Angora: A Free Software Package for Finite-Difference Time-Domain Electromagnetic Simulation

İlker R. Çapoğlu[1], Allen Taflove[2], and Vadim Backman[1]

[1]Biomedical Engineering Department of Northwestern University
2145 Sheridan Road, Evanston, IL 60208 USA
E-mail:capoglu@northwestern.edu; v-backman@northwestern.edu

[2] Electrical Engineering and Computer Science Department of Northwestern University
2145 Sheridan Road, Evanston, IL 60208 USA
E-mail: taflove@eecs.northwestern.edu

Abstract

Angora is a free, open-source software package implementing the Finite-Difference Time-Domain (FDTD) Method. This article explains how to access and use *Angora*, and introduces its features. Examples of its configuration and use are given. These include treatment of planar stratified and random materials, the creation of current sources and incident beams, near-field-to-far-field transformation, optical imaging and nano-optics, and the recording of fields.

Keywords: Open source software; finite difference methods; time domain analysis; parallel algorithms; optical imaging; FDTD

1. Introduction

The Finite-Difference Time-Domain (FDTD) method [1, 2] has gained considerable popularity in the last two decades, among other numerical electromagnetic solution methods. This is mainly due to its simple and intuitive core algorithm, which makes its software implementation more straightforward. FDTD is often the method of choice for highly inhomogeneous and/or nonlinear material distributions. FDTD can also sweep a wide frequency range with a single simulation, since it operates directly in the time domain.

There are many proprietary software tools on the market for FDTD simulation, and a few free software implementations. Examples of these can be found in [3]. In this article, we introduce a new, free FDTD software package named *Angora*. This software package is now available for download at http://www.angorafdtd.org [4]. A comprehensive user's manual can be found on this Web site. In the following sections, we will summarize the important features of *Angora*, and give some examples.

Please note that the configuration settings and features mentioned in this article are subject to frequent modification. All of the configuration files used for the examples in this article can be downloaded from the *Angora* Web site [5]. Please consult these configuration files and the *Angora* user's manual for the latest reference on any specific configuration. In the following, many configuration variables and features were omitted for brevity.

2. Important Features of *Angora*

Angora is currently available only for the GNU/*Linux* operating system. *Angora* can be downloaded from the *Angora* Web site [4] in source-code format, as well as in binary format for x86_64 GNU/*Linux* systems. On the Web site, you can also find information on how to receive updates on *Angora*.

Some important features of *Angora*, such as focused laser-beam illumination and numerical optical-image synthesis, are geared toward optics applications. Others, such as random-medium generation and multilayer support, are useful for any antenna, microwave, propagation, or remote-sensing application.

Some key features of *Angora* are as follows:

- An automatic build/install mechanism for the GNU/*Linux* operating system.
- A user-friendly configuration using text-based configuration files.
- Full parallelizability in three dimensions, based on the *Message Passing Interface* (*MPI*) standard.
- Support for planar multilayered spaces:
  - Total-field/scattered-field (TF/SF) plane-wave source.
  - Phasor-domain near-field-to-far-field transformation (NFFFT).
- Focused laser beams with arbitrary polarization, orientation, and propagation direction (also supports multilayered spaces.)
- Convolution perfectly-matched layer (CPML) absorbing boundaries.
- Support for HDF5, a portable file storage format.
- Field-value recording (two-dimensional, one-dimensional, single point) and movie generation.
- Generating random constitutive parameter distributions.
- Reading constitutive-parameter distributions from files.
- Synthesizing the optical image of the simulated sample.

As mentioned above, *Angora* supports the parallelization of the simulation by sectioning the grid along all three dimensions, and assigning each Cartesian sub-grid to an individual processor. Inter-processor communication consists only of the interchange of field information across the mutual faces of neighboring grids, in order to allow the energy to propagate from one sub-grid to the other. For minimum latency, the total surface area of the sub-grids should be minimized. *Angora* tries to minimize this surface area by making each sub-grid as close to a cube as possible. Both parallel and non-parallel binary versions are available for download on the *Angora* Web site. The precompiled parallel binary uses the *OpenMPI* implementation [6] of the *MPI* standard.

3. Configuration of *Angora*

Angora is configured using text files, called *configuration files*, which include information on every aspect of the simulation. The directives and values that specify the simulation should be provided in a certain format, based on the syntax imposed by the *libconfig* library [7]. More information on this syntax is provided in the *Angora* user's manual [8]. For example, some basic parameters of the FDTD simulation are set by placing the following lines in the configuration file:

```
dx = 1e-3;
courant = 0.98;
grid_dimension_x = 10e-2;
grid_dimension_y = 10e-2;
grid_dimension_z = 10e-2;
pml_thickness = 5e-3;
num_of_time_steps = 1500;
```

The first line sets the spatial step size, which is the dimension of the cubic FDTD voxels that make up the simulation space. All length units are assumed in meters. In the above example, the spatial step size is 1 mm. The second line sets the Courant number, which determines the ratio between the time step and the spatial step in the simulation. With the above value, the time step will be

$$\Delta t = 0.98\frac{\Delta x}{\sqrt{3}\,c} = 1.8873 \text{ ps} .$$

The next three lines set the dimensions of the simulation space in the *x*, *y*, and *z* directions. The `pml_thickness` line determines the thickness of the perfectly-matched layer (PML) absorbing boundary, which is based on the convolution perfectly-matched layer formulation [9]. With the extra perfectly-matched layer thickness, the total simulation space has dimensions of 10.5 cm × 10.5 cm × 10.5 cm. Finally, the number of time steps in the simulation is specified using the `num_of_time_steps` variable.

Other properties of the simulation are configured similarly, by assigning values to various variables. Examples of these are given in the following sections.

4. Placing Simple Objects

n *Angora*, a geometrical object is defined as a combination of a *shape* and a *material*. Shapes are defined inside the `Shapes` group[1]:

[1]Groups in *Angora* are collections of variable assignments delineated by curly brackets {...}. For more information on the *Angora* configuration syntax, see the user's manual.

```
Shapes:
{
 RectangularBoxes:
 (
 {
 shape_tag = "mybox";
 back_coord_x = -10e-3;
 front_coord_x = 10e-3;
 left_coord_y = -40e-3;
 right_coord_y = 5e-3;
 lower_coord_z = -40e-3;
 upper_coord_z = 5e-3;
 }
 );
 Spheres:
 (
 {
 shape_tag = "mysphere";
 center_coord_x = 0;
 center_coord_y = 15e-3;
 center_coord_z = 15e-3;
 radius = 25e-3;
 }
 );
};
```

In the above example, two shapes are defined: One rectangular-prism shape with the limiting coordinates in x, y, and z as listed; and a spherical shape with its center at $(0, 1.5\text{ cm}, 1.5\text{ cm})$ with respect to the origin (which is the center of the simulation grid, by default) and a radius of 2.5 cm. The shapes are given the names "mybox" and "mysphere", which will be referenced later when they are placed in the simulation space.

The materials filling the shapes are defined in a Materials list[2]. In the following, two materials named "mat1" and "mat2" are defined with relative permittivities 2.1 and 2.0, respectively:

```
Materials:
(
 {
 material_tag = "mat1";
 rel_permittivity = 2.1;
 },
 {
 material_tag = "mat2";
 rel_permittivity = 2.0;
 }
);
```

The two objects are placed in the grid by referring to the names (string tags) of the shape and material objects defined above. These names are combined in an object definition inside the Objects list. This list should be placed inside the SimulationSpace group, which processes every material-related setting in order of appearance in the configuration file:

[2]Lists in *Angora* are collections of values (integers, groups, even other lists) delineated by parentheses $(\ldots)$. In the *Angora* context, lists almost invariably consist of groups. For more information on the *Angora* configuration syntax, see the user's manual.

```
SimulationSpace:
{
 Objects:
 (
 {
 material_tag = "mat1";
 shape_tag = "mybox";
 },
 {
 material_tag = "mat2";
 shape_tag = "mysphere";
 }
 );
};
```

In Figure 1, the yz cross section of the relative permittivity distribution of the FDTD grid at $x = 0$ is shown, after the placement of these two objects into the grid.

5. Planar Stratification

Planar stratification (layering) is introduced to the simulation grid using the MaterialSlabs list. Currently, the layers can only be stacked in the z direction. This may be generalized to arbitrary directions in the future. The MaterialSlabs list should also be placed inside the SimulationSpace group, since the order of placement is important:

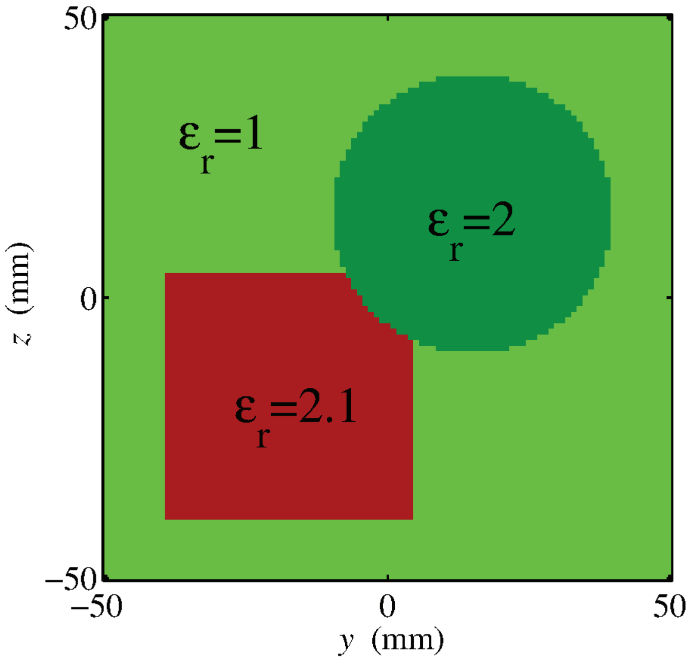


Figure 1. The yz cross section of the permittivity distribution in the FDTD grid at $x = 0$ after the placement of two simple shapes.

```
SimulationSpace:
{
 MaterialSlabs:
 (
 {
 material_tag = "mat1";
 min_coord = "min";
 max_coord = -1e-2;
 },
 {
 material_tag = "mat2";
 min_coord = 1e-2;
 max_coord = 4e-2;
 }
 );
};
```

The two groups in the `MaterialSlabs` list represent two planar layers. The first extends from the $-z$ boundary of the grid to 1 cm below the origin, and the second extends from $z = 1$ cm to $z = 4$ cm. The first layer extends into the absorbing perfectly-matched layer: it therefore represents a half-space. In Figure 2, the yz cross section of the relative permittivity distribution of the FDTD grid at $x = 0$ is shown after the placement of these two planar layers.

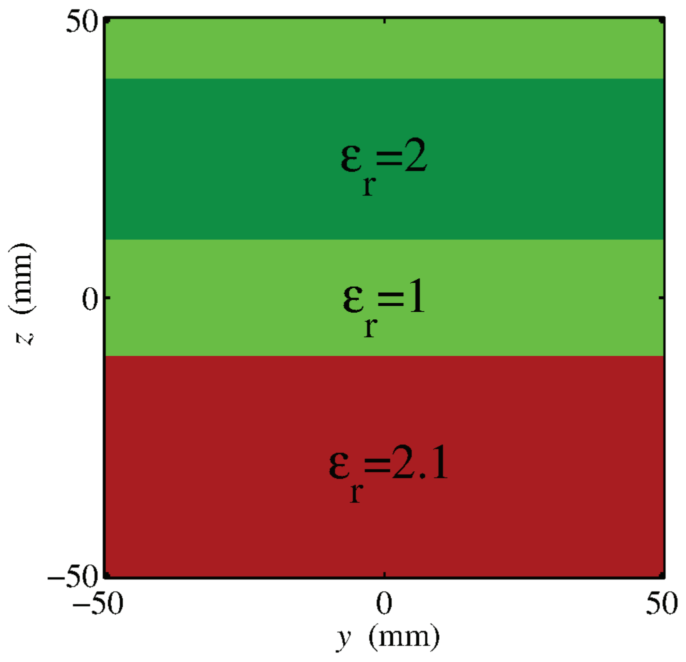


Figure 2. The yz cross section of the permittivity distribution in the FDTD grid at $x = 0$ after the placement of the two planar layers. The lowermost layer extended into the perfectly-matched layer, and was therefore a half-space.

6. Reading Constitutive Parameters from Files

A highly-inhomogeneous material region can be placed into the simulation grid by reading a file that stores the constitutive-parameter distribution of the region in a simple binary format. The `MaterialsFromFiles` list is used for this purpose:

```
SimulationSpace:
{
 MaterialsFromFiles:
 (
 {
 file_name = "materialfile";
 constitutive_param_type =
 "rel_permittivity";
 anchor = "center";
 coord_x = 0;
 coord_y = 0;
 coord_z = 0;
 datatype = "double";
 }
 );
};
```

The input file `"materialfile"` should be a binary file with the following data, in the order specified:

- n_x : An integer (four bytes) specifying the extent of the three-dimensional region (in grid cells) in the x dimension

- n_y : An integer (four bytes) specifying the extent of the three-dimensional array (in grid cells) in the y dimension

- n_z : An integer (four bytes) specifying the extent of the three-dimensional array (in grid cells) in the z dimension

- A three-dimensional floating-point array of size $\left(n_x \times n_y \times n_z\right)$, with each element of type `double` (eight bytes) or `float` (four bytes), depending on the `datatype` variable above. The array should be laid out in *column-major order*, i.e., x dimension first.

With the above assignment to the `constitutive_param_type` variable, the array read from the file will set the relative permittivity of the region. The positioning is determined by the coordinates of the *anchor* point in the three-dimensional region. In the above example, the anchor is set to the center of the region, which is then placed at $(0,0,0)$ with respect to the center of the grid.

The placement of the material region can be customized much further. Different constitutive parameters can also be superimposed using multiple definitions. For more details, please consult the user's manual.

7. Random Materials

Using the `RandomMaterials` group, a region of the simulation space can be filled with a *random* material with certain statistical properties. *Angora* supports the creation of homogeneous, isotropic, multivariate-normal-distributed, Whittle-Matérn-correlated material properties inside geometrical shapes previously defined (see the `Shapes` group in Section 4). The generated random distribution can be assigned to the relative permittivity, the relative permeability, the electric conductivity, or the magnetic conductivity. Multiple constitutive parameters with different statistical distributions can be overlaid using multiple groups in the configuration file. For example, the following assignments randomize both the relative permittivity and relative permeability distribution of a region, with the same distribution:

```
SimulationSpace:
{
 RandomMaterials:
 {
 WhittleMaternCorrelated:
 (
 {
 constitutive_param_type =
 "rel_permittivity";
 mean = 1.33;
 std_dev = 0.05;
 corr_len = 100e-9;
 m = 2.0;
 shape_tag = "rand_mat_shape";
 },
 {
 constitutive_param_type =
 "rel_permeability";
 mean = 1.33;
 std_dev = 0.05;
 corr_len = 100e-9;
 m = 2.0;
 shape_tag = "rand_mat_shape";
 }
 );
 };
};
```

The Whittle-Matérn correlation function [10] is an isotropic three-parameter stochastic model, with parameters l_c, σ^2, and m:

$$B(r) = \sigma^2 \frac{2^{5/2-m} \left(r/l_c\right)^{m-3/2}}{\Gamma(m-3/2)} K_{m-3/2}\left(r/l_c\right). \quad (1)$$

Here, $K_\nu(\cdot)$ is the modified Bessel function of second kind, and $\Gamma(\cdot)$ is the Gamma function. The parameter l_c (represented by `corr_len` above) describes the index correlation distance, and the parameter σ (represented by `std_dev` above) is the standard deviation. The third parameter, m (represented by `m` above) is a shape parameter that modifies the overall behavior of the function. The model reduces to several important specific functions for certain values of m: As $m \to \infty$, the function approaches a Gaussian (or normal) correlation. When $m = 2$, the function is a decaying exponential. A singularity exists at $m = 3/2$, and the function collapses to zero because of the normalization factor of $\Gamma(m-3/2)$. However, the unnormalized $B(r)$ becomes a delta function for $m = 3/2$, and the corresponding power-spectral density is the often-used Henyey-Greenstein function. Values of $m < 3/2$ correspond to a fractal index distribution with correlation function described by a power law. A two-dimensional slice from an example random distribution is drawn in grayscale in Figure 3.

8. Infinitesimal (Hertzian) Current Sources

An infinitesimal (Hertzian) electric-dipole source can be approximated in the FDTD by a current element at the edge of a grid voxel. Before defining the dipole itself, we have to define the waveform of its current moment. This is done inside a `Waveforms` definition:

```
Waveforms:
{
 ModulatedGaussianWaveforms:
 (
 {
 waveform_tag = "my_wf";
 modulation_type = "sine";
 tau = 1.36711e-10;
 f_0 = 7.49481e9;
 }
 );
};
```

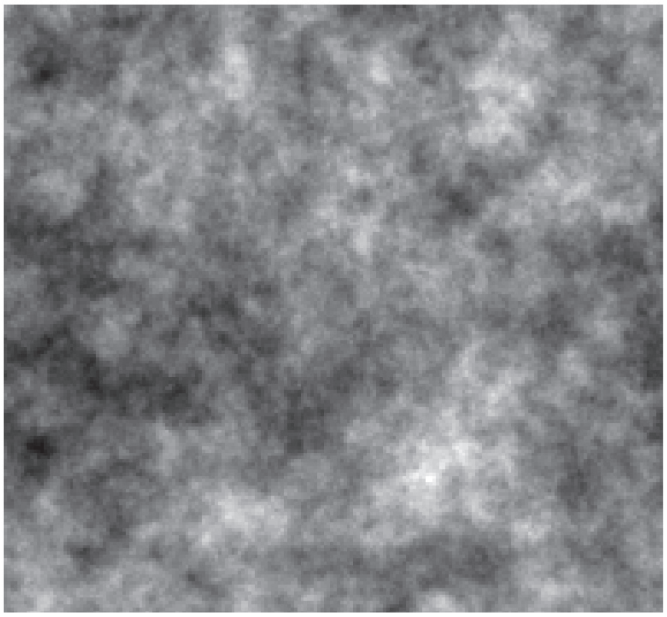


Figure 3. A two-dimensional slice of an example three-dimensional random distribution, shown in grayscale. The probability density function was multivariate normal, while the two-point correlation function was Whittle-Matérn (see Equation (1)).

A modulated Gaussian waveform is defined here, with the functional form

$$f(t) = \sin(2\pi f_0 t)\exp\left[-(t/\tau)^2/2\right],$$

and given the name "my_wf". The variables tau and f_0 correspond to τ and f_0, respectively. All time and frequency units are in seconds and Hz, respectively. We can now define the infinitesimal dipole. The PointSources list is used for this purpose:

```
PointSources:
(
 {
 coord_x = 0;
 coord_y = 0;
 coord_z = 0;
 source_orientation = "x_directed";
 waveform_tag = "my_wf";
 j_0 = 1.0;
 }
);
```

In the above assignment, the coordinates and the orientation of the dipole are determined by the first four lines. The waveform that represents the current moment of the dipole (in Ampere × m) is set by the string tag "my_wf". The variable j_0 adds an extra prefactor to the current-moment waveform.

As an example, we placed this dipole inside the grid with the two basic shapes shown in Figure 1 (Section 4). In Figure 4, the field and permittivity distributions are shown on a *yz* cross section of the FDTD grid at $x = 0$, $t = 850$ ps. The brightness of a pixel is proportional to the *x* component of the electric field at that pixel. The permittivity distribution is shown in red where it is different from one.

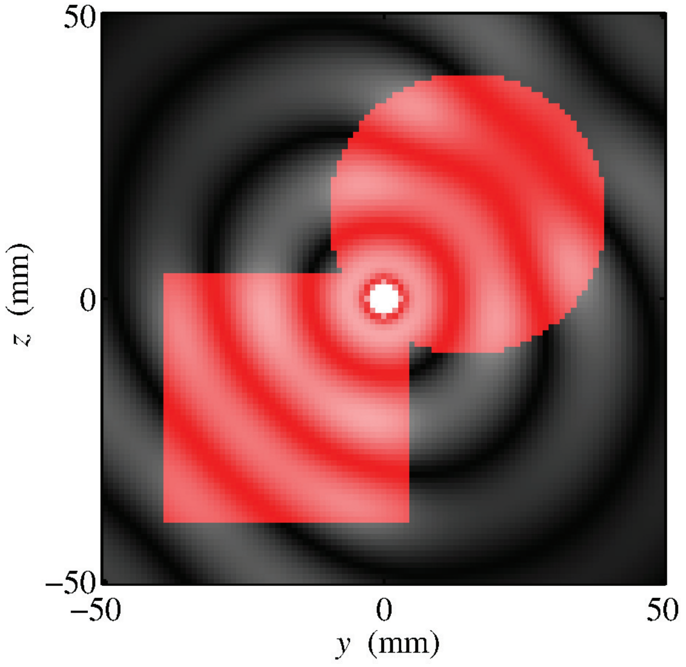


Figure 4. The field distribution created by an *x*-directed infinitesimal (Hertzian) electric current source at the origin.

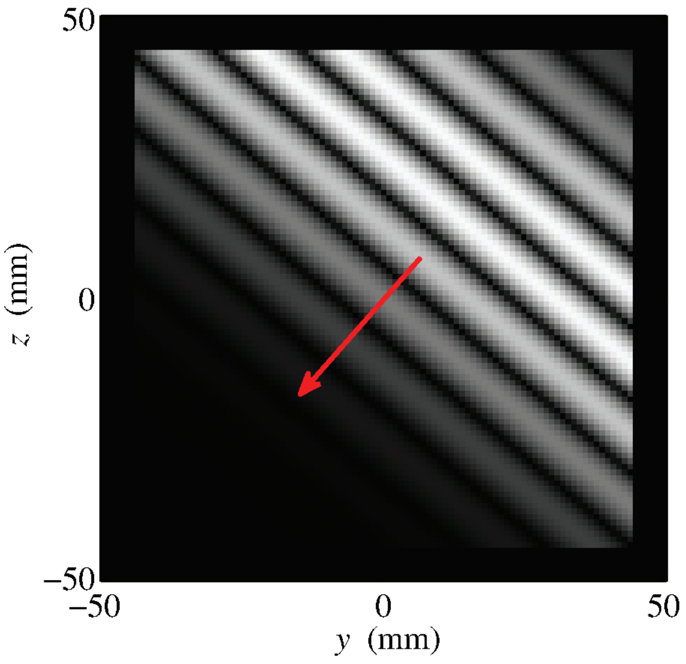


Figure 5. An electric-field snapshot of an *x*-polarized plane wave. The direction of propagation is indicated by the arrow.

9. Incident Beam Creation

For scattering problems, *Angora* can create certain types of incident beams in the simulation grid using the total-field/scattered-field (TF/SF) method [2]. All incident-beam definitions are placed inside the TFSF group.

9.1 Plane Waves

For creating plane waves, the PlaneWaves list is used. Here is a simple example:

```
TFSF:
{
 PlaneWaves:
 (
 {
 theta = 40;
 phi = 90;
 psi = 0;
 waveform_tag = "waveform1";
 }
 );
};
```

This definition will create an x-polarized plane wave propagating downward at a 40° angle with the z axis on the yz plane. The electric-field waveform of the plane wave (in V/m) is determined by the string tag `"waveform1"`, which should be defined in a `Waveforms` definition somewhere in the configuration file (see Section 8). Assuming a modulated-Gaussian waveform, a two-dimensional snapshot of the field amplitude of this plane wave on the yz plane is shown in Figure 5.

The plane wave in Figure 5 travels in free space. *Angora* uses the standard method described in [2] to compute this plane wave in the FDTD grid. If infinite planar stratification is present (see Section 5), the "incident" plane wave encompasses not only the first plane wave that impinges on the uppermost (or lowermost) layer's interface, but also the reflected and transmitted plane waves due to the layers' interfaces. Because of this, the simple approach in [2] is no longer sufficient to simulate plane-wave incidence. A more sophisticated technique, described in detail in [11], has been incorporated into *Angora*. This allows it to compute plane waves incident on an infinite planar (possibly lossy) stratification. *Angora* automatically detects the layering in the FDTD grid and chooses the appropriate TF/SF algorithm for the problem. A plane wave incident from air onto a lossy half space ($\varepsilon_r = 2.0$, $\sigma = 0.5$ S/m) is shown in Figure 6. The waveform of the incident plane wave was a Gaussian function $f(t) = \exp\left[-(t/\tau)^2/2\right]$, with $\tau = 20$ ps. In addition to the plane wave hitting the interface (at $z = 0$), the reflected and transmitted plane waves are also computed and included in the "incident" field.

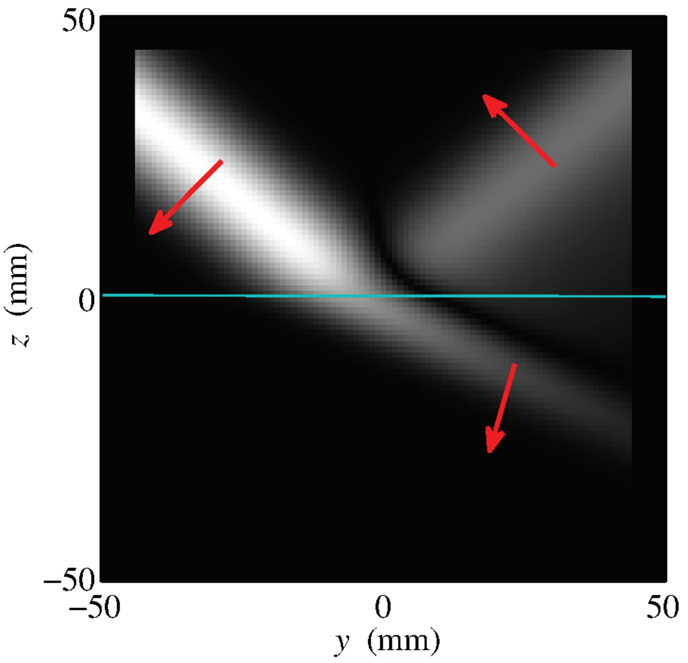


Figure 6. An electric-field snapshot of an x-polarized plane wave incident on a lossy dielectric half-space. The layer interface is indicated by the horizontal line. The propagation directions of the incident, reflected, and transmitted plane waves are shown by arrows.

9.2 Focused Hermite-Gaussian Beams

Angora can also create a Hermite-Gaussian beam (also called a TEM laser beam), focused by a lens. For this, the `FocusedLaserBeams` list is used:

```
TFSF:
{
 FocusedLaserBeams:
  (
  {
  theta = 40;
  phi = 90;
  psi = 0;
  x_order = 0;
  y_order = 0;
  waveform_tag = "waveform1";
  ap_half_angle = 80;
  back_focal_length = 0.1;
  filling_factor = 1;
  }
  );
};
```

This definition will create an x-polarized focused Gaussian beam (TEM_{00}), propagating downward at a 40° angle, with the z axis on the yz plane. The `filling_factor` variable determines the ratio of the beam's half-width to the radius of the entrance pupil of the focusing lens. The `ap_half_angle` variable specifies the half-angle of the illumination cone, and the `back_focal_length` variable specifies the focal length of the lens. Higher-order beams (TEM_{mn}) can also be defined by changing the variables `x_order` and `y_order`. The electric-field waveform of the Gaussian beam (in V/m) is determined by the string tag `"waveform1"`, which should be defined in a `Waveforms` definition somewhere in the configuration file (see Section 8). Assuming a modulated-Gaussian waveform, a two-dimensional snapshot of the field amplitude of this beam on the yz plane is shown in Figure 7.

Internally, the focused laser beam is represented as a finite summation of plane waves incident from many directions inside the illumination cone. This technique was introduced in [12] for the synthesis of a focused beam that results from illuminating a converging lens by a plane wave. The beam considered in [12] is the limiting case of a focused TEM_{00} laser beam for an infinitely overfilled aperture (i.e., as the `filling_factor` variable tends to infinity.)

10. Near-Field-to-Far-Field Transformation

Frequently, the field distribution at the far zone of a scatterer or radiator is of interest. In the FDTD, there is a technique called the near-field-to-far-field transformation (NFFFT) [2] for calculating the far-zone field (also called the radiated field) from the near-field data available in the simulation grid. This near field is commonly collected over a closed surface, surrounding the scatterer or radiator, which is called the near-field-to-far-field transformation surface. There are two main

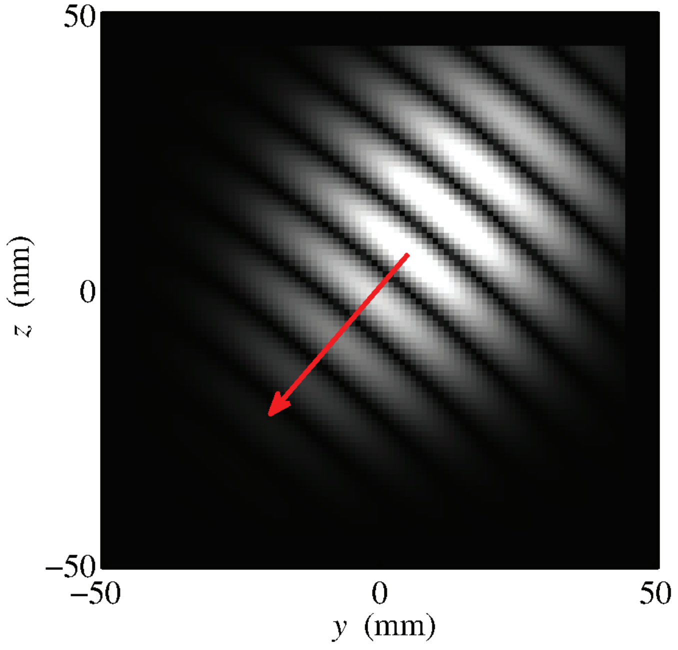


Figure 7. An electric-field snapshot of an *x*-polarized focused Gaussian beam. The direction of propagation is indicated by the arrow.

types of near-field-to-far-field transformations. In the first, the frequency (or phasor) components of the temporal Fourier spectrum of the far field are calculated at specific frequencies. In the second, the far field is obtained directly in the time domain. In the first type, there is a significant memory penalty per frequency, but a small additional computational burden per observation direction. In the second type of near-field-to-far-field transformation, the additional computational burden per observation direction is much larger; however, the memory usage is much lower. The first type (phasor-domain near-field-to-far-field transformation) is therefore desirable if very few frequencies are needed. The second (time-domain near-field-to-far-field transformation) is desirable if very few observation directions are needed. *Angora* can calculate both types of near-field-to-far-field transformations. The phasor-domain near-field-to-far-field transformation is supported for an arbitrary number of layers with any permittivity, permeability, and electric-conductivity values (see Section 5), whereas the time-domain near-field-to-far-field transformation is supported for up to three lossless planar layers. The phasor-domain near-field-to-far-field transformation for arbitrary layering was described (as an extension and generalization of the results in [13]) in [14], and the time-domain near-field-to-far-field transformation for a three-layered medium was described in [15].

10.1 Phasor-Domain NFFFT

Angora supports phasor-domain near-field-to-far-field transformations for arbitrary planar stratification with permittivity, permeability, and electric conductivity variations along the axis of symmetry.

The `PhasorDomainNFFFT` list is used to define phasor-domain NFFFTs:

```
PhasorDomainNFFFT:
(
 {
 num_of_lambdas = 50;
 lambda_min = 3e-2;
 lambda_max = 6e-2;
 lambda_spacing_type = "k-linear";
 direction_spec = "theta-phi";
 num_of_dirs_1 = 1;
 dir1_min=0.0;
 dir1_max=0.0;
 num_of_dirs_2 = 1;
 dir2_min=0.0;
 dir2_max=0.0;
 far_field_dir = "my_path";
 },
 {
 num_of_lambdas = 1;
 lambda_min = 4e-2;
 direction_spec = "theta-phi";
 num_of_dirs_1 = 360;
 dir1_min=0.0;
 dir1_max=360.0;
 num_of_dirs_2 = 1;
 dir2_min=0.0;
 dir2_max=0.0;
 far_field_dir = "my_path";
 }
);
```

In this example, two near-field-to-far-field transformations are defined in two consecutive groups. In the first group, the far field is calculated at a range of frequencies in a single direction. In the second group, the far field is calculated at a single frequency over a range of observation directions. In the first near-field-to-far-field transformation, the first four lines determine the frequencies (or free-space wavelengths) at which the far field will be calculated. Here, the far-field is calculated at 50 free-space wavelengths, spaced linearly in frequency (equivalently, in wavenumber, *k*) between 3 cm and 6 cm. In the second near-field-to-far-field transformation, lines three to nine define the directions at which the far field is to be calculated. The assignment `direction_spec="theta-phi"` says that the directions are given in terms of the spherical-coordinate angles θ and ϕ, where the *z* axis corresponds to $\theta = 0$, and the *xz* plane corresponds to $\phi = 0$. The lines that follow this assignment specify that the far field will be calculated at 360 θ values between 0° and 360°, and a single ϕ value ($\phi = 0°$). This corresponds to the far-field pattern on the *xz* plane. The last line in each group determines the directory in which the output file is placed. The output is in the standard HDF5 format, which can be inspected and modified using free software [16], or built-in *MATLAB* functions. For more information on the output of the phasor-domain near-field-to-far-field transformation, please consult the *Angora* user's manual. The far-field amplitudes are all normalized by the asymptotic *r* dependence $\exp(-jkr)/r$.

As an example, the frequency spectrum and the far-field pattern are shown in Figures 8a and 8b for the arrangement in

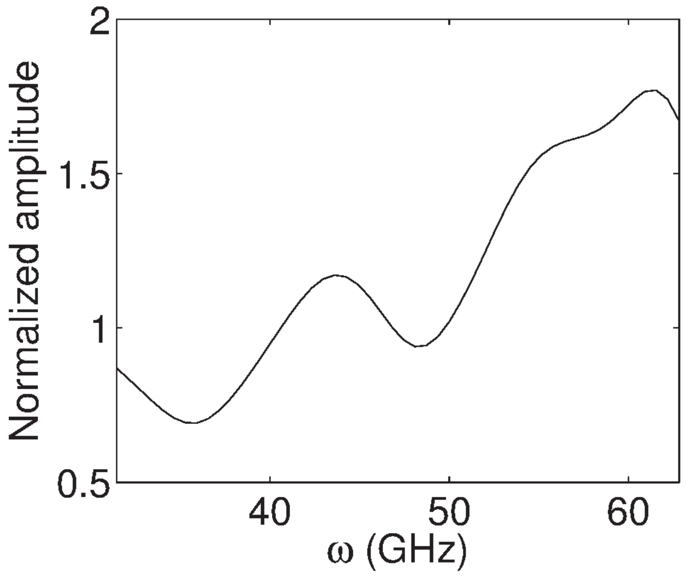


Figure 8a. The far-field amplitudes created by the Hertzian source in Figure 4: The frequency response in the $\theta = 0$ direction. The absolute value of E_θ is shown. All amplitudes were normalized by the distance, r, and the free-space amplitude created by the dipole in the $\theta = 0°$ direction.

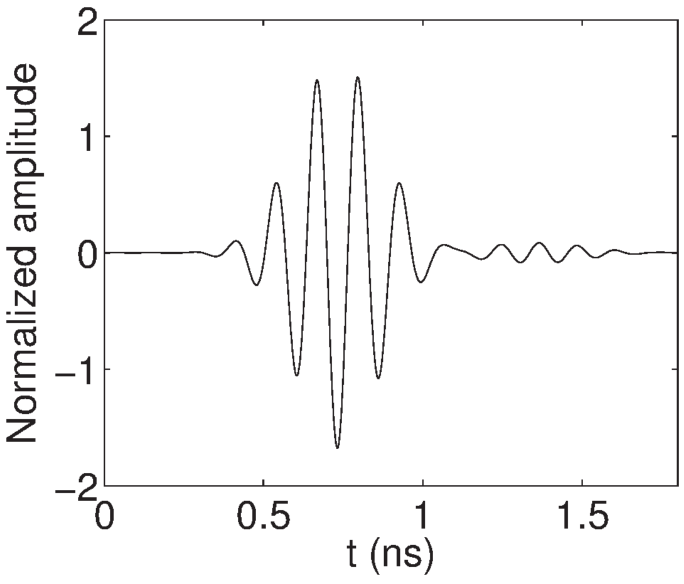


Figure 8c. The far-field amplitudes created by the Hertzian source in Figure 4: The time waveform for the θ component of the electric field at $\theta = 36°$, $\phi = 57°$. The amplitude waveform shown was time-advanced by r/c, and normalized by the maximum of the waveform created by the dipole in free space.

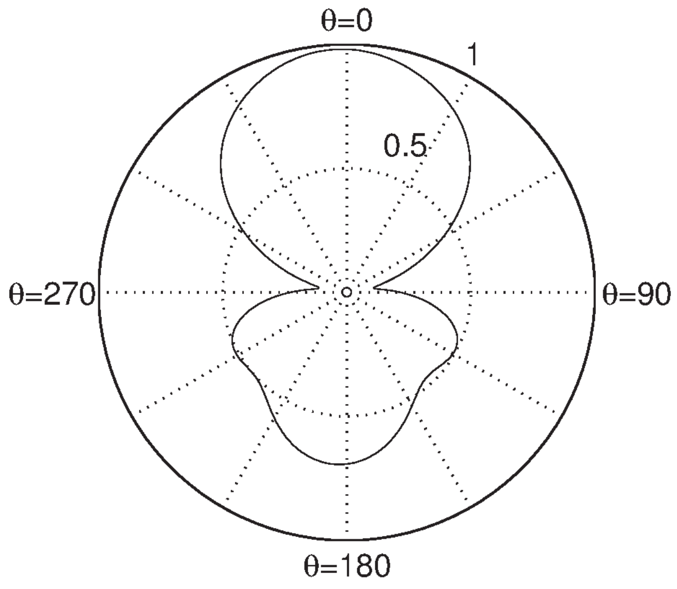


Figure 8b. The far-field amplitudes created by the Hertzian source in Figure 4: The phasor-amplitude pattern in the xz plane at $\lambda = 4$ cm. The absolute value of E_θ is shown. All amplitudes were normalized by the distance, r, and the free-space amplitude created by the dipole in the $\theta = 0°$ direction.

Figure 4 with a Hertzian dipole source at the center of the grid (see Section 8).

10.2 Time-Domain NFFFT

Angora supports time-domain near-field-to-far-field transformations for up to three lossless planar material layers with permittivity variations along the axis of symmetry. The `TimeDomainNFFFT` list is used to define phasor-domain NFFFTs:

```
TimeDomainNFFFT:
(
 {
 theta = 36;
 phi = 57;
 far_field_dir = "my_path";
 }
);
```

The `theta` and `phi` variables specify the spherical-coordinate θ and ϕ angles at which the far-field waveforms are to be calculated. The `far_field_dir` variable determines the directory in which the output file is placed. The output is in the standard HDF5 format, which can be inspected and modified using free software [16], or built-in *MATLAB* functions. For more information on the output of the time-domain near-field-to-far-field transformation, please consult the *Angora* user's manual. The far-field amplitudes are all normalized by the asymptotic r dependence $1/r$, and advanced in time by r/c.

As an example, the θ component of the electric-field waveform at $\theta = 36°$, $\phi = 57°$ is shown in Figure 8c for the arrangement in Figure 4, with a Hertzian dipole source at the center of the grid (see Section 8).

11. Optical Imaging

Angora also offers some unique features for nano-optics problems. Focused-beam illumination (see Section 9) is one of these features. Another useful feature is the ability to construct *optical images* of samples placed inside the FDTD grid. Optical images are constructed in *Angora* by merging the full-vector Maxwell's-equations-based FDTD solution with the geometrical-optics principles that are valid in the macroscopic far-field regime. The link between the microscopic FDTD domain and the macroscopic optical-component domain is provided by the near-field-to-far-field transformation (see Section 10). A detailed description of this methodology was described in [17].

Optical images are constructed using the OpticalImages list:

```
OpticalImages:
(
 {
 output_data = ["intensity_tot",
 "intensity_inc"];
 num_of_lambdas = 10;
 lambda_min = 400e-9;
 lambda_max = 700e-9;
 lambda_spacing_type = "k-linear";
 ap_half_angle = 36.87;
 magnification = 40.0;
 coll_half_space = "upper";
 image_dir = "my_path";
 }
);
```

The `output_data` array lists the desired field or intensity information in the output file. In the above example, only the total intensity and the incident intensity (the intensity created by the incident beam without any scatterers) are recorded. The next four lines specify the wavelengths at which the optical image will be synthesized. The assignments have the same meaning as those in the near-field-to-far-field transformation (NFFFT) specification (see Section 10.1). The half-angle of the collection aperture as seen from the focal point of the objective (which is by default the origin of the grid) is determined by the `ap_half_angle` variable. The `magnification` variable sets the magnification of the imaging system. The `coll_half_space="upper"` setting implies that the objective is placed above the sample (in the $+z$ direction), which corresponds to reflection microscopy. Transmission microscopy could also be simulated by setting this variable to `"lower"`. The last line specifies the directory in which the output data are placed. As with the near-field-to-far-field transformation, the output is written in HDF5 format, which can be read using freely-available tools [16] or built-in *MATLAB* functions. For more information, please consult the *Angora* user's manual.

We will now demonstrate the optical-imaging feature of *Angora* by placing a nanometer-scale sample in the shape of the AP-S logo inside the FDTD grid (see Figure 9), and calculating its optical image. The logo was read from a GIF file, and converted into a 14.62 μm × 14.18 μm × 0.4 μm dielectric embossing with relative permittivity $\varepsilon_r = 2.56$ (or refractive index $n = 1.6$). This region was then written into a file according to the format specified in Section 6, and read into the FDTD grid using a `MaterialsFromFiles` definition (see Section 6).

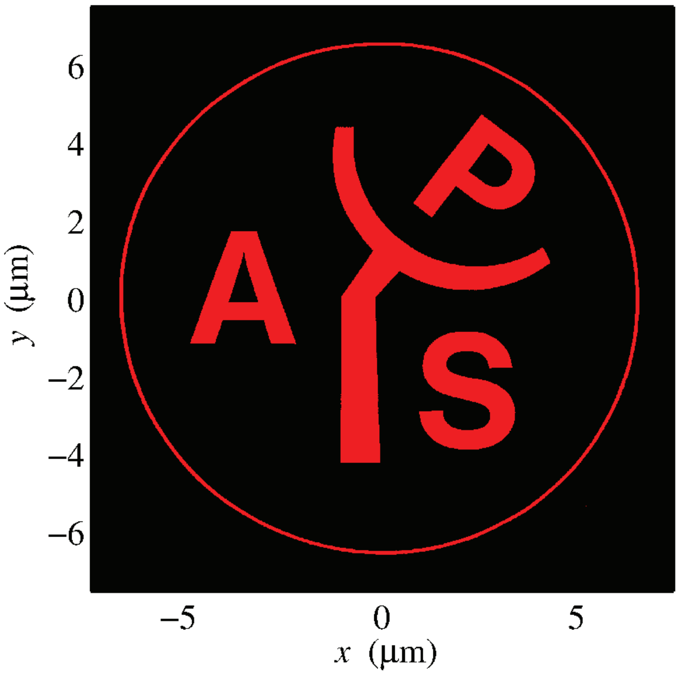


Figure 9a. The xy cross section of the geometry of the optically-imaged sample in the shape of the AP-S logo ($n = 1.6$), embossed on a glass substrate ($n = 1.5$).The embossing material is shown in red.

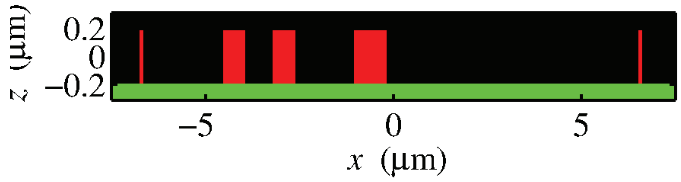


Figure 9b. The xz cross section of the sample in Figure 9a. The glass substrate is shown in green. The substrate extends to infinity in the $-z$ direction.

A glass half space ($n = 1.5$) was placed below the material region using a `MaterialSlabs` definition (see Section 5). A plane wave with a modulated Gaussian electric-field profile ($\tau = 2.12662$ fs, $f_0 = 5.88878 \times 10^{14}$ Hz) was sent onto the sample from the $+z$ direction using a `PlaneWaves` definition (see Section 9). With these parameters, the Gaussian frequency spectrum of the incident plane wave fell to -20 dB of its maximum at wavelengths of 400 nm and 700 nm, which corresponded roughly to the limits of visible light. In order to simulate un-polarized light, two orthogonal polarizations were separately sent in two consecutive runs, and the final intensities were added, since the orthogonal polarizations were independent. The optical image of this sample was calculated using the `OpticalImages` definition above. The optical intensities were all normalized by the incident intensity, which was the intensity of the plane wave reflected from the glass substrate without the embossing. In this way, the frequency dependence cancelled out, and the effective white-light response was obtained. In Figure 10a, the resulting normalized bright-field image (the intensity integrated over all frequencies) is shown in grayscale. The brightness values ranged from 0 (black) to 2 (white). In the remaining figures, the wavelength spectrum of each pixel was processed using the CIE color-matching functions [18] to obtain the color image perceived by the average human retina. In Figure 10b, the color version of the bright-field image in Figure 10a is shown. Figure 10c shows the *dark-field* image of the sample, in which only the light scattered from the sample was retained, and the incident field was completely removed from the final image. If the incident field was phase-shifted by 90° instead of being removed, the *phase-contrast* image in Figure 10d was obtained. One needed the complex phasor amplitude distribution of the image to separate the scattered and incident fields and synthesize Figures 10c and 10d. This was done by including amplitude information in the `output_data` array in the `OpticalImages` definition above. The configuration file for this example (along with the material file for the AP-S logo) can be found on the *Angora* Web site [5].

12. Field Recording

Angora can collect field values created in the grid during the FDTD simulation and save them into files. There are three types of field recording: movie recording on a cross section of the grid, line recording along a line through the grid, and field-value recording at a given point in the grid. Here, we will only describe movie recording. For the other two types, please consult the *Angora* user's manual. The settings for line and field-value recording are similar to those for movie recording, but considerably simpler. All the permittivity-profile images (such as Figure 1 and Figure 9) and the field-distribution snapshots (such as Figure 4) were produced using the movie-recording feature of *Angora*.

The `Recorder` group includes all the field-recording settings for the simulation. The `MovieRecorders` list inside this group lists the movie recorders:

```
Recorder:
{
 MovieRecorders:
  (
  {
 recorded_section = "xz";
 recorded_position = 0;
 recorded_component = "Ex";
 recording_scale = "linear";
 recording_type = "uchar1";
 movie_dir = "my_path";
  }
  );
};
```

The first line determines the cross section over which the movie is recorded. Currently, *Angora* can only record movies over cross sections perpendicular to the principal x, y, and z axes of the Cartesian grid. The second line determines the position of the recorded cross section with respect to the grid origin. The third line specifies the field component to be recorded. Currently, only the three Cartesian components of the electric or magnetic field can be recorded. Depending on the `recording_scale` variable, the output can be written as is (`"linear"`), in absolute value (`"absolute"`), or in decibels (`"dB"`). *Angora* supports two kinds of movie recording. The first, selected by setting `recording_type="uchar1"`, discretizes a given field value over a dynamic range into 256 values, and records it in a single byte. The second, selected by setting `recording_type="dbl8"`, writes the field value directly as an eight-byte `double` value. The advantage of the former type is an eight-fold gain in storage, while the second type does not introduce any loss or clipping. If the maximum value of the field over the course of the simulation can be accurately estimated beforehand, the `"uchar1"` recording can be really useful. For more information on setting the dynamic range for `"uchar1"` recording, please consult the *Angora* user's manual. The last line in the above assignments determines the directory in which the output is placed. The movie output is written in a special binary format for increased speed. This format is described in detail in the *Angora* user's manual. For the convenience of the user, a *MATLAB* script named `angora_movie.m` is distributed with the *Angora* package. This script reads an *Angora* movie file and displays it as a *MATLAB* movie. It can also save the movie in `AVI` format.

13. Future Implementation

Angora is licensed under the GNU Public License (GPL), and has been envisioned as a community project. Contributions are therefore highly welcome. Please contact help@angorafdtd.org for questions, comments, and suggestions. For bug reports, contact bugs@angorafdtd.org.

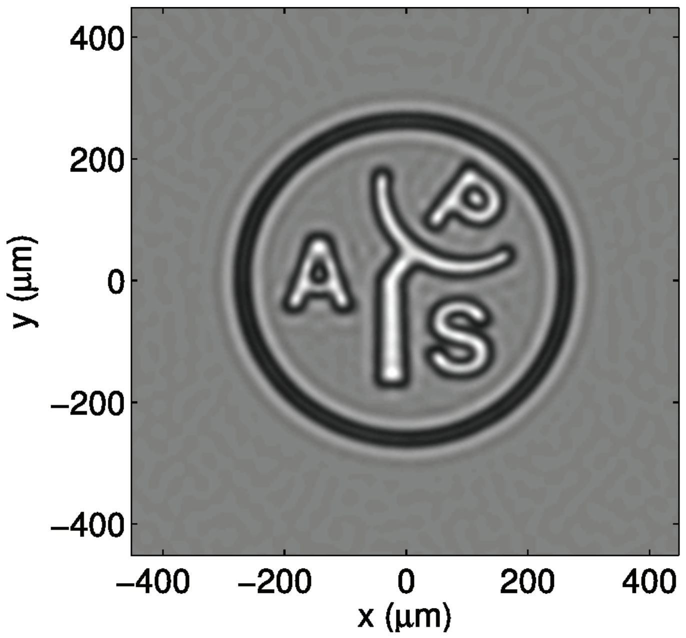


Figure 10a. Optical images of the structure shown in Figure 9: A bright-field intensity image, normalized by the reflection from the glass half-space. The grayscale limits were from 0 (black, low intensity) to 2 (white, high intensity).

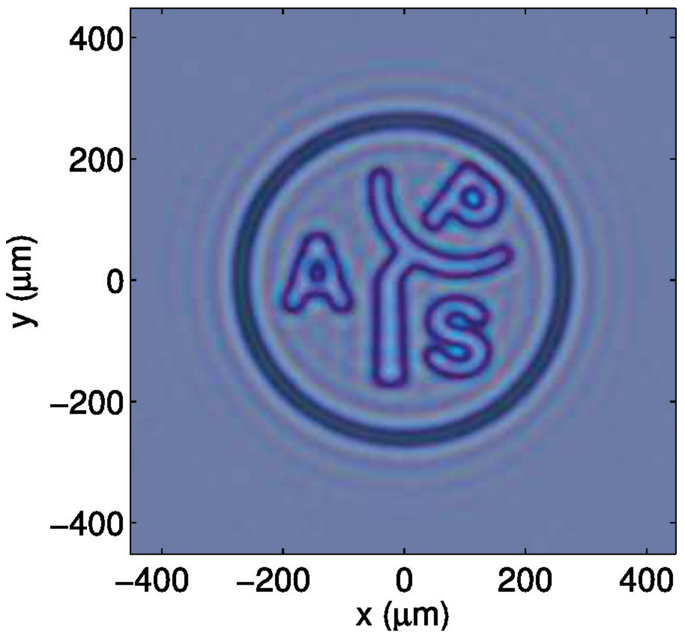


Figure 10b. Optical images of the structure shown in Figure 9: A bright-field image. The spectrum at each pixel was reduced to three values that corresponded to the responses of the three cones in the human retina. The conversion was done using the CIE color-matching functions [18].

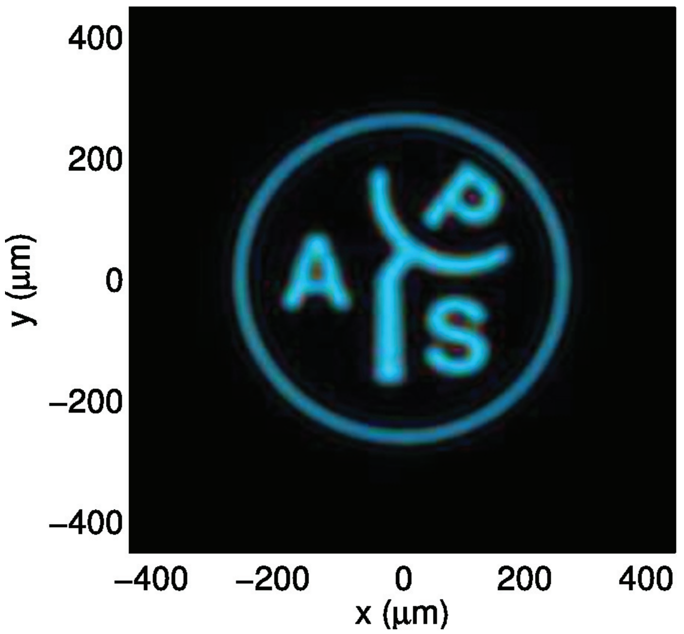


Figure 10c. Optical images of the structure shown in Figure 9: A dark-field image. The spectrum was processed as in Figure 10b.

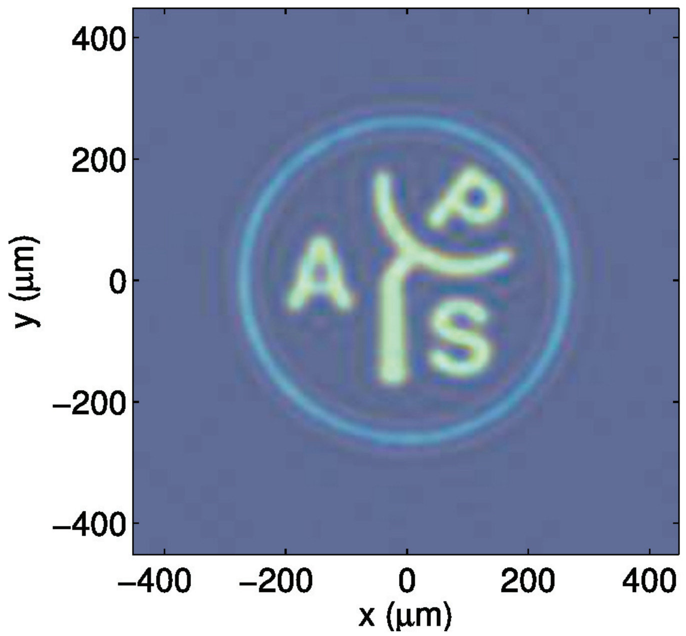


Figure 10d. Optical images of the structure shown in Figure 9: A phase-contrast image. The spectrum was processed as in Figure 10b.

14. Acknowledgment

This work was supported by NIH grants R01EB003682 and R01CA128641.

15. References

1. K. S. Yee, "Numerical Solution of Initial Boundary Value Problems Involving Maxwell's Equations in Isotropic Media," *IEEE Transactions on Antennas and Propagation*, **AP-14**, 3, May 1966, pp. 302-307.

2. A. Taflove and S. C. Hagness, *Computational Electrodynamics: The Finite-Difference Time-Domain Method, Third Edition*, Norwood, MA, Artech House, 2005.

3. Wikipedia, "Finite-Difference Time-Domain Method," 2012, accessed April 2012, available at http://en.wikipedia.org/wiki/Finite-difference_time-domain_method.

4. I. R. Capoglu, "Angora: A Free Software Package for Finite-Difference Time-Domain (FDTD) Electromagnetic Simulation," 2012, accessed April 2012, available at http://www.angorafdtd.org.

5. I. R. Capoglu, "Configuration Files for the APS Magazine Article," 2012, accessed April 2012, available at http://www.angorafdtd.org/aps_mag.

6. "The OpenMPI Project," accessed August 2012, available at http://www.open-mpi.org.

7. M. A. Lindner, "libconfig: A Library for Processing Structured Configuration Files," 2011, accessed April 2012, available at http://www.hyperrealm.com/libconfig.

8. I. R. Capoglu, "Angora User's Manual," 2012, accessed April 2012, available at http://www.angorafdtd.org/doc/ angora.html.

9. J. A. Roden and S. D. Gedney, "Convolution PML (CPML): An Efficient FDTD Implementation of the CFD-PML for Arbitrary Media," *Microwave and Optical Technology Letters*, **27**, 5, December 2000, pp. 334-9.

10. P. Guttorp and T. Gneiting, "On the Whittle-Matérn Correlation Family," NRCSE, Seattle, WA, Tech. Rep. 080, 2005.

11. I. R. Capoglu and G. S. Smith, "A Total-Field/Scattered-Field Plane-Wave Source for the FDTD Analysis of Layered Media," *IEEE Transactions on Antennas and Propagation*, **AP-56**, 1, January 2008, pp. 158-169.

12. I. R. Capoglu, A. Taflove, and V. Backman, "Generation of an Incident Focused Light Pulse in FDTD," *Optics Express*, **16**, 23, November 2008, pp. 19208-19220.

13. K. Demarest, Z. Huang, and R. Plumb, "An FDTD Near-to-Far-Zone Transformation for Scatterers Buried in Stratified Grounds," *IEEE Transactions on Antennas and Propagation*, **AP-44**, 8, August 1996, pp. 1150-7.

14. I. R. Capoglu, A. Taflove, and V. Backman, "A Frequency-Domain Near-Field-to-Far-Field Transform for Planar Layered Media," *IEEE Transactions on Antennas and Propagation*, **AP-60**, 4, April 2012, pp. 1878-1885.

15. I. R. Capoglu, *Techniques for Handling Multilayered Media in the FDTD Method*, PhD dissertation, Georgia Institute of Technology, Atlanta, GA, 2007; available at http://goo.gl/7gGN7.

16. HDF5 Group, "HDF5 Tools," accessed April 2012, available at http://www.hdfgroup.org/products/hdf5_tools.

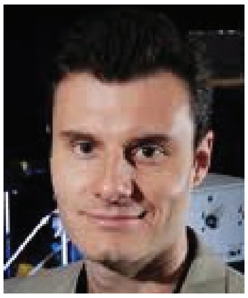
17. I. R. Capoglu, J. D. Rogers, A. Taflove, and V. Backman, "The Microscope in a Computer: Image Synthesis from Three-Dimensional Full-Vector Solutions of Maxwell's Equations at the Nanometer Scale," *Progress in Optics*, April 2012 (to appear in **57**).

18. M. D. Fairchild, *Color Appearance Models*, New York, Wiley, 2005.

Introducing the Feature Article Authors

İlker R. Çapoğlu received the PhD in Electrical and Computer Engineering from the Georgia Institute of Technology, Atlanta, in 2007. He has since been employed as a postdoctoral fellow in the Biomedical Engineering Department of Northwestern University, Evanston, IL. His research interests are time-domain methods in numerical electromagnetics, numerical modeling of electromagnetic wave propagation in multilayered media, numerical electromagnetic simulation of optical systems, and numerical simulation of the scattering of light from inhomogeneous and random media.

Allen Taflove is a Professor of Electrical Engineering and Computer Science at Northwestern University, Evanston, IL. Since 1972, he has developed fundamental theoretical approaches, algorithms, and applications of Finite-Difference Time-Domain (FDTD) computational solutions of Maxwell's equations. He coined the descriptors "finite-difference time-domain" and "FDTD" in a 1980 IEEE paper, and in 1990 was the first person to be named an IEEE Fellow in the FDTD technical area. In 2002, he was named by the Institute of Scientific Information (ISI) to its original listing of the most-cited researchers worldwide, as published in ISIHighlyCited.com. To date, his 133 journal papers and four FDTD books have received a total of more than 12,000 citations according to the ISI Web of Science, and the exact phrase "finite difference time domain" has appeared in over 44,000 articles according to Google Scholar. In May 2010, Nature Milestones: Photons recognized Prof. Taflove as one of the two principal pioneers of numerical methods for solving Maxwell's equations.



Vadim Backman is a Professor of Biomedical Engineering and the Program Leader of Cancer and Physical Sciences at the Robert H. Lurie Comprehensive Cancer Center of Northwestern University, Evanston, IL. Dr. Backman has 136 peer-reviewed publications and book chapters, and 16 patents on biological optical imaging. He is an Editor of *Biomedical Applications of Light Scattering* (McGraw Hill, 2009), and has been a Chair of the Conference on Biomedical Applications of Light Scattering since 2007. He has developed and taught classes on fundamentals of microscopy and advanced physical and applied optics.