Estimation of arbitrary time delays of multichannel synthetic data

J. Douglas Callison, Terry E. Riemer, and Russell E. Trahan

Citation: The Journal of the Acoustical Society of America **81**, 1000 (1987); doi: 10.1121/1.394671
View online: https://doi.org/10.1121/1.394671
View Table of Contents: https://asa.scitation.org/toc/jas/81/4
Published by the Acoustical Society of America

Estimation of arbitrary time delays of multichannel synthetic data

J. Douglas Callison, Terry E. Riemer, and Russell E. Trahan, Jr.
Department of Electrical Engineering, University of New Orleans, New Orleans, Louisiana 70148

(Received 13 January 1986; accepted for publication 31 October 1986)

A new method for computing arbitrary time delays between multichannel data is introduced based on the idea of using the ordinary cross-correlation technique. It is shown that, by windowing the data record, events can be isolated and identified between channels. This new technique, called the window-correlation technique (WCT), is capable of determining arbitrary delays, very accurately, down to a signal-to-noise ratio of 0 dB. Finally, results of the WCT are compared to those of the ordinary cross correlation for Gaussian signals corrupted by additive Gaussian white noise.

PACS numbers: 43.60.Gk

INTRODUCTION

Over the past few years, there has been considerable interest in the area of time-delay estimation (TDE).[1-7] This subject became so popular that it brought about a special issue in the IEEE Transactions on Acoustics, Speech, and Signal Processing.[1] However, most of the work that has been published seems to deal mainly with specific types of unbiased estimators and how well they perform compared to the Cramer–Rao lower bound[8] for the variance.[1,2] Also, the articles that deal with simulation results, showing actual estimated delays, handle a uniform delay only. In other words, the entire received signal, which is considered one event, is delayed by some integral multiple of the sample period.[1,2,5,6] The algorithm presented in this article is capable of handling multichannel data, embedded in additive Gaussian white noise, that consists of arbitrary time delays between channels. In contrast with the classic uniform delay problem, the algorithm isolates specific events, which may be delayed by arbitrary amounts between channels, and calculates the delays as compared to the corresponding events in the reference channel.

When time-delay estimation is developed for radar purposes, it is essential to find a solution that can be implemented in real time. However, there are applications where computation time is not as important as accuracy of estimation and the actual calculation of delays. The two major areas that do not necessarily require real-time application are the seismic and audio processing fields. For example, it might be desirable to know the time differences between related events in multichannel audio data. Another example could be that of a seismic trace that consists of echoes describing the earth's layers. These echoes would not necessarily be delayed a uniform amount on the received trace. Therefore, a generalized cross-correlation method would most likely not work. The procedure discussed in this article was not designed for a real-time process. Nevertheless, it does seem to give extremely accurate results for data with varying delays and corrupted by a substantial amount of noise. However, there is a slight degradation of accuracy, as seems to be the case in all TDE techniques, as the signal-to-noise ratio (SNR) becomes very small.

This article is organized as follows. Section I reviews some of the basic concepts of cross-correlation techniques as applied to time-delay estimation. This is the basis of the algorithm characterized in this article. Section II describes the algorithm developed for the arbitrary TDE case, which uses the concept of the window-correlation technique (WCT). As is the case with any newly developed technique, a suitable means of comparison and presentation of results must be fulfilled. This is accomplished in Sec. III. Finally, conclusions and discussion of further work are covered in Sec. IV.

I. REVIEW OF CROSS-CORRELATION ESTIMATION

Although the following procedure can be used for multichannel data, the two-channel case will be developed for simplicity. Assume that the two-channel signals can be written in mathematical form as follows:

$$x(t) = y(t) + n_1(t) ,$$
$$h(t) = \alpha y(t-\tau) + n_2(t) , \tag{1}$$

where $x(t)$ is the signal in channel #1, $h(t)$ is the signal in channel #2, $y(t)$ is the source signal, and α is an attenuation factor. The contaminating noise sources, $n_1(t)$ and $n_2(t)$, are statistically independent of each other and the signal, zero-mean, and having Gaussian probability density functions, resulting in the well-known additive Gaussian white noise (AGWN) model. The variable τ is the uniform delay between channels #1 and #2 and is assumed to be an integral multiple τ_m of a sample period T. Since the correlation procedure is implemented on a computer and only a finite number of data points are available, Eqs. (1) can be written in terms of discrete values as follows:

$$x(kT) = y(kT) + n_1(kT) ,$$
$$h(kT) = \alpha y(kT - \tau_m T) + n_2(kT) , \tag{2}$$
$$0 \leqslant k \leqslant N-1 ,$$

where N is the total number of data points available and T is the sample period, which we normalize to unity.

Assuming that $y(k)$ is mutually independent of $n_1(k)$ and $n_2(k)$, the cross correlation between $x(k)$ and $h(k)$ is given by[6]

 0001-4966/87/041000-07$00.80

$$R_{xh}(\lambda) = E[x(k)h(k+\lambda)] = \alpha R_{yy}(\lambda - \tau_m), \quad (3)$$

where $R_{yy}(\lambda - \tau_m)$ is the autocorrelation of the source signal $y(k)$. It is well known[9] that a reasonable estimate for the cross correlation can be given as

$$R_{xh}(\lambda) = \frac{1}{N} \sum_{k=0}^{N-|\lambda|-1} x(k)h(k+\lambda). \quad (4)$$

The expected value of $R_{xh}(\lambda)$ can easily be found to be

$$E[R_{xh}(\lambda)] = \frac{N-\lambda}{N} R_{xh}(\lambda) = \left(1 - \frac{\lambda}{N}\right) R_{xh}(\lambda), \quad 0 \leqslant \lambda < N. \quad (5)$$

Therefore, as long as N is much greater than the shift λ, the estimator is asymptotically unbiased. It can also be shown that the variance of $R_{xh}(\lambda)$ is inversely proportional to N.[9]

Now, by locating the largest peak of $R_{xh}(\lambda)$, the uniform delay τ_m can be estimated, since the autocorrelation is a maximum at zero shift. This is known as the ordinary cross-correlation method.

Since the processing is performed on a computer, an FFT can be used to decrease computation time. By using Fourier transform properties, Eq. (4) can be restated for real functions as follows:

$$R_{xh}(\lambda) = (1/N) F^{-1}[X^*(f) H(f)], \quad (6)$$

where F^{-1} denotes the inverse Fourier transform, $X(f)$ and $H(f)$ denote the Fourier transforms of $x(\cdot)$ and $h(\cdot)$, respectively, and the $*$ indicates complex conjugate. Thus, from Eq. (6), the estimate of the cross-correlation function is obtained by taking the inverse transform of the product of the transforms of each data record.

The analysis above was developed for a uniform delay between the two channels. In Sec. II, certain modifications must be made before the ordinary cross-correlation method can be used for arbitrary time delays.

II. WINDOW-CORRELATION TECHNIQUE

The window-correlation technique uses the ordinary cross-correlation technique as its basis. Assume that the two-channel system now consists of two distinct events in which each event is delayed arbitrary amounts between channels:

$$x(k) = y_1(k) + y_2(k) + n_1(k),$$
$$h(k) = \alpha y_1(k - \tau_1) + \beta y_2(k - \tau_2) + n_2(k), \quad (7)$$

where $y_1(\cdot)$ and $y_2(\cdot)$ represent different events in the data record. The two delays are τ_1 and τ_2. Both channels are again corrupted with AGWN. If we were to use the ordinary cross-correlation technique on Eqs. (7), we would expect the estimated delay to be some combination of τ_1 and τ_2. In fact, the delay turns out to be the largest of τ_1 and τ_2 when the technique can estimate the delay. Therefore, the ordinary cross-correlation method cannot distinguish between the two delays. Thus, if we can somehow isolate the two separate events, we can estimate both delays independently. This leads to the use of the window-correlation technique.

The procedure is started by windowing a section of the reference channel with a suitable window function. The windowing procedure eliminates the dependence of correlation length on the amount of delay. The reason for this arises from the fact that the window isolates *segments of data*. A least-squares minimization is used to compare window segments, as will be discussed later. Therefore, the entire length of the event is not being correlated, but only segments of it. The windowed reference channel can now be represented by

$$x_w(k) = x(k) w(k), \quad (8)$$

where

$$w(k) = \begin{cases} 1, & 0 \leqslant k \leqslant M, \\ 0, & M < k < N-1. \end{cases} \quad (9)$$

In Eq. (9) the parameter M, which determines the window length, is chosen small enough to satisfy the test data requirements discussed later. In our test problem, the length of the window is chosen in order that two events are not overlapped in a specific channel. For any problem, the selection of M is certainly a critical factor since it is closely related to the amount of delay. The selection of an optimum window is currently being researched by examining the frequency content of the signals. For the purpose of presenting the WCT, the window in this article is assumed optimum.

Since $x_w(k)$ is now the desired signal, the autocorrelation of $x_w(k)$ can be calculated to yield the desired autocorrelation function $R_{xxw}(\lambda)$. Next, the second channel is windowed by $w(k)$, yielding $h_w(k)$. The ordinary cross-correlation technique is next used to get a second correlation function $R_{xhw}(\lambda)$. Now that we have a desired function and a second function, we have to use some technique to see how close these two functions are to each other.[10] The method chosen by the authors is a squared-error function given by

$$\epsilon^2 = \sum_{\lambda=0}^{N-1} [R_{xxw}(\lambda) - R_{xhw}(\lambda)]^2. \quad (10)$$

The value calculated by Eq. (10) is stored for future use. The next process is to slide the window over one data point on channel #2 while leaving the desired autocorrelation function untouched for the time being. Another ordinary cross correlation is computed between $x_w(k)$ and the new $h_w(k)$, followed by evaluating the new squared error. This process is continued until the desired length of channel #2 has been tested. When the process is completed, the algorithm has produced a set of squared errors,

$$z = (\epsilon_1^2, \epsilon_2^2, \ldots, \epsilon_p^2), \quad (11)$$

without changing the window position on the reference signal. From Eq. (11), the least-squared error (LSE) can be found by searching for the minimum value. The position of the window on the second channel, where the LSE occurs, is chosen as the estimated delay for the section of the reference channel which has been windowed in Eq. (8). In other words, for the windowed reference channel, a delay has been estimated for the section of the second channel that best approximates the first channel.

The algorithm is repeated by moving the window one data point down the reference channel data. The same procedure is used to estimate the delay for the new windowed segment.

At this point in the algorithm, we are left with a number of estimated delays d corresponding to specific segments of the reference channel. The new problem becomes one of sorting out these delays and making a decision on the number of segments we have with actual data and what their corresponding delays are. This decision-making process is accomplished by looking for trends in the set of estimated delays. If a certain number of successive delays are particularly close to each other, over several segments of data, the decision is made that those segments contain useful information and the delay is estimated as the average of the closely associated group of delays. Ideally, these delays would be equal and the average would simply be the computed delay. However, with decreasing SNR, these delays will probably not be equal, but the average should be extremely close to the actual delay. Since the noise used is uncorrelated, no decision will be made when the packets contain only noise, because each successive delay computed would not be expected to follow a fixed pattern.

The technique described above is stated in algorithmic form below. Algorithm 1 is used to compute the estimated autocorrelation and cross correlation of the desired signals. Also, this algorithm computes the set of estimated delays d for each window position of the reference channel. Algorithm 2 is called by algorithm 1. The set of delays d is passed from algorithm 1 and the output consists of a signal r that has been reconstructed using the second channel and set of delays d. Algorithm 3 is also called by algorithm 1. This algorithm is used to identify the different events in the two channels and compute their relative positions by estimating the delay between them. This is accomplished by first searching for a specified number of successive delays that are $\pm\epsilon$ apart. If the condition on consecutive values is met, the algorithm estimates the delay for a certain interval of channel 1 as being the average of these delays. However, there arises the case where one or two values may be encountered outside the $\pm\epsilon$ interval and then the values return back within the $\pm\epsilon$ range for an extended period of time. In this case, the points outside the interval are disregarded and the average is taken using only the values inside the $\pm\epsilon$ range. We now state algorithm 1.

Algorithm 1. Delay estimation

Data:

x = Signal in channel #1
y = Signal in channel #2
N = Number of data points
W = Window width (determined empirically to be optimum for $W = N/10$)
L = Distance window is to slide over channel #2 ($0 \leqslant L \leqslant N - W - 1$)
K = Initial position of window in reference channel ($0 \leqslant K \leqslant N - W - 1$)
r = Reconstructed signal (initialized to zero).

Steps:

Step 0: Set $m = K$, $j = K$.
Step 1: Window reference channel, starting at j, producing $x_w(k)$, $0 \leqslant k \leqslant N - W - 1$.
Step 2: Compute the estimated autocorrelation, $R_{xxw}(\lambda)$, of $x_w(k)$.
Step 3: Window second channel, starting at m, producing $h_w(k)$, $0 \leqslant k \leqslant N - W - 1$.
Step 4: Compute the estimated cross correlation, $R_{xhw}(\lambda)$, between $x_w(k)$ and $h_w(k)$.
Step 5: Using Eq. (10) compute and store the squared error.
Step 6: Set $m = m + 1$. If $m \leqslant L$ go to step 3; else, go to step 7.
Step 7: For $z = (\epsilon_1^2, \epsilon_2^2, \ldots, \epsilon_L^2)$ search and store the position containing the least-square error, producing d_j, where d_j is the estimated delay of the window position in the reference channel.
Step 8: Call algorithm 2.
Step 9: Set $j = j + 1$. If $j \leqslant N - W - 1$, go to step 1; else, go to step 10.
Step 10: Call algorithm 3.
Step 11: Stop.

Algorithm 2. Channel reconstruction

Data:

Parameters passed to algorithm:
h = Signal in channel #2
W = Number of data points in window
j = Start of window in reference channel
d_j = Computed delay of window position j

Output:

r = Reconstructed signal.

Steps:

Step 0: Set $i = 1$.
Step 1: If $r(j + i) = 0$, set $r(j + i) = h(j + d_j + i)$, go to step 3; else, go to step 2.
Step 2: Set $r(j + 1) = [r(j + 1) + h(j + d_j + i)]/2$.
Step 3: Set $i = i + 1$. If $i < W$, go to step 1; else, return.

Algorithm 3. Identification of events

Data:

Parameters passed to algorithm:
N = Number of data points
W = Window width
$d = (d_1, d_2, \ldots, d_{N-W-1})$, where d is the set of delays
ϵ = Interval defining closeness of consecutive delays
p = Number of data points outside $\pm\epsilon$ interval
q = Number of data points within $\pm\epsilon$ interval
β = Limit for the number of values outside $\pm\epsilon$ interval

Output:

d_e = Position of event in channel #2
j = Start of event in reference channel
k = End of event in reference channel

Steps:

Step 0: Set $j = 1$.

Step 1: Set $i = 1, p = 0, q = 0$. Store d_j and set $s = d_j$.
Step 2: If d_{j+1} is within $\pm\epsilon$ of d_j, go to step 3; else, go to step 4.
Step 3: Set $q = q + 1, s = s + d_{j+1}$. Go to step 5.
Step 4: Set $p = p + 1$. If $p > \beta$, go to step 6; else, go to step 5.
Step 5: Set $i = i + 1$. Go to step 2.
Step 6: If $q > W$, go to step 7; else, go to step 8.
Step 7: Output: $d_e = s/(q+1), j, k = j + i$.
Step 8: Set $j = j + i + 1, i = 1$. If $j < N - W - 1$, go to step 1; else, return.

The group of algorithms described above is given the name window-correlation technique (WCT) to distinguish it from other correlation methods. Several general estimation concepts were used to adapt the ordinary cross-correlation technique to the estimation of arbitrary delays. Section III will present the results of arbitrary delay estimation using the WCT on synthetic data as compared to the ordinary cross-correlation technique.

III. RESULTS

There are several ways to test the ability of the window-correlation technique to distinguish between arbitrary delays. In this section, two methods were decided upon to compare the window method against the ordinary cross-correlation technique. In order to test this new technique, it is necessary to create a test problem which has a known solution; therefore, test signals are set up where the number of events are known *a priori*. The synthetic data chosen consist of two, uncorrelated, Gaussian-white, zero-mean events, as shown in Fig. 1(a). The delayed version of this signal is shown in Fig. 1(b). These two signals will be corrupted by AGWN, which is uncorrelated with the signals. Also, the events are resolved well enough such that they do not overlap in time. The first event has been delayed 30 sample points while the second event has been delayed 10 sample points.

Although the number of events may be given, the delays between channels must still be computed. This is a nontrivial problem since the noise present can be of the same order of magnitude as the event signals.

The first method of comparison consists of distinguishing between the two delays and identifying the events in the second channel as given by algorithm 3. The signal-to-noise ratio was decreased for each successive calculation to find the lower bound for the window method. The expression for SNR chosen by the authors is given as follows:

$$\mathrm{SNR_{dB}} = 10\log(\overline{S^2}/\overline{N^2}), \tag{12}$$

where $\overline{S^2}$ and $\overline{N^2}$ are the mean-square values of the signal and noise, respectively. These values are easily obtained by the autocorrelation function.

The results of the delay estimation for the window-correlation technique and the ordinary cross correlation are shown in Table I, where $\tau_1(\tau_2)$ is the WCT estimated delay of the first (second) event and τ_o is the estimated delay by the ordinary cross-correlation method. As can be seen from Table I, the window-correlation technique estimates the two delays very accurately down to an SNR of 0 dB. Also, as expected, the ordinary cross-correlation method could only estimate one delay, if any. That delay turned out to be the delay of the largest event, since that would indicate the greatest area of correlation.

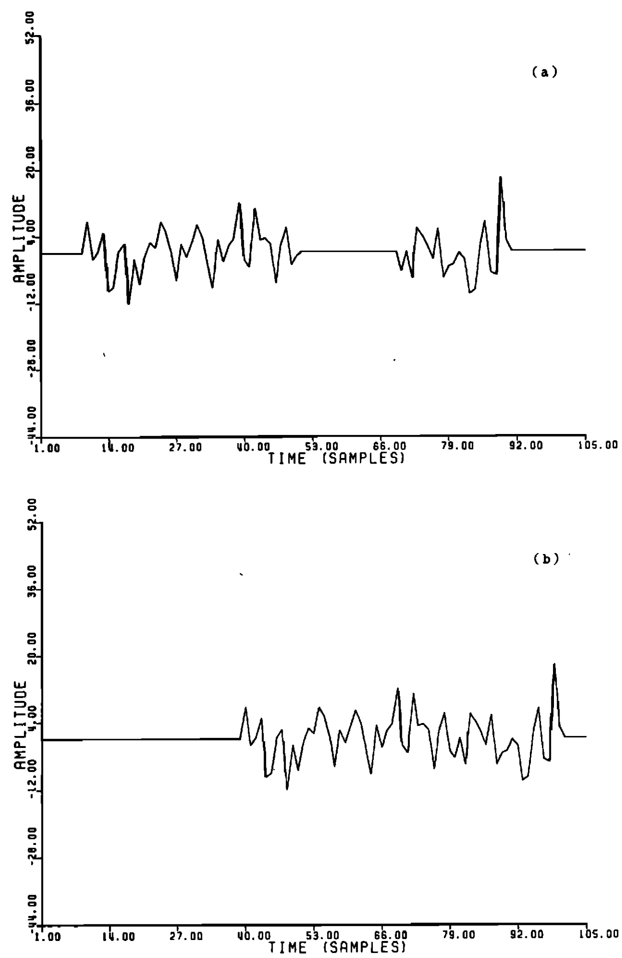


FIG. 1. (a) The signal contained in channel #1 is a Gaussian zero-mean process. The signal in (a) contains no additive noise at this point and consists of two pulses separated in time. (b) The signal in (b) is a delayed version of the original channel. The first pulse in (a) has been delayed 30 sample points while the second pulse has been delayed 10 sample points.

The second method of comparison uses one estimated delay at a time to align the second channel with the reference channel. This method does not look for patterns as in algorithm 2, but uses the complete set of estimated delays by calling algorithm 3 each time a delay is computed. The first step taken is to compute the squared error between the reference signal and the original second channel. The delays are then used to shift the second channel to the calculated position, producing the reconstructed signal. The next step is to compute the new squared-error value. These two computed values are used as follows:

$$\%\ \text{improvement in alignment} = [(\mathrm{SE1} - \mathrm{SE2})/\mathrm{SE1}]100\%, \tag{13}$$

where SE1 and SE2 are the initial and final squared errors, respectively. From Eq. (13), it is seen that, if there is complete alignment after estimation, i.e., SE2 equals zero, then the percentage improvement would be 100%.

TABLE I. Delay estimation (actual delays: $\tau_1 = 30$ samples, $\tau_2 = 10$ samples).

SNR(dB)	τ_1	τ_2	τ_o
100	30.000	10.000	30.000
80	30.000	10.000	30.000
60	30.000	10.000	30.000
40	30.000	10.000	30.000
20	29.943	10.000	30.000
10	29.846	10.000	30.000
5	29.761	10.000	30.000
0	30.000	10.476	30.000
−5	[a]	[a]	0.000
−10	[a]	[a]	0.000

[a] Indicates that the program cannot distinguish between the delays and, therefore, makes no decision.

However, with varying SNR, it is not expected to have complete alignment. Table II shows the results of the percentage improvement in alignment for both methods at different SNR. The interesting point to be made here is that, even at an SNR equal to −10 dB, the WCT can be used to align the two channels better than the ordinary cross correlation can at a very high SNR. Also, at medium to high signal-to-noise ratios, the WCT aligns the two channels very well. As mentioned previously, the noise level prevents perfect alignment.

Figure 2(a) and (b) shows the original and delayed signals corrupted by AGWN with an SNR of 40 dB. By using the WCT and reconstructing the second channel, the error in reconstruction can be seen in Fig. 2(c). Notice that the major contribution of error is in the region that contained no signal, only noise.

Again, the original and delayed signals are shown in Fig. 3(a) and (b) with an SNR of 0 dB. Figure 3(c) shows the original signal with noise versus the reconstructed signal using the WCT. As can be seen, the reconstruction is excellent, even at this low SNR. The error function is given in Fig. 3(d). Since this error looks so large and we have seen that the reconstruction was excellent, it must be concluded that the low SNR accounts for the size of the error. Also, as seen in Fig. 3(e), the WCT creates a smaller error than the ordinary cross correlation.

Figure 4(a)–(c) shows the results of the WCT with an SNR of −10 dB.

TABLE II. Percent alignment (WCT = window-correlation technique and OCC = ordinary cross correlation).

SNR(dB)	WCT(%)	OCC(%)
100	94.383	62.947
80	94.384	62.946
60	94.389	62.935
40	94.437	62.825
20	94.661	61.383
10	92.892	56.608
5	84.704	42.840
0	76.064	27.920
−5	68.020	0.000
−10	64.368	0.000

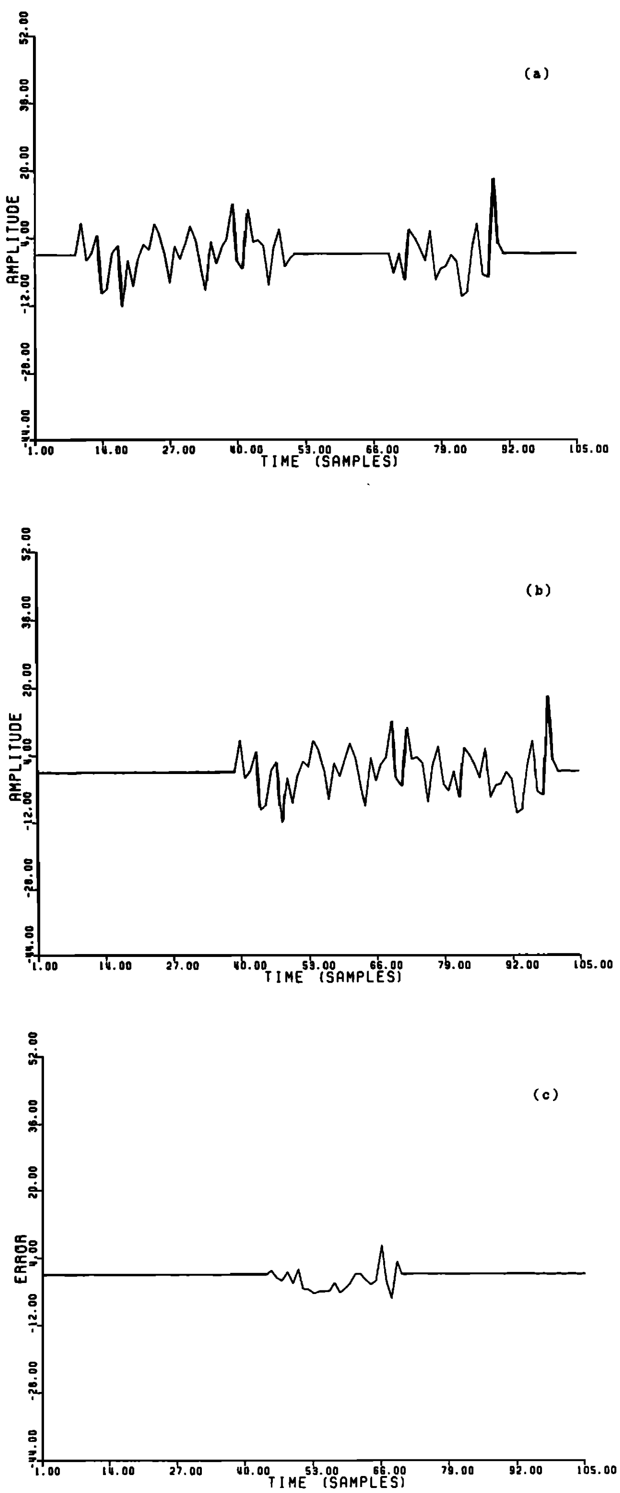
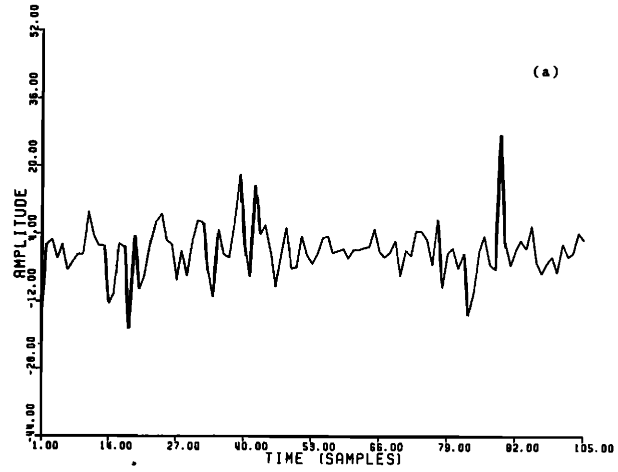
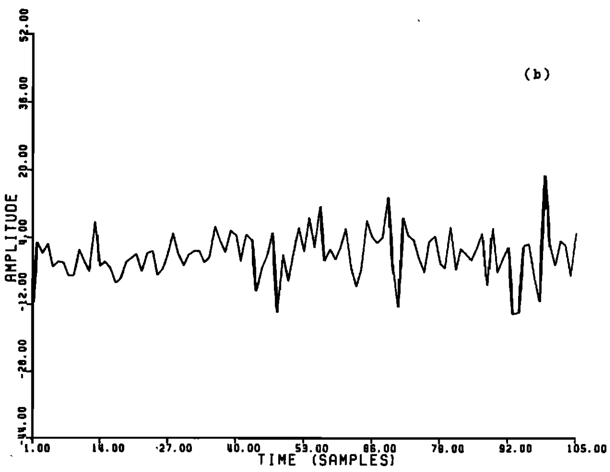


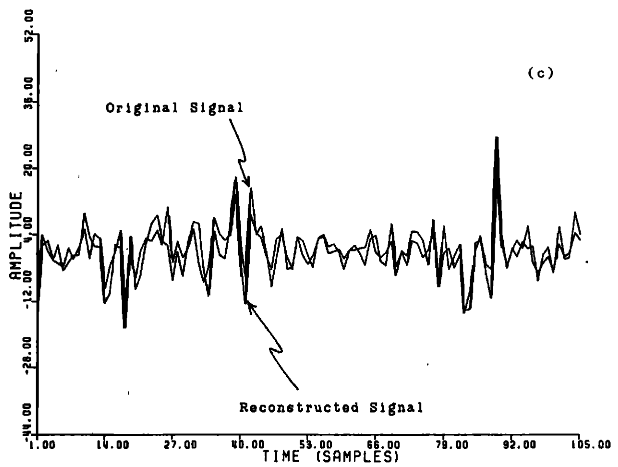
FIG. 2. (a) The original signal in Fig. 1(a) with AGWN. The signal-to-noise ratio (SNR) is 40 dB. (b) Channel #2, the delayed channel, and corrupted with AGWN. Again, the SNR was chosen to be 40 dB but the noise was not chosen to be the same random noise as in (a). (c) Error function obtained by using the window-correlation technique to reconstruct the delayed signal using the delays. The error was then found by taking the difference between the signal in (a) and the reconstructed signal.

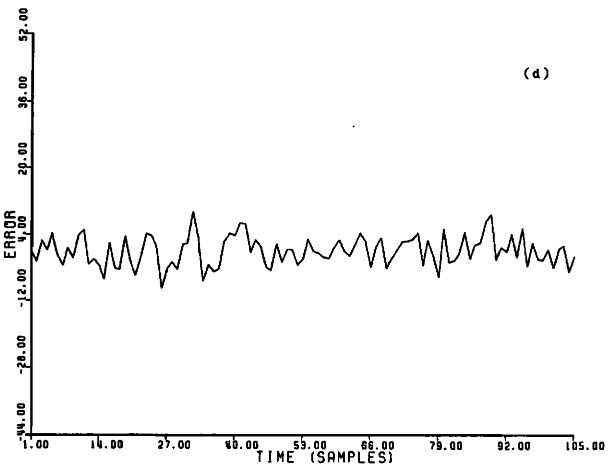
IV. DISCUSSION AND CONCLUSIONS

In this article, a method has been developed to estimate arbitrary time delays. Use has been made of the ordinary









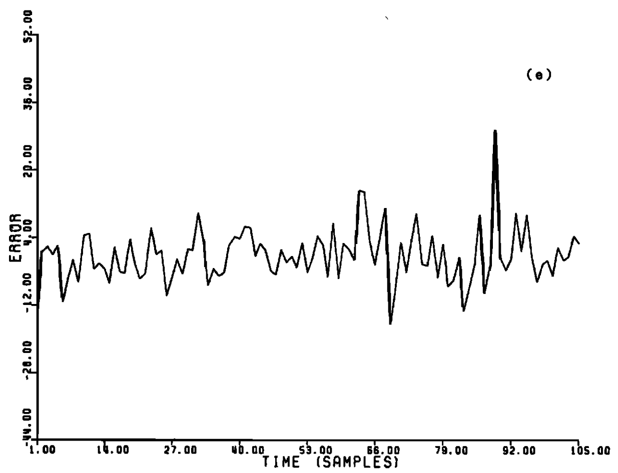


FIG. 3. (a) Original signal with SNR = 0 dB. (b) Delayed signal with SNR = 0 dB. (c) The original signal from (a) compared to the reconstructed signal obtained by using the WCT. The only differences arise from the fact that one channel contains a noise signal independent from the other. (d) Error function—the error found by using the WCT is shown here. Although the error looks rather large, it is smaller than the error in (e). Also, from (c), it is seen that the reconstruction was excellent and that most of the error is due to noise differences. (e) Error function obtained by using the ordinary cross-correlation method and reconstructing the signal.

cross-correlation technique to adapt to the case of several delays throughout a record length. The window-correlation technique uses the ordinary cross correlation along with a windowing and minimization routine to correlate small segments of the multichannel data. Then, the set of computed delays is searched for values consecutively close to each other and a decision is made as to the estimated delay for a specific window position.

As can be seen from the results, the WCT has no problem estimating arbitrary delays down to an SNR of 0 dB. However, this is provided that the events are fairly well resolved. Also, by using a complete set of delays, not just trends, the second channel can be aligned extremely well to the reference channel. Again, we did not expect the ordinary cross-correlation technique to work very well for multiple delays, and it did not. However, by making some modification, viz., the windowing process, it was used to develop the WCT and, in fact, worked very well.

There has been much consideration given to the need for further work to enhance the window-correlation technique. As mentioned in the Introduction, the WCT was not developed as a real-time process. However, there is the possibility of optimization of the algorithm at a future date. Also, work is in progress at the present time to find the effect of different window functions. Not only is this an important consideration, but the width of the window with respect to different frequencies should be examined. The concept of a dynamically changing window has been considered and is being studied at the present time. In conclusion, a method has been presented in this article that appears to be a very powerful tool in the estimation of arbitrary time delays.

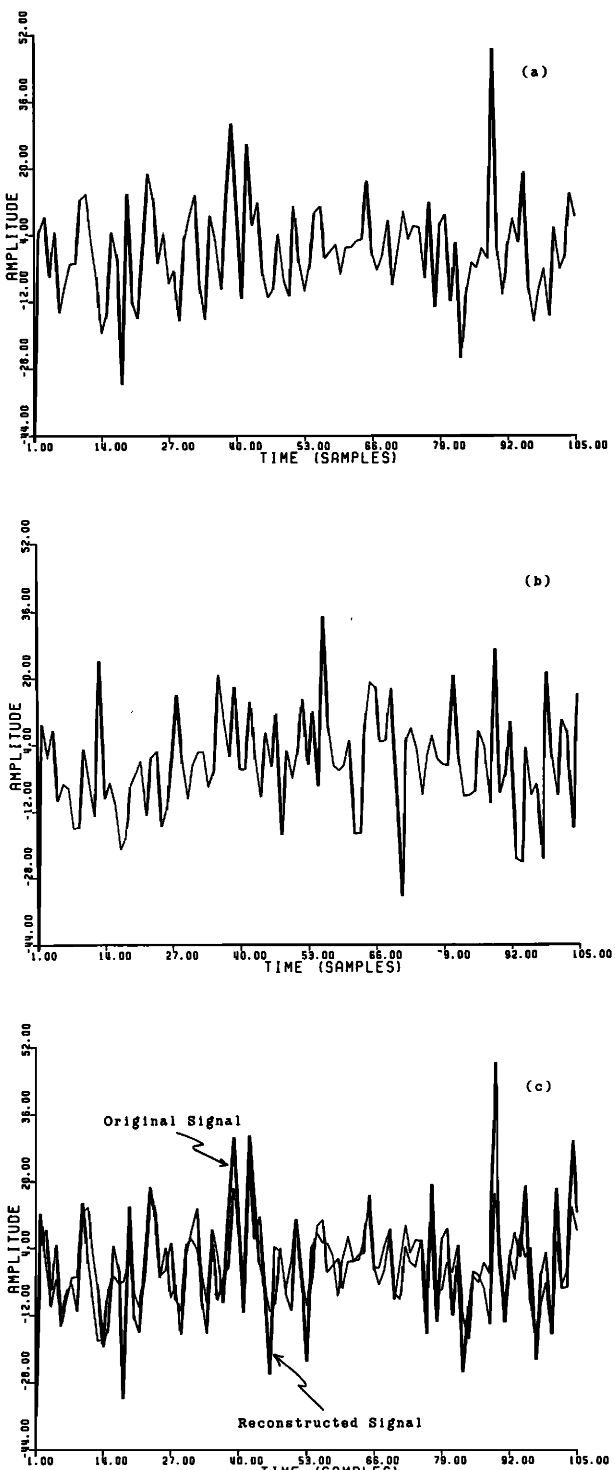


FIG. 4. (a) Original signal with SNR = − 10 dB. (b) Delayed signal with SNR = − 10 dB. (c) Original signal compared to reconstructed signal.

[1]Special issue on time delay estimation, IEEE Trans. Acoust. Speech Signal Process. **ASSP-29** (1981), Part II.

[2]J. N. Bradley and R. L. Kirlin, "Delay Estimation by Expected Value," IEEE Trans. Acoust. Speech Signal Process. **ASSP-32**, 19–27 (1984).

[3]H. Meyr and G. Spies, "The Structure and Performance of Estimators for Real-Time Estimation of Randomly Varying Time Delay," IEEE Trans. Acoust. Speech Signal Process. **ASSP-32**, 81–94 (1984).

[4]Y. Azenkot and I. Gertner, "The Least Squares Estimation of Time Delay Between Two Signals with Unknown Relative Phase Shift," IEEE Trans. Acoust. Speech Signal Process. **ASSP-33**, 308–309 (1985).

[5]C. Y. Wuu and A. E. Pearson, "On Time Delay Estimation Involving Received Signals," IEEE Trans. Acoust. Speech Signal Process. **ASSP-32**, 828–835 (1984).

[6]M. Azaria and D. Hertz, "Time Delay Estimation by Generalized Cross Correlation Methods," IEEE Trans. Acoust. Speech Signal Process. **ASSP-32**, 280–284 (1984).

[7]R. E. Boucher and J. C. Hassab, "Analysis of Discrete Implementation of Generalized Cross Correlator," IEEE Trans. Acoust. Speech Signal Process. **ASSP-29**, 609–611 (1981).

[8]A. Papoulis, *Probability, Random Variables, and Stochastic Processes* (McGraw-Hill, New York, 1965).

[9]A. Oppenheim and R. Shafer, *Digital Signal Processing* (Prentice–Hall, Englewood Cliffs, NJ, 1975).

[10]E. A. Robinson, *Stationary Time-Series* (Macmillan, New York, 1980).