Progress In Electromagnetics Research B, Vol. 68, 1–16, 2016

Design and Simulation of Arbitrarily-Shaped Transformation Optic Devices Using a Simple Finite-Difference Method

Eric A. Berry, Jesus J. Gutierrez, and Raymond C. Rumpf*

Abstract—A fast and simple design methodology for transformation optics (TO) is described for devices having completely arbitrary geometries. An intuitive approach to the design of arbitrary devices is presented which enables possibilities not available through analytical coordinate transformations. Laplace's equation is solved using the finite-difference method to generate the arbitrary spatial transforms. Simple techniques are presented for enforcing boundary conditions and for isolating the solution of Laplace's equation to just the device itself. It is then described how to calculate the permittivity and permeability functions via TO from the numerical spatial transforms. Last, a modification is made to the standard anisotropic finite-difference frequency-domain (AFDFD) method for much faster and more efficient simulations. Several examples are given at the end to benchmark and to demonstrate the versatility of the approach. This work provides the basis for a complete set of tools to design and simulate TO devices of any shape and size.

1. INTRODUCTION

Transformation optics (TO) is a design methodology that allows electromagnetic fields to be controlled through spatial transforms [1]. The most well-known example is the cylindrical electromagnetic cloak described in Ref. [2]. Their device used a closed-form expression for the spatial transform, but closed-form expressions are highly limited in the range of geometries they can describe. In order to design an arbitrarily shaped device using TO, a fully numerical method is necessary to perform the coordinate transformation and to calculate the resulting material properties for the device. This paper presents a fully numerical technique based on the finite-difference method to generate and simulate arbitrary TO devices. The arbitrary spatial transforms are generated numerically by solving Laplace's equation [3, 4]. The devices are confirmed through simulation using an improved anisotropic finite-difference frequency-domain (AFDFD) method [5] that more efficiently simulates waves in arbitrary anisotropic media. Several examples are given to demonstrate the simplicity and versatility of our approach. This is the first known description of a complete numerical tool chain for TO that is based on the finite-difference method.

Historically, methods such as quasi-conformal mapping [6] and the solution of various partial differential equations such as Laplace's equation [3, 4], Poisson's equation [7], and the Helmholtz equation [8] have been used to generate the TO material parameters numerically. The approach presented in Ref. [6] can create arbitrary dielectric waveguides using TO, but does not allow for devices such as cloaks and lenses to be designed. The techniques described in Refs. [3, 4, 7, 8] use a commercial finite element software such as COMSOL to simulate a wide assortment of devices using TO, but it is difficult or impossible to use the calculated materials in a different electromagnetic solver. In contrast, the approach discussed in this article is designed specifically to be used with custom electromagnetic methods such as the AFDFD method described in Ref. [5] and will be incorporated into a tool chain which will generate devices using the geometry generation technique discussed in Ref. [9] to synthesize spatially variant lattices of metamaterial unit cells.

Received 20 January 2016, Accepted 20 April 2016, Scheduled 10 May 2016
* Corresponding author: Raymond C. Rumpf (rcrumpf@utep.edu).
The authors are with the EM Lab, W. M. Keck Center for 3D Innovation, University of Texas at El Paso, USA.

2. NUMERICAL GRID GENERATION USING LAPLACE'S EQUATION

The following subsections detail how to generate grids for spatial transforms numerically by solving Laplace's equation using finite-differences. The discussion begins by describing the generic solution to Laplace's equation using finite-differences. Techniques are presented to enforce the boundary conditions and to isolate the solution to specific portions of a grid. The discussion ends by showing how to use this tool to generate the grids associated with the spatial transforms used in TO, but for arbitrarily shaped objects.

2.1. Finite-Difference Approximation of Laplace's Equation

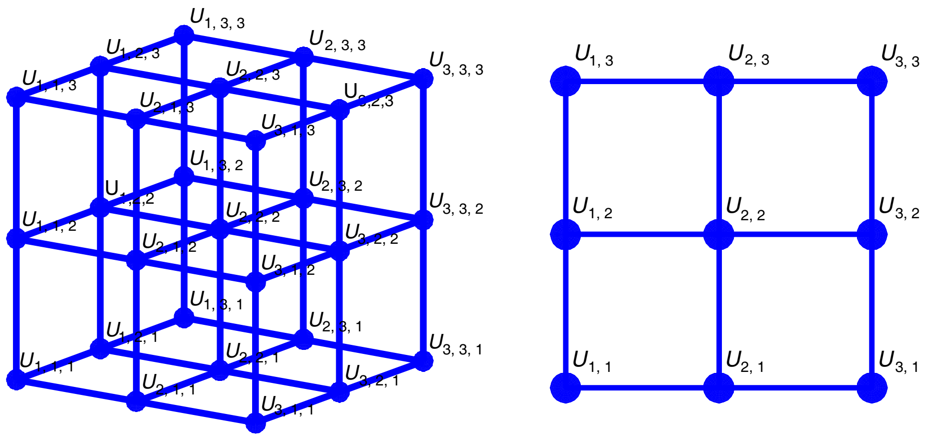
Laplace's equation for the generic function $U(x, y, z)$ is given by

$$\nabla^2 U(x, y, z) = 0. \tag{1}$$

It can be expanded into Cartesian coordinates according to

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}\right) U\left(x, y, z\right) = 0 \tag{2}$$

By setting the second-order derivatives to zero, Laplace's equation essentially dictates that the function $U(x, y, z)$ must vary linearly as a function of position. To numerically solve Eq. (2) using the finite-difference method, the function $U(x, y, z)$ is stored at discrete points on a grid as illustrated in Figure 1. This figure shows a small 3D grid and an analogous 2D grid which stores the discretized function $U_{i,j}$.



(a) (b)

Figure 1. (a) Three-dimensional grid for $U(x, y, z)$. (b) Two-dimensional grid for $U(x, y)$.

This allows the second-order derivatives in Eq. (2) to be approximated with central finite-differences as follows.

$$\frac{U_{i+1,j,k} - 2U_{i,j,k} + U_{i-1,j,k}}{\Delta x^2} + \frac{U_{i,j+1,k} - 2U_{i,j,k} + U_{i,j-1,k}}{\Delta y^2} + \frac{U_{i,j,k+1} - 2U_{i,j,k} + U_{i,j,k-1}}{\Delta z^2} \approx 0 \tag{3}$$

Equation (3) can be interpreted as enforcing Laplace's equation on the discrete function $U_{i,j,k}$ at point $(i,\ j,\ k)$. To enforce Laplace's equations across the entire grid, Eq. (3) is written once for every point on the grid. This large set of equations can be written in matrix form as

$$\mathbf{L}\mathbf{u} = \mathbf{0} \tag{4}$$

In Eq. (4), the term $\mathbf{u}$ contains all of the unknown values of $U_{i,j,k}$ throughout the entire grid reshaped into a one-dimensional column vector. The square matrix $\mathbf{L}$ operates on $\mathbf{u}$ to calculate its scalar Laplacian ∇^2. It is a banded matrix where the rows of $\mathbf{L}$ are populated using Eq. (3), but an easier and more powerful approach for constructing $\mathbf{L}$ and incorporating boundary conditions is described in the next subsection. The term $\mathbf{0}$ is a column vector the same size as $\mathbf{u}$ but contains all zeros.

2.2. Matrix Operator Approach

Given a discrete function $U_{i,j,k}$ stored in a column vector $\mathbf{u}$, it is always possible to construct a square matrix $\mathbf{L}$ that performs any linear operation on $\mathbf{u}$ including derivatives [10, 11], discrete Fourier transforms [12, 13], convolutions [13], and more. Linearity makes it possible to breakdown complex linear operations into combinations of simpler linear operations. From Eq. (2), it is possible to express the Laplacian operation $\mathbf{L}$ as the sum of three derivative operations $\mathbf{D}_x^2$, $\mathbf{D}_y^2$, and $\mathbf{D}_z^2$ to get Eq. (5).

$$\mathbf{L} = \mathbf{D}_x^2 + \mathbf{D}_y^2 + \mathbf{D}_z^2 \tag{5}$$

The terms $\mathbf{D}_x^2$, $\mathbf{D}_y^2$, and $\mathbf{D}_z^2$ are square banded matrices the same size as $\mathbf{L}$ that calculate second-order derivatives of the function $\mathbf{u}$ across the entire grid. These are simpler to construct than $\mathbf{L}$ and easier to verify if they are correct, so it is often advantageous to construct these first and then calculate $\mathbf{L}$ from them using Eq. (5).

As an example, the matrix Laplacian $\mathbf{L}$ will be constructed for the two-dimensional 3×3 point grid shown in Figure 1. First, suppose we wish to calculate the second-order derivative of $U_{i,j}$ in the x direction. The finite-difference approximation for this can be written as

$$\frac{\partial^2 U_{i,j}}{\partial x^2} \approx \frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{\Delta x^2}. \tag{6}$$

This equation is written once for every point on the grid and the large set of equations is collected into a single matrix equation. The matrix equation is expressed as a square matrix operating on the column vector $\mathbf{u}$. The square matrix derived from this process is the matrix derivative operator $\mathbf{D}_x^2$ as expressed in Eq. (7).

$$\mathbf{D}_x^2\mathbf{u} = \frac{1}{\Delta x^2}\begin{bmatrix} U_{2,1} - 2U_{1,1} \\ U_{3,1} - 2U_{2,1} + U_{1,1} \\ -2U_{3,1} + U_{2,1} \\ U_{2,2} - 2U_{1,2} \\ U_{3,2} - 2U_{2,2} + U_{1,2} \\ -2U_{3,2} + U_{2,2} \\ U_{2,3} - 2U_{1,3} \\ U_{3,3} - 2U_{2,3} + U_{1,3} \\ -2U_{3,3} + U_{2,3} \end{bmatrix} = \frac{1}{\Delta x^2}\begin{bmatrix} -2 & 1 & & & & & & & \\ 1 & -2 & 1 & & & & & & \\ & 1 & -2 & 0 & & & & & \\ & & 0 & -2 & 1 & & & & \\ & & & 1 & -2 & 1 & & & \\ & & & & 1 & -2 & 0 & & \\ & & & & & 0 & -2 & 1 & \\ & & & & & & 1 & -2 & 1 \\ & & & & & & & 1 & -2 \end{bmatrix}\begin{bmatrix} U_{1,1} \\ U_{2,1} \\ U_{3,1} \\ U_{1,2} \\ U_{2,2} \\ U_{3,2} \\ U_{1,3} \\ U_{2,3} \\ U_{3,3} \end{bmatrix}. \tag{7}$$

A problem arises when writing Eq. (6) at both the left and right sides of the grid where values of $U_{i,j}$ are needed from outside of the grid. The manner in which this is handled is called a numerical boundary condition [14, 15]. To arrive at Eq. (7), it was assumed that all values of $U_{i,j}$ were zero outside of the grid. This is called the Dirichlet boundary condition [16, 17]. Second, the finite-difference approximation for the second-order derivative of $U_{i,j}$ in the y direction can be written as

$$\frac{\partial^2 U_{i,j}}{\partial y^2} \approx \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{\Delta y^2}. \tag{8}$$

This equation is also written once for every point on the grid and the large set of equations is collected into a single matrix equation expressed as a square matrix operating on the column vector $\mathbf{u}$. The square matrix derived from this process is the matrix derivative operator $\mathbf{D}_y^2$ as expressed in Eq. (9).

$$\mathbf{D}_y^2\mathbf{u} = \frac{1}{\Delta y^2}\begin{bmatrix} U_{1,2} - 2U_{1,1} \\ U_{2,2} - 2U_{2,1} \\ U_{3,2} - 2U_{3,1} \\ U_{1,3} - 2U_{1,2} + U_{1,1} \\ U_{2,3} - 2U_{2,2} + U_{2,1} \\ U_{3,3} - 2U_{3,2} + U_{3,1} \\ -2U_{1,3} + U_{1,2} \\ -2U_{2,3} + U_{2,2} \\ -2U_{3,3} + U_{3,2} \end{bmatrix} = \frac{1}{\Delta y^2}\begin{bmatrix} -2 & & & 1 & & & & & \\ & -2 & & & 1 & & & & \\ & & -2 & & & 1 & & & \\ 1 & & & -2 & & & 1 & & \\ & 1 & & & -2 & & & 1 & \\ & & 1 & & & -2 & & & 1 \\ & & & 1 & & & -2 & & \\ & & & & 1 & & & -2 & \\ & & & & & 1 & & & -2 \end{bmatrix}\begin{bmatrix} U_{1,1} \\ U_{2,1} \\ U_{3,1} \\ U_{1,2} \\ U_{2,2} \\ U_{3,2} \\ U_{1,3} \\ U_{2,3} \\ U_{3,3} \end{bmatrix}. \tag{9}$$

Dirichlet boundary conditions were used again here at the top and bottom boundaries of the grid. Third, the Laplacian matrix $\mathbf{L}$ for two dimensions is calculated by summing the above two derivative matrices.

$$\mathbf{L} = \mathbf{D}_x^2 + \mathbf{D}_y^2 \tag{10}$$

2.3. Enforcing Physical Boundary Conditions

At this point, our matrix equation has the form $\mathbf{Lu} = \mathbf{0}$. If this was solved for $\mathbf{u}$ only a trivial solution would be obtained. The matrix equation can be written in a solvable form $\mathbf{Lu} = \mathbf{b}$ when the physical boundary conditions are incorporated. To do this, at least some values of $U_{i,j}$ on the grid must be known. Each known point is incorporated into the matrix equation by: (1) replacing the entire row in the matrix equation corresponding to that point with all 0's, (2) placing a 1 in the diagonal position in $\mathbf{L}$, and (3) placing the known value of $U_{i,j}$ into that same row of $\mathbf{b}$.

All of this can be accomplished in a more straightforward manner as follows. First, a force matrix $\mathbf{F}$ is constructed. This is a diagonal matrix with 1's placed in the positions along diagonal that correspond to points with known vales of $U_{i,j}$. Second, a force function $U_{F,i,j}$ is constructed that contains all of the known values placed at the correct points on the grid. The force function is reshaped into a one-dimensional column vector $\mathbf{u}_F$ called the force vector. It may seem tedious or more complicated to construct these functions and matrices, but they need to be constructed anyway to describe arbitrary devices. Given the force matrix $\mathbf{F}$ and the force vector $\mathbf{u}_F$, the matrix equation incorporating these physical boundary conditions is

$$\mathbf{L}'\mathbf{u} = \mathbf{b} \tag{11}$$

where

$$\mathbf{L}' = \mathbf{F} + (\mathbf{I} - \mathbf{F})\,\mathbf{L} \tag{12}$$
$$\mathbf{b} = \mathbf{F}\mathbf{u}_F \tag{13}$$

The term $\mathbf{I}$ is the identity matrix. A solution to Eq. (11) is calculated as $\mathbf{u} = \mathbf{L}'^{-1}\mathbf{b}$. An example for a 13×13 point grid is provided in Figure 2. At left is the empty grid highlighting the known points, or physical boundary conditions. At right is the function $U_{i,j}$ after solving Laplace's equations with these boundary conditions. From this, we see that Laplace's equation provides a way to fill in all the other values of a function so that they vary linearly in the space between the physical boundary conditions.

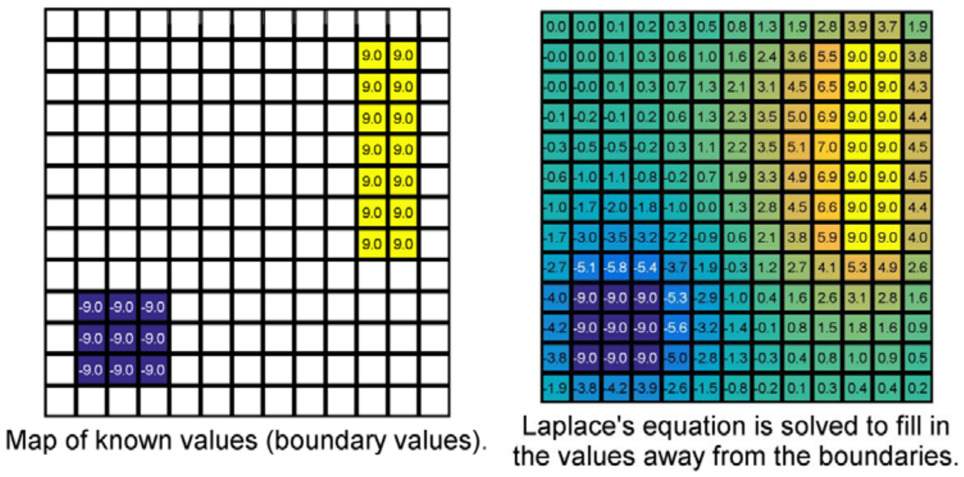

Map of known values (boundary values).
Laplace's equation is solved to fill in the values away from the boundaries.

Figure 2. Two-dimensional grid to illustrate solution of Laplace's equation.

2.4. Enclosed Problems

It sometimes occurs that the physical boundary conditions completely enclose a portion of the grid and the solution to Laplace's equation is only needed within this enclosed region. It is most numerically

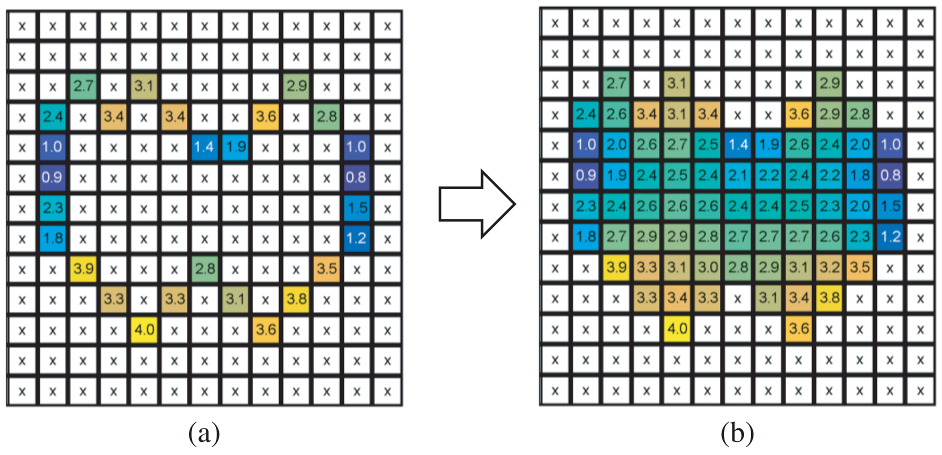


Figure 3. (a) Physical boundary conditions enclose a portion of the grid. (b) Solution to Laplace's equations obtained only in the enclosed portion of the grid.

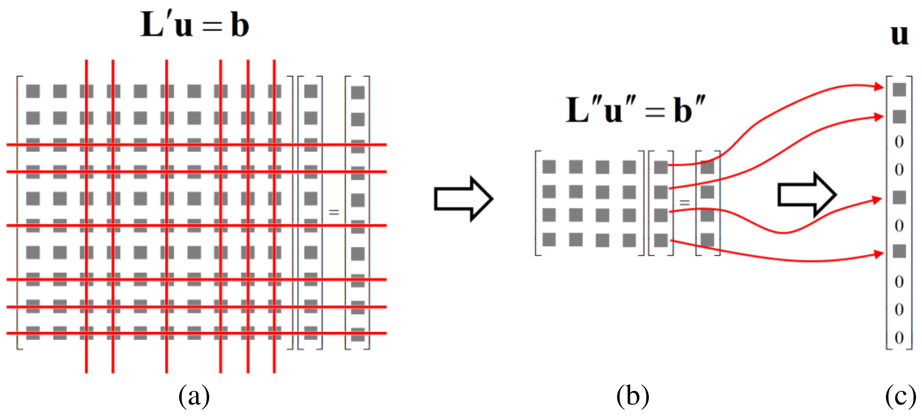

$\mathbf{L}'\mathbf{u}=\mathbf{b}$
$\mathbf{L}''\mathbf{u}''=\mathbf{b}''$
$\mathbf{u}$
(a)
(b)
(c)

Figure 4. (a) Matrix equation $\mathbf{L}'\mathbf{u} = \mathbf{b}$ for entire grid. (b) Reduced Laplace's equation. (c) Repopulated full size column vector $\mathbf{u}$.

efficient to only solve Laplace's equation within the enclosed region and not outside. This situation arises when designing devices by TO since the transform is typically contained within the volume of the device. A simple case of this concept is illustrated in Figure 3 where an arbitrary region of the grid is enclosed by the physical boundary conditions. The solution to Laplace's equation fills in the values inside the enclosed region, but no solution is obtained outside of the enclosed region.

The procedure to reduce Laplace's equation is simple. All of the points on the grid are identified where a solution to Laplace's equation is needed. The rows and columns corresponding to all other points are dropped from the matrix equation. The reduced matrix equation can be written as

$$\mathbf{L}''\mathbf{u}'' = \mathbf{b}''. \tag{14}$$

The concept of reducing the matrix equation is illustrated in Figure 4. To the remaining equations, the eliminated points have values of zero. However, this has no numerical effect because the unknown values are completely enclosed by the known points (i.e., physical boundary conditions) and not the eliminated points. After obtaining a solution to $\mathbf{u}'' = \mathbf{L}''^{-1}\mathbf{b}''$, the values from $\mathbf{u}''$ must be inserted into the correct positions in the full-size column vector $\mathbf{u}$. Values outside of the enclosed region can be set to anything because they are not of interest. Values of zero are a convenient choice.

2.5. Numerical Grid Generation

Numerical grid generation is used in fields such as oceanography [18], computational fluid dynamics [19–21], and conformal mapping in electromagnetics [6]. The approach allows complex geometries to be handled by mapping an arbitrary grid to a Cartesian grid. The most common approach is based on solving Laplace's equation [20, 22] for the transformed coordinates because it generates coordinates that vary linearly. Given the boundary conditions for an arbitrary coordinate transformation from $\{x, y, z\}$ to $\{x', y', z'\}$, Laplace's equation can be used to solve for the transformed coordinates using the following equations.

$$\nabla^2 x' = \left(\frac{\partial^2}{\partial^2 x} + \frac{\partial^2}{\partial^2 y} + \frac{\partial^2}{\partial^2 z}\right) x' = 0 \tag{15}$$

$$\nabla^2 y' = \left(\frac{\partial^2}{\partial^2 x} + \frac{\partial^2}{\partial^2 y} + \frac{\partial^2}{\partial^2 z}\right) y' = 0 \tag{16}$$

$$\nabla^2 z' = \left(\frac{\partial^2}{\partial^2 x} + \frac{\partial^2}{\partial^2 y} + \frac{\partial^2}{\partial^2 z}\right) z' = 0 \tag{17}$$

Occasionally, the backward coordinate transformation is advantageous. Examples of such cases will be discussed later. In the case where the backward coordinate transformation is favored, Laplace's equation can be solved for the backward transformation according to

$$\nabla^2 x = \left(\frac{\partial^2}{\partial^2 x'} + \frac{\partial^2}{\partial^2 y'} + \frac{\partial^2}{\partial^2 z'}\right) x = 0 \tag{18}$$

$$\nabla^2 y = \left(\frac{\partial^2}{\partial^2 x'} + \frac{\partial^2}{\partial^2 y'} + \frac{\partial^2}{\partial^2 z'}\right) y = 0 \tag{19}$$

$$\nabla^2 z = \left(\frac{\partial^2}{\partial^2 x'} + \frac{\partial^2}{\partial^2 y'} + \frac{\partial^2}{\partial^2 z'}\right) z = 0. \tag{20}$$

The solutions of Eqs. (15)–(20) yields only the trivial solution until the boundary conditions are enforced. Consider the two arbitrary boundaries shown in Figure 5. The boundary conditions along the outer boundary Γ'_1 and the inner boundary Γ'_2 are given by

$$U\left(x', y'\right)|_{\Gamma'_1(x,y)} = U\left(x, y\right)|_{\Gamma_1(x,y)} \tag{21}$$

$$U\left(x', y'\right)|_{\Gamma'_2(x,y)} = U\left(x, y\right)|_{\Gamma_2(x,y)} \tag{22}$$

The force matrix and force vector are used to incorporate the boundary conditions into Eqs. (21) and (22). The force matrix $\mathbf{F}$ contains 1's along the diagonal in the positions that correspond to the boundaries of the device which have known coordinate values. The force vector $\mathbf{u}_F$ is a column vector

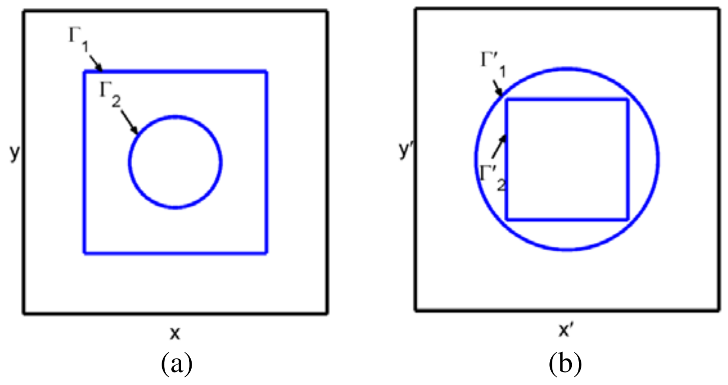


Figure 5. Boundary conditions for the coordinate transformation in Eqs. (21) and (22). (a) Original coordinate system. (b) Transformed coordinate system.

containing the values of $U(x,y)$ at the boundaries. Given these two terms, the matrix equation can be put into a solvable form consistent with Eqs. (11)–(13). The matrix equation is further reduced to the form of Eq. (14) by dropping all rows and columns for points outside of the enclosed region being solved. This final equation can be solved using any standard algorithm such as an optimized LU decomposition algorithm for sparse linear systems [23] or an iterative solver such as the conjugate gradient method [24].

As the boundary problem shown in Figure 5 is only defined on the x-y plane, the z coordinate is not necessary and can be omitted from the Laplace's equations leaving just the following two equations to solve.

$$\nabla^2 x' = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) x' = 0 \tag{23}$$

$$\nabla^2 y' = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) y' = 0 \tag{24}$$

Equations (23) and (24) are approximated with finite-differences and put into matrix form. The boundary conditions are applied using the force matrix $\mathbf{F}$ and two force vector $\mathbf{u}_{x,F}$ and $\mathbf{u}_{y,F}$. The force vectors $\mathbf{u}_{x,F}$ and $\mathbf{u}_{y,F}$ contain the x coordinates and y coordinates respectively defined by the boundary conditions. The matrix equation is then reduced by eliminating the rows and columns corresponding to points outside of the outer boundary. This gives two matrix equations to be solved within the region being transformed.

$$\mathbf{L}''\mathbf{x}'' = \mathbf{b}''_x \tag{25}$$

$$\mathbf{L}''\mathbf{y}'' = \mathbf{b}''_y \tag{26}$$

These are solved according to

$$\mathbf{x}'' = \mathbf{A}''^{-1}\mathbf{b}''_x \tag{27}$$

$$\mathbf{y}'' = \mathbf{A}''^{-1}\mathbf{b}''_y. \tag{28}$$

Figure 6 displays the result of solving the boundary value problem for the coordinate transformation from $\{x,y\} \to \{x',y'\}$.

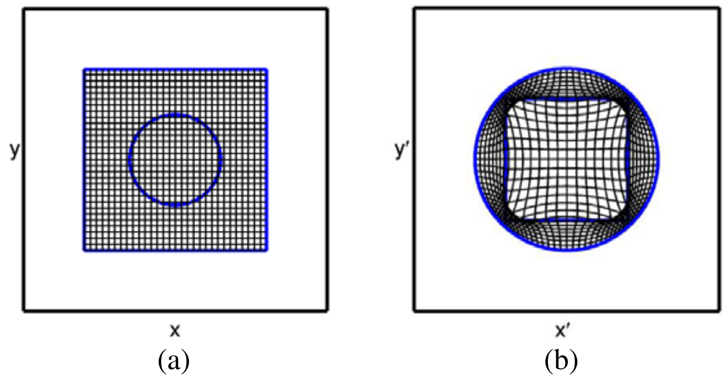


Figure 6. Coordinate transformation boundary value problem shown in Figure 5. (a) Original Cartesian coordinate system within the transformation boundaries. (b) Transformed coordinate system determined using the boundaries shown in Figure 5 and defined in Eqs. (21) and (22).

3. CALCULATING THE PERMITTIVITY AND PERMEABILITY FUNCTIONS

The permittivity function $[\varepsilon'_r]$ and permeability function $[\mu'_r]$ of the final device are calculated via TO from the background permittivity tensor $[\varepsilon_r]$, background permeability tensor $[\mu_r]$, and the spatial transforms discussed above. This reduces to calculating the following two equations at each point on the grid [1].

$$\left[\varepsilon_r'\right] = \frac{\mathbf{\Lambda}\left[\varepsilon_r\right]\mathbf{\Lambda}^T}{\det\mathbf{\Lambda}} \tag{29}$$

$$\left[\mu_r'\right] = \frac{\mathbf{\Lambda}\left[\mu_r\right]\mathbf{\Lambda}}{\det\mathbf{\Lambda}} \tag{30}$$

These equations come from a forward coordinate transform so they involve the Jacobian matrix $\mathbf{\Lambda}$ defined by

$$\mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Lambda}_{xx} & \mathbf{\Lambda}_{xy} & \mathbf{\Lambda}_{xz} \\ \mathbf{\Lambda}_{yx} & \mathbf{\Lambda}_{yy} & \mathbf{\Lambda}_{yz} \\ \mathbf{\Lambda}_{zx} & \mathbf{\Lambda}_{zy} & \mathbf{\Lambda}_{zz} \end{bmatrix} = \begin{bmatrix} \frac{\partial x'}{\partial x} & \frac{\partial x'}{\partial y} & \frac{\partial x'}{\partial z} \\ \frac{\partial y'}{\partial x} & \frac{\partial y'}{\partial y} & \frac{\partial y'}{\partial z} \\ \frac{\partial z'}{\partial x} & \frac{\partial z'}{\partial y} & \frac{\partial z'}{\partial z} \end{bmatrix} \tag{31}$$

The elements of the Jacobian matrix can be computed numerically using finite-differences [5, 25] and stored as arrays as follows.

$$\mathbf{\Lambda}_{xx}(i,j,k) = \frac{\partial x'}{\partial x} \cong \frac{x'_{i+1,j,k} - x'_{i-1,j,k}}{2\Delta x} \tag{32}$$

$$\mathbf{\Lambda}_{xy}(i,j,k) = \frac{\partial x'}{\partial y} \cong \frac{x'_{i,j+1,k} - x'_{i,j-1,k}}{2\Delta y} \tag{33}$$

$$\mathbf{\Lambda}_{xz}(i,j,k) = \frac{\partial x'}{\partial z} \cong \frac{x'_{i,j,k+1} - x'_{i,j,k-1}}{2\Delta z} \tag{34}$$

$$\mathbf{\Lambda}_{yx}(i,j,k) = \frac{\partial y'}{\partial x} \cong \frac{y'_{i+1,j,k} - y'_{i-1,j,k}}{2\Delta x} \tag{35}$$

$$\mathbf{\Lambda}_{yy}(i,j,k) = \frac{\partial y'}{\partial y} \cong \frac{y'_{i,j+1,k} - y'_{i,j-1,k}}{2\Delta y} \tag{36}$$

$$\mathbf{\Lambda}_{yz}(i,j,k) = \frac{\partial y'}{\partial z} \cong \frac{y'_{i,j,k+1} - y'_{i,j,k-1}}{2\Delta z} \tag{37}$$

$$\mathbf{\Lambda}_{zx}(i,j,k) = \frac{\partial z'}{\partial x} \cong \frac{z'_{i+1,j,k} - z'_{i-1,j,k}}{2\Delta x} \tag{38}$$

$$\mathbf{\Lambda}_{zy}(i,j,k) = \frac{\partial z'}{\partial y} \cong \frac{z'_{i,j+1,k} - z'_{i,j-1,k}}{2\Delta y} \tag{39}$$

$$\mathbf{\Lambda}_{zz}(i,j,k) = \frac{\partial z'}{\partial z} \cong \frac{z'_{i,j,k+1} - z'_{i,j,k-1}}{2\Delta z} \tag{40}$$

For two-dimensional devices like this that are uniform in the z-direction, it is only necessary to calculate four elements of the Jacobian, $\mathbf{\Lambda}_{xx}$, $\mathbf{\Lambda}_{xy}$, $\mathbf{\Lambda}_{yx}$, and $\mathbf{\Lambda}_{yy}$. The elements $\mathbf{\Lambda}_{xz}$, $\mathbf{\Lambda}_{yz}$, $\mathbf{\Lambda}_{zx}$, and $\mathbf{\Lambda}_{zy}$ are arrays of all zeros and $\mathbf{\Lambda}_{zz}$ is an array of all ones.

Iterating through the grid point-by-point, the Jacobian matrix at each grid position is assembled according to Eq. (31) with the derivatives evaluated at each point. Depending on how the coordinate transformations were determined, it is sometimes necessary to calculate the Jacobian for the backward coordinate transformation according to

$$\mathbf{\Lambda}' = \begin{bmatrix} \mathbf{\Lambda}'_{xx} & \mathbf{\Lambda}'_{xy} & \mathbf{\Lambda}'_{xz} \\ \mathbf{\Lambda}'_{yx} & \mathbf{\Lambda}'_{yy} & \mathbf{\Lambda}'_{yz} \\ \mathbf{\Lambda}'_{zx} & \mathbf{\Lambda}'_{zy} & \mathbf{\Lambda}'_{zz} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial x'} & \frac{\partial x}{\partial y'} & \frac{\partial x}{\partial z'} \\ \frac{\partial y}{\partial x'} & \frac{\partial y}{\partial y'} & \frac{\partial y}{\partial z'} \\ \frac{\partial z}{\partial x'} & \frac{\partial z}{\partial y'} & \frac{\partial z}{\partial z'} \end{bmatrix} \tag{41}$$

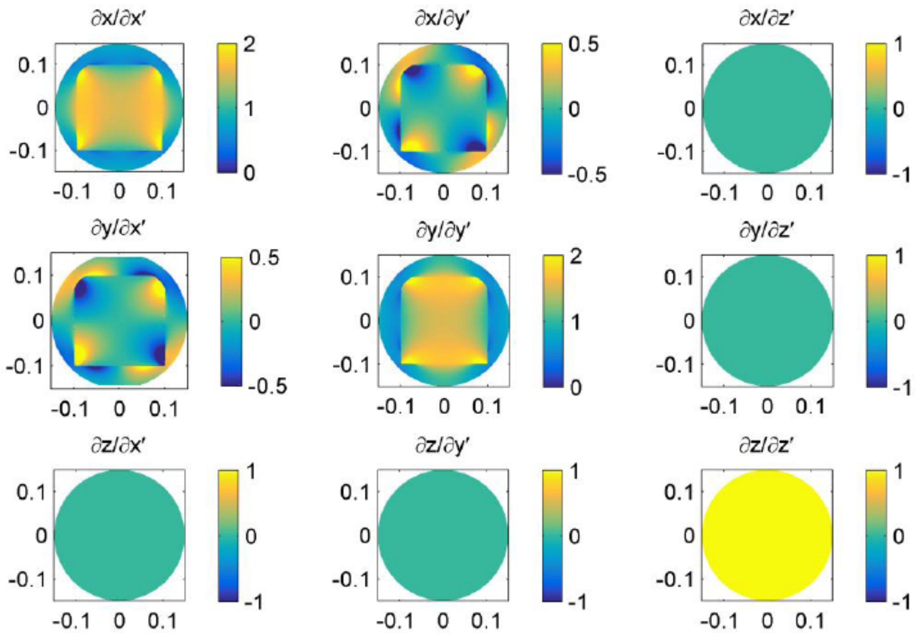


Figure 7. Elements composing the Jacobian matrix for the coordinate transformation shown in Figure 6.

The calculated derivatives are each stored in an array the same size as the grid as shown in Figure 7.

The forward Jacobian matrix can be found from the backward Jacobian matrix by calculating the inverse [26–28]. The permittivity tensor is calculated at the point i, j, k within the grid using Eq. (29) with the Jacobian matrix appropriate to the coordinate transformation. The permittivity tensor can be written where each element is stored as an array.

$$\left[\varepsilon_r'(i,j,k)\right] = \begin{bmatrix} \varepsilon_{xx}'(i,j,k) & \varepsilon_{xy}'(i,j,k) & \varepsilon_{xz}'(i,j,k) \\ \varepsilon_{yx}'(i,j,k) & \varepsilon_{yy}'(i,j,k) & \varepsilon_{yz}'(i,j,k) \\ \varepsilon_{zx}'(i,j,k) & \varepsilon_{zy}'(i,j,k) & \varepsilon_{zz}'(i,j,k) \end{bmatrix} \tag{42}$$

Likewise, the permeability tensor can be written as

$$\left[\mu_r'(i,j,k)\right] = \begin{bmatrix} \mu_{xx}'(i,j,k) & \mu_{xy}'(i,j,k) & \mu_{xz}'(i,j,k) \\ \mu_{yx}'(i,j,k) & \mu_{yy}'(i,j,k) & \mu_{yz}'(i,j,k) \\ \mu_{zx}'(i,j,k) & \mu_{zy}'(i,j,k) & \mu_{zz}'(i,j,k) \end{bmatrix} \tag{43}$$

4. IMPROVED ANISOTROPIC FINITE-DIFFERENCE FREQUENCY-DOMAIN

Electromagnetic simulation of TO devices can be performed using the AFDFD method, which is described in detail in Ref. [5]. The key matrix equation to be solved in the conventional AFDFD is

$$\left(\mathbf{C}^h\left[\boldsymbol{\mu}_r\right]^{-1}\mathbf{C}^e - \left[\boldsymbol{\varepsilon}_r\right]\right)\vec{\mathbf{e}} = \mathbf{0}. \tag{44}$$

where $\mathbf{C}^h$ and $\mathbf{C}^e$ are the curl operator matrices for the magnetic field intensity and electric field intensity respectively, $[\boldsymbol{\mu}_r]$ is the permeability tensor, and $[\boldsymbol{\varepsilon}_r]$ is the permittivity tensor as described in detail in Ref. [5]. When simulating devices with magnetically anisotropic materials, the matrix division involving the permeability tensor is very slow and numerically inefficient. To mitigate this, Eq. (44) is instead solved using the impermeability matrix $[\boldsymbol{\psi}]$

$$\left(\mathbf{C}^h\left[\boldsymbol{\psi}\right]\mathbf{C}^e - \left[\boldsymbol{\varepsilon}_r\right]\right)\vec{\mathbf{e}} = \mathbf{0} \tag{45}$$

$$\left[\boldsymbol{\psi}\right] = \left[\boldsymbol{\mu}_r\right]^{-1} \tag{46}$$

To calculate $[\boldsymbol{\psi}]$ most efficiently, the elements are calculated when they are still functions stored in arrays and not yet diagonal matrices. In this context, we have

$$[\psi(i,j,k)]=\begin{bmatrix} \psi_{xx}(i,j,k) & \psi_{xy}(i,j,k) & \psi_{xz}(i,j,k) \\ \psi_{yx}(i,j,k) & \psi_{yy}(i,j,k) & \psi_{yz}(i,j,k) \\ \psi_{zx}(i,j,k) & \psi_{zy}(i,j,k) & \psi_{zz}(i,j,k) \end{bmatrix}=\begin{bmatrix} \mu_{xx}(i,j,k) & \mu_{xy}(i,j,k) & \mu_{xz}(i,j,k) \\ \mu_{yx}(i,j,k) & \mu_{yy}(i,j,k) & \mu_{yz}(i,j,k) \\ \mu_{zx}(i,j,k) & \mu_{zy}(i,j,k) & \mu_{zz}(i,j,k) \end{bmatrix}^{-1}. \tag{47}$$

The elements of $[\psi(i,j,k)]$ are derived from the tensor $[\mu_r(i,j,k)]$ by calculating the matrix inverse symbolically.

$$[\psi(i,j,k)]=\frac{\begin{bmatrix} \mu_{yy}\mu_{zz}-\mu_{yz}\mu_{zy} & \mu_{xz}\mu_{zy}-\mu_{xy}\mu_{zz} & \mu_{xy}\mu_{yz}-\mu_{xz}\mu_{yy} \\ \mu_{yz}\mu_{zx}-\mu_{yx}\mu_{zz} & \mu_{xx}\mu_{zz}-\mu_{xz}\mu_{zx} & \mu_{xz}\mu_{yx}-\mu_{xx}\mu_{yz} \\ \mu_{yx}\mu_{zy}-\mu_{yy}\mu_{zx} & \mu_{xy}\mu_{zx}-\mu_{xx}\mu_{zy} & \mu_{xx}\mu_{yy}-\mu_{xy}\mu_{yx} \end{bmatrix}}{\mu_{xx}\mu_{yy}\mu_{zz}-\mu_{xx}\mu_{yz}\mu_{zy}-\mu_{xz}\mu_{yy}\mu_{zx}-\mu_{xy}\mu_{yx}\mu_{zz}+\mu_{xy}\mu_{yz}\mu_{zx}+\mu_{xz}\mu_{yx}\mu_{zy}} \tag{48}$$

For brevity, the dependence on the array indices i, j, and k have been dropped from these equations. All of these calculations are performed point-by-point on arrays instead of with matrices. After the elements of $[\psi(i,j,k)]$ are calculated, they are reshaped into diagonal matrices and assembled into the overall impermeability matrix as follows.

$$[\boldsymbol{\psi}]=\begin{bmatrix} \boldsymbol{\psi}_{xx} & \boldsymbol{\psi}_{xy} & \boldsymbol{\psi}_{xz} \\ \boldsymbol{\psi}_{yx} & \boldsymbol{\psi}_{yy} & \boldsymbol{\psi}_{yz} \\ \boldsymbol{\psi}_{zx} & \boldsymbol{\psi}_{zy} & \boldsymbol{\psi}_{zz} \end{bmatrix} \tag{49}$$

where $\boldsymbol{\psi}_{mn}=\text{diag}\{\psi_{mn}(i,j,k)\}$.

Using the impermeability tensor rather than the permeability tensor to solve the wave equation in Eq. (45) decreased the time required to solve a two dimensional AFDFD simulation for a flat TO lens on a 90×77 cell grid from 727.18 seconds to 8.33 seconds. Also the amount of memory required to solve AFDFD using the impermeability tensor was drastically reduced compared to the conventional AFDFD algorithm using the permeability tensor. The simulation with the impermeability tensor required 105.7 MB to be allocated whereas the simulation with the permeability tensor required 6.962 GB to be allocated.

5. BENCHMARK AND EXAMPLES

5.1. Cylindrical Electromagnetic Cloak

To benchmark our approach, we duplicated the famous cylindrical electromagnetic cloak presented in Ref. [2]. This was based on the closed form coordinate transformation given by

$$\begin{aligned} r' &= R_1 + r\frac{R_2-R_1}{R_2} \\ \theta' &= \theta \\ z' &= z \end{aligned} \tag{50}$$

where the cloak has an inner radius R_1 and outer radius R_2. Figure 8 shows the outlines of the boundaries of the coordinate transformation given by Eq. (50).

In order to calculate the coordinate transformation numerically, the following boundary conditions were applied

$$\Gamma_2'(x',y') = \Gamma_2(x,y) \tag{51}$$

$$\Gamma_1'(x',y') = 0 \tag{52}$$

Given Eq. (50), the transformation from $\{x,y,z\}\rightarrow\{x',y',z'\}$ experienced a strong singularity in the cloaked region. This was overcome by calculating the backward rather than the forward coordinate transformation. The material tensors were calculated by following the approach described previously. The resulting permittivity tensor is shown in Figure 10. For a cylindrical cloak in free space, the permeability tensor is identical to the permittivity tensor.

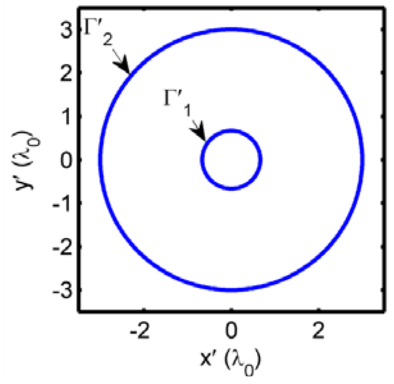


Figure 8. Coordinate transformation boundary conditions for a cylindrical cloak.

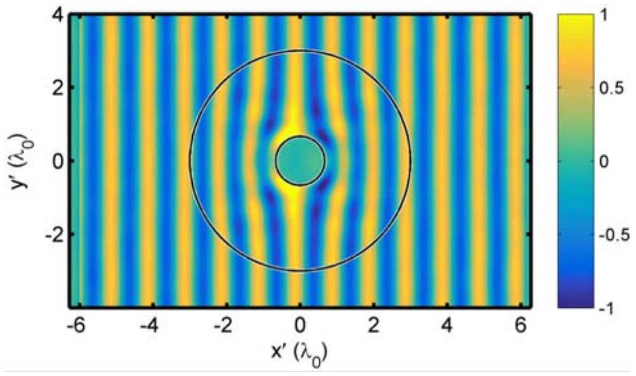


Figure 9. AFDFD simulation of cylindrical electromagnetic cloak.

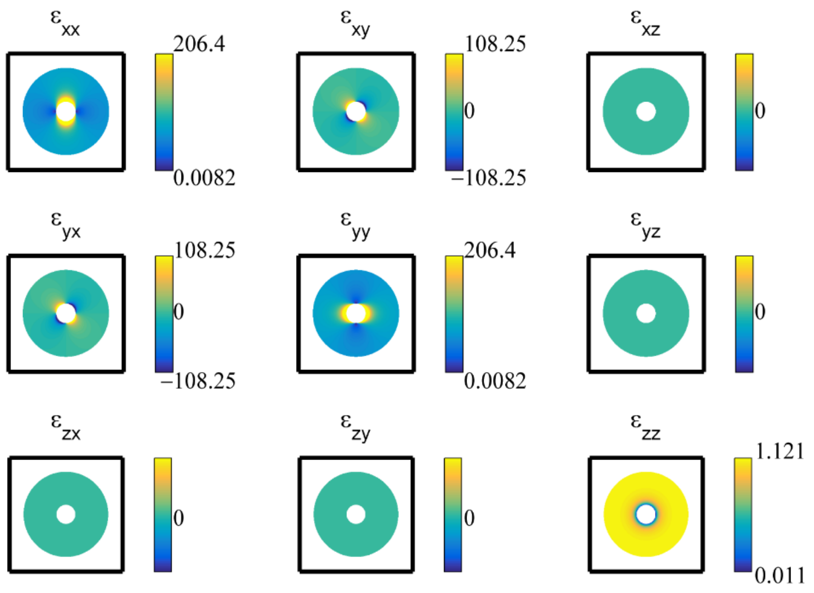


Figure 10. Permittivity tensor for cylindrical electromagnetic cloak. The permeability tensor is identical for a cloak based in free space. All tensor components which are not visualized are equal to one for the elements along the diagonal and zeros for the off-diagonal elements. The values determined for this cloak are based on a grid resolution of $\lambda_0/120$. As the grid resolution is made smaller, the values of the materials required would approach infinity or zero due the cloaked region being a singularity.

The device was simulated using the improved AFDFD. Dirichlet boundary conditions were used for the y-axis boundaries while a uniaxial perfectly matched layer (UPML) was used for the x-axis boundaries. The result is shown in Figure 9.

5.2. Flat Lens

Figure 11 shows the boundary conditions necessary to design a flat lens using TO as described in Ref. [29]. By solving Laplace's equation for the boundaries

$$\begin{aligned} \Gamma_1 &\rightarrow \Gamma'_1 \\ \Gamma_2 &\rightarrow \Gamma'_2 \\ \Gamma_3 &\rightarrow \Gamma'_3 \\ \Gamma_4 &\rightarrow \Gamma'_4 \end{aligned} \tag{53}$$

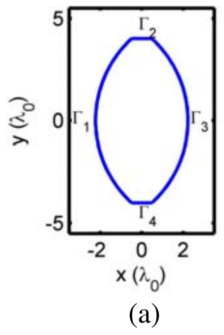
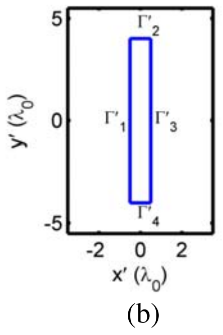


Figure 11. Coordinate transformation for transformation optics bend. (a) is the original coordinate space while that (b) is the transformed space.

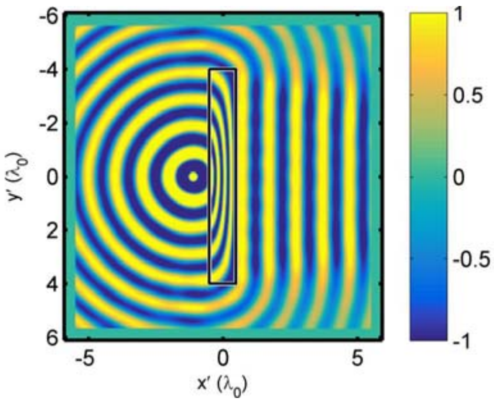


Figure 12. AFDFD simulation of flat TO lens.

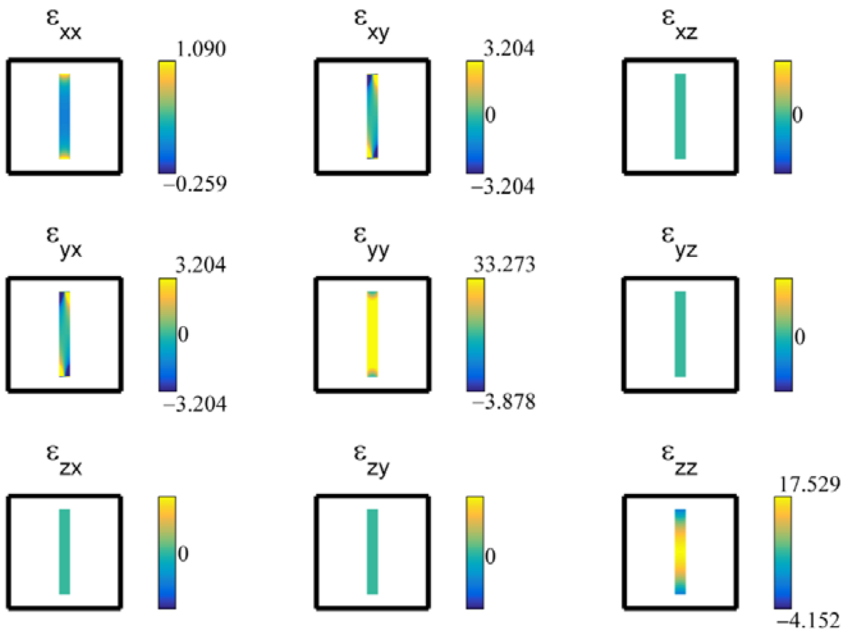


Figure 13. Permittivity tensors for flat transformation optics lens. The permeability tensor is identical for a bend based in free space. All tensor components which are not visualized are zero throughout the grid. The material values for this lens are for a grid resolution of $\lambda_0/120$.

the coordinate transformation was computed numerically.

With the coordinates in the new system known, the Jacobian matrix was calculated using Eq. (31) and the material tensors determined using Eqs. (29) and (30). The resulting permittivity is shown in Figure 13. The permeability is identical to the permittivity in this situation because the original space is vacuum.

Using AFDFD, the material tensors were placed within a grid surrounded by a UPML and a cylindrical source was used. Aside from diffraction around the edges, the electric field in Figure 12 can be seen to be converted to a plane wave as would be expected from a conventional lens.

5.3. Arbitrary Cloak

This section will demonstrate the method implemented above with the arbitrary metamaterial cloak shown in Figure 14. The pickaxe shape bounded by Γ'_1 is being cloaked by the region bounded by Γ'_2. In order to cloak the object, a coordinate system in which the axes wrap around the object without penetrating the object itself must be implemented within the region between the two boundaries. In order to avoid singular Jacobian matrices in the cloaked region, the backward coordinate transformation was implemented rather than the forward transformation. The boundary conditions, which were derived from those described in Ref. [3], are given as follows.

$$\Gamma'_2\left(x', y'\right) = \Gamma_2\left(x, y\right) \tag{54}$$

$$\Gamma'_1\left(x', y'\right) = 0 \tag{55}$$

Following the procedure described in the previous sections of this paper the material tensors were calculated. The resulting permittivity tensor is shown in Figure 16 with the permeability tensor being the same.

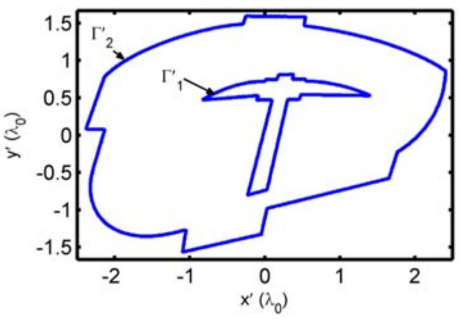


Figure 14. Outline of arbitrary electromagnetic cloak and object to be cloaked.

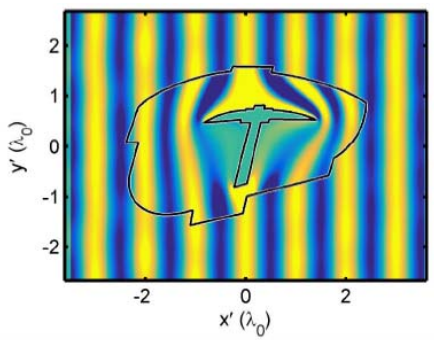


Figure 15. AFDFD simulation of arbitrary electromagnetic cloak.

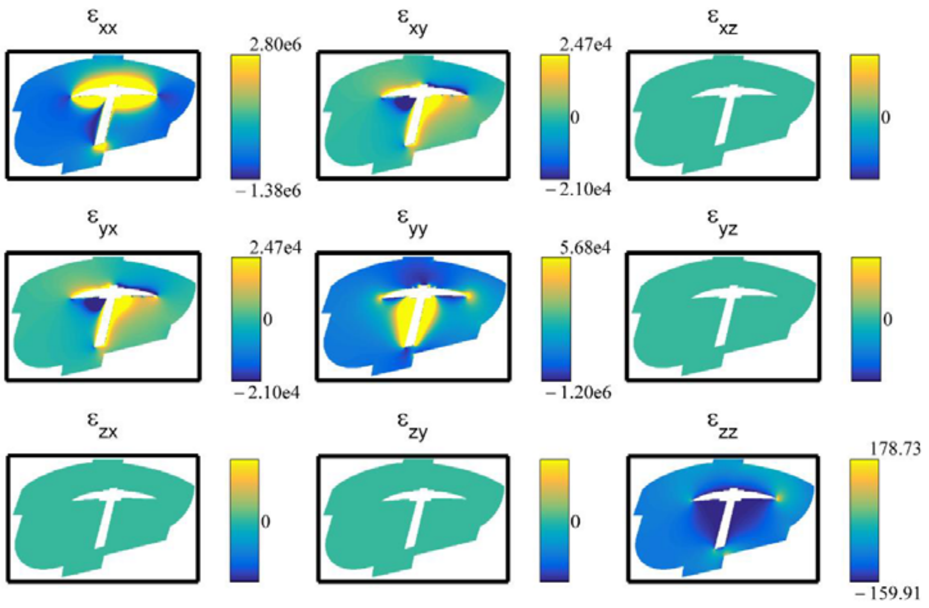


Figure 16. Permittivity tensor for arbitrary electromagnetic cloak. The permeability tensor is identical for a cloak based in free space. All tensor components which are not visualized are equal to one for the elements along the diagonal and zeros for the off-diagonal elements.

Using AFDFD, the TO cloak was simulated for a TM polarized wave as shown in Figure 15. The material tensors for the cloak were placed between a UPML on the x-axis boundaries and Dirichlet boundary conditions on the y-axis boundaries. As can be seen, very little power is scattered and the phase of the wave is very well preserved.

5.4. Transformation Optics Bend

Creating a bend using numerical TO involves solving Laplace's equation for the boundary value problem shown in Figure 17. Rather than transform from $\{x, y\}$ to $\{x', y'\}$, the backward transformation was chosen because mapping the arbitrarily distorted boundaries shown on the right in Figure 17 to the boundaries of the original system could be done with simple analytical expressions. Solving Laplace's

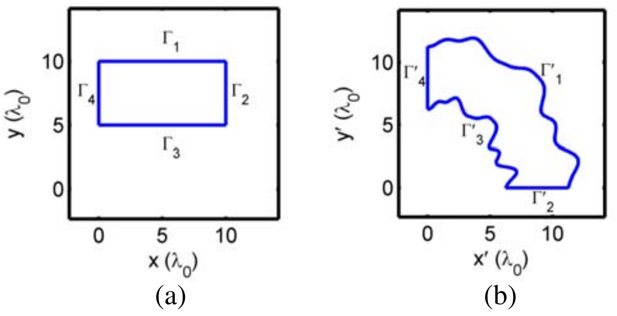


Figure 17. Coordinate transformation for transformation optics bend. (a) is the original coordinate space while that (b) is the transformed space.

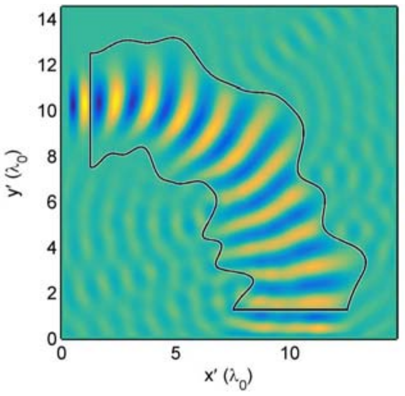


Figure 18. AFDFD simulation of TO electromagnetic bend.

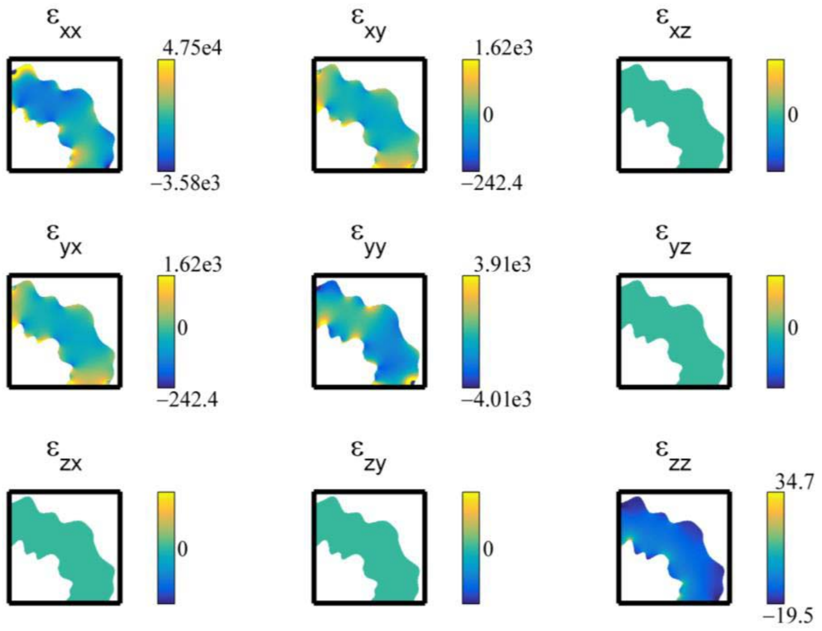


Figure 19. Permittivity tensors for arbitrary electromagnetic bend. The permeability tensor is identical for a bend based in free space. All tensor components which are not visualized are zero throughout the grid.

equation with the boundary values

$$\begin{aligned} \Gamma_1' &\to \Gamma_1 \\ \Gamma_2' &\to \Gamma_2 \\ \Gamma_3' &\to \Gamma_3 \\ \Gamma_4' &\to \Gamma_4 \end{aligned} \tag{56}$$

provides the coordinates of the backward coordinate transformation.

Using the procedure described above, the material tensors were calculated as shown in Figure 19 and placed in a grid surrounded by a UPML. This was simulated using AFDFD and the result is shown in Figure 18.

6. CONCLUSIONS

A simple technique for generating and simulating TO devices with arbitrary geometry was described in this article. Laplace's equation was used to generate the coordinates of the spatial transforms. The method described uses a finite-difference approach so it is simple and straightforward to implement. Other approaches in the literature used commercial software such as COMSOL, making it difficult or impossible to import and export complex data. The technique described in this paper can be implemented independent of a commercial solver and in virtually any programming environment. This allows much greater flexibility in manipulating the data and importing the generated materials into custom electromagnetic solvers such as finite-difference time-domain, finite-difference frequency-domain, beam propagation method, and more. Further, an improvement to the standard AFDFD method was outlined that greatly improves the speed and efficiency for simulating TO devices. The new approach utilized the impermeability tensor instead of the permeability tensor in the matrix wave equation.

Several examples were presented to benchmark the approach and demonstrate its versatility. The devices included the classical cylindrical cloak, an arbitrarily shaped cloak, a flat lens, and an arbitrarily shaped bend. Devices with arbitrary shapes were verified through simulation using the improved AFDFD. The simulation results showed excellent performance of the devices.

REFERENCES

1. Pendry, J. B., D. Schurig, and D. R. Smith, "Controlling electromagnetic fields," *Science*, Vol. 312, 1780–1782, 2006.
2. Schurig, D., J. Mock, B. Justice, S. A. Cummer, J. B. Pendry, A. Starr, et al., "Metamaterial electromagnetic cloak at microwave frequencies," *Science*, Vol. 314, 977–980, 2006.
3. Hu, J., X. Zhou, and G. Hu, "Design method for electromagnetic cloak with arbitrary shapes based on Laplace's equation," *Optics Express*, Vol. 17, 1308–1320, 2009.
4. Chang, Z., X. Zhou, J. Hu, and G. Hu, "Design method for quasi-isotropic transformation materials based on inverse Laplace's equation with sliding boundaries," *Optics Express*, Vol. 18, 6089–6096, 2010.
5. Rumpf, R. C., C. R. Garcia, E. A. Berry, and J. H. Barton, "Finite-difference frequency-domain algorithm for modeling electromagnetic scattering from general anisotropic objects," *Progress In Electromagnetics Research B*, Vol. 61, 55–67, 2014.
6. Landy, N. I. and W. J. Padilla, "Guiding light with conformal transformations," *Optics Express*, Vol. 17, 14872–14879, 2009.
7. Ma, J.-J., X.-Y. Cao, K.-M. Yu, and T. Liu, "Determination the material parameters for arbitrary cloak based on Poisson's equation," *Progress In Electromagnetics Research M*, Vol. 9, 177–184, 2009.
8. Chen, X., Y. Fu, and N. Yuan, "Invisible cloak design with controlled constitutive parameters and arbitrary shaped boundaries through Helmholtz's equation," *Optics Express*, Vol. 17, 3581–3586, Mar. 2, 2009.

9. Rumpf, R. C. and J. Pazos, "Synthesis of spatially variant lattices," *Optics Express*, Vol. 20, 15263–15274, 2012.
10. Smith, G. D., *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, Oxford University Press, 1985.
11. LeVeque, R. J., "Finite difference methods for differential equations," *Draft Version for Use in AMath*, Vol. 585, 1998.
12. Golub, G. H. and C. F. van Loan, *Matrix Computations*, Vol. 3, JHU Press, 2012.
13. Johnson, H. and C. S. Burrus, "On the structure of efficient DFT algorithms," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 33, 248–254, 1985.
14. Iserles, A., *A First Course in the Numerical Analysis of Differential Equations*, Cambridge University Press, 2009.
15. Chapra, S. C. and R. P. Canale, *Numerical Methods for Engineers*, Vol. 2, McGraw-Hill, 2012.
16. Arfken, G. and H. Weber, *Mathematical Methods for Physicists*, 6th Edition, Academic, New York, 2005.
17. Morgan, M. A., *Finite Element and Finite Difference Methods in Electromagnetic Scattering*, Elsevier, 2013.
18. Luong, P., "A mathematical coastal ocean circulation system with breaking waves and numerical grid generation," *Applied Mathematical Modelling*, Vol. 21, 633–641, 1997.
19. Eiseman, P. R., "Grid generation for fluid mechanics computations," *Annual Review of Fluid Mechanics*, Vol. 17, 487–522, 1985.
20. Thompson, J. F., Z. U. Warsi, and C. W. Mastin, *Numerical Grid Generation: Foundations and Applications*, Vol. 45, North-holland Amsterdam, 1985.
21. Sanmiguel-Rojas, E., J. Ortega-Casanova, C. del Pino, and R. Fernandez-Feria, "A Cartesian grid finite-difference method for 2D incompressible viscous flows in irregular geometries," *Journal of Computational Physics*, Vol. 204, 302–318, 2005.
22. Akcelik, V., B. Jaramaz, and O. Ghattas, "Nearly orthogonal two-dimensional grid generation with aspect ratio control," *Journal of Computational Physics*, Vol. 171, 805–821, 2001.
23. Davis, T. A., "Algorithm 832: UMFPACK V4.3 — An unsymmetric-pattern multifrontal method," *ACM Transactions on Mathematical Software (TOMS)*, Vol. 30, 196–199, 2004.
24. Paige, C. C. and M. A. Saunders, "Solution of sparse indefinite systems of linear equations," *SIAM Journal on Numerical Analysis*, Vol. 12, 617–629, 1975.
25. Rumpf, R. C., "Simple implementation of arbitrarily shaped total-field/scattered-field regions in finite-difference frequency-domain," *Progress In Electromagnetics Research B*, Vol. 36, 221–248, 2012.
26. Schutz, B., *A First Course in General Relativity*, Cambridge University Press, 2009.
27. Hobson, M. P., G. P. Efstathiou, and A. N. Lasenby, *General Relativity: An Introduction for Physicists*, Cambridge University Press, 2006.
28. Liseikin, V., "Coordinate transformations," *Grid Generation Methods*, 31–66, Springer, Netherlands, 2010.
29. Kwon, D.-H. and D. H. Werner, "Transformation electromagnetics: An overview of the theory and applications," *IEEE Antennas and Propagation Magazine*, Vol. 52, 24–46, 2010.