NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

RADAR TARGET IMAGING USING TIME-REVERSED PROCESSING

by

Yosuke Inaba

September 2001

Thesis Advisor:	Michael A. Morgan
Second Reader:	David C. Jenn

Approved for public release; distribution is unlimited

Report Documentation Page

Report Date 30 Sep 2001	**Report Type** N/A	**Dates Covered (from... to)** -

Title and Subtitle Radar Target Imaging Using Time-Reversed Processing	**Contract Number**
	Grant Number
	Program Element Number
Author(s) Yosuke Inaba	**Project Number**
	Task Number
	Work Unit Number
Performing Organization Name(s) and Address(es) Research Office Naval Postgraduate School Monterey, Ca 93943-5138	**Performing Organization Report Number**
Sponsoring/Monitoring Agency Name(s) and Address(es)	**Sponsor/Monitor’s Acronym(s)**
	Sponsor/Monitor’s Report Number(s)
Distribution/Availability Statement Approved for public release, distribution unlimited	
Supplementary Notes	
Abstract	
Subject Terms	
Report Classification unclassified	**Classification of this page** unclassified
Classification of Abstract unclassified	**Limitation of Abstract** UU
Number of Pages 102	

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2001	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE: Title (Mix case letters) Radar Target Imaging Using Time-Reversed Processing	**5. FUNDING NUMBERS**
6. AUTHOR(S) Yosuke Inaba	
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) N/A	**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 is unlimited	**12b. DISTRIBUTION CODE**

13. ABSTRACT *(maximum 200 words)*

This thesis investigates and demonstrates the workability of the time-reversed process for radar imaging applications, particularly, for bi-static or multi-static radars. One benefit of the time-reversed process is its ability to reduce the calculation to determine the targets' shape. The finite-difference-time-domain (FDTD) method is used to demonstrate the time-reversed process.

Following an overview and description of the principles of the time-reversed process, the FDTD method is applied to the wave equation and the time reversed-process in 2-D space. The FDTD numerical model is developed and used for producing fundamental examples on conducting targets. The examples reveal that the time-reversed process can be employed for radar imaging within certain constraints. Finally, conclusions regarding the time-reversed-process are presented and recommendations for future research are provided.

14. SUBJECT TERMS Time-reversed process, the wave equation, FDTD	**15. NUMBER OF PAGES** 102
	16. PRICE CODE

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

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

RADAR TARGET IMAGING USING TIME –REVERSED PROCESSING

Yosuke Inaba
Lieutenant, Japan Maritime Defense Force
B.S., National Defense Academy in Japan,1993

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2001

Author: Yosuke Inaba
Yosuke Inaba

Approved by: M. A. Morgan
Michael A. Morgan, Thesis Advisor

David C. Jenn
David C. Jenn, Second Reader

Dan C Boger
Dan C. Boger, Chairman
Information Warfare Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis investigates and demonstrates the workability of the time-reversed process for radar imaging applications, particularly, for bi-static or multi-static radars. One benefit of the time-reversed process is its ability to reduce the calculation to determine the targets' shape. The finite-difference-time-domain (FDTD) method is used to demonstrate the time-reversed process.

Following an overview and description of the principles of the time-reversed process, the FDTD method is applied to the wave equation and the time reversed-process in 2-D space. The FDTD numerical model is developed and used for producing fundamental examples on conducting targets. The examples reveal that the time-reversed process can be employed for radar imaging within certain constraints. Finally, conclusions regarding the time-reversed-process are presented and recommendations for future research are provided.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I would especially like to thank my thesis advisor, Professor Michael A. Morgan, for his patience and support. I would not have completed my thesis work without him.

A special thanks goes to my parents in Japan who have encouraged me during my pursuit at the Naval Postgraduate School.

Finally, I would like to recognize those who supported me during my stay in the United States.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. OVERVIEW

Bi-static or multi-static radar has been recognized as an effective tool for detecting and tracking low observable targets, such as stealth aircraft [Ref.1]. The development of these systems emphasized measurement of the target's precise location in range and angle. However, in general, they are not capable of identifying the shape of a target. In many situations, this information may be important to make an appropriate decision or to distinguish a friendly target from the enemy. Because of its importance, radar imaging has been a topic of intense research for the past several decades, being driven by the need to identify friend from foe in future radar systems.

At present some useful radar techniques are available, such as SAR/ISAR, which produce very detailed information of a target. These techniques require, however, a relatively long process in obtaining the information. Therefore, more timely processing techniques are desired in many situations.

To obtain information relating target identity to the received radar signature, inverse scattering problems have to be solved. Solutions to the forward scattering problem, which predict scattering from known objects, are diverse and well known. Once the scattered wave is known implicitly or explicitly, determining a target's shape is possible, in theory, by inverting the forward scattered solution process. Many methods for inverse scattering have been investigated. For example, Colton and Kress [Ref.2] introduced inverse scattering theory in which Colton, Giebermann and Monk [Ref.3] provided numerical examples of obtaining the shape of an obstacle in three dimensions using time-harmonic incident and scattered acoustic waves. These methods were induced by solving the frequency- domain Helmholtz equation.

Some useful methods for solving scattering problems in the time- domain exist, such as time-domain physical optics, time-domain integral equations and finite-difference time-domain (FDTD). These methods can be used, in theory, to determine or estimate the shape of a scattering obstacle. Although there are many publications available on the inverse scattering time-domain methods, these deal mainly with an electric or magnetic field integral equation [Ref.4]. The integral equation approach requires enormous computational resources to determine the shape of a complex target and requires a very high signal to noise ratio in the received scattered signatures.

This thesis investigates a new technique for estimating the location and shape of one or more targets by processing scattering signature information captured by sensors on the perimeter of a region. A reversed-time solution within the region is performed using the FDTD method. Such a method has the potential for employment by bi-static or multi-static radar systems.

Fink [Ref.5] introduced the concept of "Time Reversed Acoustic Imaging" associated with wave field propagation. Fink's colleagues [Ref. 6, 7, 8, 9] have worked with this concept. The basic idea of this process is based on an elementary fact of time-reversal invariance of the wave equation in a lossless medium. Briefly, the process occurs when the wave field from an obstacle or a target is propagated and is captured on the surrounding surface using a number of transducers. These transducers combine the functions of microphone and loudspeaker, emitting a time-reversed replica of the received signature from each transducer. These emissions generate a time-reversed field within the enclosed region, which collapses upon the target. In other words, the wave field is recorded on the boundary and is reradiated as an appropriate reversed field. The reversed field, then, focuses on the original source. This process may be extended to electromagnetic waves.

The "Time Reversed Imaging" technique, which is the subject of study in this thesis, uses the "Time Reversed Acoustics" concept. This concept focuses on wave field propagation, in particular wavefronts. This implies that the field is not always present at the target location, which may reduce computational time. Moreover, focusing on wavefronts enables this concept to be used for a wide frequency range and for weak scattering cases. In general terms, this concept seems simple and logical, but historically it has not been used for electromagnetic waves.

B. THESIS OBJECTIVE

This thesis investigates application of the time-reversed process to electromagnetic waves and employment for bi-static or multi-static pulsed radar applications. The thesis develops needed numerical modeling and simulation using the FDTD method, which is an efficient way to represent this process for vector fields.

In Chapter 2, the general principle of the time reversal process is introduced. Chapter 3 focuses on the numerical representation and description of modeling and simulation. Chapter 4 presents numerical examples using MATLAB. Finally, Chapter 5 summarizes and concludes this study with suggestions for further research.

THIS PAGE INTENTIONALLY LEFT BLANK

II. GENERAL PRINCIPLE OF TIME- REVERSED PROCESS

This chapter presents an overview of the time-reversed process. Details can be found in Reference 8.

A. TIME REVERSAL OF ELECTRIC FIELDS

Maxwell's equations induce the wave equation for the electric field in lossless homogeneous media. In free space, this is written as

$$\nabla^2 E(\mathbf{r},\mathrm{t}) - \frac{1}{c^2}\frac{\partial^2 E(\mathbf{r},\mathrm{t})}{\partial t^2} = 0 \tag{1}$$

where ∇^2 represents the Laplacian operator with respect to the spatial **r**- coordinates and c represents the phase velocity in free space.

Looking at equation (1), it contains only a second order time–derivative operator. It follows, therefore, that if $E(\mathbf{r}, \mathrm{t})$ is a solution, then $E(\mathbf{r}, \mathrm{-t})$ is also a solution. That is, the wave equation is unchanged under a time–reversal transform if there is no absorption during propagation in the medium. This property is the starting point of the time-reversal principle. In order for the time–reversal invariance to remain valid, no loss or absorption during propagation is assumed.

The special property of time reversal has been observed by Stokes [Ref. 9] in the classical experiment of reflection and transmission of a plane wave between two different media. Considering an incident plane wave of amplitude $E_{0\mathrm{i}}$ propagating from medium 1 to medium 2, we can observe the amplitude of the reflected field ($E_{0\mathrm{r}}$) and the transmitted field ($E_{0\mathrm{t}}$). Here, introducing r as the amplitude reflection coefficient and t as the amplitude transmission coefficient leads to E_{0r} = r E_{0i} and E_{0t}= t $E_{0i.}$ Following this, a solution of the wave equation, $E(\mathbf{r}, t)$, results from three plane waves. Figure 1 indicates this situation.

In the case of no absorption, a wave must be reversible in accordance with the reciprocity theorem leading to the principle of reversibility. Thus, the condition of Figure 2 must also be physically possible. Then, the time- reversed solution, $E(\mathbf{r}, -t)$, can be described by $E_{0i,}$ E_{0r} and $E_{0t.}$ Accordingly, when examining the situation in Figure 3, there are two incident waves of amplitude: rE_{0i} and tE_{0i}. One wave whose amplitude is tE_{0i} is both reflected and transmitted at the interface. Letting r' and t' be the amplitude reflection and transmission coefficient, respectively, for a wave incident from medium 2 to medium 1, the reflected portion is tr'E_{0i} and the transmitted portion is tt'E_{0i}. Similarly, the incoming wave whose amplitude is rE_{0i} splits into both the amplitude rr E_{0i} and rt E_{0i}. Using the superposition principle, it follows that

$$tt'E_{0i} + rr\,E_{0i} = E_{0i} \qquad (2)$$

$$tr'E_{0i} + rt\,E_{0i} = 0 \qquad (3)$$

Therefore,

$$tt' = 1 - r^2 \qquad (4)$$

$$r' = -r \qquad (5)$$

If the interface between medium 1 and medium 2 is a perfect conductor, t and t' are zero. This results in either r= 1, r'= -1 or r= -1, r'=1. For simplicity, this thesis considers only targets or obstacles whose surfaces have the condition of a perfect conductor. Furthermore, it is important to note that the two relationships written above are only valid if the reflected and transmitted plane waves are able to propagate without attenuation, which implies that they have a real wave number. The incident field we consider in this thesis does not contain any evanescent waves that cannot be time-reversed.

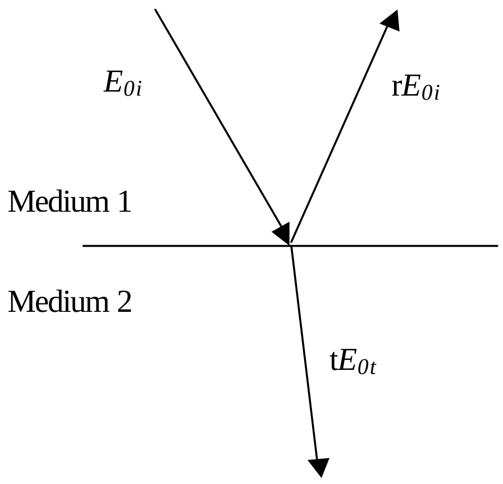


Figure 1. Reflection and transmission of a plane wave along the interface between medium 1 and medium 2

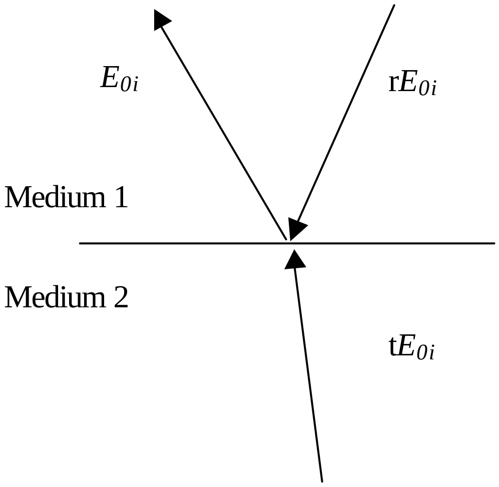


Figure 2. Time- reversal of Figure 1

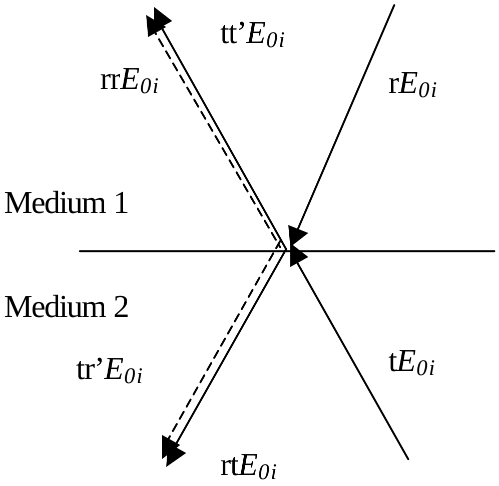


Figure 3. Explanation of time- reversal for reflection and transmission

B. PRACTICAL CONSIDERATION

The initial conditions of a target or an obstacle and the boundary conditions determine a unique solution, $E(\mathbf{r}, t)$ to the wave equation in (1). In the time–reversed operation, the solution $E(\mathbf{r}, -t)$ is given by modifying the initial condition and the boundary condition. In practice, however, because every physical phenomenon requires causality, $E(\mathbf{r}, -t)$ is not a valid solution. Therefore, we consider $E(\mathbf{r}, T-t)$ under the limited time interval T, where T is sufficiently advanced in time so that $E(\mathbf{r}, t)$ is regarded as zero for $t > T$. To assume otherwise requires "initial condition" knowledge of the fields $E(\mathbf{r}, t)$ in the whole three-dimensional volume during the time interval T in order to generate the time–reversed solution $E(\mathbf{r}, T-t)$. A more realistic way to generate the time–reversed solution is to use the advantage of Huygens' principle. Based on this principle, the time-reversed operation in the three-dimensional volume requires time-reversed boundary conditions on an enclosing two-dimensional surface. Using this approach, focusing on a target can be described in the following way.

During the recording step, the target within the volume surrounded by a finite number of receivers generates a field, $E(\mathbf{r}, t)$, which produces an expanding wavefront. The receivers sample the field at locations on the enclosing surface and have the capability to measure the field without disturbing the propagation of the field. Therefore, the field is propagated in a free unbounded space. During the time-reversed reconstruction, the target behaves as a passive source or is ignored. Each receiver then generates the field, $E(\mathbf{r}, T-t)$, that corresponds exactly to the time-reversal of the corresponding field measured during the recording step. A time-reversed field back-propagates inside the recording surface and is focused on the initial target position. Figures 4 and 5 illustrate the recording and reconstruction steps, respectively.

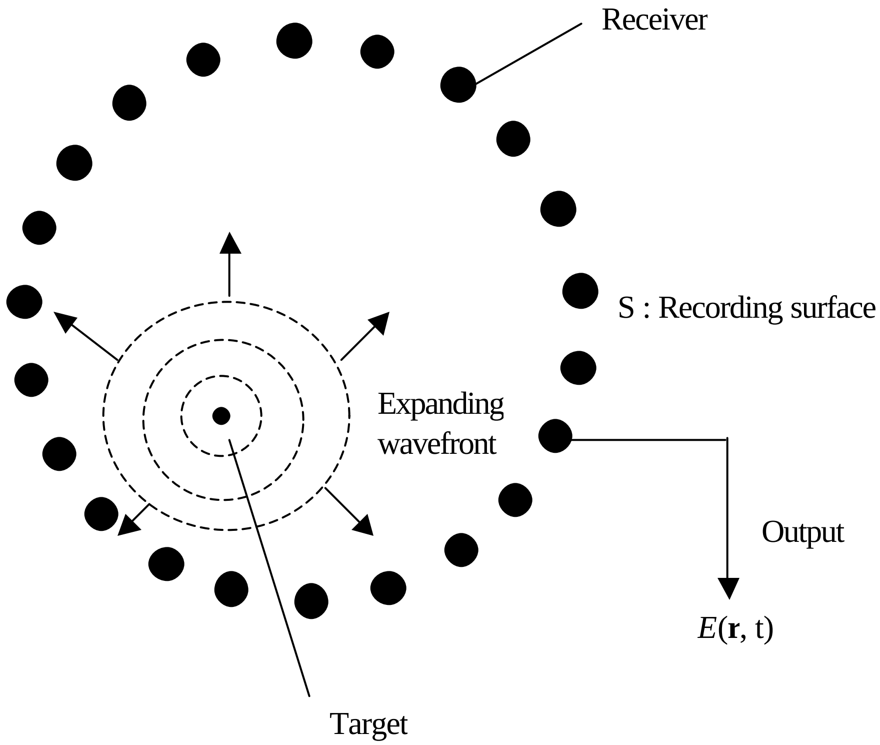


Figure 4. Recording step of the time- reversal process with a closed surface

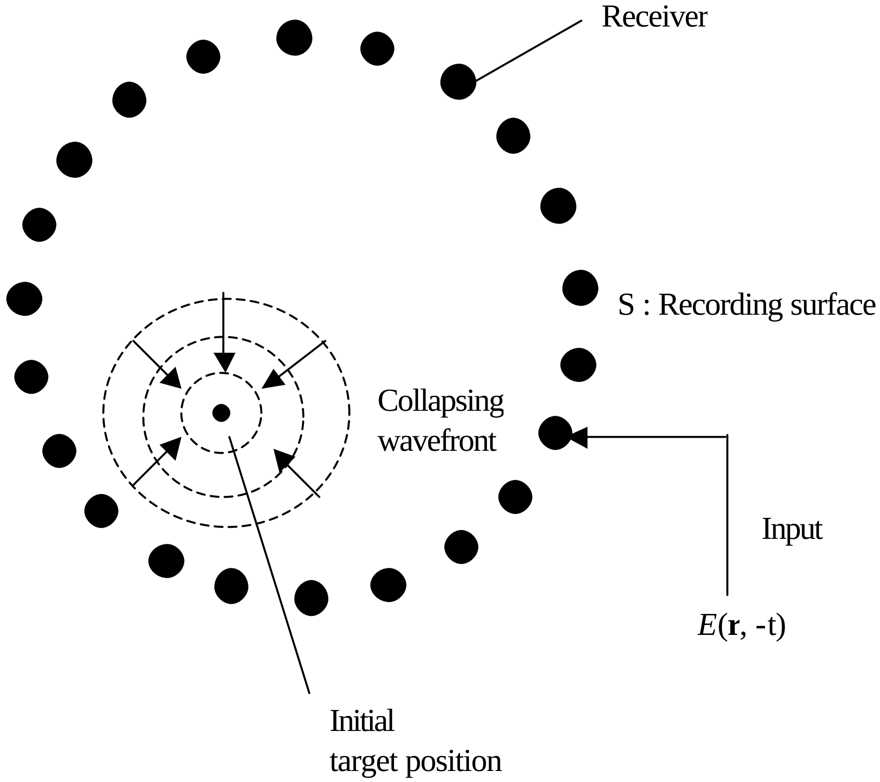


Figure 5. Reconstruction step of the time-reversal process with a closed surface

THIS PAGE INTENTIONALY LEFT BLANK

III. NUMERICAL MODELING OF TIME –REVERSED PROCESS

This chapter presents an overview of the finite-difference time-domain (FDTD) numerical model used for demonstrating the time-reversed process in 2-D space. The FDTD method is an appropriate way to represent this process, applying approximations for the space and time partial derivatives within the wave equation. Space and time discretizations are selected to ensure the numerical stability of the algorithm.

For the region defined, the wave equation is solved subject to initial value and boundary value conditions according to electromagnetic theory. This is the same procedure applied to the FDTD method. Overall, the FDTD is a time-marching procedure discretely simulating the continuous fields that solve the wave equation. The solution is subject to the initial and boundary conditions. At each time step, the system of equations is used to update the field components and is fully explicit so that setting up or solving linear equations is not required. Time stepping is continued, allowing one to observe electromagnetic wave propagation and scattering. Details on the FDTD can be found in many publications. Some of these are listed in References 10 and 11.

The next section presents numerical modeling of the wave equation followed by the forward and reverse processes, and is based on Reference 12. For simplicity, a 3-D (2-dimentional plus time) rectangular region with $E_z(x, y, t)$ (TM$_z$ waves) is assumed throughout this thesis.

A. THE FDTD SOLUTION OF THE WAVE EQUATION

This section briefly provides details on how the wave equation can be solved with the FDTD method.

Consider solving the wave equation in a rectangular region, 0= x = a, 0= y = b and t = 0, as depicted in Figure 6,

$$\nabla^2 E_z(x,y,t) - \frac{1}{c^2}\frac{\partial^2 E_z(x,y,t)}{\partial t^2} = 0 \tag{6}$$

with initial conditions

$$E_z(x,y,0) = f(x,y) \tag{7}$$

$$\frac{\partial E_z}{\partial t}(x,y,0) = g(x,y) \tag{8}$$

and boundary conditions.

$$\begin{aligned} E_z(0,y,t) &= e_1(y,t), \qquad E_z(a,y,t) = e_2(y,t) \\ E_z(x,0,t) &= e_3(x,t), \qquad E_z(x,b,t) = e_4(x,t) \end{aligned} \tag{9}$$

Here, the assumption is that a = Mh and b= Nh, where ?x=?y=h is an equal grid spacing in x and y with M and N as arbitrary integers. Letting ?t be the time step and T be the desired time interval, the grid positions are given by

$$x_m = (m-1)h,\ y_n = (n-1)h \text{ and } t_p = (p-1)\Delta t \tag{10}$$

where $m = 1, M$, $n = 1, N$ and $p = 1, T$.

We define $E_z(m,n,p)$ to be the discrete space-time sample of $E_z(x_m, y_n, t_p)$.

Applying the finite difference approximations in space and time at the point (*m* ,*n*, *p*) to each term of the wave equation gives,

$$\nabla^2 E_z(x_m, y_n, t_p) \approx \frac{1}{h^2}\{E_z(m+1,n,p) + E_z(m-1,n,p) + E_z(m,n+1,p) + E_z(m,n-1,p) - 4E_z(m,n,p)\} \tag{11}$$

$$\frac{1}{c^2}\frac{\partial^2 E_z(x_m, y_n, t_p)}{\partial t^2} \approx \frac{1}{c^2(\Delta t)^2}\{E_z(m,n,p+1) + E_z(m,n,p-1) - 2E_z(m,n,p)\} \tag{12}$$

Substituting (11) and (12) into (6) and solving for E_z at t_{p+1}, yields:

$$E_z(m,n,p+1) \approx \left(\frac{c\Delta t}{h}\right)^2 \{E_z(m+1,n,p) + E_z(m-1,n,p) + E_z(m,n+1,p) + E_Z(m,n-1,p)\} + 2\left\{1 - 2\left(\frac{c\Delta t}{h}\right)^2\right\} E_z(m,n,p) - E_z(m,n,p-1) \tag{13}$$

Equation (14) implies that we can predict E_z at the point (x_m, y_n, t_{p+1}) from six previous points in the grid as shown in Figure 7.

By introducing 2-D spatial arrays E_p =E_z(: , : , p), where E_p is an MxN array at time step, p, we can rearrange equation (14) to the form,

$$E_{p+1} = \left(\frac{c\Delta t}{h}\right)^2 \{A \cdot E_p + E_p \cdot B\} + 2\left\{1 - 2\left(\frac{c\Delta t}{h}\right)^2\right\} E_p - E_{p-1} \tag{14}$$

where

$$A = \begin{vmatrix} 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & \ddots & \ddots & \vdots \\ 0 & 1 & \ddots & 1 & 0 \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & 0 & 1 & 0 \end{vmatrix} \quad \text{and } B = \begin{vmatrix} 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & \ddots & \ddots & \vdots \\ 0 & 1 & \ddots & 1 & 0 \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & 0 & 1 & 0 \end{vmatrix}$$

are respectively MxM and NxN unit off-diagonal arrays. Equation (14) is the time-marching equation for non-boundary terms. To insure the stability of solutions, the next condition must satisfy the 2-D Courant condition:

$$\frac{c\Delta t}{h} \le \frac{1}{\sqrt{2}} \tag{15}$$

At p = 1, the initial condition (7) is used and the value at p = 0 is established by the initial condition (8), which is:

$$\frac{\partial E_z(x, y, 0)}{\partial t} \approx \frac{E_1 - E_0}{\Delta t} = g(x_m, y_n) \tag{16}$$

Using E_1 and E_0, the non-boundary value of E_2 is given by equation (14). Then the boundary conditions are added to E_2 and the process is repeated to find E_3 from E_2 and E_1 until the desired time is reached.

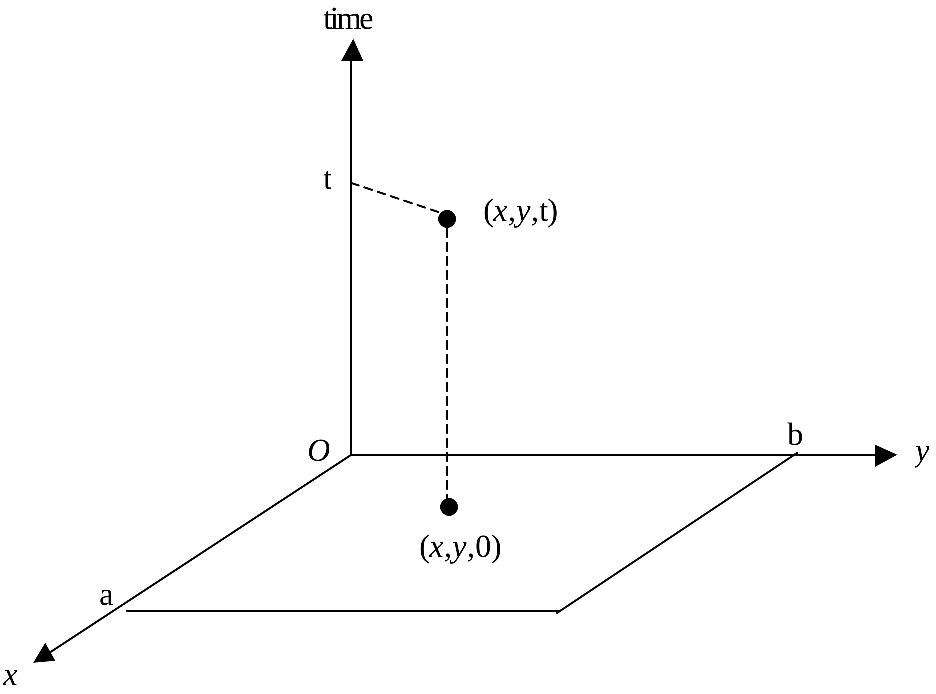


Figure 6. Geometry of the space-time rectangular region

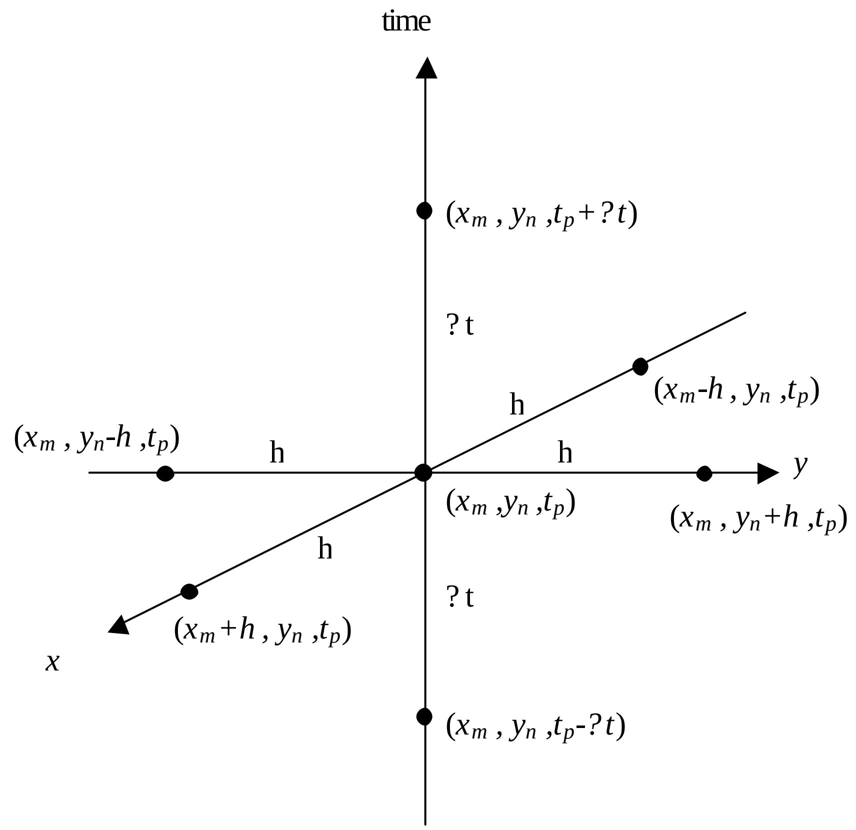


Figure 7. FDTD space-time node diagram

B. MODEL DESCRIPTION

During the forward or recording step, equation (14) is used to find the forward time solutions of the wave equation using the initial conditions and the boundary conditions. Equation (14) is also used during the reverse or reconstruction step to find the reverse time solutions of the wave equation by defining the solutions found at the desired time step T as the initial and the boundary conditions.

Since it is not possible to consider an infinite region to propagate the fields, a limited region is required to terminate the simulation. Rather than utilizing absorbing type boundary conditions which yield only approximate cancellation of reflections, it was decided to avoid all reflections by employing a distant terminating boundary whose reflection, although large, does not arrive back at the sensor boundary until t > T.

1. Model

Denoting M_p and N_p as the number of receivers on the recording surface of the x- and y-segment, and M_e and N_e as the spacing between each of the x- and y-segment receivers, respectively, the number of grid nodes on the sensor surface segments becomes

$$M_s=(M_p-1)\times M_e+1 \quad \text{(x-segment)}$$

$$N_s=(N_p-1)\times N_e+1 \quad \text{(y-segment)} \qquad (17)$$

The number of nodes on the distant boundary will be set in this work to be

$$M=2\times M_s \quad \text{(x-segment)}$$

$$N=2\times N_s \quad \text{(y-segment)} \qquad (18)$$

The sensor surface is located at the approximate center of the distant boundary by defining its edges as follows:

m1= (M-M$_s$)/2+1 (Left edge), m2= m1+Ms-1 (Right edge) (x-segment)

n1= (N-N$_s$)/2+1 (Lower edge), n2= n2+N$_s$-1 (Upper edge) (y-segment) (19)

The geometry is described in Figure 8.

For simplicity, the spacing of each node (h) is set to 1 m throughout this thesis. As a result, the 2-D stability condition (15) is rewritten as

$$q = \frac{1}{c\Delta t} \geq \sqrt{2} \tag{20}$$

Usually, in two or three- dimensional problems the stability condition q is chosen to be twice that needed. Therefore, this model also chooses q to be 2. This ensures two samples between two adjacent points at each time step.

2. Initial And Boundary Conditions

Since this thesis considers only the source-free region, no fields exist within the recording surface and the distant boundary at the initial time. Therefore, the initial conditions within the distant boundary are given by:

$$E_z(m,n,1) = 0$$

$$\frac{\partial E_z(m,n,1)}{\partial t} = 0 \tag{21}$$

where m = 1, M and $n = 1, N$.

Assuming the scattered field is to be reflected at the distant boundary, the boundary conditions at the distant boundary become:

$$\begin{aligned} &E_z(1,n,p) = 0, \quad E_z(M,n,p) = 0 \\ &E_z(m,1,p) = 0, \quad E_z(m,N,p) = 0 \end{aligned} \tag{22}$$

where m =1, M, n =1, N and p =1, T. These boundary conditions are maintained throughout the forward time step.

The field data at the recording surface must be stored at each time step. Letting *EzBC1*, *EzBC2*, *EzBC3*, and *EzBC4* denote the arrays storing the field data of each side of the recording surface during the forward time step:

$$\begin{aligned} &EzBC1(p) = E_z(m_r, n1, p), \quad EzBC2(p) = E_z(m_r, n2, p) \\ &EzBC3(p) = E_z(m1, n_r, p), \quad EzBC4(p) = E_z(m2, n_r, p) \end{aligned} \tag{23}$$

where $m_r = m1, m1+M_e, m1+2M_e, \ldots\ldots m2$, $n_r = n1, n1+N_e, n1+2N_e, \ldots\ldots n2$, and $p=1,T$.

The field data stored at the recording surface at any forward time step, p = P and p = P-1, where P is less than T, can be defined as the initial conditions for the time-reversed process:

$$\begin{aligned} &E_z^r(m_r, n1, 1) = EzBC1(P), \quad E_z^r(m_r, n2, 1) = EzBC2(P) \\ &E_z^r(m1, n_r, 1) = EzBC3(P), \quad E_z^r(m2, n_r, 1) = EzBC4(P) \end{aligned} \tag{24}$$

and

$$\begin{aligned} &E_z^r(m_r, n1, 2) = EzBC1(P-1), \quad E_z^r(m_r, n2, 2) = EzBC2(P-1) \\ &E_z^r(m1, n_r, 2) = EzBC3(P-1), \quad E_z^r(m2, n_r, 2) = EzBC4(P-1) \end{aligned} \tag{25}$$

where E_z^r is the time-reversed field and P ≤ T. Based on conditions (24) and (25), equation (14) is used to update the time-reversed field inside the recording surface until the reversed time step, P.

If M_e = N_e = 1, the time-reversed field is easily updated because ?h is assumed to be 1. If M_e ? 1 or N_e ? 1, however, the field data requires interpolation at grid points between each two adjacent transducers to maintain the continuity of the field. To satisfy this condition, we use a ‘cubic spline’ interpolation algorithm in MATLAB.

3. Incident Field

The incident field used here is represented by a Gaussian impulse plane wave propagating in the –x direction, centered at $x = x_0$ at t = 0, with the form,

$$f(x,t) = e^{-(ct+x-x_0)^2/2s^2} \tag{26}$$

where s is the standard deviation of spatial width of the pulse. Viewing the impulsive waveform at $x = x_0$ gives

$$f(x_0,t) = e^{-A^2t^2} \tag{27}$$

where $A^2 = \dfrac{c^2}{2s^2}$. Its frequency spectrum is given by:

$$F(x_0,\, f) = e^{-p^2f^2/A^2}\,. \tag{28}$$

The 3dB bandwidth of this baseband pulse is given by solution of

$$F(x_0,\, f_{3dB}) = \frac{1}{\sqrt{2}} \tag{29}$$

which yields,

$$f_{3dB} = \frac{A}{p}\sqrt{\frac{\ln 2}{2}} = \frac{c}{2ps}\sqrt{\ln 2} \tag{30}$$

Note the reciprocal relationship between the spatial standard deviation and the 3dB bandwidth.

4. Field Representation

A scattered field results when an incident field interacts with a target. Interactions include polarization currents excited within dielectric materials and surface currents induced near to the surface of metallic conductors. Assuming a target composed of perfectly conducting metallic surfaces, the total tangential E-field is forced to zero at the surface. Since the total field is given by $E_{total} = E_{inc} + E_s$, where E_{total}, E_{inc}, and E_s represent the total field, the incident field and the scattered field, respectively, the tangential scattered field is equal to the negative of the incident tangential field at the metallic surface of the target.

To properly execute the FDTD time-reversed field simulation of the scattered field when the field focuses on a metallic target we need to provide the negative incident field boundary condition at all surface target nodes. However, we usually do not know where the target is or when the incident field reaches the target. Unless this field boundary condition is applied at these target nodes during the time-reversed process, field interactions occur between adjacent nodes and the time-reversed scattered field solution does not become extinguished for times prior to the initial plane wave impact.

Suppose that a point-like target exists inside the recording surface and the total scattered field on the recording surface is already known. The total scattered field inside the recording surface would then consist of the scattered field generated by the time-reversed boundary conditions on the recording surface and from the negative incident field boundary conditions caused by nature on the target, or

$$E^{S}_{Z\,total}(\vec{r},-t) = E^{S}_{Z\,B}(\vec{r},-t) + E^{S}_{Z\,T}(\vec{r},-t) \tag{31}$$

where $E^{S}_{Z\,total}$ is the total scattered field inside the surface, $E^{S}_{Z\,B}$ is the scattered field generated by the recording boundary and $E^{S}_{Z\,T}$ is the scattered field interaction with the target. We can solve for $E^{S}_{Z\,B}$ by using only the boundary data on the recording surface without knowing the location of the target. $E^{S}_{Z\,total}$ is able to be set to zero before the time ,t$_0$, when the incident field reaches the target. As a result, we can predict $E^{S}_{Z\,T}$ as:

$$E^{S}_{Z\,T}(\vec{r}_T,-t) = -E^{S}_{Z\,B}(\vec{r}_T,-t) \tag{32}$$

where $\vec{r}_T$ is the location of the target. Once the node location is identified, we can simulate the target node interaction and properly terminate the time- reversed process.

5. Conservation Of Energy

The above sequential node identification process may be extended in theory to multiple target nodes. However, a different concept is applied in this thesis to yield a cost function for solution optimization. This concept is that of conservation of energy. The total scattered field energy accumulated within the recording surface during the time-reversed process will become constant at all times prior to the initial impact of the incident field on the target. This same result will occur in the time-reversed FDTD simulation if the reversed field collapses while the correct target node locations are given negative incident field boundary conditions. If the accumulated total energy in the time-reversed solution continues increasing before this initial impact time, then one or more assumed target nodes are wrong.

The energy metric within the sensor surface at an arbitrary time step, t$_1$, is defined by:

$$Energy(t_1) = \sum_{m=m1}^{m2} \sum_{n=n1}^{n2} [E_z(m,n,t_1)]^2 \tag{33}$$

where $E_z(m\ \ ,n\ \ ,t_1)$ is the time-reversed field at each node within the recording surface at the time step, t_1. The accumulated total energy to the time step, T, is defined by

$$Energy(t) = \sum_{t=1}^{T} \sum_{m=m1}^{m2} \sum_{n=n1}^{n2} [E_z(m,n,t)]^2 \tag{34}$$

If the accumulated energy in (34) becomes constant, this implies that the time-reversed field is collapsing onto the exact target node locations.

An example is presented in the next chapter to illustrate this concept. In the example, we enforce the negative incident field on the exact target nodes during the reversed time step and obtain the accumulated total energy. Next, the assumed target nodes are slightly perturbed from the original location to demonstrate the increase of accumulated total energy.

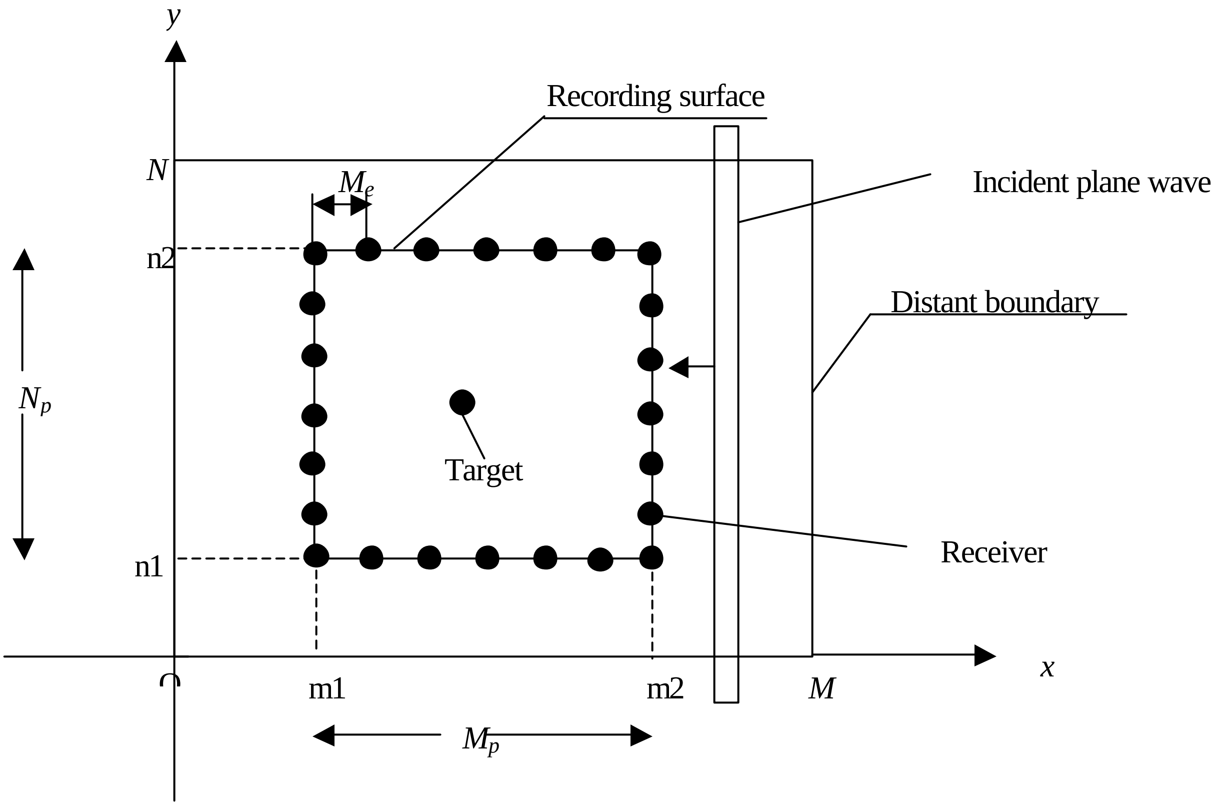


Figure 8. Simulation model diagram

IV. EXAMPLES OF TIME-REVERSED IMAGING

This chapter presents fundamental examples of the time-reversed imaging process. The first example illustrates the basic time-reversed process, while the other examples consider the process working under practical consideration, such as pulsed radar applications. However, this chapter does not consider some aspects of the real environment, such as noise, actual 3-D shapes and materials of targets and any kind of information loss by the media. In all examples, the dimensions of the FDTD grid and number of time steps are set as follows: $M_S = N_S = 101$, $T = 300$. This selection sets the recording surface as a 100m × 100m square grid and the distant boundary as a 200m × 200m square grid with 201 by 201 points.

All programs used for calculations shown in this chapter are presented in the Appendix.

A. TARGETS NODES WITH INITIAL CONDITIONS ONLY

This first example considers two aircraft-like targets present inside the recording surface. The targets are represented by specific grid points that are excited by initial conditions, E_Z (x, y, t=0) =1, but with no subsequent boundary conditions. Grid points surrounding the target nodes are also endowed with initial conditions, but which rapidly decrease with distance, per a Gaussian distribution. No incident field is considered and there are no boundary field interactions for the target nodes, such as with conducting targets. This example demonstrates how the reversed field converges onto the target nodes for the fundamental initial value problem.

1. Forward Time Step Solutions

The forward time simulation is described first to help understand the reversed time simulation described later. Figure 9 shows the initial condition of the simulation. For convenience, the recording surface is presented as a dashed line. Every 60th time step is presented in Figure 10. Figure 11 represents the final time step. Time-reversed recorded field values on the inner grid boundary are used to drive the time-reversed solution while null initial conditions are assumed within the inner grid.

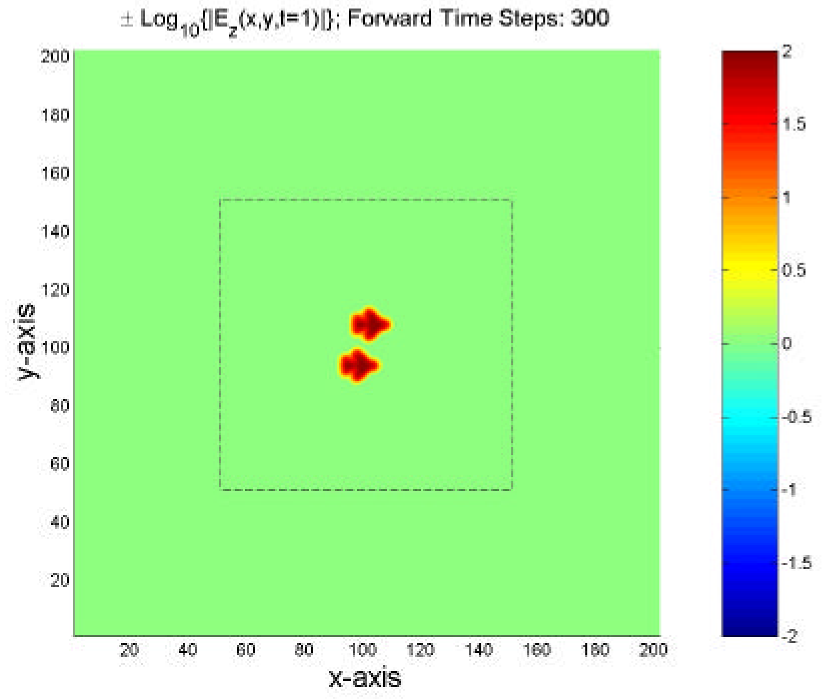


Figure 9. Initial condition of the forward time step for two aircraft-like targets

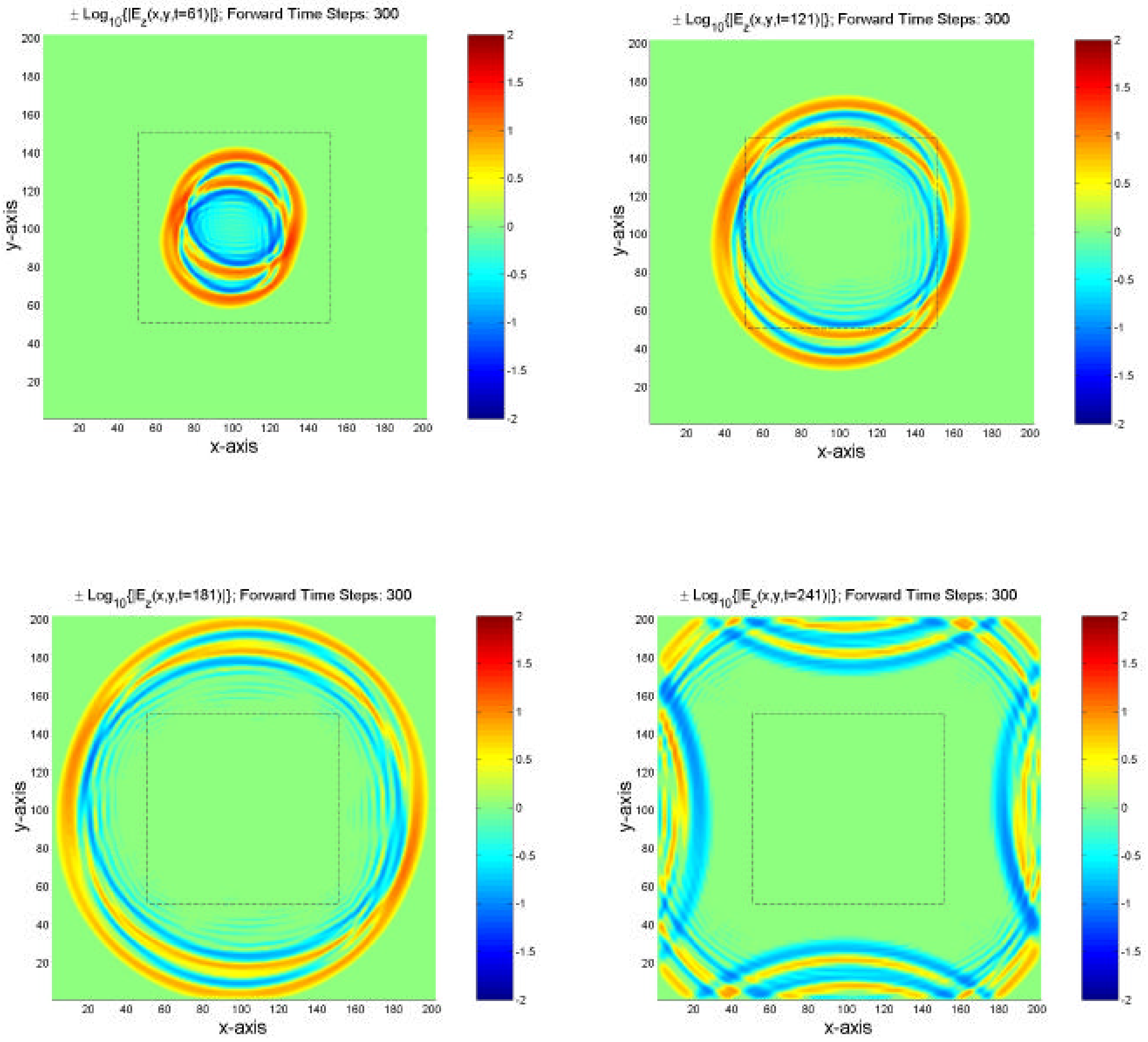


Figure 10. Forward time step for two aircraft-like targets at T = 61, T = 121, T = 181 and T = 241

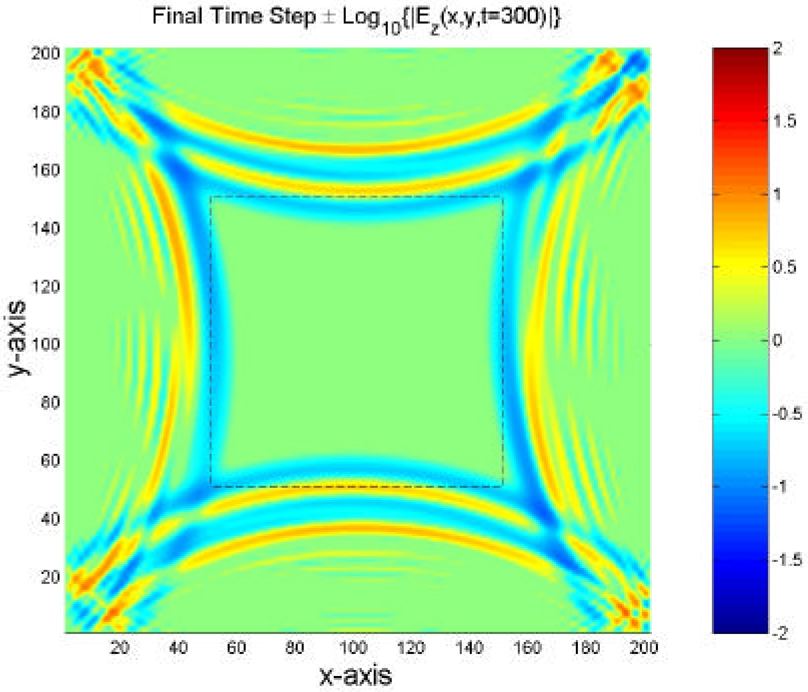


Figure 11. Final time step T = 300 for two aircraft-like targets

2. Reversed Time Step Solution with $M_e = N_e = 1$

This subsection presents the reversed time step simulation in the case of $M_e = N_e = 1$. In this case, each node acts as a transducer; thus, the number of transducers is $M_p = N_p = 101$. Selecting $M_e = N_e = 1$ provides the most accurate field representation although some errors occur due to the computational truncation. Therefore, the final reversed time step condition does not indicate the exact initial condition of the forward time step. As seen in Figure 11, the field reflected from the outer boundary has crossed the inner recording surface sometime before the T=300 time step. This reflection must not be used in the reverse-time process. Thus, the reverse-time processing will begin at T=250. Figure 12 shows the initial condition of the reversed time step with T = 250. Every 25th reversed time step is displayed in Figure 13 and Figure 14, with T = 25 and T = 0 shown in Figure 15.

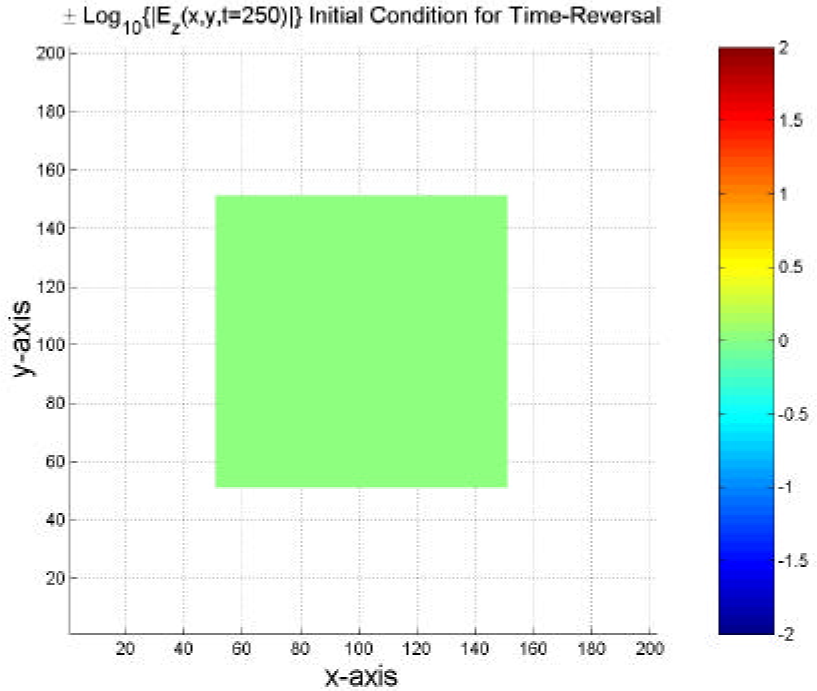


Figure 12. Initial condition of the reversed time step for two aircraft-like targets with M_e= N_e= 1 case

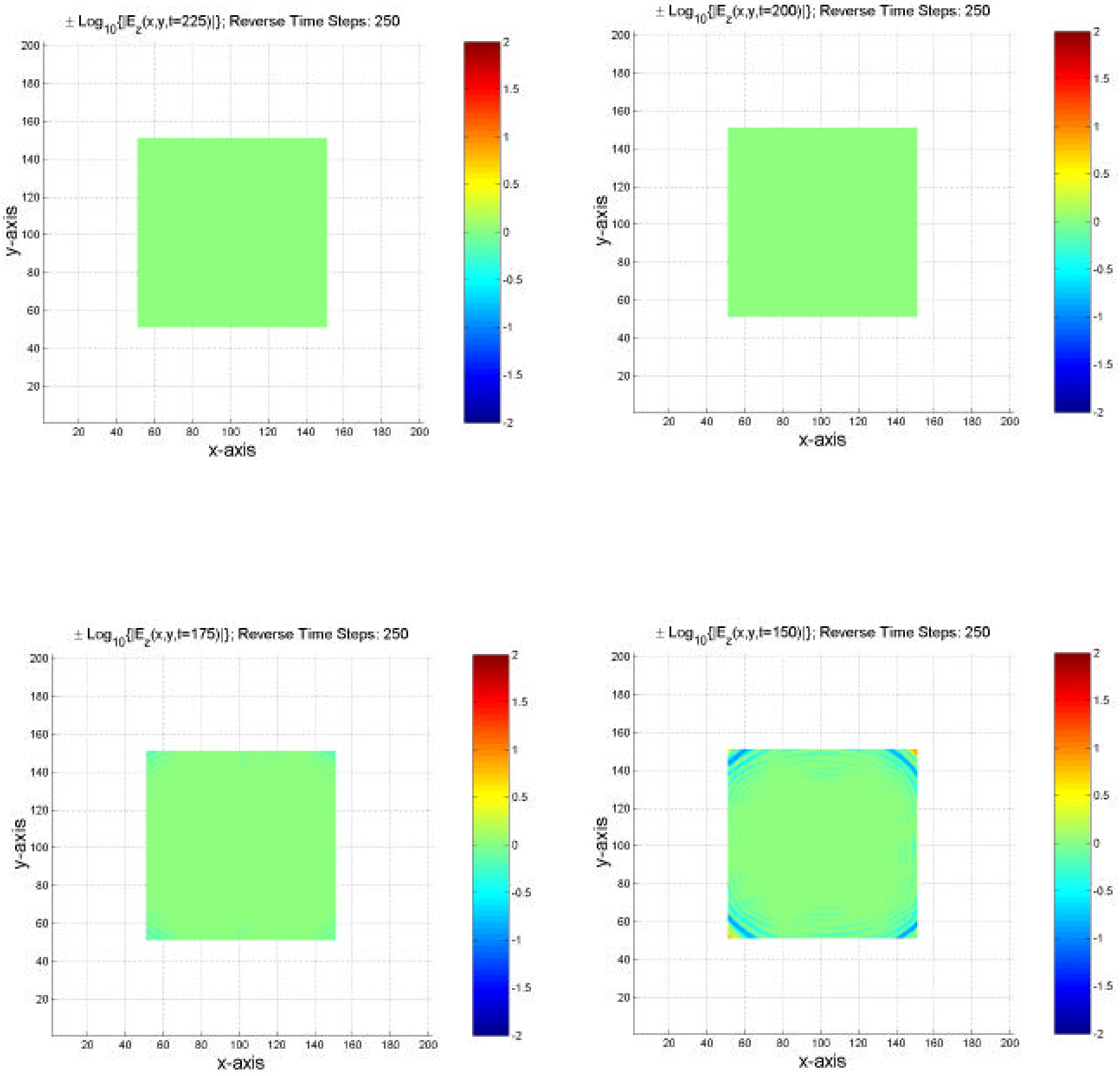


Figure 13. Reversed time step for two aircraft-like targets with M_e= N_e= 1 at T= 225, T= 200, T= 175 and T= 150

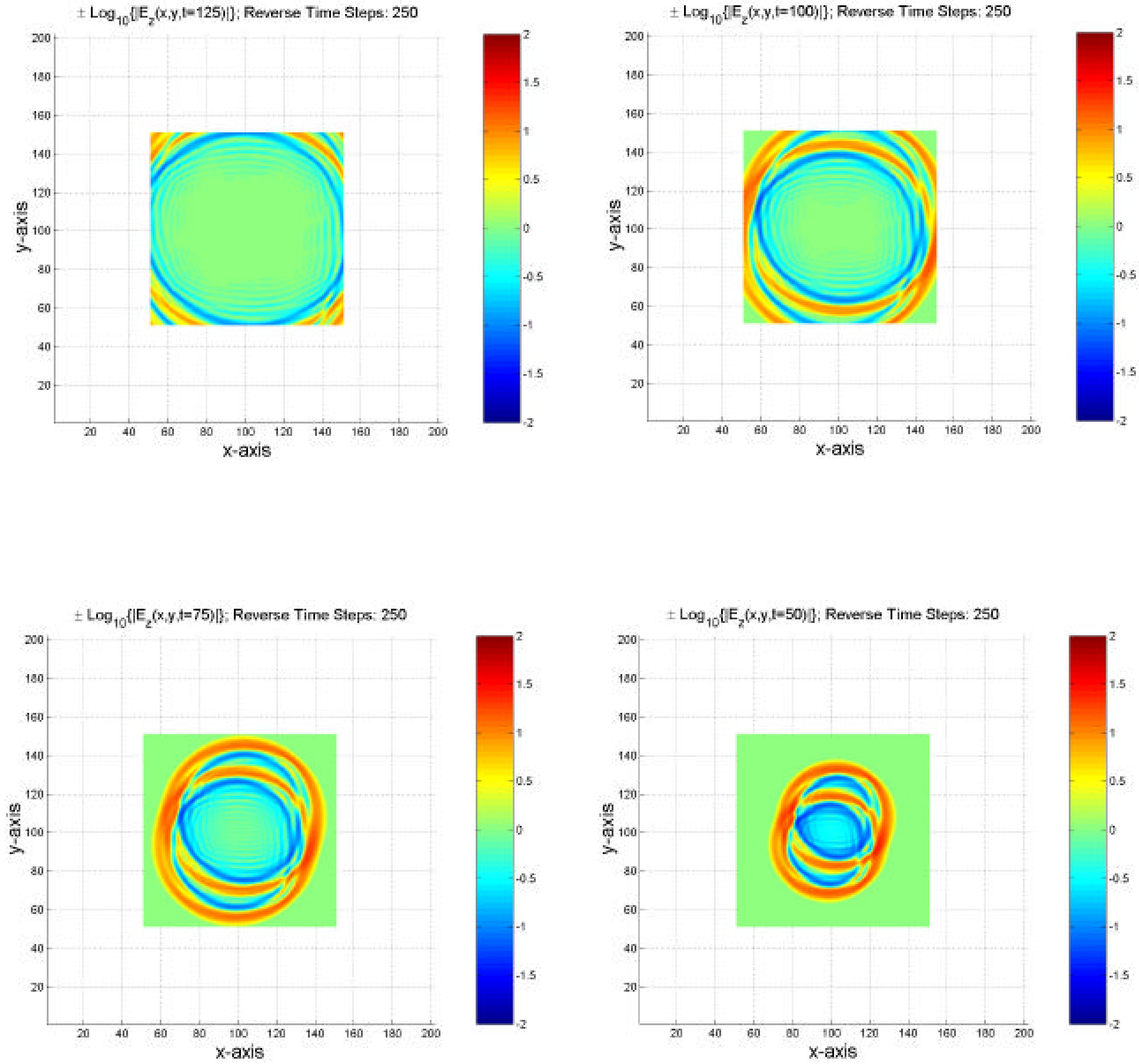
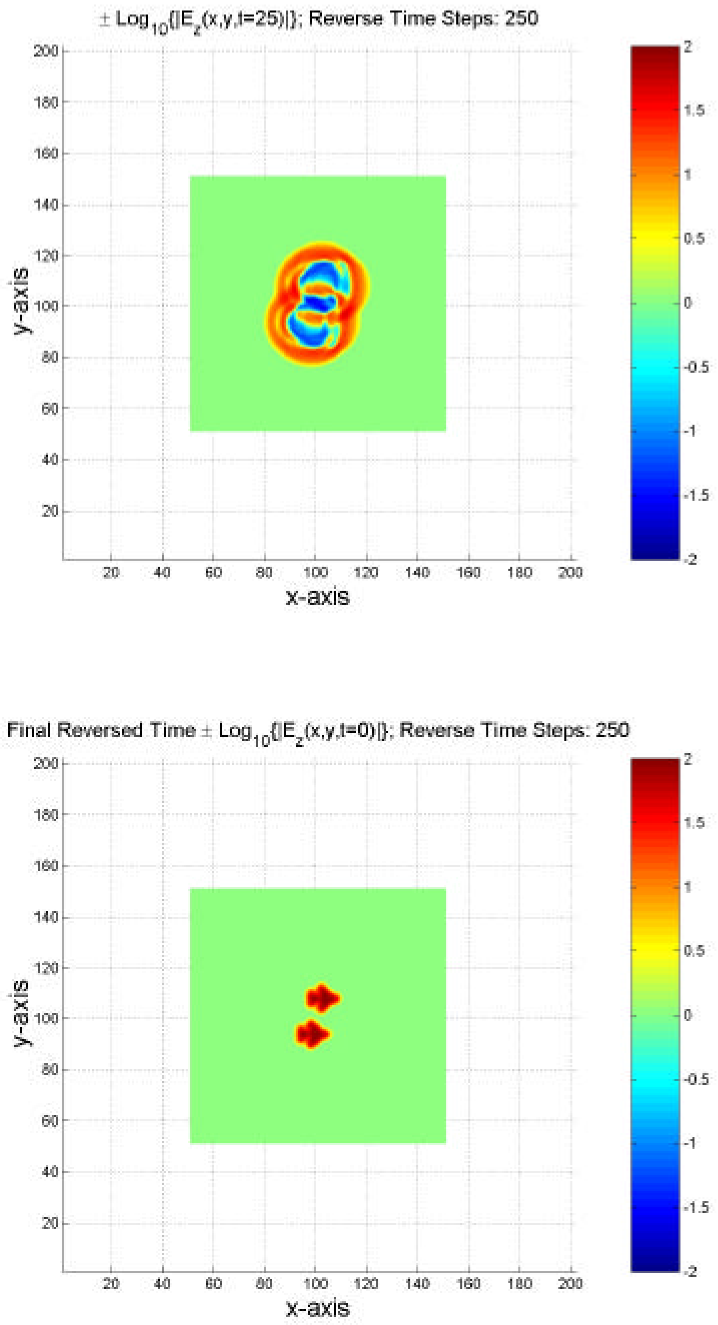


Figure 14. Reversed time step for two aircraft-like targets with $M_e = N_e = 1$ at T= 125, T= 100, T= 75 and T= 50



± Log$_{10}${|E$_z$(x,y,t=25)|}; Reverse Time Steps: 250

Final Reversed Time ± Log$_{10}${|E$_z$(x,y,t=0)|}; Reverse Time Steps: 250

Figure 15. Reversed time step for two aircraft-like targets with M_e = N_e= 1 at T= 25 and T= 0

3. Reversed Time Step Solution with $M_e = N_e = 5$

This subsection presents the case of M_e = N_e = 5, which corresponds to M_p = N_p= 21. This selection means that every 5th point data on the recording surface is used to calculate the reversed field. Figure 16 shows the initial condition of the reversed time step at T = 250. The red circles in Figure 16 represent the transducers. Figure 17 and 18 display every 25th time step, while Figure 19 shows the final condition of the reversed time step at T=0. This M_e = N_e= 5 case provides less accuracy than the case of M_e = N_e =1 does, but it still gives useful targets information.

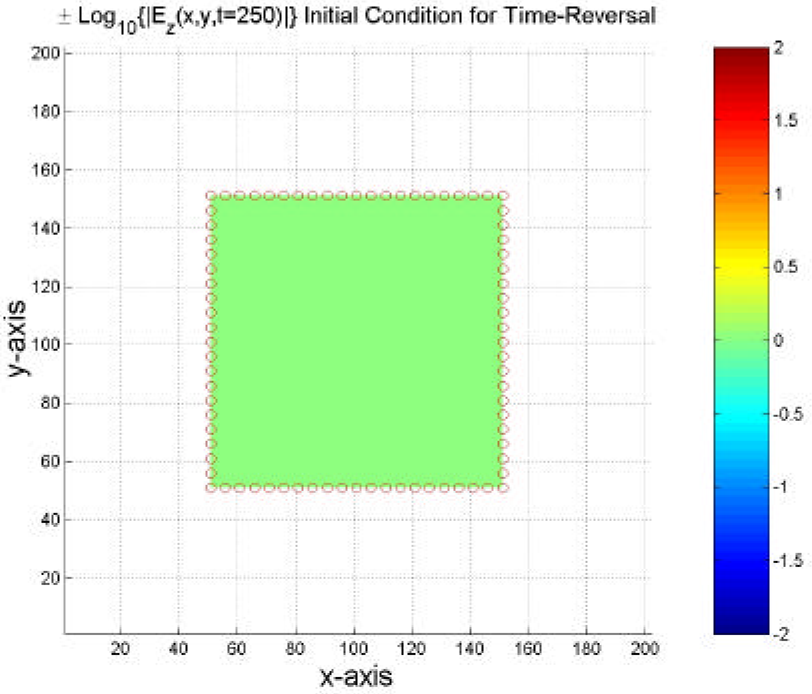


Figure 16. Initial condition of the reversed time step for two aircraft-like targets with M_e = N_e = 5

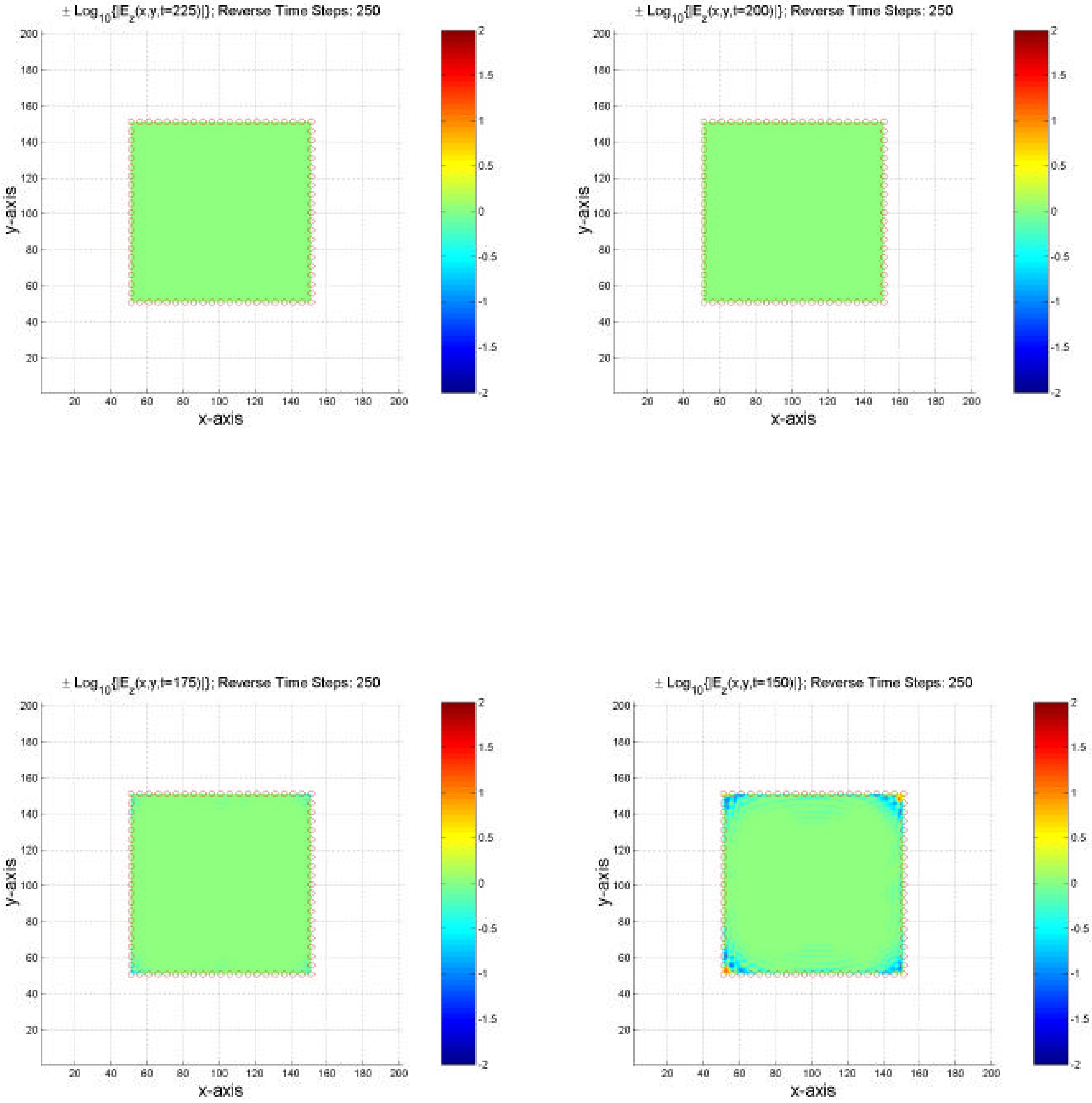


Figure 17. Reversed time step for two aircraft-like targets with $M_e = N_e = 5$ at T= 225, T= 200, T= 175 and T= 150

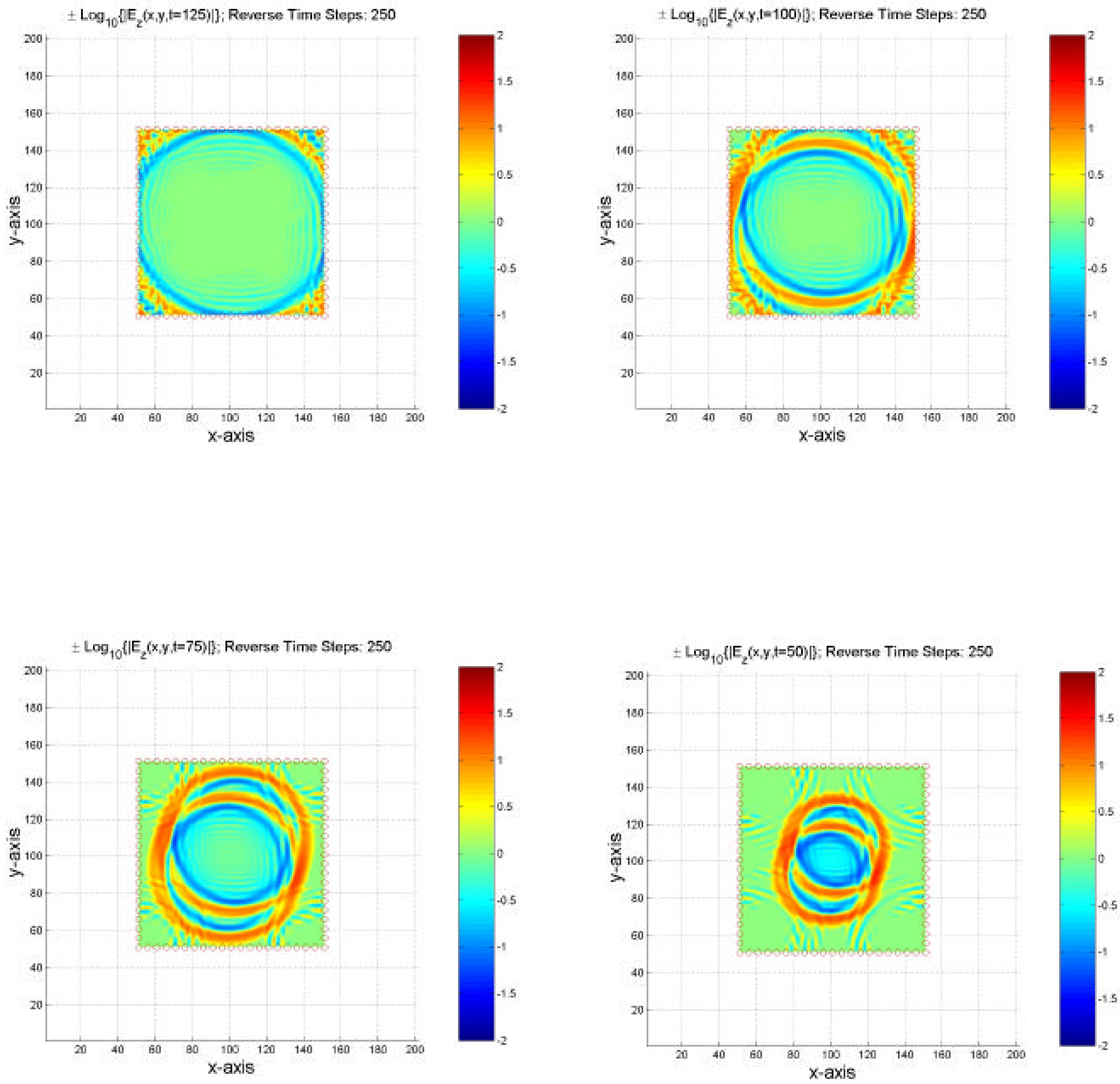


Figure 18. Reversed time step for two aircraft-like targets with $M_e = N_e = 5$ at T= 125, T= 100, T= 75 and T= 50

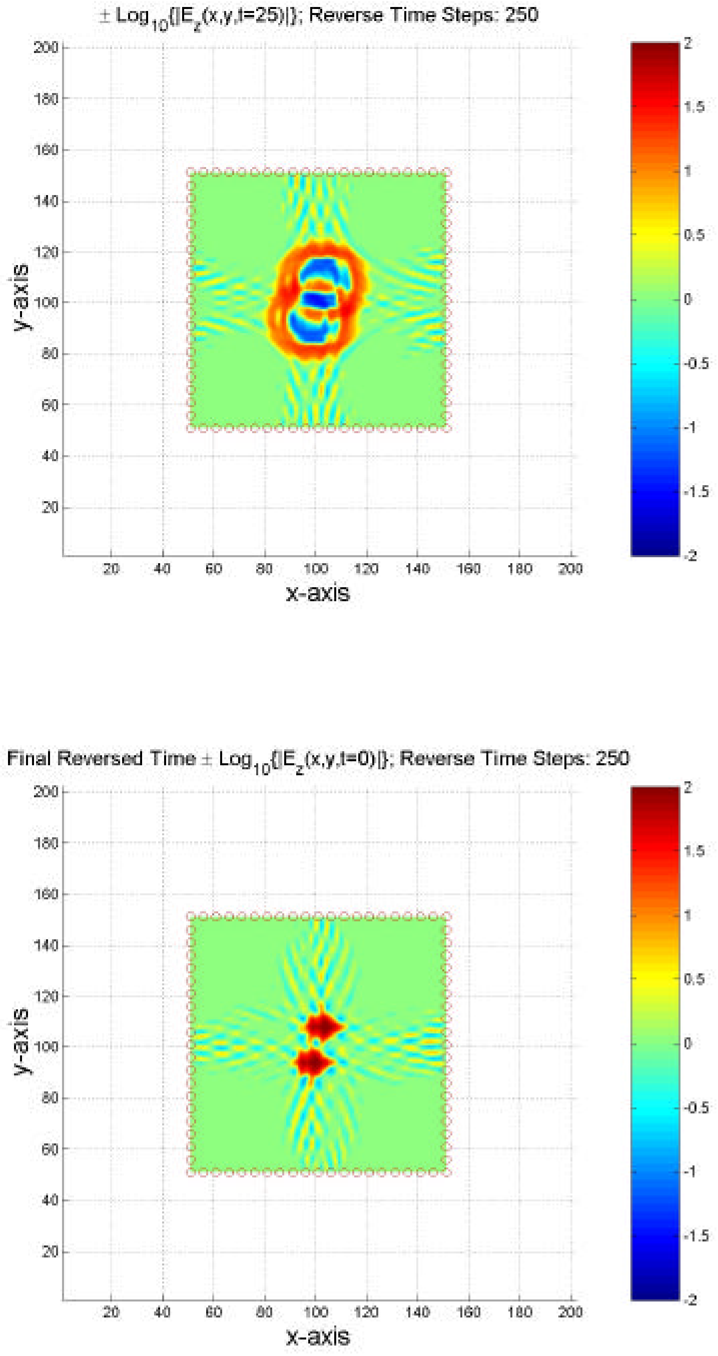


Figure 19. Reversed time step for two aircraft-like targets with $M_e = N_e = 5$ at T= 25 and T= 0

4. Reversed Time Step Solution With $M_e = N_e = 10$

In this final section, the case of M_e = N_e= 10 case is simulated. In this situation, the corresponding number of the transducers is M_p =N_p =11 indicating that every 10th point on the recording surface is used. Figure 20 shows the initial condition of the reversed time step at T = 250. Figure 21 and 22 display every 25th time step. Figure 23 shows the final step of the reversed time step, which makes the targets' shape difficult to identify. In this case, the images of the targets are blurred.

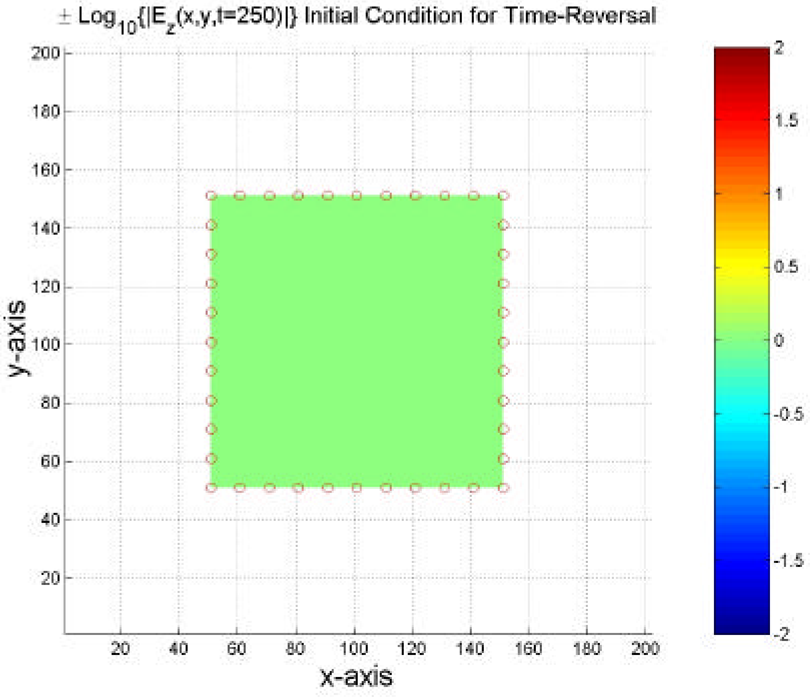


Figure 20. Initial condition of the reversed time step for two aircraft-like targets with M_e= N_e= 10

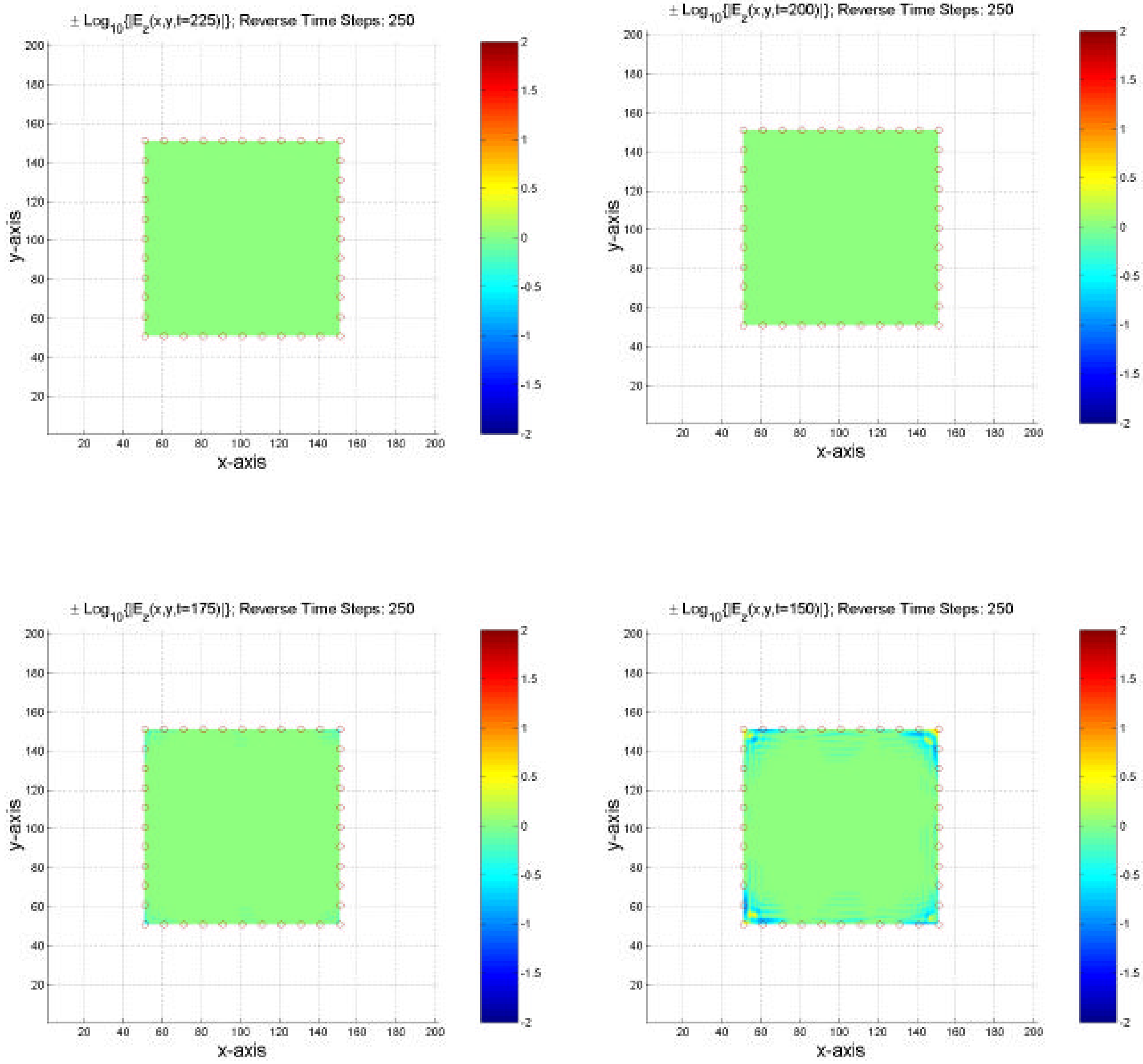


Figure 21. Reversed time step for two aircraft-like targets with $M_e = N_e = 10$ at T= 125, T= 100, T= 75 and T= 50

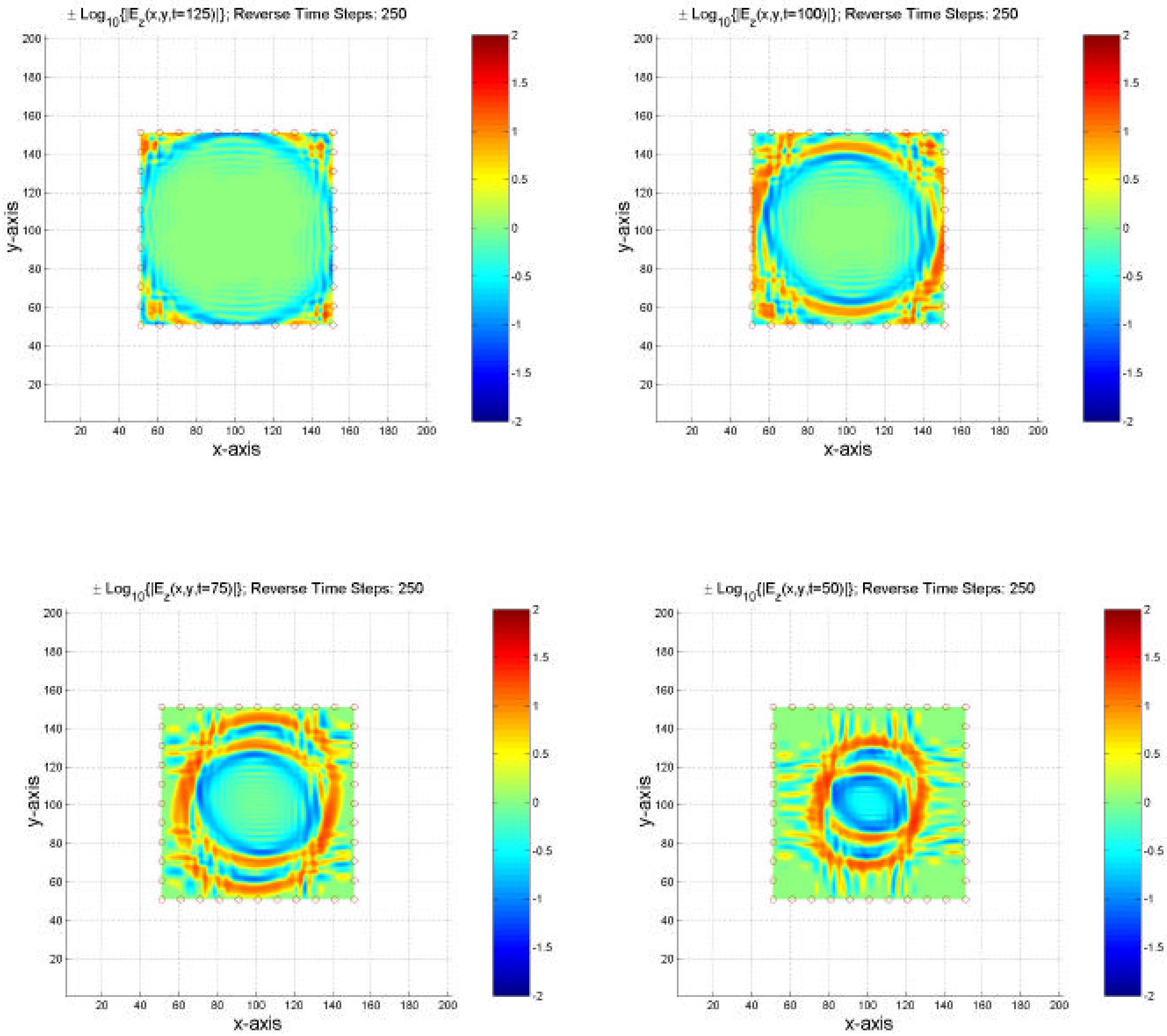
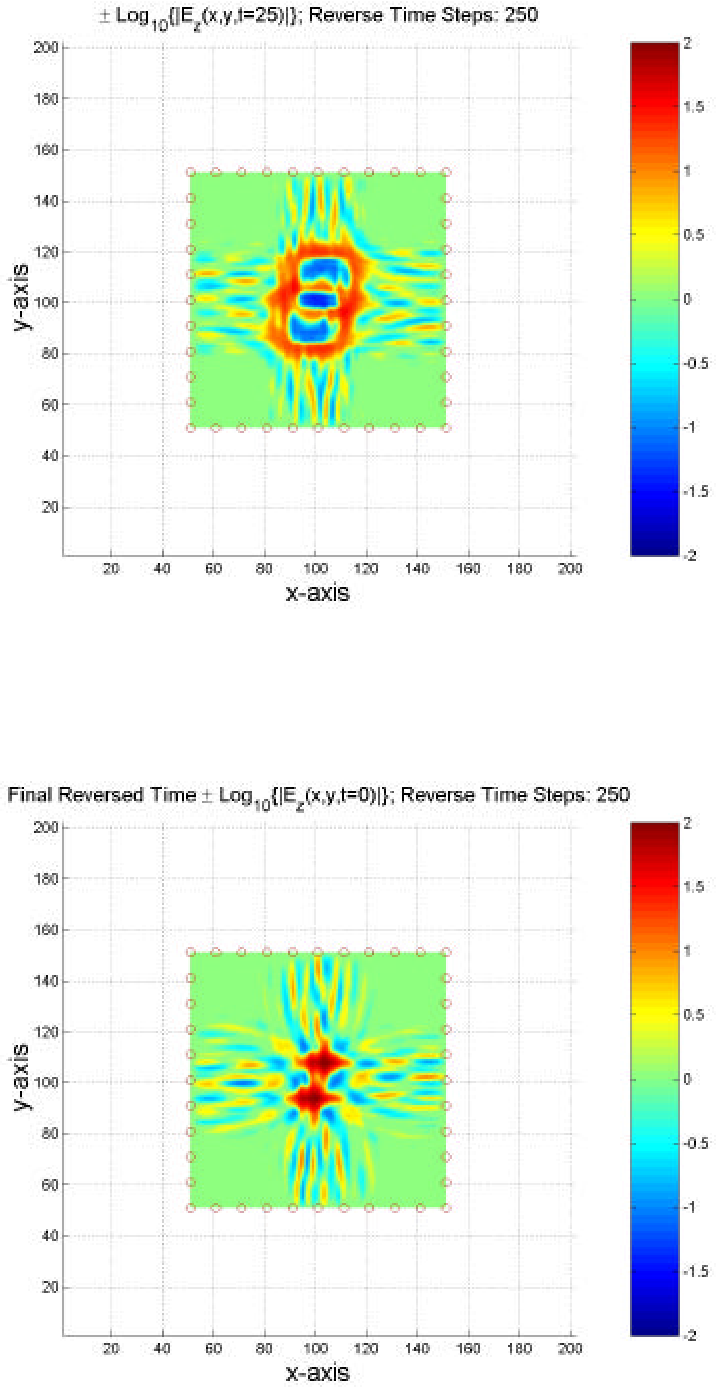


Figure 22. Reversed time step for two aircraft-like targets with M_e = N_e= 10 at T= 125, T= 100,T= 75 and T= 50



$\pm\ \mathrm{Log}_{10}\{|E_z(x,y,t=25)|\}$; Reverse Time Steps: 250

Final Reversed Time $\pm\ \mathrm{Log}_{10}\{|E_z(x,y,t=0)|\}$; Reverse Time Steps: 250

Figure 23. Reversed time step for two aircraft-like targets with $M_e = N_e = 10$ at T= 25 and T= 0

B. TWO POINT-LIKE SCATTERING TARGETS

In the previous example of two aircraft-like targets an outbound field is generated due only to initial conditions without an incident field. In this section we consider an impulse radar example with scattering of a Gaussian plane wave by two point-like metallic targets. The Gaussian plane wave employed in this model has a standard deviation, s =1m, which corresponds to a 3dB effective bandwidth, BW_{3dB} ˜ 125 MHz. In this example, two point-like targets are used to process the reversed time operation. Again, the size of the recording surface is $M_s = N_s = 101$, which is a 100m × 100m square grid for the recording surface and a 200m × 200m square grid for the distant boundary. The forward time step solutions are presented in the next subsection followed by the reversed time step solutions.

1. Forward Time Step Solutions For Two Point-Like Targets

This subsection presents the forward time step solution generated by an impulsive plane wave incident field. Boundary data at the recording surface is recorded at each time step. Two point-like perfect conductor targets forcing the total E-field to zero exist at the approximate center of the grid. The incident field starts marching from x =120. For convenience, the targets are represented by two black dots in all figures and the recording surface is displayed as dashed lines. Figure 24 shows the initial condition of the forward time step solution for two point-like targets with an incident field. Figure25 displays the results at T= 51,T= 101,T= 151 and T= 201. Finally, the result of T= 251 and the final condition at T= 300 are presented on Figure 26.

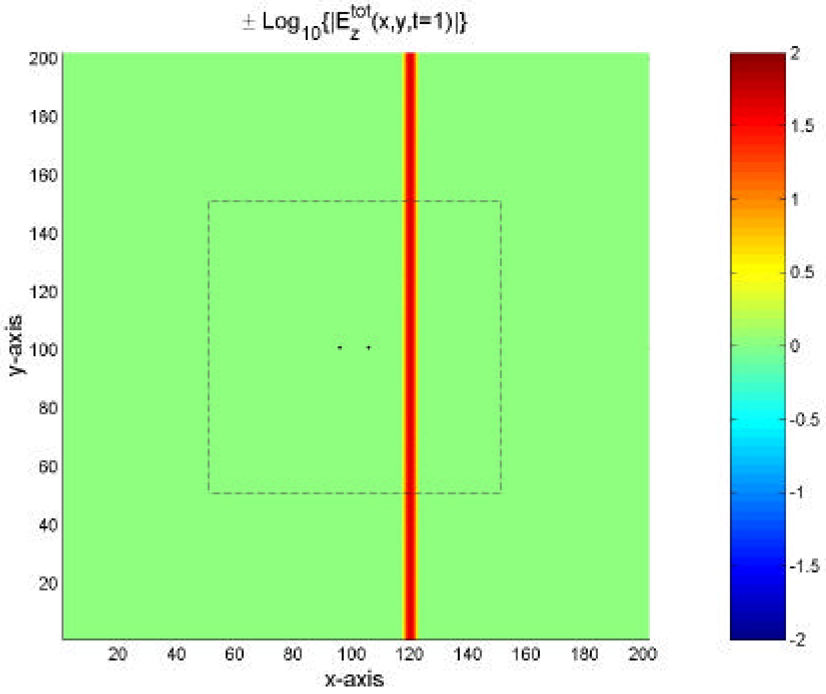


Figure 24. Initial condition of the forward time step solution for two point-like targets

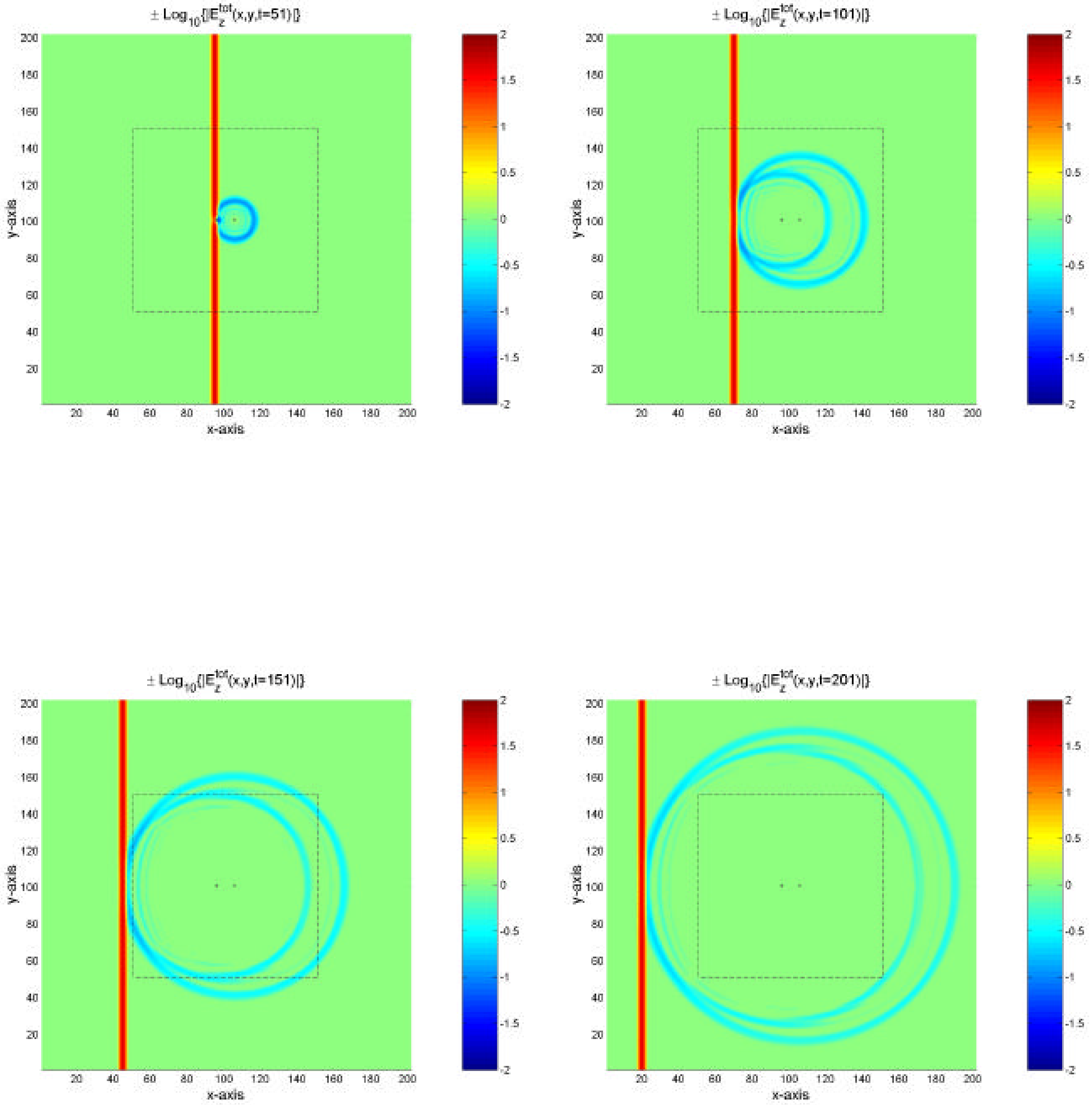


Figure 25. Forward time step for two point-like targets at T= 51, T= 101, T= 151 and T= 201

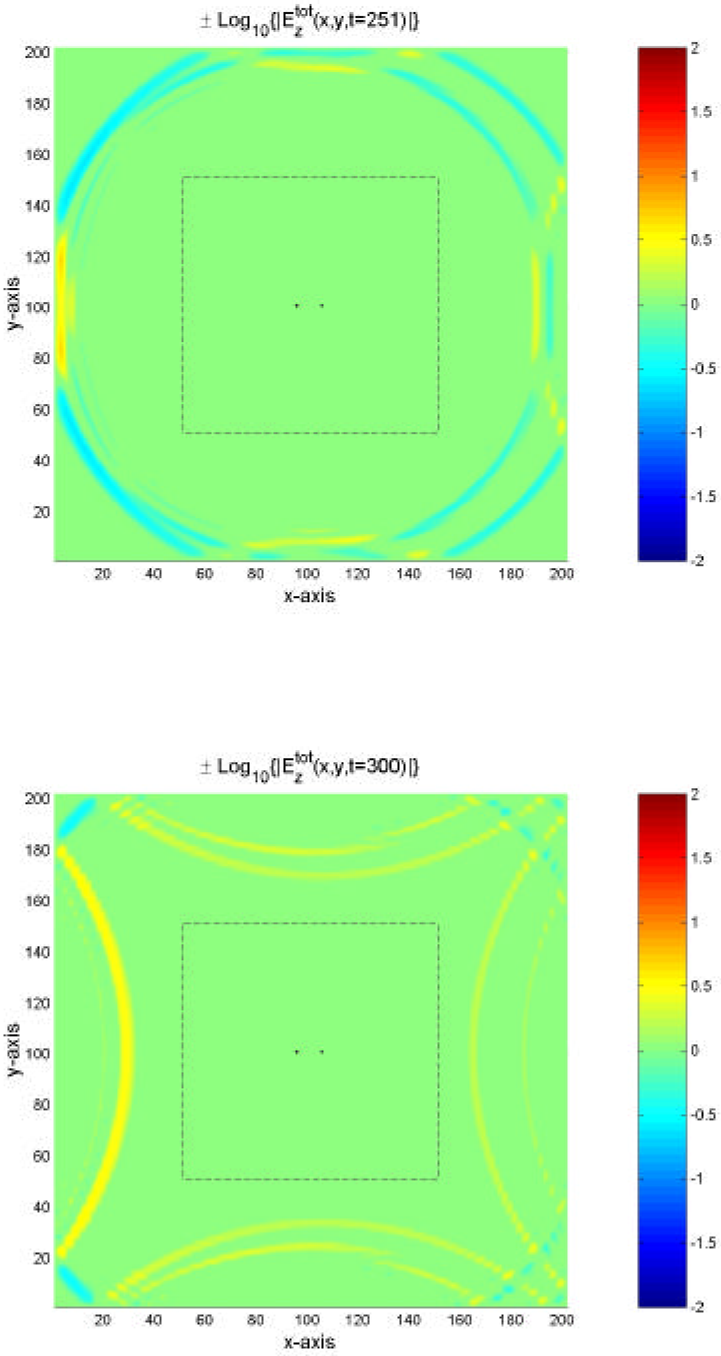


Figure 26. Forward time step for two point-like targets at T= 251 and T= 300

2. Reversed Time Step Solutions For The Two Point-Like Targets

This subsection provides the results of the reversed time step simulation. To simulate the exact time-reversed solution we must enforce the PEC boundary condition of zero total E-field at each target node. Our goal is to recover the targets' shape, which is assumed to be unknown, so determining the correct nodes to enforce the PEC boundary condition will be akin to solving the imaging problem. A correct selection of the target nodes will make the reversed scattered field collapse onto the targets and become completely extinguished at all times prior to the initial impact of the incident plane wave.

As seen in the forward time step results, the time step, T = 200, is a good choice to begin the reversed-time simulation. In this example the spacing of each transducer is assumed to be 1: $M_e = N_e = 1$. Only the scattered field is displayed in this example. For convenience a red dash line is used to represent the location of the incident field peak at each time step. The two point-like targets, represented by two black dots, are located at Target 1: (x, y) = (96,101) and Target 2: (x, y) = (106,101). During the simulation, the scattered field is forced to be the negative incident field at the target nodes. Because the simulation uses the data as exactly recorded on the recording surface during the corresponding forward time step, the scattered field collapses on the targets. Figure 27 shows the initial condition of the reversed time step at T = 200. Figure 28 and 29 display every 25th time step. Figure 30 indicates the accumulated total energy inside the recording surface, where the x-axis corresponds to the reversed time step, T = 200 - t with t being the forward time step. The accumulated total energy inside the recording surface remains constant after the scattered field collapses onto the targets.

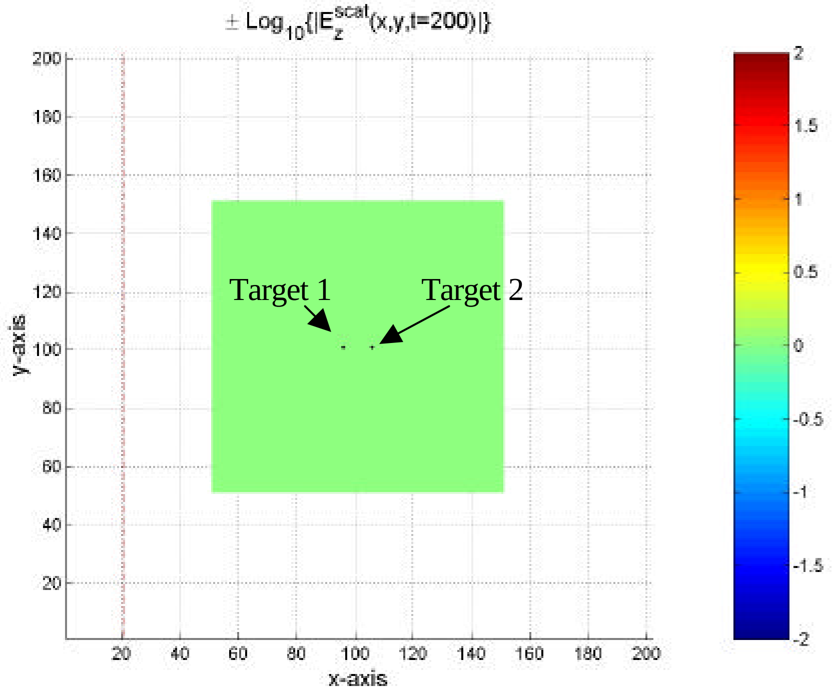


Figure 27. Initial condition of the reversed time step for the exact two point-like targets at T = 200

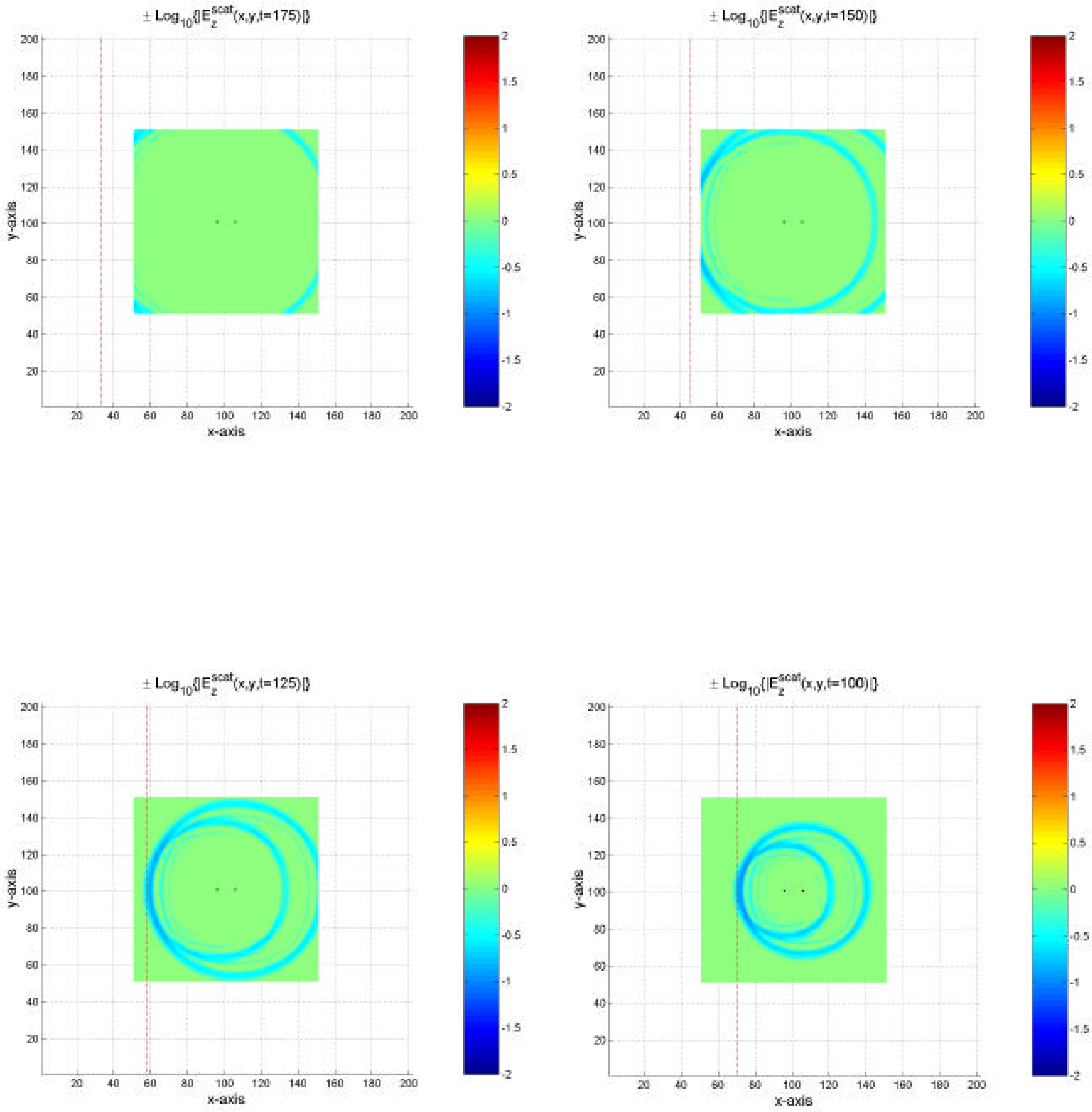


Figure 28. Reversed time step for the exact two point-like targets at T = 175, T = 150,T = 125 and T = 100

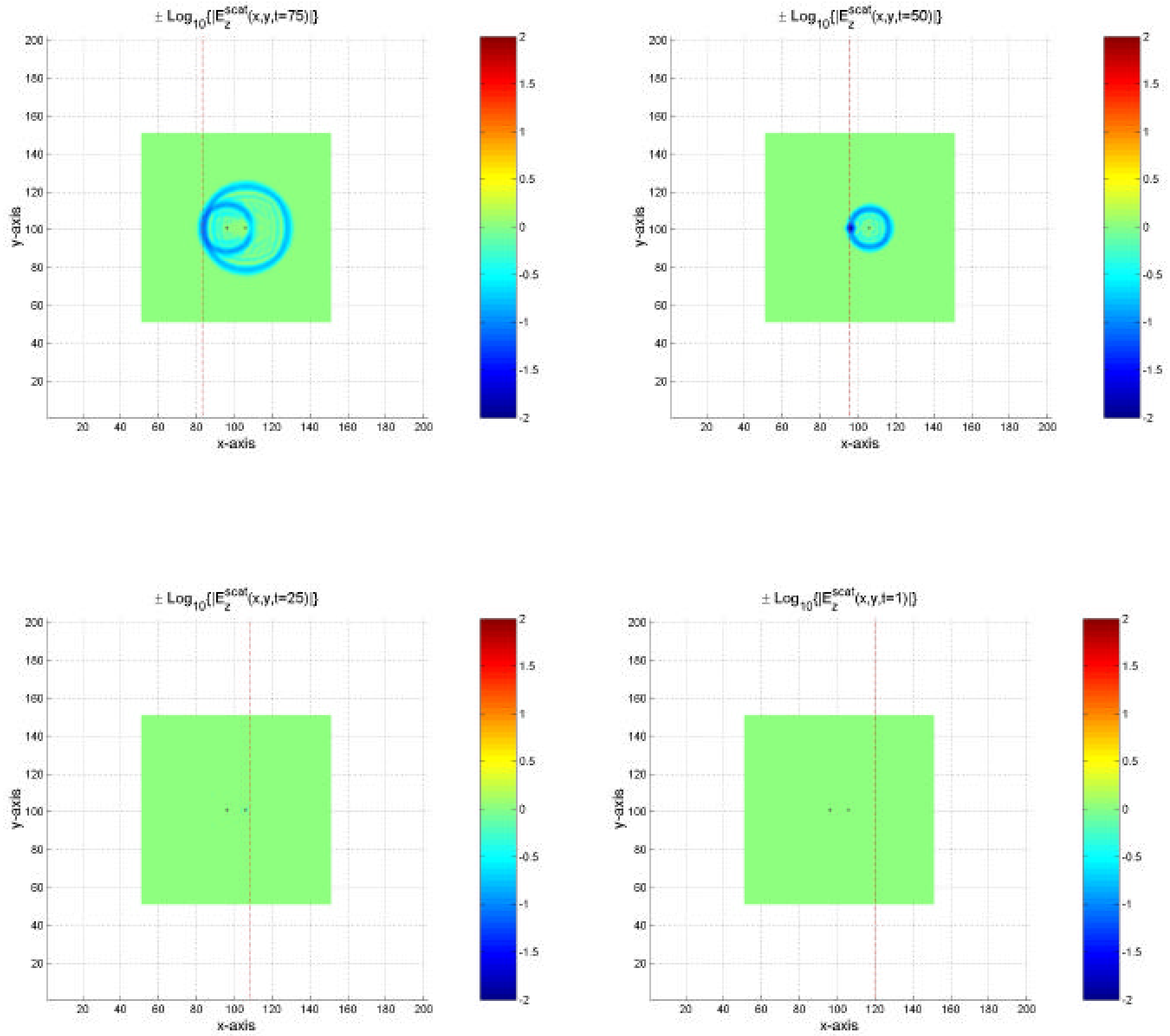


Figure 29. Reversed time step for the exact two point-like targets at T = 75, T = 50,T = 25 and T = 1

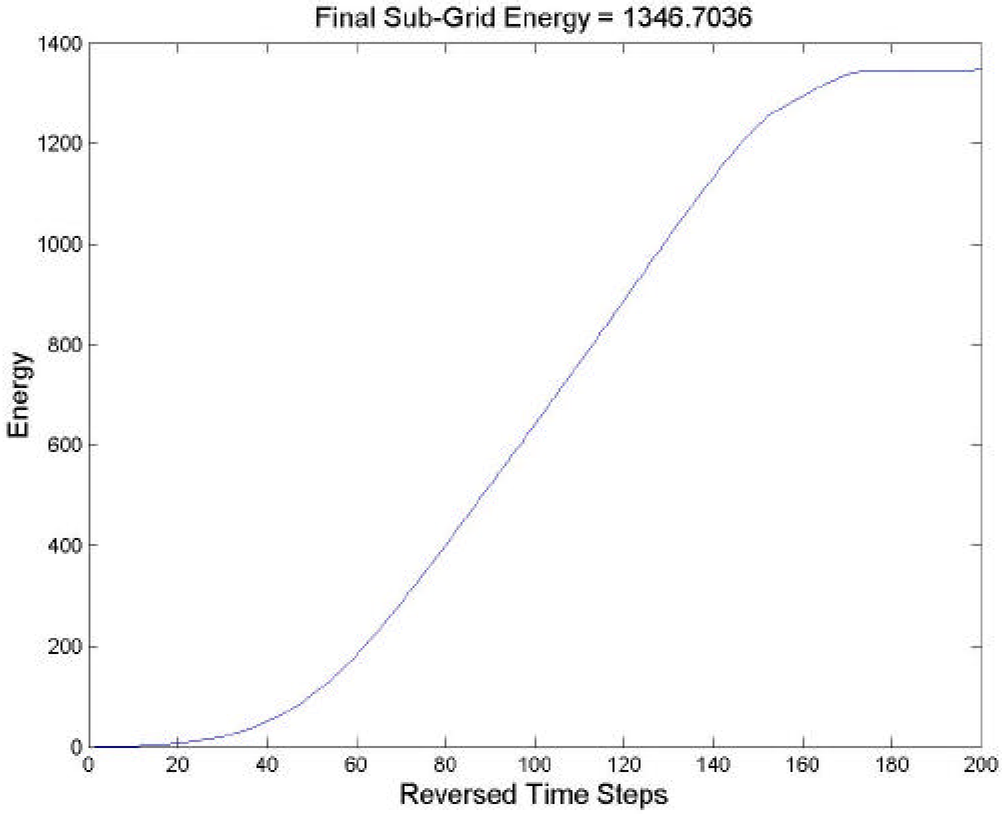


Figure 30. Accumulated total energy inside the recording surface

3. Reversed Time Step Solutions For Perturbed Two Point-like Targets

This subsection provides the reversed time solutions for PEC boundary conditions enforced at locations slightly in error from the exact nodal locations of the two point-like targets.

The negative incident field is enforced at nodes represented by red dots while the exact target node locations are again indicated by black dots. Since the data used for the reversed time step is based on the forward time step solutions in which the targets are located at the correct nodes, the time-reversed scattered field does not become extinguished, but diverges after the reversed field reaches the new targets. Furthermore, the accumulated total energy inside the recording surface keeps increasing although the incident field initially does not reach the targets in the forward time step. To confirm the concept of the accumulated total energy, three cases are examined. In the first case, Target 1 is perturbed, but Target 2 is not perturbed. In the second case, Target 1 stays at the original coordinate, while Target 2 is perturbed. In the last case, both targets are perturbed.

a. Case 1: Target 1 Is Perturbed, Target 2 Is Not

In this case, Target 1 is perturbed to (x, y) = (90,101), but Target 2 is stationed at the correct location, (x, y) = (106, 101). The initial condition at T = 200 is shown in Figure 31. Figure 32 displays the results of every 40th time step. Figure 33 shows the final condition at T = 1. The accumulated total energy inside the recording surface is shown in Figure 34. This case causes the accumulated total energy to increase after the reversed field reaches the new targets.

b. Case 2: Target 1 Is Not Perturbed, Target 2 Is Perturbed

In this case, Target 1 is located at the correct coordinate, (x, y) = (96, 101), but Target 2 is perturbed to (x, y) = (110,101). The initial condition at T=200 is shown in Figure 35. Figure 36 displays every 40th time step and Figure 37 shows the final condition at T=1. Figure 38 shows the accumulated total energy.

c. Case 3: Both Targets Are Perturbed

Target 1 is perturbed to (x, y) = (90,101) and Target 2 is perturbed to (x, y)=(110, 101). Figure 39 shows the initial condition at T = 200. Figure 40 displays every 40th time step and Figure 41 shows the final condition at T = 1. The accumulated total energy is plotted in Figure 42.

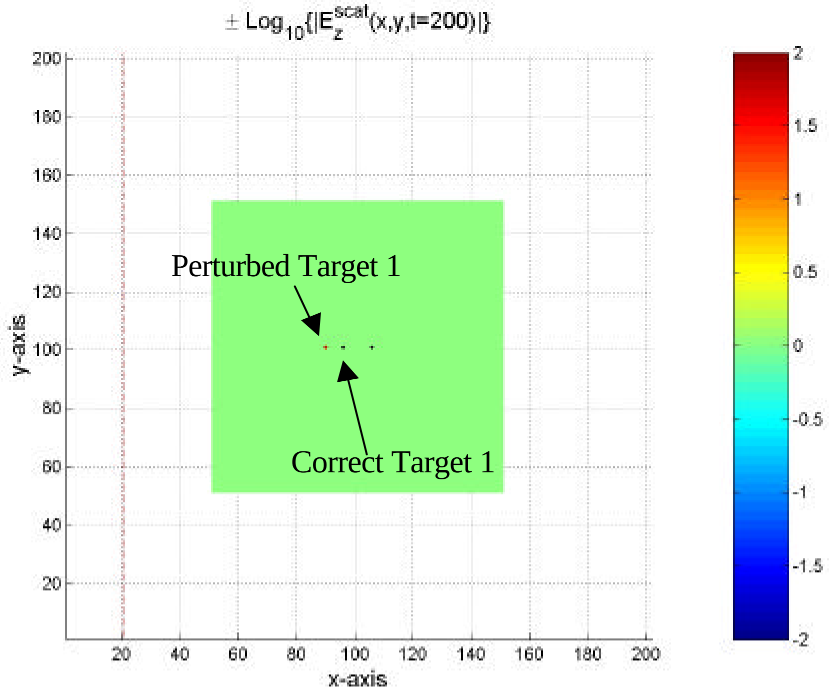


Figure 31. Initial condition of the reversed time step for Case 1

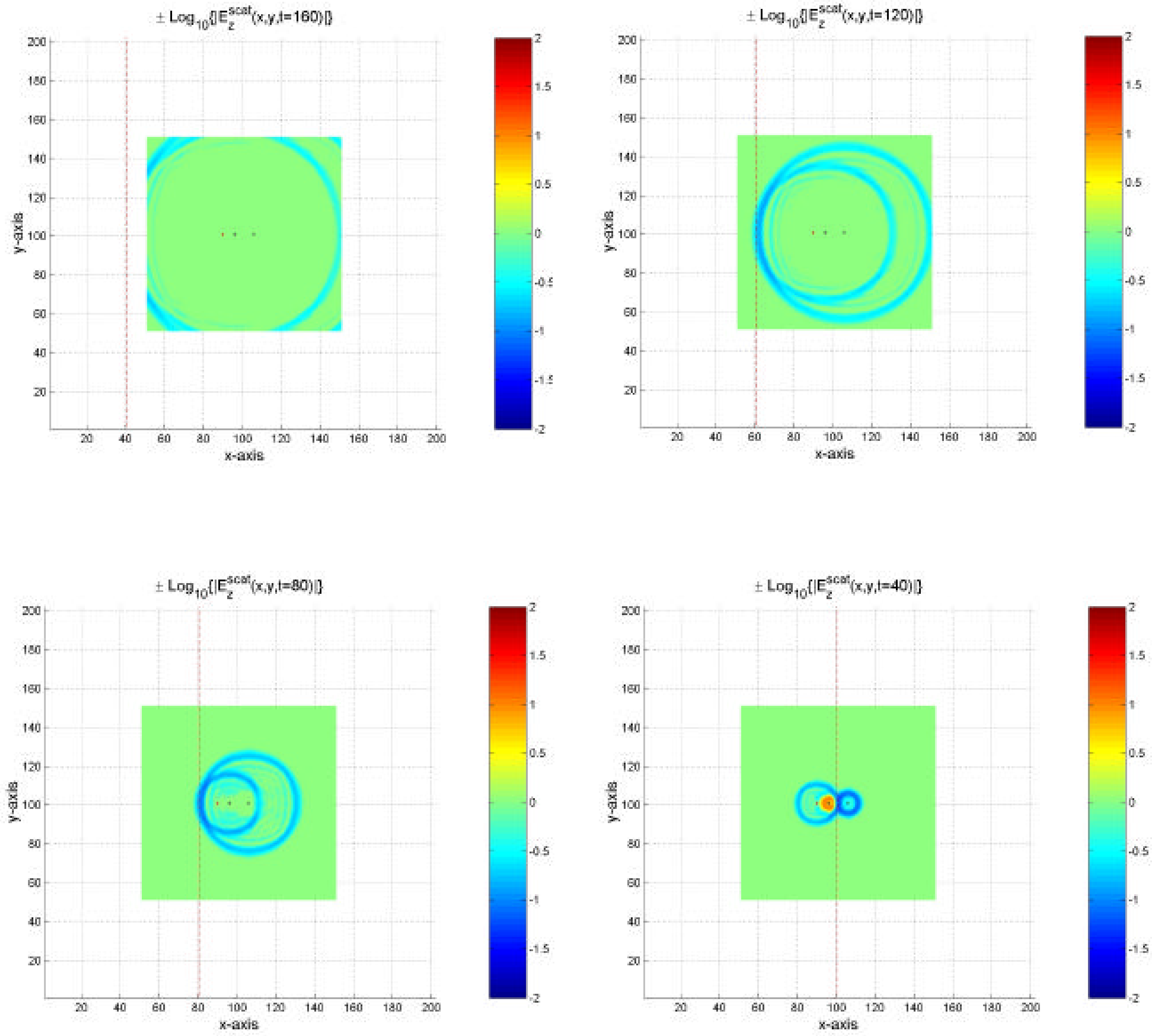


Figure 32. Reversed time step solutions for Case 1 at T = 160, T = 120, T = 80 and T = 40

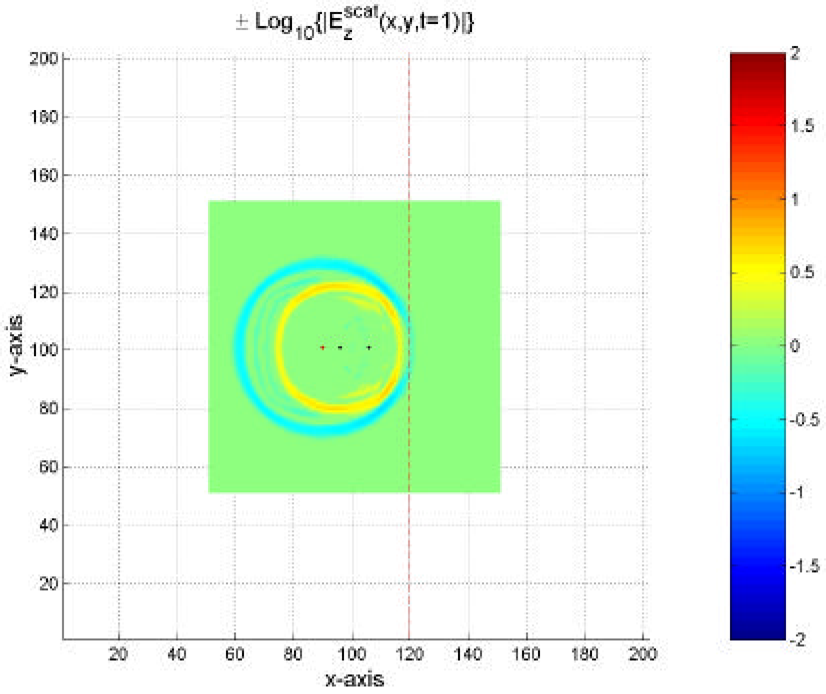


Figure 33. The final condition of the reversed time step for Case 1

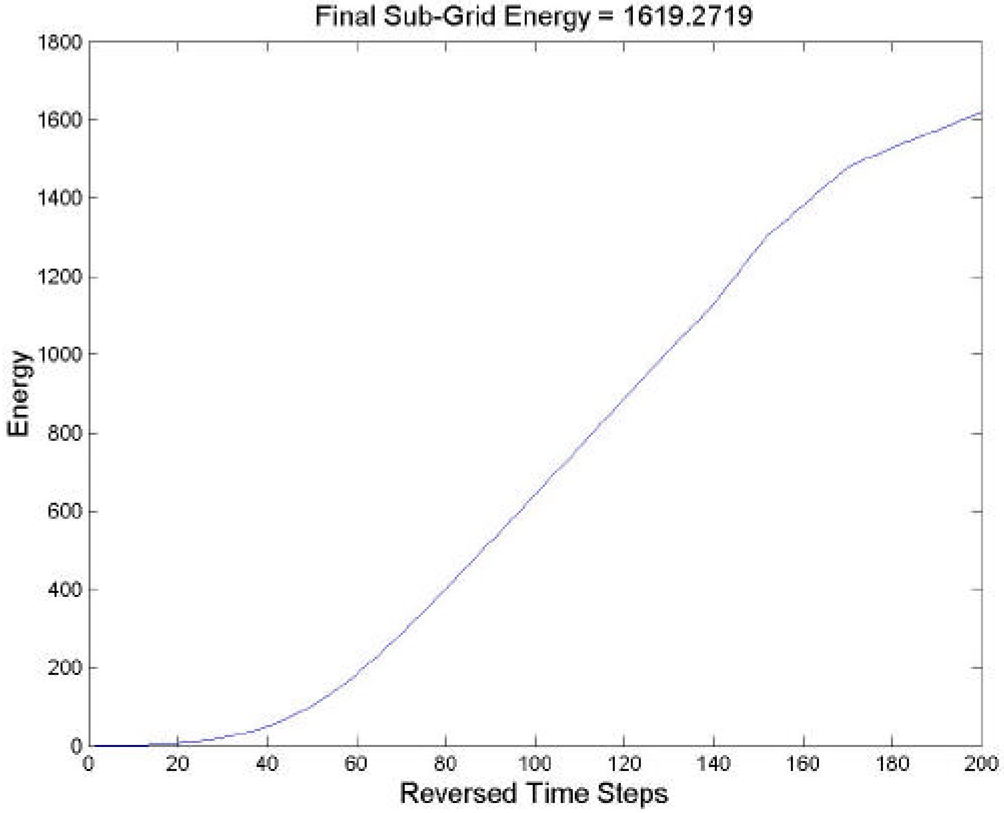


Figure 34. Accumulated total energy inside the recording surface for Case 1

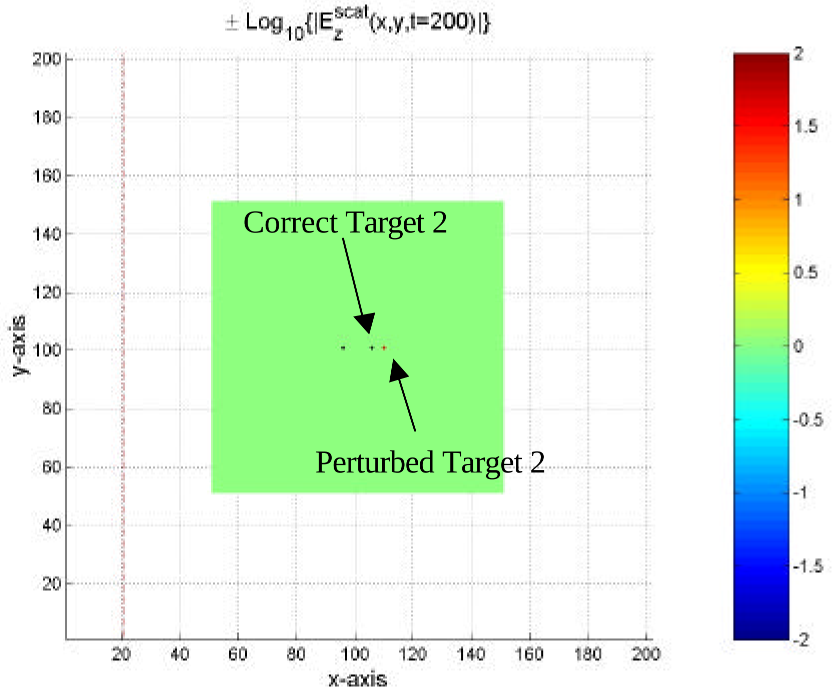


Figure 35. Initial condition of the reversed time step for Case 2

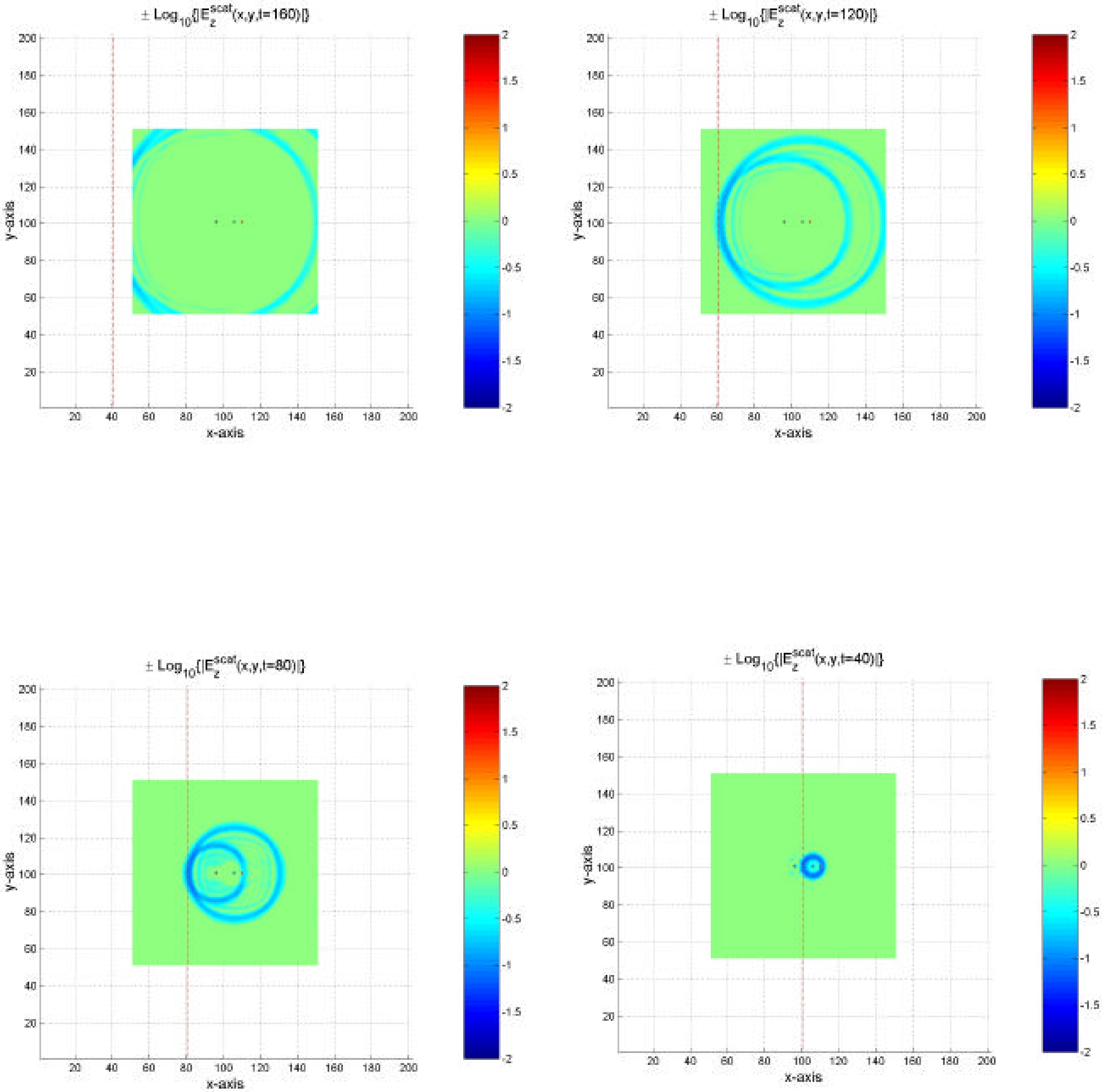


Figure 36. Reversed time step solutions for Case 2 at T = 160, T = 120, T = 80 and T = 40

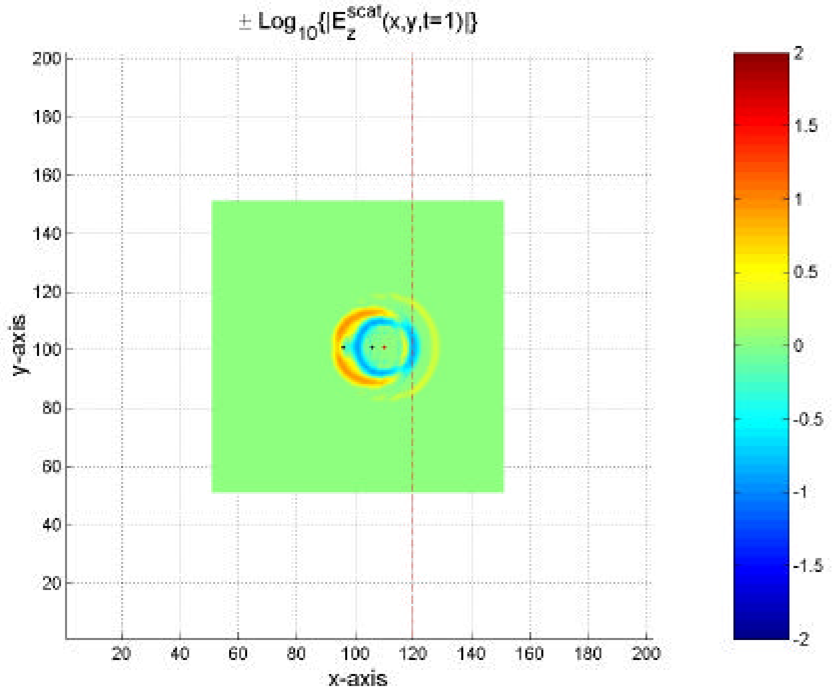


Figure 37. The final condition of the reversed time step for Case 2

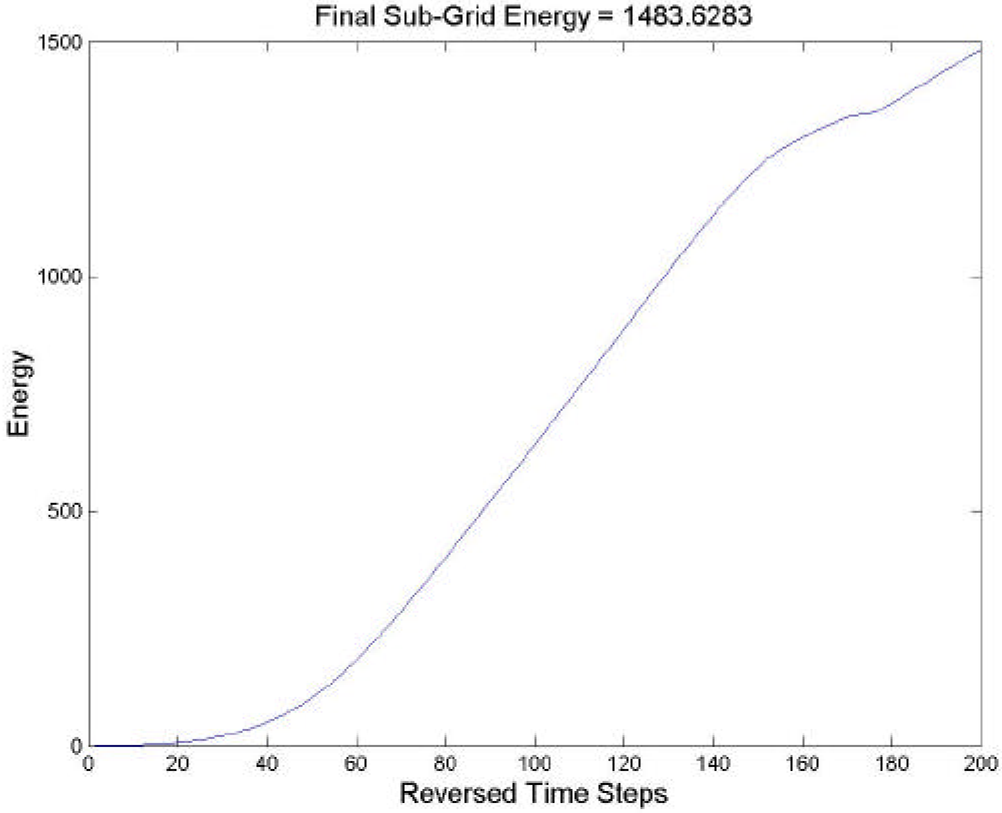


Figure 38. Accumulated total energy inside the recording surface for Case 2

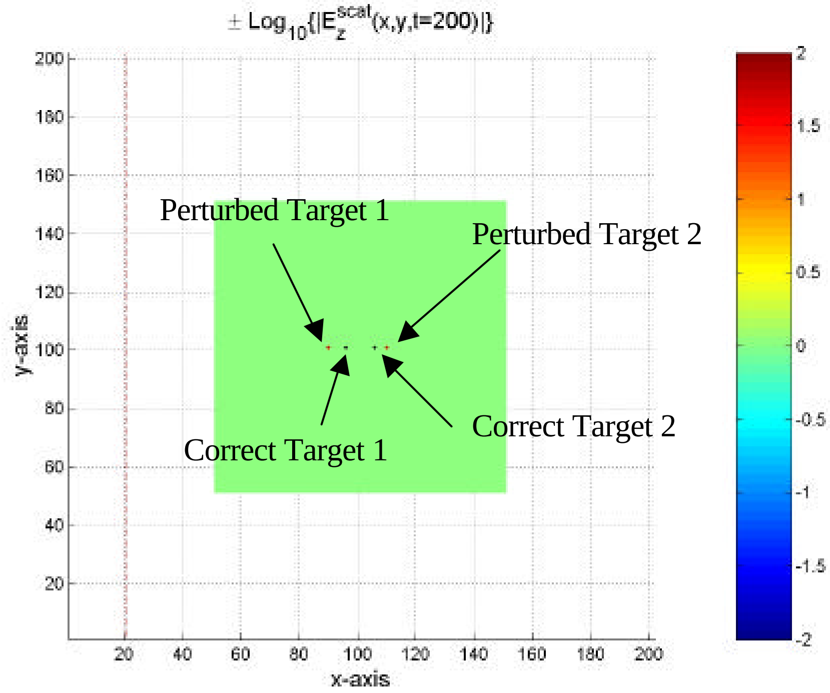


Figure 39. Initial condition of the reversed time step for Case 3

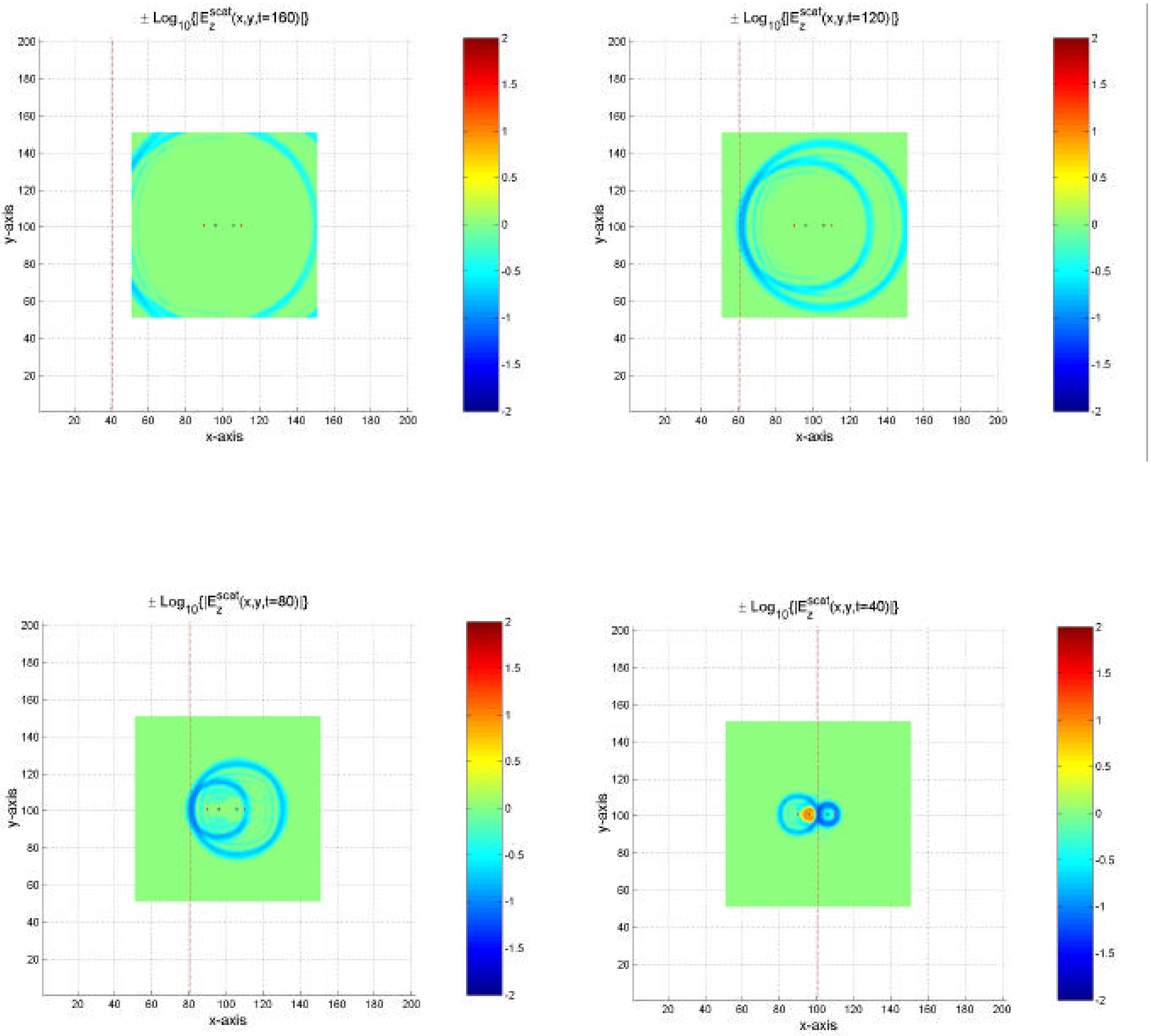


Figure 40. Reversed time step solutions for Case 3 at T = 160, T = 120, T = 80 and T = 40

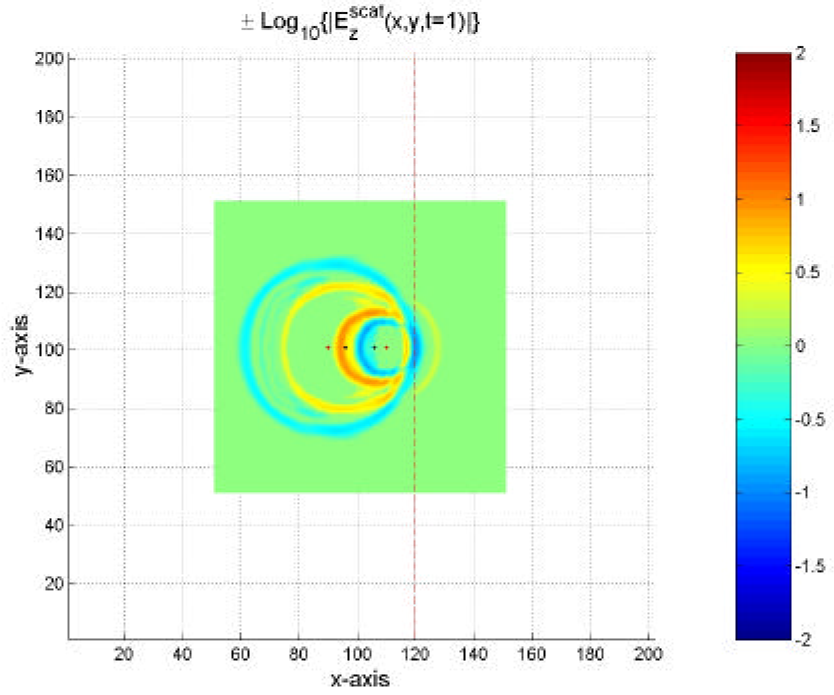


Figure 41. The final condition of the reversed time step for Case 3

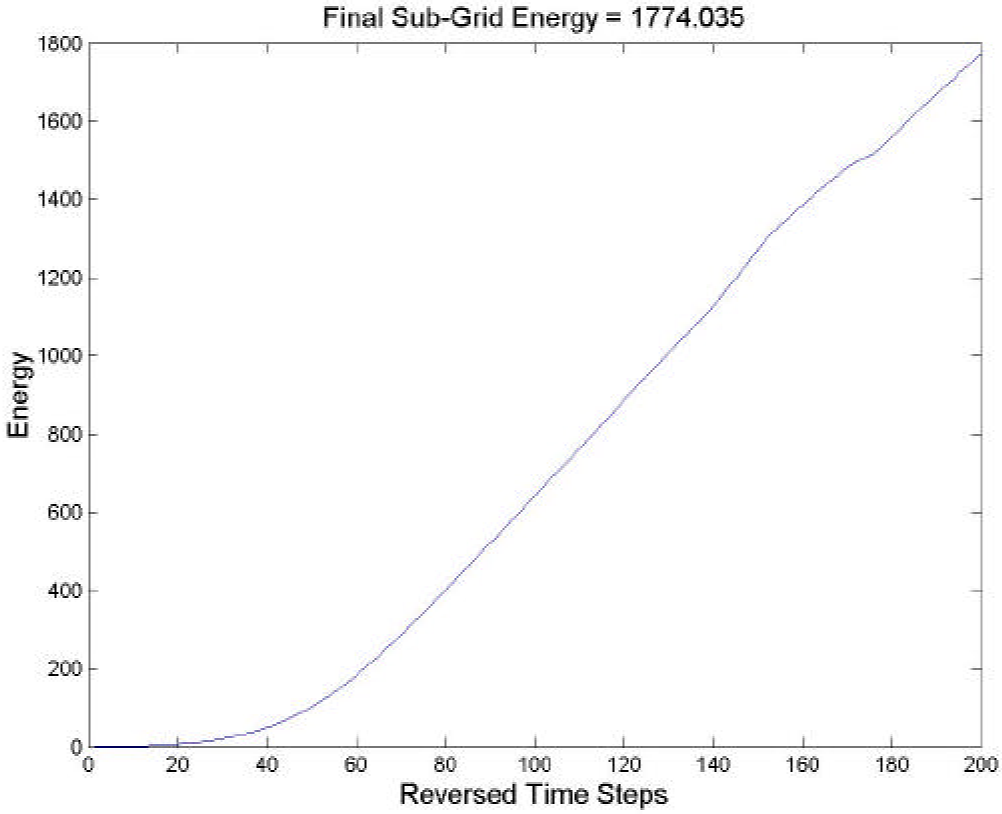


Figure 42. Accumulated total energy inside the recording surface for Case 3

C. SUMMARY

This chapter presented the fundamental examples of the time-reversed process for electromagnetic waves by using the FDTD method. The first example demonstrated the time-reversed process when the target nodes radiate due only to enforced initial conditions without an incident field. Essentially perfect reconstruction of the two aircraft-like target images is obtained when all nodes on the recording surface are used to drive the time-reversed solution. The effect of thinned spacing between the transducers was then examined. This example, however, is not related to the radar scattering case.

The next example, which considered an incident Gaussian impulse plane wave, showed that the time-reversed scattered field collapses onto the point-like targets, with complete extinction only if the PEC boundary condition is enforced on the target nodes during the reversed time step. A wrong selection for any target node location yields a reversed-time scattered field that initially collapses but then expands outward. Incorrect target node selections provide scattered field energy at times earlier than the initial impact of the incident field. The accumulated energy in the scattered field continues to increase for any wrong target node selection as shown in the three cases examined. It thus appears that accumulated energy in the reversed-time scattering solution can be employed as a “cost function” in an optimization routine designed to find the correct target nodes.

V. SUMMARY AND CONCLUSIONS

A. SUMMARY

This thesis presents an initial attempt to demonstrate target imaging by using a numerical time-reversed process for the radar imaging applications, in particular, for bi-static or multi-static radars.

Demonstration of the time-reversed process is done by the FDTD method. The first example shows that the time-reversed process can image multiple targets when the only driving function for the field is initial conditions at the target nodes. This, however, is not the radar scattering case.

The radar scattering case was examined next, albeit for a simple example of two point targets illuminated by an incident Gaussian pulse plane wave. A perfect time-reversed simulation provides a scattered field that collapses onto the target nodes and becomes extinguished for all times earlier than the initial plane wave impact. Such a solution requires enforcement of metallic boundary conditions (scattered field equals negative of the incident field) at each target node. This, of course, requires a priori knowledge of the target location, orientation and geometry – the very knowledge being sought in the time-reversed process. All is not lost however, by what appears to be a chicken and egg dilemma. Wrong selections of even one target node for enforcing the metallic boundary condition yields time-reversed scattering solutions which first collapse then radiate outward from the target region rather than being extinguished. The accumulated energy in the solution can thus be used as a cost function for optimization of the scattering nodes, with the minimum cost solution corresponding to the correct locations.

B. CONCLUSIONS AND RECOMMENDATIONS

The function of a radar imaging system is to gain early identification of non-cooperative targets. The objective of this thesis was to initially investigate the potential utility of the time-reversed numerical process for employment in multi-static radar imaging applications. Using an initial condition solution as the first case to investigate, it was found that superb imaging is possible for multiple targets when all boundary nodes, without thinning, are used to drive the time-reversed solution. The radar scattering case, however, is a different animal that will require solution of an optimization problem where the target nodes are the parameters to be optimized (correctly placed) and the cost function is the accumulated scattered field energy in the time-reversed numerical solution.

Future work on this subject should include investigation of optimization algorithms for solution of the correct target nodes. The genetic algorithm may be the best bet for this task since it is robust in exploring a wide range of local cost minima in solving for the best global solution within the parameter space. In addition, a three-spatial dimension model supporting volumetric regions should be developed. To represent a realistic environment, this model should include provision for uneven spacing of the transducers with 3-D targets. The model should be evaluated relative to required operating frequencies, signal-to-noise ratio and the number and material properties of realistic targets.

APPENDIX. PROGRAM LISTINGS

A. PROGRAM LISTINGS

These programs are used to demonstrate the forward scattering and the time-reversed process in MATLAB. The program used for the first example allows inputs for the number of transducers, the spacing between the transducers, the stability condition, time step, the initial condition of the region and the dynamic range for the plot. The program used for the second example allows inputs for the number of transducers, time step, the initial position of the incident field and the standard deviation of the incident field. Target functions are also listed.

These programs are written by Prof. M. A. Morgan and modified by LT. Inaba.

1. Program RTWE_2D5 For The First Example

```
Mp=input('Enter number of field sensors on each x-segment of the sub-grid boundary: ');
Me=input('Enter node spacing between each x-segment subgrid boundary sensor: ');
Np=input('Enter number of field sensors on each y-segment of the sub-grid boundary: ');
Ne=input('Enter node spacing between each y-segment subgrid boundary sensor: ');

Ms=(Mp-1)*Me + 1; Ns=(Np-1)*Ne + 1;
disp(['Subgrid: Ms=',int2str(Ms),'; Ns=',int2str(Ns)]);

% Default Mesh
M=2*Ms; N=2*Ns;

m1=fix((M-Ms)/2)+1; m2=m1+Ms-1;
n1=fix((N-Ns)/2)+1; n2=n1+Ns-1;
x=1:M; y=1:N; Y=y'*ones(1,M); X=ones(N,1)*x;
xs=m1:m2; ys=n1:n2; Ys=ys'*ones(1,Ms); Xs=ones(Ns,1)*xs;
disp(['Grid: M=',int2str(M),' m1=',int2str(m1),' m2=',int2str(m2),...
     '; N=',int2str(N),' n1=',int2str(n1),' n2=',int2str(n2)]);
xe=(m1:Me:m2)'; ye=(n1:Ne:n2)';    % sub-grid sensor x and y node numbers

disp('Note q >= sqrt(2) Courant requirement for convergence ');
q=input('Enter Integer Value for q=dh/(c*dt) (1, 2, etc): ');

Nmin=min(1.5*M-m2,1.5*N-n2); Pmax=fix(q*Nmin);

P=input(['Enter number of time-steps ( < ',int2str(Pmax),') : ']);
```

```
name1=input('Enter Forward Soln Files "Name" for  Name_nn.jpg ? (Enter Key to Skip):
if ~isempty(name1),
   dp1=input('Enter Time Step Increment Between Stored Frames: ');
   pj1=1:dp1:P;
   nj=1; njmx=length(pj1);
end

Q1=1/(q*q); Q2=(2-4*Q1);

Ez=zeros(N,M,P); % Reserving 3D array
EzIC=zeros(N,M);
EzBC1=zeros(Ms,P); EzBC2=zeros(Ns,P); EzBC3=EzBC1; EzBC4=EzBC2;
% Subgrid BC's

% Constructing Forward-Time Evolution Arrays
d=ones(1,max(M,N));
A=diag(d(1:N-1),1)+diag(d(1:N-1),-1);
B=diag(d(1:M-1),1)+diag(d(1:M-1),-1);

% Define Basic Aircraft Shaped Grid Locations Centered at y=x=0
[ny,mx]=Air2(N,M);
Ntgt=length(ny);

% Defining IC Node Centers
for nic=1:Ntgt
   EzIC(ny(nic),mx(nic))=1;
end

% Define Unit Peak Gaussian Shaped Initial Condition
s=input('Enter IC Std Dev in Grid Spaces (0 to use point ICs): ');
if ~isempty(s),
   if s > 0,
   [ny mx]=find(EzIC==1);               % Array locations for IC nodes
   Nic=length(ny);
   for n=1:Nic
      Xc=X-x(mx(n)); Yc=Y-y(ny(n));
      R2=(1/(2*s*s))*(Xc.*Xc+Yc.*Yc);
      Ez(:,:,1)=Ez(:,:,1) + exp(-R2);   % Gaussian spread about each IC node
   end
   else,
   Ez(:,:,1)=EzIC;                      % Using unit node IC's for s <= 0
   end
end
if isempty(s), Ez(:,:,1)=EzIC; end

% Initial Subgrid BC's
if Me == 1,
   EzBC1(:,1)=Ez(n1,xs,1).'; EzBC3(:,1)=Ez(n2,xs,1).';
else,
   EzBC1(:,1)=spline(xe,Ez(n1,xe,1).',xs');
   EzBC3(:,1)=spline(xe,Ez(n2,xe,1).',xs');
```

```
end
if Ne == 1,
   EzBC2(:,1)=Ez(ys,m2,1); EzBC4(:,1)=Ez(ys,m1,1);
else,
   EzBC2(:,1)=spline(ye,Ez(ye,m2,1),ys');
   EzBC4(:,1)=spline(ye,Ez(ye,m1,1),ys');
end

% Assuming dEz/dt=G=0 at p=1 to take initial step to p=2 (see 5/10/00 note)
Ez(:,:,2)=0.5*(Q1*(A*Ez(:,:,1) + Ez(:,:,1)*B) + Q2*Ez(:,:,1));
% p=2 Subgrid BC's
if Me == 1,
   EzBC1(:,2)=Ez(n1,xs,2).'; EzBC3(:,2)=Ez(n2,xs,2).';
else,
   EzBC1(:,2)=spline(xe,Ez(n1,xe,2).',xs');
   EzBC3(:,2)=spline(xe,Ez(n2,xe,2).',xs');
end
if Ne == 1,
   EzBC2(:,2)=Ez(ys,m2,2); EzBC4(:,2)=Ez(ys,m1,2);
else,
   EzBC2(:,2)=spline(ye,Ez(ye,m2,2),ys');
   EzBC4(:,2)=spline(ye,Ez(ye,m1,2),ys');
end

for p=3:P;            % Equation of Evolution
Ez(:,:,p)=Q1*(A*Ez(:,:,p-1)+Ez(:,:,p-1)*B)+Q2*(Ez(:,:,p-1)-Ez(:,:,p-2));
% Explicitely Enforce Ez=0 BC's on Grid Boundary For Update
Ez(1,:,p)=zeros(1,M); Ez(N,:,p)=zeros(1,M);
Ez(:,1,p)=zeros(N,1); Ez(:,M,p)=zeros(N,1);
Ezmx(p)=max(max(abs(Ez(:,:,p))));
% Subgrid BC Update
if Me == 1,
   EzBC1(:,p)=Ez(n1,xs,p).'; EzBC3(:,p)=Ez(n2,xs,p).';
else,
   EzBC1(:,p)=spline(xe,Ez(n1,xe,p).',xs');
   EzBC3(:,p)=spline(xe,Ez(n2,xe,p).',xs');
end
if Ne == 1,
   EzBC2(:,p)=Ez(ys,m2,p); EzBC4(:,p)=Ez(ys,m1,p);
else,
   EzBC2(:,p)=spline(ye,Ez(ye,m2,p),ys');
   EzBC4(:,p)=spline(ye,Ez(ye,m1,p),ys');
end
end

while 1
disp('Movie Modes: ')
disp('1 --> 2-D log-scale color plan view')
disp('2 --> 2-D linear scale color plan view')
disp('3 --> 3-D linear scale copper tint')
vsel=input('Select Choice: ');
```

```
if ~isempty(vsel), if vsel==1 | vsel==2 | vsel==3, break; end; end
end

EzMax=max(Ezmx);

if vsel==1,
% Using bi-polar log scaling to retain Ez polarity with selected dynamic range
DR=input('Enter 2-D Plot Dynamic Range, e.g. 100, 1000,... : ');
SFac=EzMax/DR; Cmax=log10(DR); C=[-Cmax Cmax]; Zc=2*Cmax;
XBC=[m1 m2 m2 m1 m1]'; YBC=[n1 n1 n2 n2 n1]'; ZBC=Zc*[1 1 1 1 1]';
 % sub-grid outline
XBS=[xe; xe; m1*ones(Np,1); m2*ones(Np,1)];
 % sub-grid sensor x and y node numbers
YBS=[n1*ones(Mp,1); n2*ones(Mp,1); ye; ye];
ZBS=Zc*ones(2*(Mp+Np),1);
end

if vsel==2,
% Using Linear Ez plot
C=[-EzMax EzMax]; Zc=2*EzMax;
XBC=[m1 m2 m2 m1 m1]'; YBC=[n1 n1 n2 n2 n1]'; ZBC=Zc*[1 1 1 1 1]';
% sub-grid outline
XBS=[xe; xe; m1*ones(Np,1); m2*ones(Np,1)];
  % sub-grid sensor x and y node numbers
YBS=[n1*ones(Mp,1); n2*ones(Mp,1); ye; ye];
ZBS=Zc*ones(2*(Mp+Np),1);
end

if vsel==3, v(5)=-EzMax; v(6)=EzMax; end

% Initializing Frames
clf reset; v(1)=1; v(2)=M; v(3)=1; v(4)=N;

if vsel==1,
   EzScl=Ez(:,:,1)/SFac;                % scaling so abs(EzScl) <= DR
   [p1 q1]=find(0 <= EzScl & EzScl < 1); % nonlinear remapping for |EzScl| < 1
   EzScl(p1+(q1-1)*N)= 1;                 % to zero log plot
   [p1 q1]=find(-1 < EzScl & EzScl < 0);
   EzScl(p1+(q1-1)*N)= -1;
   EzLog=sign(Ez(:,:,1)).*log10(abs(EzScl));

   surf(X,Y,EzLog); shading interp

   title(['Initial Condition \pm Log_{10}\{|E_z(x,y,t=0)|\}; Forward Time Steps: '...
         ,int2str(P)],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis equal; axis(v); caxis(C); colorbar
   view(0,90); hold on
   plot3(XBC,YBC,ZBC,'--k');
   if Me ~= 1 | Ne ~= 1, plot3(XBS,YBS,ZBS,'or'); end
   hold off
```

```
end

if vsel==2,
   surf(X,Y,Ez(:,:,1)); shading interp
   title(['Initial Condition E_z(x,y,t=0); Forward Time Steps: '...
        ,int2str(P)],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis equal; axis(v); caxis(C); colorbar
   view(0,90); hold on
   plot3(XBC,YBC,ZBC,'--k');
   if Me ~= 1 | Ne ~= 1, plot3(XBS,YBS,ZBS,'or'); end
   hold off
end

if vsel==3, surfl(Y,X,Ez(:,:,1));
   title(['Initial Condition E_z(x,y,t=0); Forward Time Steps: ',int2str(P)],...
      'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis(v); colormap(copper);
end

Frame=moviein(P,gcf);

hcpy=input('Print a Hard Copy ? (Y/N): ','s');
if ~isempty(hcpy), if hcpy == 'Y' | hcpy == 'y', print; end; end
name=input('Enter "Name" to Save as Name.jpg File ? (Hit Enter Key to Skip): ','s');
if ~isempty(name), eval(['print ',name,' -djpeg99']); end

for p=1:P;    % Recording animation frames
   if vsel==1,
   EzScl=Ez(:,:,p)/SFac;                  % scaling so abs(EzScl) <= DR
   [p1 q1]=find(0 <= EzScl & EzScl < 1); % nonlinear remapping for |EzScl| < 1
   EzScl(p1+(q1-1)*N)= 1;                 % to zero log plot
   [p1 q1]=find(-1 < EzScl & EzScl < 0);
   EzScl(p1+(q1-1)*N)= -1;
   EzLog=sign(Ez(:,:,p)).*log10(abs(EzScl));
   surf(X,Y,EzLog); shading interp
   title(['\pm Log_{10}\{|E_z(x,y,t=',int2str(p),')|\}; Forward Time Steps: ',...
        int2str(P)],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis equal; axis(v); caxis(C); colorbar
   view(0,90); hold on
   plot3(XBC,YBC,ZBC,'--k');
   if Me ~= 1 | Ne ~= 1, plot3(XBS,YBS,ZBS,'or'); end
   hold off
     if ~isempty(name1),
        if p == pj1(nj), name=[name1 '_' int2str(nj)];
           eval(['print ',name,' -djpeg99']); nj=min(nj+1,njmx);
     end; end
end
```

```
  if vsel==2,
  surf(X,Y,Ez(:,:,p)); shading interp
  title(['E_z(x,y,t=',int2str(p),'); Forward Time Steps: ',...
        int2str(P)],'FontSize',14)
  xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
  axis equal; axis(v); caxis(C); colorbar
  view(0,90); hold on
  plot3(XBC,YBC,ZBC,'--k');
  if Me ~= 1 | Ne ~= 1, plot3(XBS,YBS,ZBS,'or'); end
  hold off
    if ~isempty(name1),
      if p == pj2(nj), name=[name1 '_' int2str(nj)];
        eval(['print ',name,' -djpeg99']); nj=min(nj+1,njmx);
    end; end
  end

  if vsel==3, surfl(X,Y,Ez(:,:,p))
  title(['E_z(x,y,t=',int2str(p),'); Forward Time Steps: ',int2str(P)],'FontSize',14)
  xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
  axis(v); colormap(copper);
    if ~isempty(name1),
      if p == pj1(nj), name=[name1 int2str(nj)];
        eval(['print ',name,' -djpeg99']); nj=min(nj+1,njmx);
    end; end
  end

Frame(p)=getframe(gcf);
end

if vsel==1, title(['Final Time Step \pm Log_{10}\{|E_z(x,y,t=',...
      int2str(P),')|\}'],'FontSize',14)
else
  title(['Final Time Step E_z(x,y,t=',int2str(P),')'],'FontSize',14)
end

hcpy=input('Print a Hard Copy ? (Y/N): ','s');
if ~isempty(hcpy), if hcpy == 'Y' | hcpy == 'y', print; end; end
name=input('Enter "Name" to Save as Name.jpg File ? (Hit Enter Key to Skip): ','s');
if ~isempty(name), eval(['print ',name,' -djpeg99']); end

FrameFile=input('Enter File Name to Save Movie Frame Array as *.mat
 (Press Enter to Skip): ','s');
if ~isempty(FrameFile), eval(['save ',FrameFile,' M N Frame']); end

clear Frame

% Computation Using Reverse-Time BC's Stored From Sub-Grid Boundary

% Constructing Sub-Grid Evolution Arrays
ds=ones(1,max(Ms,Ns));
As=diag(ds(1:Ns-1),1)+diag(ds(1:Ns-1),-1);
```

```
Bs=diag(ds(1:Ms-1),1)+diag(ds(1:Ms-1),-1);

Ps=input('Enter time-step to begin time-reversed BC data: ');

name2=input('Enter Reverse Soln "Name" for Name_nn.jpg ? (Enter Key to Skip): ','s');
if ~isempty(name2),
   dp2=input('Enter Time Step Increment Between Stored Frames: ');
   pj2=1:dp2:Ps;
   nj=1; njmx=length(pj2);
end

disp('Select Reverse-Time Data to Use: ')
disp('  1 ==> Full Grid IC and Full Stored BCs (Exact Reverse-Time Solution)')
disp('  2 ==> No ICs and Stored BC Subset (Realistic Measured Boundary Data)')
Rdat=input('Make Selection: ');

Ezs=zeros(Ns,Ms,Ps); % Reserving 3D array spaces
if Rdat == 1,
% Set IC's at p=Ps and p=Ps-1
Ezs(:,:,1)=Ez(ys,xs,Ps); Ezs(:,:,2)=Ez(ys,xs,Ps-1);
end

if Rdat == 2,
% Explicitely Enforce Ez=0 BC's on Grid Boundary For Update
Ezs(1,:,1)=EzBC1(:,Ps); Ezs(:,Ms,1)=EzBC2(:,Ps).';
Ezs(Ns,:,1)=EzBC3(:,Ps); Ezs(:,1,1)=EzBC4(:,Ps).';
Ezs(1,:,2)=EzBC1(:,Ps-1); Ezs(:,Ms,2)=EzBC2(:,Ps-1).';
Ezs(Ns,:,2)=EzBC3(:,Ps-1); Ezs(:,1,2)=EzBC4(:,Ps-1).';
end

for p=3:Ps;           % Equation of Evolution
Ezs(:,:,p)=Q1*(As*Ezs(:,:,p-1)+Ezs(:,:,p-1)*Bs)+Q2*Ezs(:,:,p-1)-Ezs(:,:,p-2);
% Explicitely Enforce Ez=0 BC's on Grid Boundary For Update
Ezs(1,:,p)=EzBC1(:,Ps-p+1); Ezs(:,Ms,p)=EzBC2(:,Ps-p+1).';
Ezs(Ns,:,p)=EzBC3(:,Ps-p+1); Ezs(:,1,p)=EzBC4(:,Ps-p+1).';
end

clf reset;

if vsel==1,
   EzScl=Ezs(:,:,1)/SFac;              % scaling so abs(EzScl) <= DR
   [p1 q1]=find(0 <= EzScl & EzScl < 1); % nonlinear remapping for |EzScl| < 1
   EzScl(p1+(q1-1)*Ns)= 1;             % to zero log plot
   [p1 q1]=find(-1 < EzScl & EzScl < 0);
   EzScl(p1+(q1-1)*Ns)= -1;
   EzLog=sign(Ezs(:,:,1)).*log10(abs(EzScl));
   surf(Xs,Ys,EzLog); shading interp
   title(['\pm Log_{10}\{|E_z(x,y,t=',int2str(Ps),...
        ')|\} Initial Condition for Time-Reversal'],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis equal; axis(v); caxis(C); colorbar; view(0,90); hold on
   if Me ~= 1 | Ne ~= 1, plot3(XBS,YBS,ZBS,'or'); end
   hold off
end

if vsel==2,
   surf(Xs,Ys,Ezs(:,:,1)); shading interp
   title(['E_z(x,y,t=',int2str(Ps),...
         ') Initial Condition for Time-Reversal'],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis equal; axis(v); caxis(C); colorbar; view(0,90); hold on
   if Me ~= 1 | Ne ~= 1, plot3(XBS,YBS,ZBS,'or'); end
   hold off
end

if vsel==3, surfl(Ys,Xs,Ezs(:,:,1));
   title(['E_z(x,y,t=',int2str(Ps),') Initial Condition for Time-Reversal'],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis(v); colormap(copper);
end

Frame=moviein(Ps,gcf);

hcpy=input('Print a Hard Copy ? (Y/N): ','s');
if ~isempty(hcpy), if hcpy == 'Y' | hcpy == 'y', print; end; end
name=input('Enter "Name" to Save as Name.jpg File ? (Hit Enter Key to Skip): ','s');
if ~isempty(name), eval(['print ',name,' -djpeg99']); end

for p=1:Ps;    % Recording animation frames
   if vsel==1,
   EzScl=Ezs(:,:,p)/SFac;                 % scaling so abs(EzScl) <= DR
   [p1 q1]=find(0 <= EzScl & EzScl < 1); % nonlinear remapping for |EzScl| < 1
   EzScl(p1+(q1-1)*Ns)= 1;                % to zero log plot
   [p1 q1]=find(-1 < EzScl & EzScl < 0);
   EzScl(p1+(q1-1)*Ns)= -1;
   EzLog=sign(Ezs(:,:,p)).*log10(abs(EzScl));
   surf(Xs,Ys,EzLog); shading interp
   title(['\pm Log_{10}\{|E_z(x,y,t=',int2str(Ps-p+1),')|\}; Reverse Time Steps: ',...
         int2str(Ps)],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis equal; axis(v); caxis(C); colorbar; view(0,90); hold on
   if Me ~= 1 | Ne ~= 1, plot3(XBS,YBS,ZBS,'or'); end
   hold off
      if ~isempty(name2),
         if p == pj2(nj), name=[name2 '_' int2str(nj)];
            eval(['print ',name,' -djpeg99']); nj=min(nj+1,njmx);
      end; end
   end

   if vsel==2,
   surf(Xs,Ys,Ezs(:,:,p)); shading interp
   title(['E_z(x,y,t=',int2str(Ps-p+1),'); Reverse Time Steps: ',...
         int2str(Ps)],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   axis equal; axis(v); caxis(C); colorbar; view(0,90); hold on
   if Me ~= 1 | Ne ~= 1, plot3(XBS,YBS,ZBS,'or'); end
   hold off
      if ~isempty(name2),
         if p == pj2(nj), name=[name2 '_' int2str(nj)];
            eval(['print ',name,' -djpeg99']); nj=min(nj+1,njmx);
      end; end
   end

   if vsel==3, surfl(Xs,Ys,Ezs(:,:,p))
   title(['E_z(x,y,t=',int2str(Ps-p+1),'); Reverse Time Steps: ',int2str(Ps)],'FontSize',14)
   xlabel('x-axis','FontSize',18); ylabel('y-axis','FontSize',18)
   v(5)=-EzMax; v(6)=EzMax; axis(v); colormap(copper);
      if ~isempty(name2),
         if p == pj1(nj), name=[name2 int2str(nj)];
            eval(['print ',name,' -djpeg99']); nj=min(nj+1,njmx);
      end; end
   end

Frame(p)=getframe(gcf);
end

if vsel==1, title(['Final Reversed Time \pm Log_{10}\{|E_z(x,y,t=0)|\};
Reverse Time Steps: ',...
         int2str(Ps)],'FontSize',14)
else,
   title(['Final Reversed Time E_z(x,y,t=0); Reverse Time Steps: ',...
         int2str(Ps)],'FontSize',14)
end

hcpy=input('Print a Hard Copy ? (Y/N): ','s');
if ~isempty(hcpy), if hcpy == 'Y' | hcpy == 'y', print; end; end
name=input('Enter "Name" to Save as Name.jpg File ? (Hit Enter Key to Skip): ','s');
if ~isempty(name), eval(['print ',name,' -djpeg99']); end

FrameFile=input('Enter File Name to Save Movie Frame Array as *.mat
(Press Enter to Skip): ','s');
if ~isempty(FrameFile), eval(['save ',FrameFile,' M N Frame']); end
```

2. Programs FT_FDTD2D1 And RT_FDTD2D1m For The Second Example

a. FT_FDTD2D1 For The Forward Time Step

```
clear all
opengl neverselect
% Modified to use natural unrotated x(cols) y(rows) array definition
Ns=input('Enter number of x-points on sensor sub-grid boundary: ');
Ms=input('Enter number of y-points in sensor sub-grid boundary: ');
N=2*Ns;M=2*Ms;
Ez=zeros(M,N);
x=1:N; y=1:M; Y=y'*ones(1,N); X=ones(M,1)*x;

P=input('Enter number of time-steps: ');
name1=input('Enter Forward "Name" for  Name_nn.jpg ?
(Enter Key to Skip): ','s');
if ~isempty(name1),
   dp1=input('Enter Time Step Increment Between Stored Frames: ');
   pj1=1:dp1:P;
   nj=1; njmx=length(pj1);
end

q=2; % q=dh/(c*dt)=2 for half-step algorithm
Q1=1/(q*q); Q2=(2-4*Q1); cdt=1/q;


n1=fix((N-Ns)/2)+1; n2=n1+Ns-1;
m1=fix((M-Ms)/2)+1; m2=m1+Ms-1;
xs=n1:n2; ys=m1:m2;        % Sensor grid numbers
Nb=2*(m2-m1+n2-n1);        % Number of inner boundary nodes = 2*(Ns+Ms-2)

% Defining ordered pairs and absolute addresses of sub-grid boundary nodes
in (M,N) array
mys=zeros(Nb,1); nxs=mys;
mys(1:Ns)=m1; mys(Ns+1:Ns+Ms-1)=m1+1:m2;
mys(Ns+Ms:2*Ns+Ms-2)=m2; mys(2*Ns+Ms-1:Nb)=m2-1:-1:m1+1;
nxs(1:Ns)=n1:n2; nxs(Ns+1:Ns+Ms-1)=n2;
nxs(Ns+Ms:2*Ns+Ms-2)=n2-1:-1:n1; nxs(2*Ns+Ms-1:Nb)=n1;
% Absolute array addresses allows no-loop loading
mns=(nxs-1)*M + mys;
% Storage for saving scattered field boundary data
Ezb=zeros(Nb,P);

% User Supplied Function Defines Metallic Target Nodes Where Ez=0
[my nx]=Air2(M,N);
Ntgt=length(nx);
xtgt=x(nx); ytgt=y(my); ztgt=ones(Ntgt,1);
mntgt=(nx-1)*M + my;       % Absolute array addresses allows no-loop loading

% Displaying Target Nodes and Subgrid
```

```
v=[1 N 1 M -1.1 1.1]; cv=[-1.1 1.1];
% sub-grid outline
YBC=[m1 m2 m2 m1 m1]'; XBC=[n1 n1 n2 n2 n1]'; ZBC=[1 1 1 1 1]';
clf reset; surf(X,Y,Ez); shading interp
axis(v); axis equal; caxis(cv); hold on
plot3(xtgt,ytgt,ztgt,'.k'); hold on
plot3(XBC,YBC,ZBC,'--k');
view(0,90); figure(1)
xlabel('x-axis','FontSize',14); ylabel('y-axis','FontSize',14)
title('Target Nodes and Sub-grid - Press a Key to Continue ...','FontSize',14);
hold off

pause

% Constructing Evolution Arrays
d=ones(1,max(N,M));
A=sparse(Q1*(diag(d(1:M-1),1)+diag(d(1:M-1),-1)));
B=sparse(Q1*(diag(d(1:N-1),1)+diag(d(1:N-1),-1)));

% IC's for -x Propagating Gaussian Impulse Plane-Wave (8 Mar 2001)
disp('Unit Peak Gaussian Impulse Plane Wave Propagates in -x Direction');
x0=input('Enter IC Peak Location x0 in Grid Units for the Gaussian Plane Wave: ');
sig=input('Enter Standard Deviation of the Gaussian Plane Wave in Grid Units: ');
E0=1; xc=(x-x0); sig2=2*sig*sig; ct=0:cdt:(P-1)*cdt; xcp=x0-ct; % inc wave center

% Setting IC's at t=-2 and -1 time steps
Ez1=Ez; Ez2=Ez;

% p=-2
Ezex=E0*ones(M,1)*exp(-((xc-2*cdt).^2)/sig2);
Ez1(mntgt)=-Ezex(mntgt);

% p=-1
Ezex=E0*ones(M,1)*exp(-((xc-cdt).^2)/sig2);
Ez2(mntgt)=-Ezex(mntgt);

% Using bi-polar log scaling to retain Ez polarity with selected dynamic range
DR=input('Enter 2-D Plot Dynamic Range, e.g. 100, 1000, etc : ');
SFac=2*E0/DR;           % Assuming max|Ez|=2*E0
Cmax=log10(DR); cv=[-Cmax Cmax];
v=[1 N 1 M -Cmax Cmax]; ZBC=Cmax*[1 1 1 1 1]';

% Computing Scattered Field and Displaying Total Field

pshow=(5:5:P);  % every 5 time-step indices to display progress

for p=1:P;   % Time-Stepping

% Compute Exact Plane Wave
Ezex=E0*ones(M,1)*exp(-((xc+ct(p)).^2)/sig2);
```

```
% Equation of Evolution
Ez = A*Ez2 + Ez2*B + Q2*(Ez2-Ez1);

% Enforce PEC BC's on Target Nodes
Ez(mntgt)=-Ezex(mntgt);

% Enforce PEC BC's on Grid Boundary
Ez(:,1)=0;  % x=1 BC
Ez(1,:)=0;  % y=1 BC
Ez(:,N)=0;  % x=M BC
Ez(M,:)=0;  % y=N BC
% Time-shift arrays
Ez1=Ez2; Ez2=Ez;

% Saving scattered field at boundary
Ezb(:,p)=Ez(mns);

% Displaying +/- Log10(Total Field)
EzScl=(Ez+Ezex)/SFac;                    % scaling so abs(EzScl) <= DR
[p1 q1]=find(0 <= EzScl & EzScl < 1);    % nonlinear remapping for |EzScl| < 1
EzScl(p1+(q1-1)*M)= 1;                   % to zero log plot
[p1 q1]=find(-1 < EzScl & EzScl < 0);
EzScl(p1+(q1-1)*M)= -1;
EzLog=sign(Ez+Ezex).*log10(abs(EzScl));
surf(X,Y,EzLog); shading interp
axis(v); axis equal; caxis(cv); colorbar; hold on
% Highlight Target Locations and Sub-Grid Boundary
plot3(xtgt,ytgt,ztgt,'.k'); hold on
plot3(XBC,YBC,ZBC,'--k');
view(0,90);
xlabel('x-axis','FontSize',14); ylabel('y-axis','FontSize',14)
title(['\pm Log_{10}\{|E_z^{tot}(x,y,t=',int2str(p),')|\}'],'FontSize',14);
hold off
figure(1)
if ~isempty(name1),
      if p == pj1(nj), name=[name1 '_' int2str(p)];
        eval(['print ',name,' -djpeg99']); nj=min(nj+1,njmx);
    end; end
pause(.01)
end
name=input('Enter "Name" to Save as Name.jpg File ? (Hit Enter Key to Skip): ','s');
if ~isempty(name), eval(['print ',name,' -djpeg99']); end

name2=input('Enter "Name" for Name.Mat to Save [N M P Ns Ms x0 sig nx my Ezb]
(Enter Key to Skip): ','s');
if ~isempty(name2), eval(['save ',name2,' N M P Ns Ms x0 sig nx my Ezb']); end
```

b. RT_FDTD2D1m For The Reversed Time Step

```
clear all

dir *.mat
name=input('Enter "Name" for Name.Mat to Retrieve Boundary and Target Data: ','s');
eval(['load ',name,' N M P Ns Ms x0 sig nx my Ezb']);

% N=number of x-points in full grid
% M=number of y-points in full grid
% P=number of time-steps
% Ns=number of x-points on sensor sub-grid boundary
% Ms=number of y-points in sensor sub-grid boundary
% x0=Gaussian Plane Wave Initial Peak Location in Grid Units
% sig=Gaussian Plane Wave Standard Deviation in Grid Units
% nx(1,Ntgt)=x-target nodes in (M,N) full grid
% my(1,Ntgt)=y-target nodes
% Ezb(Nb,P)=scattered field at subgrid boundary

n1=fix((N-Ns)/2)+1; n2=n1+Ns-1;
m1=fix((M-Ms)/2)+1; m2=m1+Ms-1;

xs=n1:n2; ys=m1:m2;        % Inner grid coordinates in (M,N) grid
Ys=ys'*ones(1,Ns); Xs=ones(Ms,1)*xs;

Nb=2*(m2-m1+n2-n1);        % Number of inner boundary nodes = 2*(Ns+Ms-2)

% Computing Boundary Node Indices in Inner Sub-Grid System
myb=zeros(Nb,1); nxb=myb;
myb(1:Ns)=1; myb(Ns+1:Ns+Ms-1)=2:Ms;
myb(Ns+Ms:2*Ns+Ms-2)=Ms; myb(2*Ns+Ms-1:Nb)=Ms-1:-1:2;
nxb(1:Ns)=1:Ns; nxb(Ns+1:Ns+Ms-1)=Ns;
nxb(Ns+Ms:2*Ns+Ms-2)=Ns-1:-1:1; nxb(2*Ns+Ms-1:Nb)=1;
mnb=(nxb-1)*Ms + myb;      % Absolute array addresses allows no-loop loading

% Metallic Target Nodes Where Ez=-Ez^inc
Ntgt=length(nx); ztgt=ones(Ntgt,1); nn=(1:Ntgt)'; nn1=(1:Ntgt)';
nx1=nx;my1=my;
disp('Exact Target Nodes (node#, nx, my) in Full Grid:')
disp([nn nx my]);
yn=input('Change Assumed Target Nodes ? (Y/N): ','s');
if yn == 'Y' | yn == 'y',
   while 1,
      n=input('Enter Node# to Change (0 to End): ');
      if n < 1 | n > Ntgt, break; end
      nxmy=input('Enter New [nx my] as Vector: ');
      nx1(n)=nxmy(1); my1(n)=nxmy(2);
      disp('Revised Target Nodes (node#, nx, my) in Full Grid:')
      disp([nn1 nx1 my1]);
   end
```

```
end
% Assumed Target Node Numbers in Inner Grid Coordinates
mys=my-m1+1; nxs=nx-n1+1;

% Absolute Array Addresses for Assumed Target Nodes
mnstgt=(nxs-1)*Ms + mys;

 % Assumed Target Node Numbers in Inner Grid Coordinates
mys1=my1-m1+1; nxs1=nx1-n1+1;

% Absolute Array Addresses for Assumed Target Nodes
 mnstgt1=(nxs1-1)*Ms + mys1;

q=2; % Assuming q=dh/(c*dt)=2 for half-step algorithm
Q1=1/(q*q); Q2=(2-4*Q1); cdt=1/q; ct=cdt*(0:(P-1));

E0=1; xcp=x0-ct; sig2=2*sig*sig;        % Incident field peak location


% Constructing Sub-Grid Evolution Arrays
ds=ones(1,max(Ns,Ms));
As=Q1*(diag(ds(1:Ms-1),1)+diag(ds(1:Ms-1),-1));
Bs=Q1*(diag(ds(1:Ns-1),1)+diag(ds(1:Ns-1),-1));

Ps=input(['Enter time-step <= ',int2str(P),' to initiate time-reversed FDTD solution: ']);

% Using bi-polar log scaling to retain Ez polarity with selected dynamic range
DR=input('Enter 2-D Plot Dynamic Range, e.g. 100, 1000, etc : ');
SFac=2*E0/DR;           % Assuming max|Ez|=2*E0
Cmax=log10(DR); cv=[-Cmax Cmax];
v=[1 N 1 M -Cmax Cmax]; yi=[1 N]'; zi=[Cmax Cmax]';
name1=input('Enter "Name" for  Name_nn.jpg ? (Enter Key to Skip): ','s');
if ~isempty(name1),
   dp1=input('Enter Time Step Increment Between Stored Frames: ');
   pj1=Ps:-dp1:1;
   nj=1; njmx=length(pj1);
end

% Reserving space for evolution field arrays
Ez=zeros(Ms,Ns); Ez1=Ez; Ez2=Ez;

Energy=zeros(Ps,1);  % Accumulated Energy
% Reverse Time Evolution
for p=Ps:-1:1;
xi=[xcp(p) xcp(p)]'; % For Incident Field Dashed Line
% Initializing BC's
   if p==Ps % p=Ps
      Ezi=E0*ones(Ms,1)*exp(-((xs-xcp(Ps)).^2)/sig2);   % Incident field
      if yn == 'Y' | yn == 'y',
         Ez1(mnstgt1)=-Ezi(mnstgt1);  % Target data
     else
```

```
         Ez1(mnstgt)=-Ezi(mnstgt);  % Target data
     end
      Energy(1)=sum(sum(Ez1.*Ez1));
   elseif p==Ps-1% p=Ps-1
      Ez2(mnb)=Ezb(:,Ps-1);      % Boundary data
      Ezi=E0*ones(Ms,1)*exp(-((xs-xcp(Ps-1)).^2)/sig2); % Incident field
      if yn == 'Y' | yn == 'y',
         Ez2(mnstgt1)=-Ezi(mnstgt1);  % Target data
     else
      Ez2(mnstgt)=-Ezi(mnstgt); % Target data
     end
      Energy(2)=Energy(1)+sum(sum(Ez2.*Ez2));
 clf reset
   else
      Ezi=E0*ones(Ms,1)*exp(-((xs-xcp(p)).^2)/sig2);   % Incident Field

      Ez = As*Ez2 + Ez2*Bs + Q2*(Ez2-Ez1);             % FDTD Evolution
      if yn== 'Y' | yn== 'y',
         Ez(mnstgt1)=-Ezi(mnstgt1);                      % Target Node PEC BC's
      else
         Ez(mnstgt)=-Ezi(mnstgt);
      end
          Ez(mnb)=Ezb(:,p);                               % Boundary Data Update

      Ez1=Ez2; Ez2=Ez;                                  % Time-shift arrays

      Energy(Ps-p+1)=Energy(Ps-p)+sum(sum(Ez.*Ez));
   end
% Displaying +/- Log10(Total Field)
EzScl=Ez/SFac;   % Scattered Field (scaling so abs(EzScl) <= DR)
% EzScl=(Ez+Ezi)/SFac;   % Total Field
[p1 q1]=find(0 <= EzScl & EzScl < 1);    % nonlinear remapping for |EzScl| < 1
EzScl(p1+(q1-1)*Ms)= 1;                  % to zero log plot
[p1 q1]=find(-1 < EzScl & EzScl < 0);
EzScl(p1+(q1-1)*Ms)= -1;
EzLog=sign(Ez).*log10(abs(EzScl));      % Scattered Field
% EzLog=sign(Ez+Ezi).*log10(abs(EzScl));  % Total Field
surf(Xs,Ys,EzLog); shading interp
axis equal; axis(v); caxis(cv); colorbar; view(0,90); hold on
% Highlight Target Locations and Sub-Grid Boundary
if yn == 'Y' | yn== 'y';
   plot3(nx1,my1,ztgt,'.r'); hold on
end
plot3(nx,my,ztgt,'.k'); hold on
plot3(xi,yi,zi,'--r');
% plot3(XBC,YBC,ZBC,'--k');
xlabel('x-axis','FontSize',14); ylabel('y-axis','FontSize',14)
title(['\pm Log_{10}\{|E_z^{scat}(x,y,t=',int2str(p),')|\}'],'FontSize',14);

hold off
if ~isempty(name1),
      if p == pj1(nj), name=[name1 '_' int2str(nj)];
        eval(['print ',name,' -djpeg99']); nj=min(nj+1,njmx);
    end; end

  %figure(1)
pause(.01)
end

name=input('Enter "Name" to Save as Name.jpg File ? (Hit Enter Key to Skip): ','s');
if ~isempty(name), eval(['print ',name,' -djpeg99']); end

tp=(1:Ps)';
figure(2); plot(tp,Energy)
xlabel('Reversed Time Steps','FontSize',14)
ylabel('Energy','FontSize',14)
title(['Final Sub-Grid Energy = ',num2str(Energy(Ps))],'FontSize',14)
```

3. Target Functions

a. Air2: Two Aircraft-like Targets

```
function [ny, mx]=Air2(N,M)
% [ny mx] are column arrays of target nodes for 2 aircraft from RTWE_2D5.m

% Define Basic Aircraft Shaped Grid Locations Centered at y=x=0
Tgt=zeros(29,2);   % Basic Target Node Indices (Y,X) ordered
Tgt(1:11,2)=(-5:5)';
Tgt(12,2)=-5; Tgt(12,1)=2;
Tgt(13,2)=-4; Tgt(13,1)=1;
Tgt(14,2)=-1; Tgt(14,1)=4;
Tgt(15,2)=-1; Tgt(15,1)=3;
Tgt(16,2)= 0; Tgt(16,1)=3;
Tgt(17,2)= 0; Tgt(17,1)=2;
Tgt(18,2)= 1; Tgt(18,1)=2;
Tgt(19,2)= 1; Tgt(19,1)=1;
Tgt(20,2)= 2; Tgt(20,1)=1;
Tgt(21:29,2)=Tgt(12:20,2);
Tgt(21:29,1)=-Tgt(12:20,1);

mc=fix((M-1)/2)+1; nc=fix((N-1)/2)+1;  % Approx Grid Center

% Defining Nodes for Two Offset Targets
ny=zeros(58,1); mx=ny;
ny(1:29)=Tgt(:,1) + nc + 7;   % y offset for tgt #1
mx(1:29)=Tgt(:,2) + mc + 2;   % x offset for tgt #1
ny(30:58)=Tgt(:,1) + nc - 7;  % y offset for tgt #2
mx(30:58)=Tgt(:,2) + mc - 2;  % x offset for tgt #2
```

b. Pt2: Two Point-Like Targets

```
function [my, nx]=Pt2(M,N)
% [my nx] are column arrays of simple 2-point target nodes
% 27 June 01 Mod from Point2.m reversing M and N roles with x and y

Tgt=zeros(2,2);   % Basic Target Node Indices (Y,X) ordered
Tgt(1:2,2)=[-5 5]';

mc=fix((M-1)/2)+1; nc=fix((N-1)/2)+1;  % Approx Grid Center

% Defining Nodes
my=zeros(2,1); nx=my;
my(1:2)=Tgt(:,1) + mc;
nx(1:2)=Tgt(:,2) + nc;
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

1. N. J. Willis, *Bistatic Radar*, Technology Service Corporation, 1995.

2. David Colton and Rainer Kress, *Inverse Acoustic and Electromagnetic Scattering Theory,* 2d ed., Applied Mathematical Sciences 93, Springer-Verlag, 1998.

3. David Colton, Klaus Giebermann and Peter Monk, "The linear sampling method for three-dimensional inverse scattering problems", *Australian & New Zealand Industrial and Applied Mathematics Journal*, vol. 42 Part C, pp. C434-C460, 7 August 2000.

4. Meir Morag, *Radar Target Imaging By Time-Domain Inverse Scattering*, Master's Thesis, Naval Postgraduate School, Monterey, California, March 1981.

5. Mathias Fink, " Time Reversed Acoustics", *Physics Today*, Vol. 50, No. 3, pp. 34-40, March 1997.

6. Francois Wu, Jean-Louis Thomas, and Mathias Fink," Time-Reversal of Ultrasonic Fields-Part .: Experimental Results," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, Vol. 39, No. 5, pp. 567-578, September 1992.

7. Didier Cassereau and Mathias Fink, " Time-Reversal of Ultrasonic Fields- Part .: Theory of the Closed Time-Reversal Cavity," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, Vol. 39, No. 5, pp. 579-592, September 1992.

8. Carsten Draeger, Didier Cassereau, and Mathias Fink, “ Theory of the time-reversal process in solids,’ *Journal of Acoustical Society of America*, 102 (3), pp. 1289-1295, September 1997.

9. Mickaël Tanter, Jean-Louis Thomas, and Mathias Fink, “ Time reversal and the inverse filter,” *Journal of Acoustical Society of America*, 108 (1), pp. 223-234, July 2000.

10. Mathias Fink, “ Time Reversal of Ultrasonic Fields – Part .: Basic Principles,” *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, Vol. 39, No. 5, pp. 555-566, September 1992.

11. Eugene Hecht, *Optics*, 3rd ed., pp. 135-137, Addison Wesley Longman, Inc., 1998.

12. Sadasiva M. Rao, and others, *Time Domain Electromagnetics*, pp. 151-235, Academic Press, 1999.

13. Matthew N. O. Sadiku, *Numerical Techniques in Electromagnetics*, 2nd ed. pp. 121-234,CRC Press LLC, 2000.

14. Michael A. Morgan, The Finite-Difference Time-Domain Solution Method, Unpublished Class Notes for EC4600 Advanced EM Theory, 1997.

15. Michael A. Morgan, Time-Reversed EM in Target Imaging, ONR Program Review Presentation Slides, 12 Sep 2000.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center..2
8725 John J. Kingman Road, Suite 0944
Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library...2
Naval Postgraduate School
411 Dyer Road
Monterey, CA 93943-5101

3. Group Chairman………………………………………………………….. 1
Code CC
Information Warfare Academic Group
Naval Postgraduate School
Monterey, California 93943
dboger@nps.navy.mil

4. Dr. Ronald Radlinski 2
Code 334
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5660

5. Professor Michael A. Morgan …………………………………………………….1
Code EC/Mw
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000
mmorgan@nps.navy.mil

6. Professor David C. Jenn…………………………………………………...1
Code EC/Jn
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5000
jenn@nps.navy.mil

7. Lt. Yosuke Inaba ……………………………………………………………….1
18-58 Momijigaoka, Atami
Shizuoka, Japan, 413-0027