NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

ACOUSTIC IMAGING OF ULTRASONIC WAVE PROPAGATION

by

Benito E. Baylosis

December 1994

Thesis Advisor: John P. Powers

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE December 1994	3. REPORT TYPE AND DATES COVERED **Master's Thesis**

4. TITLE AND SUBTITLE Acoustic Imaging of Ultrasonic Wave Propagation	5. FUNDING NUMBERS
6. AUTHOR(S) BAYLOSIS, Benito E.	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)* A pulsed ultrasonic collection facility was used to study the accuracy of a previous program which models pulsed ultrasonic wave propagation. The previous program modeling was initially reviewed for its validity and found to have a scaling factor error. This error was corrected before a comparison of the theoretical and the experimental pulsed response was conducted. The excitation studied was a circular piston impulse input. For proper comparison, the acoustic potential produced by the program modeling (at a given distance from the acoustic source) was expressed in terms of acoustic pressure. Two separation distances were used to compare the output produced by the theoretical modeling and the measured experimental response. A general comparison of the experimental and the theoretical pulsed response appears to be in good agreement. The MATLAB program was used to perform all necessary mathematical computations and manipulations to produce a graphical representation of the collected data. The graphics program, AXUM, was used to compare the results graphically.

14. SUBJECT TERMS Green's function, spatial impulse response, diffraction, ultrasonic propagation, potential response, pressure response.	15. NUMBER OF PAGES 113
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited.

ACOUSTIC IMAGING OF ULTRASONIC
WAVE PROPAGATION

Benito E. Baylosis
Lieutenant, United States Navy
B.S.ET, Old Dominion University, 1987

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1994

Author: Benito E. Baylosis
Benito E. Baylosis

Approved by: John P. Powers
John P. Powers, Thesis Advisor

Ron J. Pieper
Ron J. Pieper, Thesis Second Reader

Charles W. Therrien (for)
Michael A. Morgan, Chairman
Department of Electrical and Computer Engineering

ABSTRACT

A pulsed ultrasonic collection facility was used to study the accuracy of a previous program which models pulsed ultrasonic wave propagation. The previous program modeling was initially reviewed for its validity and found to have a scaling factor error. This error was corrected before a comparison of the theoretical and the experimental pulsed response was conducted. The excitation studied was a circular piston impulse input. For proper comparison, the acoustic potential produced by the program modeling (at a given distance from the acoustic source) was expressed in terms of acoustic pressure. Two separation distances were used to compare the output produced by the theoretical modeling and the measured experimental response. A general comparison of the experimental and the theoretical pulsed response appears to be in good agreement. The MATLAB program was used to perform all necessary mathematical computations and manipulations to produce a graphical representation of the collected data. The graphics program, AXUM, was used to compare the results graphically.

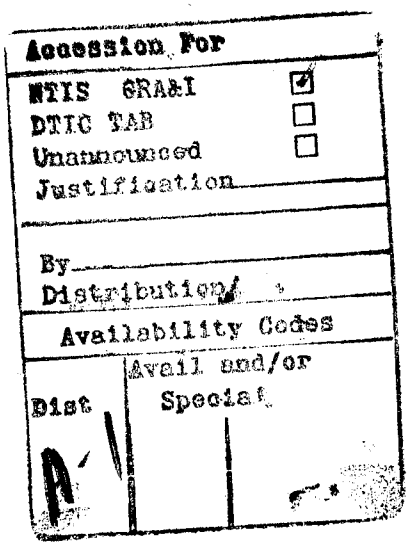


TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

The angular spectrum approach or Fourier decomposition is one of the more powerful methods of understanding transducer field analysis [Ref. 1]. Another approach to solving a diffraction field, the Rayleigh-Sommerfeld diffraction formula, uses the Green's function and the modified Kirchhoffs's theory [Ref. 2]. However, because of the inconsistencies found in this theory, the angular spectrum approach is often the preferred method of propagation modeling. Another advantage of angular spectrum approach over the Rayleigh-Sommerfeld theory is the faster computational speed due to a use of the fast Fourier transform algorithm (FFT). Because recent applications of ultrasound uses a pulsed signal more frequently [Ref. 3] than a continuous signal, this thesis focused mainly on the pulsed response of a transducer. A previously developed computer program which predicts and models the propagation wave diffraction at a given distance was available (but also contained a mathematical scaling error). The initial goal of this thesis was to examine and to correct the existing programs developed by Reid [Ref. 4] so that the results were consistent with the associated mathematics in the works of Stephanishen [Ref. 3]. Additionally, a pulsed ultrasonic chamber was available for collection of actual waveforms. This thesis modified the set-up of the ultrasonic collection chamber to increase the signal-to-noise ratio of the collected data to facilitate data processing. A data averager program was used on each pulsed response to further increase the signal-to-noise ratio of the collected signal. The collected waveforms can be reconstructed to form a

diffraction wave. The ultimate goal of this thesis was to compare the collected data wavefronts and the theoretical model wavefronts in order to confirm the program's validity and accuracy. The comparison between the collected data and the theoretical model was facilitated by loading the data and results generated by MATLAB into the graphics program, AXUM. AXUM commands are thoroughly explained in Ref 5.

The theory used by Reid [Ref. 4] on his computer modeling of diffraction wave is based on the linear system approach of Guyomar and Powers [Refs. 6, 7, and 8]. The linear system approach is derived from the spatial filter approach solved by Stephanishen [Refs. 9, 10, and 11] and reviewed by Harris [Ref. 12]. The difference between Stephanishen and Guyomar/Powers theory is that the linear theory brings out the importance of a total impulse response and its equivalent Green's function. The linear system approach of Guyomar and Powers shows the propagation of an impulsively excited source as a time-varying spatial filter applied to a spatial spectrum of the input excitation [Ref. 9]. As noted earlier, this technique enabled Reid to use the fast Fourier transform (FFT) algorithm, thereby reducing the processing time drastically. Another advantage of this technique includes the validity of the modeling in the near-field of acoustical process as described in Ref. 1. However, one major restriction of the linear system approach is the requirement that the source plane and the observation plane must be parallel.

The collection facility developed by Gatchell [Ref. 13] was designed to collect data to test the angular spectrum method. It consists of a water tank, a scanning device, a pulse generator, a waveform digitizer, and a 486-DX 33-MHz personal computer. The personal computer controls the hydrophone positioning system, fires the pulse wave

generator, collects data, and stores the data at the designated directory with the aid of its controlling software, LABVIEW. The mechanics of LABVIEW are explained fully in Gatchell's work [Ref. 13]. The set-up shown in Gatchell's [Ref. 13] work was slightly modified in this thesis to increase the signal-to-noise ratio of the collected waveform to facilitate data processing.

Chapter II of this thesis consists of a problem description for the sound wave propagation modeling which includes the mathematical inconsistency of the program modeling. It also deals with a problem description for the pulsed ultrasonic data collection facility developed by Gatchell [Ref. 13] and the modification required to increase the collected signal-to-noise ratio. Chapter III consists of the experimental techniques and procedures. This chapter also covers the data conversion program using C++ to convert the collected data from a binary text format to a MATLAB format. The second part of this chapter consists of a data averager program using MATLAB to increase the signal-to-noise ratio. Chapter IV contains the experimental results. This chapter contains several comparisons between the theoretical wavefronts and the actual wavefronts at two different separation distances. All graphs data were generated by the program MATLAB and presented using the graphical program, AXUM. Following Chapter V, which is the thesis summary and conclusions, are the Appendices. Appendix A, C, and E give the detailed explanation of the programs PREDICT.M, CONVOL.M, and AVG.M, respectively. Following each detailed explanations are the source code for each program located in Appendices B, D, and F, respectively. Appendix G contain the source code for

the data conversion program, DATO2.CPP, which was originally developed by Ray van deVeire of the Naval Postgraduate School.

II. PROBLEM DESCRIPTION

Before a proper comparison of a theoretical pulsed response and an actual wavefront pulsed response can be accomplished, a correction must be made to an error found in Reid's work [Ref. 4]. Although Reid's theoretical modeling results were consistent in shapes and trends, their impulse response elapsed time was inaccurate and inconsistent with the works of Guyomar and Powers [Refs. 6, 7, and 8]. Additionally, hardware set-up modifications were required in Gatchell's work [Ref. 13] in order to use the required receiver/transducer. The first section of this chapter deals with the theoretical modeling of pulse ultrasonic waves done by Guyomar and Powers [Refs. 6, 7, and 8] which leads to the correction made on Reid's work of propagation modeling. The second section deals with the impulse response verification to ensure that the elapsed time was consistent with the predicted impulse response produced by Stephanishen [Refs. 3, 9, 10, and 11]. The latter part of the second section deals with program modeling modifications. These modifications include the Green's function modeling, the reduced array techniques, and some miscellaneous modifications to ensure that the program was consistent with the latest version of MATLAB. The third section of this chapter deals with handling the theoretical non-impulsive time input which was required to properly compare the theoretical results with the data from the actual experiment. The fourth section of this chapter is a brief description of the pulsed ultrasonic data collection facility produced by Gatchell [Ref. 13]. The last section of this chapter deals with the modifications made on

the hardware area of the ultrasonic data collection facility in order to facilitate the use of the required receiver/transducer and to increase the signal-to-noise ratio of the collected data. The theory presented in the first section is a repeat of works done by Reid and Guyomar/Powers for clarity and for correction of the mathematical error. [Refs. 4, 6, 7, and 8]

A. ULTRASONIC WAVE MODELING

The problem geometry is shown in Fig. 1. Assume that the medium between the source plane and the observation plane is linear, homogeneous, and lossless. With a z-directed velocity $v_z(x,y,0,t)$ over a baffled region of an arbitrary shape at $z = 0$ plane, the program's goal is to find the acoustic velocity potential $\phi(x,y,z,t)$ at some point in the positive-z half-space. Assume a separable z-directed velocity of the form

$$v_z(x,y,0,t) = T(t)s(x,y). \tag{1}$$

According to the works of Guyomar/Powers, the relation between the acoustic velocity potential and the output z-velocity is

$$\phi(x,y,z,t) = T(t) \overset{*}{_t} p(x,y,z,t) \tag{2}$$

where $\overset{*}{_t}$ indicates convolution in time. The "spatial impulse response" $p(x,y,z,t)$ is the velocity potential that results when the source is excited by a z-velocity of the form $s(x,y)\delta(\tau)$.

Equation 2 shows that "acoustic velocity potential" $\phi(x,y,z,t)$ can be solved given the spatial impulse response $p(x,y,z,t)$. This spatial impulse response can be solved using linear system theory. [Refs. 6, 7, and 8]

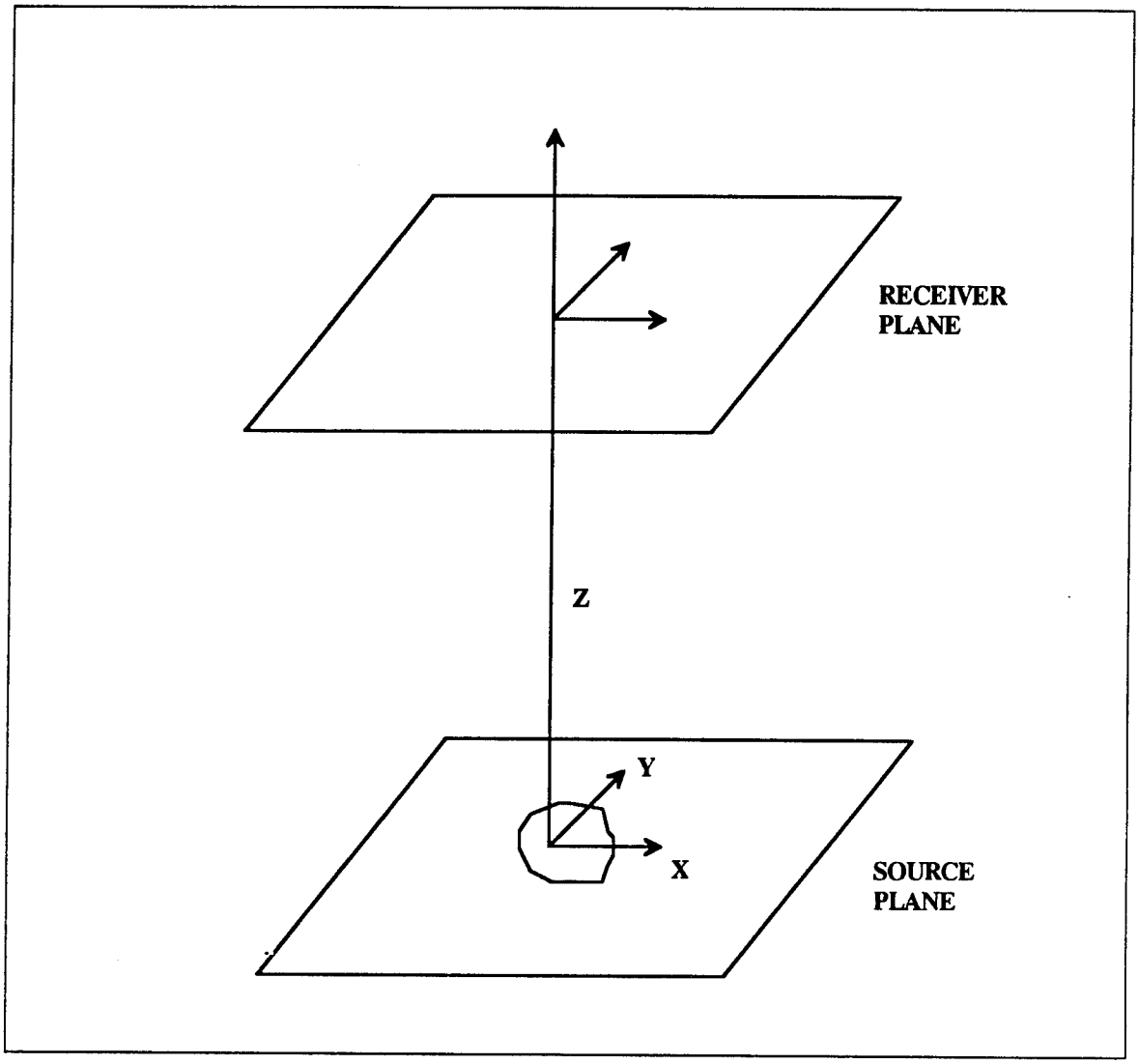


Figure 1. Source/receiver geometry.

Using linear system theory, the "total impulse response" $h(x,y,z,t)$ of the system is the result of an impulsive input of the form $\delta(x,y)\delta(t)$, represented in Fig. 2a. The system that is linear and space-invariant, as in a linearly homogeneous medium, with a temporal impulse input of the form $s(x,y)\delta(t)$ will produce the "spatial impulse response" $p(x,y,z,t)$ (as in Fig. 2b). The spatial impulse response is related to the total impulse response, $h(x,y,z,t)$, and can be expressed as

$$p(x,y,z,t) = s(x,y) \overset{*}{}_{x} \overset{*}{}_{y} h(x,y,t). \tag{3}$$

From Eqs. 2 and 3, it can be deduced that

$$\phi(x,,z,t) = T(t)^{*}_{t} s(x,y) \overset{*}{}_{x} \overset{*}{}_{y} h(x,,y,t). \tag{4}$$

Combining Eq. 4 with Fig. 2a results in a system shown in Fig. 2c. It follows from Eq. 2 that the acoustic velocity potential $\phi(x,y,z,t)$ of the system depends largely on the spatial impulse response $p(x,y,z,t)$ of the system. From Eq. 3, the spatial impulse response $p(x,y,z,t)$ is also dependent on the total impulse response $h(x,y,z)$. Figure 2a shows that the total impulse response $h(x,y,z)$ is the propagation field resulting from a source of the form $\delta(x,y)\delta(\tau)$ that solves the wave equation and satisfies the boundary condition. This solution is equivalent to the Green's function.

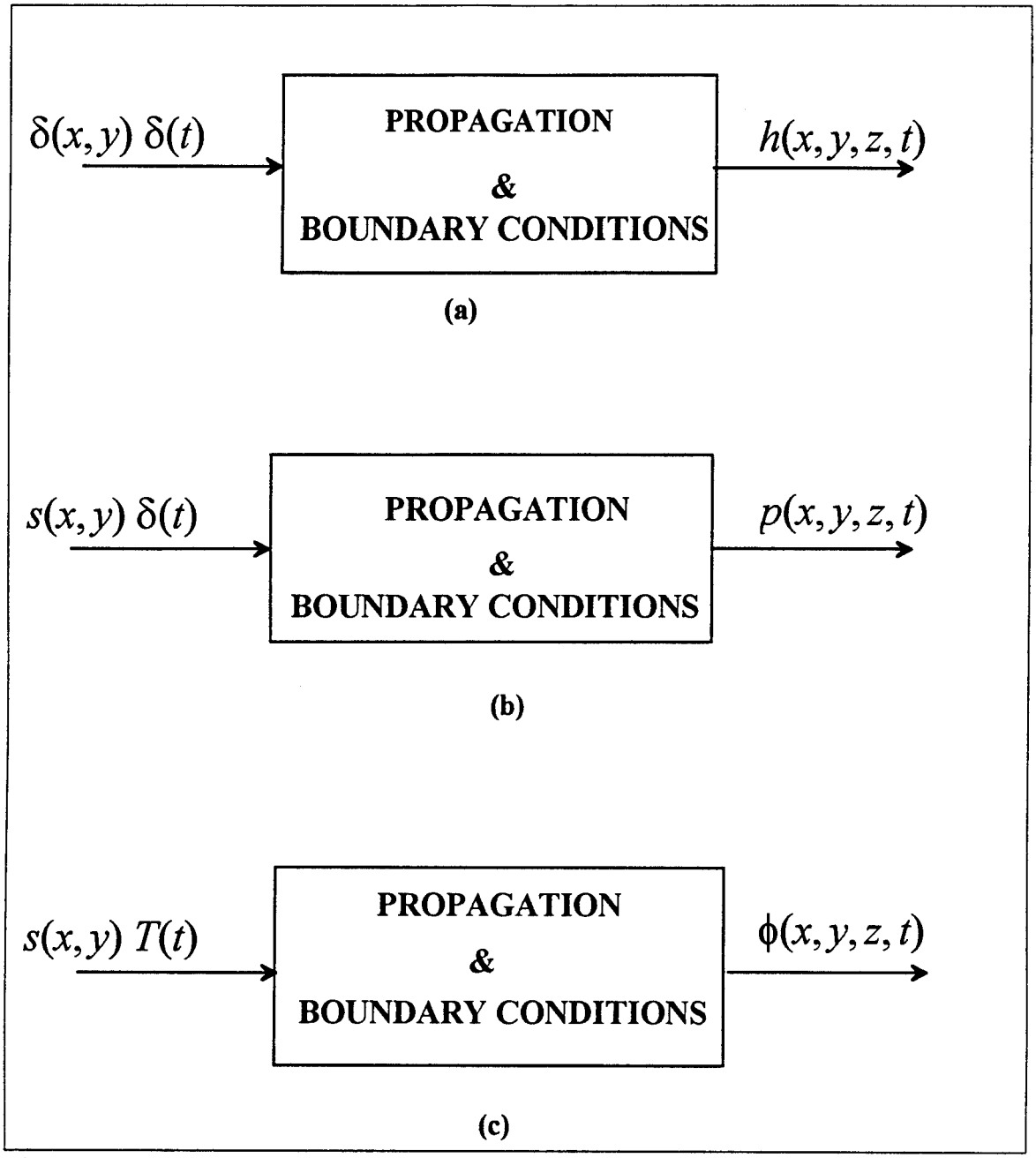


Figure 2. Block diagram of the (a) propagation impulse response, (b) spatial impulse response, and (c) general solution.

The Green's function for the lossless media for rigid baffle is of the form

$$g(x,y,z,t) = \frac{\delta(ct-R)}{2\pi R} \tag{5}$$

where $R = \sqrt{x^2 + y^2 + z^2}$ and c is the speed of sound in the medium. The two-dimensional fourier transform of the Green's function $g(x,y,z,t)$ is of the form

$$g(f_x,f_y,z,t) = 2J_o\left(\rho\sqrt{c^2t^2 - z^2}\right)H(ct - z). \tag{6}$$

Since the solution of the system is equivalent to the Green's function as stated earlier, the total impulse response $h(x,y,z,t)$ can be expressed as

$$h(x,y,z,t) = \frac{\delta(ct-R)}{2\pi R}. \tag{6a}$$

The two-dimensional Fourier transform of the total impulse response is of the form

$$\Im\{h(x,y,z,t)\} = 2J_o\left(\rho\sqrt{c^2t^2 - z^2}\right)H(ct - z) \tag{6b}$$

where $\rho = 2\pi\sqrt{f_x^2 + f_y^2}$ and $H(t)$ is the step function. Substituting Eq. 6a for the total impulse response $h(x,y,z,t)$ into Eq. 3 yields

$$p(x,y,z,t) \quad = \quad 2s(x,y) \overset{*}{}_{x} \overset{*}{}_{y} \delta[t - (R/c)] / 2\pi R. \tag{7}$$

The double spatial convolution of Eq. 6 is difficult to implement computationally. An alternative way to solve the problem is by working in the spatial frequency domain which converts the convolution operator to a multiplication operator. As shown in Fig. 3, Eq. 3 then becomes

$$\tilde{p}(f_x,f_y,z,t) \quad = \quad \tilde{s}(f_x,f_y)\,\tilde{h}(f_x,f_y,t) \tag{8}$$

where $\tilde{s}(f_x,f_y)$ is the source input spatial transform and $\tilde{h}(f_x,f_y)$ is the propagation transfer function. After performing the operation of Eq. 8, the last step in finding the final

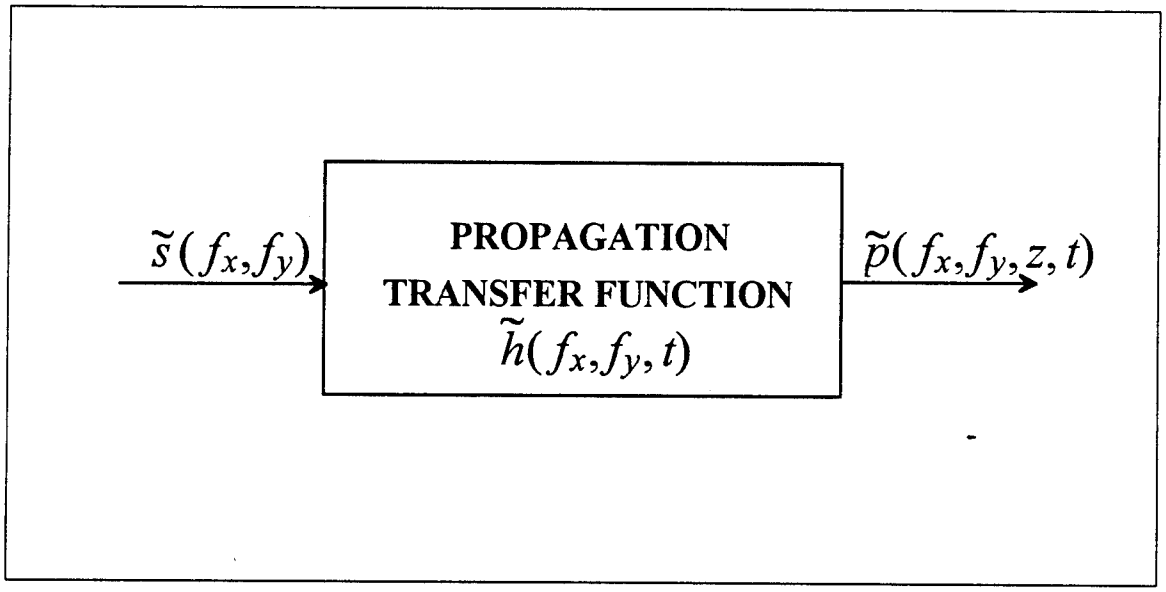


Figure 3. Block diagram of the propagation transfer function.

solution reduced to taking the inverse 2-D spatial Fourier transform of the spatial response $\tilde{\rho}$. From Eq. 6b, it can be shown that the propagation transfer function is a Bessel function of the first kind of zero order, which means that the high spatial frequencies are relatively smaller in magnitude than the lower spatial frequencies as shown in Fig. 4. [Refs. 4, 6, 7, and 8]

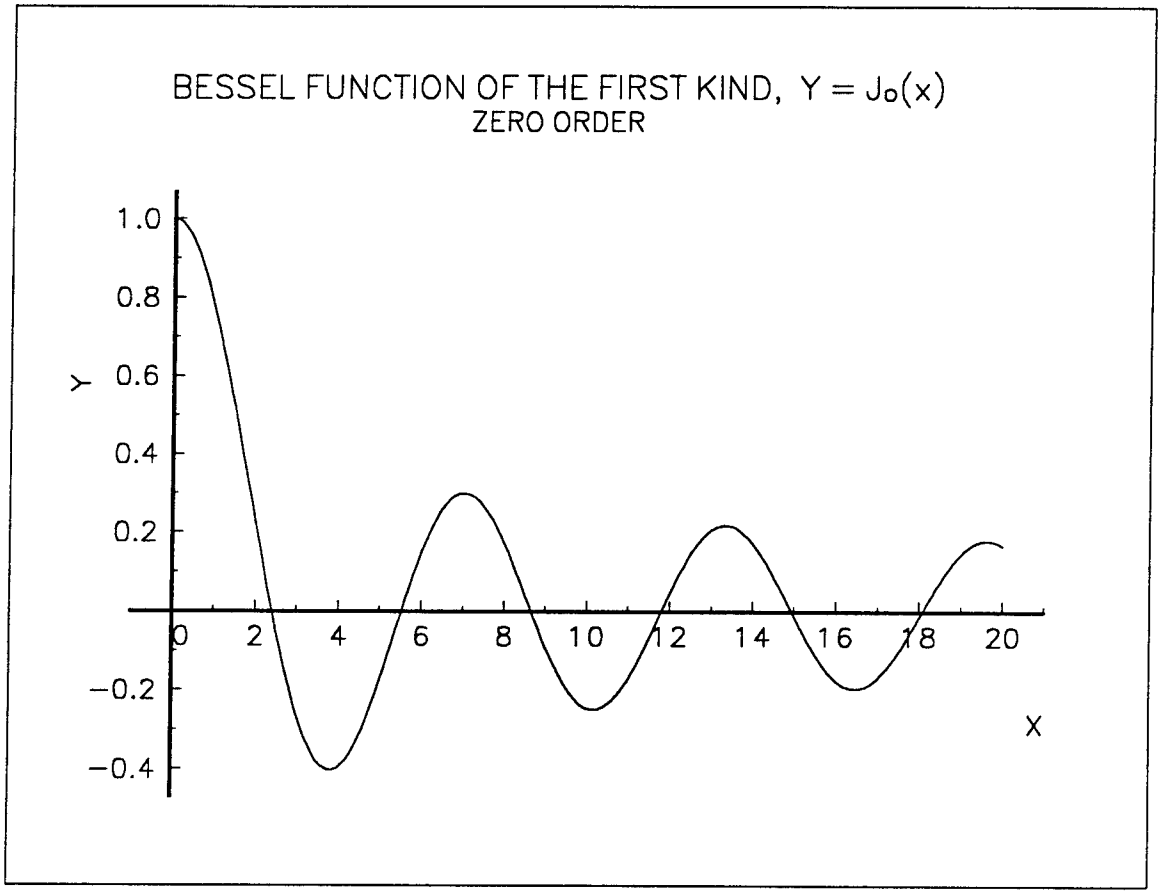


Figure 4. Bessel function of the first kind, zero order. $y = J_0(x)$.

Reid's work of microcomputer simulation [Ref. 4] was purely for an input that was temporally impulsive and spatially arbitrary (i.e., $p(x,y,z,t)$). Therefore, a two-dimensional inverse-Fourier transform on Eq. 7 gave the impulse response of the system. Reid's thesis generated the propagation transfer function in the program module AC_FIL.M for each time slice. The program module AC_PROP.M generated the input excitation chosen by the user through the used of the MATLAB function tool. It then recovers the appropriate filter generated by the AC_FIL.M and multiplies it with the input excitation for each time slice. Finally, AC_PROP.M performs a two-dimensional inverse-Fourier transform for each time slice to produce the diffracted wave for each time slice. The middle row of the diffracted wave for each time slice are then extracted and combined into a single matrix to represent the center row of the impulse response as a function of time, $p(x,0,z,t)$. The detailed implementation of Reid's program propagation modeling is explained fully in Ref. 4.

Equation 2 is the acoustic velocity potential $\phi(x,y,z,t)$ of the system and is repeated below for convenience as

$$\phi(x,y,z,t) = T(t) \overset{*}{_t} p(x,y,z,t). \tag{9}$$

However, the acoustic waveforms generated by the acoustic source was sensed by the acoustic receiver as a pressure waveform. Therefore, to properly compare the theoretical waveform with the measured waveform, the term $\phi(x,y,z,t)$ must be expressed in terms of

pressure. According to the works of Guyomar and Powers [Ref. 7], the acoustic pressure can be expressed as the time derivative of the acoustic velocity potential $\phi(x,y,z,t)$ and can be expressed as

$$\psi(x,y,z,t) = \rho_o \frac{d\phi}{dt} \tag{10}$$

where $\psi(x,y,z,t)$ is the acoustic pressure and ρ_o is the medium's density (1000 kg/m^3 for water). Combining Eq. 2 with Eq. 10 gives

$$\psi(x,y,z,t) = \rho_o \frac{d}{dt}[T(t) * p(x,y,z,t)] \tag{11}$$

where the $T(t)$ is the time-varying excitation function. Since the results of the computer simulation are normalized to maximum values, the scaling factor of ρ_o was dropped in this thesis for convenience. Using the associative and the commutative properties of the Fourier transform [Ref. 14], Eq. 11 can also be written as

$$\psi(x,y,z,t) = \rho_o T(t) * \frac{d}{dt} p(x,y,z,t). \tag{12}$$

Equation 12 is the theoretical pressure for response for an arbitrary time excitation input. The derivative of the spatial impulse response $p(x,y,z,t)$ was performed in the modified

program of AC_PROP.M. The convolution of Eq. 12 was performed in the program CONVOL.M which is detailed in Appendix D and explained in Appendix C.

B. VERIFICATION OF IMPULSE RESPONSE AND PROGRAM MODELING MODIFICATIONS

1. Impulse Response Approach

As stated earlier, the propagation transfer function can be modeled as the Green's function in Eq. 6 and is repeated here for convenience as

$$\Im\{h(x,y,z,t)\} = 2J_o\left(\rho\sqrt{c^2t^2 - z^2}\right)H(ct - z) \qquad (13)$$

where $\rho = 2\pi\sqrt{f_x^2 + f_y^2}$, c is the speed of sound in the medium, t is the time in seconds, and z is the longitudinal distance of the transmitter and the receiver. Reid's [Ref. 4] mathematical error was made in this stage. Reid used $\rho = \sqrt{f_x^2 + f_y^2}$ for the variable ρ in Eq. 13 as opposed to $\rho = 2\pi\sqrt{f_x^2 + f_y^2}$ as stated in Ref. 8. As a result, Reid's program impulse response was theoretically incorrect. The method used to verify the inaccuracy of Reid's work was derived from Stephanishen's work [Refs. 3, 9, 10, and 11]. The problem geometry is shown in Fig. 5. The normalized impulse response of a circular piston in a planar baffle, as shown in Fig. 5, can be solved by the equations below. There are two basic set of equations. The first set of equations deals with the points inside the active region of the transmitter where $r/a < 1$ while the second set of equations deals with the point outside the active region of the transmitter, where $r/a > 1$.

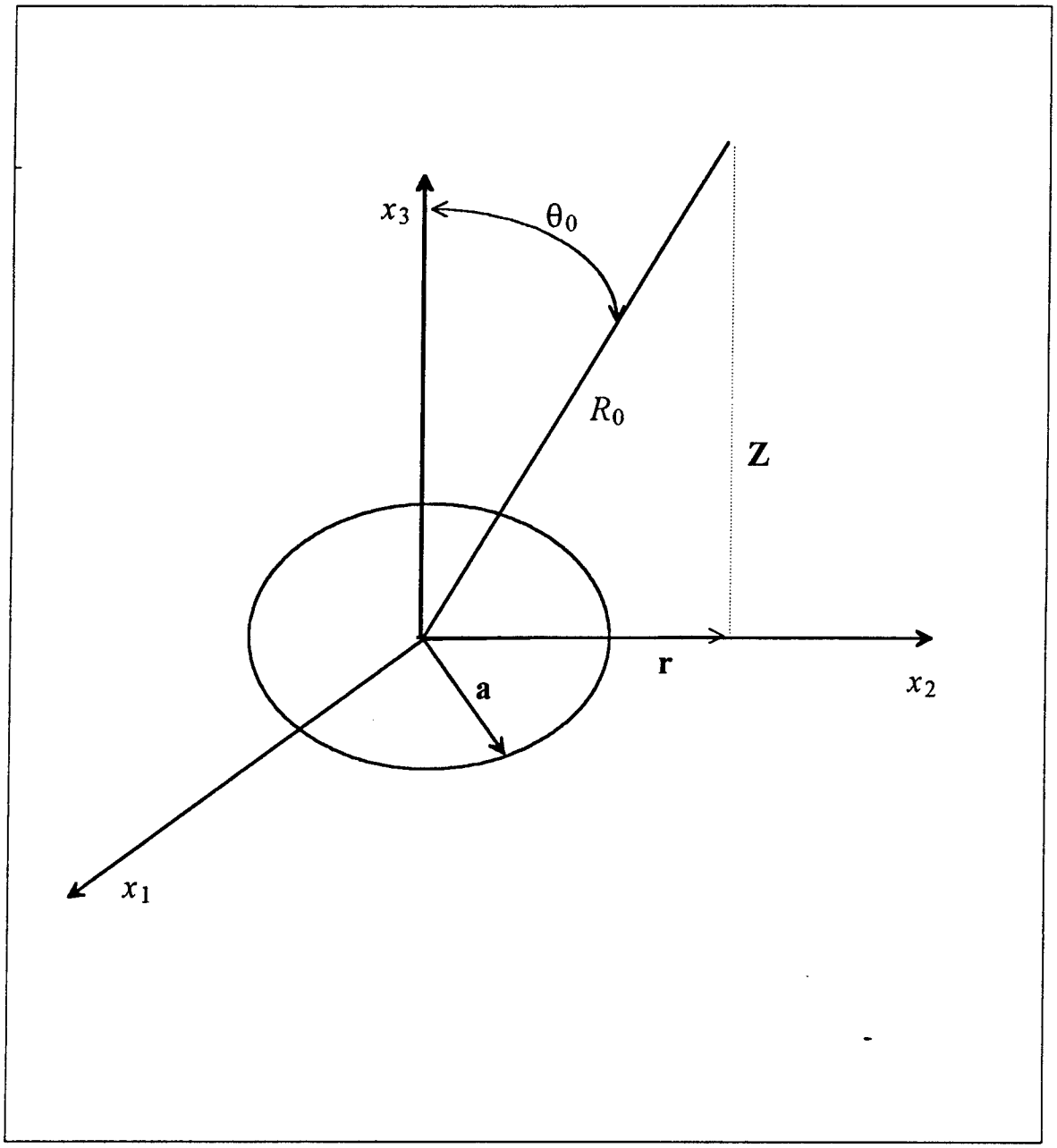


Figure 5. A circular piston in a planar baffle.

When $r/a < 1$, the normalized impulsed response can be expressed as:

$$p(x',\tau) = \left\{ \begin{array}{ll} 0, & \tau < z/a \\ 1, & z/a \le \tau \le R_1 \\ \frac{1}{\pi}\cos^{-}\left\{ \dfrac{\tau^2 - \left(\frac{z}{a}\right)^2 + \left(\frac{r}{a}\right) - 1}{2\frac{r}{a}\left[\tau^2 - \left(\frac{z}{a}\right)\right]^{0.5}} \right\}, & R_1 \le \tau \le R_2 \\ 0, & R_2 < \tau \end{array} \right\}, \quad \textbf{(14)}$$

where $R_1 = \sqrt{(z/a)^2 + ((r/a) - 1)^2}$, $R_2 = \sqrt{(z/a)^2 + ((r/a) + 1)^2}$, $\tau = ct/a$, $x' = x/a$, r is the radial distance, z is the separation distance between the transmitter and the receiver, and $p(x', \tau)$ is the normalized impulsed response. When $r/a > 1$, the normalized impulsed response can be expressed as:

$$p(x',\tau) = \left\{ \begin{array}{ll} 0, & \tau < R_1 \\ \frac{1}{\pi}\cos^{-}\left\{ \dfrac{\tau^2 - \left(\frac{z}{a}\right)^2 + \left(\frac{r}{a}\right)^2 - 1}{2\frac{r}{a}\left[\tau^2 - \left(\frac{z}{a}\right)^2\right]^{0.5}} \right\}, & R_1 \le \tau \le R_2 \\ 0, & R_2 < \tau \end{array} \right\}. \quad \textbf{(15)}$$

In the far-field region which can be defined as $z_o/a >> 1$, Eq. 14 reduces to

$$p(z_o/a, 0, \tau) = (\rho c a \pi / 2 z_o)\, \delta(\tau - z_o/a) \quad \textbf{(16)}$$

where z_o is the separation between the transmitter and the receiver and ρc is the specific acoustic resistance of the medium. [Refs. 3, 9, 10, and 11]

Equations 14 and 15 were implemented in a MATLAB program called PREDICT.M. The results are shown in Fig. 6, which is based on the circular piston with a receiver separation distance of 0.1 m. The three positions examined (dependent on the radial distance of the focal point) are: a point located at the center of the active region ($r=0$), a point located off center but still inside the active region ($r = 0.5a$), and a third point located outside the active region ($r = 2a$).

Figure 6a, which represents the center ($r = 0$), has an impulse response of a rectangle with time duration equal to the difference in the propagation of sound from the center and from the edges of the piston. Figure 6b, which represents a point off center but still inside the active region ($r = 0.5a$), has an impulse response similar to Fig. 6a for a period of time then decreases monotonically. The constant response part of Fig. 6b has a time duration equal to the difference in propagation of sound from the element directly in front of the receiver and from the edge closest to the receiver. The period of time in Fig. 6b where output decreases monotically corresponds to the time difference in the propagation of sound from the closest edge to the farthest edge. The time duration of Fig. 6b corresponds to the difference in propagation of sound from the closest edge and from the farthest edge. Finally, Fig. 6c, which represents the outside of the active region ($r=2a$), has an impulse response that is smaller in magnitude compared to the magnitude of Figs. 6a and 6b. Figure 6c also takes a nonsymmetrical shape. The results in Fig. 6 are consistent with the impulse response discussed in Ref. 11.

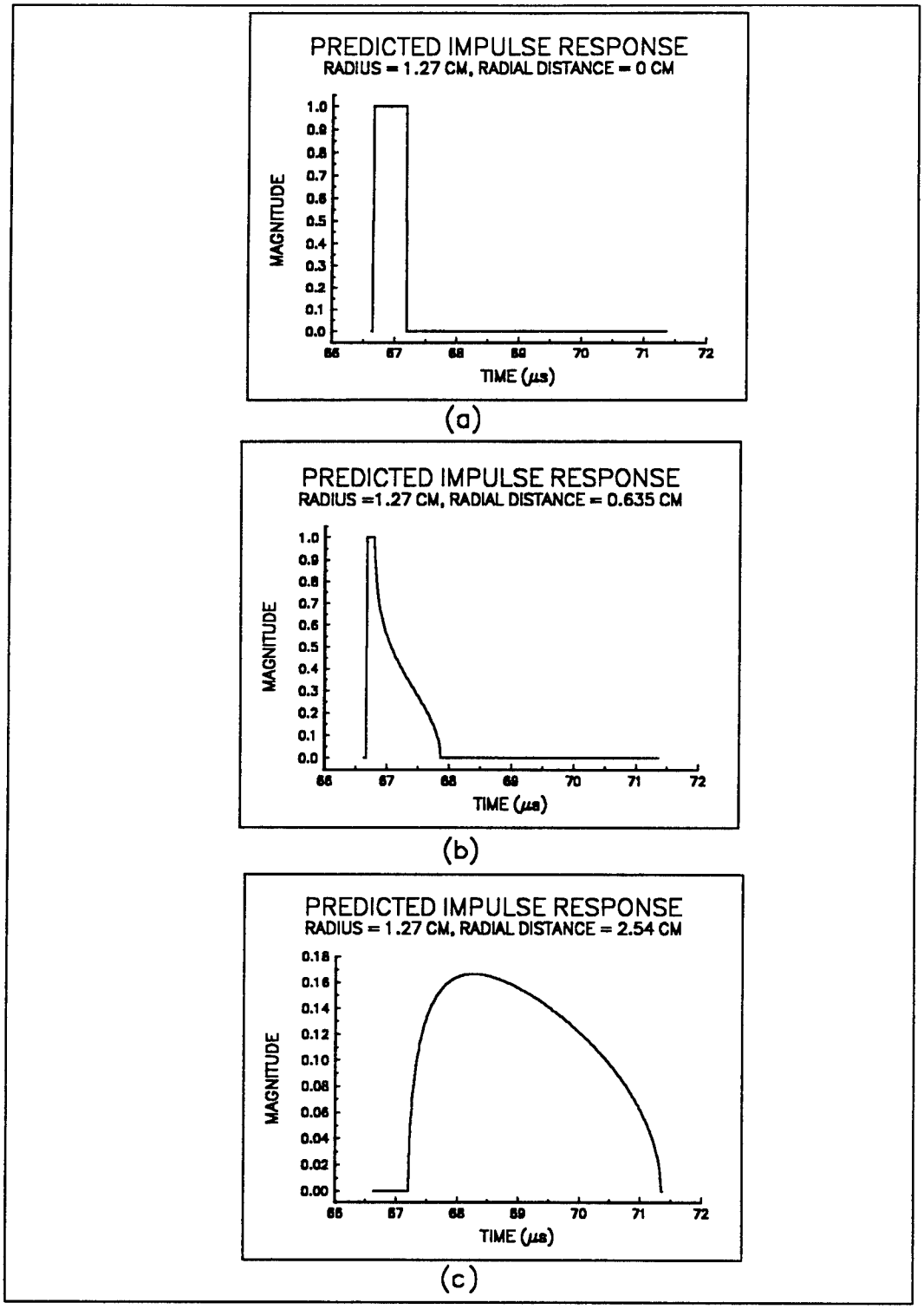


Figure 6. Three basic forms of an impulse response. (a) At the center ($r = 0$), (b) off center inside the active region ($r = a/2$) , and (c) outside the active region ($r = 2a$). Source radius is 1.27 cm and the separation distance is 0.1 m.

The comparison of an impulse response produced by Reid's program and the corrected program at the radial distance of $r = 0$, $r = 0.5a$, and $r = 2a$ are shown in Figs. 7, 8, and 9. To acquire the impulse response at the radial distance of $r = 0$, the center row must be extracted from the impulse response result matrix. The equation is

$$row = \frac{Z}{2} - \frac{r}{\Delta x} \tag{17}$$

where Z is the size of reduced base array ($N/2$), N is the size of base array, r is the size radial distance, and $\Delta x = x / N$. (The concept of reduced base array is explained in the next section). The summary of impulse response time duration are listed on Table I.

Radial Distance	$r = 0$	$r = 0.5a$	$r = 2a$
Predicted	0.60 μs	1.3 μs	~ 4.1 μs
Modified Program	0.60 μs	1.3 μs	~ 4.1 μs
Reid's Program	19.4 μs	39.4 μs	~ 85 μs

Table I. Summary of an impulse response time duration.

Comparison of the total impulse response results between Reid's program and the modified program at 0.1 m and 0.2 m separation distances are shown in Figs. 10 and 11.

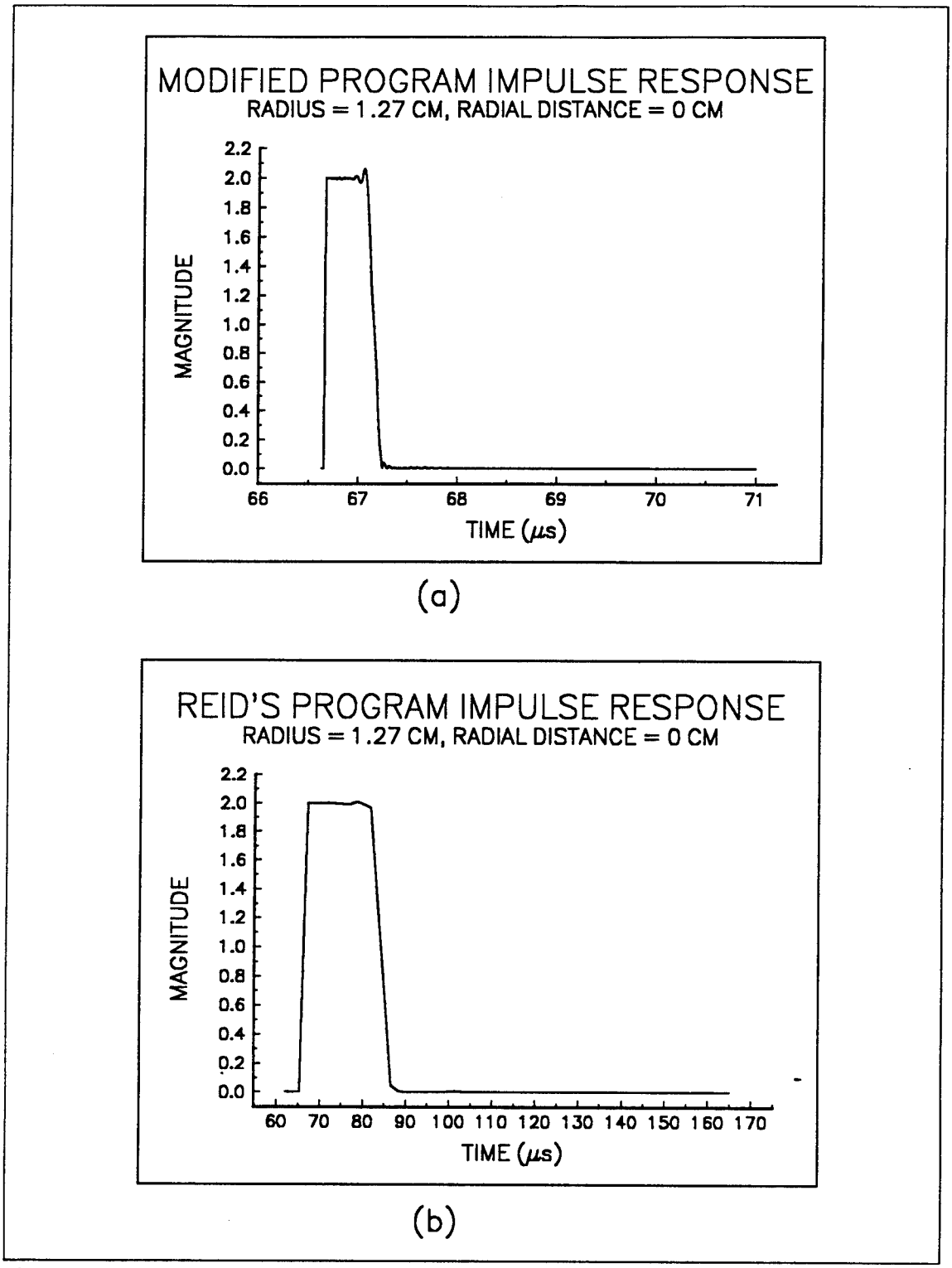


Figure 7. Comparison of an impulse response at radial distance $r = 0$ between (a) Reid's modified program and (b) Reid's program.

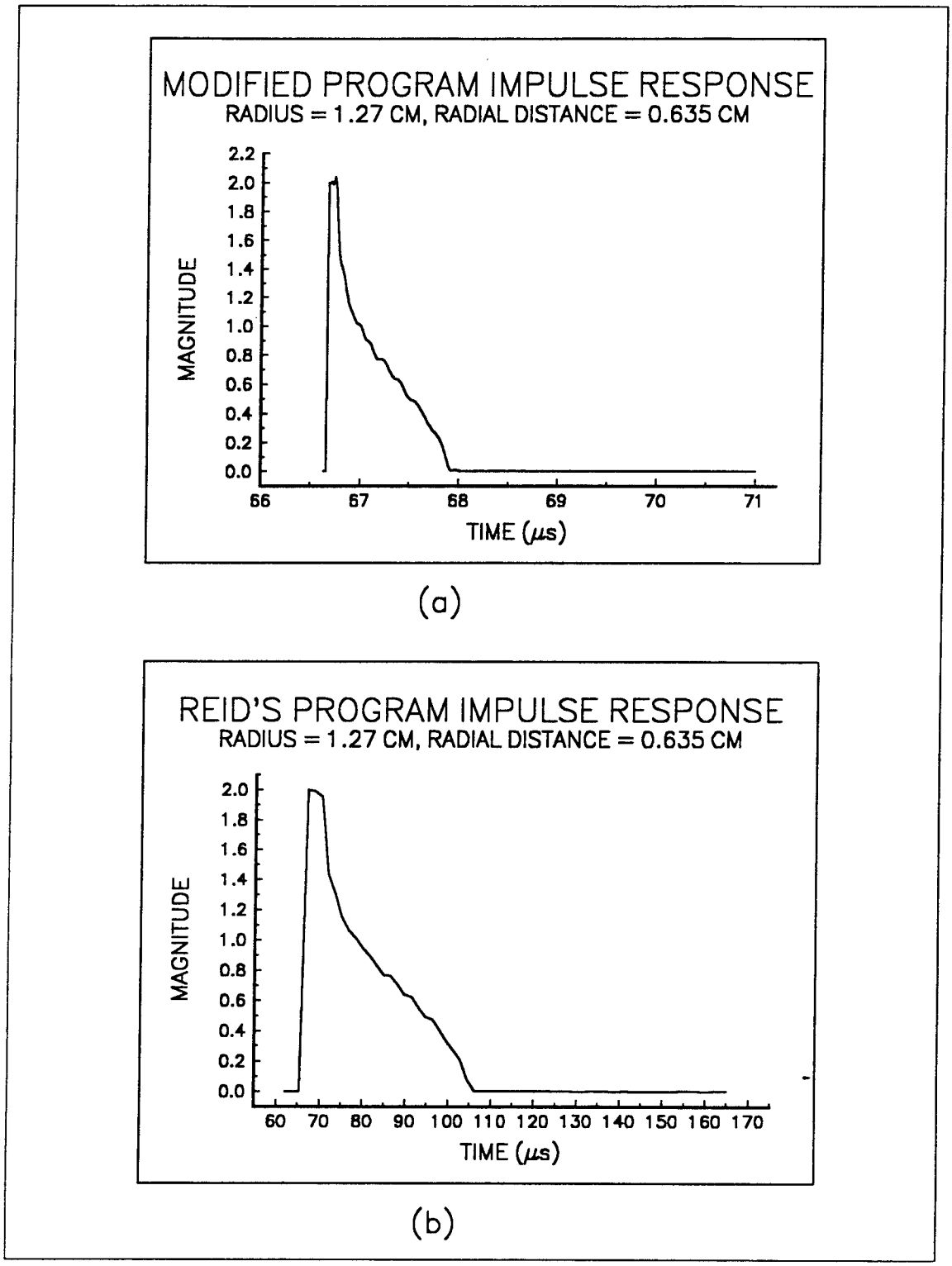
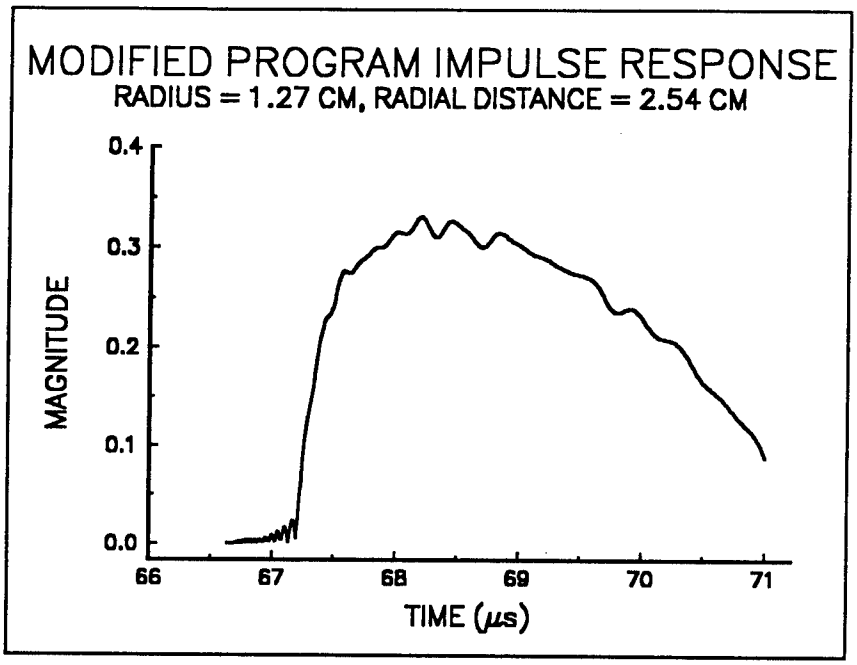
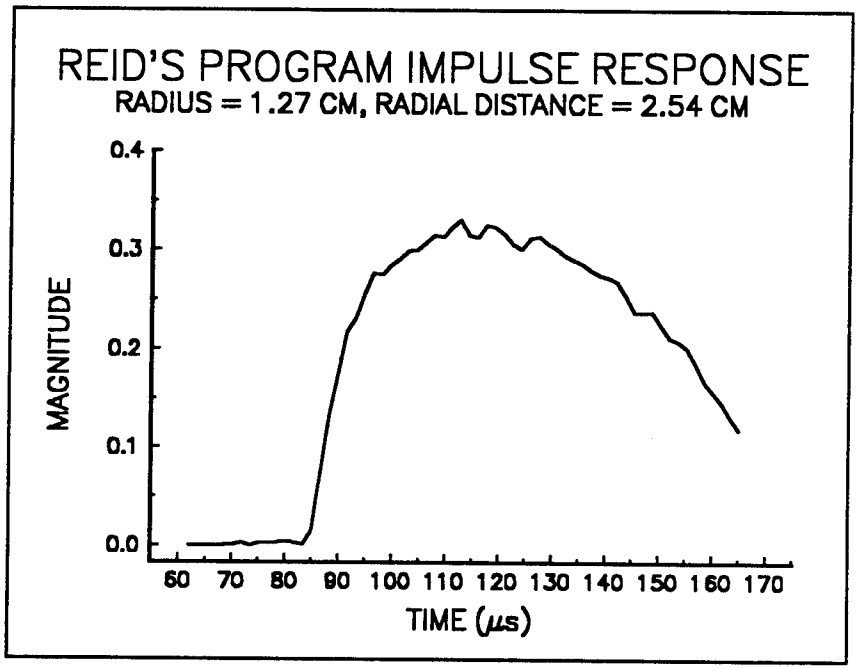


Figure 8. Comparison of an impulse response at radial distance $r = 0.5a$ between (a) Reid's modified program and (b) Reid's program.



(a)



(b)

Figure 9. Comparison of an impulse response at radial distance $r = 2a$ between (a) Reid's modified program and (b) Reid's program.

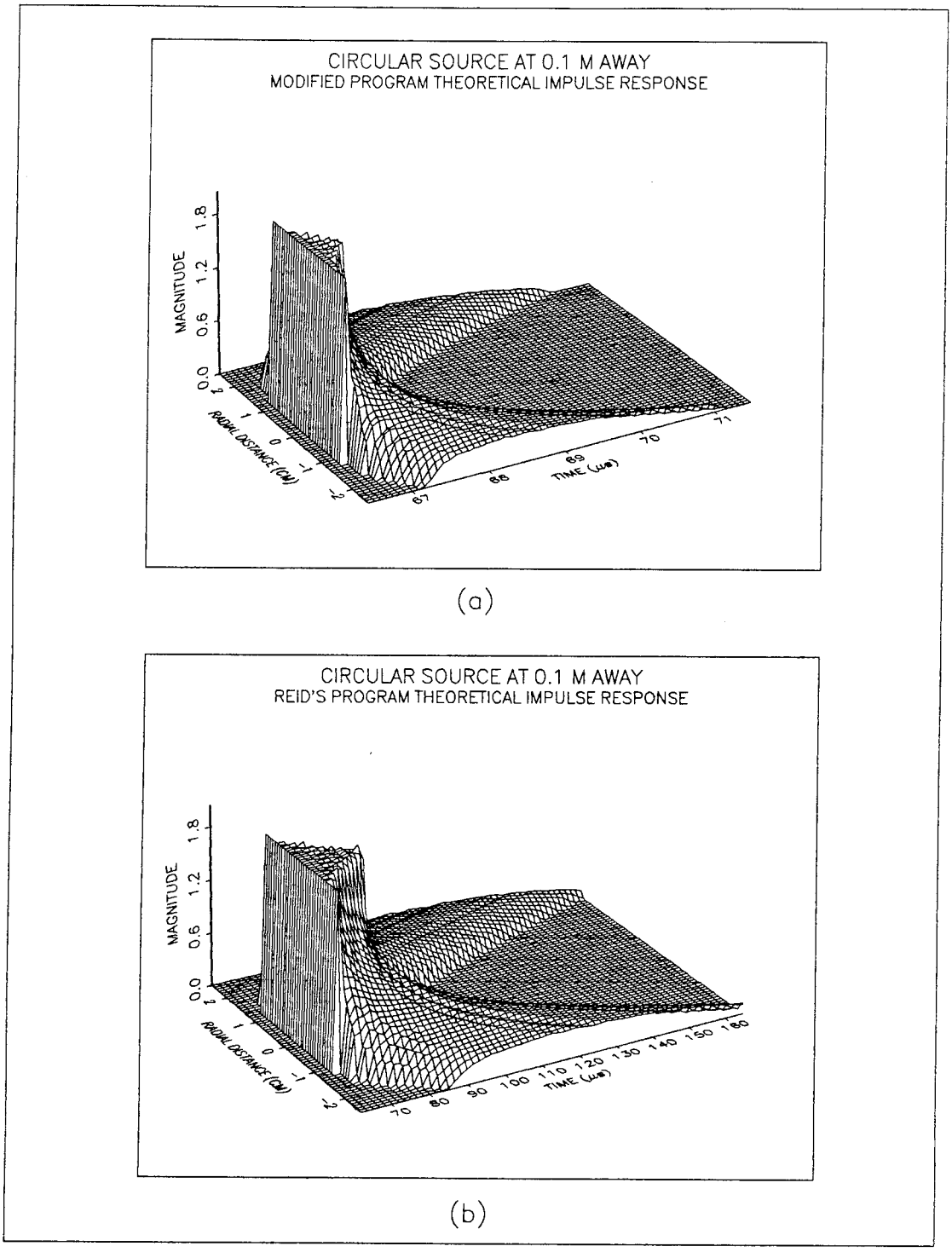


Figure 10. Comparison of an impulse response at z = 0.1 m between the (a) modified program and (b) Reid's program.

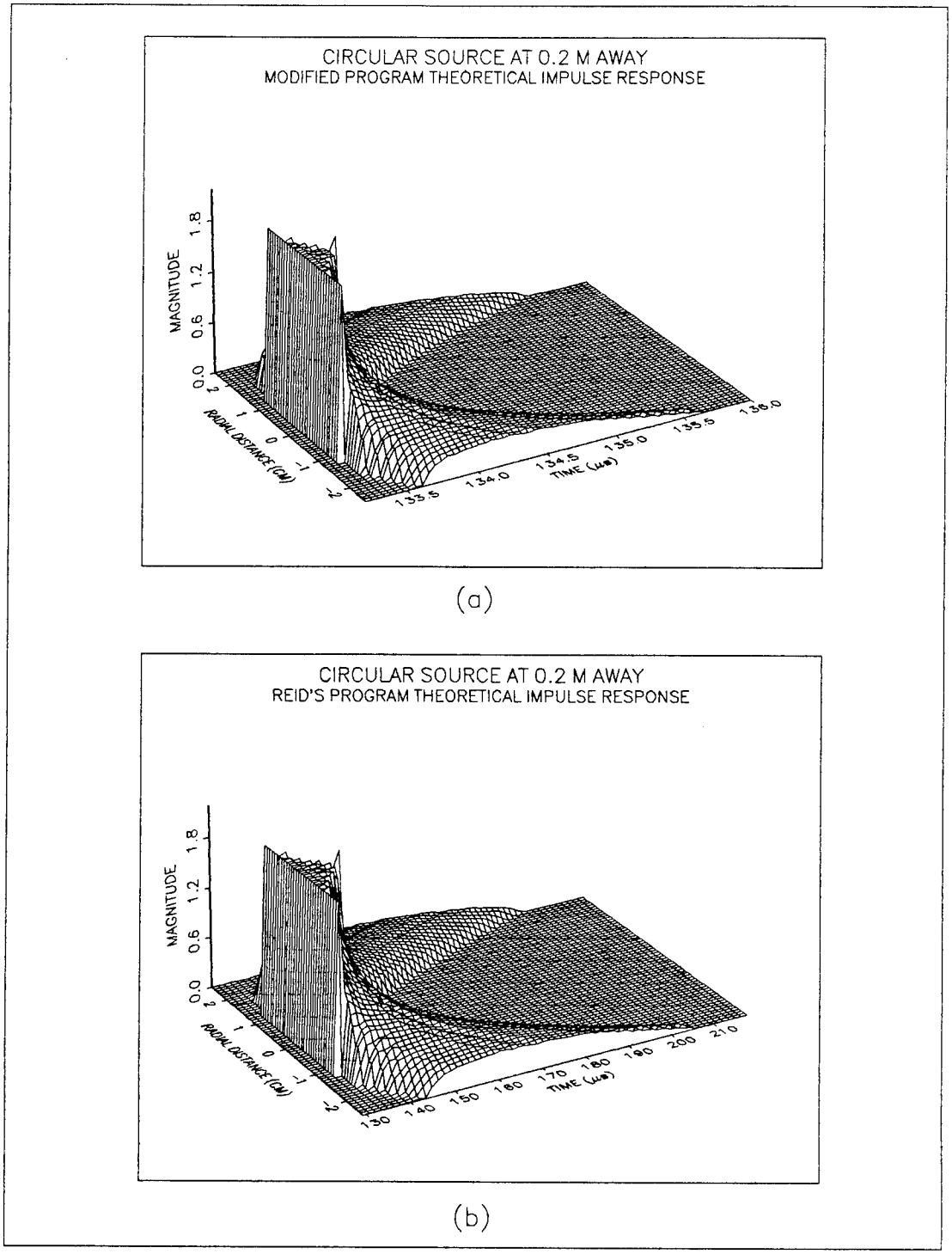


Figure 11. Comparison of an impulse response at z = 0.2 m between the (a) modified program and (b) Reid's program.

2. Program Modeling Modifications

a. Green's Function Modeling

The most important modification made on Reid's program is the modeling of the two-dimensional Fourier transform of the total impulse response which is equivalent to the two-dimensional Fourier transform of the Green's function. As shown in Eq. 13, the two-dimensional Fourier transform of the total impulse response is repeated here again for convenience as

$$\tilde{h} = 2J_o\left(\rho\sqrt{c^2t^2 - z^2}\right)H(ct - z) \qquad \textbf{(18)}$$

where $\rho = 2\pi\sqrt{f_x^2 + f_y^2}$. However, Reid's work ignored the multiplicative constant 2π in the equation for ρ. Since ρ is an argument in the Bessel function, ignoring it produces a mistake in time domain as shown in Table I.

b. Reduced Base Array

Another modification done on Reid's work deals with boundary conditions. The actual experiment was done in a tank much bigger than the observation area of the computations. As a result, the sound propagation continued without bounds beyond the observation area. However, the DFFT in Reid's program modeled the observation area as responding to a periodic array of sources. As a result, the impulse response near the boundary "bounced back" at the boundary due to aliasing, creating an unwanted humps as shown in Fig. 12. To overcome this problem, this thesis modeled the observation area

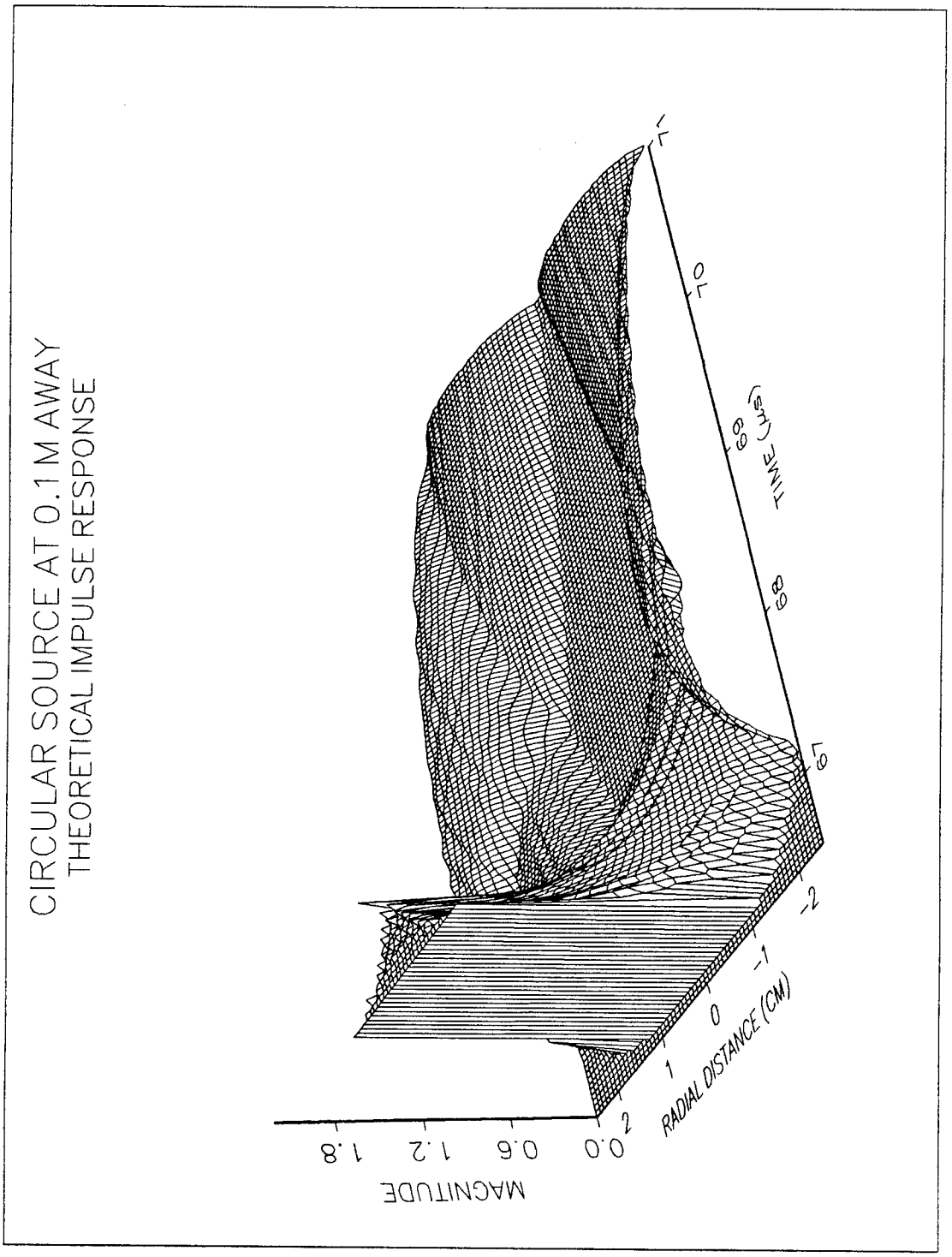


Figure 12. Impulse response without reduced base array techniques. (Note: waves behaving as if they were reflected by boundary.)

twice as large as the desired observation area. To extract the desired observation area without the effect of the boundaries, a reduced area centered at the middle of the modeled area was extracted as shown in Fig. 13. The impulse response corresponding to this reduced area is shown in Fig. 14.

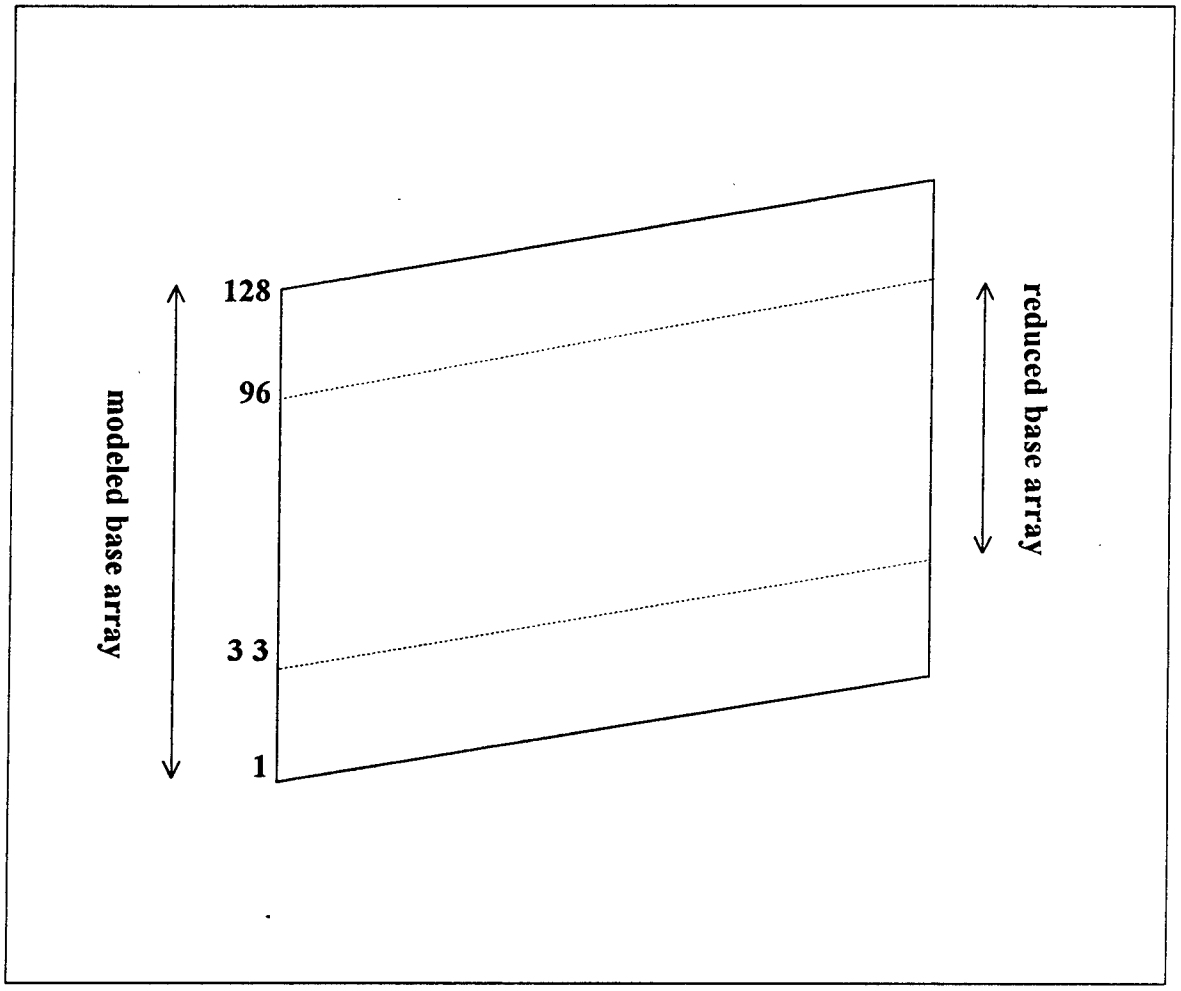


Figure 13. Reduced base array.

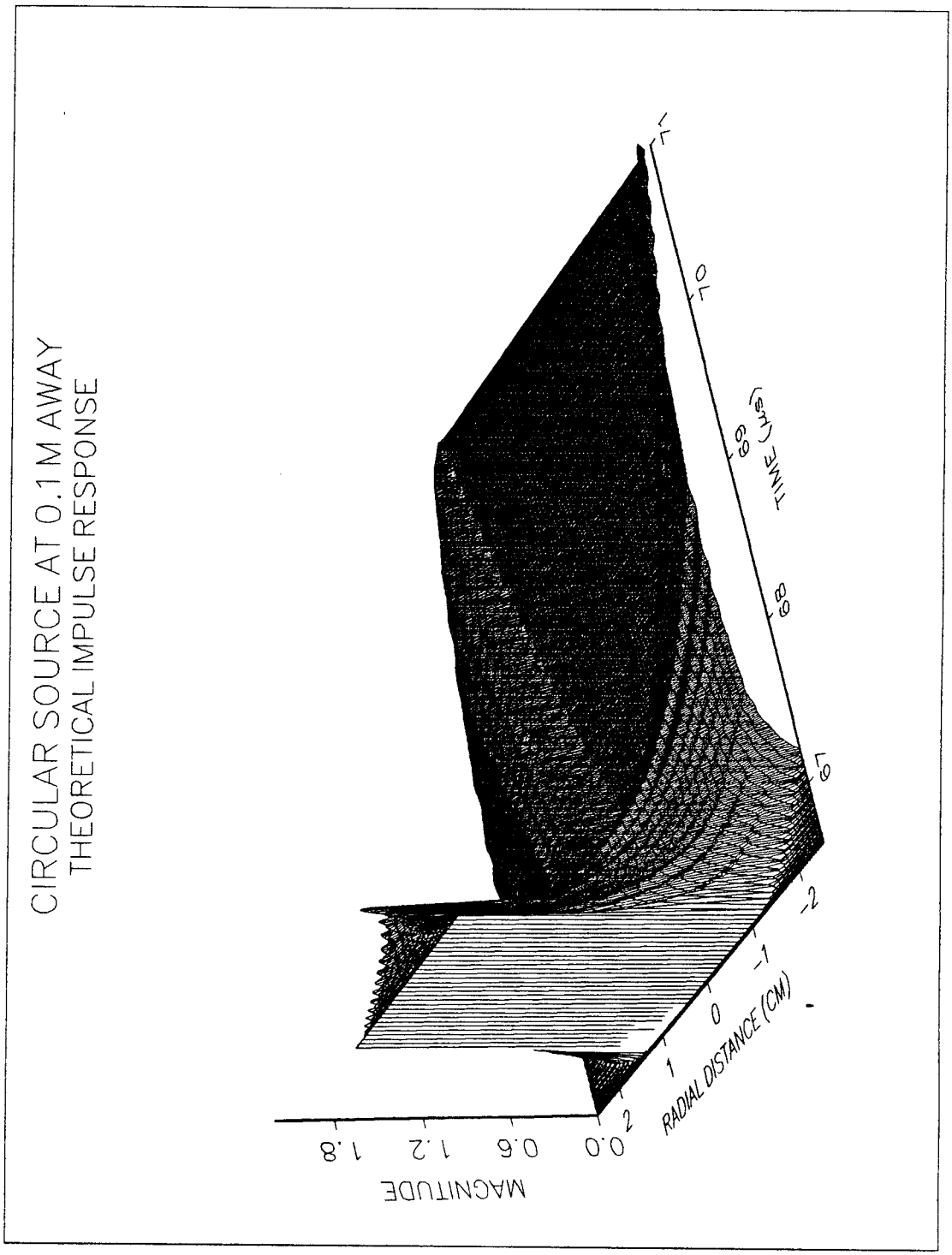


Figure 14. Impulse response with reduced base array techniques.

c. Miscellaneous Modifications

Other miscellaneous modifications done on Reid's program include the use of the **bessel** and **diff** commands. The command **temp = 2* besseln(0,arg)** from the program AC_FIL.M using MATLAB 3.5 was no longer valid on the updated version of MATLAB. MATLAB 4.0, the latest version of MATLAB, has modified and improved the command **bessel.** The command line **temp = 2* besseln(0,arg)** has been changed to **temp = 2 * bessel (0,arg)** in this thesis. This command line calls **besselj(0,arg)** if *arg* is real and **besseli(0,arg)** is *arg* is imaginary. The **diff** command was added at the end of the program AC_PROP.M to implement the time derivative of the spatial impulse response $\frac{d}{dt}p(x,y,z,t)$ of Eq. 12. The result of the time derivation of the spatial impulse response (at a separation distance of z = 0.1 m and a circular piston input) is shown in Fig. 15. Figure 15 is consistent with the theoretical pressure impulse response discussed in Ref. 7. The program code generation of AC_FIL.M and AC_PROP.M are explained thoroughly in Ref. 11. For detailed explanation of all MATLAB commands, see Ref. 15.

C. NON-IMPULSIVE TIME INPUT

Reid's work reflects an input that was temporally impulsive and spatially arbitrary. However, the actual experiment used a one cycle sine wave burst. This sine wave burst, shown in Fig. 16, is slightly distorted due to ringing effect introduced by the transmitter transducer. To account for this input, a convolution of Eq. 2 must be done as shown

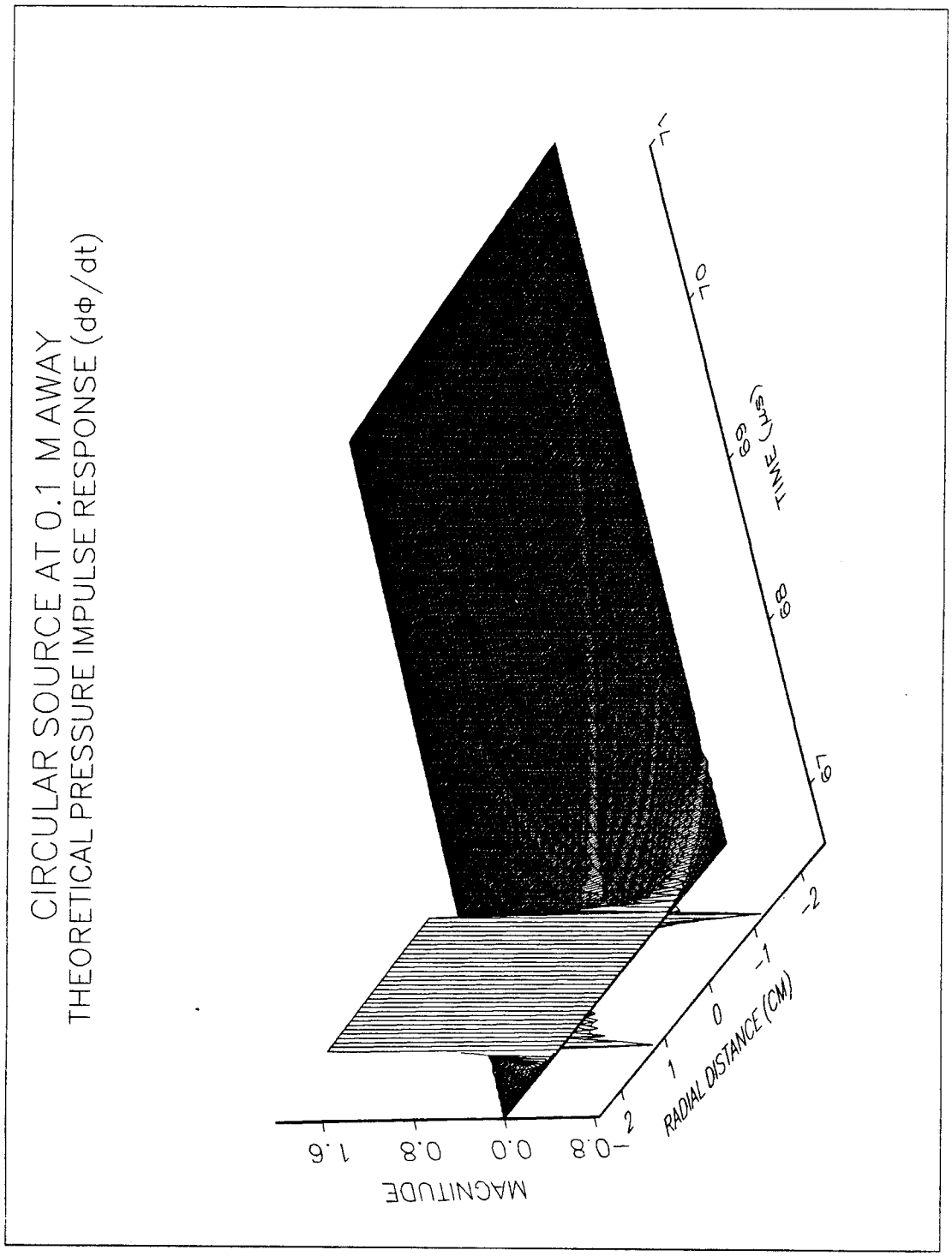


Figure 15. Theoretical pressure impulse response at z = 0.1 m.

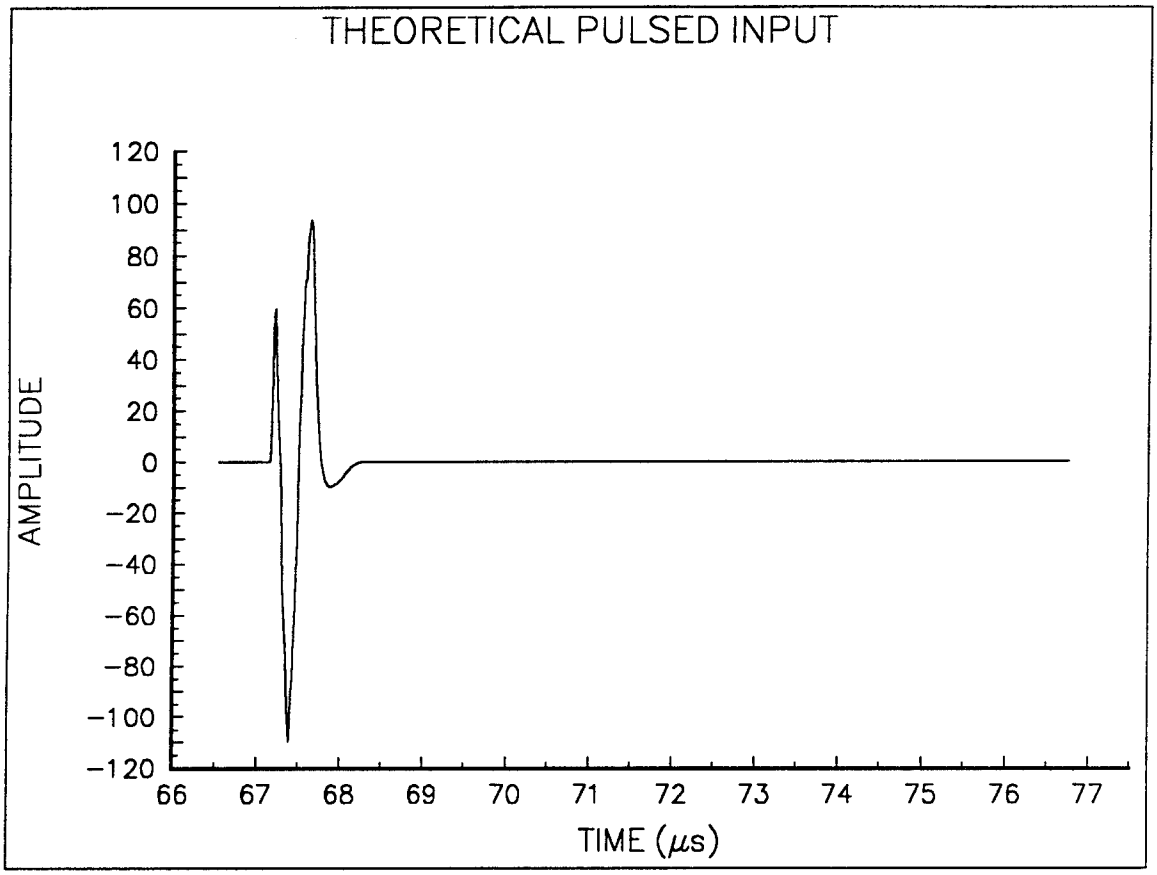


Figure 16. Theoretical pulsed input.

$$\phi(x,y,z,t) = T(t) \overset{*}{_t} \Im^{-1}\{\tilde{p}(f_x,f_y,z,t)\} \tag{19}$$

where $T(t)$ represents the input sine wave in Fig. 16. $\phi(x,y,z,t)$ is called the theoretical potential response for pulsed input. The theoretical pressure response for pulsed input is repeated below as

$$\psi(x,y,z,t) = T(t) \overset{*}{_t} \frac{d}{dt} p(x,y,z,t). \tag{20}$$

The program that implements the Eqs. 19 and 20 is called CONVOL.M. CONVOL.M is discussed in Appendix C and detailed in Appendix D. The theoretical potential response for pulsed input at separation distance of 0.1 m is shown in Fig. 17. The theoretical pressure response for pulsed input at separation distance of 0.1 m is shown in Fig. 18.

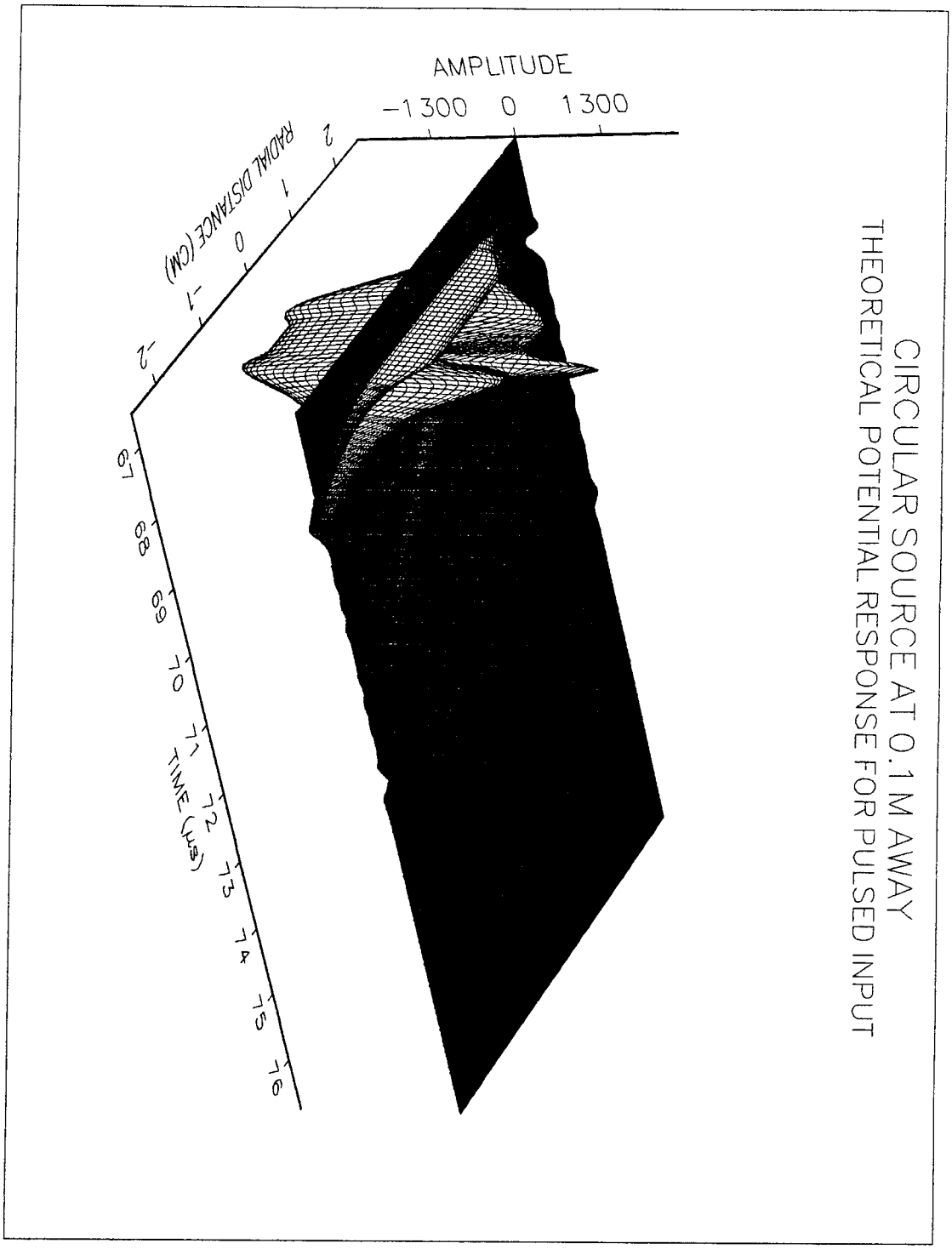


Figure 17. Theoretical potential response for pulsed input at z = 0.1 m

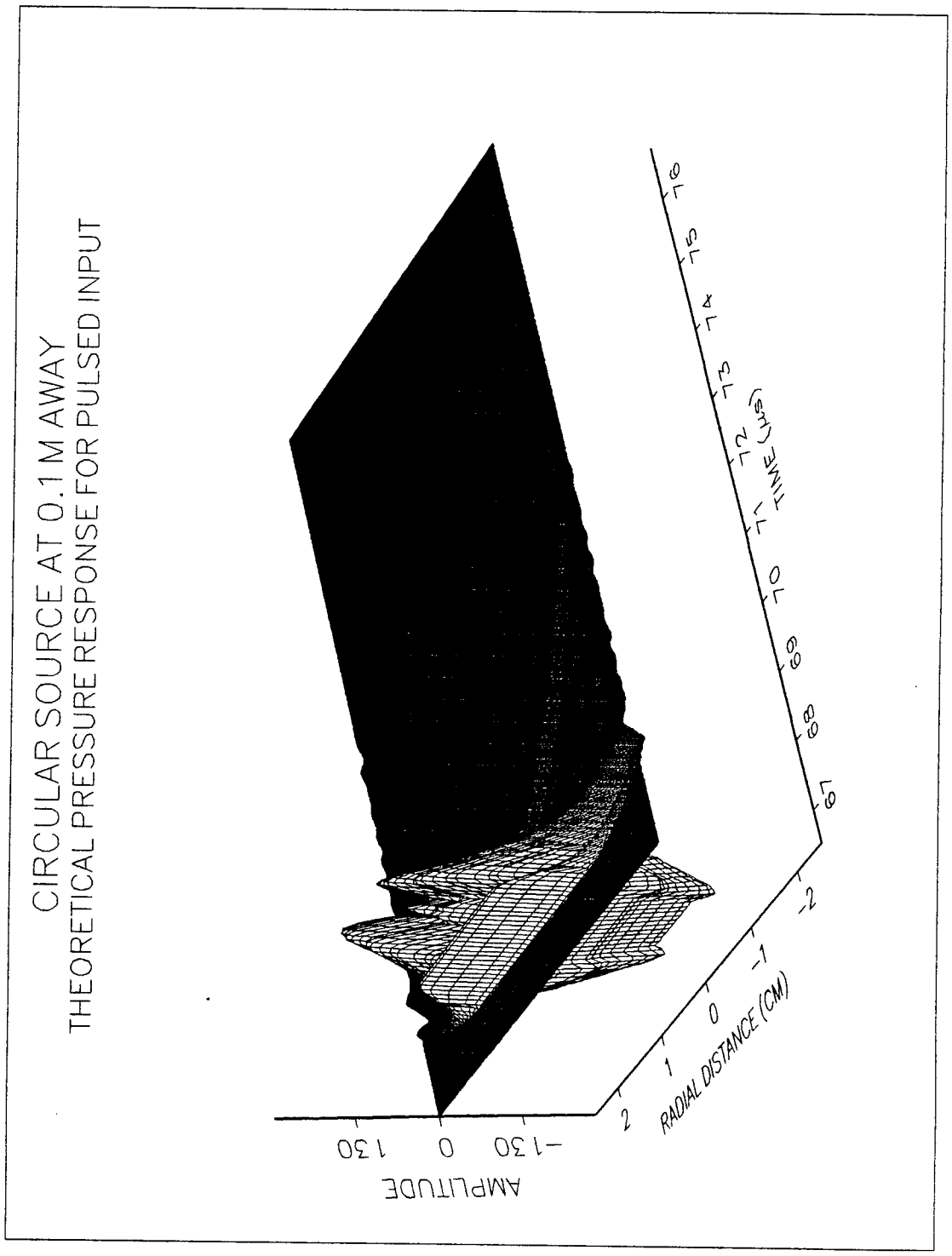


Figure 18. Theoretical pressure response for pulsed input at z = 0.1 m

D. DESCRIPTION OF THE PULSED ULTRASONIC DATA COLLECTION FACILITY

The overall experimental arrangement is shown in Fig. 19. It consists of a water tank, a scanning device, a pulse generator, a waveform transmitter, a waveform receiver, a waveform digitizer, an oscilloscope, a Bernoulli drive and a 486-DX 33-MHz personal computer. A 24-gallon water tank which contained ionized water is made of an aluminum sheet lined with styrofoam on the inside walls to absorb sound. It was placed on top of a wheeled cart which was positioned under the tank frame for easy adjustment and water changes. The water in the tank was always freshly deionized before each experiment to maintain a homogeneous medium throughout the tank and to prevent foreign material from causing acoustic reverberation. The acoustic source holder was stationary in the horizontal direction (X) with a manually adjusted slide in a vertical direction (Y). The separation distance between transducers was adjusted by manually moving the acoustic source holder back and forth along the tank frame. The available acoustic source used in this experiment is an immersion-type transducer made by Panametrics. Immersion transducers were specifically designed to transmit ultrasound in situations where the casing is partially or wholly immersed in fresh water as in this experiment. The acoustic source transducer is encased in a two-inch-diameter casing with a one-inch active region centered in the middle. The operating frequencies of the acoustic source range from 1.0 to 25 MHz. The acoustic source was connected to a pulse

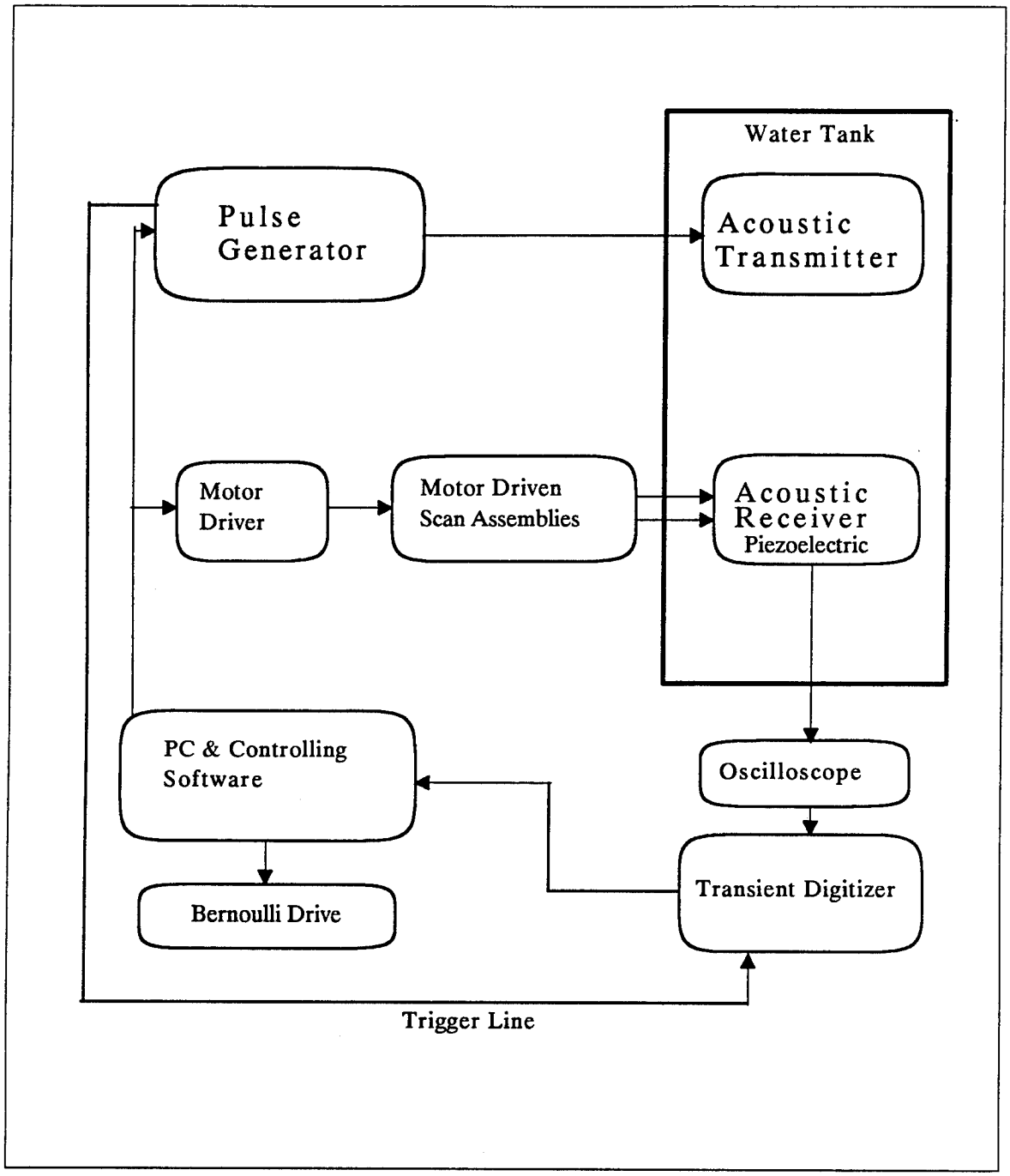


Figure 19. Overall experimental arrangement. (From Ref. 13.)

generator which provides a one-cycle burst of a 2 MHz sine wave to be transmitted by the acoustic source. A Wavetek model 270 12-MHz programmable function generator implemented the pulse generator block on the diagram of Fig. 19. The pulse generator was connected to a personal computer via a general-purpose interface bus (GPIB) cable and to the acoustic source via a coaxial cable. Its synchronous output was also connected to the external trigger input of the transient digitizer to ensure that the digitizer only captured received waveforms transmitted by the pulse generator. The acoustic receiver was made of a piezoelectric quartz-type acoustic transducer with an active receiving diameter of 0.2 centimeters. It has an output impedance of 1 MΩ. It was attached on the acoustic receiver holder which was moved by a program-controlled stepper motor-driven screw. This stepper motor driver was connected to a personal computer as shown in Fig. 19 via a GPIB cable and to the vertical (Y) and horizontal (X) motors via an electrical cable. The movement of the stepper motors during the experiment was controlled by two programs (MD-2SUBC.C and PREPOS.C) provided by ARRICK Robotics. These two programs, written in C programming language, were compiled and linked to LABVIEW using WATCOM C/C++ compiler. The movement of the stepper motors could be performed manually in the DOS environment or automatically within the LABVIEW environment. The source code for MD-2SUBC.C and PREPOS.C are detailed in Gatchell's thesis [Ref. 13]. A Tektronix RTD 720A waveform digitizer implemented the transient digitizer block of Fig. 19. This block provided a means to view and capture a fast transient events and eventually store them in the Bernoulli drive for further

processing. The input impedance of Tektronix RTD 720A is 50 Ω. Shown in Fig. 19, the acoustic receiver was connected to the transient digitizer via an oscilloscope. The oscilloscope used in this experiment is the COS61000M oscilloscope (100 MHz bandwidth) made by Kikusui Electronics and has an input impedance of 1 MΩ and an output impedance of 50 Ω. This oscilloscope provide a means of impedance matching the acoustic receiver and the transient digitizer. The connection between the acoustic receiver and the channel 1 input of the oscilloscope was made via a coaxial cable. Similarly, the connection from the channel 1 output (located on the back of the oscilloscope) and the transient digitizer was made via a coaxial cable. As mentioned earlier, the transient digitizer's external trigger was connected to the pulse generator's synchronous output via a coaxial cable. This scheme allows the transient digitizer to only capture and digitize waveform transmitted by the pulse generator via the acoustic source transducer. The captured and digitized waveforms were stored in the Bernoulli drive for ease of data transfer and to free the much-needed disk drive memory in the personal computer. The specific type of Bernoulli drive used in this experiment was the Iomega Bernoulli Box II 5.25-inch, dual external drive, each with 20 MB capacity. The Bernoulli drives were configured to be drive D and drive E. The personal computer used in this experiment is an IBM compatible PC with an Intel 486DX 33 MHz processor and 8 MB of random access memory (RAM). This personal computer is the central control for data acquisition of this experiment and is where all of the applicable software for synchronizing events was loaded. Detailed descriptions of each component of Fig. 19 are provided in Gatchell's

thesis [Ref. 13]. The specific connection and addressing of all GPIB connected equipments and the implementation of the LABVIEW instrumentation software are explained thoroughly in Gatchell's thesis [Ref. 13].

The three specific requirements to be considered in assembling the data collection facility are: (1) the geometry of the source-to-receiver, (2) the positional precision and mechanical stability of the scanning mechanism, and (3) the alignment of the scanning plane with the source plane (which must be completely parallel as it is required in the angular spectrum method) [Ref. 1]. Requirement number one was accomplished by setting the observation area larger than the active region of the acoustic source. The observation area was also made so that it is freely floating in the medium and away from the tank wall to eliminate sound reflections. Requirement number two was accomplished using a program- controlled stepper motor. The program used to control, to link, and to synchronize all motor drivers, function generators, and digitizers was LABVIEW. Finally, due to a lack of a precision alignment mechanism, requirement number three was marginally accomplished. The planar alignment between the source and the receiver planes was done by manually triggering the pulse generator at both the left and the right extremes and ensuring the sound propagation arrived at the receiver point with approximately the same time duration. If sounds arrived with different time durations, then the source transducer holder was manually repositioned until they were exactly the same.

E. MODIFICATIONS ON THE PULSED ULTRASONIC DATA COLLECTION FACILITY

Hardware modifications were made in order to provide a higher signal-to-noise ratio. One piece of hardware that was added was a wideband power amplifier just before the source transducer to increase the signal power. This wideband power amplifier has an operating frequency of 10 KHz to 250 MHz and a gain of 23 dB. The other hardware modification was the addition of a dual amplifier just before the digitizer to amplify the received signal. This dual amplifier has an operating frequency of 0.1 – 400 MHz and a gain of 20 dB. These modifications were not required when using a broadband quartz-type receiver due to its higher sensitivity. However, these modifications were essential when using a narrowband transducer. The last modification made was the replacement of an acoustic transducer from the piezoelectric quartz acoustic transducer to a polyvinylidene flouride (PVDF) bilaminar shielded-membrane hydrophone made by Sonic Technologies. The active element on this transducer is around 0.5 mm diameter. The advantage of this receiver lies in its ability to received signals with many harmonics of fundamental frequency. The comparison of the signal received with piezoelectric quartz acoustic transducer and the PVDF ultrasonic transducer is shown in Figure 18. The signal received using a piezoelectric quartz acoustic transducer (Fig. 20a) showed more ringing waves than the received signal using PVDF ultrasonic transducer (Fig. 20b) which is undesirable in this thesis. This is due to a wave superposition at different modes. The modified set-up is shown in Fig. 21.

The impulse response of the system can be expressed in terms of the inverse 2-D spatial Fourier transform of the spatial response $\tilde{p}$. The spatial response is the product of the spatial excitation (which is a circular piston impulse input for this thesis) and the Fourier transform of the total impulse response *h(x,y,t)*. The total impulse response is equivalent to the Green's function with an additional scaling factor of 2π added to Reid's work [Ref. 4]. Other program modifications made on Reid's work include the expansion of the observation area and the used of the modified version of the **bessel** command (MATLAB 4.0, which was used in this thesis, has improved and modified the use of the **bessel** command from MATLAB 3.5 [used in Reid's work]). To account for an input other than $\delta(t)$, the modeled diffracted wave was a convolution of the system impulse response and the temporal excitation of the acoustic source. Section D and E describes the original design of the pulsed ultrasonic data collection facility and the modified set-up used in this thesis. The next chapter describes the experimental procedures for data collection and data processing. It was divided into three separate sections. They are: (1) data collection which describes the series of events that happened during the data collection phase, (2) data conversion which explains the requirement to convert the collected data from a binary text format into a MATLAB format, and (3) data averaging which describes the need for the data averaging and the process in which the data were averaged.

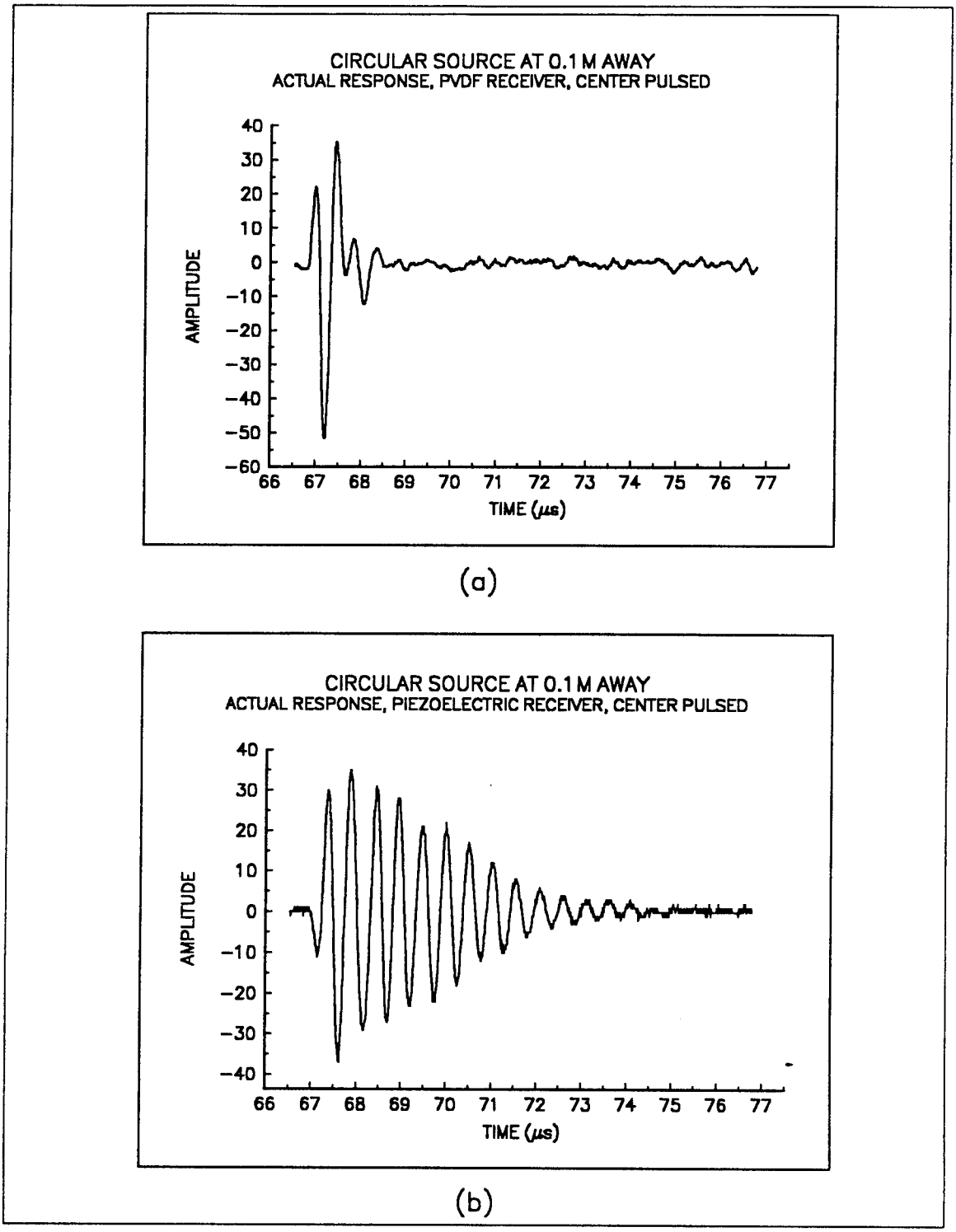


Figure 20. Comparison of a received signal at z =0.1 m using (a) a PVDF narrowband receiver and (b) a piezoelectric quartz type acoustic receiver.

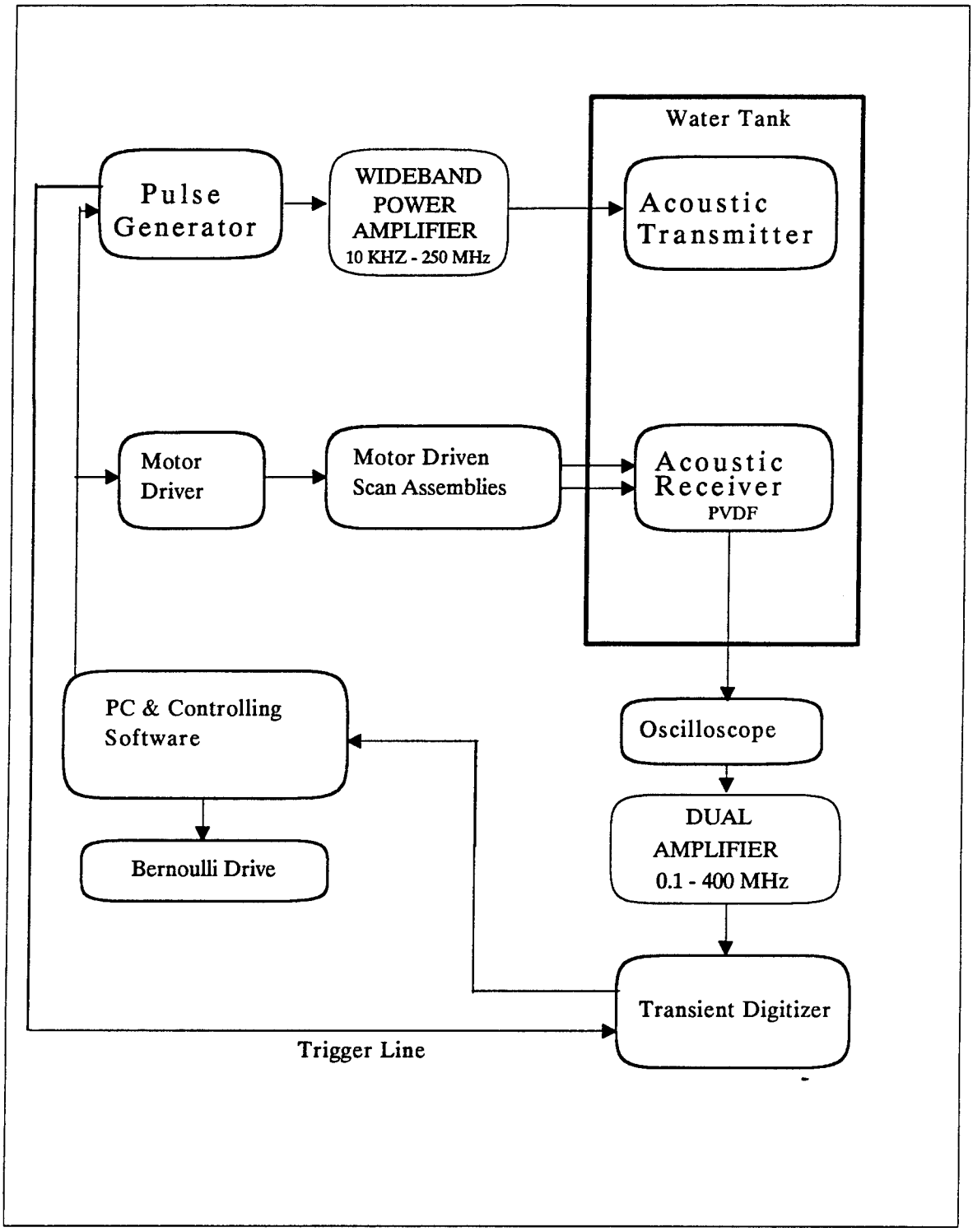


Figure 21. Modified experimental arrangement. (After Ref. 13.)

III. EXPERIMENTAL PROCEDURES

A. DATA COLLECTION

The data collection section of this thesis was accomplished using the set-up shown in Fig. 21. The data collection was configured to have a scan area of two inches by two inches with the acoustic source at the center. The collection density was set-up to be 64 by 64 steps with a step travel increment of 0.794 mm as shown in Fig. 20. The data was collected under the control of the LABVIEW instrumentation software. The series of events began with the acoustic receiver being pre-positioned at the starting point of the collection phase which is in the upper left corner of the scan area (point (1,1) in the Fig. 22). It was followed by the pulse generator triggering a single-cycle burst of a sine wave with a frequency of 2 MHz and amplitude of 10 V. This signal was then amplified by the wideband power amplifier. The output of the wideband amplifier was connected to the acoustic source which was positioned fully immersed in deionized water and located in front of an acoustic receiver. A narrowband acoustic receiver received the transmitted signal and was followed by the dual amplifier where the signal was amplified once more. The signal was fed into the RTD 720A transient digitizer which was externally triggered by the pulse generator. Once the trigger was recognized, the data acquisition sequence began. The duration of the data acquisition is called the "sample time". The signal value during the sample time was held for analog-to-digital conversion and then stored

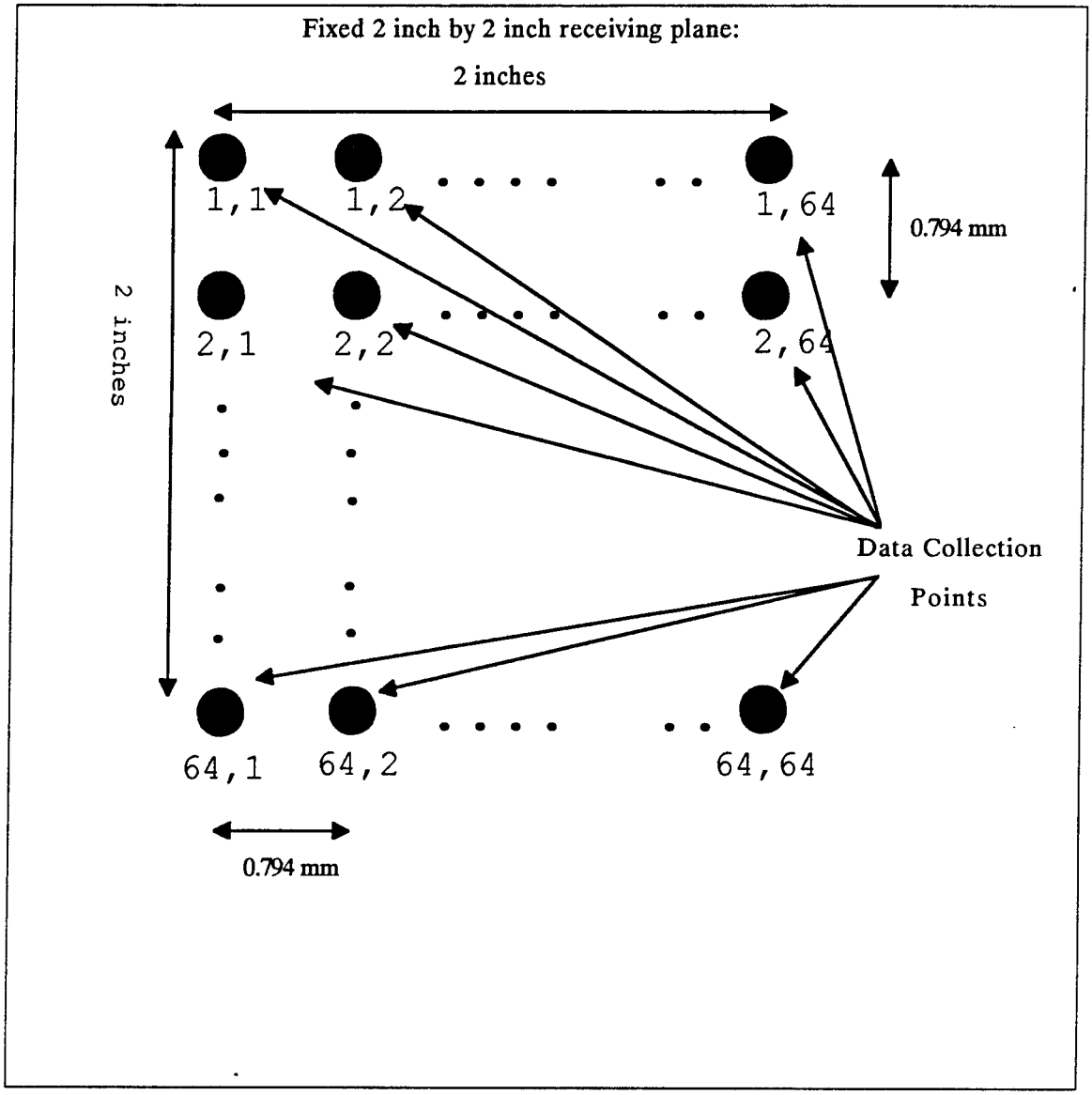


Figure 22. 64x64 collection density plane. (After Ref. 13.)

in the Bernoulli drive with a file name "data*.txt". The " * " stands for a number starting with the numerical value of 1 and ending with 64 which corresponds to the row of the collection area being scanned. LABVIEW then moved the acoustic receiver to the next collection point and repeated the process until all 64 collection points within that row were completed and stored within the same file. The next row with 64 collection points has a similar file name with an exception of the number incremented by 1. Before the experiment can be conducted, several parameters must be entered on the LABVIEW instrumentation program. For detailed operation of the GPIB, see Ref. 13.

Other information that must be entered onto the LABVIEW program includes the channel 1 range, channel 1 coupling, acquire interval, acquire length, trigger source, trigger coupling, and trigger position. The detailed explanation of each parameter is explained fully in Gatchell's thesis [Ref. 13].

The data collection phase is completed when all 64 rows are sampled. The next phase of the experiment is the conversion of data from binary text data format to a MATLAB format for further processing of data. The data conversion technique is discussed in the next section of this chapter. However, due to the noise of the collected data, a data averager is essential which is discussed in section C.

B. DATA CONVERSION

Once the entire data acquisition is completed, the collected data must be converted from a binary text format into a MATLAB format for further processing. The DATO2.CPP program was created to convert the collected data. A Borland C++

(version 3.1) compiler was used to compile the source code DATO2.CPP to produce the executable program DATO2.EXE which actually converts the collected data. To perform the conversion, all data files must reside in one directory along with DATO2.EXE. The final step in converting all collected data is to type "DATO2 *.txt" from the command line of the directory where all data are located. The result of the conversion are files with the same file name as the binary text data and with an extension of ".mat". Appendix G is a listing of DATO2.CPP.

C. DATA AVERAGING

In order to increase the signal-to-noise ratio of the collected data without creating a complex filter, a data averager was used. This data averager was written in MATLAB code and is called AVG.M. The process of data averaging was performed in the time domain starting from data(1) up to data(1024). Each signal time index was averaged by first adding "x" number of data to its left position, "x" number of data to its right position, and the number itself on the current time index. After performing data the additions, it must be divided by the number equal to twice the value of "x" plus 1. If the index number is less than one, then the data at the end of the signal must be added until "x" number of data is added. If the index number exceeds the maximum number of data, then the data beginning with the first one is added until "x" number of data is added. This is called the "wrap around" on the source code. The number representing "x" is arbitrary and can be changed anytime (Note: processing time increases as the number of "x" increase). The value of "x" can be changed in the command line **for b=a-3:a+3** (x =3, here) which is in

the fourth loop of the source code AVG.M. The source code AVG.M is detailed in Appendix F and explained in Appendix E. This process continues until all data points are averaged within the given pulse. This can be expanded to perform averaging of all 64 pulses within a given row. After each averaged pulsed, the program saves the averaged pulsed to a file with a different file name. All 64 pulses are saved into the same file. The program AVG.M is also capable of averaging all 64 rows totaling 4096 pulses. A sample comparison of raw data with an averaged data is shown in Fig. 23. A comparison between the averaged data and the raw data shows that the averaged data is smoother. The detailed explanation of the program AVG.M is included in Appendix E and is detailed in Appendix F.

The data collection phase of this thesis was totally automated. Detailed explanation of the data collection program can be found on Gatchell's work [Ref. 12]. The source code for the data conversion program (DATO2.CPP) was developed by Ray van deVeire of the Naval Postgraduate School and is found in Appendix G. The next chapter presents some of the graphical comparisons between the theoretical and the experimental pulsed response.

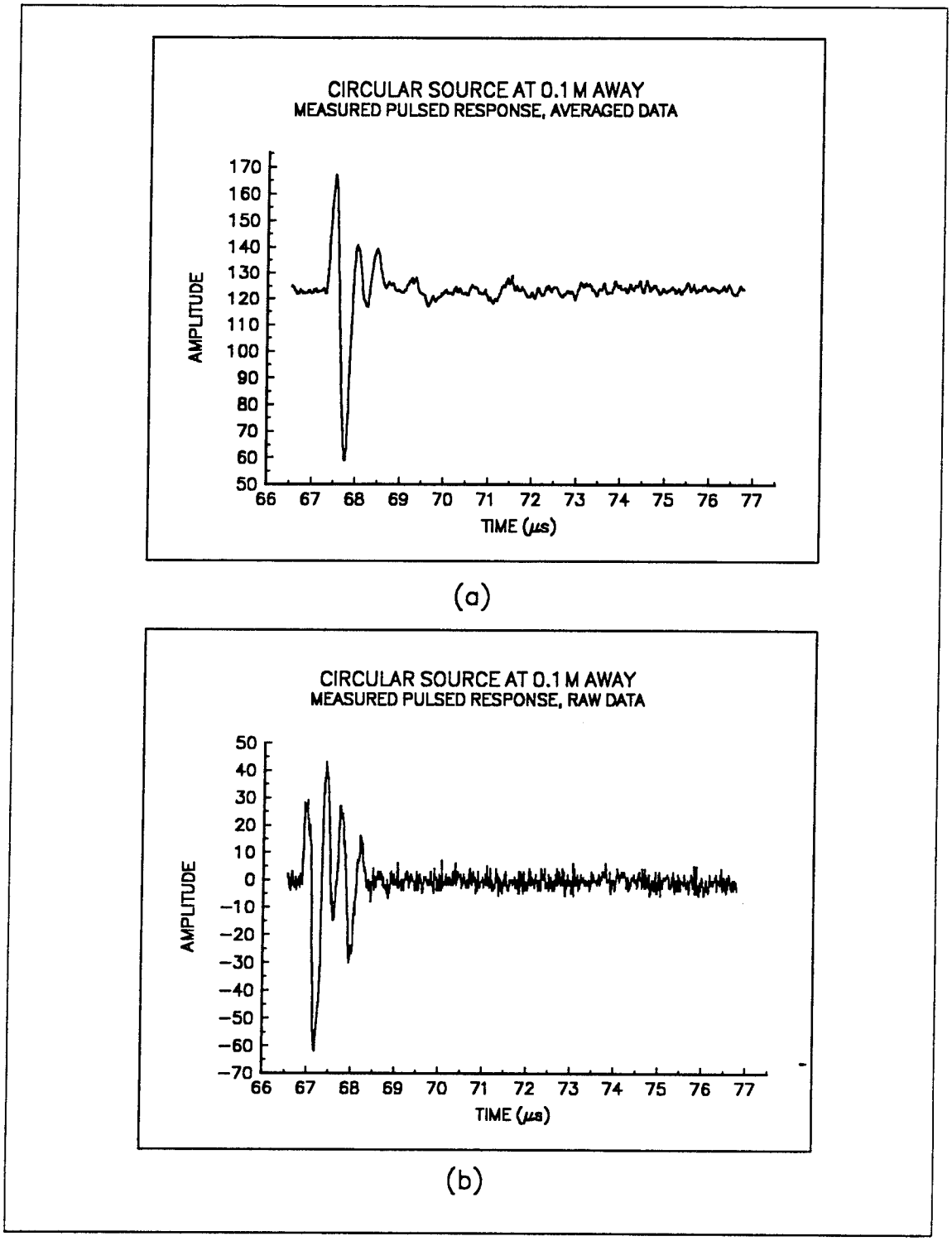


Figure 23. Comparison of an (a) averaged data and a (b) raw data at $z = 0.1$ m and at 32nd pulsed.

IV. EXPERIMENTAL RESULTS

Figure 21 shows the experimental set-up used for the data acquisition phase of this thesis. Two independent experiments were conducted using the same acoustic source and acoustic receiver at different separation distances. The input source used in the theoretical side was a circular piston impulse input with a diameter of 1.27 cm (which is equal to 31 samples). The goal of this experiment was to compare the theoretical pulsed response with the measured pulsed response at a given separation distance. The first section of this chapter deals with separation distance of 0.1 m and the second section deals with the separation distance of 0.2 m.

A. SEPARATION DISTANCE OF 0.1 M.

Table II shows the summary of the parameters used in the theoretical experiment. These parameters are thoroughly explained in Reid's work [Ref. 4]. Table III shows the value of the parameters entered in the program LABVIEW before the data acquisition was started for the experiment. Figure 24 shows the measured pulsed response at the 32nd row obtained by means of the data acquisition system shown in Fig. 21. Figure 25 shows the theoretical pulsed response using the modified Reid's program with parameters shown in Table 2. Comparison of Figs. 24 and 25 shows the similarity in trends and shapes between the two with the exception of the center spike which exists in the theoretical pulsed response but not in the measured pulsed response. Figures 26 and 27 are the absolute value of the pulsed response shown in Figures 24 and 25, respectively. The

comparison of Figs. 26 and 27 shows the comparison of the relative magnitude between the experimental and the theoretical pulsed response. It shows that the relative magnitude of the peaks between the two figures are very similar. Figures 28, 29, 30 and 31 are the additional figures used to compare the theoretical and the measured pulsed response at four different radial locations. These locations are: center pulse (r = 0 m), off center but still inside the active region (r = 0.635 cm), at the edge of the active region (r = 1.27 cm), and outside the active region (r = 1.905 cm). Figure 28 shows that the comparison between the theoretical and the experimental pulsed response at center pulse (r=0) are not in total agreement especially at the end of the pulsed response.

DEFINING PARAMETERS FOR THEORETICAL IMPULSE RESPONSE WITH Z = 0.1 M		
NAME	**VALUE**	**SUMMARY**
N	128	Size of square base array
M	433	Number of time slices
Step	3	Number of leading zero rows
c	1,500	Acoustic velocity in media (m/s)
z	0.1	Distance, source to receiver (m)
time_max	71e-6	Maximum time of propagation (s)
rho_max	629	Spatial radius of filters ($length^{-1}$)
d	31	diameter of circle (samples)

Table II. Summary of parameters used in Reid's program for separation distance of 0.1 m.

Figure 29 shows that the comparison between the theoretical and the experimental pulsed response at the off-center location (but still inside the active region) are in agreement. Comparison of the pulsed response between the theoretical and the experimental output at the edge of the active region (shown in Fig. 30) and outside the active region (shown in Fig. 31) also appears to be in agreement. The possible sources of disagreement at the center pulse are discussed in the next chapter.

DEFINING PARAMETERS FOR MEASURED PULSED RESPONSE AT Z =0.1 M		
NAME	**VALUE**	**SUMMARY**
Directory	d:\sonic1	Location of the collected data
Channel 1 range	50 mV	Full scale of the vertical acquisition window
Channel coupling	ac	Method of coupling: AC, DC, or OFF
Frequency	2,000,000	Pulse sine wave frequency
Amplitude	10	Pulse sine wave amplitude
Acquire interval	0.01 μs	Interval between waveform data samples
Acquire length	1,024	Record length
Trigger source	Extrenal	Source of trigger signal
Trigger coupling	ac	Coupling for the trigger event: AC, DC, of HF Reject
Trigger position	6,592	Sets the amount of pre-trigger or post-trigger

Table III. Summary of parameters entered on the LABVIEW program for the experimental data acquisition for z =0.1 m.

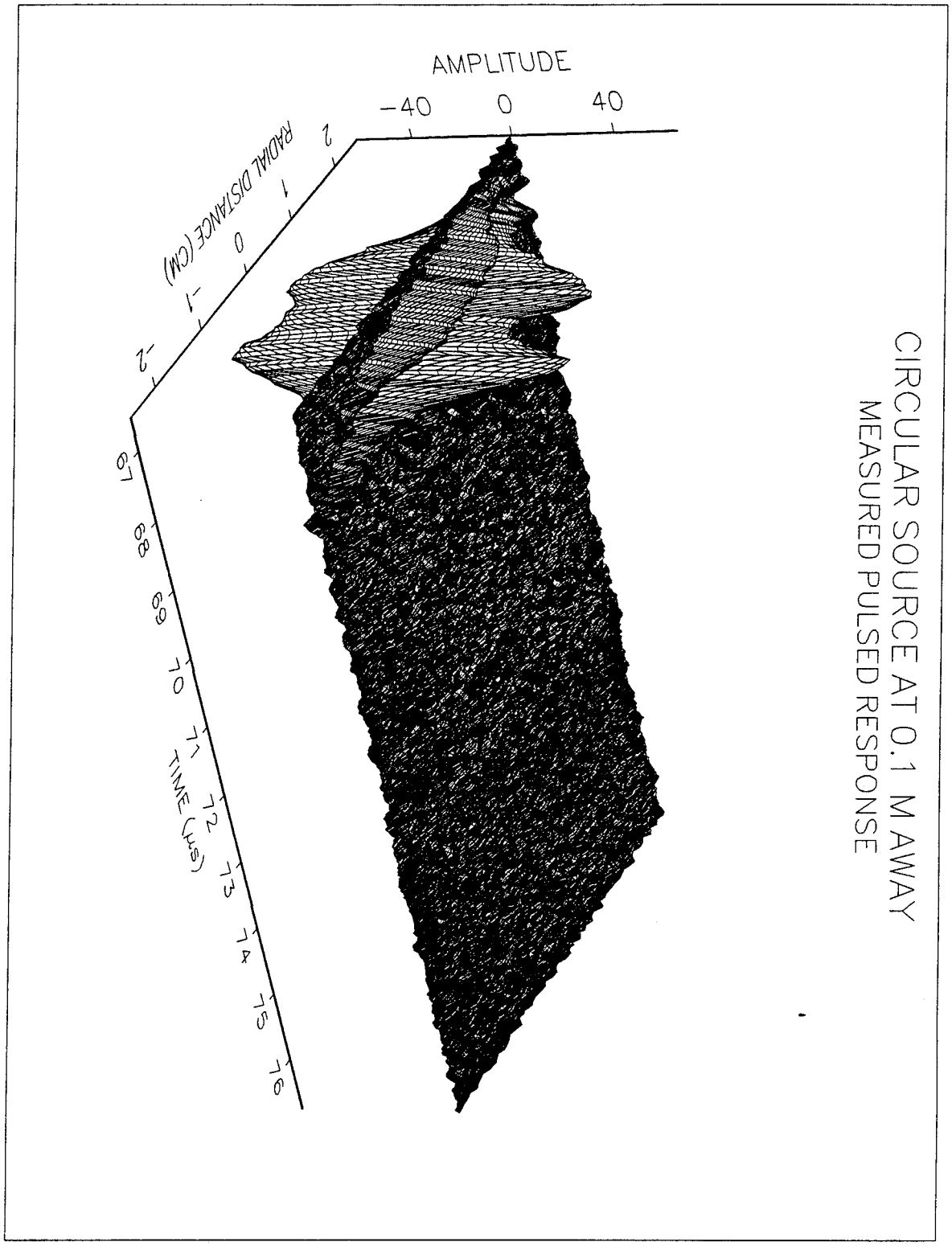


Figure 24. Measured pulsed response at z = 0.1 m. Circular source.

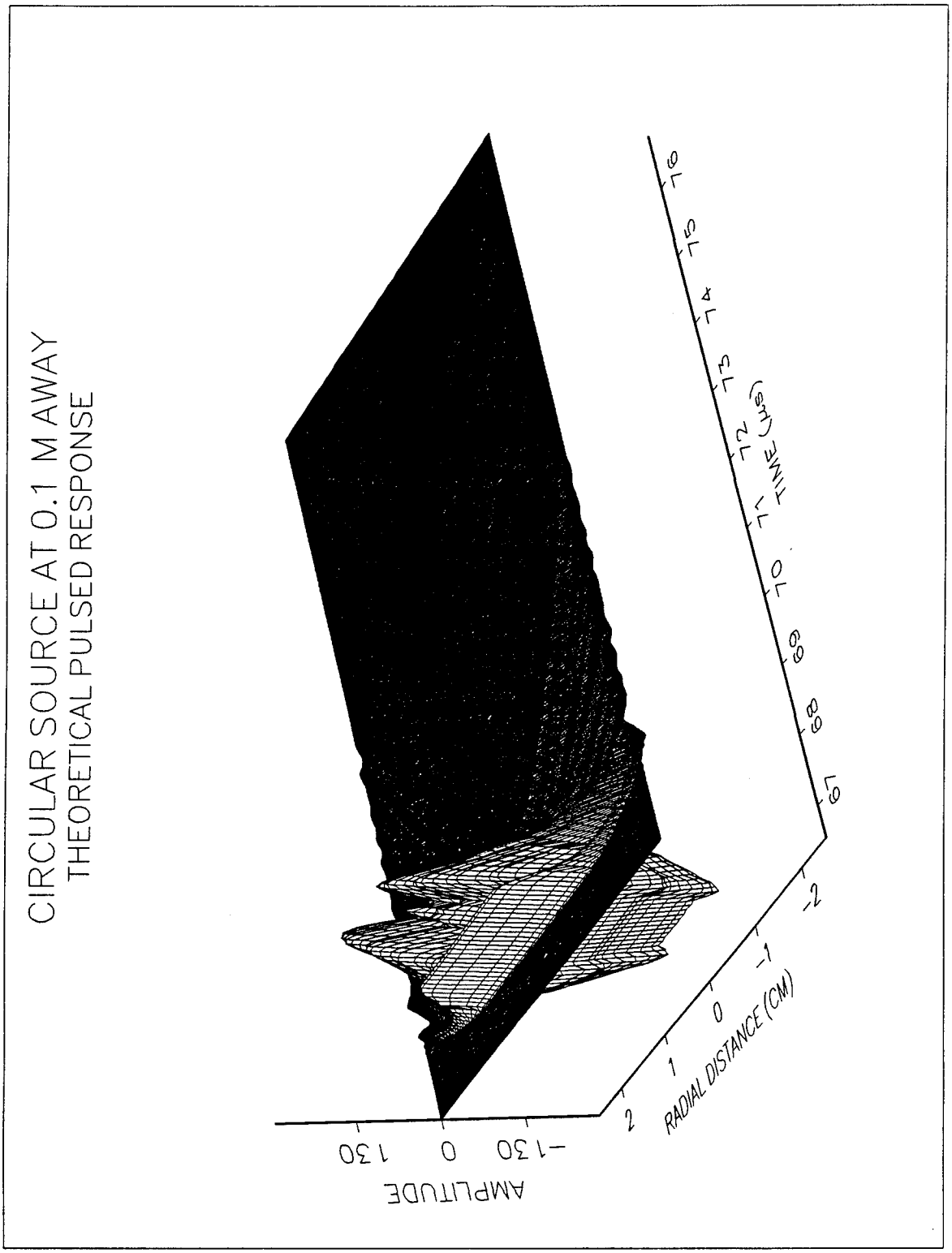


Figure 25. Theoretical pulsed response at z = 0.1 m. Circular source.

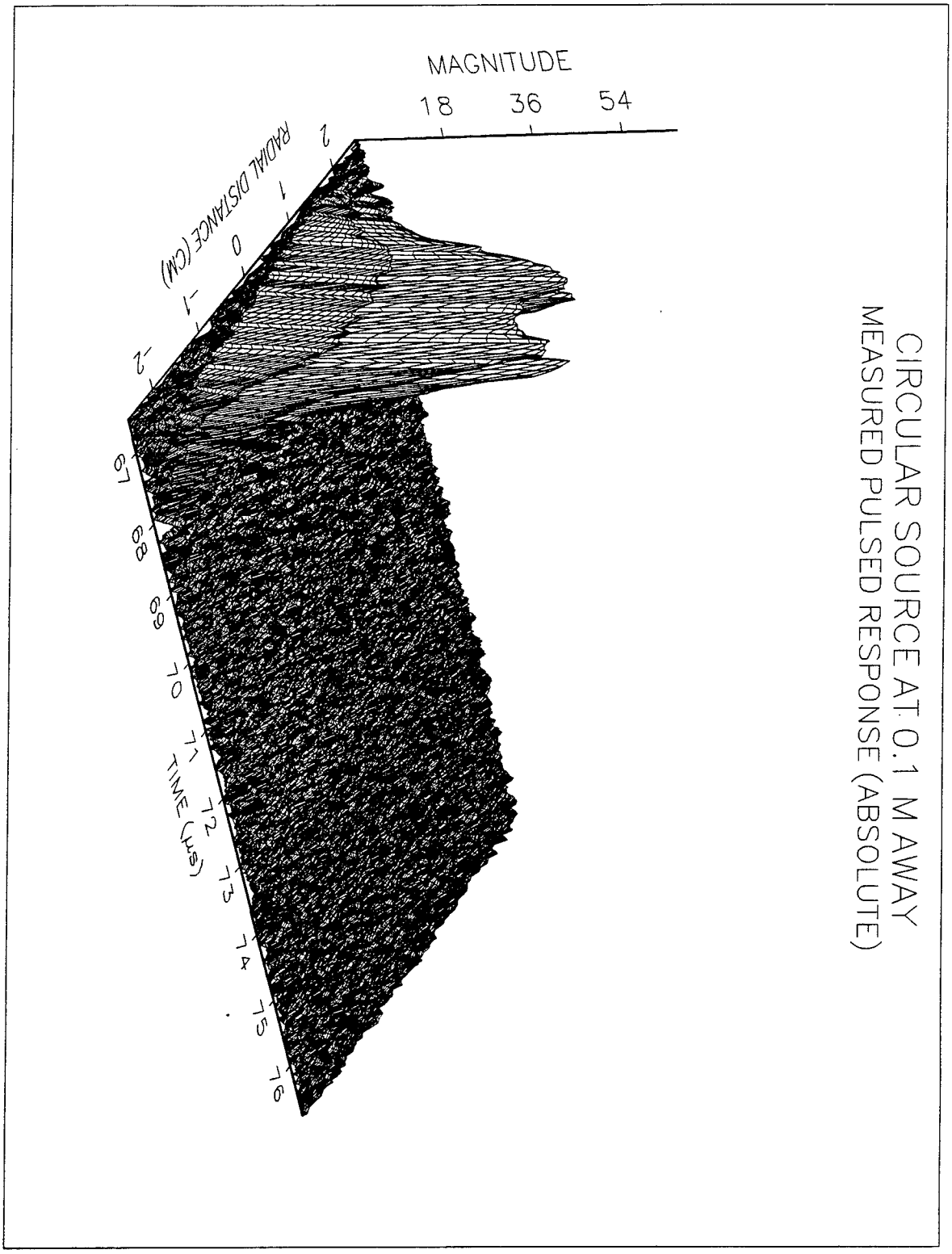


Figure 26. Measured pulsed response (absolute) at z = 0.1 m. Circular source

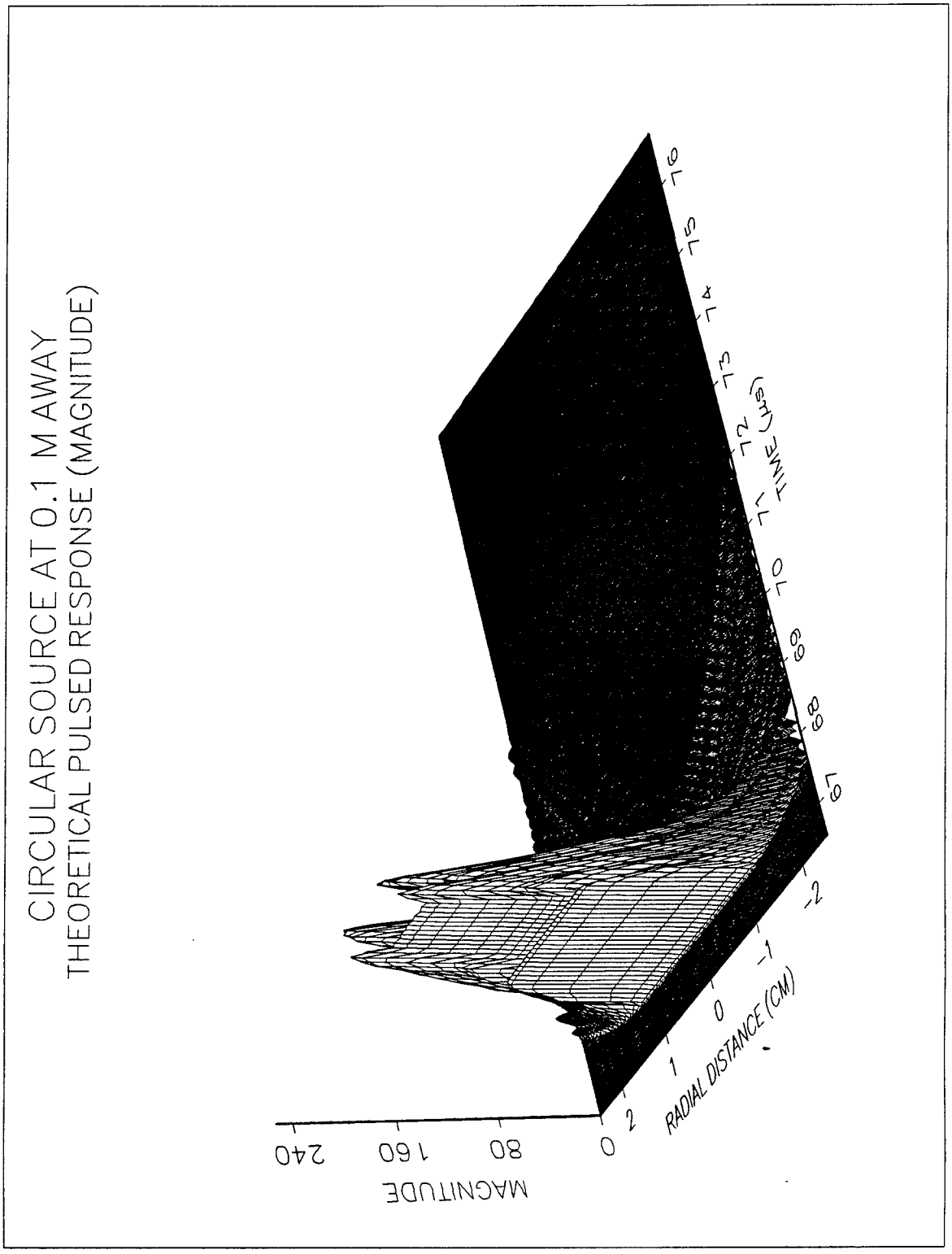


Figure 27. Theoretical pulsed response (Magnitude) at z = 0.1 m. Circular source.

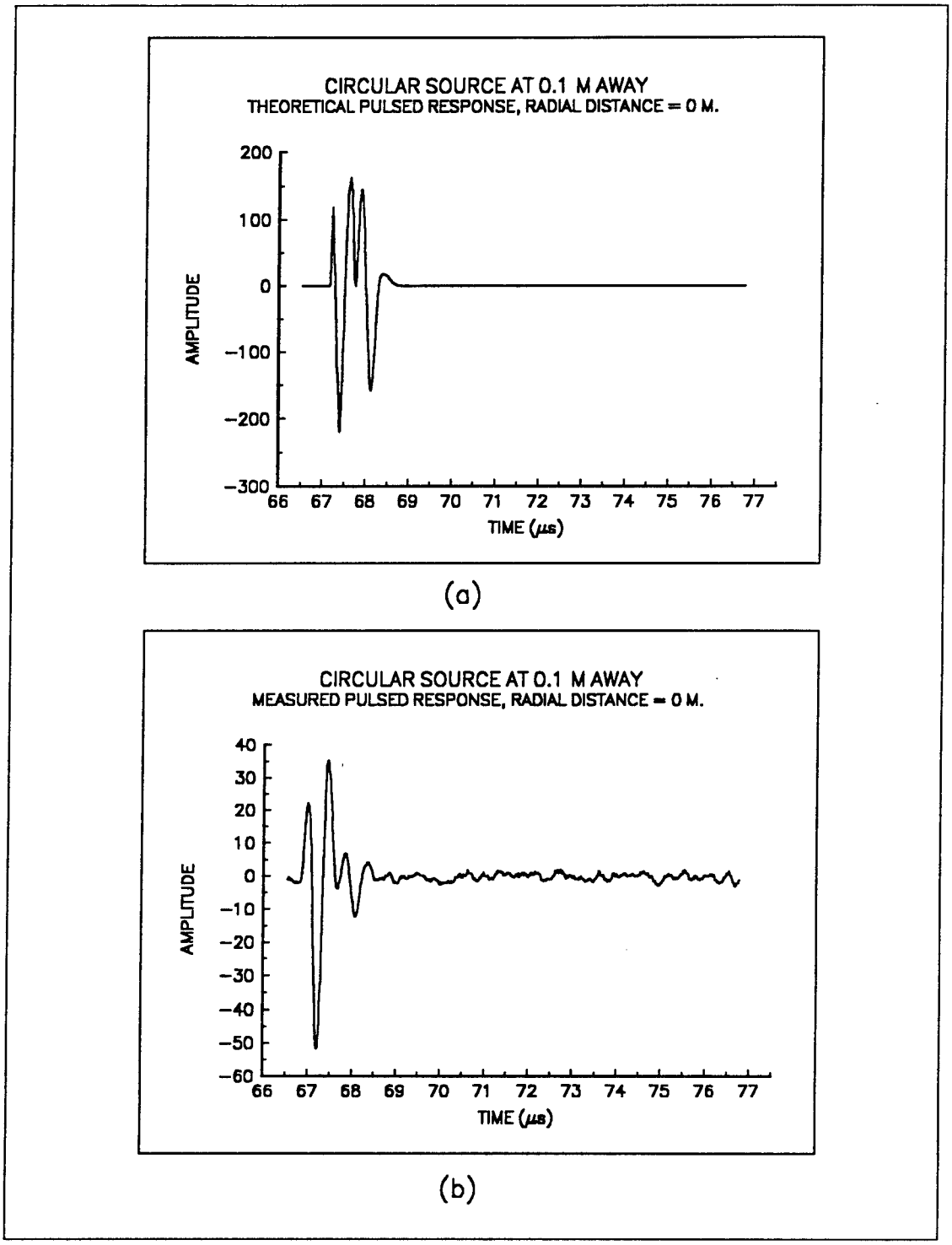


Figure 28. Comparison of the (a) theoretical and (b) measured pulsed response at a radial distance of 0 cm and a separation distance of 0.1 m.

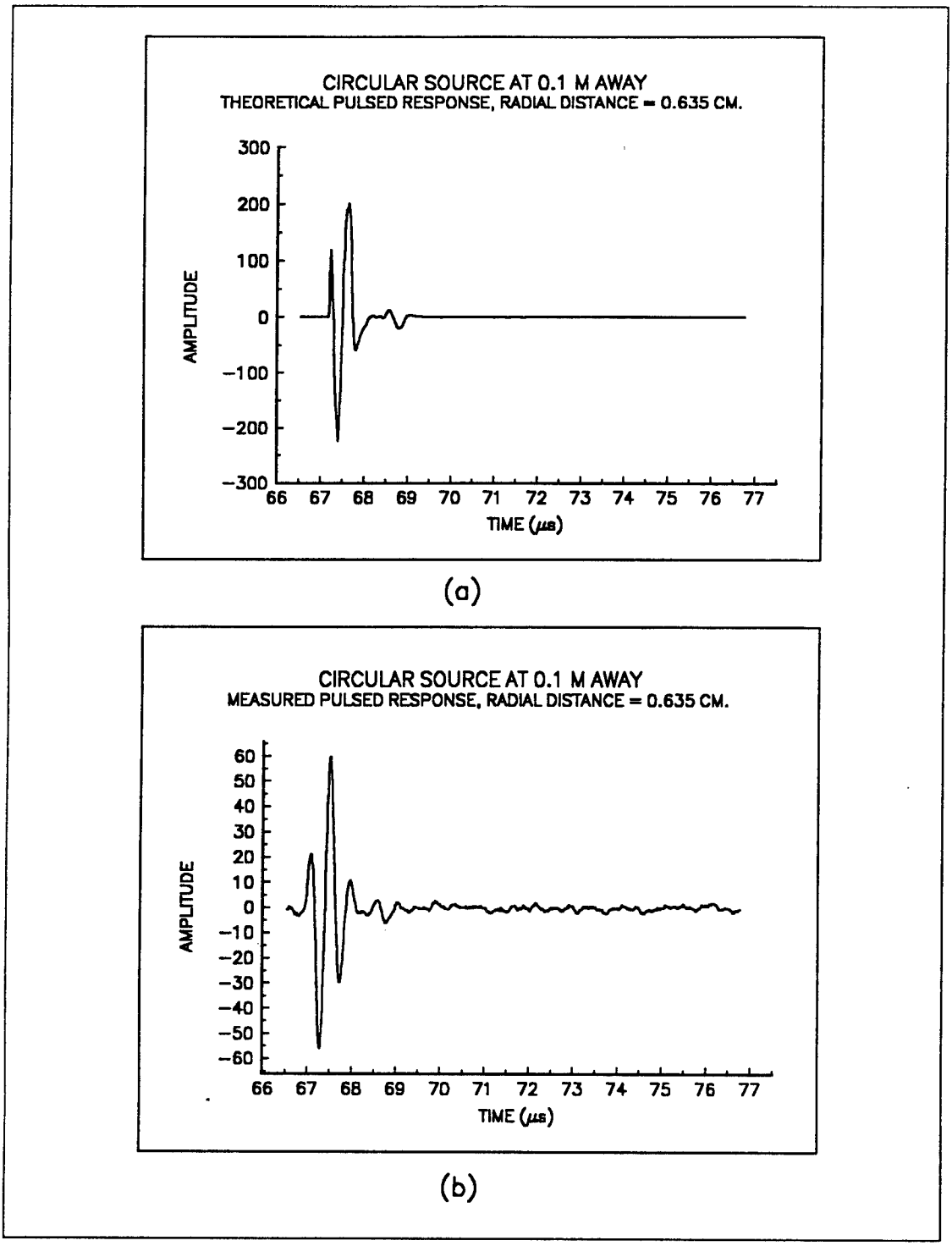


Figure 29. Comparison of the (a) theoretical and (b) measured pulsed response at a radial distance of 0.635 cm and a separation distance of 0.1 m.

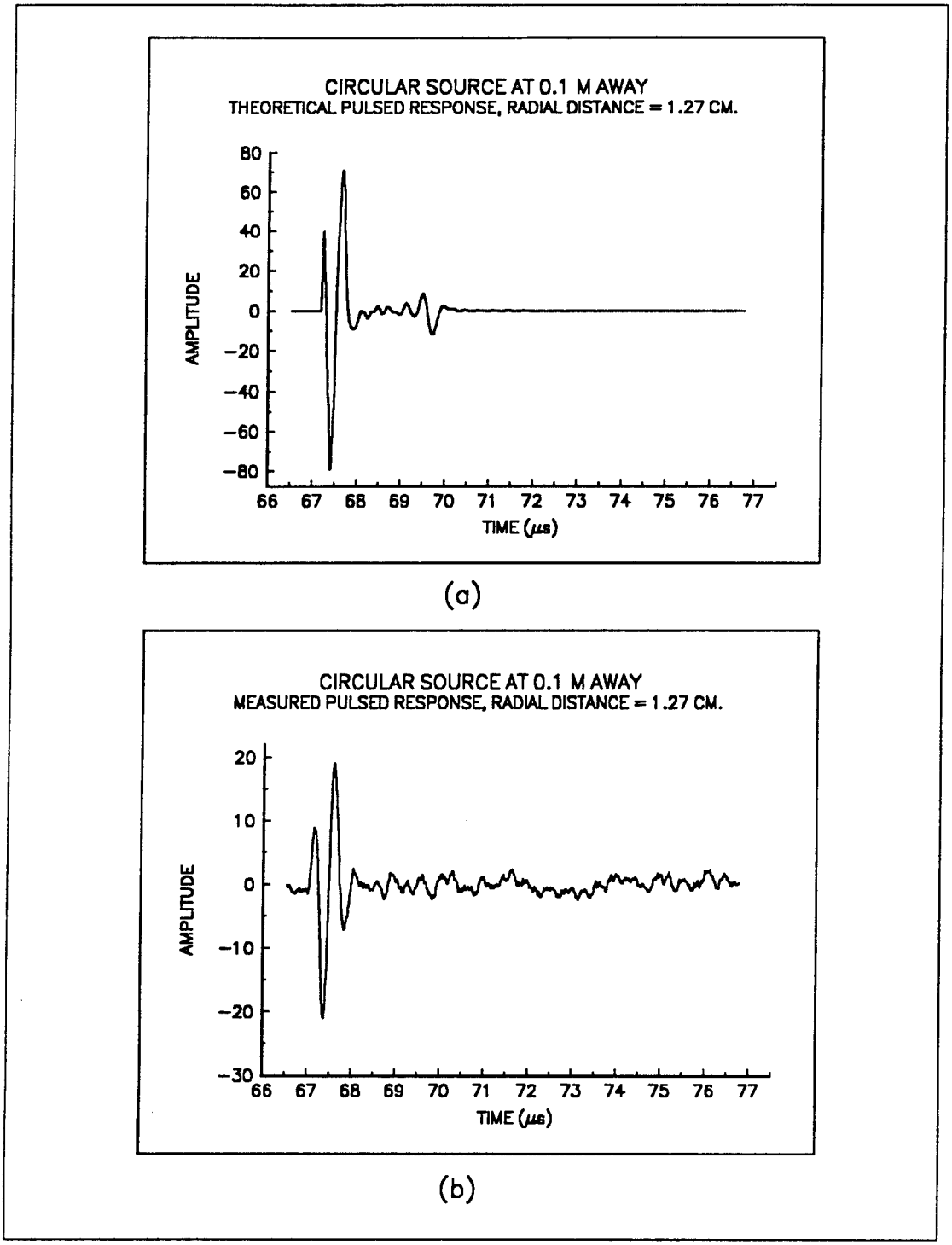


Figure 30. Comparison of the (a) theoretical and (b) measured pulsed response at a radial distance of 1.27 cm and a separation distance of 0.1 m.

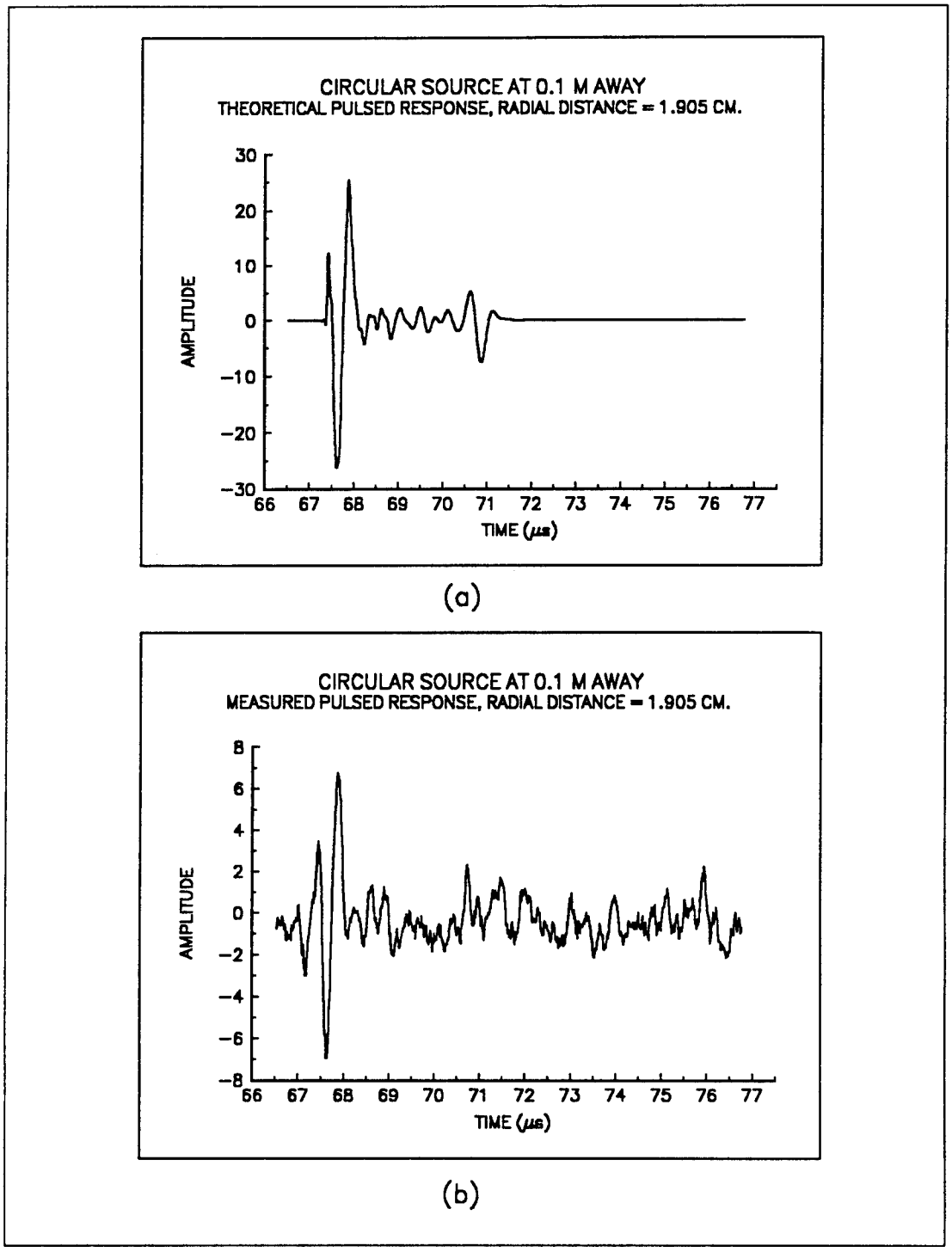


Figure 31. Comparison of the (a) theoretical and (b) measured pulsed response at a radial distance of 1.905 cm and a separation distance of 0.1 m.

B. SEPARATION DISTANCE OF 0.2 M.

The procedures used for the experiment at separation distance of 0.2 m were the same as those used in the experiment at separation distance of 0.1 m. Some of the major differences include: moving the acoustic source holder at 0.2 m away from the acoustic receiver, changing the parameters in the Reid's modified program, and changing the parameters entered in the LABVIEW program. Table IV shows the summary of the parameters used in the theoretical experiment and Table V shows the value of the parameters entered on the program LABVIEW.

DEFINING PARAMETERS FOR THEORETICAL IMPULSE RESPONSE WITH Z = 0.2 M		
NAME	**VALUE**	**SUMMARY**
N	128	Size of square base array
M	106	Number of time slices
Step	3	Number of leading zero rows
c	1,500	Acoustic velocity in media (m/s)
z	0.2	Distance, source to receiver (m)
time_max	144e-6	Maximum time of propagation (s)
rho_max	629	Spatial radius of filters (length^{-1})
d	31	diameter of circle (samples)

Table IV. Summary of parameters used in Reid's program for separation distance of 0.2 m.

Figure 32 shows the measured pulsed response at the 32nd row obtained by means of the data acquisition system shown in Fig. 21. Figure 33 shows the theoretical pulsed response using the modified Reid's program with parameters shown in Table V. Comparison of Figs. 32 and 33 shows the similarity in trends and shapes between the two with a slight disagreement at the center negative peak. This could be due to several reasons.

DEFINING PARAMETERS FOR MEASURED PULSED RESPONSE AT Z =0.2 M		
NAME	**VALUE**	**SUMMARY**
Directory	d:\sonic2	Location of the collected data
Channel 1 range	50 mV	Full scale of the vertical acquisition window
Channel coupling	ac	Method of coupling: AC, DC, or OFF
Frequency	2,000,000	Pulse sine wave frequency
Amplitude	10	Pulse sine wave amplitude
Acquire interval	0.01 μs	Interval between waveform data samples
Acquire length	1,024	Record length
Trigger source	Extrenal	Source of trigger signal
Trigger coupling	ac	Coupling for the trigger event: AC, DC, of HF Reject
Trigger position	13,056	Sets the amount of pre-trigger or post-trigger

Table V. Summary of parameters entered on the LABVIEW program for the experimental data acquisition for z = 0.2 m.

The effect of the unparalleled scanning plane with the source plane has greater effect at the center. This is because the combination of pulsed responses from all other locations are not being combined constructively (out-of-phase). Theoretically, the combinations of the all other pulsed responses (with the exception of center pulse) at the center are combined constructively (in-phase). Parallelism between the scanning plane and the source plane is one of the major restrictions in using the angular spectrum approach.

Figures 34 and 35 are the absolute value of the figures shown in Figs. 32 and 33. Figures 34 and 35 shows the relative magnitude of the signal. The comparison of Figs. 34 and 35 shows that their peaks are again in good agreement with the exception of the center pulsed response. Figures 36, 37, 38 and 39 are the additional figures used to compare the theoretical and the measured pulsed response at four different radial locations similar to Figs. 28, 29, 30, and 31 of section A. Fig. 36, which is the comparison between the theoretical and the experimental pulsed response at the center location (r = 0 cm), appears to be in good agreement. Figures 37, 38, and 39 which are the comparison between the theoretical and the experimental pulsed response at other locations (r = 0.635 cm, r = 1.27 cm, and r = 1.905 cm) also appears to be very similar in relative magnitude and trends.

The comparison of the theoretical and the experimental pulsed response shows a general agreement. However, the slight dissimilarities between the theoretical and the experimental still exists. These discrepancies can be due to several reasons which are discussed in the next chapter.

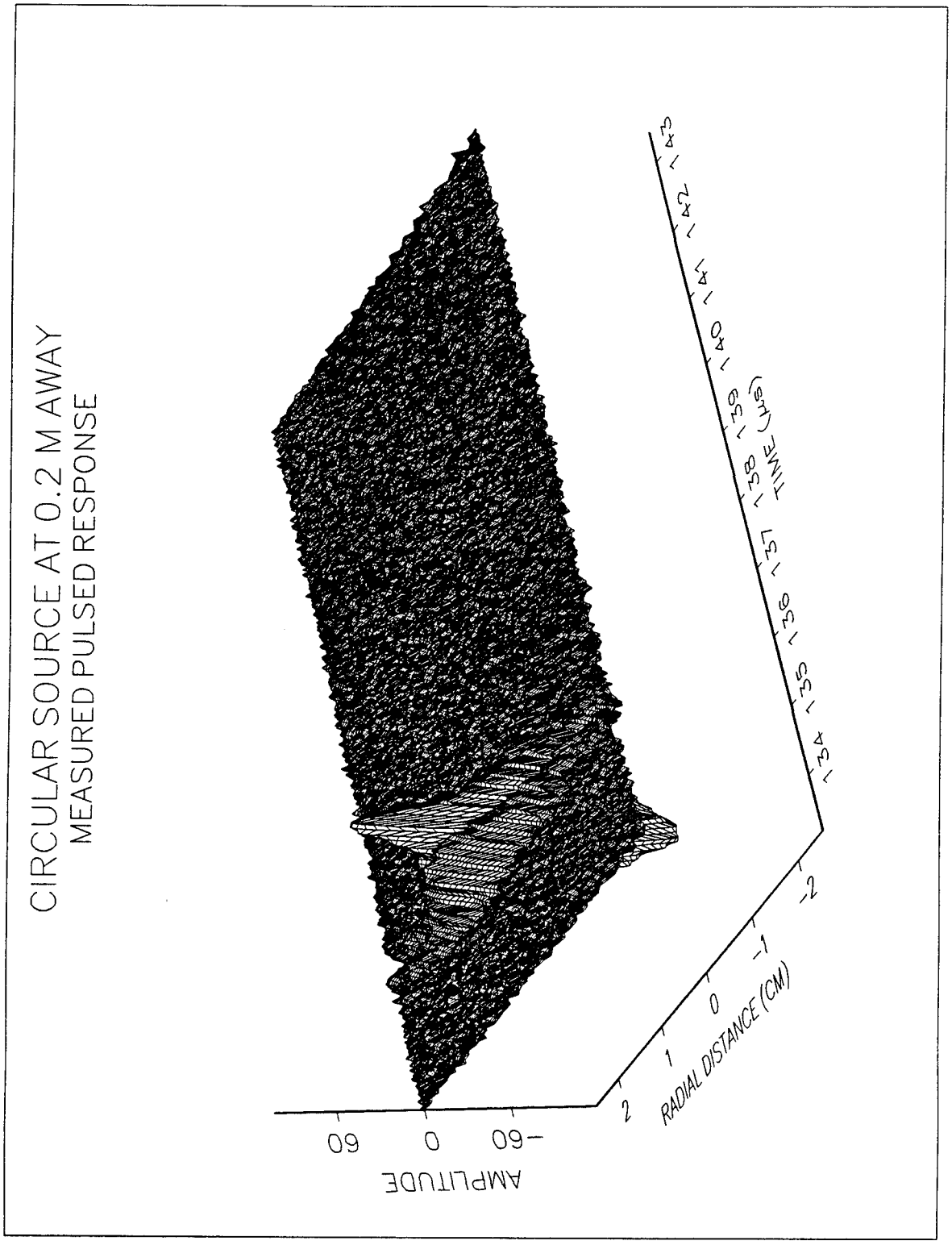


Figure 32. Measured pulsed response at z = 0.2 m. Circular source.

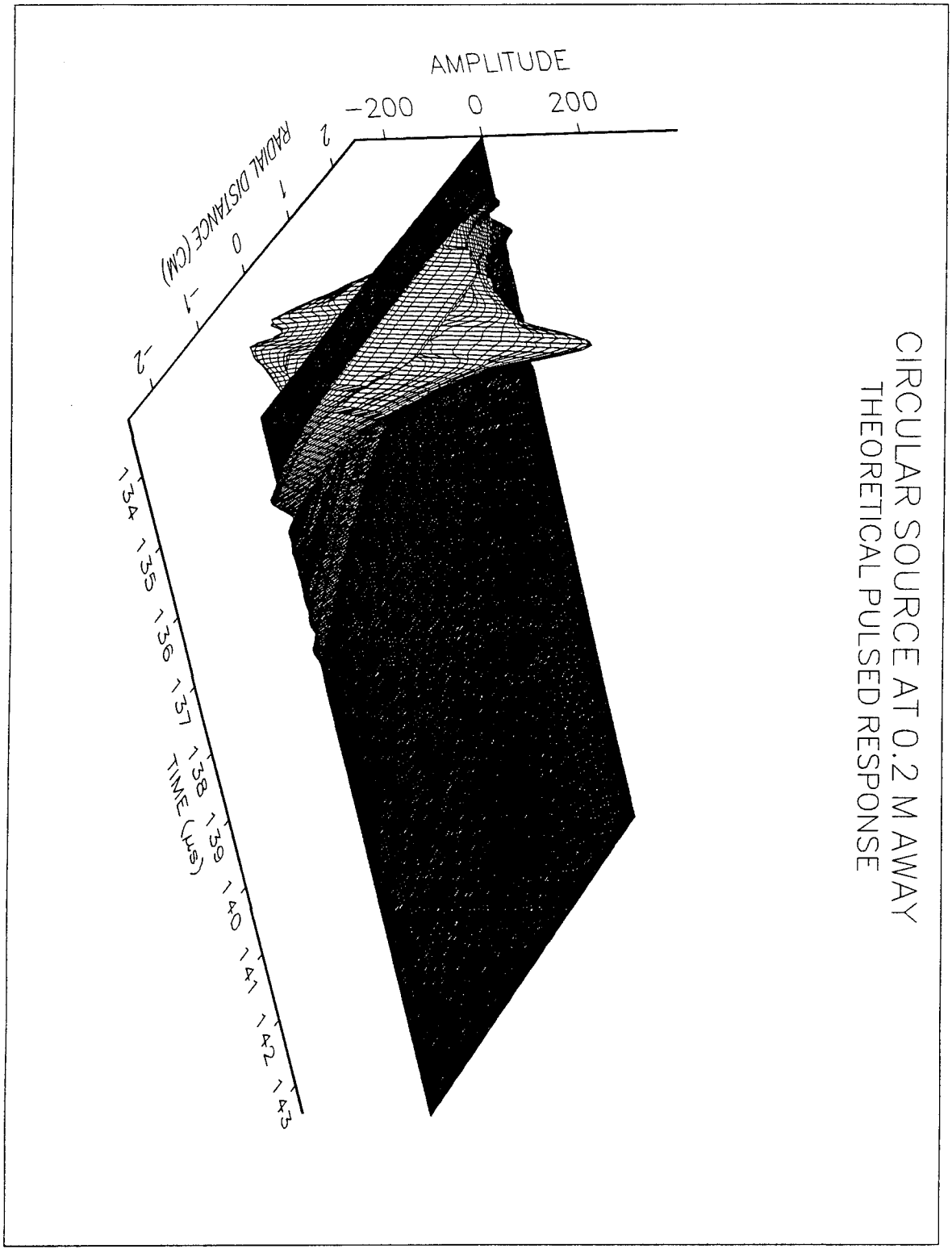


Figure 33. Theoretical pulsed response at z = 0.2 m. Circular source.

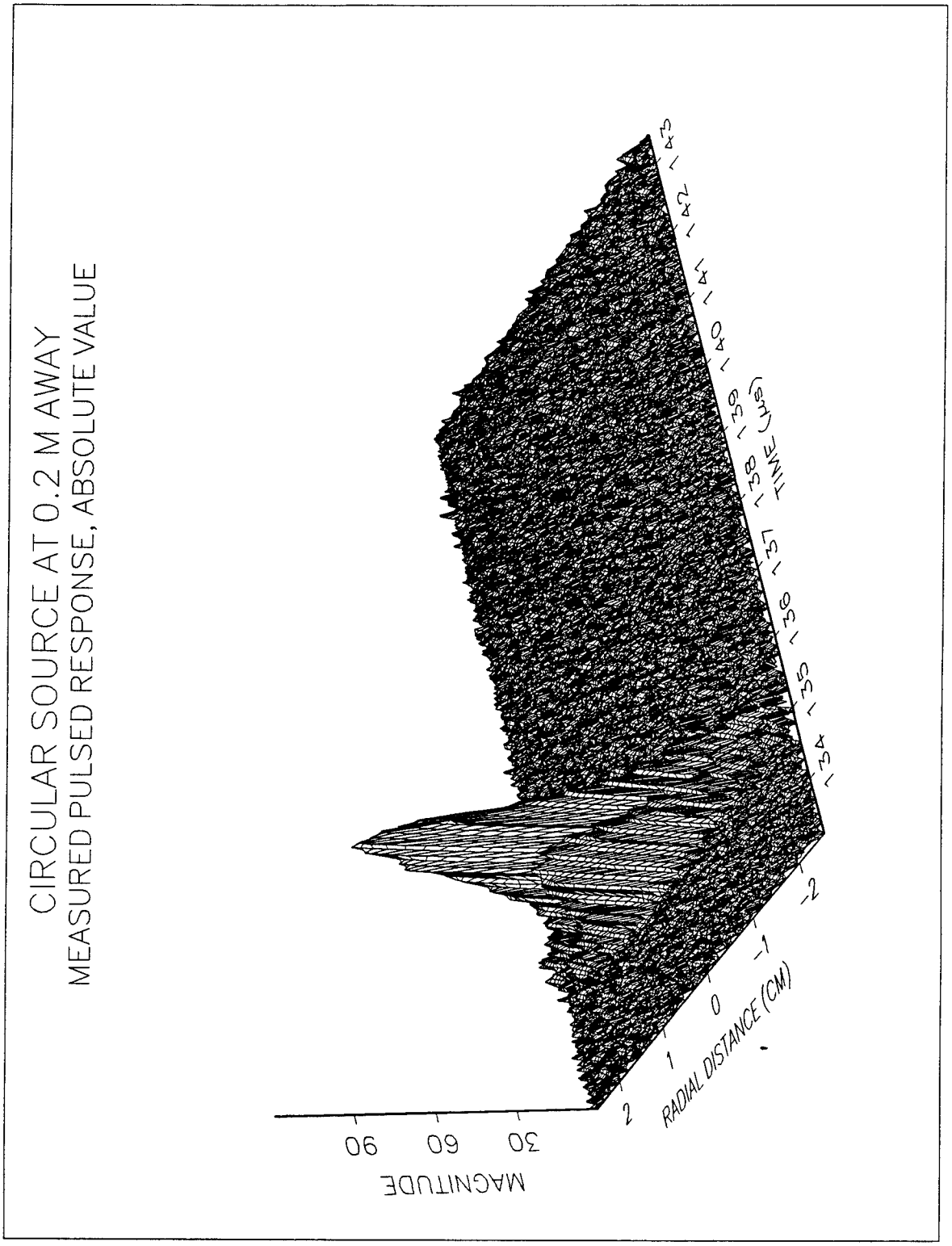


Figure 34. Measured pulsed response (absolute) at z = 0.2 m. Circular source.

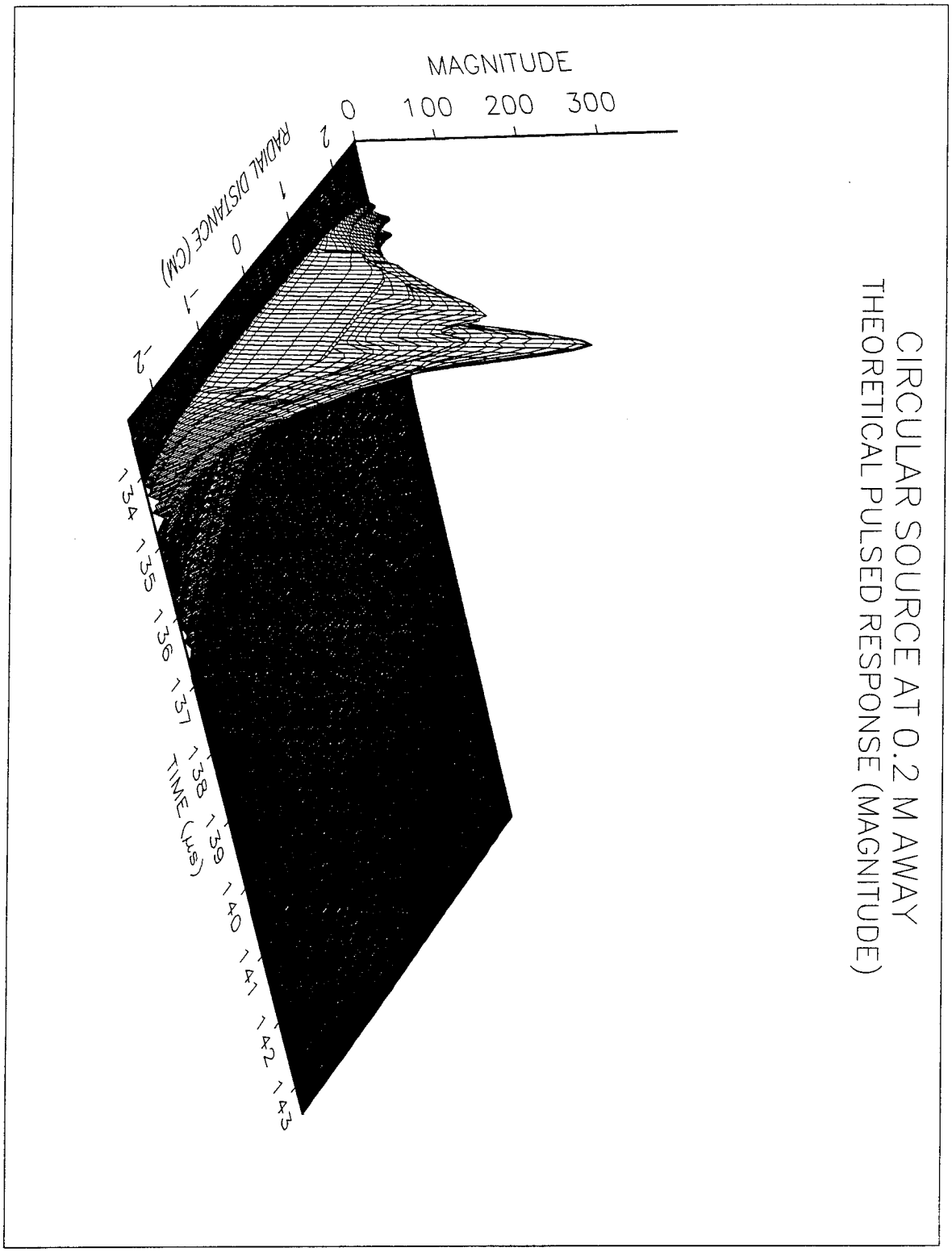


Figure 35. Theoretical pulsed response (Magnitude) at z = 0.2 m. Circular source.

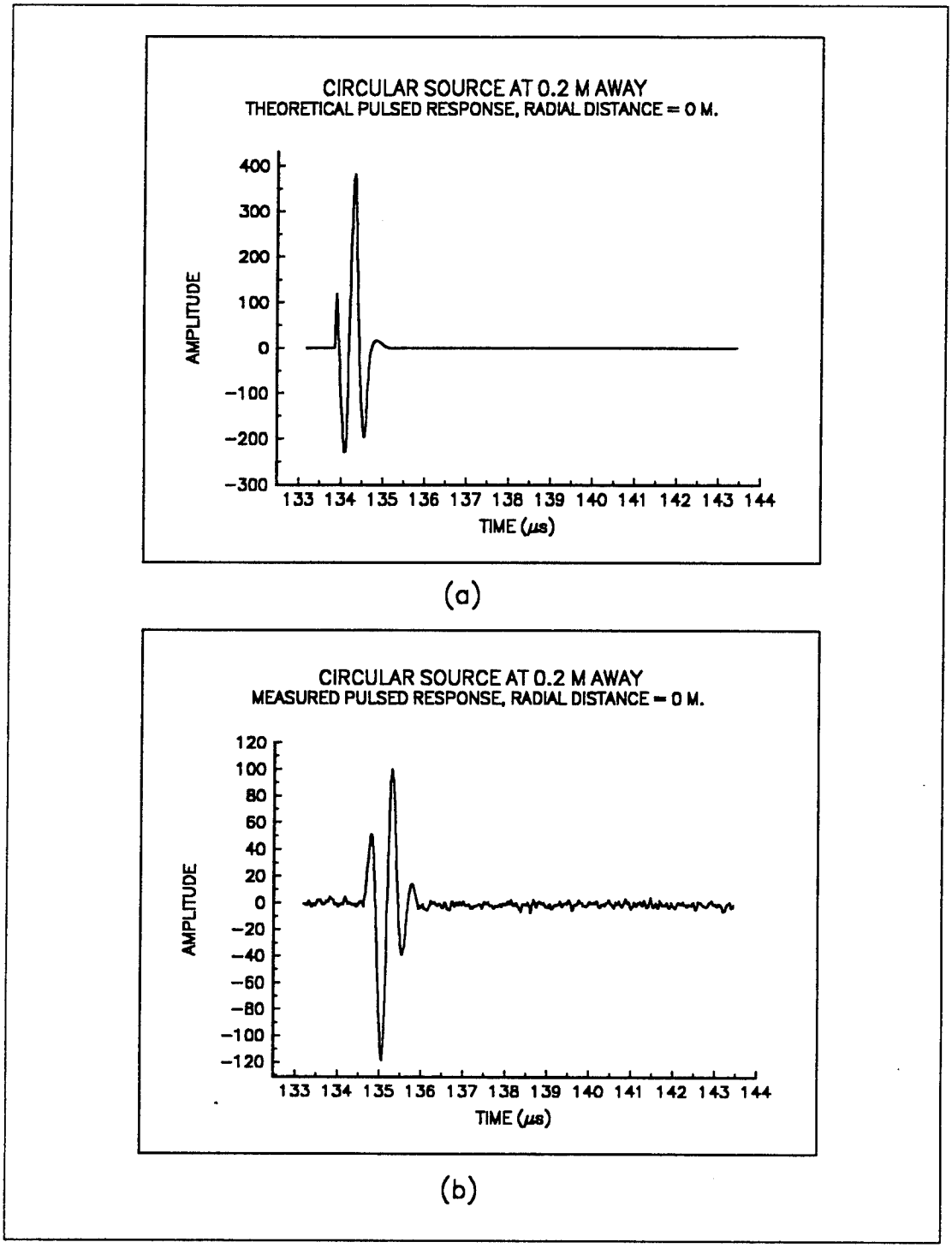


Figure 36. Comparison of the (a) theoretical and (b) measured pulsed response at a radial distance of 0 cm and a separation distance of 0.2 m.

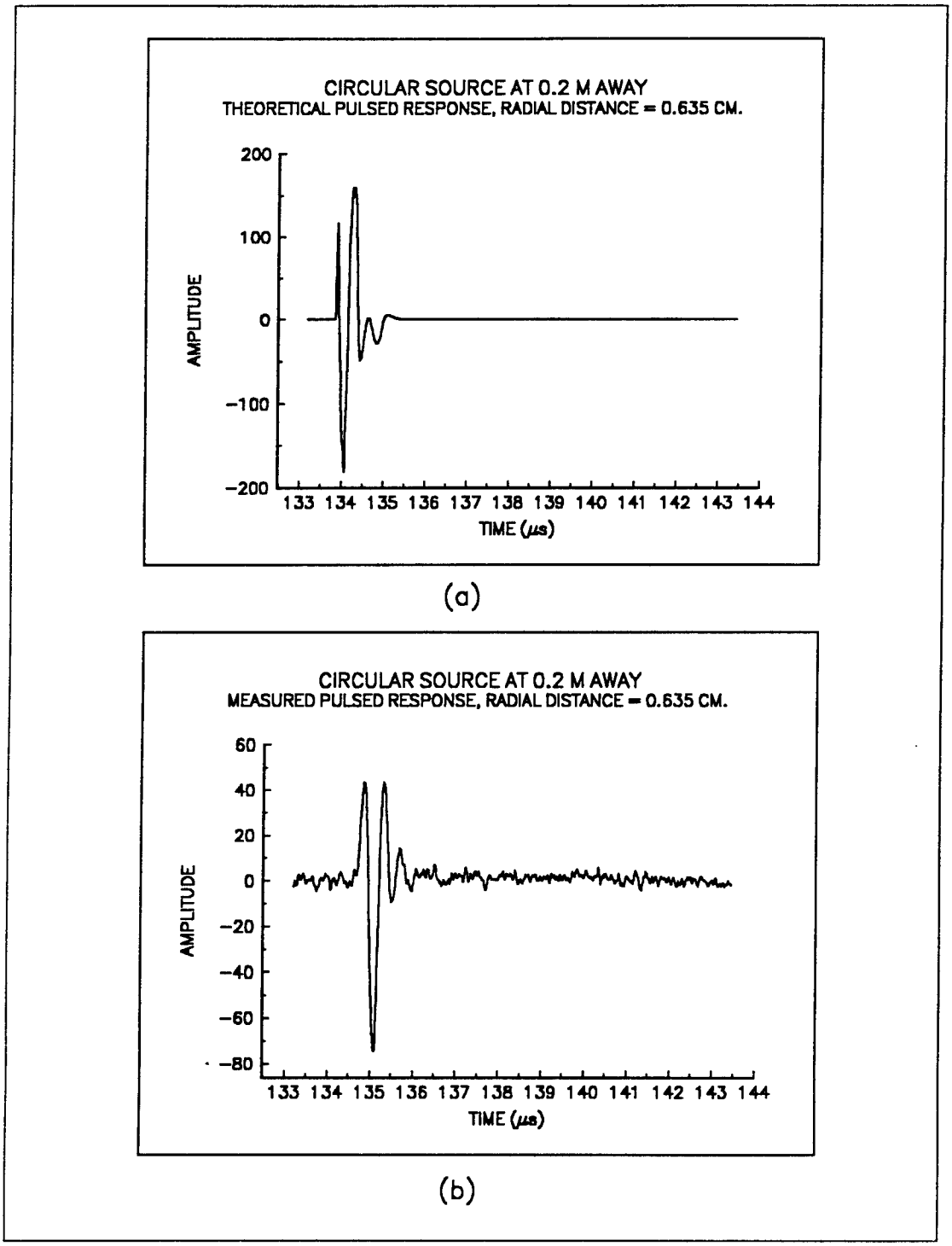


Figure 37. Comparison of the (a) theoretical and (b) measured pulsed response at a radial distance of 0.635 cm and a separation distance of 0.2 m.

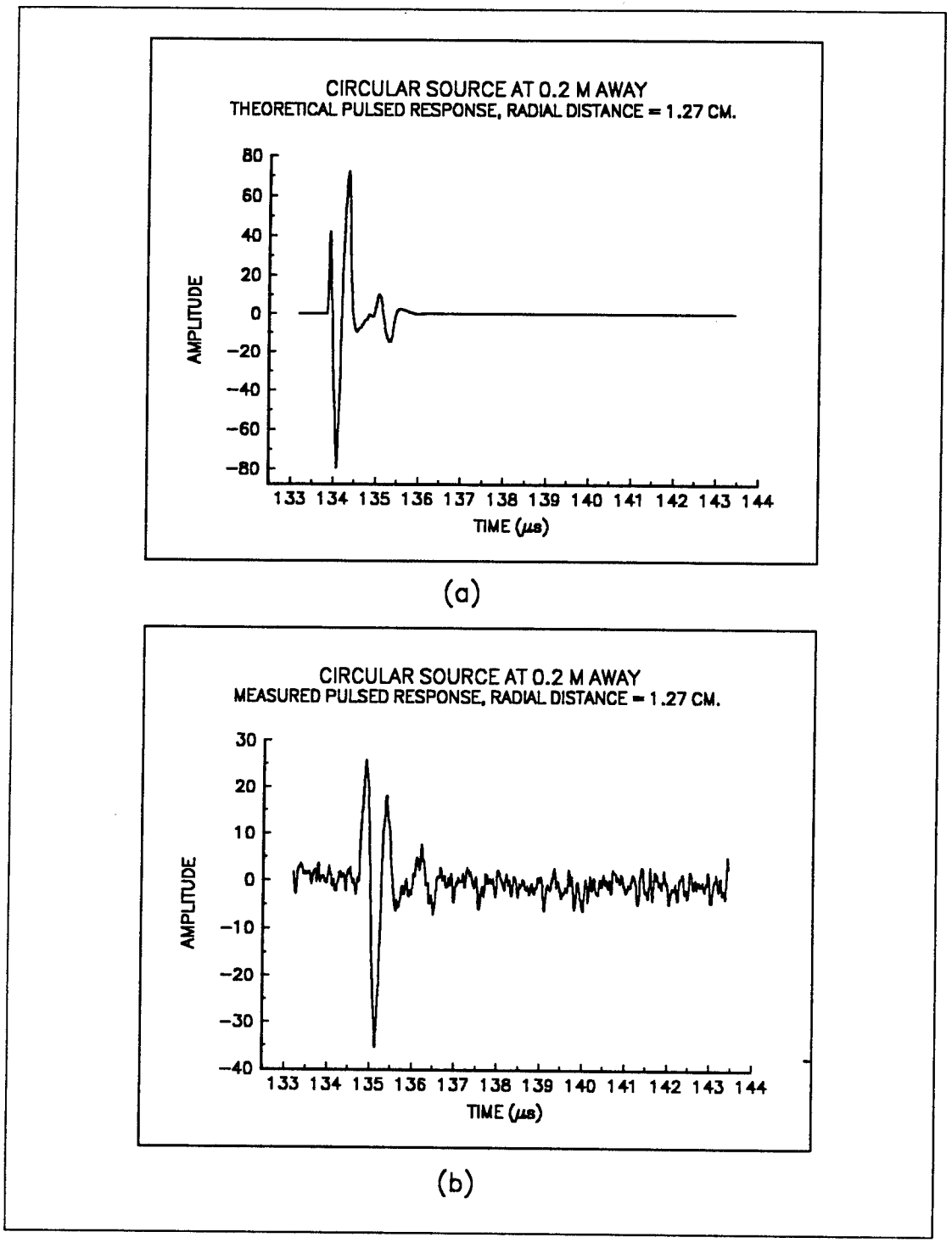


Figure 38. Comparison of the (a) theoretical and (b) measured pulsed response at a radial distance of 1.27 cm and a separation distance of 0.2 m.

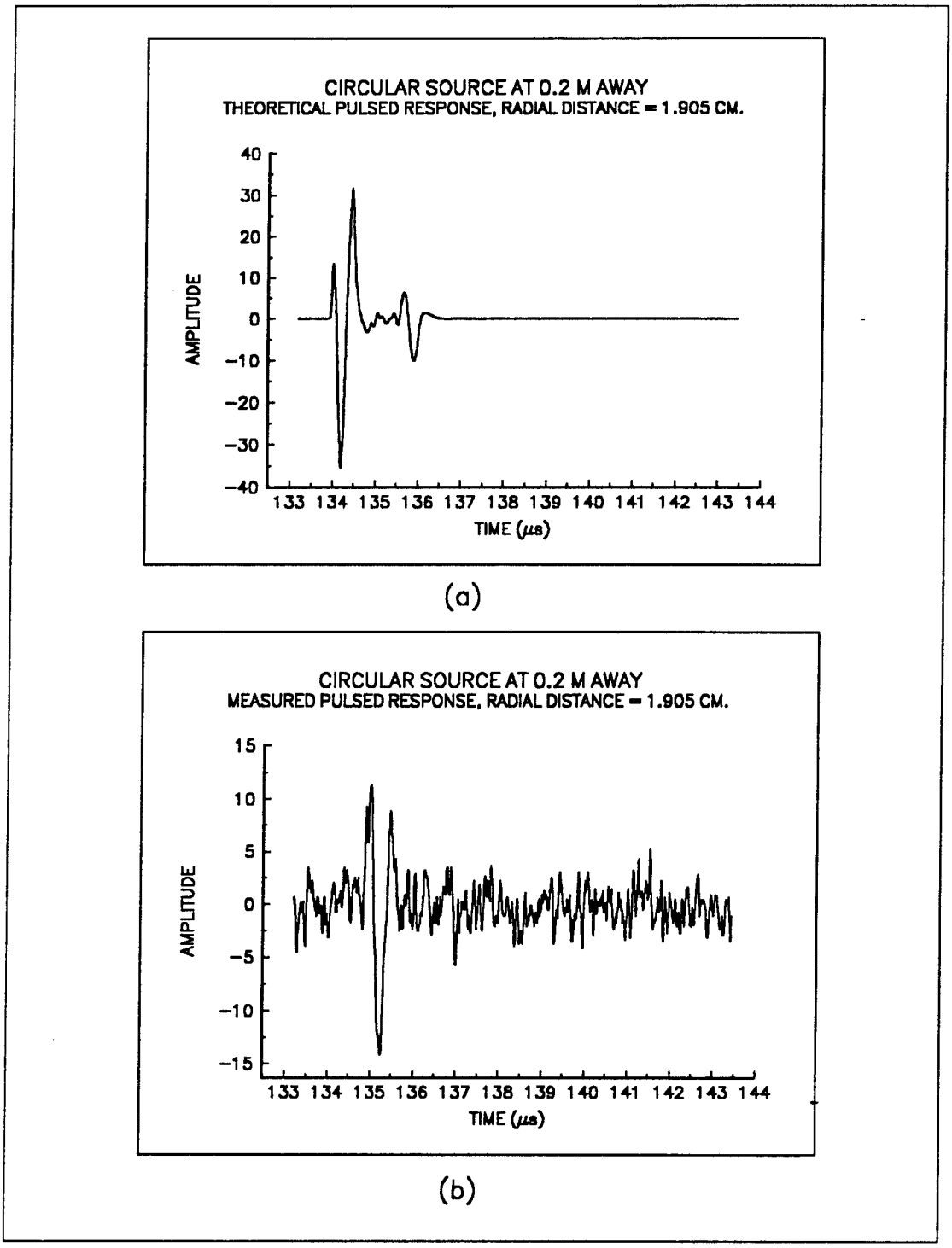


Figure 39. Comparison of the (a) theoretical and (b) measured pulsed response at a radial distance of 1.905 cm and a separation distance of 0.2 m.

V. SUMMARY AND CONCLUSIONS

This thesis presented a comparison between the theoretical pulsed response and the actual pulsed response. The initial goal of this thesis was to examine and correct the existing programs that implements a Fourier approach to ultrasonic wave propagation. In the problem description chapter of this thesis, it was shown that the existing program does contain a mathematical error. This error was isolated in the modeling of Green's function where the value of ρ was off by a factor of 2π. Once the program was corrected, the focus of this thesis shifted onto the hardware part of the existing pulsed ultrasonic data collection facility. The problem in the collection facility lies in the sensitivity of the receiver/transducer resulting in the poor signal-to-noise ratio of the collected signal. Modifications were made on the hardware set-up as shown in Chapter II, Section D to improve the signal-to-noise ratio of the collected data. Another improvement that could enhance the collection facility would be the ability to precisely control the plane axis of the receiver and transmitter plane to be perpendicular to the acoustic axis instead of manually moving the transmitter holder by hand. A slight misalignment in the plane axis led to the disagreement at the center point of the center axis pulsed response as shown in Chapter IV. This is due to the signal phase differential which led to the cancellation of signals at the center.

An experimental verification of the source code which models a pulsed ultrasonic wave propagation has been performed. After correcting the existing scaling error, it

shows that this program does provide an efficient means to analyze a pulsed ultrasonic wave propagation. Overall, the comparison between the theoretical and experimental pulsed response are in goo agreement with a slight disagreement at the center. There are several reasons that could have cause these dissimilarities. One major reason could be due to the misalignments between the source plane and the receiver plane. Using the available ultrasonic data collection facility, it was difficult to align the transmitter transducer plane perpendicular to the acoustic axis with manual hand alignment. The source transducer must be modified so that it can be adjusted precisely using two-axis precision gimbals. The next major contributor to the dissimilarities between the two could be related to the noise of the system and the sensitivity of the receiver. This can be improved by modifying the existing hardware set-up so that all components are impedance matched.

APPENDIX A. DETAILED EXPLANATION OF PREDICT.M

The following is a detailed explanation of the source code PREDICT.M. The source code for PREDICT.M is given in Appendix B. The MATLAB commands are written in lower case bold in this explanation while the actual variable of the source code are written in italics.

All lines leading with a symbol %% are comments. Other lines which start off with a symbol %*% are meant to be an optional instruction which can be activated by simply removing them. The first non-comment line of PREDICT.M is a **clear** which removes all variables from the random access memory (RAM) of the computer. It is followed with a three input command functions requiring the user to enter the separation distance, the radius of the active element of the acoustic source, and the radial distance of the acoustic receiver with respect to the acoustic axis. The value of c is made to be 1500 m/s. The next task is to generate a time vector using the value entered for the separation distance and the speed of sound in the medium. The maximum time of propagation is made to be 470 times 0.01 μs with an interval of 0.01 μs. This can be changed by simply changing the value from 470 to the desired length. The next line after creating the time vector is the definition for the parameter *tau* which is used in Eqs. 10 and 11. Next is the definition of the parameter R_1 and R_2 shown in Eq. 10. The next forty-three lines model Eqs. 10 and 11 using **if, elseif,** and **else** statements to control branching and conditional statements. The first conditional statement of **if r/a <= 1** tests whether the acoustic receiver is inside the

active element with respect to the acoustic axis. If the answer to the conditional statement is a non-zero, then the program computes the impulse response using Eq. 10. Otherwise, the program uses Eq. 11 to compute the impulse response. After the impulse response is solved for each time unit, then the program plots the impulse response with respect to the time vector. The remainder of the program saves the time vector and the impulse response vector in ASCII format so that it can be used in other programs for presentation.

APPENDIX B. SOURCE CODE FOR PREDICT.M

The following is the source code to predict the impulse response of the system. This code was essential to predict the impulse response time duration of a given system. The code includes some optional instructions which are indicated by the leading %*% symbol. Removing this symbol will activate the instruction while leaving the symbol simply turns the code into comments.

PREDICT.M

```
%%%%%%%%%        ****    PREDICT.M    *****
%% This program predicts the impulse response of a circular
%% source transmitter to verify the inaccuracy of Reid's
%% impulse response program modeling.  It also verify's
%% the accuracy of Reid's modified program
%%      Benito E. Baylosis          December 1994
%%
%%
!clear                              % Remove all variables from RAM
z = input('Enter z >>  ');          % distance between transmitter
                                    % and receiver in meters

a = input('Enter radius of transmitter >> ');
                                    % active region of the transmitter
                                    % in meters
%% Enter the desired pulse to be verified in meters (min=0, max=a)
disp('Enter the transmitter position with respect to radial distance');
r = input('                          >> ');

c = 1500;                           % Speed of sound in the medium

%% generate time vector with an interval of 0.01e-6 starting at
%% 3 times the time interval before the first wave get to the
%% transmitter and the maximum at the 470 times the time interval
%% after the first wave got to the transmitter
```

```
t = (z/c)-(3*0.01e-6):0.01e-6:(z/c)+(470*0.01e-6);

tau = (c*t)/a;

R1=((z/a).^2 + ((r/a)-1).^2 ).^0.5;    % Minimum distance
R2=((z/a).^2 + ((r/a)+1).^2 ).^0.5;    % Maximum distance

if r/a <= 1
 if R2 == R1                    % center pulse with r=0;
                                % then receiver must be at the center

   for i=1:1:length(tau)
     if tau(i) <  (z/a)         % before the first wave arrival
        h(i) = 0;
     elseif tau(i) > R1         % after the last wave arrival
        h(i) = 0;
     else
        h(i) =1;
     end
   end

 elseif R2 > R1
   %% the transmitter is off center;

   for i=1:1:length(tau)
     if tau(i) <  (z/a)
        h(i) = 0;
     elseif tau(i) >= R2
        h(i) = 0;
     elseif  (tau > (z/a))  & (tau < R1) ;
        h(i) = 1;
     pause;
     else
        ((tau >= R1) & (tau <= R2));
        num(i) = (tau(i)).^2 - (z/a).^2 + (r/a).^2 -1;
        denom(i) = 2*(r/a)*((tau(i)).^2-(z/a).^2).^0.5;
        h(i) =(1/pi)*(acos(num(i)/denom(i)));
     end;
   end;

end;
```

```
else
    for i=1:1:length(tau)
      if tau(i) <= R1                          % before the first wave arrival
         h(i) = 0;
      elseif tau(i) > R2                 % after the last wave arrival
         h(i) = 0;
      else
        %% ((tau >= R1) & (tau <= R2));
         num(i) = (tau(i)).^2 - (z/a).^2 + (r/a).^2 -1;
         denom(i) = 2*(r/a)*((tau(i)).^2-(z/a).^2).^0.5;
         h(i) =(1/pi)*(acos(num(i)/denom(i)));
      end;
    end;

end;

plot(t,real(h));
title ([' Impulse Response at ',int2str(r*1000),' mm radial distance']);
xlabel(' Time (microsec)');
ylabel('Amplitude');

%% Save the time vector in  ASCII
   t=(t*1e6)';
   Time_name = ['T',int2str(z*10),'x',int2str(r*1000),'.dat'];
   eval(['save ',Time_name,' t -ascii'] );
   %*%eval(['save a:\ ',Time_name,' t -ascii'] );

%% Save the impulse response h in ascii
   h=h';
   Impulse = ['Imp',int2str(z*10),'x',int2str(r*1000),'.dat'];
   eval(['save ',Impulse,' h -ascii'] );
   %*%eval(['save a:\ ',Impulse,' h -ascii'] );
```

APPENDIX C. DETAILED EXPLANATION OF CONVOL.M

The following is a detailed explanation of the source code CONVOL.M. The source code for CONVOL.M is given in Appendix D. The MATLAB commands are written in lower case bold in this explanation while the actual variable of the source code are written in italics.

All lines leading with a symbol %% are comments. Other lines which start off with a symbol %*% are meant to be an optional instructions which can be activated by simply removing them. The first non-comment line of CONVOL.M is a **clear** which removes all variables from the random access memory (RAM) of the computer. The first two lines are the definition of the speed of sound (c) in the medium which is equal to 1500 m/s and the number of time slices (N) which is equal to 1024. The fourth non-comment line is an input command function prompting the user to enter the separation distance. If the user press the return key without entering a value, the program uses the default value of 0.1 m. The next task includes the loading of the theoretical input and the theoretical impulse response. The theoretical input and the theoretical impulse response must reside in the same directory as the program. If they do not reside in the same directory, then the user must alter the program so that the **load** command specifies the specific directory where the files are located. Since the impulse response was saved from Reid's AC_PROP.M with matrix dimension of M x $N/2$ where M is the time slice and N is the size of base array, the impulse response matrix must be transposed. This was accomplished by the transpose

command indicated by a " ' " on **ar31x430 = ar31x430 '.** Following the transposition of the theoretical impulse response is a **for** loop command which performs the convolution between the theoretical input and the theoretical impulse response. Since the impulse response is given in a three-dimensional matrix and the theoretical input is a two-dimensional matrix, an impulse response at a given radial distance must be extracted in order to convolve each pulsed response with the theoretical input. This was accomplished by combining a string matrices to form the a single impulse response at a given radial distance. Radial distances are broken down into 64 positions. Therefore, a single impulse response can be designated by this command line **data_var = ['ar31x430(',int2str(n),',:)]** where *n* loops from 1 to 64. When *n* is equal to 1, data_var equals to *ar31x430(1,:).* Another variable is formed using the same technique to create a new variable where the result of the convolved data will be placed. This is shown in the line that reads **newdata = ['result(',int2str(n),',:)'].** When *n* is equal to 1, newdata is equal to *result(1,:).* The actual convolution is performed using the command **eval. Eval** is a function within MATLAB that works with strings matrix. In a command line that reads **eval ([newdata,'=conv(imp,',data_var,');']),** the string matrix and the string stored in the variable *newdata* and *data_var* are to be issued as a MATLAB command. When the loop is completed, the variable *result* will hold the result of the convolved impulse response which is referred to as the theoretical pulsed response in this thesis. Note that an optional command to view each convolved impulse response is included. The remaining command of the program deals with generation of time vector and saving

the value of the variable *result* is an ASCII format so that it can be used in other program for presentations. Note that the matrix stored in a variable *result* is reduced to a length of 1024 so that it can be compared with the actual pulsed response with the same time durations. The command line **result2 = abs (result)** takes the absolute value of the variable *result.* The absolute value is used in the comparison of the theoretical and the actual pulsed response. The command line **result1 = [result2 time]** combines the matrix stored in the variable *result2* with the variable stored in *time* for ease of loading in the program AXUM. The value of the variable *result, result1,* and *result2* are all saved using the command eval. The optional command of viewing the three-dimensional plot of the variable *result2* is included at the end of the program.

APPENDIX D. SOURCE CODE FOR CONVOL.M

The following is the source code to convolve the impulse response of the system at a given distance and transmitter with the theoretical input. The code includes some optional instructions which is indicated by the leading %*% symbol. Removing this symbol will activate the instructions. Leaving the %*% symbol simply turns the instructions into a comments.

CONVOL.M

```
%%%%%%          **** CONVOL.M ****
%% This program performs convolution of the system impulse response
%% and the actual system input.  It uses the output_plot produced
%% in AC_PROP.M to be convolved with the near-theoretical input
%% signal.
%%      Benito E. Baylosis                  December 1994
%%
%%
clear;                          % Remove all variables from RAM.

c = 1500;                       % Velocity of acoustic wave, (m/s).
M = 1024;                       % Number of time slices.

disp(' Please enter the separation distance in meter:');

 z = input(' >>>  ');           % Separation distance.
   if isempty(z)
       z = 0.1;                 % Default value, 0.1m.
   end

 %% load the actual system input
    load  input.dat;        % Theoretical input signal
```

```
 %% load the impulse response
    load ar31x430.dat          % System impulse response
    ar31x430=ar31x430';        % Transpose
%%%%%%%%%%% START  LOOP %%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    for n=1:1:64;              % all 64 column pulses
      disp(' 1.. 64 pulses');
      disp('n =  ');
      n
      imp=(input);             % Set imp equals the input

     %% Impulse response data variable for each pulse
      data_var =['ar31x430(',int2str(n),',:)'];
      %% Convolved result variable for each pulse
      newdata =['result(',int2str(n),',:)'];

     %% Perform the convolution
     eval([newdata,'= conv(imp,',data_var,');']);

     %% An optional plot for each pulse
     %*% eval(['plot(abs(',newdata,'));']);
     %*% pause(2);
    end
%%%%%%%%%%%%%%%%%%% END  LOOP %%%%%%%%%%%%%%%%%%%%%%%%

%% time interval of the convolved response
   delta_t = 0.01e-6;

%% Create a time matrix to be added at the last column for used
%% in other program like AXUM
   time_name = ['Ta',int2str(z*10),'x',int2str(n),'.dat'];

%% time matrix
   start = (z/c)-(13*delta_t);
   time = start:delta_t:start+((M-1)*delta_t);

%%  Convert the time matrix in microseconds
   time = (time/1e-6)';     %% in microseconds

%% Save the real value of time to be used on other graphic program
  eval(['save ',time_name,' time -ascii'] );
```

```
%% Reduce the time length of the convolved matrix to match the
%% actual time length
   result =result';
   result = result(1:1024,:);

%% Take the absolute value of output_plot for display
  result2 = abs(result);
%% Attach the time matrix as the last column of output_plot2 to be used
%% on AXUM
  result1 = [result2 time];

%% Save the contents of "result"  as an ASCII  file.
  %% Actual result in ASCII
  rname1 = ['NX',int2str(z*10),'x',int2str(n),'.dat'];
   %% Actual result with time matrix attached at the last column
   rname2 = ['NY',int2str(z*10),'x',int2str(n),'.dat'];
   %% Absolute value of result
   rname3 = ['NZ',int2str(z*10),'x',int2str(n),'.dat'];


%% Save the actual value of the result.
  eval(['save ',rname1, ' result  -ascii'] );

%% Save the actual value of the result with time matrix attached
%% at the last column to be used on other graphic programs like AXUM
  eval(['save ',rname2,' result1 -ascii'] );

%% Save the absolute value of result
   eval(['save ',rname3,' result2 -ascii'] );

%% An optional plot of three views of "result2."
%*% subplot(121), mesh(rot90(result2,1));
%*% subplot(122),.mesh(rot90(result2,2));
%*% pause(2); clg;
%*% mesh(rot90(result2,3)),title('THEORETICAL RESPONSE');
%*% pause(3);
```

APPENDIX E. DETAILED EXPLANATION OF AVG.M

The following is a detailed explanation of the source code AVG.M. The source code for AVG.M is given in Appendix F. The MATLAB commands are written in lower case bold in this explanation and the actual variable of the source code are written in italics.

All lines leading with the symbol %% are comments. Other lines which start off with a symbol %*% are meant to be an optional instruction which can be activated by simply removing them. The first non-comment line of AVG.M is a **clear** which removes all variables from the random access memory (RAM) of the computer. Following the **clear** command are four **for** loop commands. The first **for** loop loads all raw data file starting from data1.mat to data64.mat. In this thesis, there are 64 pulses within a row and there are 64 data rows within the experiment. The purpose of the command line that reads **for n=1:1:64** is to load each data file containing 64 data pulses. These are the data pulses which have not been averaged and are the result of the program DATO2.CPP. The loading of files is accomplished by creating a string vector which changes as n changes as in the command line **data_name =['data',int2str(n)]**. A new string vector was also created so that the averaged data can be saved on this new variable. This is done in the command line **avgdat =['avgdta1,int2str(n)]**. The actual loading of the file was accomplished using the **eval** command. The **eval** command can manipulate text strings so that it is executed as a MATLAB command. In the command line

eval(['load',data_name,'.mat;']), the string stored in the variable *data_name* with an extension of .mat is loaded onto the memory. In this AVG.M 's program case when $n = 1$, the file *data1.mat* is loaded. Another string vector was created since the variable in the files that are loaded have a different name. To demonstrate this point: when the file data1.mat is loaded, the actual variable within that file is DATA1. The next three lines simply inform the user about the progress of the program and that the program should not be interrupted until finished. The second **for** loop separates each pulsed response which corresponds to the number of square base array (*M*) in Reid's AC_FIL.M. Another string vector was created in this loop so that each averaged pulsed is separated into a single variable *y* as in the command line that reads **y =[avgdat,'(:,',int2str(i),')']**. The raw data was also separated into a single pulsed. The next task is to define some variables which are used in the program. *Start* was set to equal to 1, *out* was set to equal to a single pulsed raw data, *stop* was set to equal to a length of variable *out,* and *long* was set to equal to the size of variable *out*. Another **eval** command was used to initialize the string stored in the variable *y* as zero. This initialization saves processing time. The actual data averaging is another **for** loop which start at the value of variable *start* and stops at the value of variable *stop*. In this thesis *start* is equal to 1 and *stop* is equal to 1024. The third **for** loop keeps track of the data index within the pulsed response starting from 1 to 1064. The last **for** loop keeps track of the weight of the averaging and performs additions of all data as shown in command line **for b=a-3:a+3**. The number 3 in this loop command gives the number of data required to be added. In this case, 3 data values to the left of

the current data index, 3 data values to the right of the current index and the data on the current index are all added. The added data values is divided by the number of data added after each **for** loop (which is 7 in this case). When the averaged data is acquired for that index, the index is incremented by 1 to get the next averaged data. This continues until all 1024 data points for that pulsed response are averaged. When all the pulsed responses for the current row are averaged, they are saved under one file name and the next row pulsed response is loaded and averaged. The remainder of the program is optional information given to the user about the progress of the program.

APPENDIX F. SOURCE CODE FOR AVG.M

The following is the source code to predict the impulse response of the system. This code was essential to predict the impulse response time duration of a given system. The code includes some optional instructions which is indicated by the leading %*% symbol. Removing this symbol will activate the command line.

AVG.M

```
%%%%%          **** AVG.M ****
%% This program performs averaging of the system pulsed response
%%
%%     Benito E. Baylosis                December 1994
%%
%%
clear;                                    %Remove all variables from RAM.

%%%%%%%%%%%%%  START FIRST LOOP  %%%%%%%%%%%%%%%%%%%%
%% Load each data file from data1.mat to data2.mat

for n=1:1:64;                             %% for all data row 1 .. 64
  data_name = ['data',int2str(n)];        %% data1 .. data64
  avgdat = ['avgdta1',int2str(n)];        %% New averaged data

  eval ([ 'load ', data_name,'.mat;']);   %% load the data from the
                                          %% memory

  data_var =['DATA',int2str(n)];          %% data variable DATA1 .
                                          %% DATA64
disp('  This will take a while.  Please do not interrupt !! ');
disp('n= ');
disp(n);


%%%%%%%%%%%%%% START SECOND LOOP %%%%%%%%%%%%%%%%%%%%
```

```
%% Separate Each 64 pulsed response from a single data file
  for i=1:1:64;

    %%%% load each column vector with 64 pulses  %%%%%

    y = [avgdat,'(:,',int2str(i),')'];          %% new averaged data
                                                %% for each column pulse

    %% separate each data row into 64 data pulse
    eval(['data_time_col =',data_var,'(:,',int2str(i),');']);


     start=1;
     out = data_time_col;                       %%Put variable name here
     stop=length(out);
     long=size(out);
     b=1;
     eval([y,' = zeros(long);']);

%%%%%%%%%%%%% START THIRD LOOP %%%%%%%%%%%%%%%%%%%%
%%  Start the index count of each pulsed response

         for a=start:stop;
%%%%%%%%%%%%% START FOURTH LOOP %%%%%%%%%%%%%%%%%%%%
%% Perform the data averaging

       for b=a-3:a+3,
         if b < 1,
           b= b+stop;                                %%Wrap around
         end
         if b > stop,
           b=b-stop;                                 %% Wrap around
         end
%% Inform the user of the progress of the program
    %*%     disp('a= ');
    %*%     disp(a);

        ya = [avgdat,'(',int2str(a),',',int2str(i),')'];
       outb =['out(',int2str(b),')'];           %% Extract neighboring data
        eval([ya,'=',ya,'+',outb,';']);         %% Add all data
      end
%%%%%%%%%%%%% END FOURTH LOOP %%%%%%%%%%%%%%%%%%%%
```

```
    eval([ya,'=',ya,'/7;']);            %% Averaged the added data

%% Inform the user the progress of the program
 %*%        if ~rem(a,200),
 %*%        S=sprintf('there are %d points left to average\r', (length(out)-a) );
 %*%        disp(S)
 %*%   end

 %*%     disp('i=');
 %*%     disp(i);

     end
%%%%%%%%%%%%%%%%%END THIRD LOOP %%%%%%%%%%%%%%%%%%%

 %% eval([avgdat,'(:,',int2str(i),')=',y]);
 %% eval(['plot(',y,');']);
 %% clear y;

%*%disp(i)

 end
%%%%%%%%%%%%%%% END SECOND LOOP %%%%%%%%%%%%%%%%%%%
%% Clear the previous DATA from the RAM

 eval (['clear ',data_var]);
%% Save the averaged data
 eval(['save avg6_',int2str(n),' ',avgdat]);
 eval(['clear ',avgdat]);
 disp(n);
end
%%%%%%%%%%%%%%%%%% END FIRST LOOP %%%%%%%%%%%%%%%%%%
```

APPENDIX G. SOURCE CODE FOR DATO2.CPP

The following C++ source code is used to convert a text format data sets to a MATLAB format data sets. This code generates a DATO2.EXE when compiled. DATO2.EXE is then used to convert the text format data sets to convert them into a MATLAB format. The source code DATO2.CPP was developed by Ray van deVeire of the Naval Postgraduate School.

DATO2.CPP

```
//   **** DATO2.CPP *****
// This program convert data sets in text format to a MATLAB data format
// in order to manipulate data in a MATLAB program.
//
//  Developed by: Ray van deVeire          Naval Postgraduate School
//  Used by: Benito E. Baylosis            December 1994
//
//
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <alloc.h>
#include <iostream.h>   //cerr, endl

FILE* fp, *outfp;
unsigned int huge data_array[16500];
unsigned char huge  buf[16500];
int plength;
char  zxc;

typedef struct {
     long type;     // type
     long mrows;    // row dimension
```

```
    long ncols;   // column dimension
    long imagf;   // flag indicating there is an imaginary part
    long namelen; // length of the file name
} Fmatrix;

Fmatrix x= {40, 512, 32, 0, 1};

int main(int argc, char* argv[])
{

char matrix_name[9], input_file[12], output_file[12];
char* ptr, c;

for (int fileNo = 1; fileNo < argc; fileNo++)
{
    //Setup file and matrix names
    strcpy(input_file, argv[fileNo]);
    strcpy (output_file, input_file);
    ptr = strchr(output_file, '.');
    if (ptr) *ptr= '\0';
    strcpy(matrix_name, output_file);
    strcat (output_file, ".MAT");
    x.namelen = strlen(matrix_name);

    if ( (fp = fopen(input_file, "rb") ) == NULL)
    {
            cerr << "Cannot open input file " << input_file << endl;
            return 1;
    }

    if ( (outfp = fopen(output_file, "wb") ) == NULL)
    {
            cerr << "Cannot open output file." << endl;
            return 1;
    }


/*  WRITE MATLAB HEADER ***/
    int num_bytes = fwrite(&x, sizeof(Fmatrix), 1, outfp);  // MATLAB file header data
    num_bytes = fwrite(matrix_name, sizeof(char), x.namelen, outfp);

/* REWIND AND THEN READ PAST THE GPIB HEADER DATA ***/
/* READ IN DATA ONE ROW AT A TIME ***/
```

```
    while ( (fread(buf, 1, 524, fp) == 524) )
    { //fread(void *ptr, size item, # items, FILE *stream);
            cout << "\n\nRead in " << num_bytes << " data bytes." << endl;
            for (int j=11; j<524; j++)
            {
                    /*scale data and transfer to float buffer */
                    data_array[j-11] = int(buf[j]);
            }

            num_bytes = fwrite(data_array, sizeof(unsigned int), 512, outfp);
            cout << "Wrote "<< num_bytes << " points from to " << output_file <<
endl;
    }


    int ok = fcloseall();
    cout << "operation completed, files closed = " << ok <<endl;
}
return 1;
}
```

LIST OF REFERENCES

1. Marvin C. Ziskin, "The angular spectrum method of transducer characterization", *Ultrasonic Exposimetry*, Chapter 9, pp. 257-282, CRC Press, Inc., Philadelphia, PA 1993.

2. J. W. Goodman, *Introduction to Fourier Optics*, pp. 48-54, McGraw-Hill Inc., New York 1968.

3. P. R. Stephanishen, "Transient radiation from pistons in an infinite planar baffle," *Journal of Acoustical Society of America*, vol. 49, no. 5, pp. 1629-1637 (1971).

4. W. H. Reid, *Microcomputer Simulation of a Fourier Approach to Ultrasonic Wave Propagation*, Master's Thesis, Naval Postgraduate School, Monterey, California, December 1992.

5. AXUM, TECHNICAL GRAPHICS AND DATA ANALYSIS, Transformation and Programming Language, TriMetrix, Inc.,Seattle, WA, March 1993.

6. D. Guyomar and J. Powers, "A Fourier approach to diffraction of pulsed ultrasonic waves in lossless media" *Proceedings of the 1985 IEEE Ultrasonics Symposium*, B. R. McAvoy, Ed.: IEEE Press, New York, pp. 692-695 (1985).

7. D. Guyomar and J. Powers, "Boundary effects on transient radiation fields from vibrating surfaces" *Journal of Acoustical Society of America*, vol. 77, no. 3, pp. 907-915 (1985).

8. D. Guyomar and J. Powers, "A Fourier approach to diffraction of pulsed ultrasonic waves in lossless media" *Journal of Acoustical Society of America*, vol. 82, no. 1, pp. 354-359 (1987).

9. P. R. Stephanishen, "Acoustic transients in the far-field of a baffled circular piston using the impulse response approach," *Journal of Sound and Vibration*, vol. 32, no. 3, pp. 295-310 (1974).

10. P. R. Stephanishen, "Acoustic transients from planar axissymetric vibrators using the impulse response method," *Journal of Acoustical Society of America*, vol. 70, no. 4, pp. 1176-1181 (1981).

11. P. R. Stephanishen and G. Fisher, "Experimental verification of the impulse response method to evaluate transient acoustic fields," *Journal of Acoustical Society of America*, vol. 69, no. 6, pp. 1610-1617 (1981).

12. G. R. Harris, "Review of transient field theory for a baffled planar piston," *Journal of Acoustical Society of America*, vol. 70, no. 1, pp. 10-20 (1981).

13. P. A. Gatchell, *Experimental Hardware Development for a Pulsed Ultrasonic Data Collection Facility*, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1994.

14. E. O. Brigham, *The Fast Fourier Transform*, pp. 11-71, Prentice-Hall, Inc., New Jersey, 1974.

15. MATLAB, High-Performance Numeric Computation and Visualisation Sofware, for Microsoft Windows, Users Guide by the Mathworks, Inc.,Natick, MA., February 1993.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2. Library, Code 52 Naval Postgraduate School Monterey, California 93943-5101	2
3. Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5121	1
4. Professor John P. Powers, Code EC/Po Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5121	4
5. Professor Ron J. Pieper, Code EC/Pr Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5121	1
6. Professor Mathias Fink Lab, Ondes Acoustique ESPCI 10 Rue Vauquetin Paris 7500S France	1
7. Dr. James Greenleaf Department of Physiology and Biophysics Mayo Clinic Foundation 200 First St., SW Rochester, Minnesota 55905	1

8. Professor Hua Lee 1
Department of Electrical and Computer Engineering
University of California
Santa Barbara, California 93106-9560

6. Lt. Benito E. Baylosis, USN 2
2317 Palmetto St.,
Norfolk, Virginia 23513