



*Simulation of Photo-Sensitive Devices
with FDTD Method*

CROSLIGHT
Software Inc.

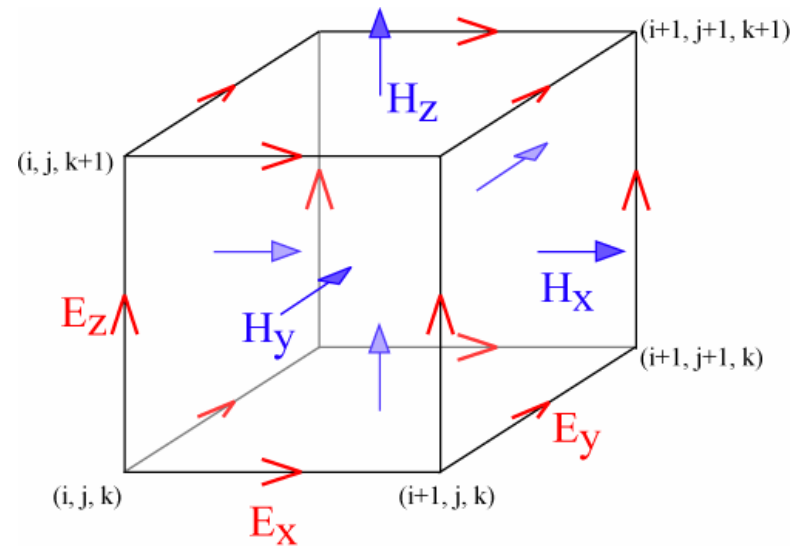
Copyright 2008 Crosslight Software Inc.
www.crosslight.com

What is FDTD method?

- FDTD = Finite Difference Time Domain
- FDTD method solves Maxwell's equations on Yee lattice

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} - 4\pi \vec{j}_m$$
$$\nabla \times \vec{B} = \frac{\partial \vec{E}}{\partial t} + 4\pi \vec{j}_e$$

Maxwell's equation



Yee lattice

http://en.wikipedia.org/wiki/Finite-difference_time-domain_method

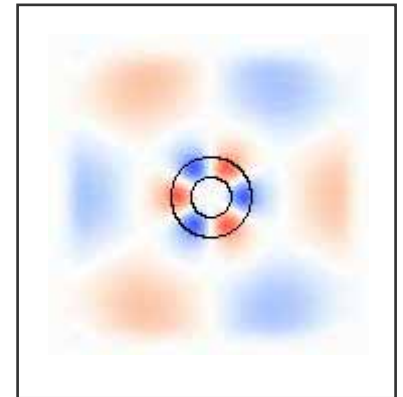
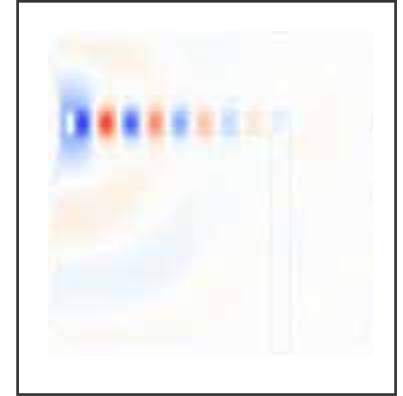


Applications of FDTD method

- Photo-detectors with submicron fine structure
 - LEDs and Lasers with textured surface
 - Solar cells
 - Photonic crystals
 - Waveguide analysis
 - Analysis of microwave circuits and antennas
- etc ...

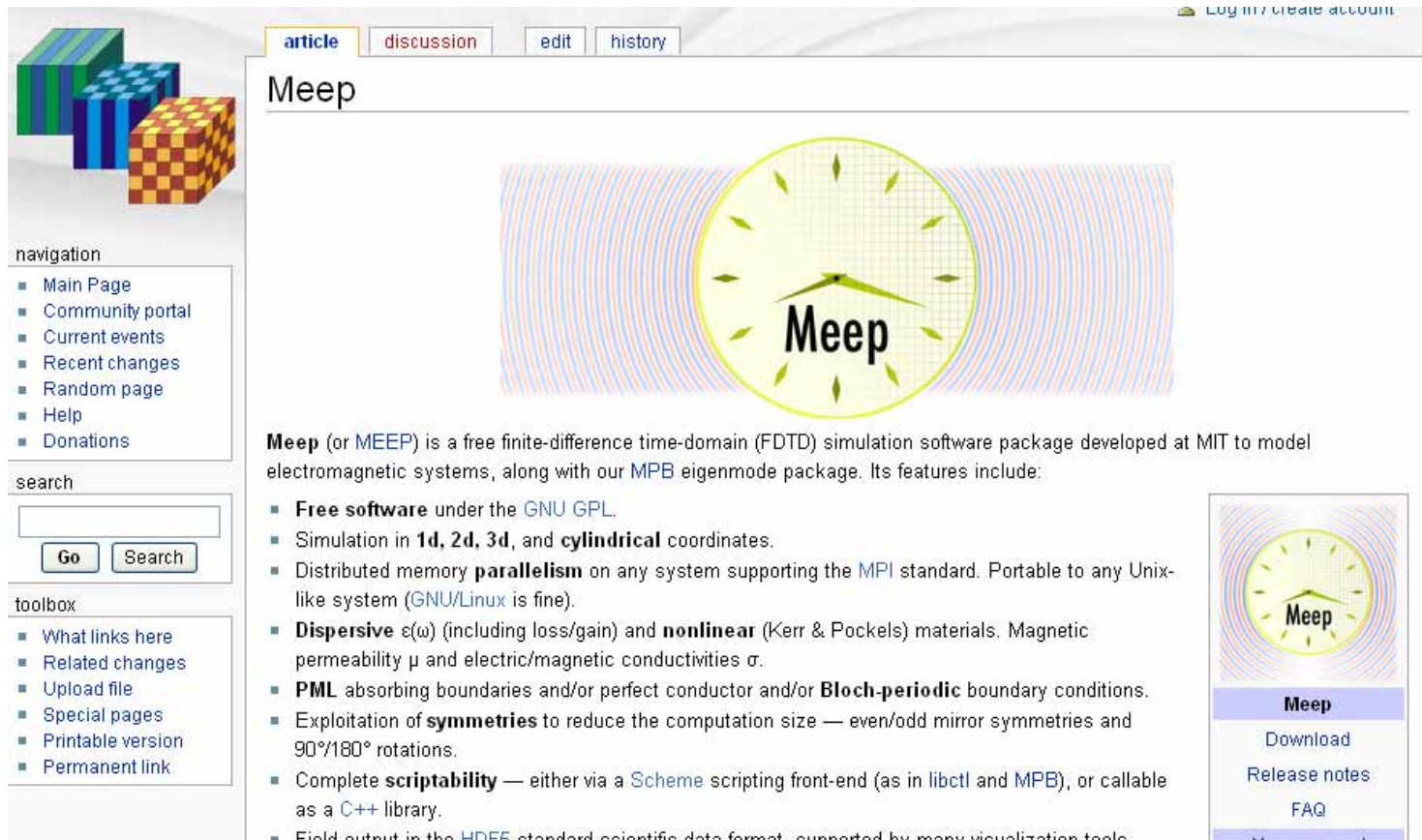
Advanced Feature of FDTD Method

- Capable of tracking time evolution of field pattern.
- Fourier transformation of field can yield transmission/reflection spectrum.
- Single run of simulation can obtain wide range of response spectrum.
- Simple, robust and numerically stable.




http://ab-initio.mit.edu/wiki/index.php/Meep_Tutorial

FDTD Simulation Software "MEEP" developed at MIT



article discussion edit history

Meep



Meep (or **MEEP**) is a free finite-difference time-domain (FDTD) simulation software package developed at MIT to model electromagnetic systems, along with our **MPB** eigenmode package. Its features include:

- **Free software** under the [GNU GPL](#).
- Simulation in **1d**, **2d**, **3d**, and **cylindrical** coordinates.
- Distributed memory **parallelism** on any system supporting the [MPI](#) standard. Portable to any Unix-like system ([GNU/Linux](#) is fine).
- **Dispersive** $\epsilon(\omega)$ (including loss/gain) and **nonlinear** (Kerr & Pockels) materials. Magnetic permeability μ and electric/magnetic conductivities σ .
- **PML** absorbing boundaries and/or perfect conductor and/or **Bloch-periodic** boundary conditions.
- Exploitation of **symmetries** to reduce the computation size — even/odd mirror symmetries and 90°/180° rotations.
- Complete **scriptability** — either via a [Scheme](#) scripting front-end (as in [libctl](#) and [MPB](#)), or callable as a C++ library.
- Field output in the [HDF5](#) standard scientific data format — supported by many visualization tools.

navigation


- Main Page
- Community portal
- Current events
- Recent changes
- Random page
- Help
- Donations

search

Go Search

toolbox

- What links here
- Related changes
- Upload file
- Special pages
- Printable version
- Permanent link



Meep

Download

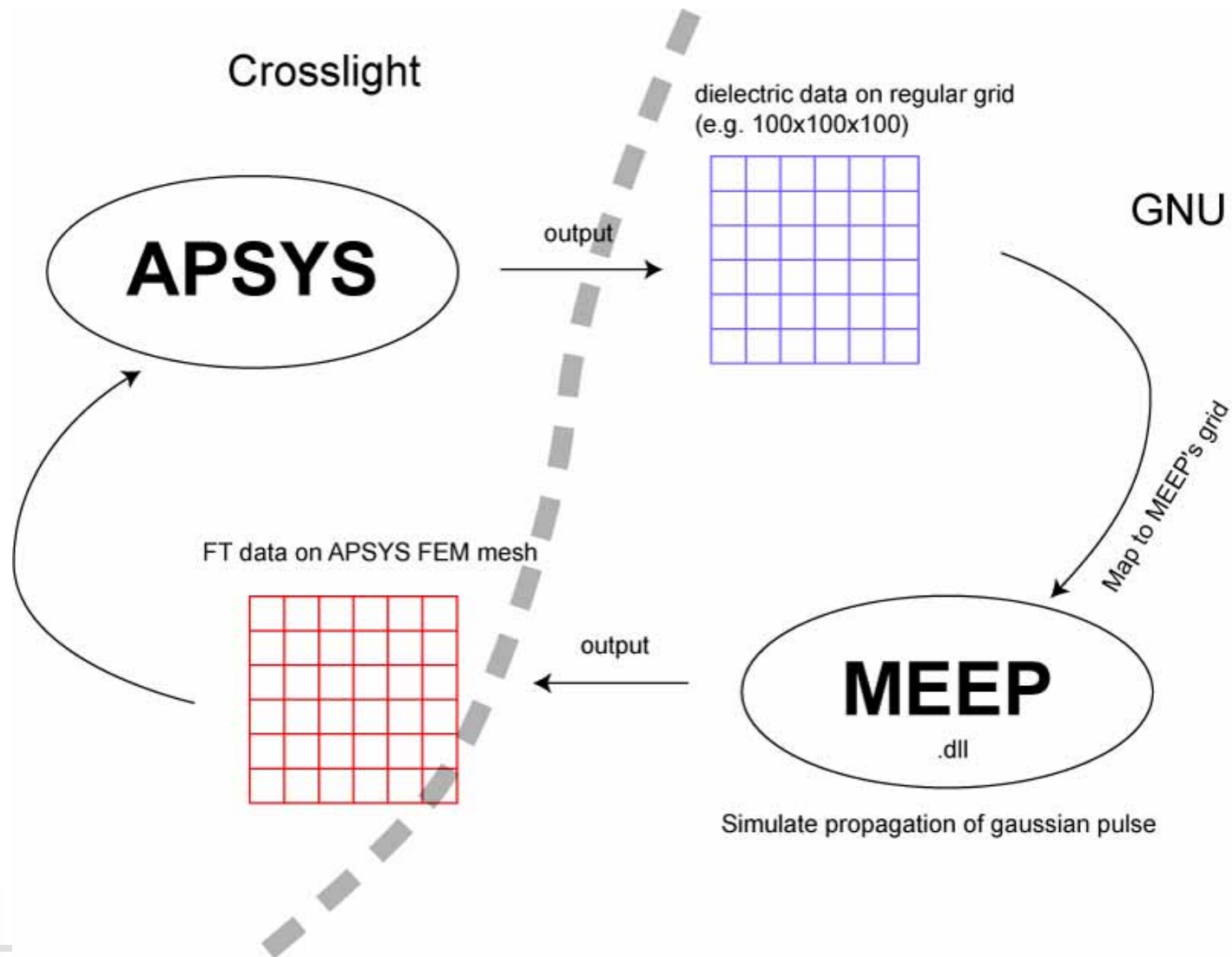
Release notes

FAQ

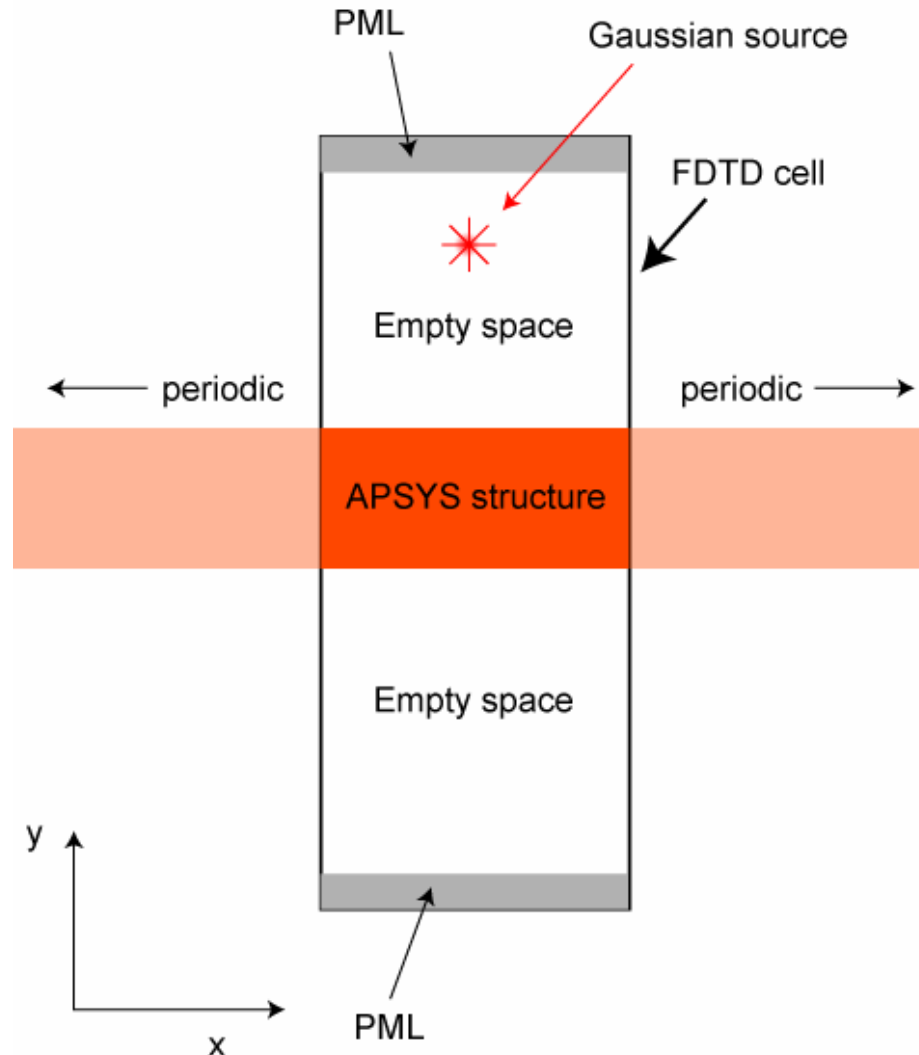
Meep manual

<http://ab-initio.mit.edu/wiki/index.php/Meep>

Interface between APSYS and MEEP



Typical configuration of APYSY-FDTD simulation





Role of PML

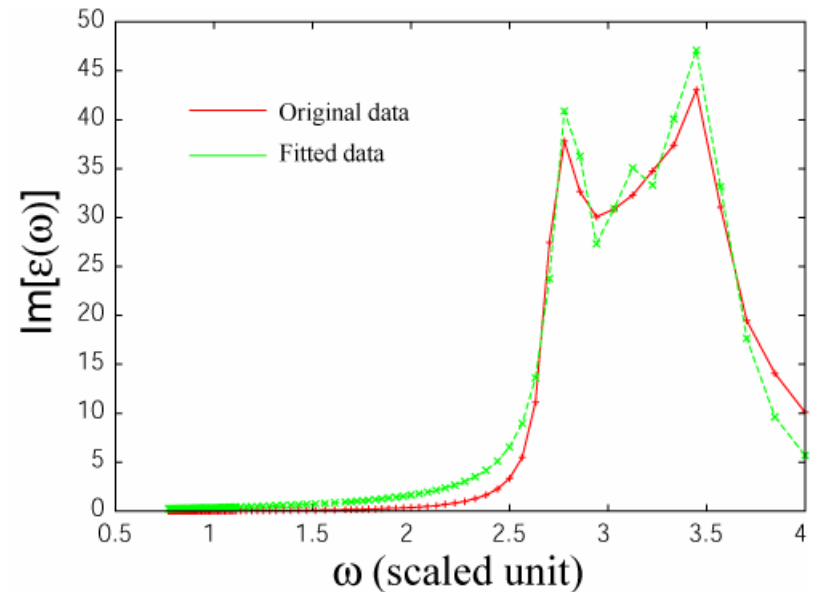
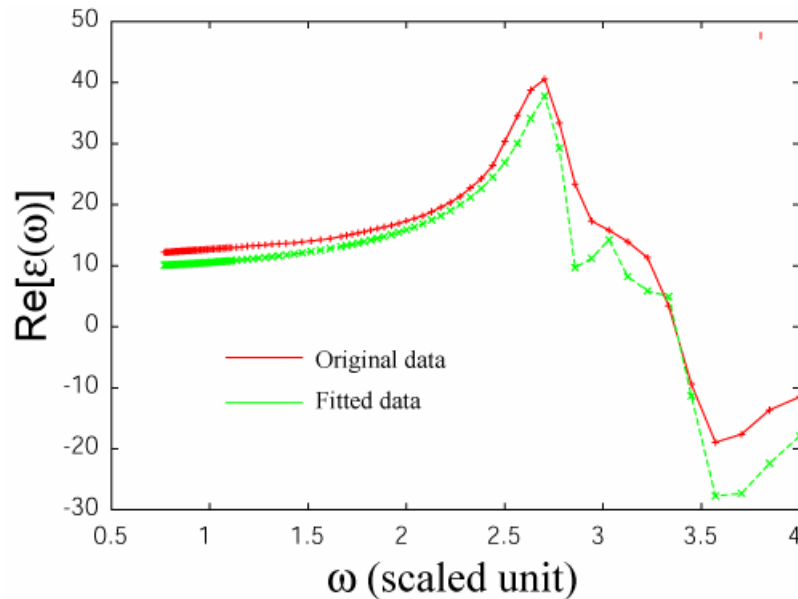
- **PML** = **P**erfectly **M**atched **L**ayer
- **PML** is one of the absorbing boundary conditions.
- **PML** absorbs electromagnetic waves. There is no reflected wave.
- Ideal for simulating open boundaries.

Material Dispersion

- Material dispersion becomes important in photo-sensitive devices, where photo-carriers are generated by absorption of light.
- In **MEEP**, material dispersion is expressed by a sum of harmonic oscillators.

$$\varepsilon(\omega, \mathbf{x}) = \varepsilon_{\infty}(\mathbf{x}) + \sum_n \frac{\sigma_n(\mathbf{x}) \cdot \omega_n^2 \Delta \varepsilon_n}{\omega_n^2 - \omega^2 - i\omega\gamma_n}$$

Example of fitting result



Fitting result of dielectric function of a-Si by two oscillator terms.



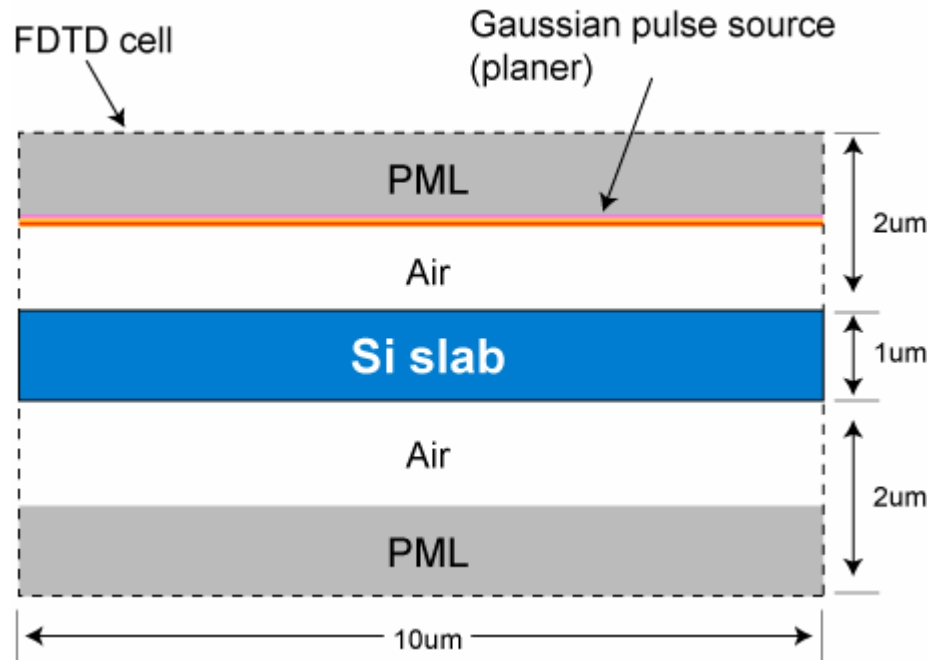
Limitations of FDTD method

- Grid spacing should be $\sim \lambda/10$.
- According to Courant's stability condition, time step Δt becomes small when **FDTD** grid spacing becomes small.
- In 3-D simulation, simulation time scales like N^4 , and required memory size scales like N^3 .
- Application is restricted to relatively small size.

APSYS-FDTD Simulation Example 1

2-D Silicon slab

- Configuration of **FDTD** simulation



Settings for FDTD

- **fddt_source** statement is used to specify position, size and component of gaussian source pulse.

```
fddt_source component=Ex &&  
  center=(5.0 2.0 0.0) size = (10.0 0.0 0.0)  
fddt_source component=Ey &&  
  center=(5.0 2.0 0.0) size = (10.0 0.0 0.0)  
fddt_source component=Ez &&  
  center=(5.0 2.0 0.0) size = (10.0 0.0 0.0)
```

Settings for FDTD (cont.)

- **fdd_model** statement is used to configure **FDTD** simulation.

```
fdd_model export_var = density wavel_range = [0.55,0.65] &&  
PML_thickness = 1 boundary_type = [1,0,1] &&  
buffer_y = [2,2] nb_wavel = 10 nb_mesh = [20,150,0] &&  
extra_time = 0 auto_dt = 20 auto_dt2 = 5 auto_finish = yes &&  
watch_point1 = [5,0.5,0]
```

Here, the duration of FDTD simulation is determined automatically by measuring magnitude of electric field.

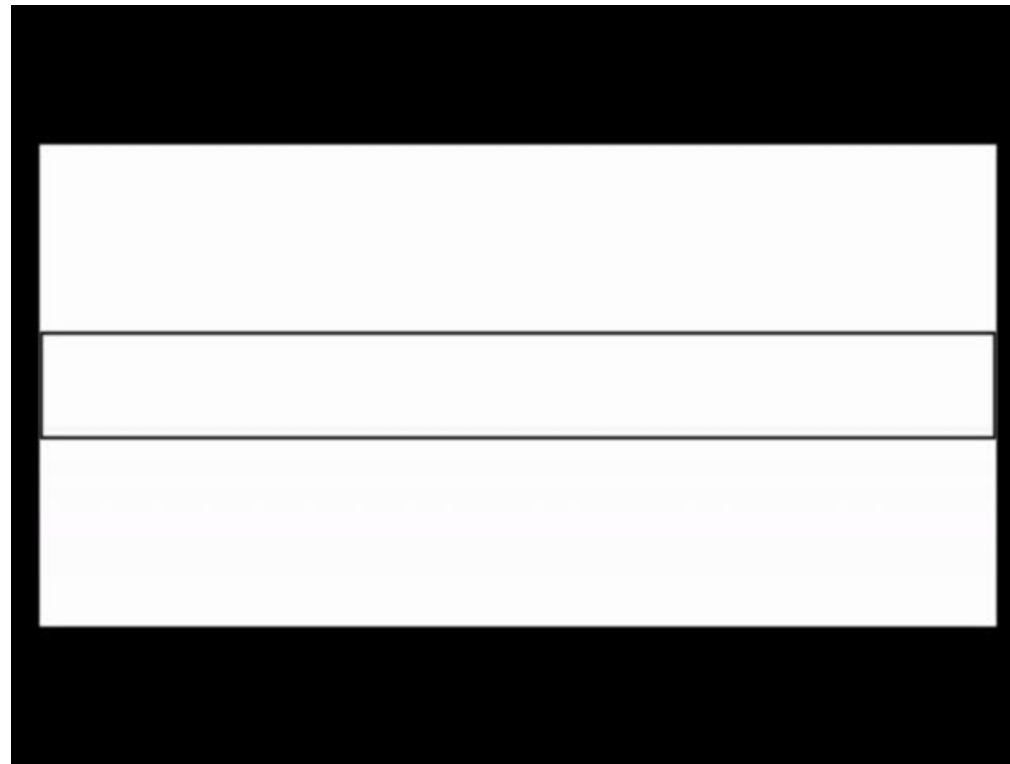
Settings for FDTD (cont.)

- **fddt_dispersion** statement is used to give material dispersion for particular material.

```
fddt_dispersion mater=1 order=2 &&  
eps0=2.33019 &&  
omega1=2.83138 gamma1=0.24013 delta_eps1=2.39085 &&  
omega2=3.38094 gamma2=0.60544 delta_eps2=7.41185
```

$$\varepsilon(\omega, \mathbf{x}) = \varepsilon_{\infty}(\mathbf{x}) + \sum_n \frac{\sigma_n(\mathbf{x}) \cdot \omega_n^2 \Delta \varepsilon_n}{\omega_n^2 - \omega^2 - i\omega\gamma_n}$$

Propagation of Gaussian pulse

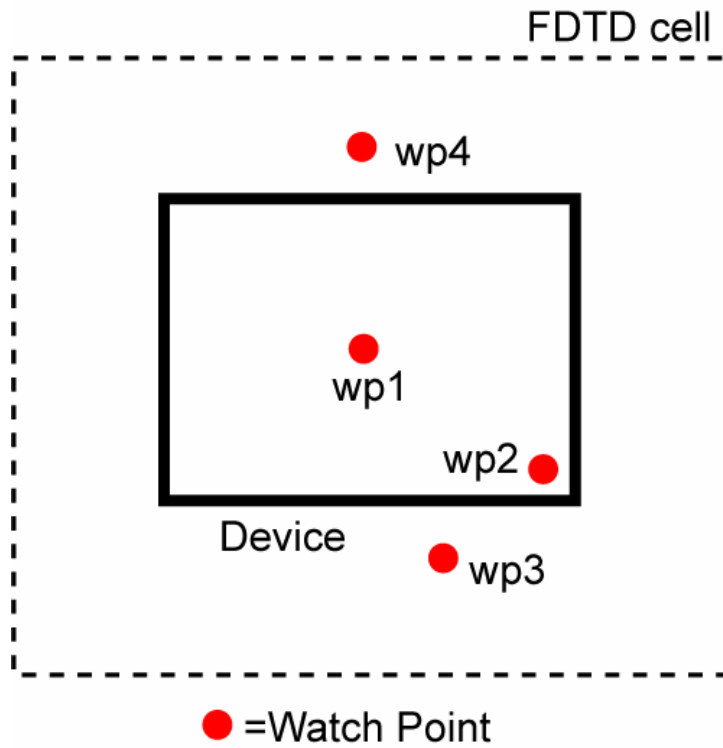


Silicon

Click the picture to animate

How long should we run FDTD?

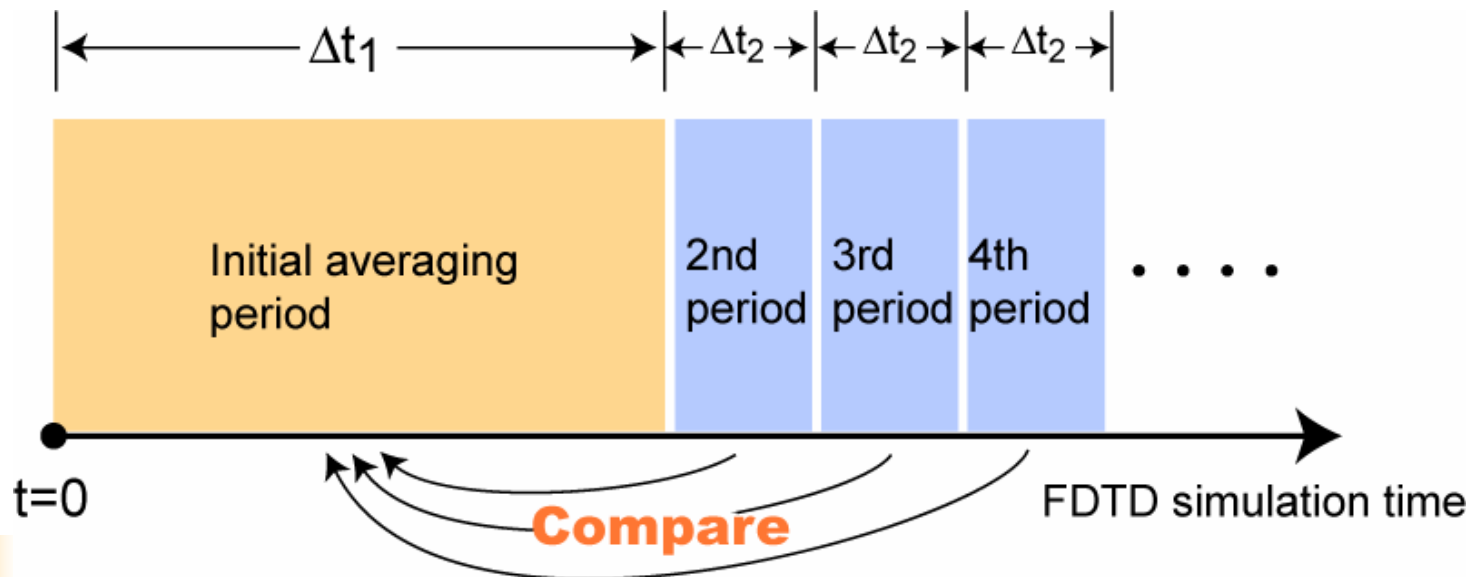
- Crosslight implemented “Watch and compare” method to judge if wave is fully decayed.



Watch point can be placed at any location.

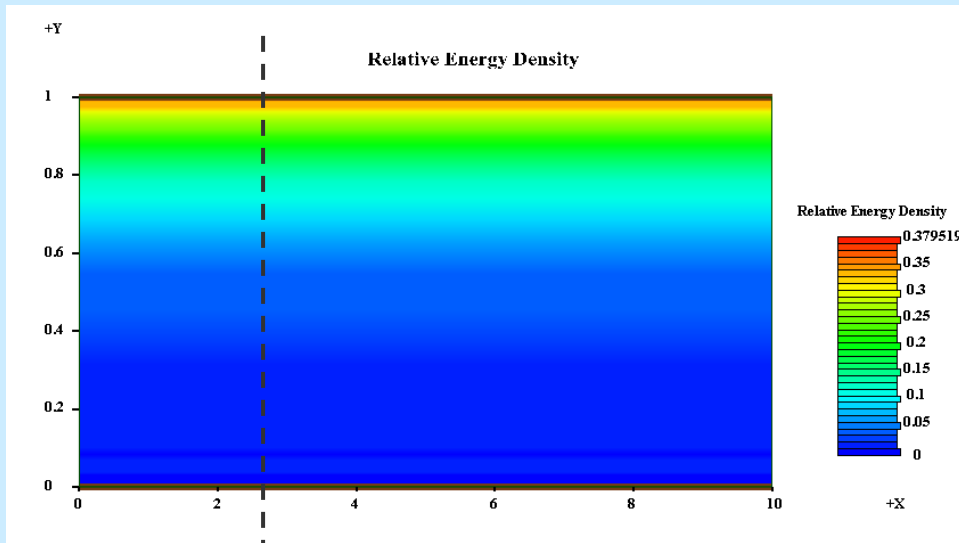
How long should we run FDTD? (cont.)

- Magnitude of electric field at each watch point is averaged over certain time duration.
- Decay of electric field is measured by comparing current averaged value and initial averaged value of electric field magnitude.

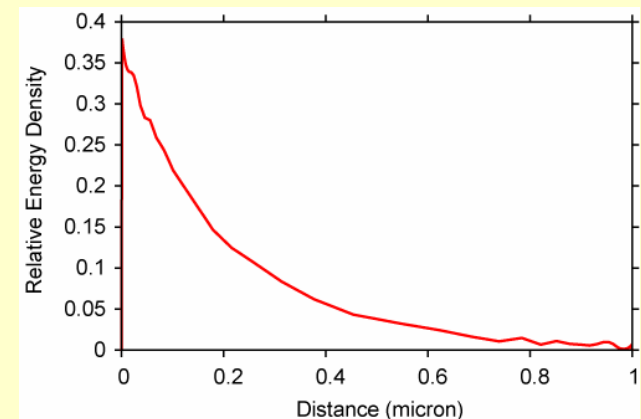


Δt_1 should be long enough so as to catch propagating wave at every watch points.

Relative energy density

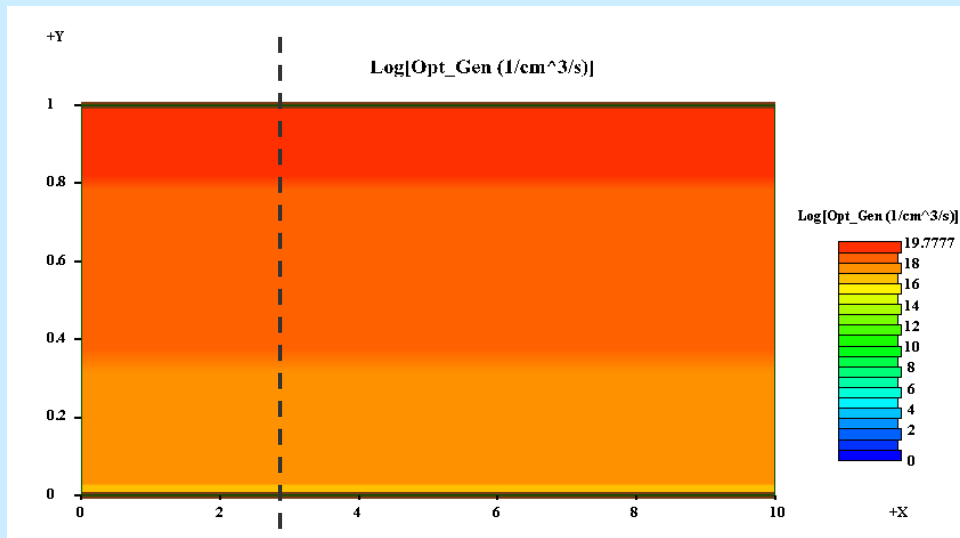


1D slice

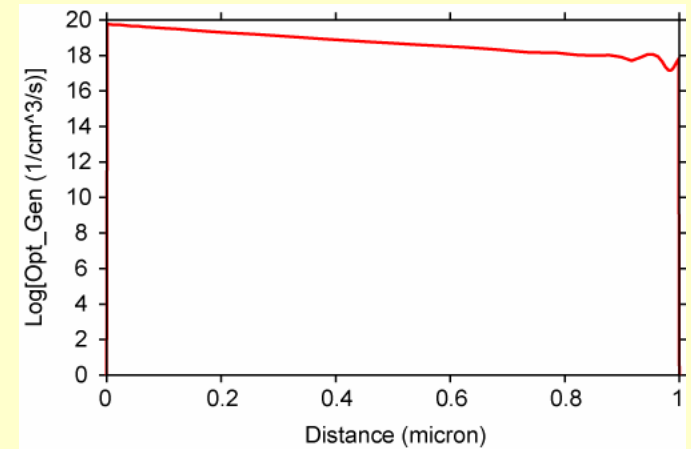


Optical energy decreases exponentially along incident path.

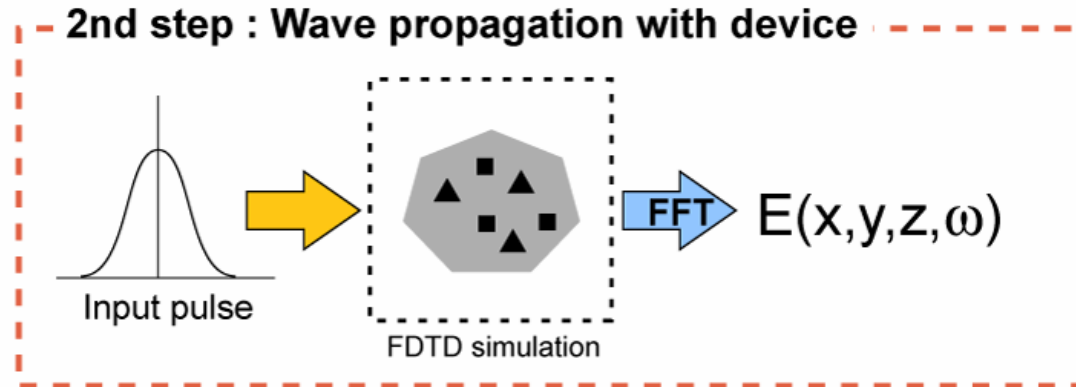
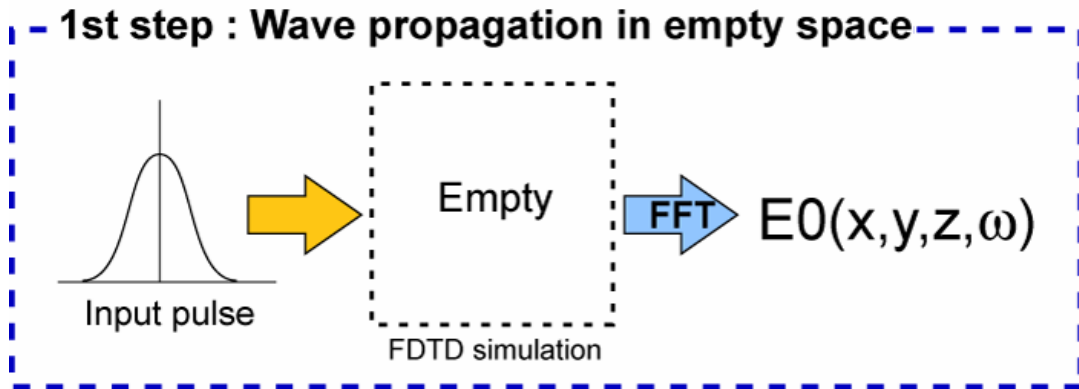
Optical generation rate (log scale)



1D slice



How could we obtain spectrum data of relative energy density out of FDTD?

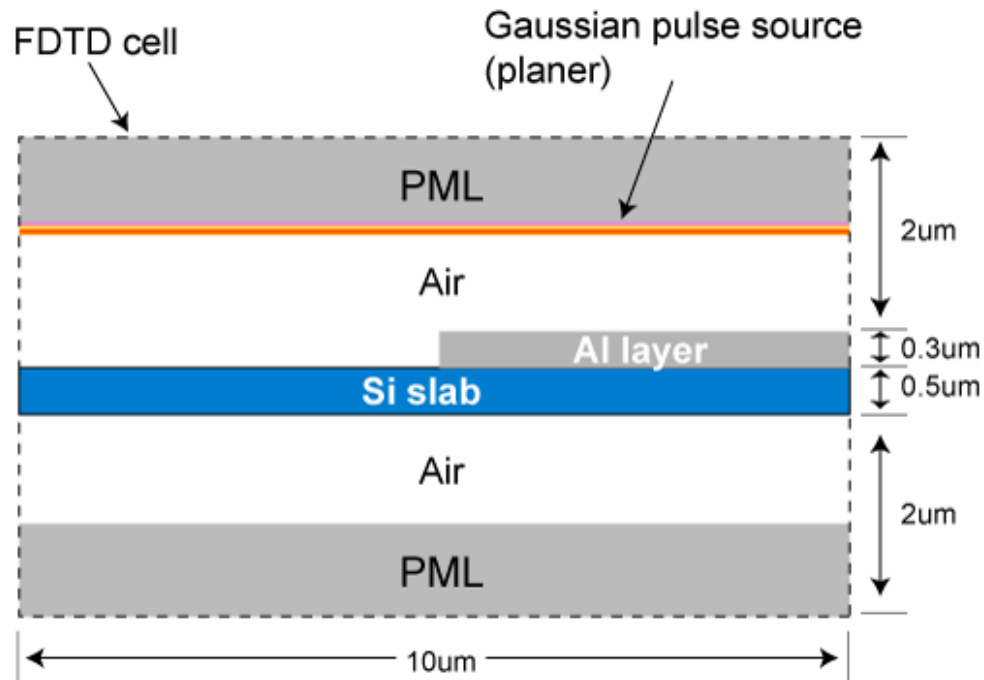


$$\frac{|E(x,y,z,\omega)|^2}{|E_0(x,y,z,\omega)|^2} \sim D(x,y,z,\omega)$$

APSYS-FDTD Simulation Example 2

2-D Silicon slab half-covered by Al layer

- Configuration of **FDTD** simulation

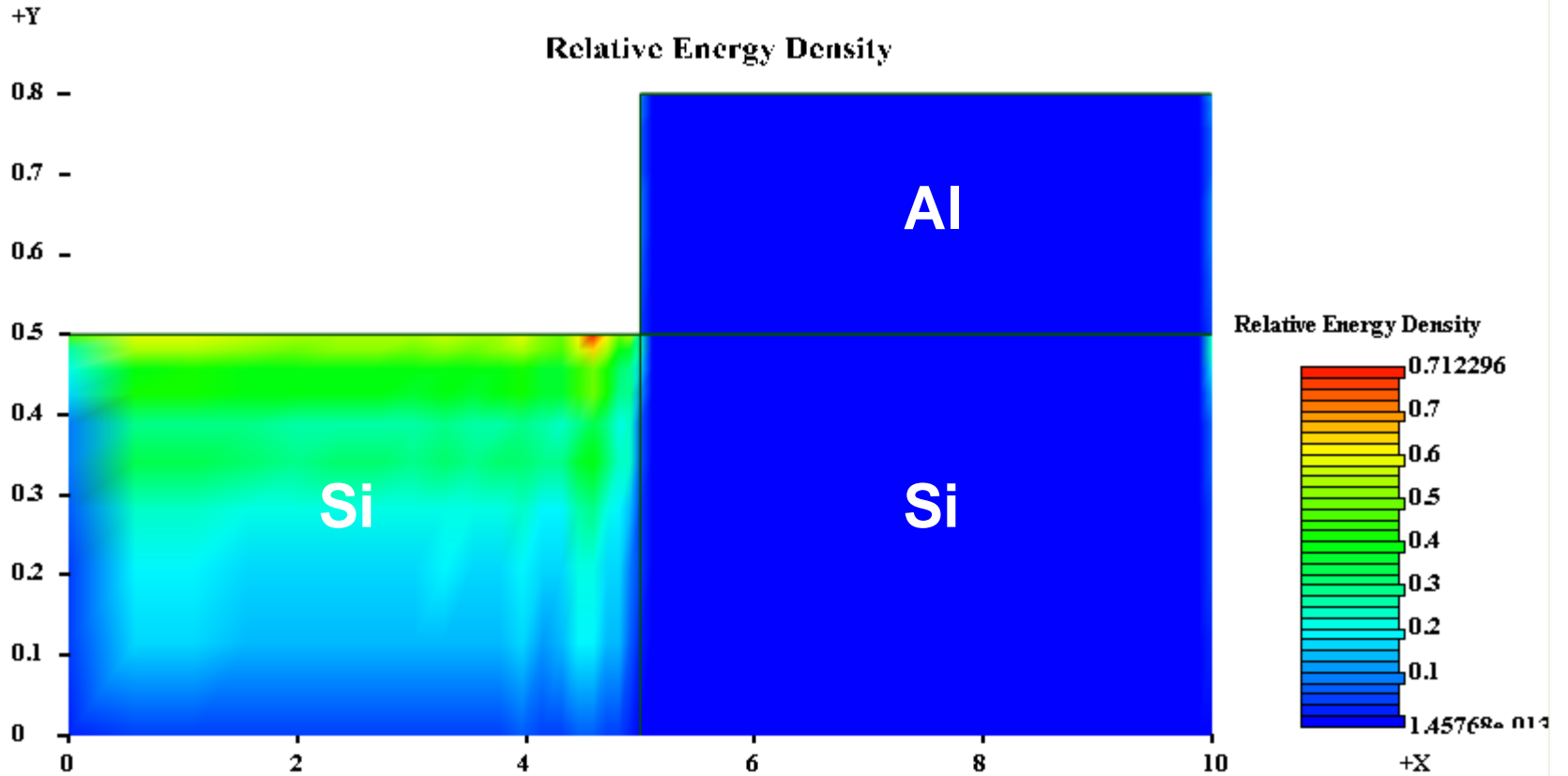


Propagation of Gaussian pulse

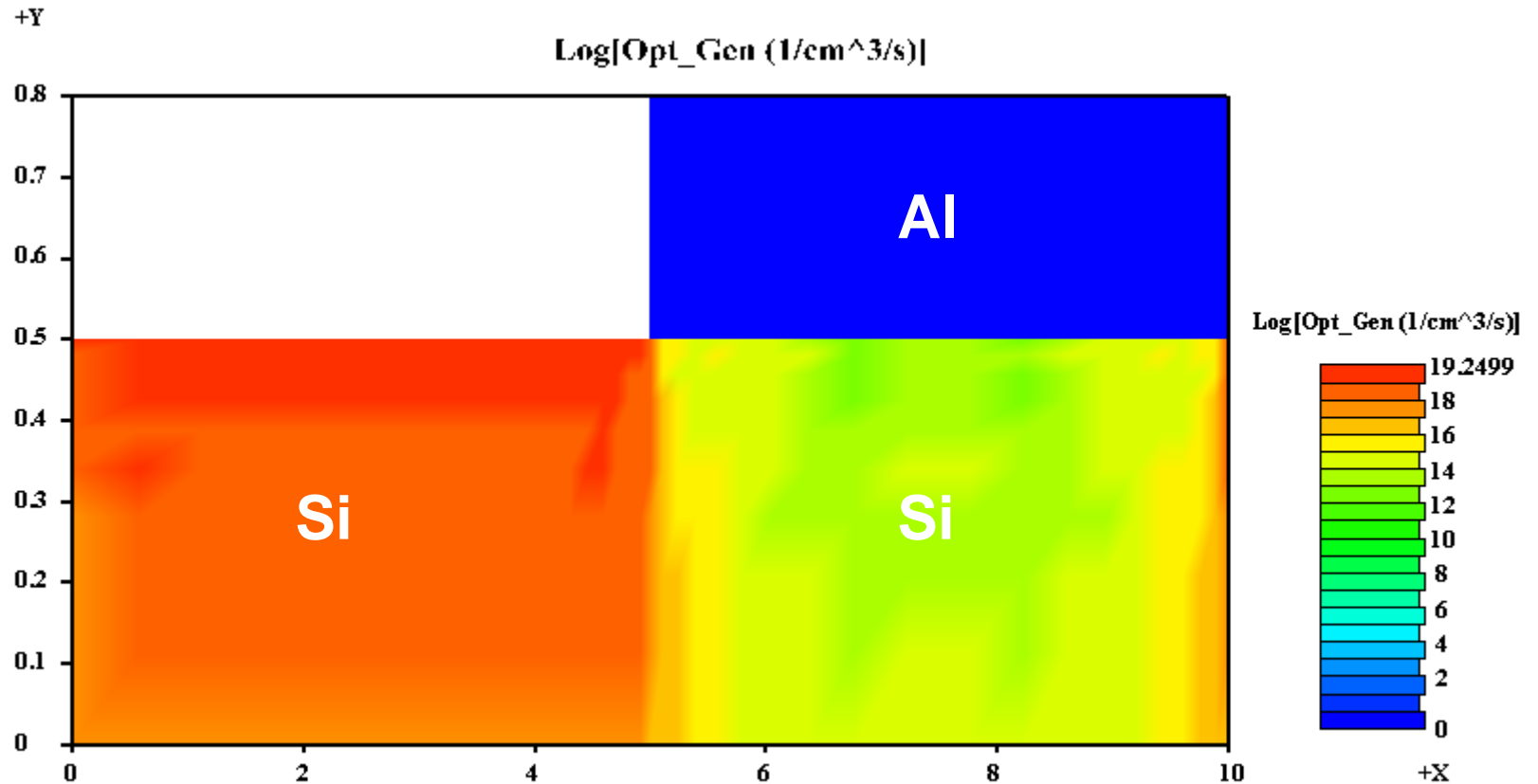


Click the picture to animate

Relative energy density

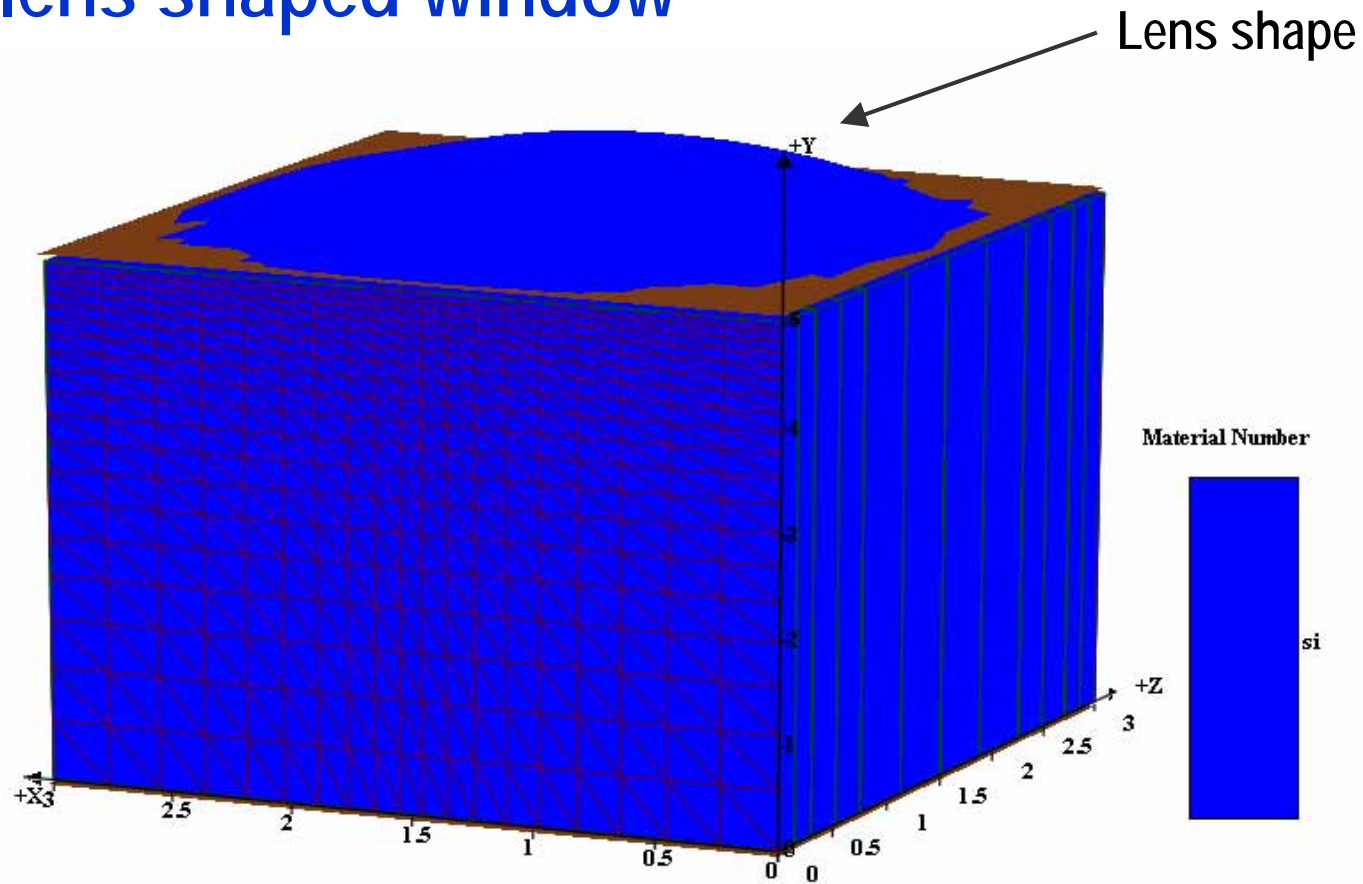


Optical generation rate (log scale)

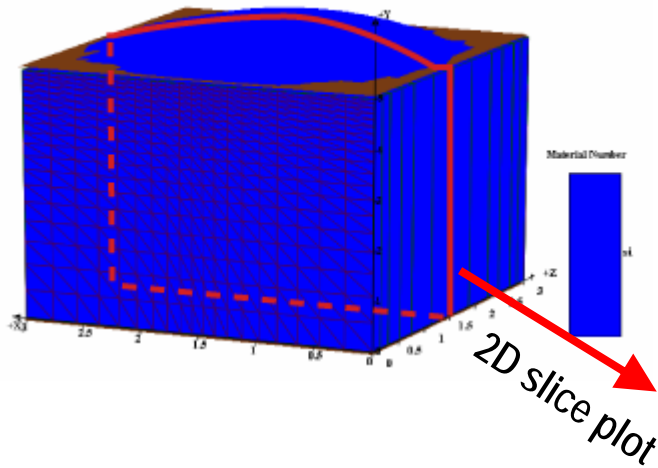


APSYS-FDTD Simulation Example 3

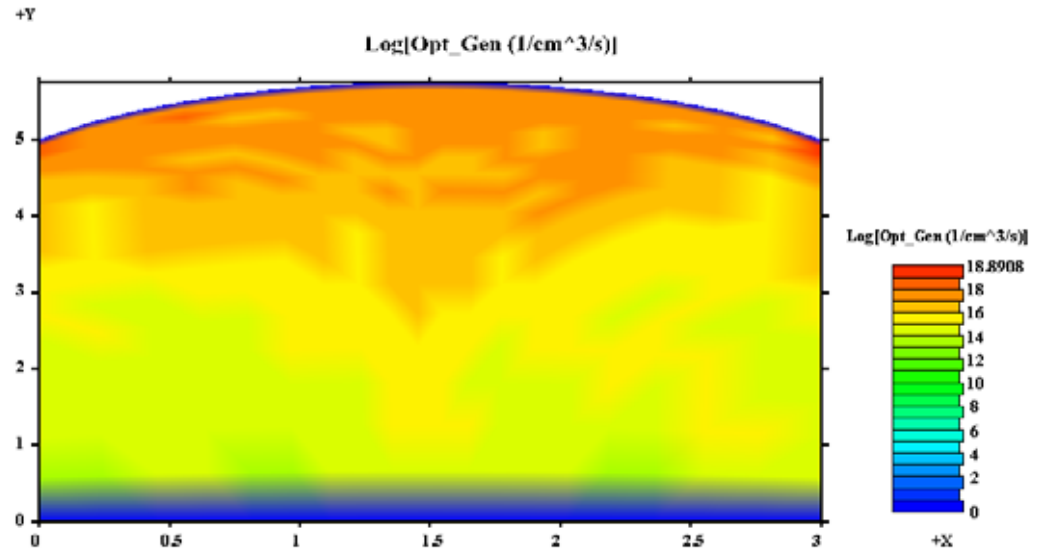
Focusing effect in 3-D Photodetector with lens shaped window



Optical Generation Rate (log scale)

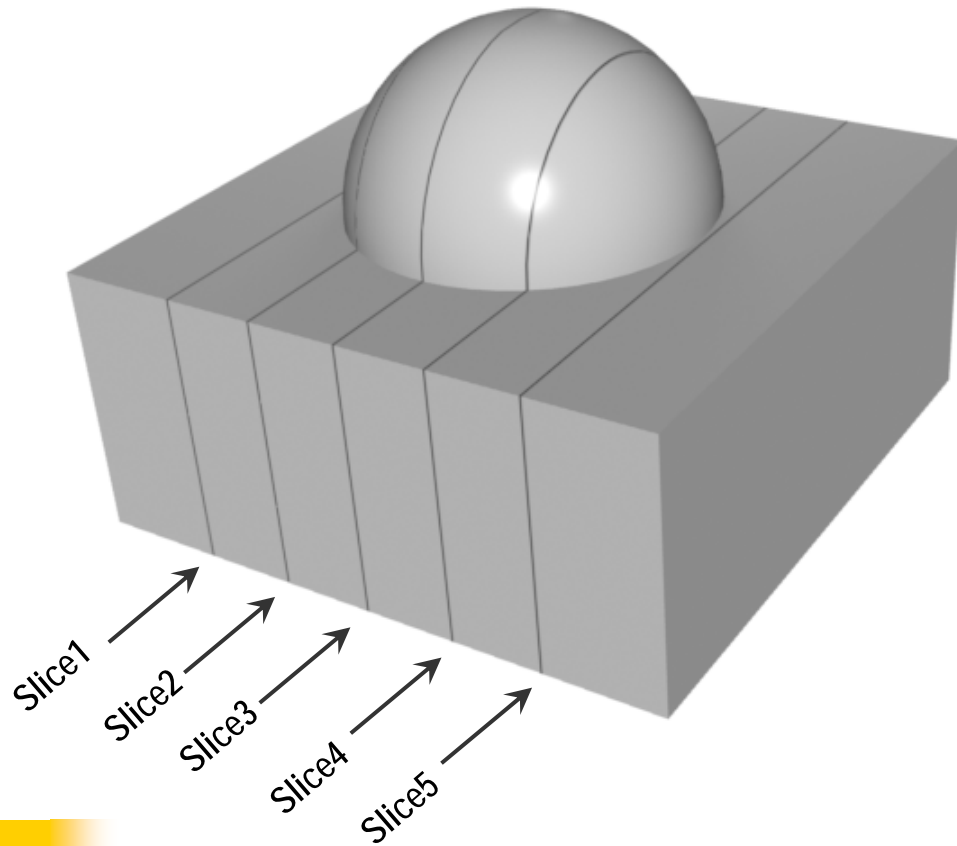


2D slice plot



Focusing effect can be observed clearly

Modeling 3D Texture

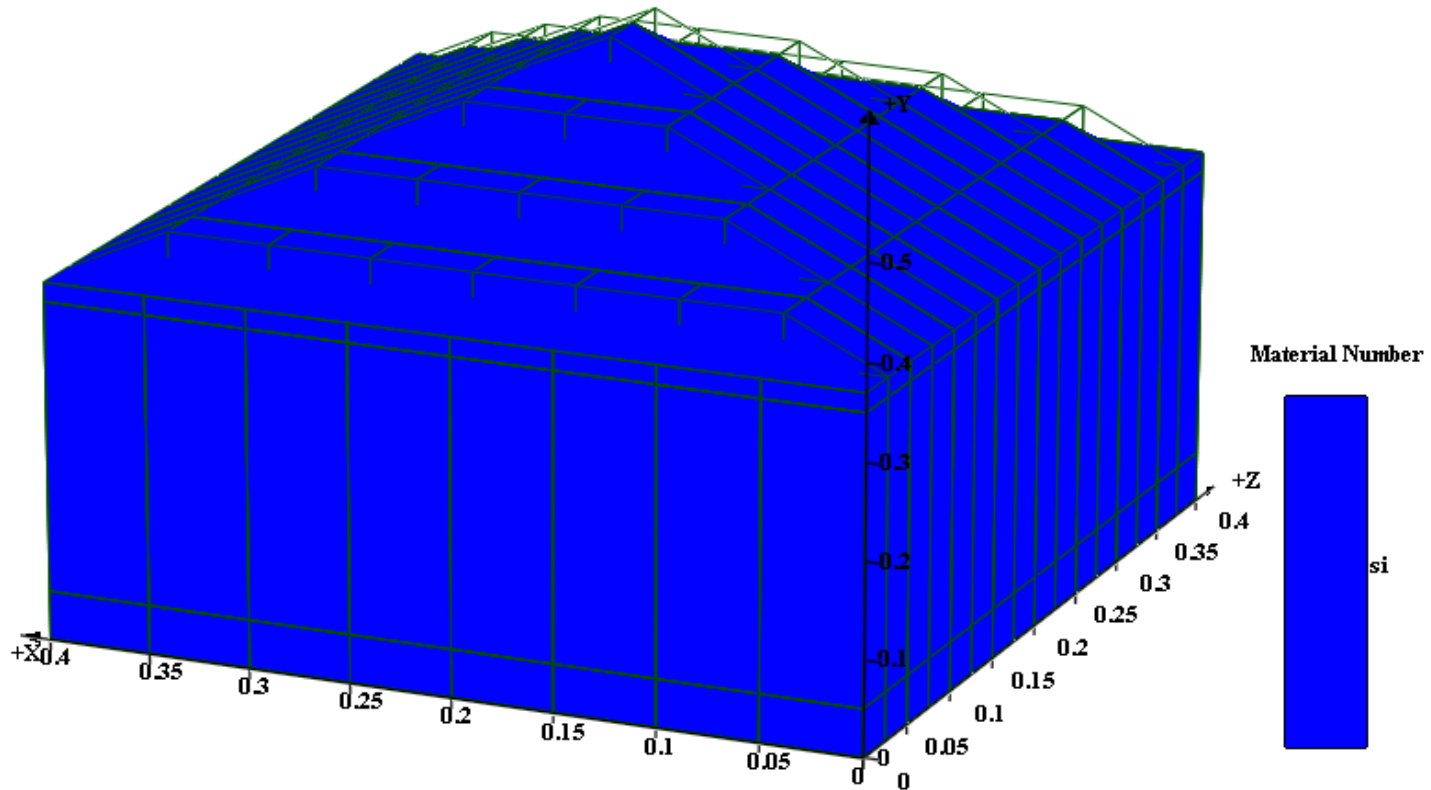


Complicated 3D geometry can be defined by connecting multiple 2D slices.

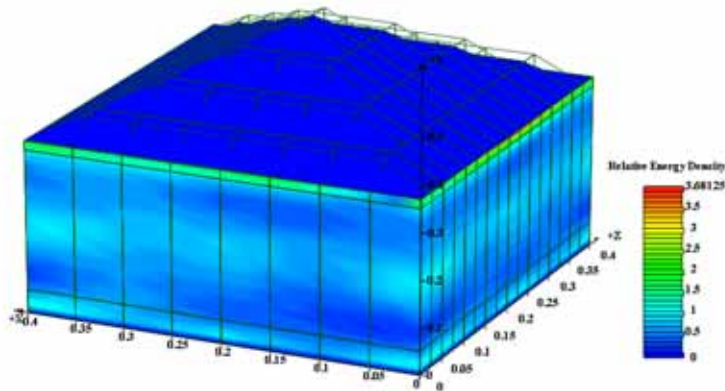
3D texture data generated by CAD or 3DCG software may be imported by APSYS-FDTD package in the future.

APSYS-FDTD Simulation Example 4

Pyramidal texture for Si solar cell

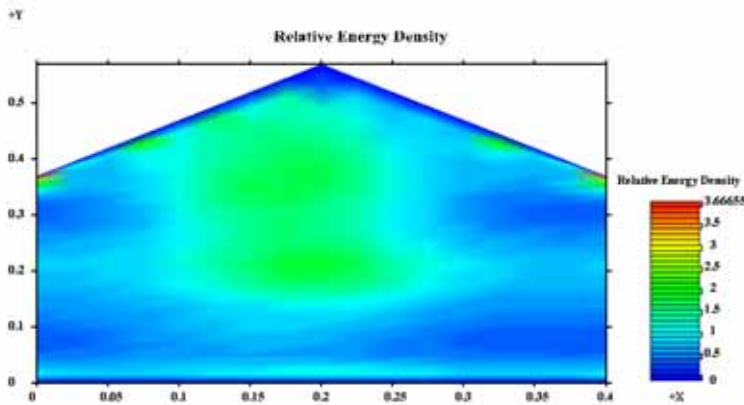


Relative Energy Density

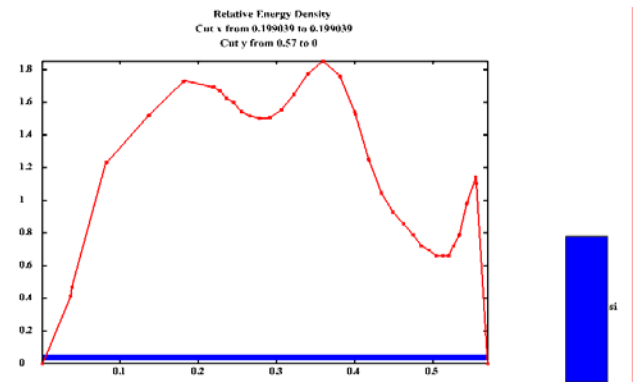


3D View

Pyramidal texture enhances transmission of incident light.

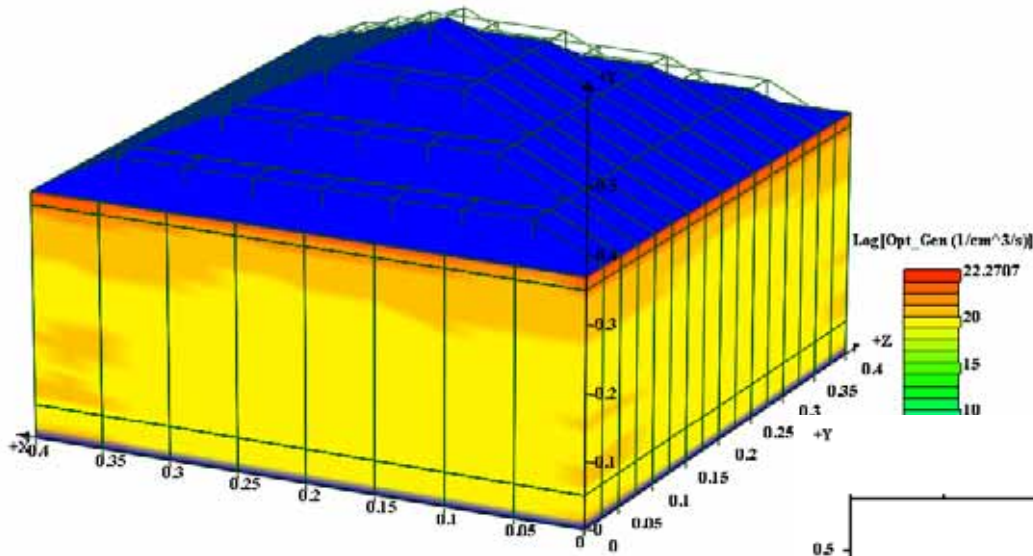


2D Slice at center

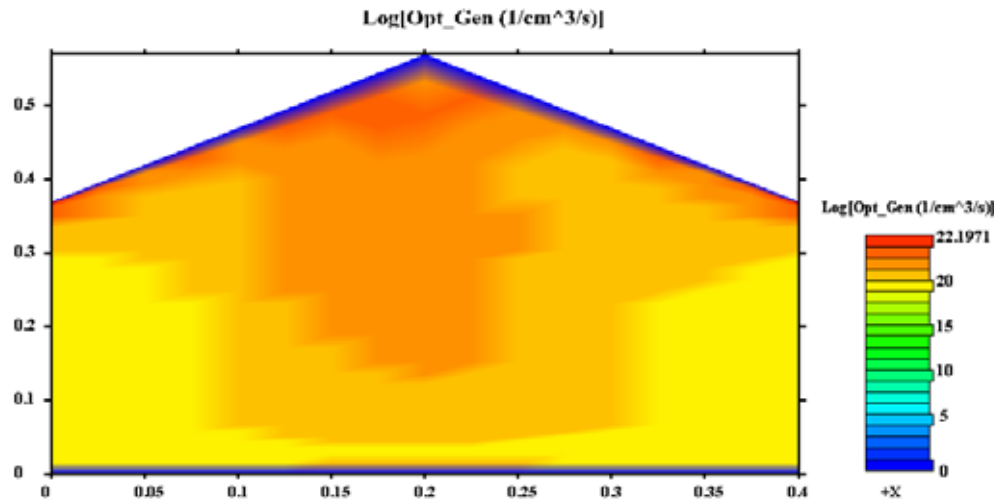


1D Slice at center

Optical Generation Rate (log scale)

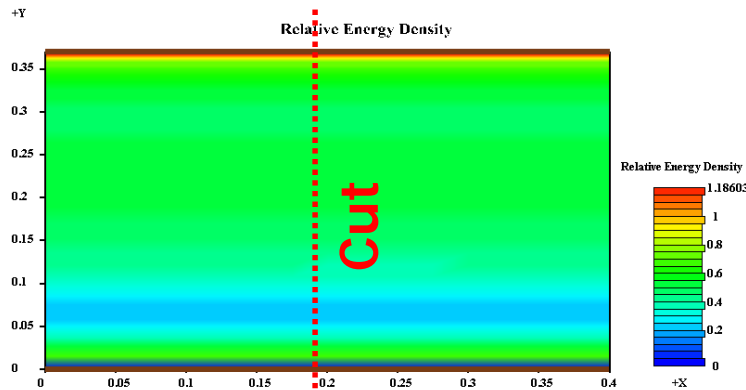


3D View

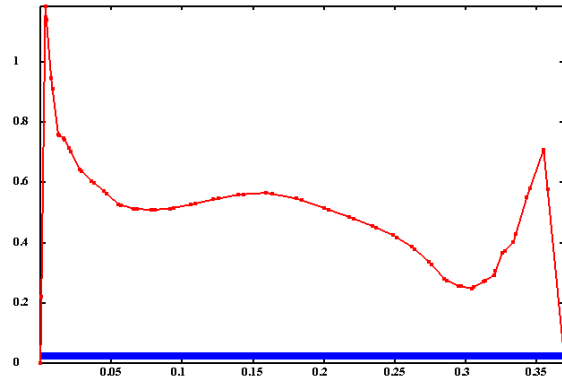


2D Slice at Center

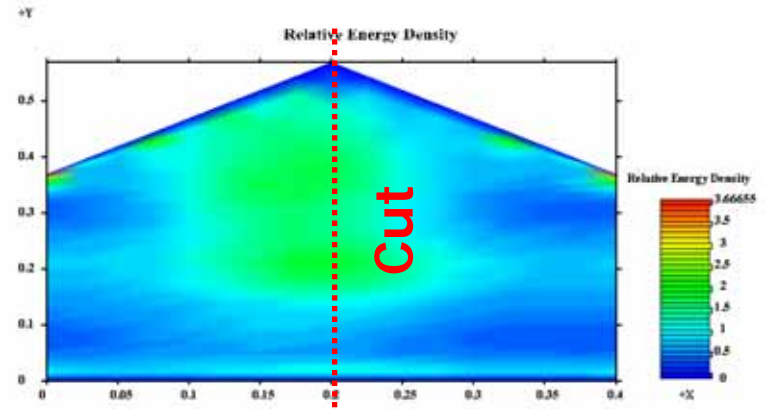
Comparison of Flat and Textured surface



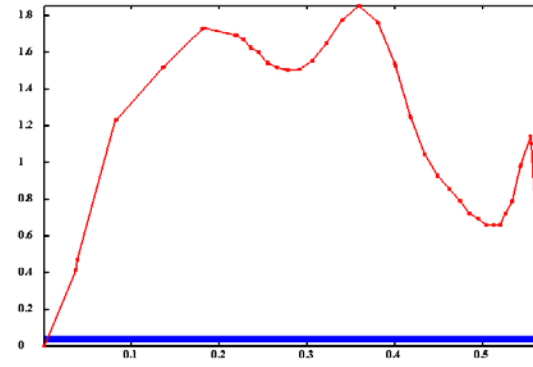
Relative Energy Density
Cut x from 0.195332 to 0.195332
Cut y from 0.37 to 0



Distance(μm)



Relative Energy Density
Cut x from 0.199039 to 0.199039
Cut y from 0.57 to 0



Distance(μm)

Textured surface yields high intensity light distribution around center of device.



System Requirement

- 2GB memory recommended for **3D** simulation.
- **64-bit** CPU and OS is recommended if user deals with big structure.
- **FDTD** simulation time varies from few minutes to few hours depending on problem. Obviously, faster CPU is better.



Summary

- **APSYS** and **FDTD(MEEP)** now work together seamlessly.
- 2D and 3D structures were used to demonstrate capability of **APSYS-FDTD** package.
- 64-bit CPU and OS may be necessary to carry out complicated 3D structure. **APSYS** is **64-bit ready** software.