The focus measurement technique for estimation of arbitrary time delays in multichannel, multievent systems

Ben-Chung Jan, Juan A. Henriquez, Terry E. Riemer, and Russell E. Trahan

Citation: The Journal of the Acoustical Society of America **90**, 2480 (1991); doi: 10.1121/1.402051
View online: https://doi.org/10.1121/1.402051
View Table of Contents: https://asa.scitation.org/toc/jas/90/5
Published by the Acoustical Society of America

The focus measurement technique for estimation of arbitrary time delays in multichannel, multievent systems

Ben-Chung Jan, Juan A. Henriquez, Terry E. Riemer, and Russell E. Trahan, Jr.
Department of Electrical Engineering, University of New Orleans, New Orleans, Louisiana 70148

(Received 9 November 1990; revised 7 March 1991; accepted 6 July 1991)

The estimation of arbitrary time delays in multichannel, multievent systems is studied using a new method called the focus measurement technique (FMT). The method introduced uses the window concept to isolate segments of the data record. However, instead of calculating the generalized cross correlation to estimate delays between data segments, the FMT (based on the assumption of normal distribution) combines windowed-segment points of every two channels to obtain the focal length of an ellipse which contains a certain amount of these distributed points. Based on simple geometric analysis, the FMT utilizes a windowed-segment distribution in order to estimate the event delays in the system. Simulation results show that this technique can find delays very accurately down to a signal-to-noise ratio (SNR = mean-square value of signal/mean-square value of noise) of 0 dB while reducing the amount of computation involved.

PACS numbers: 43.60.Gk

INTRODUCTION

Time delay estimation is one of the important subjects in the field of signal detection and estimation. In recent years, many papers have been published concerning this research;[1–12] in general, the methods found in the literature can be separated into two major subjects. First, there are methods used in the detection of underwater acoustic signals (by spatially separated sensors) for which it is desired to estimate delays between received signals in the presence of uncorrelated noise. Second, there are methods provided to find the lower bound for the variance of unbiased estimators in order to analyze the performance of realizable systems. Most of these methods however, only deal with fixed delay systems. Thus, no matter how many events are contained in one channel, these are delayed by some fixed sample interval in another channel.

When estimating a fixed delay between two channels, the popular techniques are based on cross correlation and generalized cross-correlation estimation implemented in the time domain or in the frequency domain. But in the real world, signals exist which contain arbitrary delays. One example is the analysis of audio signals where we might be interested in determining the time differences between related signal events in a multichannel audio system. Another example is the analysis of a reflection seismogram where, after shooting a source signal down into the earth, the reflected signal (recorded by sensors) can be considered as composed of delayed replicas of the source signal.

When channels have multievents with different delays, it can be predicted that the estimation of every delay by cross-correlation methods will fail since these techniques can only estimate the one event with maximum value. Therefore, our interest is to find algorithms that can be used to determine those arbitrary delays in multichannel, multievent systems embedded with uncorrelated noise. To estimate the event delays in these systems, a method called the window-correlation technique (WCT) was introduced by Callison *et al.*[13] Basically, this technique uses a (fixed-length) window to isolate a data segment from both channels of a two-channel system, and then computes the squared error between the autocorrelation of the reference segment and the cross correlation of both segments in order to estimate arbitrary event delays. The advantage of the WCT is that the estimation of every event turns out to be very accurate even in very noisy (low SNR) environments. However, the central processing unit (CPU) time consumed is extremely high since this technique needs to slide a window in both channels as well as to compute the cross correlation at every movement of the window. Another method called the modified window-correlation technique (MWCT) was introduced by Jan *et al.*[14] The MWCT is found to reduce the computation time but the method is only accurate for systems containing a small amount of events.

The purpose of this paper is to introduce the focus measurement technique (FMT) as a new method for the estimation of time delays which reduces computation time in systems containing multievents. The FMT is similar to the WCT; it uses a fixed-length window to isolate data segments but instead of computing cross-correlated values for the estimation of event delays, the former method combines every data point in one segment along with every corresponding data point in another segment to obtain a two-dimensional scatter plot. The procedure then calculates the focal length of an ellipse containing a statistically significant number of these distributed points. The window is then adjusted until the focal length is maximized, thereby indicating a maximum correlation. Simulation results show that the FMT can successfully estimate arbitrary delays down to a signal-to-noise ratio (SNR) of 0 dB while considerably reducing the CPU time. This paper is organized as follows: Sec. I describes the FMT and introduces algorithms for the estimation of time delays, Secs. II and III include the simulation results and conclusions, respectively.

I. THE FOCUS-MEASUREMENT TECHNIQUE

A. Problem analysis

We start this analysis by first defining a model for a multichannel, multievent system. For simplification, we only consider the case of two channels with two separate events occurring in channel 1 and we assume those events also occur in channel 2 as attenuated and delayed versions of the events in channel 1. Also, for ease of computer simulation, we assume the system has already been discretized with unit sample frequency; i.e., $f = 1/T \equiv 1$. Thus the system can be described as

$$x(k) = h_1(k) + h_2(k) + n_1(k),$$
$$y(k) = \gamma h_1(k-\tau_1) + \mu h_2(k-\tau_2) + n_2(k), \quad (1)$$
$$0 \leqslant k \leqslant N-1,$$

where x is the signal received from channel 1, y is the signal received from channel 2, h_1, h_2 are the different (transmitted) signal events in both channels, k is the sample time, τ_1, τ_2 are the arbitrary time delays, γ, μ are the attenuation factors, n_1, n_2 are additive Gaussian white noise (AGWN), and N is the data record length. It is assumed that $h_1(k)$ and $h_2(k)$ are two separate events that do not overlap in time. Similarly, the delayed events in channel 2 do not overlap in time. For this presentation, we assume $h_1(k)$ to be a burst of transmitted white noise for a given length of time; likewise, $h_2(k)$ will be a burst of white noise that occurs after the appearance of $h_1(k)$. That is, there is no overlap of the two bursts.

To successfully separate each event, the first step is to isolate it by using a window. Define the windowed channels as

$$x_w(k) = x(k)w(k), \quad y_w(k) = y(k)w(k), \quad (2)$$

where

$$w(k) = 1, \quad 0 \leqslant k \leqslant W-1$$
$$= 0, \quad W \leqslant k \leqslant N-1, \quad (3)$$

and W is the window length.

Consider the windowed data segments x_w and y_w. If we combine every element of x_w with a corresponding element of y_w, i.e., $[x_w(0), y_w(0)]$, $[x_w(1), y_w(1), \ldots, x_w(W-1), y_w(W-1)]$, an $x_w - y_w$ coordinate system can be built such that every pair of data points represents a unique point in the $x_w - y_w$ plane. Based on simple geometric analysis, we find two distinct distributions. First, if x_w and y_w represent the same windowed segment, without noise embedded in the system, the two elements in every pair will match exactly. Thus, the distribution will stay in a straight line which passes through the origin and has slope equal to the attenuation factor (γ or μ). But with noise present in the system, the distribution is scattered around this line. Second, if x_w and y_w do not contain exactly the same windowed segment, the distribution will have some random form which strongly depends on the combined data. Figure 1 shows these simple geometric interpretations. The distribution of two matched segments without noise is shown in Fig. 1(a) and with noise present is shown in Fig. 1(b). The distribution of two nonmatched segments without noise is shown in Fig. 1(c) and with noise present is shown in Fig. 1(d).

B. System without noise present

We can describe the FMT conceptually by considering the system without noise added. If we window both channels with a fixed length window, combine every corresponding element as a pair, and map these points into a two-dimensional plane, the distribution shall implicitly show the relation between the two windowed segments. If the window of channel 1 is fixed and the window of channel 2 is moved step by step through the entire channel, a sequence of distributed information is obtained which can be used to identify the time delay of the particular windowed segment of channel 1. Obviously, the match of the windowed segments from each channel will have a distribution following a straight line. Therefore, by looking at those distributions, we can easily find the delay.

It is possible to slide the window of channel 1 step by step and to repeat the same process until the window has moved through the entire channel 1. A sequence of numbers is then obtained which represents the time delays of the corresponding windowed segments. Looking at these delay numbers, we can observe that there are certain successive delays having the same value. Indeed, this value is exactly the time shift of the specific event in channel 2. Note that the chosen window length is an important factor. If the window of channel 1 covers both events, there is no way to obtain a distribution following a straight line.

C. System with noise present

When noise is present in the system, the distribution will no longer follow a straight line when both windowed segments match. The procedure becomes more complex than just looking at the distribution and an estimation technique is needed. Fortunately, with the FMT, the time delay of every windowed segment in a noisy environment can be successfully measured.

From the previous analysis, we know that the distribution will have a tendency to stay around a straight line when both windowed segments match. Since there is now no straight line in the distribution [Fig. 1(b)], we must estimate it. Our strategy is to find a line passing through the origin and to obtain the best fit to the distributed points in the least-distance-squared sense (note that this criterion is different from the "method of least squares," a popular technique for linear regression).

Refer to Fig. 2 and assume the line to be found is $Y = \alpha X$, where α is the slope and there are M distributed points represented as (X_i, Y_i), $i = 1,2,\ldots,M$. The distance between one distributed point and the estimated line is

$$D_i = \sqrt{(X_i - X)^2 + (Y_i - Y)^2}. \quad (4)$$

Now, by simple algebraic derivations along with the fact that the product of the slopes of two perpendicular lines is -1, we can find the relation

$$X = (X_i + \alpha Y_i)/(1+\alpha^2). \quad (5)$$

Therefore, our criterion can be defined as

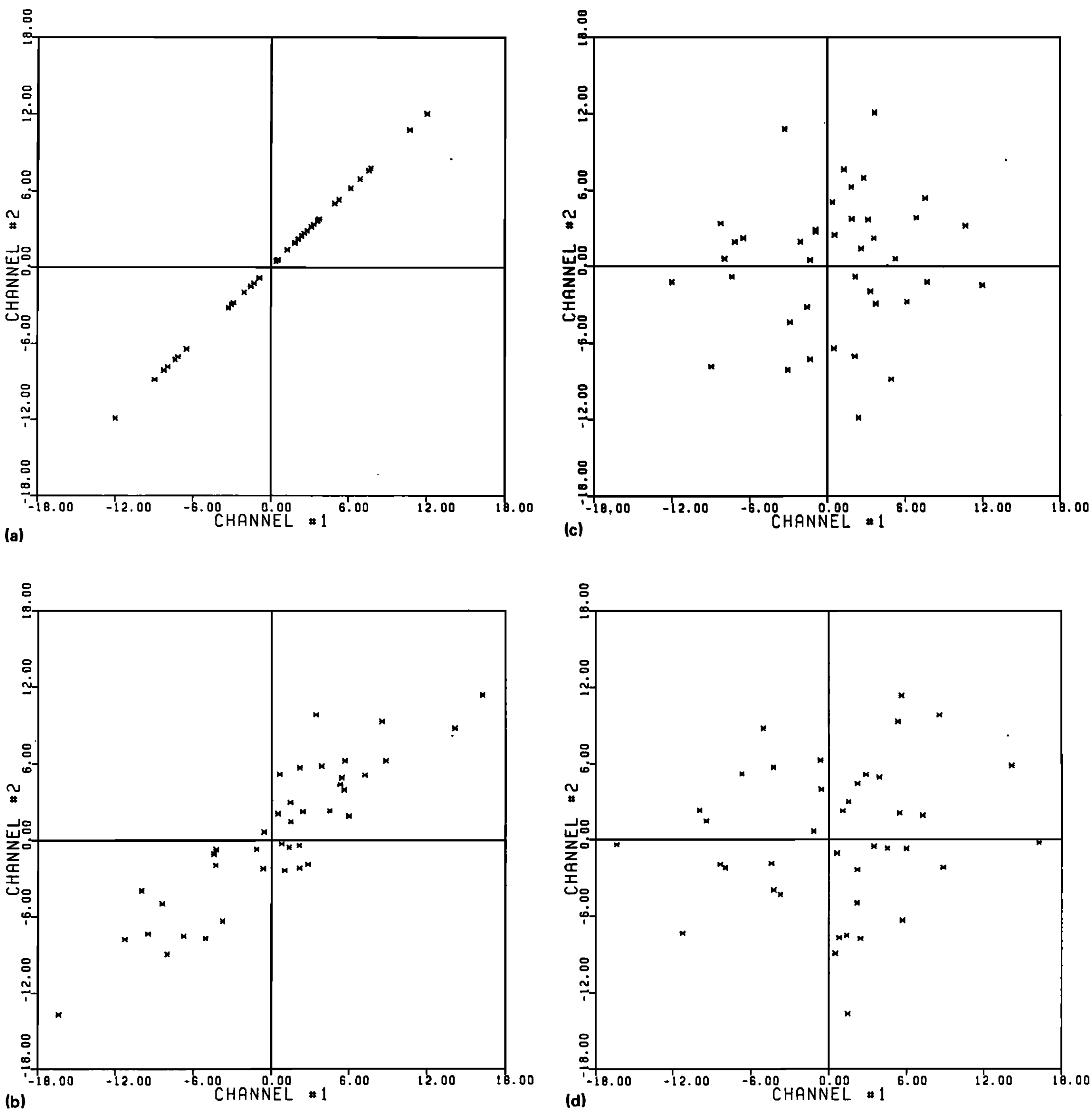


FIG. 1. Distribution of two matched segments (a) without noise and (b) with noise present. Distribution of two nonmatched segments (c) without noise and (d) with noise present.

$$\min \sum_{i=1}^{M} D_i^2 = \min \sum_{i=1}^{M} (X_i - X)^2 + (Y_i - \alpha X)^2$$
$$= \min \sum_{i=1}^{M} \left(X_i - \frac{X_i + \alpha Y_i}{1+\alpha^2} \right)^2 + \left(Y_i - \alpha \frac{X_i + \alpha Y_i}{1+\alpha^2} \right)^2. \quad (6)$$

To find the value of α satisfying Eq. (6), we simply take its first and second derivatives with respect to α. These derivatives are

1st derivative: $[2/(1+\alpha^2)^2](A\alpha^2 + B\alpha - A)$, (7)

2nd derivative:

$$[2/(1+\alpha^2)^3](-2A\alpha^3 - 3B\alpha^2 + 6A\alpha + B), \quad (8)$$

where

$$A = \sum_{i=1}^{M} X_i Y_i, \quad B = \sum_{i=1}^{M} X_i^2 Y_i^2. \quad (9)$$

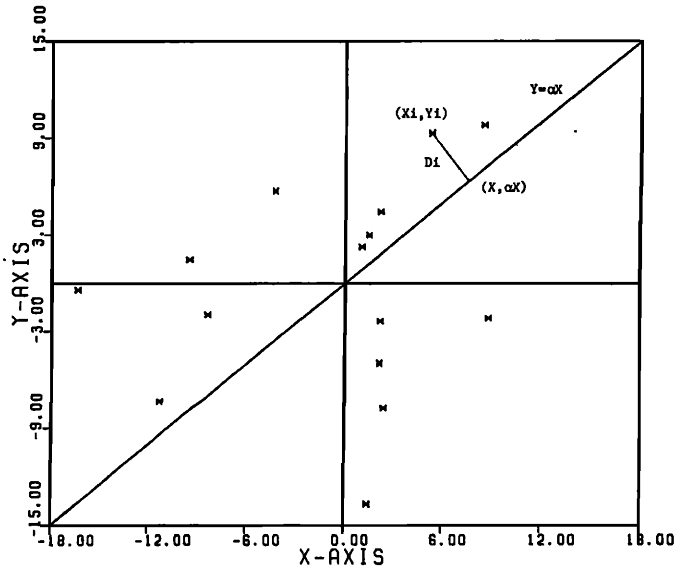


FIG. 2. Geometric expression of the make-up line $Y = \alpha X$, where α is the slope, the set (X_i, Y_i) represents the distributed points, and D_i [as given by Eq. (4)] is the distance between one distributed point and the estimated line.

The roots of the equation $A\alpha^2 + B\alpha - A = 0$ are

$$\alpha_{1,2} = (-B \pm \sqrt{B^2 + 4A^2})/2A. \tag{10}$$

We can use the above expression to identify which value, α_1 or α_2, will meet the minimum criterion of Eq. (6). Note that, no matter what values A and B take, $B^2 + 4A^2 > 0$ is always true, so there always exist two real solutions to Eq. (10). Meanwhile, it can be easily shown that the product of α_1 and α_2 is -1. This implies that these two solutions form two perpendicular slopes. Therefore, if one meets the minimum criterion, the other will meet the maximum criterion. Let us call the line with the slope satisfying the minimum criterion the *main-axis line* and the other line the *auxiliary-axis line*.

The main-axis line divides the distributed points into two parts. Define the distances in one part as positive and the distances in the other part as negative. Since we are dealing with added Gaussian white noise, we can assume these distances have a normal distribution with zero mean. Based on this assumption, the distance which includes 95% of the whole distribution can be calculated by the standard normal distribution formulas. Let us call this estimated distance the short axis (SA). Then, it can be shown that

$$\mathrm{SA} = \mu_{\mathrm{main}} + 1.96\sigma_{\mathrm{main}}, \tag{11}$$

where μ is the mean value of the distances and σ is the standard deviation of the distances.

The constant 1.96 in Eq. (11) comes from the standard normal distribution table. We want to measure the distance value d which has 95% of the total distribution between $\pm d$, and 95% of the total distribution is enclosed within ± 1.96 of the normal curve; i.e., $P[-1.96 \leqslant Z \leqslant 1.96] = 0.95$, where Z is a random variable from the standard normal distribution. Similarly, the same procedure can be used with the auxiliary-axis line and a long axis (LA) can be calculated as

$$\mathrm{LA} = \mu_{\mathrm{aux}} + 1.96\sigma_{\mathrm{aux}}. \tag{12}$$

The SA and LA parameters play the most important roles in our technique. When two windowed segments match, the distribution will stay closer to the main-axis line than for those that do not match. Therefore, the distribution with the longest LA and shortest SA implies the most "likeness" of the two windowed segments. The remaining problem consists of choosing a reasonable measure for the estimation. In this study, this parameter is chosen as the measured focal length (MFL), or

$$\mathrm{MFL} = \sqrt{\mathrm{LA}^2 - \mathrm{SA}^2}. \tag{13}$$

From the above equation, it can be seen that this parameter is the measured focal length of an ellipse with LA as the major axis and SA as the main axis. From the properties of an ellipse, the longer the focal length, the sharper the ellipse. Therefore, a large MFL implies that most points of the distribution stay closer to the main-axis line. This certainly represents the best match of the two windowed segments. The cases corresponding to two matched and two nonmatched windowed segments are shown in Fig. 3(a) and (b), respectively.

Fixing the window of channel 1 and sliding the window of channel 2 through the entire channel, we can get a sequence of MFLs. Sorting through this sequence, we then find the maximum MFL and subtract the position where the maximum MFL occurs from the position where the fixed window of channel 1 shows an example of a sequence of MFLs when the fixed window of channel 1 starts at position 1. As we can see, the maximum value is at position 31, thus the estimated delay of this specific windowed segment is 30 (see Fig. 4). Moving the window of channel 1 by one sample and repeating the same procedure, we find the time delay d_2 of this windowed segment. After moving the window through the entire channel 1, a sequence of numbers is obtained which represents the delays of the corresponding windowed segments of channel 1. These segment-delays, $\{d_1, d_2, \ldots, d_N\}$, are used to find the actual event delays as discussed later in this section.

Based on the above analysis, we present the following algorithm to estimate the delay of every windowed segment.

1. Algorithm I: Estimate segment delays

Input. x is the signal received from channel 1, y is the signal received from channel 2, N is the data record length, and W is the window length.

Output. $\{d_1, d_2, \ldots, d_N\}$ is the set of delays.

Steps. Step 0: Set $i = 0, j = 0$.

Step 1: Window channel 1, starting at i; producing $x_w(n)$, $i \leqslant n \leqslant W + i - 1$.

Step 2: Window channel 2, starting at j; producing $y_w(m)$, $j \leqslant m \leqslant W + j - 1$.

Step 3: Compute the slopes $\alpha_{1,2}$ by Eq. (10).

Step 4: Identify the main-axis line and the auxiliary-axis line by Eq. (10).

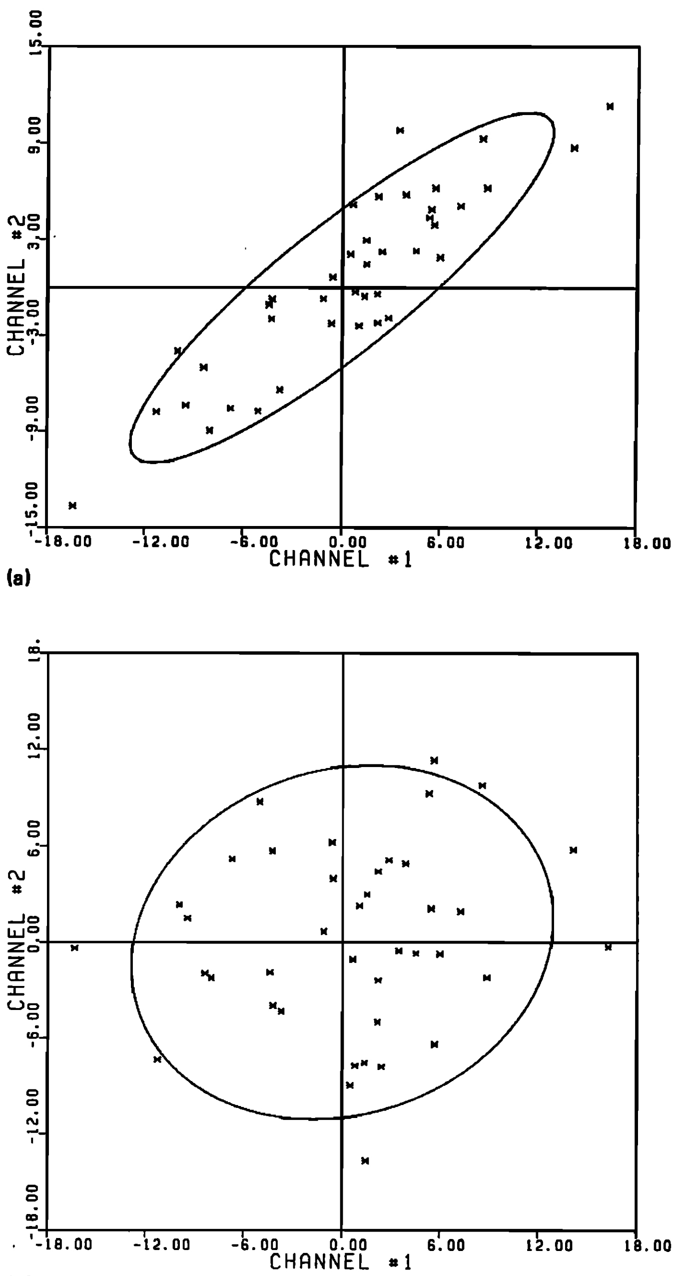


FIG. 3. (a) Estimated ellipse enclosing the distribution points of two matched segments. From Eq. (13), the measured focal length (MFL) will predict a sharp ellipse. (b) Estimated ellipse enclosing the distribution points of two nonmatched segments. From Eq. (13), the measured focal length (MFL) is smaller than that of (a) and thus the ellipse is not well defined.

Step 5: Compute μ_{main} and σ_{main}.

Step 6: Compute μ_{aux} and σ_{aux}.

Step 7: Compute SA and LA by Eq. (11) and Eq. (12), respectively.

Step 8: Compute $\text{MFL}(j) = \sqrt{\text{LA}^2 - \text{SA}^2}$.

Step 9: Set $j = j + 1$.

If $j \leqslant N - W + 1$, go to step 2;

else, go to step 10.

Step 10: Search for $J_{\max}$ where the maximum value of array $\text{MFL}(j)$ occurs, and compute $d_i = J_{\max} - i$.

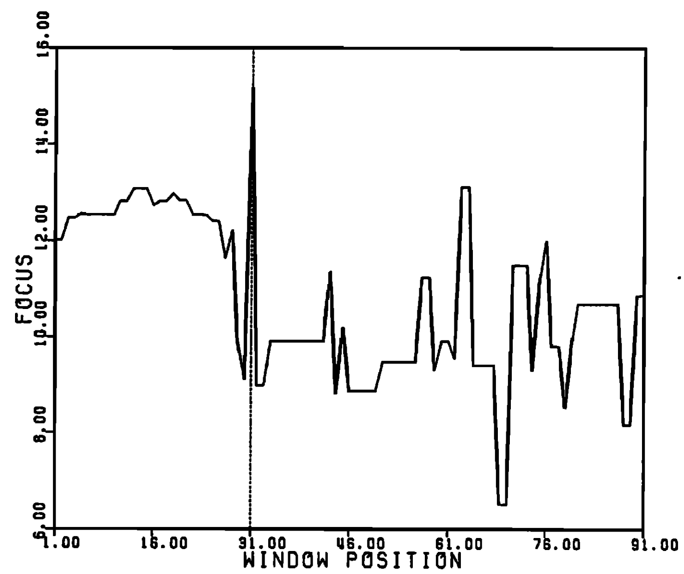


FIG. 4. Example showing a sequence of MFLs when the fixed window of channel 1 starts at position 1. The maximum value occurs at position 31 and thus the estimated delay for this windowed segment is 30.

Step 11: Set $i = i + 1$; reset $j = 0$.

If $i \leqslant N - W + 1$, go to step 1;

else, go to step 12.

Step 12: Save $\{d_1, d_2, \ldots, d_N\}$. Stop.

So far, the above technique only gives us a sequence of segment delays $\{d_1, d_2, \ldots, d_N\}$. Our next concern is to find the actual event delays based on these numbers.

Before presenting the algorithm that identifies the event delays, we wish to analyze the details of the estimation procedure. Note that after windowing one data segment from channel 1, the segment delay is then estimated. Therefore, for simplification, we assume this process is finished after every window movement in channel 1. Now, consider the case when the SNR is so high that the noise does not contribute seriously to the system and assume the window of channel 1 is at position one. After the segment delay is found, the window is moved by one sample unit through the entire channel. Depending on the window position, several situations can be observed and these are summarized below.

(1) The window does not cover any event point. The estimation is actually redundant in this case; this means that one delay is still estimated, but it is actually a "feigned number" due to the noise.

(2). The window touches the initial boundary of one event and starts to move into the event. The success of the estimation will then depend on how the event points in the windowed segment dominate the estimation. Generally, more event points have more chance to succeed.

(3) The window is inside the event. In this case, the estimated delays should all have the same value.

(4) The window starts to cross the final boundary and moves outside the event. Here, the success of the estimation will again depend on how the event points dominate the estimation.

The event delays can be found by sorting out these delay numbers in order to find a certain amount of successive delays having the same value. Note that when the window is moving into the event, there is a tendency to increase the event points, and when the window is moving out of it there is a tendency to increase the nonevent points. This implies that before the window moves into the event, the estimated delays will start to have the same values, and before the window moves out of it, the feigned numbers will appear. Thus, when the window of channel 1 is moving through one event, it will have a sequence of the same delay numbers; however, the starting and ending points for these numbers will not be exactly the same as those of the event.

Similar situations can be obtained when the SNR decreases. However, when the system is corrupted by more noise, the feigned numbers will increase not only near the final boundary, but also inside the event. Therefore, to calculate the estimation, we need another algorithm. When a certain amount of consecutive delays showing a particular value is obtained, the algorithm can identify the event delay by ignoring or weighting down the feigned numbers.

The "identification of events" algorithm,[13] satisfies the above requirements and can be used to solve the problem just described. However, instead of weighting down the feigned numbers inside a sequence of particular numbers, this algorithm sets up one small bound, i.e., $\pm\epsilon$. After a certain number of successive values stays inside the $\pm\epsilon$ interval, any delay number detected outside this bound is considered a feigned number. Otherwise, the average of all delays is taken to include every number inside the $\pm\epsilon$ interval and the estimated event delay is calculated. Since this algorithm is used in the simulation, we include it here with some notation changed.

2. Algorithm 2: Identify event delays

Input. $\{d_1, d_2, ..., d_N\}$ is the set of delays, N is the data record length, W is the window length, ϵ is the interval defining closeness of consecutive delays, p is the number of data points outside $\pm\epsilon$ interval, q is the number of data points within $\pm\epsilon$ interval, and β is the limit for the number of values outside $\pm\epsilon$ interval.

Output. D_e is the position of event in channel 2, j is the start of event in channel 1, and k is the end of event in channel 1.

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.

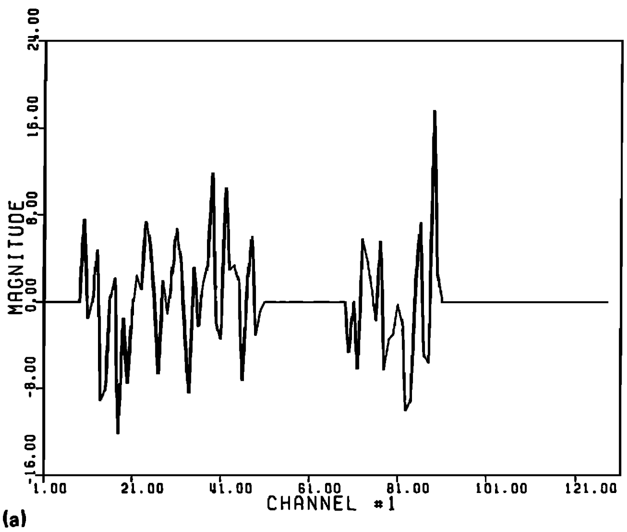
Step 7: Output $D_e = s/(q+1), j$, and $k = j + i$.

Step 8: Set $j = j + i + 1$, reset $i = 1$.
If $j < N - W - 1$ go to step 1;
else go to step 9.

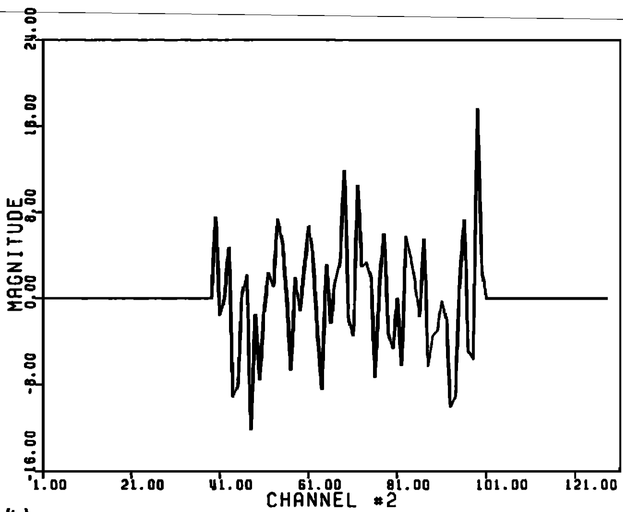
Step 9: Stop.

II. SIMULATION RESULTS

The algorithms described in the previous section have been successfully implemented on a DEC VAX-8600 cluster computer system. Our primary intention for testing the capabilities of the FMT is not only to see whether it can accurately estimate the event delays, but also to see whether this method provides any improvement when compared to the WCT. Thus the same set of synthetic data implemented in Ref. 13 is used as the test signals for the FMT. Each signal consists of two uncorrelated Gaussian white zero-mean events as shown in Fig. 5. Note that there are 128 sample



(a)



(b)

FIG. 5. (a) Synthetic data record of channel 1. This record consists of 128 samples containing two events, the first event occurs from samples 10 to 50 while the second one occurs from samples 71 to 90. (b) Synthetic data record of channel 2. This record consists of 128 samples containing two events, the first event occurs from samples 40 to 80 while the second one occurs from samples 81 to 100.

TABLE I. Identification of event-delays for several values of SNR (actual delays: $\tau_1 = 30$, $\tau_2 = 10$ sample points). The asterisk indicates that the program cannot identify the delay.

SNR (dB)	FMT τ_1	FMT τ_2	WCT τ_1	WCT τ_2
100	30.000	10.000	30.000	10.000
80	30.000	10.000	30.000	10.000
60	30.000	10.000	30.000	10.000
40	30.000	10.000	30.000	10.000
20	30.000	10.000	29.943	10.000
10	30.000	10.000	29.846	10.000
5	30.000	10.000	29.761	10.000
0	30.000	10.000	30.000	10.476
−5	*	*	*	*
−10	*	*	*	*

TABLE III. Starting and ending points of events from the FMT (fixed window length = 15 sample points). Actual start of event 1 = 10, end of event 1 = 50. Actual start of event 2 = 71, end of event 2 = 90

SNR (dB)	Start of event 1	End of event 1	Start of event 2	End of event 2
100	4	39	69	89
80	4	39	69	89
60	4	39	69	89
40	4	39	69	89
20	4	41	69	89
10	4	40	69	89
5	4	41	69	89
0	4	41	69	89

points in each channel. In channel 1 (2), the first event occurs from sample 10(40) to sample 50(80) and the second event occurs from sample 71(81) to sample 90(100). Thus delays are chosen in channel 2 to be 30 sample points from the first event and 10 sample points from the second event with respect to channel 1. Moreover, there are no overlaps in time between the events.

The channel signals are corrupted by different AGWNs (for testing at several values of SNR). These AGWNs are generated using the same seeds of the same random number generators. The system SNR is defined as

$$\mathrm{SNR}_{\mathrm{dB}} = 10 \log(\overline{S^2} / \overline{N^2}), \tag{14}$$

where $\overline{S^2}$ is the mean-square value of the signal and $\overline{N^2}$ is the mean-square value of the noise. Note that these values can be calculated from the autocorrelation function.[15]

After running the "estimate segment delays" algorithm of the FMT and the WCT separately, the output delays, $\{d_1, d_2, \ldots, d_N\}$, are used as the input to the "identify event delays" algorithm. The estimated event delays are computed for several values of SNR and shown in Table I. As can be observed from this table, both techniques can estimate the two non-uniform event delays very accurately down to an SNR of 0 dB. Meanwhile, Tables II and III list the individual identification results for the starting and ending points of the events for the WCT and the FMT, respectively. Note that no matter which technique is used, these values change when different window lengths are used but they all start(end) before the actual event starting(ending) point.

The advantage of using the FMT over the WCT is shown by comparing the average CPU time needed to run the estimate segment-delays algorithm. While the WCT's CPU time is 7 min and 54.56 s,[13] that of the FMT is 11.07 s (over 40 times faster). Note that the CPU time needed to run the identify event delays algorithm is not included since we only consider the comparable part between these techniques; moreover, the WCT's CPU time is less than 0.2 s, which is so small that it can be neglected here.

A parameter called the percentage improvement in alignment (PIA) is used in order to test the accuracy of the technique. This parameter is obtained by computing the squared error between the original two channel signals at every position (SE1), using the estimated delays to align the two channels by shifting the signal of channel 2 to the calculated positions, and then computing the squared error between them (SE2) again. These squared error values, SE1 and SE2, are used to find the PIA parameter as

$$\mathrm{PIA} = [(\mathrm{SE1} - \mathrm{SE2})/\mathrm{SE1}] \times 100. \tag{15}$$

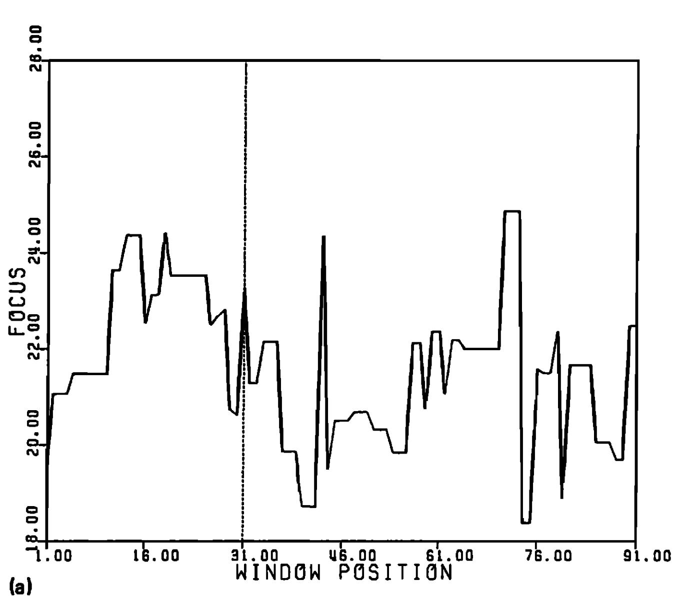
As can be expected, the alignment and the accuracy of the estimated delay will increase as the SNR increases. If the estimated delay in every position is correct, and there is complete alignment, SE2 is only the squared error of the noise in both channels and the percentage improvement should be close to 100%. However, when the SNR decreases, the amount of feigned numbers will increase and the PIA will decrease.

TABLE II. Starting and ending points of events from the WCT (fixed window length = 15 sample points). Actual start of event 1 = 10, end of event 1 = 50. Actual start of event 2 = 71, end of event 2 = 90.

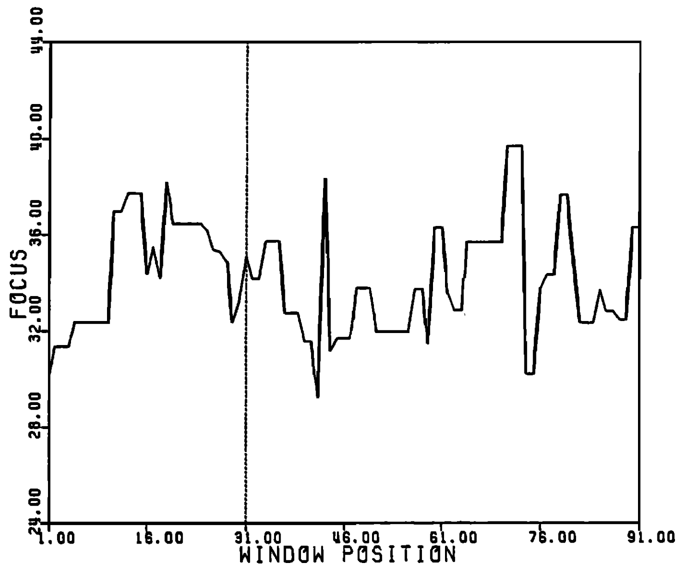
SNR (dB)	Start of event 1	End of event 1	Start of event 2	End of event 2
100	1	48	63	90
80	1	48	63	90
60	1	48	63	90
40	1	48	63	90
20	1	48	63	89
10	1	42	63	89
5	1	42	64	89
0	4	40	69	89

TABLE IV. Percentage improvement in alignment.

SNR(dB)	FMT(%)	WCT(%)
100	96.598	98.458
80	96.598	98.458
60	96.598	98.457
40	96.598	98.397
20	96.180	98.207
10	91.421	94.011
5	83.034	88.958
0	69.043	79.868
−5	59.884	72.908
−10	57.120	70.630



(a)



(b)

FIG. 6. (a) Plot of MFL values for SNR = − 5 dB. The plot shows the local maxima surrounding the estimated segment delay at position 31. (b) Plot of MFL values for SNR = − 10 dB. The plot shows the increase in local maxima surrounding the estimated segment delay at position 31.

Table IV shows the results of this test for both techniques for several values of SNR. As we can see from this table, there is a high percentage improvement in alignment until the SNR drops below 10 dB. When the SNR falls below 10 dB, the WCT provides some improvement (higher PIA) over the FMT.

III. CONCLUSIONS

The focus-measurement technique (FMT) has been proposed for the estimation of nonuniform time delays in multichannel, multievent systems. The method starts by first isolating (through a fixed-length window) a data segment from both channels of a two-channel system. The FMT (based on the assumption of normal distribution) then combines every windowed segment point of one channel with a corresponding point from the other channel in order to calculate the focal length of an ellipse containing a certain amount of these distributed points. A sequence of focal lengths is obtained by fixing the window of the reference channel (1), while moving the window through the processing channel (2). The position where the maximum focal length occurs is then the estimated delay of the specific window segment. The FMT proceeds by moving the window through the reference channel and repeating the above procedure until a sequence of windowed segment delays are found (by algorithm 1) and used to estimate the event delays (by algorithm 2).

Simulation results show that the FMT can accurately estimate event delays down to an SNR of 0 dB while reducing the computation time considerably (in our test case, the FMT is over 40 times faster than the WCT). Although the test results show that both the FMT and the WCT cannot estimate the event delays when the SNR drops below 0 dB, some information is still available. Note that the segment-delay is estimated by finding the global maximum (or minimum) measured focal length (MFL); however, when the SNR drops below 0 dB, there is a local maximum (or minimum) at the delay position. Figure 6 shows the plots of the MFLs when the window of channel 1 is at position 1, and the SNR is at − 5 dB [Fig. 6(a)] and at − 10 dB [Fig. 6(b)]. As can be observed from this figure, the value at position 31 (where the delay of this windowed segment occurred) is a local maximum. Thus the FMT is still capable of estimating this delay even when there are several maxima located elsewhere than are greater than this value. Also note that lowering the SNR [Fig. 6(b)] increases the local maxima surrounding this value.

Finally, the FMT can identify the arbitrary delay of one event very accurately; however, it cannot estimate the starting and ending points of the event. Therefore, we only know whether the delay exists around one segment of the data and we can thus identify it. Also, several window lengths have been used in the test. It is known that different events need different window lengths to obtain a better estimate. Our future research will focus on finding this optimum window length.

[1] C. Y. Knapp and G. C. Carter, "The Generalized Correlation Method for Estimation of Time Delay," IEEE ASSP **24**, 320–370 (1976).
[2] D. Hertz, "Time Delay Estimation by Combining Efficient Algorithms and Generalized Cross-Correlation Methods," IEEE ASSP **34**, 1–7 (1986).
[3] J. C. Hassab and R. E. Boucher, "Optimum Estimation of Time Delay by a Generalized Correlator," IEEE ASSP **27**, 373–380 (1979).
[4] L. C. Ng and Y. Bar-Shalom, "Multisensor Multitarget Time Delay Vector Estimation," IEEE ASSP **34**, 669–678 (1986).
[5] G. Su and M. Morf, "The Signal Subspace Approach for Multiple Wideband Emitter Location," IEEE ASSP **31**, 1478–1492 (1983).
[6] S. M. Azaria and D. Hertz, "Time Delay Estimation by Generalized Cross Correlation Methods," IEEE ASSP **32**, 280–285 (1984).
[7] H. Meyr and G. Spies, "The Structure and Performance of Estimators for Real-Time Estimation Randomly Varying Time Delay," IEEE ASSP **32**, 81–94 (1984).
[8] S. K. Chow and P. M. Schultheiss, "Delay Estimation Using Narrowband Processes," IEEE ASSP **29**, 478–484 (1981).
[9] J. P. Ianniello, "Time Delay Estimation Via Cross-Correlation in the Presence of Large Estimation Errors," IEEE ASSP **30**, 998–1003 (1982).

[10] A. J. Weiss and E. Weinstein, "Fundamental Limitation in Passive Time Delay Estimation—Part I: Narrowband System," IEEE ASSP **31**, 472–486 (1983).
[11] E. Weinstein and A. J. Weiss, "Fundamental Limitation in Passive Time Delay Estimation—Part II: Wideband System," IEEE ASSP **32**, 1064–1078 (1984).
[12] J. P. Ianniello, "Lower Bounds on Worst Case Probability of Large Error for Two Channel Time Delay Estimation," IEEE ASSP **33**, 1102–1110 (1985).
[13] J. D. Callison, T. E. Riemer, and R. E. Trahan, Jr., "Estimation of Arbitrary Time Delays of Multi-Channel Synthetic Data," J. Acoust. Soc. Am. **81**, 1000–1006 (1987).
[14] B.-C. Jan, J. A. Henriquez, T. E. Riemer, and R. E. Trahan, Jr., "The Modified Window-Correlation Technique for Estimation of Arbitrary Time Delays in Multi-Channel Multi-Event Systems," for publication as correspondence, ref. manuscript 747, currently on revision, J. Acoust. Soc. Am. (submitted).
[15] G. Cooper and C. McGillem, *Probablistic Methods of Signal and System Analysis* (CBS College, New York, 1986), 2nd. ed., p. 245.