1 \end{aligned}$$

From (10) we see that the estimate of $R_{XY}(n, \tau)$ in (7) is an unbiased estimate.

The time constant $T(\beta_1)$ of the convergence associated with (9) can be found by letting $\hat{R}_{XY}(n, \tau) = e^{-1}$ and $\hat{R}_{XY}(0, \tau) = 1$, i.e.,

$$\beta_1^{T(\beta_1)} = e^{-1} \quad (11)$$

Solving (11), we get

$$T(\beta_1) = \frac{-1}{\ln \beta_1} \quad (12)$$

From the properties of logarithm, we know that

$$\ln \beta_1 \leq \beta_1 - 1 \quad (13)$$

Further, when $\beta_1 \approx 1$,

$$\ln(\beta_1) \approx \beta_1 - 1$$

From (12) and (13) we get

$$\begin{aligned} T(\beta_1) &= - \frac{1}{\beta_1 - 1} \\ &= \frac{1}{1 - \beta_1} \end{aligned} \quad (14)$$

For example, if $\beta_1 = 0.999$ then the associated time constant is 1000 samples. This is equivalent to 4 seconds for a data sequence sampled at 256 sps.

Next, we determine the steady state variance of the estimate $\hat{R}_{XY}(n, \tau)$. Equation (5) can be rewritten as

$$H(z) = \frac{(1 - \beta_1)z}{z - \beta_1} \quad (15)$$

Thus, we can interpret our estimator to be a discrete time system whose impulse responses is given by

$$h(n) = (1 - \beta_1) \beta_1^n, \quad n \geq 0$$

The impulse response is an exponentially damped sequence since $0 < \beta_1 < 1$. The input and output variances of this filter are thus related via the equation (see Fig. 5)

$$\sigma_{\text{out}}^2 = \sigma_{\text{in}}^2 \left\{ \sum_{n=0}^{\infty} h^2(n) \right\} \quad (17)$$

where, σ_{in}^2 is the variance of the input $X(n) Y(n+\tau)$, and σ_{out}^2 is the variance of the output $R_{XY}(n, \tau)$, which is desired. Equation (17) states that the variance of the estimate is product of the input variance and the sum of squares of the impulse response. The sum of squares of the impulse response $\sum_{n=0}^{\infty} h^2(n)$, is given by

$$\sum_{n=0}^{\infty} h^2(n) = (1 - \beta_1^2) \sum_{n=0}^{\infty} \beta_1^{2n} = \frac{(1 - \beta_1)^2}{1 - \beta_1^2}, \quad |\beta_1| < 1.$$

That is

$$\sum_{n=0}^{\infty} h^2(n) = \frac{(1 - \beta_1)}{(1 + \beta_1)} = \frac{1 - \beta_1}{2}, \quad \text{for } \beta_1 < 1. \quad (18)$$

The input variance σ_{in}^2 is found as follows,

$$\sigma_{\text{in}}^2 = E \left[\{ X(n) Y(n+\tau) \} \right]^2 - \left[E \{ X(n) Y(n+\tau) \} \right]^2, \quad (19)$$

That is

$$\sigma_{\text{in}}^2 = E(\alpha^2) - [E(\alpha)]^2$$

where,

$$\alpha = X(n) Y(n+\tau) \text{ and } E \{ X(n) Y(n+\tau) \} = R_{XY}(\tau).$$

Now, (19) can be written as

$$\sigma_{\text{in}}^2 = E(\alpha^2) - R_{XY}^2(\tau) \quad (20)$$

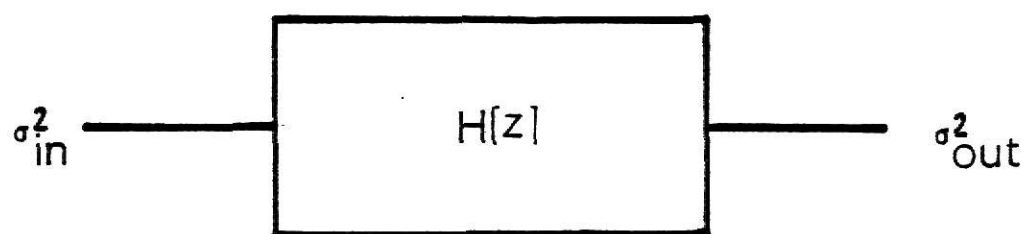


Figure 5. Pertains to equation 17.

The term $E(\alpha^2)$ is given by

$$E(\alpha^2) = E \{X(n) Y(n+\tau) X(n) Y(n+\tau)\} . \quad (21)$$

Since we are assuming that the input process is Gaussian we can make use of the fact that [5],

$$\begin{aligned} E[g_1 g_2 g_3 g_4] &= E[g_1 g_2] E[g_3 g_4] + E[g_1 g_3] E[g_2 g_4] + \\ &\quad E[g_1 g_4] E[g_2 g_3] \end{aligned} \quad (22)$$

where the g_i are zero mean Gaussian random variables. In this particular case

$$g_1 = g_3 = X(n)$$

and

$$g_2 = g_4 = Y(n+\tau) .$$

Thus (21) and (22) yield

$$E[\alpha^2] = 2 R_{XY}^2(\tau) + R_{XX}(0) R_{YY}(0) \quad (23)$$

Substituting (23) into (20), we obtain

$$\sigma_{in}^2 = R_{XY}^2(\tau) + R_{XX}(0) R_{YY}(0) . \quad (24)$$

Combining (17), (18) and (24) we get the result

$$\sigma_{out}^2 = \frac{(1-\beta_1)}{2} \left[R_{XX}(0) R_{YY}(0) + R_{XY}^2(\tau) \right] , \quad |\tau| \leq L \quad (25)$$

which is the steady state variance of the crosscorrelation estimate in (7).

From (25) it is evident that when $\beta_1 \approx 1$, the steady-state variance of the estimate is not zero, but is the product of a small number, which is the reciprocal of twice the time constant of convergence, and the quantity $(R_{XX}^2(0) R_{YY}^2(0) + R_{XY}^2(\tau))$. Hence the STCOR estimate is inconsistent.

IV. APPLICATIONS

We now discuss two possible applications that are relevant to point sensor intrusion-detection schemes.

1. Directionality

In certain intrusion-detection applications, it may be necessary to ignore alarms due to sources that are located in the region to be protected. To illustrate, we consider Fig. 6, which shows geophone array. Geophones are electromechanical transducers which produce an output signal voltage proportional to the velocity of ground motions. The output of each geophone is processed by a detection algorithm, which involves an adaptive digital predictor. Details of the detection algorithm are given in [6], which is reproduced in Appendix I. The output of detection algorithm is a 0 and 1, where output of 1 indicates an alarm condition.

Referring to Fig. 6 it is apparent that false alarms can originate in a number of ways; e.g. traffic on a road close to the sensor array, or a thumping action due to some construction activity. In such cases, it is clear that an intrusion detection system must be capable of determining source direction, i.e., whether the source is north or south of the geophone array. Then alarms due to a source north of the array in Fig. 6 will be classified as possible intrusion, while those due to sources south of the array will be ignored.

A possible approach to solve this problem could be to use pairs of geophones as depicted in Fig. 7. The basic idea is to determine whether a source is closer to the geophone 1 or geophone 2 in Fig. 7. It is apparent that signals due to sources in the region to be protected will arrive at geophone 2 first and then at geophone 1.

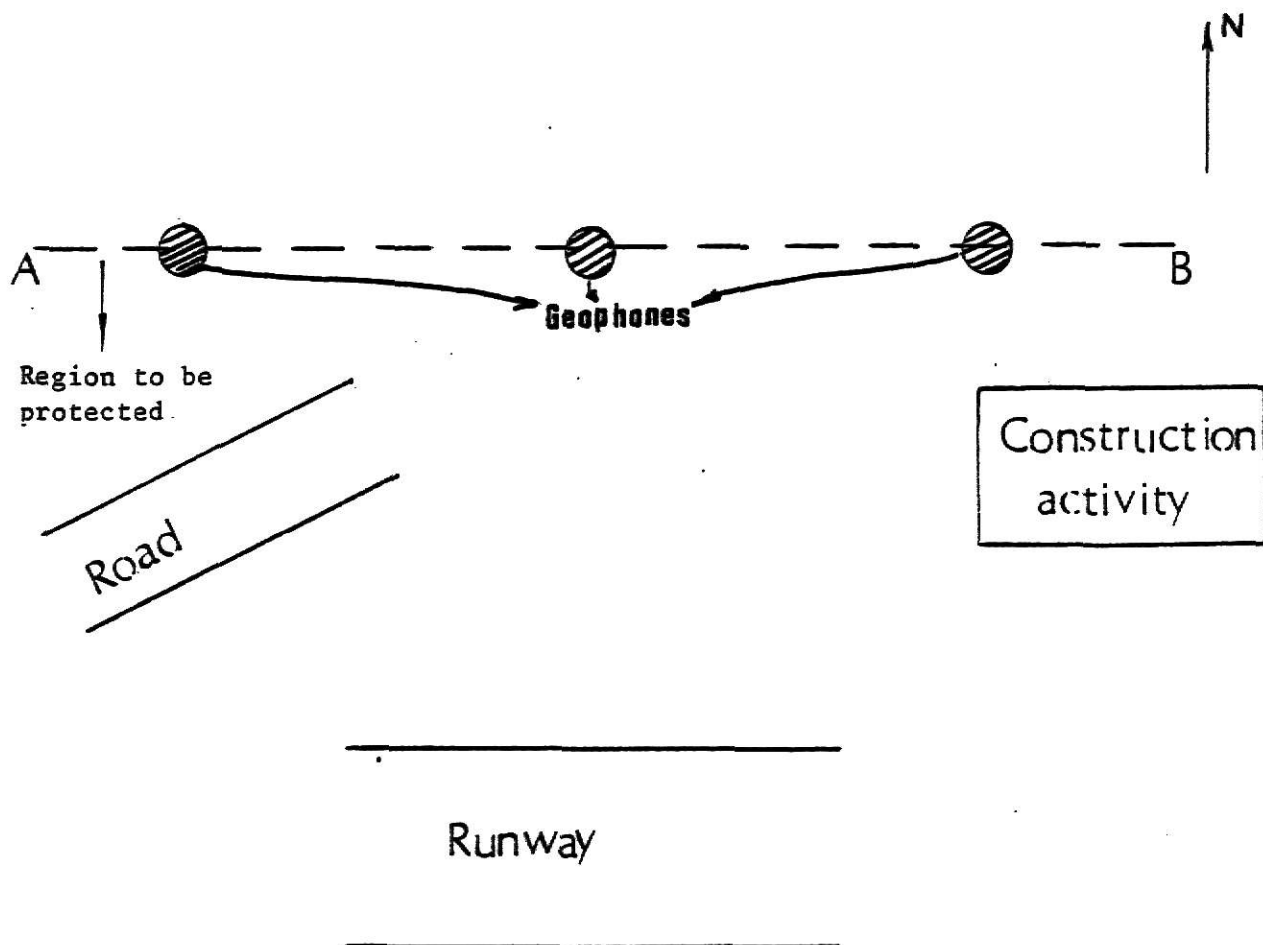


Figure 6. Pertains to directionality.

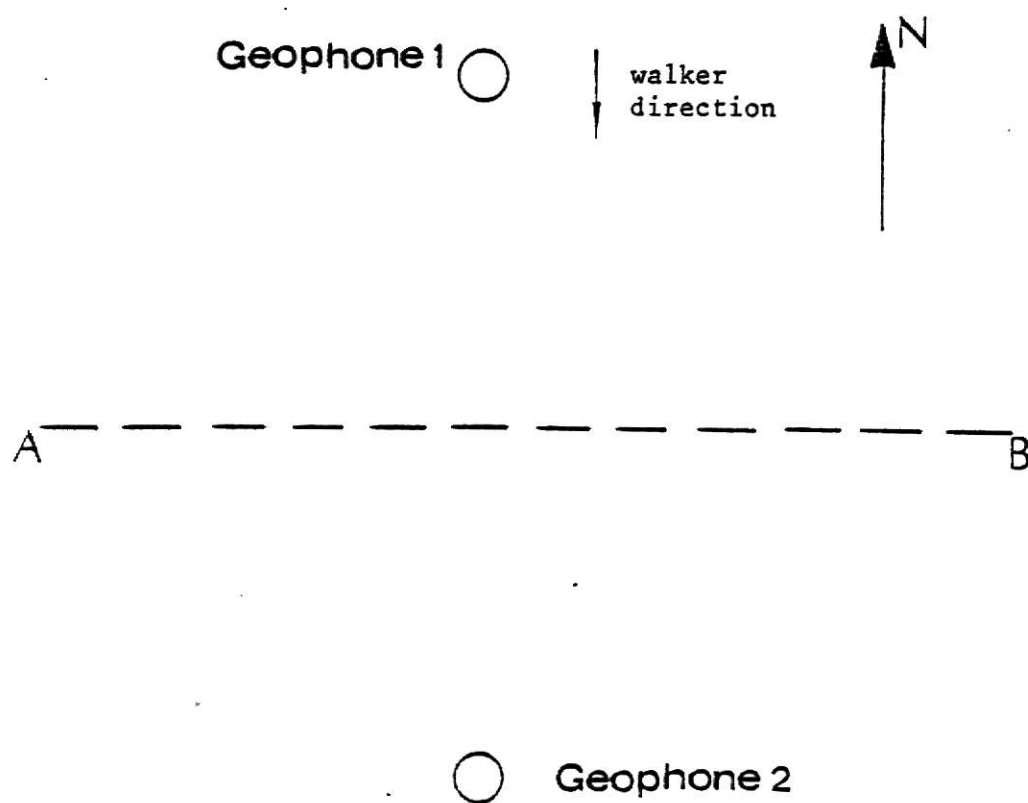
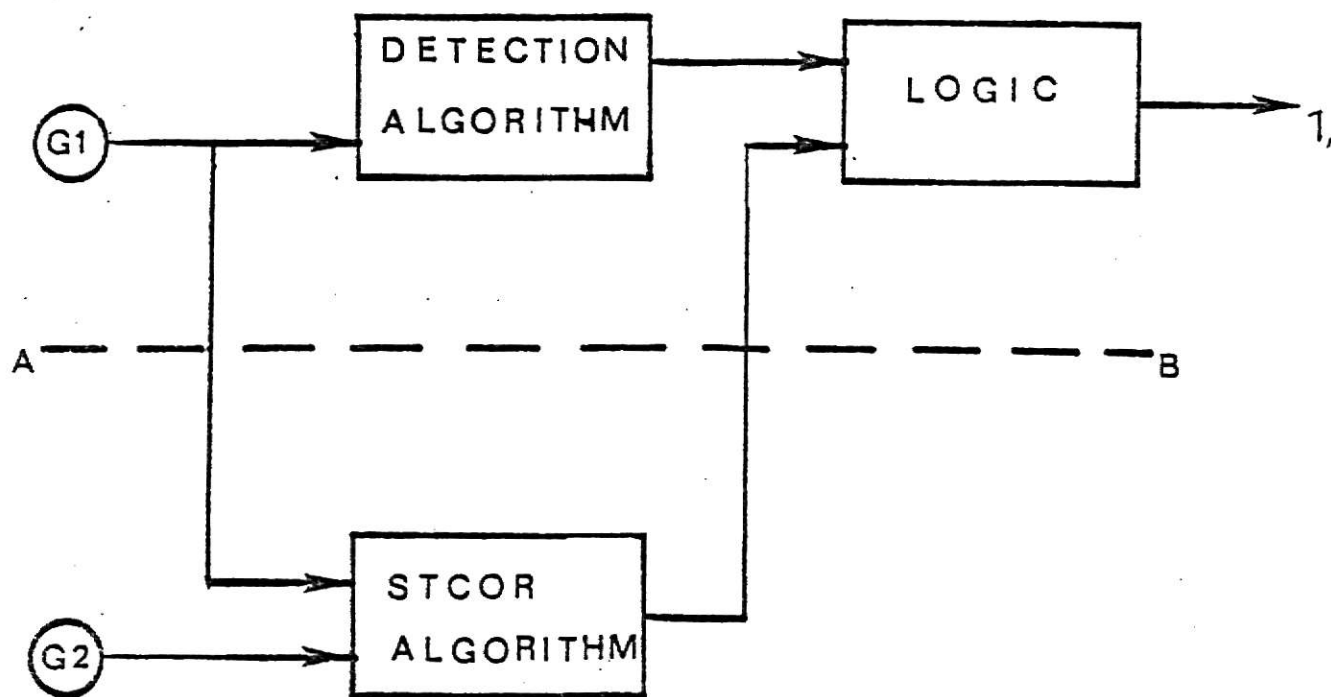


Figure 7. Pertains to directionality experiment on walker data.

Thus there will be time delay between the arrival times of signals at the two geophones. There are several algorithms that can be used to estimate the related time delay. However, most of these involve frequency domain crosscorrelation techniques [1-3], and thus would be difficult to implement in a microcomputer system which would be part of a field unit. As such, we propose the STCOR algorithm, as depicted in Fig. 8. Here the output of the STCOR algorithm is a 0 and 1, depending on whether the source location is to the south or north of the line AB, respectively in Fig. 8. A simple logic stage utilizes the outputs of the detection algorithm, and the STCOR algorithm to indicate a final output of a 1 or 0 to indicate a qualified alarm or no alarm condition, respectively. For illustration purposes we now present some experimental results.

Experiment 1: The data for this experiment was acquired as illustrated in Fig. 7. A person (intruder) walked parallel to the line joining geophones 1 and 2, and the output voltages of these geophones were digitized and recorded. The sampling frequency employed for A/D conversion was 256 Hz. The data is illustrated in Fig. 9(a). The intruder walked in the south direction, and a crossover to the other side of the line AB in Fig. 7 occurred around 14 seconds into the data file. This crossover is indicated by the symbol ' \uparrow ' in Fig. 9(a).

For each value of the time index n in (3), the value of $\hat{\tau}$, for which $\hat{R}_{XY}(n, \hat{\tau})$ attains the peak value (see Fig. 1) was stored. The other parameters for estimating the crosscorrelation function, were $L = 16$, $\beta_1 = 0.999$, $\beta_2 = 0.98$ and $D = 0$. These values



G1 and G2 = Geophone

Figure 8. Proposed scheme for intruder detection and qualifying the alarm.

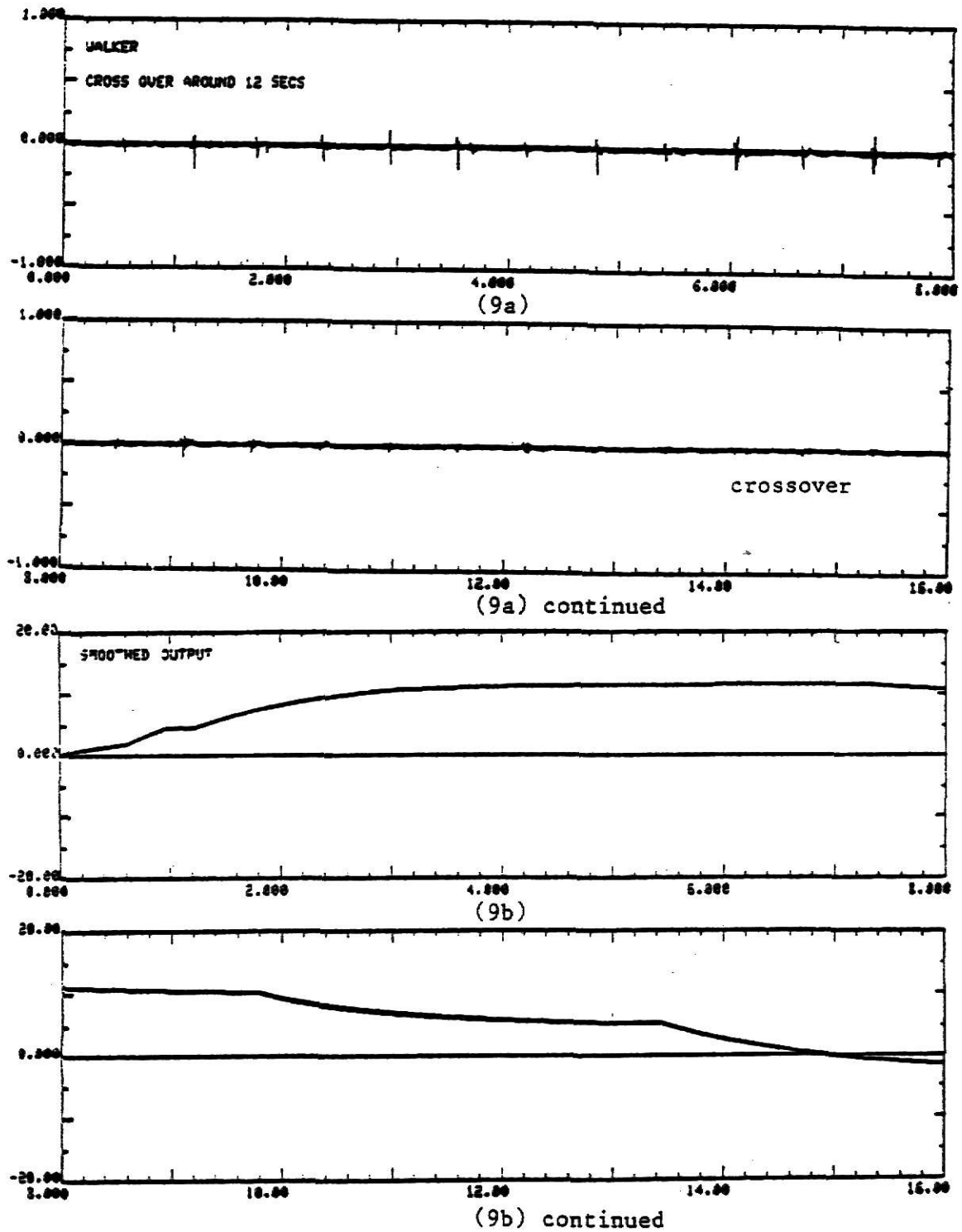


Figure 9. Experimental results on walker direction.

of $\hat{\tau}$ were then smoothed using the first order smoothing filter

$$\gamma(n) = \beta\gamma(n-1) + (1-\beta)\hat{\tau}(n) \quad (26)$$

where $\gamma(n)$ is the smoothed estimate of τ . Here $\beta = 0.997$ was used as the smoothing parameter. The resulting $\gamma(n)$ is plotted in Fig. 9(b).

From Fig. 9(b), we can see that the TDF (time delay function) $\gamma(n)$ changes sign around 15 seconds, indicating a crossover around the midpoint of the line joining the geophones in Fig. 7. Clearly, this compares well with the actual crossover point which was around 14 seconds. The value of TDF can be encoded as 1 or 0, to indicate the directions north or south of the line AB respectively; see Fig. (8). Thus the alarm conditions of the detection algorithm can be qualified, depending upon whether intruder is north or south of the line AB in Fig. 7.

Experiment 2: A vehicle (pickup truck) was driven parallel to a geophone pair; see Fig. (10). The data files consisted of 48 seconds of data, of which the segment from 15 to 39 seconds acquired at geophone 1 is displayed in Fig. 11(a). The vehicle crossed over the line AB around 31 seconds.

Crosscorrelation estimates obtained via the STCOR algorithm are displayed in Fig. 11(b) and Fig. 11(c) respectively. These estimates were obtained at 23 and 39 seconds into the data file, respectively. The corresponding TDF is plotted in Fig. 11(d), which indicates that the truck crossed the line AB in Fig. 10 around 33 seconds. This result again compares well with the actual value of 31 seconds when the actual crossover occurred. This illustrates that the STCOR algorithm is capable of tracking the source direction

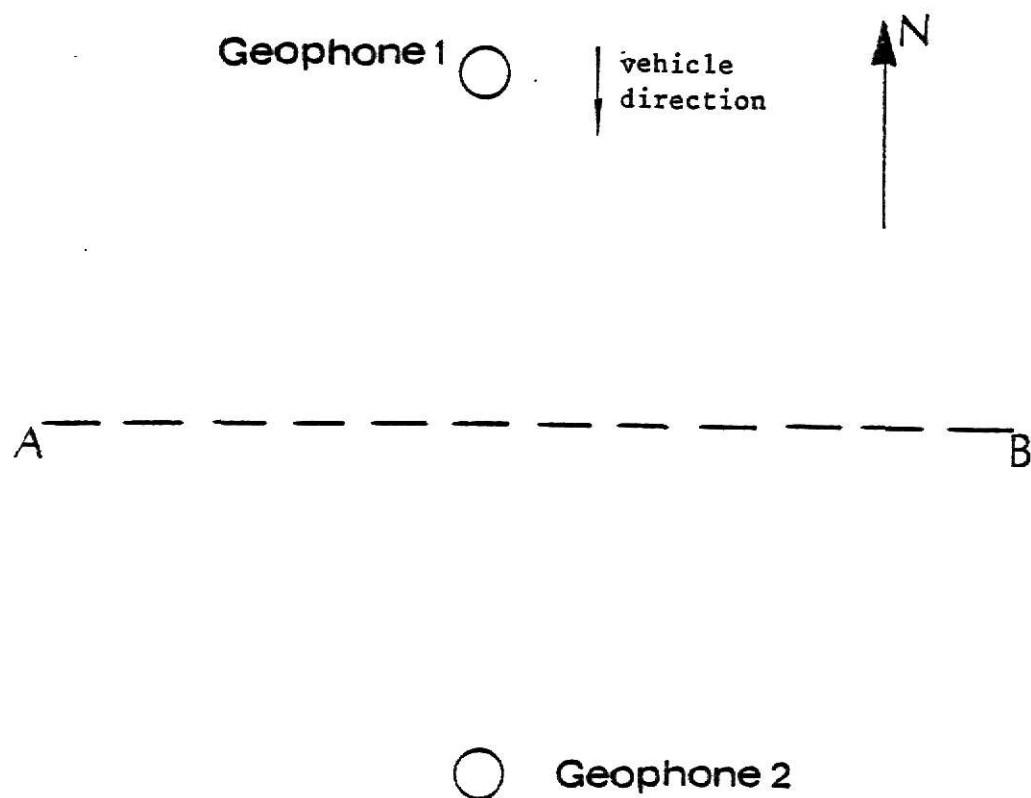


Figure 10. Pertains to directionality experiment on vehicle data.

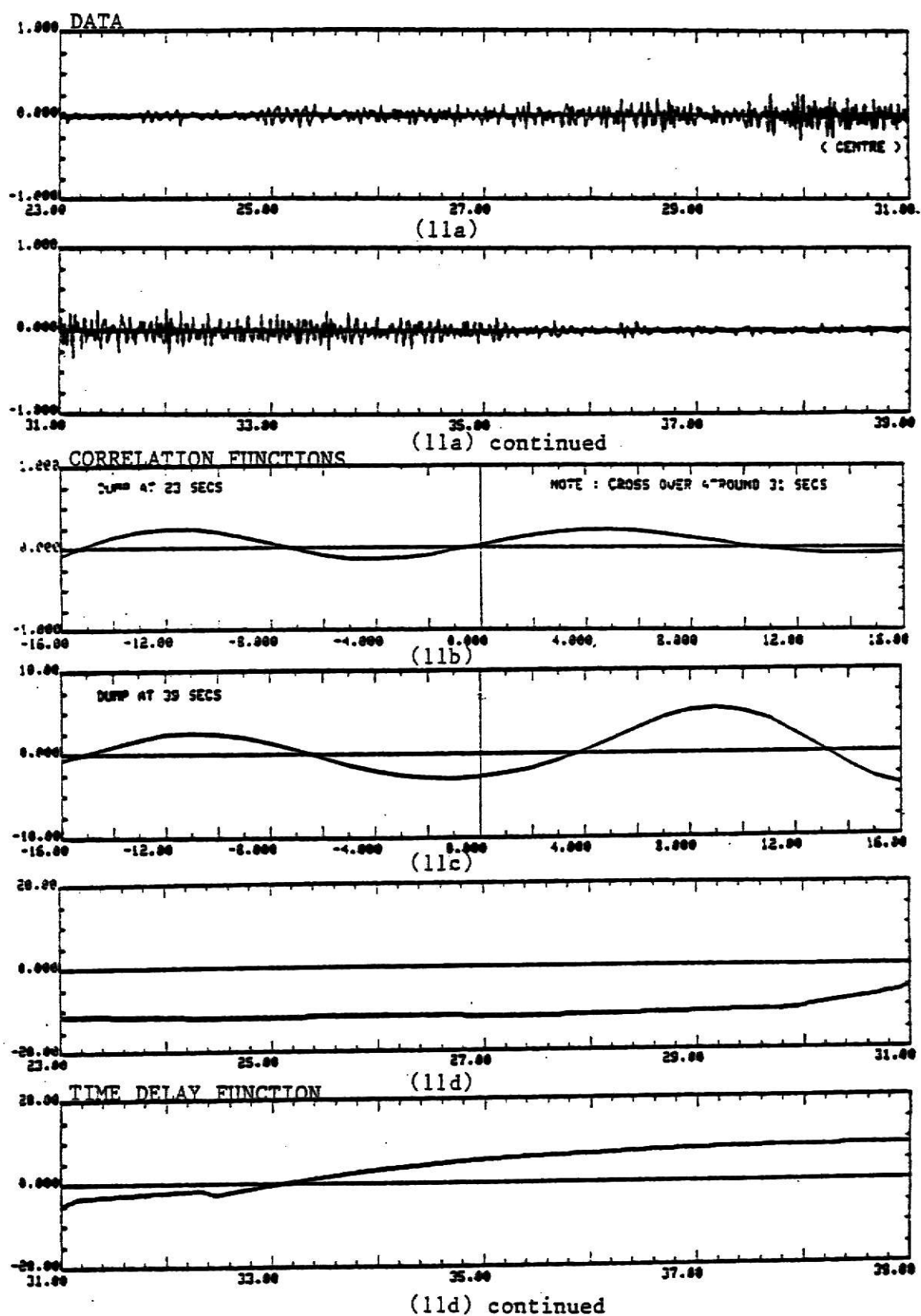


Figure 11. Experimental results on vehicle data.

with reference to the line AB in Fig. 7.

The parameters used for Experiment 2 were identical to those in Experiment 1; i.e. $L = 16$, $\beta_1 = 0.999$, $\beta_2 = 0.98$, $D = 0$ and $\beta = 0.997$.

2. Estimation of dominant frequency component:

This application concerns sinusoidal-like signals, where we are interested in continuously estimating the dominant frequency component in such signals. To illustrate, we present 32 seconds of helicopter data which is plotted in Fig. 12(b). This represents the response of a geophone placed on the surface of earth, as depicted in Fig. 12(a) with a helicopter approaching from the west.

The acoustical waves which impinge the earth create seismic waves which propagate and cause the geophone to respond. The power density spectra of the data in Fig. 12(b) is displayed in Fig. 12 (c). From these spectra we can observe the sinusoidal nature of the geophone output signals. We note that there is a dominant component around 24 Hz. Our objective is to estimate this component on a continuous basis. To this end, we use STCOR algorithm in (3) to estimate the autocorrelation function $\hat{R}_{XY}(n, \tau)$. The parameters used for the STCOR algorithm were $\beta_1 = 0.999$, $\beta_2 = 0.98$ and $L = 16$. Hence the associated time constant is 1000 samples (or 4 seconds), at 256 Hz sampling frequency. The autocorrelation is computed noting that it is symmetric about the point $\tau = 0$, i.e. $\hat{R}_{XY}(n, -\tau) = \hat{R}_{XY}(n, \tau)$. With $L = 16$, the total length of $\hat{R}_{XY}(\tau, n)$ is thus given by

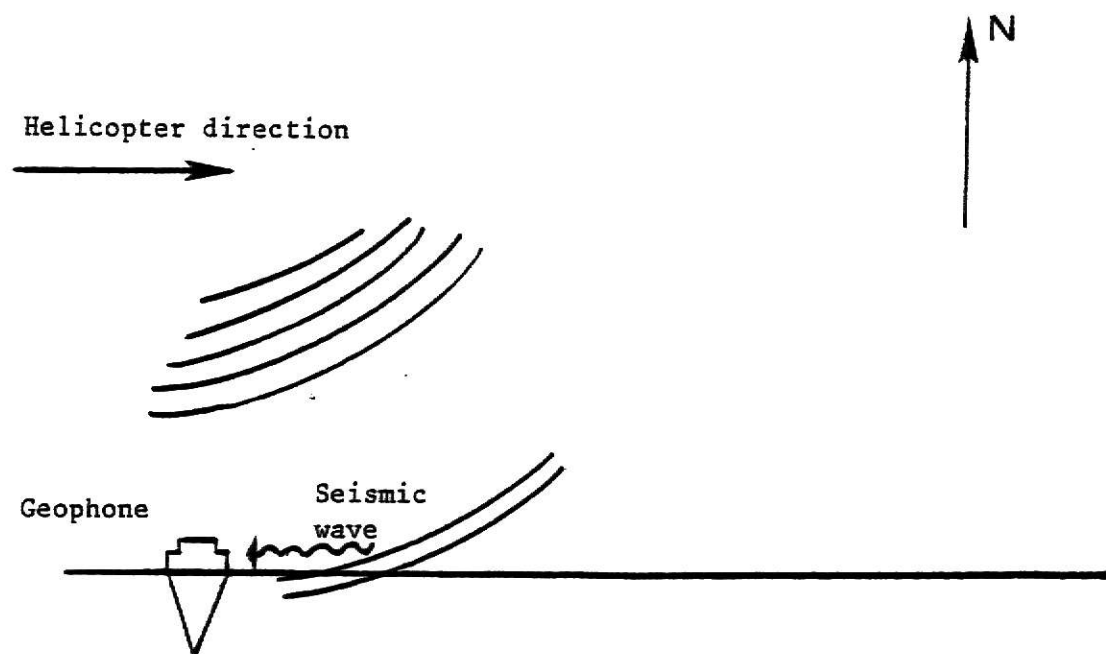


Figure 12(a). Pertains to Helicopter data.

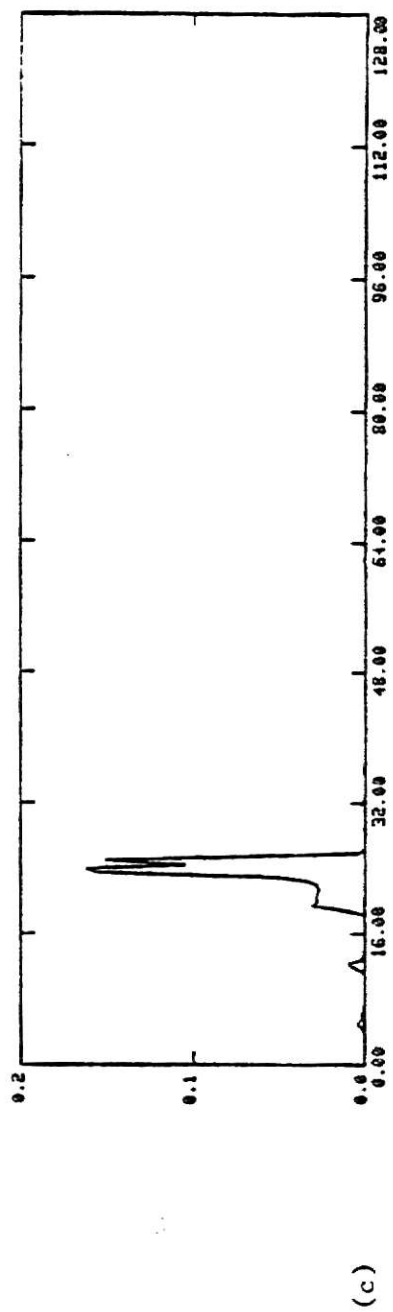
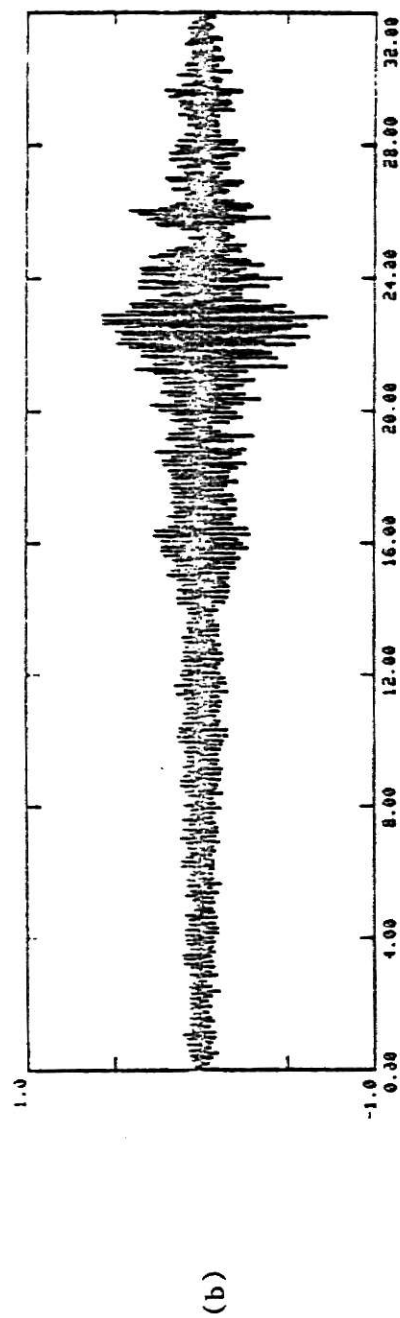


Figure 12(b) Helicopter data.

Figure 12(c) Power density spectrum of helicopter data.

$$T = 32/256 = 1/8 \text{ second}$$

since the sampling frequency is 256 Hz. Thus If $Z(n)$ denotes the number of sign changes that occur in $\hat{R}_{XX}(n, \tau)$ at time n , the frequency associated with the dominant component in the input signal at time n , say $f(n)$, is estimated as

$$\begin{aligned} f(n) &= Z(n)/2T, \\ &= 4Z(n) \end{aligned}$$

since $T = 1/8$ seconds.

As an example, plots of $\hat{R}_{XY}(n, \tau)$ are displayed in Fig. 13 at 8, 16, 24 and 32 seconds into the data file. We observe that they all tend to be sinusoidal functions and the number of sign changes in each of them is 6. Thus the dominant component at the instants 8, 16, 24 and 32 seconds is 24 Hz.

Our approach is to construct a histogram for the dominant component variable $f(n)$; i.e. the number of times the component $f(n)$ occurs during a specified number of iterations. To illustrate, histograms are obtained over 8-second intervals for the helicopter data and are displayed in Fig. 14, at 8, 16, 24 and 32 seconds into the data file. We note that the peak values of these histograms are at 2048. This implies that the dominant component at each of 2048 iterations (= 8 second interval) is 24 Hz. Thus the dominant component in the helicopter data is 24 Hz as estimated by the STCOR algorithm, with a time constant of 1000 samples. This value for the dominant frequency component agrees with the power density spectrum information in Fig. (12c) which also indicates that most of the power is around 24 Hz. The PDS estimate employed overlapped FFT's and used all the 48 seconds of data.

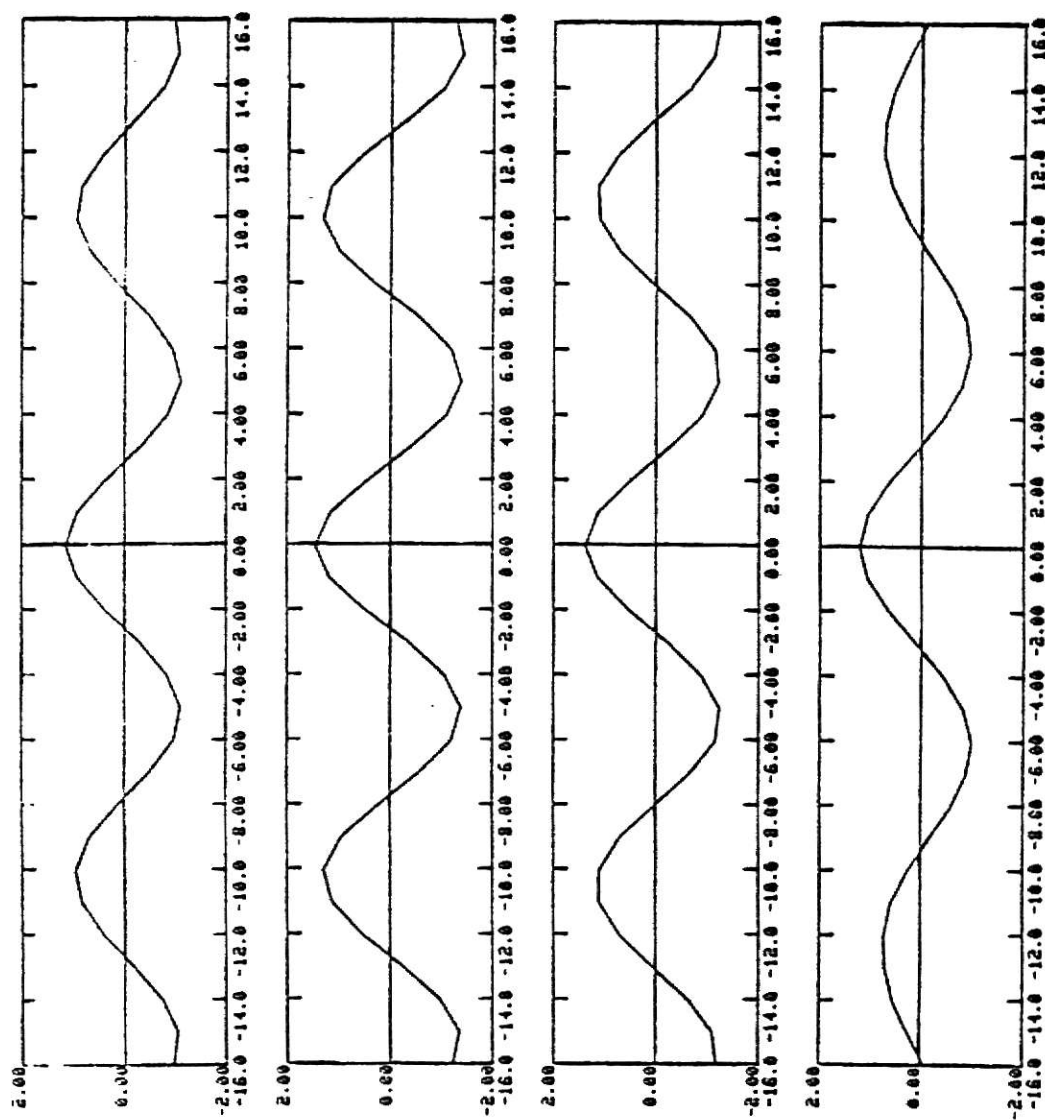


Figure 13. Correlation plots for helicopter data, at the end g

8, 16, 24 and 32 seconds.

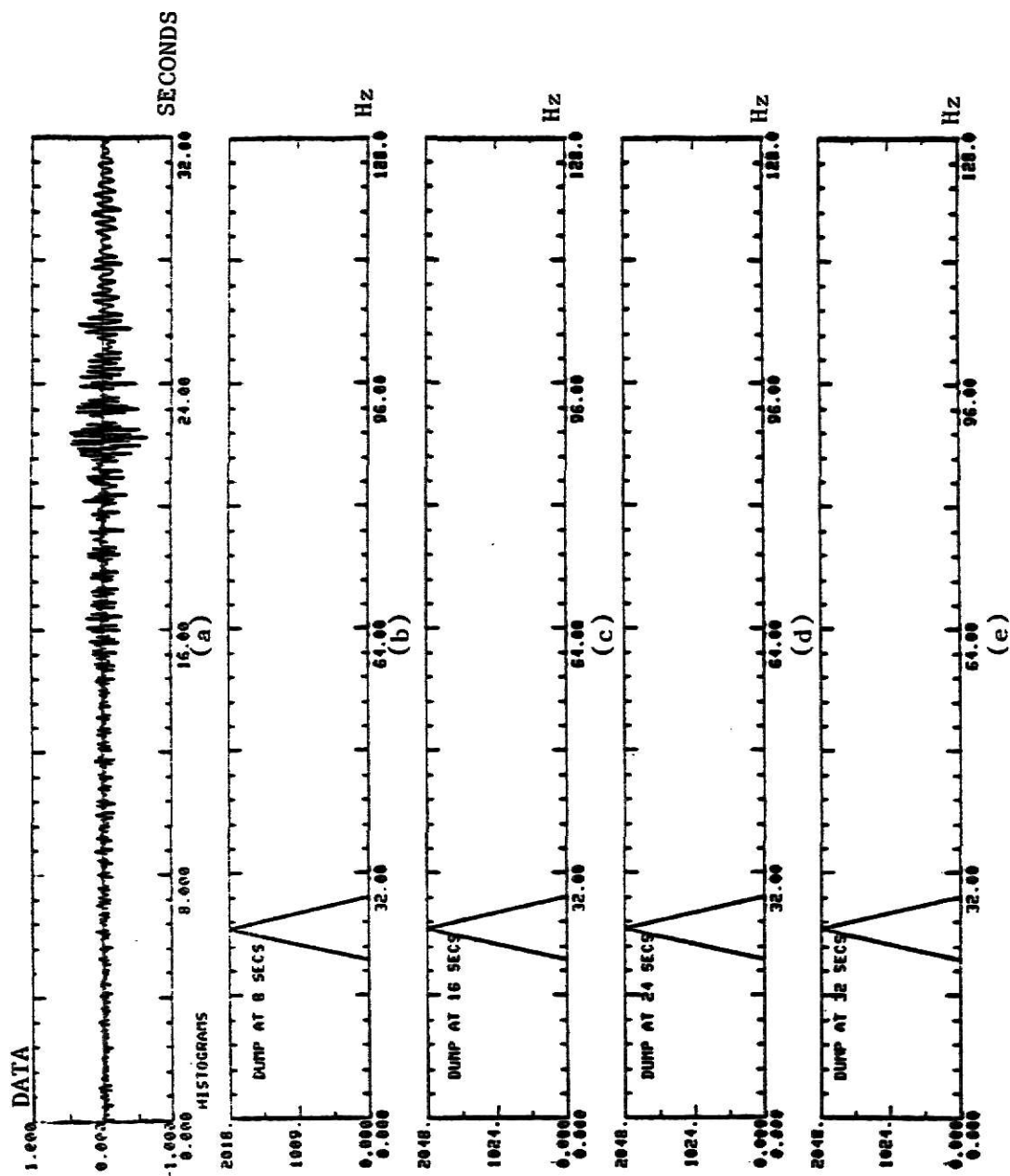


Figure 14. Experimental results on Helicopter data.

It is apparent that the STCOR algorithm is capable of tracking the dominant frequency component in sinusoidal-like data, in the event the dominant component is changing. This aspect is illustrated via an experiment on synthetic data, which is displayed in Fig. 15. The frequency components of the sinusoids are changed over each segment of the data. The data consists of 6 segments of 45, 60 and 90 Hz sinusoids. The additive noise is white, and the SNR is approximately 0 dB. From Figures 15(b) through 15(g), it is clear that the algorithm tracks the frequency of sinusoid as it changes from segment to segment.

V. CONCLUSION

This study of STCOR algorithm shows that the algorithm could be useful for two possible applications. They are

1. Direction determination in a point sensor intruder detection scheme, and
2. Estimation of the dominant component in sinusoidal-like signals and tracking them.

An attractive feature of the STCOR algorithm is that it can be easily implemented on a microcomputer system, and hence is a promising candidate for real time applications.

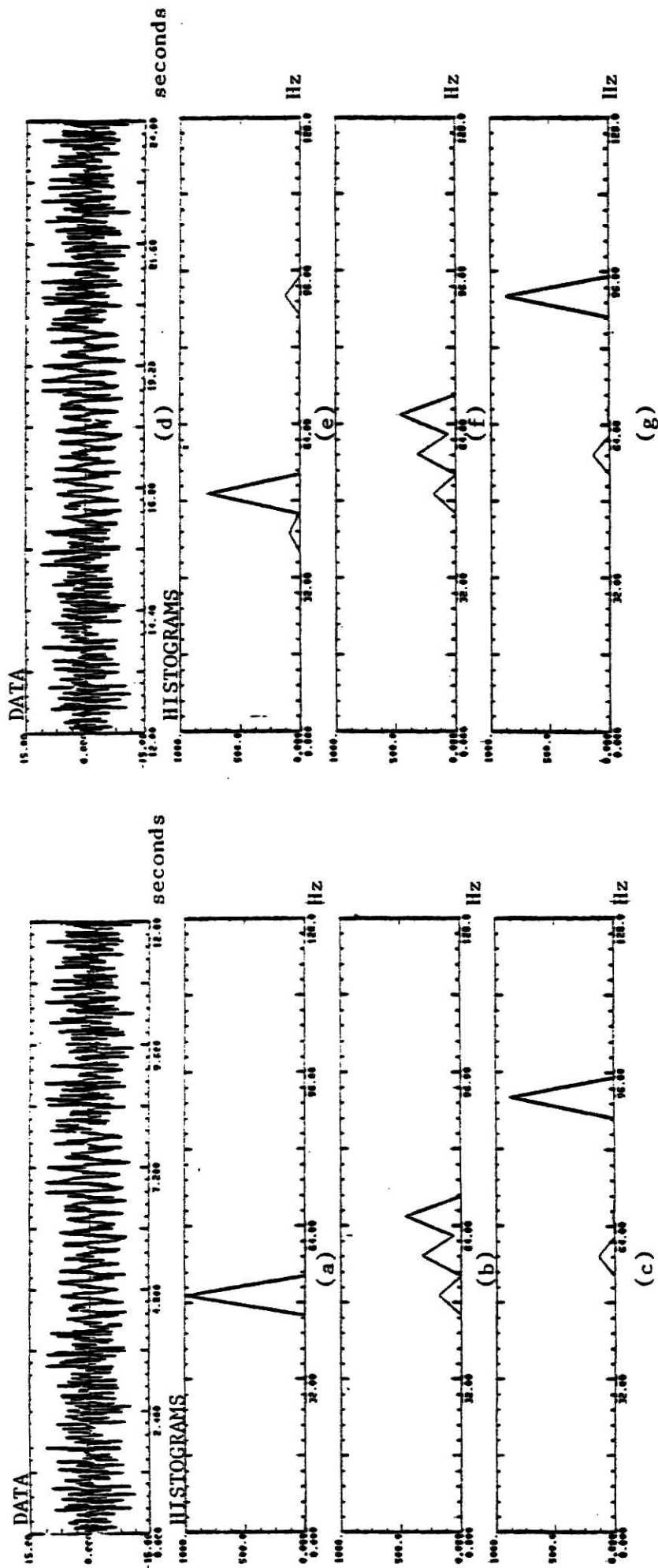


Figure 15. Experimental results on synthetic data.
The data consists of White noise and Sinusoids of frequency
45, 60 and 90 Hz.

APPENDIX I

Paper presented at the "15th Asilomar conference on Circuits and Systems",
November, 1981.

POINT SENSOR ADAPTIVE SIGNAL PROCESSING FOR INTRUSION-DETECTION*

N. Ahmed and S. Vijayendra
Electrical Engineering Department
Kansas State University, Manhattan, KS 66506

R. J. Fogler and G. R. Elliott
Sandia Laboratories, Org. 1738
Albuquerque, NM 87185

Abstract

This paper addresses some adaptive signal processing aspects of point-sensor based intrusion-detection systems. The sensors considered are geophones and microphones. Several examples are given of systems which can process signals from these sensors to materially reduce the affects of noise and yet reliably detect the presence of the walking intruder. The problem of acoustically-generated noise is discussed in some detail. Some aspects of implementation in field hardware are also included.

Introduction

Velocity geophones are transducers which produce an output signal voltage proportional to the velocity of ground motions. These ground motions can be caused by a person walking (intruder) or by vehicular traffic, portable power generators, wind gusts, trains, tractors, and so on (noise sources). Geophone signals produced by seismic waves caused by the motion of the sources on the ground will be referred to as type 1 signals. Acoustical noise sources such as overhead aircraft also produce seismic waves in an indirect manner, as depicted in Fig. 1. We observe that the process of sound waves impinging on the soil in proximity of the geophone results in seismic waves to which the geophone responds. Geophone signals produced by this acoustic coupling will be referred to as type 2 signals.

Studies related to the geophone response to footsteps of a person have been reported in Ref. 1. At distances under 2 metres, strong signal components have been found well below 20 Hz with weaker components in the 50 to 100 Hz range. At distances greater than 2 metres, the components in the 50 to 100 Hz range become dominant. These spectral components are largely independent of site conditions. Since no significant footstep energy exists beyond 100 Hz, our investigations have been restricted to signals that are band-limited to 100 Hz.

*This work was sponsored and funded by the Base Installation Security Systems Program Office, Electronic Systems Division of the Air Force Systems Command, Hanscom Air force Base, MA 01731.

A sample geophone signal for a walking intruder is depicted in Fig. 2. The intruder is walking in a circle of constant radius with the geophone at its center. A time domain plot of the resultant geophone signal is shown in Fig. 2a. The power density spectrum averaged over several footsteps is shown in Fig. 2b.

The intruder signal is not always as easily discernible—as depicted in Fig. 3. The walking intruder signals in Figs. 2 and 3a are the same magnitude but the latter is buried in type 2 noise generated by a helicopter hovering near the geophone. From Fig. 3a, it is apparent that in the interest of reliable intruder detection in the presence of such noise, some form of filtering to reduce noise must be done prior to using a detection algorithm. Furthermore, since soil characteristics can vary depending upon location and climatic conditions, spectral properties of noise signals may not be stationary.* As such, we will consider solutions which involve adaptive systems.

Algorithms

We consider two systems, the first of which is shown in Fig. 4, and is referred to as System 1. It consists of an adaptive digital predictor (ADP) and an adaptive threshold detector (ATD).

The ADP in System 1 can be implemented as a transversal filter or a lattice structure. We consider a symmetrical lattice implementation which has been effective in a recent intrusion-

detection algorithm^{2,3}. The lattice coefficients (weights) are updated by an algorithm proposed by Griffiths, details related to which are available elsewhere^{4,5}.

The ATD stage in Fig. 4 computes a ratio, $R(n)$,

of two signal energies $e^2(n)$ and $e^2(n-D)$ at time n ; see Fig. 5. The parameters N_1 and N_2

denote the number of samples over which the

energies $e^2(n)$ and $e^2(n-D)$ are computed, while D is a delay parameter. The quantity

$e^2(n)$ represents the intruder energy if one is

present, while $e^2(n-D)$ is a measure of the noise energy D samples earlier. The value of N_1 depends on the typical duration of an

intruder footstep. At a sampling frequency of 256 Hz, the value $N_1=24$ samples has been found

to be satisfactory. Corresponding values for N_2 and D that have worked well are 48 and 32

samples, respectively⁶. The value of N_2 is

chosen so that a reasonably smooth estimate of the ambient noise is obtained.

The role of the ADP in System 1 is to decorrelate the input noise. As such, it reduces nuisance alarms, and at the same time, improves the signal-to-noise ratio if a footstep is present. Footsteps by themselves produce broad-band transient signals which pass through the ADP with little change.

An assortment of data files were processed using System 1 with 8 stages for the ADP, and the related sampling frequency was 256 Hz. The convergence constant (α in Ref. 5) was 0.02, while the smoothing parameter (β in Ref. 5) was 0.98. Parameter values for the detector part of System 1 were as follows: $N_1=24$, $N_2=48$, $D=32$,

$\epsilon=2^{-17}$ and $\theta_1=4$. Noise sources included

helicopters, turboprops, jet aircraft, idling vehicles, and wind gusts. An example of the results so obtained is displayed in Figs. 3b and 6. Eight seconds of a 16-second data file are shown. We note that the ADP output is significantly reduced compared to the input. This is because the ADP in System 1 removes correlated noise present in the geophone output prior to detection. To illustrate this important property, we present the input-output power density spectra shown in Fig. 7. These correspond to the ADP input and output shown in Fig. 6, and were computed by taking the logarithm (to the base 10) of each spectrum value. It is apparent that the ADP output spectrum is relatively "white" (flat) compared to the corresponding input spectrum.

The results obtained led to the conclusion that System 1 is a promising candidate for intrusion-detection applications where type 2 noise (e.g., due to aircraft) is moderate. An array of geophones could be deployed in linear fashion to provide perimeter protection where the output of each geophone is processed by System 1.

Improvement Considerations.

The performance of System 1 may not be satisfactory in environments where type 2 noise is high, when one seeks a high probability of detection (POD) and a low nuisance alarm (NA) rate.

The key to arriving at a system whose perform-

ance is better than that of System 1 is to introduce a second transducer, namely a microphone. This is because a study using crosscorrelation

and coherence^{7,8,9} methods has shown that type 2 sources produce geophone and microphone signals that are correlated. As such, adaptive noise

cancellation techniques¹⁰ may be employed as indicated in System 2; see Fig. 8. This system differs from System 1 (Fig. 4) in four respects:

- (a) It has an ANC stage.
- (b) It has an alarm qualifier (ALQUAL) stage.
- (c) The product of the ANC and ADP outputs becomes the input to the ATD, instead of just the ADP output.
- (d) An alarm condition is determined by the outputs of ALQUAL and ATD via a logic stage.

In the following, the above differences are discerned on an individual basis.

- (a) The ANC decorrelates type 2 noise caused by overhead aircraft, thereby enhancing footsteps buried in noise (see Fig. 9). The ANC output is also used by a stage called ALQUAL (Alarm Qualifier), which will be discussed later. We use the nonsymmetrical lattice implementation for the ANC proposed

by Griffiths⁵. It is described by equations 13a and 13b in Ref. 5. Eight lattice schemes are used with a convergence parameter (α in Ref. 5) of 0.02, and a smoothing parameter (β in Ref. 5) of 0.98.

- (b) ALQUAL eliminates nuisance alarms that may be caused by overhead aircraft. It consists of a short-term correlator (STC) and a threshold, θ_2 ; see Fig. 8. The STC out-

puts a measure of the short-term correlation between the ANC and ADP outputs. Since the ANC and ADP both remove correlated noise present in their inputs, it is reasonable to expect that the short-term correlation will be higher for intruders' footsteps as compared to aircraft noise. To illustrate, the STC output corresponding to an intruder in helicopter noise is shown in Fig. 10. The related ANC and ADP outputs are also plotted. It is observed that footsteps produce a larger output than helicopter noise. Thus, if the STC outputs exceed a specified threshold θ_2 , then ALQUAL outputs a 1.

Otherwise, its output is 0. Several aircraft noise files were examined to obtain a value for θ_2 which eliminates nuisance

alarms due to overhead aircraft, and yet qualifies most intruder footsteps.

The STC stage of ALQUAL first computes the following function $C(n,m)$, where n is the time index, and m is a shift (lag) index

$$C(n,m) = \frac{1}{S_x(n-D)} C(n-1,m) + \frac{(1-\beta_1)x(n)y(n+m)}{S_x(n-D)S_y(n-D)} \quad (1)$$

where

$x(n)$ and $y(n)$ denote the ANC and ADP outputs, respectively; β_1 is a smoothing parameter;

D is a delay parameter; S_x and S_y are

measures of signal strengths in the ANC and ADP outputs, and are estimated recursively via the relations

$$S_x(n) = \beta_2 S_x(n-1) + (1-\beta_2) x(n)$$

and

$$S_y(n) = \beta_2 S_y(n-1) + (1-\beta_2) y(n)$$

where β_2 is again a smoothing parameter.

Equation (1) represents the process of shifting the sequence $y(n)$ with respect to $x(n)$ by m time steps, and then averaging the related products. A total of $(2M+1)$ first-order computations are involved at each time step n , since the shift parameter m takes the values $-M, -M+1, \dots, 0, 1, \dots, M$. Next, the STC computation process picks the peak value of $C(n,m)$ for each n . Hence, the final output of the STC stage is a single number $C^*(n,m)$ for each n :

$$C^*(n,m) = \max C(n,m) \quad (2)$$

The parameter values associated $C(n,m)$ in Eq. (1) are as follows:

$$\beta_1 = 0.5, \beta_2 = 0.98, D=41, \text{ and } M=20.$$

Again, the "best" value for the threshold θ_2 was found to be 15.5, based on various

data sets that were processed.

- (c) The process of using the product of the ANC and ADP outputs as the input to the ATD helps to enhance the ATD output in the presence of strong type 2 noise. This aspect is illustrated in Fig. 11, from which it is apparent that the ratio output $R(n)$ obtained via System 2 is much larger than that obtained via System 1. Hence, the chances of detecting footsteps under poor SNR conditions are improved.

The ATD parameters we use for System 2 are:

$$N_1=24, N_2=48, \theta_1=4, \text{ and } \epsilon=2^{-26}.$$

- (d) The logic stage is basically an AND function. Prior to computing the AND function, the outputs of ALQUAL and ATD are windowed; i.e., the outputs of ALQUAL and ATD are 1, if there is at least one 1 in a string of zeros and ones in a specified window width. Thus, if the windowed outputs of ALQUAL and ATD are both 1, the output of the logic stage is 1,

indicating an intrusion.

Some Experimental Results

The performances of System 1 and System 2 were evaluated using an assortment of field-acquired data. Two example plots (8 seconds each) are displayed in Fig. 12 and Fig. 13, respectively. These results demonstrate the superior performance of System 2 over System 1 under strong noise conditions. In these figures, "+" and N/A denote intruder footsteps and nuisance alarms, respectively.

The value of θ_1 used for both systems was 4.

The value of θ_1 for system 3 must be set so

that no nuisance alarms are obtained with type 1 noise signals—i.e., signals due to sources such as wind gusts, vehicular traffic, etc. This is because such noise sources could yield STC outputs that exceed the related threshold θ . For the results reported here, $\theta_1=4$ was found to

be adequate.

Hardware Considerations

The proposed algorithms could be implemented using special purpose digital hardware, charge transfer device technology, or microprocessors. We restrict our attention to microprocessor-based implementation in this discussion. We do not mean to imply that this is the most judicious approach. It merely serves to indicate the degree of complexity associated with these algorithms.

Any digital implementation must necessarily involve binary number representations of algorithm variables and parameters. Because the numerical range of these quantities can greatly exceed the computational range of fixed-point numerical representations in 16-bit binary or smaller word sites, a floating-point representation is needed.

In floating-point processes, numbers are represented in normalized magnitude sign-exponent formats. Such number representations are not typically supported in small microcomputer hardware and thus entail a considerable amount of

software overhead. In recent literature^{11,12}, a number representation, termed block floating-point (BFP), is described wherein the exponent is implied and the numeric range and precision of parameters are variable within a given process and can be optimized at each computational stage. Since the exponent is not computed at each arithmetic operation, there is very little software overhead. This results in computation speeds approaching those of fixed-point processes while providing a much greater computational range. In the discussion that follows, it is assumed that all arithmetic operations are performed in BFP format.

We have restricted our attention to microprocessors fabricated in either I^2L or CMOS technologies because of noise immunity and power consumption considerations.

We have used the Texas Instruments SBP9900 16-bit I^2L microprocessor in another signal processing application² and have begun evaluation

of the new SBP9989 microprocessor which possesses some additional software instructions including signed 16x16 bit multiply and divide, an important consideration in two's complement BFP real-time implementation. We have followed the development of a second microprocessor, the National Semiconductor NSC800 8-bit CMOS microprocessor, with interest. Although it does not possess a hardware multiply/divide, it has a powerful instruction set and has excellent potential if augmented by external signed multiply/divide hardware. The NSC800 is both less expensive than the SBP9989 and is simpler to power. The NSC800 needs only single supply voltage but the SBP9989 requires both a supply voltage for memories and miscellaneous logic as well as a constant injector current at low voltage. There are also other microprocessors which undoubtedly function as well as these two. Their inclusion here stems from our familiarity with them and a desire to discuss implementation with active hardware. Although the execution speed performance figures given in the following discussion relate specifically to the SBP9989 microprocessor, a NSC800 microprocessor augmented by external signed multiply/divide hardware may approach the SBP9989 execution speed.

A SBP9989 operating at a clock frequency of 6 MHz with no memory wait states can implement System 1 which consists of an adaptive digital predictor (ADP) and adaptive threshold detector (ATD) at a sampling rate of 400 sps (samples per second). Thus, the speed of the SBP9989 is more than adequate for processing the 100 Hz bandwidth signals in this application.

System 2 is much more complex, and therefore, more computationally burdensome. The major computations are found in three of the System 2 functional blocks—the noise canceller, predictor, and correlator. The lattice noise canceller involves approximately 1.5 times more computation than the predictor. Thus, a SBP9989 operating at 6 MHz could execute the noise canceller at approximately 266 samples per second. The correlator, assuming that the smoothing parameter β_1 is an integral power of 2 allowing

multiplication by arithmetic shift, involves approximately 3.25 times the computation necessary to implement the lattice predictor. A single SBP9989 could implement the correlator at only 120 sps. By distributing the computation of the predictor, noise-canceller, and correlator in four SBP9989 microprocessors, System 3 could be implemented in real-time.

It is apparent from the above discussion that System 1 could be implemented in a single microprocessor without difficulty but that implementation of System 2 would not be as straightforward. Serious consideration should be given to other approaches for System 2 such as special purpose digital hardware or charge transfer device technology.

References

1. D. H. Cress, Seismic Sensor Applications: Advantages and Limitations of Seismic Phenomena and Selected Transducers, Report of the Third Sensor Technology Symposium, 25-27 Sept. 1979, Vol. 1, Nov. 1980 pp 71-96.
2. N. Ahmed and G. R. Elliott, "On Using an Adaptive Algorithm for Intrusion-Detection," Proceedings of the 14th Asilomar Conference, Nov. 17-19, 1981, Asilomar, CA, pp. 407-410.
3. N. Ahmed and G. R. Elliott, "An Intrusion-Detection System for Perimeter Sensors," Proceedings of the 24th Midwest Symposium, June 29-30, 1981, Albuquerque, NM.
4. L. J. Griffiths, "A Continuously-Adaptive Filter Implemented as a Lattice Structure," Proceedings of the Int. Conf. on Acoust., Speech, and Signal Processing (ICASSP), Hartford, Ct, May 1977, pp. 683-686.
5. L. J. Griffiths, "An Adaptive Lattice Structure for Noise-Cancelling Applications," Proceedings ICASSP, Tulsa, OK, April 1978, pp. 87-90.
6. J. P. Claassen and M. M. Patterson, "A Comparative Study of Adaptive Noise Cancellation Algorithms for Intrusion Detection Systems," Proceedings of the 24th Midwest Symposium, June 29-30, 1981, Albuquerque, NM.
7. J. S. Bendat and A. G. Piersol, Engineering Applications of Correlation and Spectral Analysis, Wiley & Sons, 1980, Chap. 3.
8. S. D. Stearns, "Applications of the Coherence Function in Comparing Seismometers," Sandia Laboratories Tech. Report, SAND79-1633, Dec. 1979.
9. K. Scarbrough, N. Ahmed, and G. C. Carter, "On the Simulation of a Class of Time Delay Estimation Algorithms," IEEE Trans. ASSP, Vol. 19, No. 3, June 1981, pp. 534-540.
10. B. Widrow, et al, "Adaptive Noise Cancelling: Principles and Applications," Proc. IEEE, Vol. 63, No. 12, Dec. 1975, pp. 1692-1716.
11. A. V. Oppenheim and R. W. Schaffer, Digital Signal Processing, Englewood Cliffs, NJ; Prentice-Hall, Inc. 1975, pp. 455-456.
12. J. E. Simpson, A Block Floating-Point Notation for Signal Processes, SAND79-1823, March 1981.

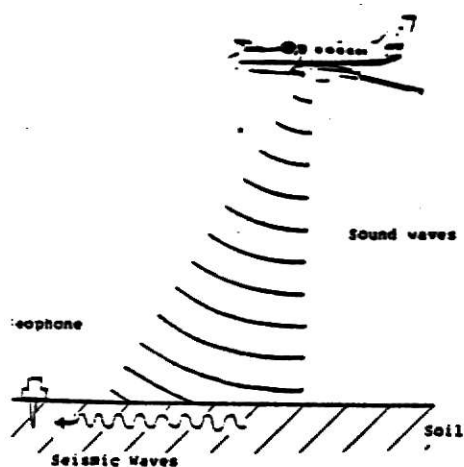


Figure 1. Response of a Geophone to an Acoustical Source

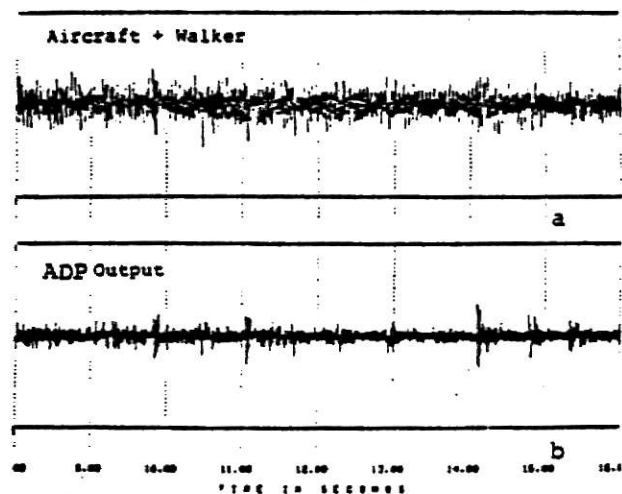


Figure 3. Adaptive Noise Canceller Performance on Aircraft Noise

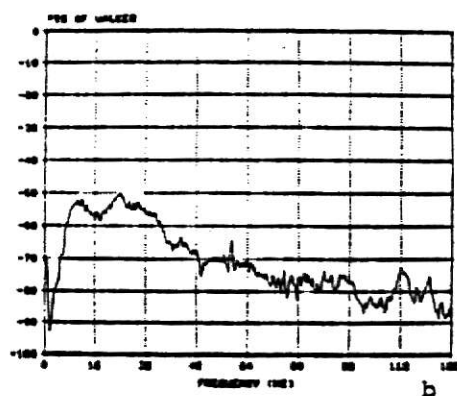
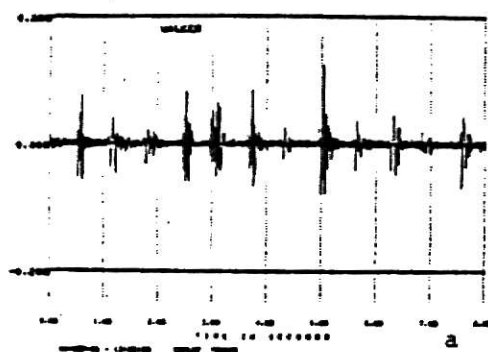


Figure 2. Geophone Response to a Walker Intruder

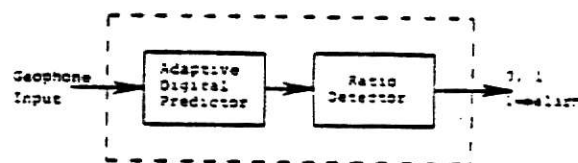


Figure 4. System 1

$$e^2(n) = \sum_{j=n}^{n-N_1} s^2(j)$$

$$e^2(n-D) = \sum_{j=n-D}^{n-D-N_2} s^2(j)$$

$$R(n) = \frac{e^2(n)}{\max[e^2(n-D), c]}$$

where c is a predetermined constant.

Figure 5. Ratio Computation

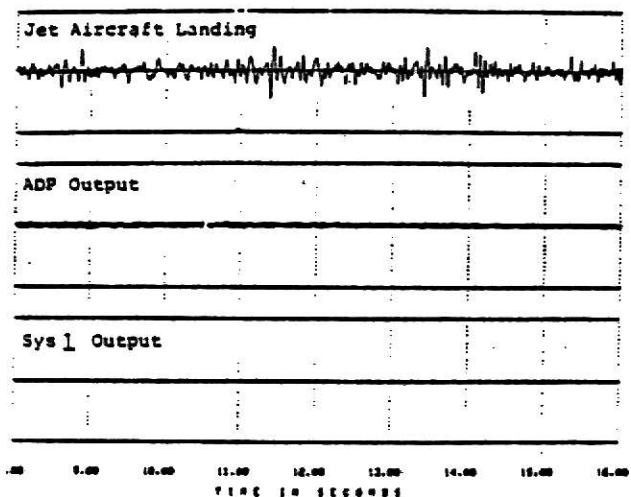


Figure 6. System 1 Performance with Aircraft Noise

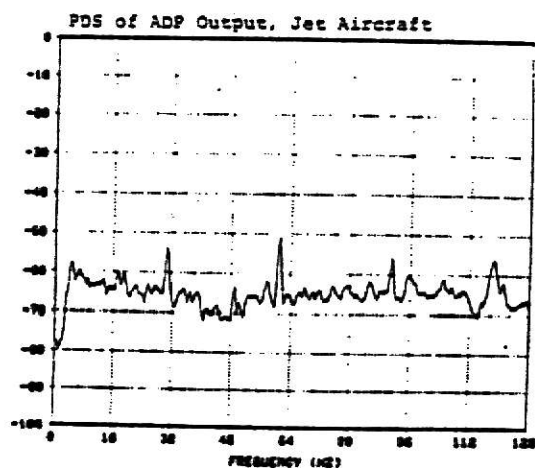
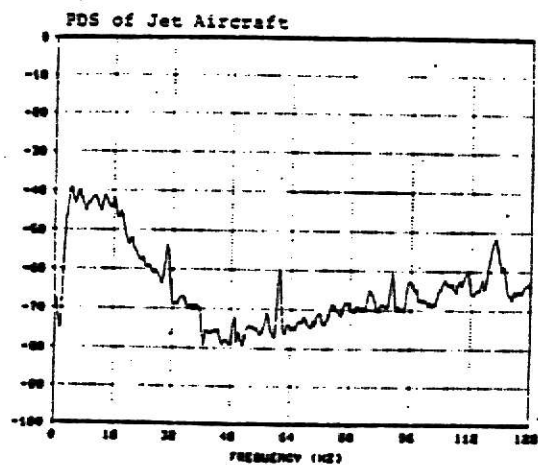


Figure 7. System 1 Frequency Characteristics

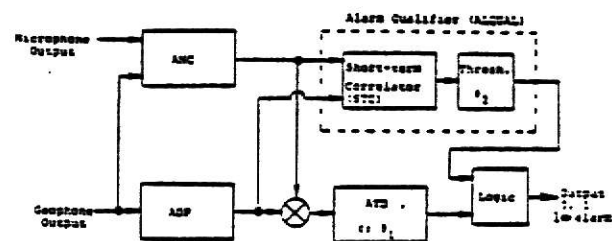


Figure 8. System 2

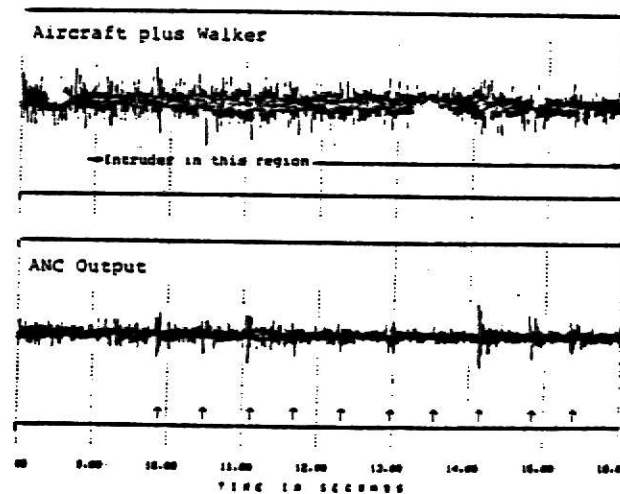


Figure 9. System 2 Performance with Heavy Acoustical Noise

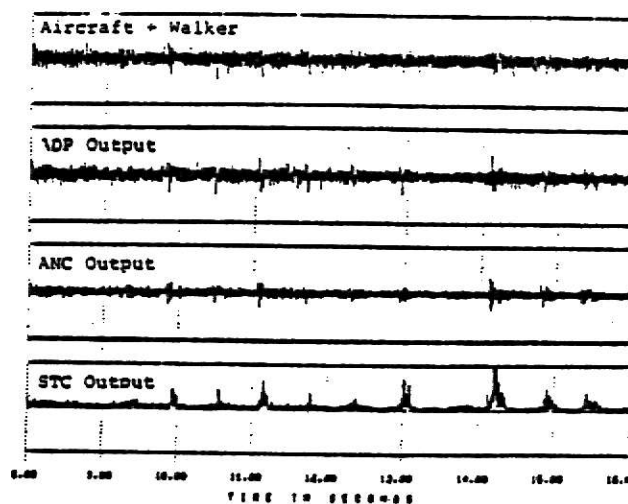


Figure 10. Output of Various Stages of System 2

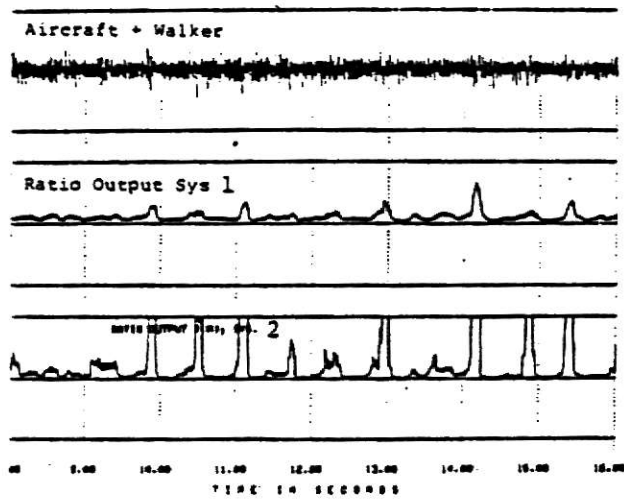


Figure 11. Comparison of System 1
and System 2

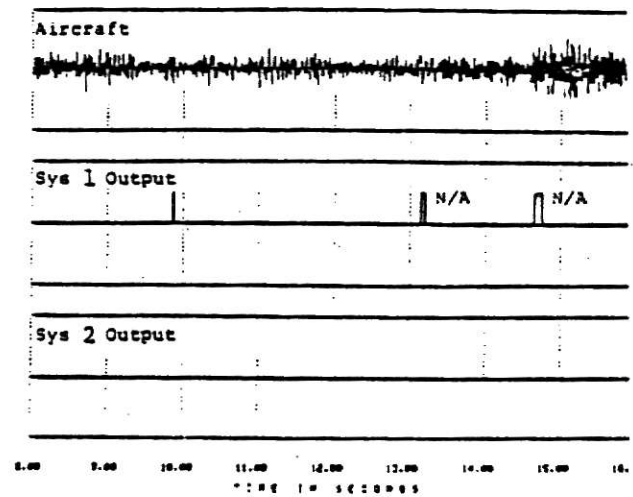


Figure 13. Improved Nuisance Alarm
Rejection of System 2

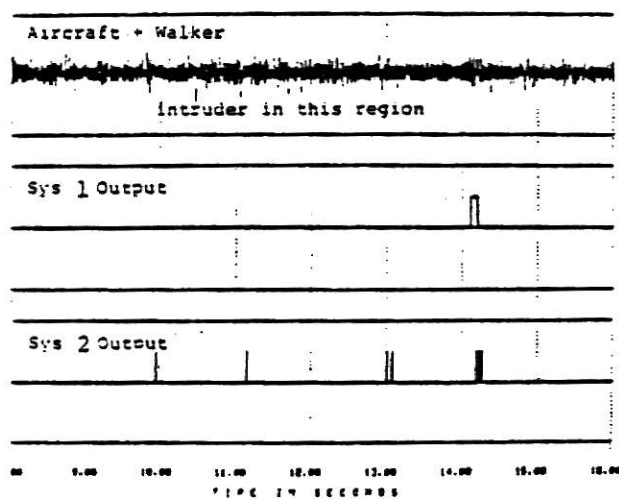


Figure 12. Improved Detection
of System 2

APPENDIX II

Computer programs.


```

      INTEGER NEWPTR,PTR,MIDPTR,OLDPTR,TEMPPTR,VNWPTR,VOLPTR
      INTEGER MON,DAY,YEAR,HRS,MIN,SEC
      LOGICAL DENPRO,STOR,ZECROSS,PCPNT,NORM

C
C
1  FORMAT(/'--- SHORT TERM CORRELATOR --- ')
C
2  FORMAT('BETA1(0.0-1.0) (FOR CORRELATION)(F)? = ',Z)
C
3  FORMAT('BETA2(0.0-1.0) (FOR NORMALIZATION)(F)? = ',Z)
C
C
4  FORMAT('DENOMINATOR EPSILON (F) ? = ',Z)
C
5  FORMAT(/' ---PROGRAM TERMINATION --- ')
C
6  FORMAT('DENOMINATOR (1=PROD,0=SUM)(I) ? = ',Z)
C
7  FORMAT(F15.12)
C
8  FORMAT(I5)
C
9  FORMAT('MON ',I2,' DAY ',I2,' YEAR ',I2,' ')
C
12  FORMAT(' HRS ',I2,'2X 'MIN ', I2,'2X 'SEC ', I2,' ')
C
13  FORMAT(' EPSILON(MIN ABS VALUE FOR ZECROSS) : ',Z)
C
14  FORMAT(' DO YOU WISH TO STORE CORRELATION(1=YES,0=NO)? : ',Z)
C
15  FORMAT(' DO YOU WISH TO COMPUTE # OF ZEROCROSSINGS(1=YES,0=NO)? : ',Z)
C
16  FORMAT(' DO YOU WISH TO STORE CORR PEAK POINT(1=YES,0=NO)? : ',Z)
C
C
17  FORMAT(' DO YOU WISH TO NORMALISE BY SIGNAL POWER(1=YES,0=NO)? : ',Z)
C
      CALL FGDAY(MON,DAY,YEAR)
      WRITE(10,9) MON,DAY,YEAR
      WRITE(10,1)
500  CONTINUE
      CALL FSTIME(0,0,0)
C
C  (QUERY AND GET RESPONSE)
C
C
      WRITE(10,14)
      READ(11,8) I
      IF(I.NE.0) STOR = .TRUE.
      IF(I.EQ.0) STOR = .FALSE.
      IF(STOR) ACCEPT ' DUMP AT ITER'S MULTIPLE OF : ',IDUMP
C
      WRITE(10,15)
      READ(11,8) I
      IF(I.NE.0) ZECROSS = .TRUE.
      IF(I.EQ.0) ZECROSS = .FALSE.
C

```



```

        WRITE(10,16)
        READ(11,8) I
        IF(I.NE.0) PCPNT = .TRUE.
        IF(I.EQ.0) PCPNT = .FALSE.
C
C
        WRITE(10,17)
        READ(11,8) I
        IF(I.NE.0) NORM = .TRUE.
        IF(I.EQ.0) NORM = .FALSE.
C
C
C
        OBTAIN PARAMETERS
C
100  CONTINUE
        WRITE(10,2)
        READ(11,7) BETA1
        BETA11 = 1.0 - BETA1
        IF(BETA1.LT.0.0.OR.BETA1.GT.1) GO TO 100
C
110  CONTINUE
        IF(.NOT.NORM) GO TO 120
        IF(NORM) WRITE(10,3)
        READ(11,7) BETA2
        BETA21 = 1.0 - BETA2
        IF(BETA2.LT.0.0.OR.BETA2.GT.1) GO TO 110
C
120  CONTINUE
        ACCEPT ' NUMBER OF SHIFTS M (1 - 64)? = ',M
        M1 = M + 1
        MS = 2*M
        MS1 = MS + 1
        IF(M.LT.1.OR.M.GT.64) GO TO 120
C
130  CONTINUE
        IF(.NOT.NORM) GO TO 140
        IF(NORM) ACCEPT ' VARIANCE ESTIMATE DELAY(1-129)? = ',ND
        ND1 = ND + M + 1
        IF(ND.LT.0.0.OR.ND.GT.129) GO TO 130
C
140  CONTINUE
        IF(.NOT.NORM) GO TO 150
        IF(NORM) WRITE(10,4)
        READ(11,7) EPSILON
        IF(EPSILON.LE.0.0.OR.EPSILON.GT.1.0) GO TO 140
C
150  CONTINUE
        IF(.NOT.NORM) GO TO 160
        IF(NORM) WRITE(10,6)
        READ(11,8) I
        IF(I.NE.0) DENPRO=.TRUE.
        IF(I.EQ.0) DENPRO=.FALSE.
C
160  CONTINUE
        IF(.NOT.ZECROSS) GO TO 170
        IF(ZECROSS) WRITE(10,13)

```

```

      READ(11,7) EPS
      (INITIALIZATION)
170  CONTINUE
C
C
      DO 210 I=1,MS1
      XD(I)=0.0
      YD(I)=0.0
      C(I)=0.0
210  CONTINUE
C
      IF(.NOT.NORM) GO TO 230
      DO 220 I=1,ND1
C
C      (INITIALISE VALUES)
C
      VXD(I)=1.0
      VYD(I)=1.0
220  CONTINUE
C
      PX = 1.0
      PY = 1.0
C
230  CONTINUE
C
      SET UP PARAMETERS
C
      NB = MS1 * 4
      NBYTE = 256 * 4
C
      PTR = 0
      NEWPTR=MS1
      MIDPTR=M1
      OLDPTR=1
      IF(.NOT.NORM) DENOM = 1.0
      IF(STOR) NDUMP = IDUMP
      IFDUMP = 0.0
C
      VNEWPTR=ND1
      VOLPTR=1
C
C      I/O SET UP
C
      CALL OPENR(0,' INPUT FILE - X   : ',NBYTE,F2)
C
      CALL OPENR(1,' INPUT FILE - Y   : ',NBYTE,SIZE)
C
      IF(PCPNT) CALL OPENW(3,' CORR. PEAK POINT FILE : ',4,SIZE)
C
      IF(STOR) CALL OPENW(4,' SMOOTH CORR. FILE      : ',NB,SIZE1)
C
      IF(ZECROSS) CALL OPENW(5,' ZERO CROSSINGS FILE   : ',4,SIZE2)
C
      FIND OUT HOW MANY POINTS IN FILE
C
C
C

```

```

      XP = F2/4.0
      NP = IFIX(XP)
      ITER = NP/256
      TYPE ' * OF POINTS IN FILE : ',NP
      TYPE ' * ITERATION          : ',ITER

C
C      MAIN LOOP
C
C      DO 410 JJ=1,ITER
C
C      READ 256 POINTS OF DATA
C
C      CALL READR(0,JJ,X,1,IER)
C      CALL READR(1,JJ,Y,1,IER)
C
C      (ALGORITHM EXECUTION)
C
      DO 420 I = 1,256
      IFDUMP = IFDUMP + 1
      XD(NEWPTR)=X(I)
      YD(NEWPTR)=Y(I)
      PX = BETA2*PX + BETA21*ABS(XD(NEWPTR))
      PY = BETA2*PY + BETA21*ABS(YD(NEWPTR))
      IF(.NOT.NORM) GO TO 430
      VXD(VNUPTR) = PX
      VYD(VNUPTR) = PY
      VXOLD=VXD(VOLPTR)
      VYOLD=VYD(VOLPTR)
      IF(DENPRO) DENOM=VXOLD*VYOLD
      IF(.NOT.DENPRO) DENOM=(VXOLD+VYOLD)/2
      IF(VXOLD.LT.EPSILON.OR.VYOLD.LT.EPSILON) DENOM=1.0
C
C      430 CONTINUE
      TEMPTR = OLDPTR
      CPEAK = 0.0
      IPEAK = 1
      IF(ZECROSS) ZERO = 0.0
C
C      FILTER BANK
C
      DO 440 J =1,MS1
      C(J)=BETA1*C(J) + BETA11*XD(MIDPTR)*YD(TEMPTR)/DENOM
      CNORM = C(J)
      TEMPTR=TEMPTR+1
      IF(TEMPTR.GT.MS1) TEMPTR=1
      IF(CNORM.LT.CPEAK) GOTO 440
      IPEAK=J
      CPEAK=CNORM
C      440 CONTINUE
C
      IF(.NOT.STOR) GO TO 450
      IF(IFDUMP.NE.NDUMP) GO TO 450
      PTR = PTR + 1
      NDUMP = IFDUMP + IDUMP
      CALL WRITR(4,PTR,C,1,IER)

```

```

C
450  CONTINUE
C
C      COUNT # OF ZEROCROSSINGS IN CORRELATION FUNCTION
C
      IF(.NOT.ZECROSS) GO TO 460
      DO 470 J = 1,MS
      IF(ABS(C(J)),LT.EPS) GO TO 470
      P = C(J)*C(J+1)
      IF(P.LT.0.0) ZERO = ZERO + 1
470  CONTINUE
      WRITE BINARY(5) ZERO
C
460  CONTINUE
C      (COUNTERS FOR VARIOUS VARIABLES)
C
      NEWPTR=NEWPTR+1
      IF(NEWPTR.GT.MS1) NEWPTR=1
      MIDPTR=MIDPTR+1
      IF(MIDPTR.GT.MS1) MIDPTR=1
      OLDPTR=OLDPTR+1
      IF(OLDPTR.GT.MS1) OLDPTR=1
      UNWPTR=UNWPTR+1
      IF(UNWPTR.GT.ND1) UNWPTR=1
      VOLPTR=VOLPTR+1
      IF(VOLPTR.GT.ND1) VOLPTR = 1
C
      IF(.NOT.PCPNT) GO TO 480
C      ESTIMATE THE DELAY
C
      ALARM=FLOAT(IPEAK-M1)
      IF(ALARM.LT.0) ALARM = -1
      IF(ALARM.GT.0) ALARM = 1
      WRITE BINARY(3) ALARM
C
480  CONTINUE
C
420  CONTINUE
C
410  CONTINUE
C
C      PROGRAM TERMINATION
C
490  CONTINUE
      CALL RESET
      WRITE(10,5)
C
      CALL FGTIME(HRS,MIN,SEC)
C
      TYPE ' ----- '
C
      TYPE ' EXECUTION TIME '
      WRITE(10,12) HRS,MIN,SEC
C
      TYPE ' ----- '
      CALL QUERY('<7><12> RE EXECUTE (Y/N) : ',IQ)

```

```
IF(IQ.EQ.1) GOTO 500
```

```
C
```

```
STOP
```

```
END
```

```

C*****
C
C      SMOOTH.FR
C
C      THIS PROGRAM IMPLEMENTS THE EQUATION BELOW.
C
C       $SUM = BETA * SUM + (1.0 - BETA) * X(J)$ 
C
C      VIJAYENDRA                      9/24/81
C
C*****
C
C      DIMENSION X(128),Y(128)
C      REAL SUM,BETA
C
C      ACCEPT ' BETA FOR SHOOTING : ',BETA
C
C      NB = 128 * 4
C      CALL OPENR(0,'INPUT FILE NAME : ',NB,F1)
C      CALL OPENW(1,'OUTPUT FILE NAME : ',NB,F2)
C
C      I = 0
C      SUM = 0.0
100  CONTINUE
C      I = I + 1
C      CALL READR(0,I,X,1,IER)
C      IF(IER.EQ.9) GO TO 150
C      DO 120 J = 1,128
C      Y(J) = SUM
C       $SUM = BETA * SUM + (1.0 - BETA) * X(J)$ 
120  CONTINUE
C      CALL WRITR(1,I,Y,1,IER)
C      GO TO 100
C
C
150  CONTINUE
C      CALL RESET
C      STOP
C      END

```

REFERENCES

- [1] "Random Data: Analysis and measurement procedures", Bendat and Piersol, Wiley-Inter-science, Chapter 9.
- [2] "The Use of Fast Fourier Transform for the Estimation of Power Spectra: A method based on time averaging over short, modified periodograms", P.D. Welch, I.E.E.E. Transactions on Audio and Electro Acoustics, AU-15, No. 2, June 1967.
- [3] "Probability, Random Variables and Random Signal Principals", P.Z. Peebles, Jr., McGraw Hill Book Co., 1980.
- [4] "High Resolution Differential Time of Arrival and Differential Doppler Estimation", D.L. Johnstone and O.L. Frost, ARGO Systems, January 1977.
- [5] "Random Signals and Noise", Davenport and Root, McGraw Hill Book Co., 1958, pp 168-169.
- [6] "Point Sensor Adaptive Signal Processing for Intruder Detection", N. Ahmed, S. Vijayendra, R.J. Fogler and G.R. Elliott", SAND 81-2536C, Report prepared by Sandia Laboratories, N.M.

ACKNOWLEDGEMENTS

I wish to thank my major professor Dr. Nasir Ahmed for his consistent guidance and encouragement throughout my studies. I also thank Dr. D.R. Hummels and Dr. L.E. Fuller for serving on my committee. Thanks are also due to Mr. R.J. Fogler and Mr. G.R. Elliott of Sandia Laboratories. I wish to thank the Department of Electrical Engineering, Kansas State University and Sandia National Laboratories, New Mexico for thier financial support. Thanks are also due to Mrs. Estella Creel for her splendid typing of this report.

Finally I would like to thank my family for their support and inspiration.

A STUDY OF A SHORT TERM CORRELATION ALGORITHM

by

Siravara Vijayendra
B.E., Bangalore University, India 1979

An abstract of a MASTER'S report
submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

ELECTRICAL ENGINEERING

COLLEGE OF ENGINEERING

KANSAS STATE UNIVERSITY, MANHATTAN, KANSAS

1982

ABSTRACT

The purpose of this report is to study a Short-Term Correlator (STCOR) algorithm and two possible applications, that are related to point sensor intrusion-detection schemes. The STCOR can be implemented as a bank of infinite impulse response (IIR) filters, each of which estimates the value of the desired crosscorrelation function for a particular lag value. Experimental results which help assess the performance of this algorithm are also included.