UNIVERSITÄT DUISBURG ESSEN

Open-Minded

openEMS in open source EDA

at OpenPDK Workshop 2023, Frankfurt (Oder), Germany

Jan Taro Svejda — jan.svejda@uni-due.de

General and Theoretical Electrical Engineering (ATE),
University of Duisburg-Essen,
CENIDE – Center for Nanointegration Duisburg-Essen,
D-47048 Duisburg, Germany

What is openEMS ?

an open source electromagnetic field solver

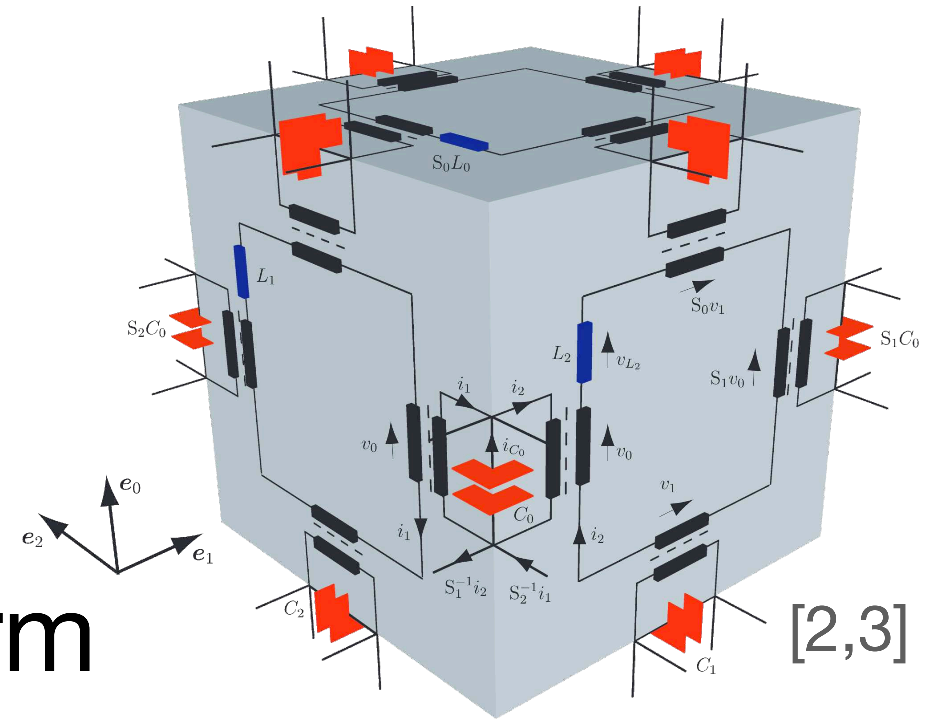
- Filed calculations with EC-FDTD [1,2] scheme
- Supported coordinate systems: cartesian and cylindrical
- Controlled by powerful scripting interface in
  - Matlab | Octave
  - Python
- Result evaluation directly within scripting language or e.g. with ParaView
- Support for remote and cluster simulation using MPI
- Software available for Linux, MacOS and Windows

Underlying EC-FDTD Engine

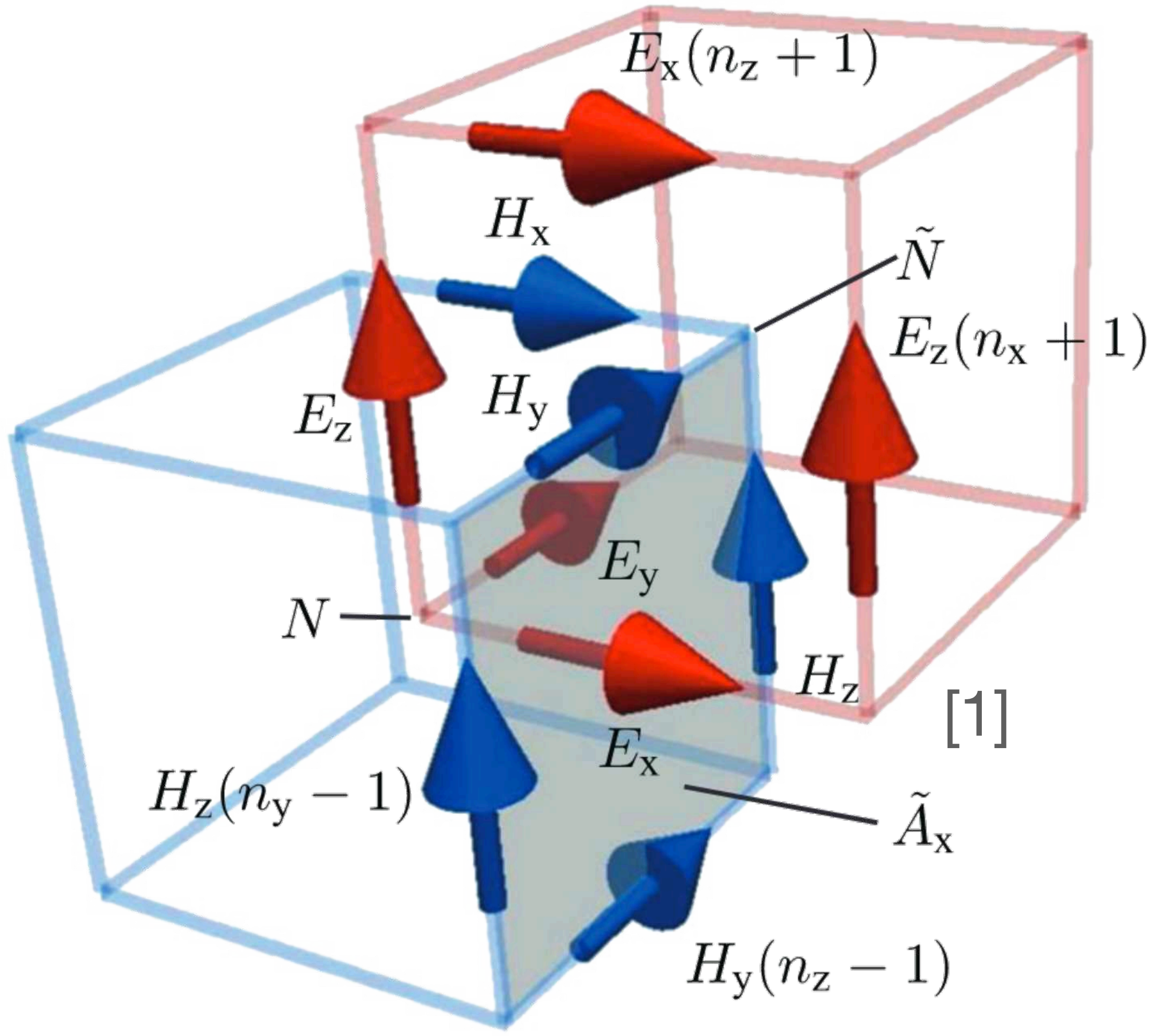
Equivalent Circuit — Finite Differences in Time Domain

$$\oint_{\partial\Sigma} \mathbf{E} \cdot d\boldsymbol{\ell} = -\frac{d}{dt} \iint_{\Sigma} \mathbf{B} \cdot d\mathbf{S}$$

$$\oint_{\partial\Sigma} \mathbf{B} \cdot d\boldsymbol{\ell} = \mu_0 \left(\iint_{\Sigma} \mathbf{J} \cdot d\mathbf{S} + \varepsilon_0 \frac{d}{dt} \iint_{\Sigma} \mathbf{E} \cdot d\mathbf{S} \right)$$

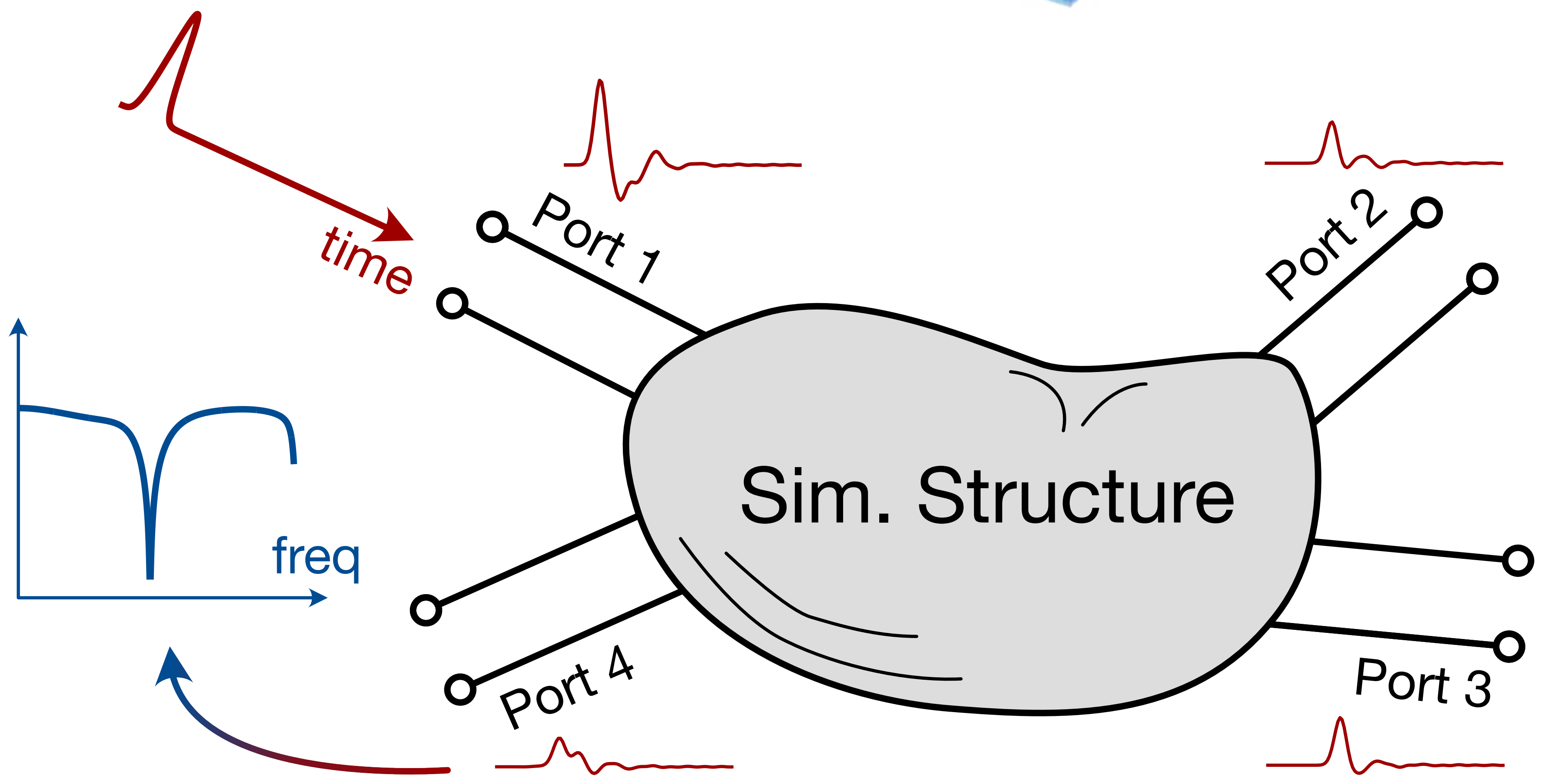


[2,3]



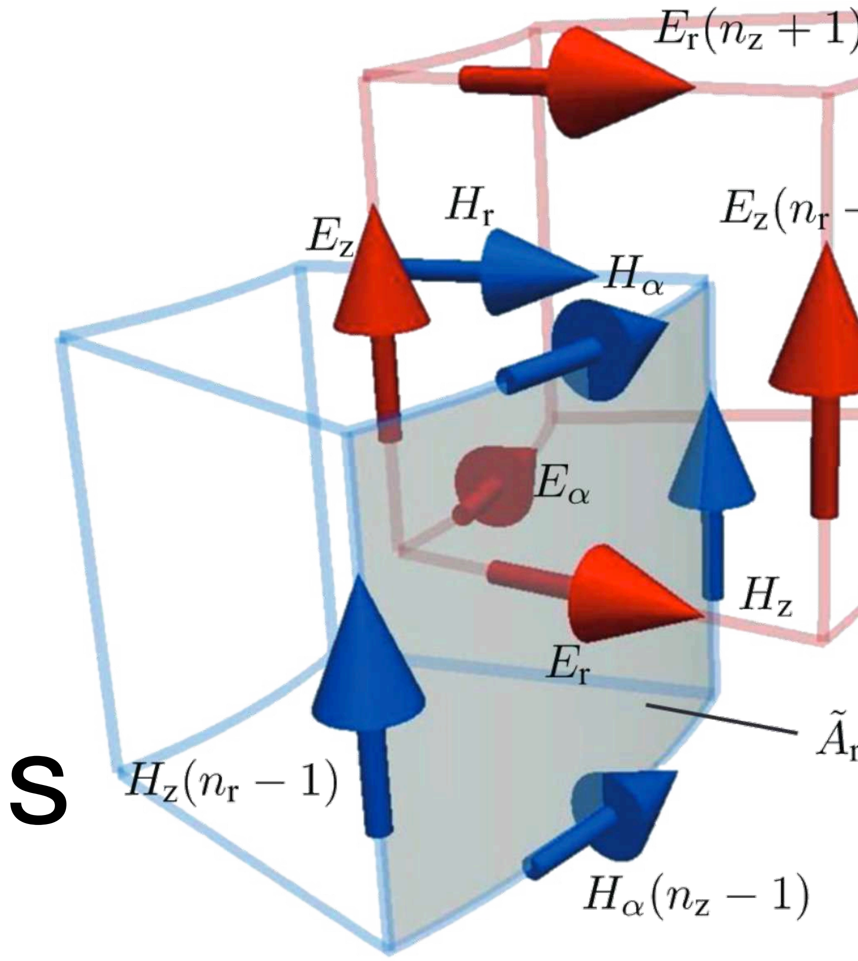
[1]

- Numerical equivalent to 'conventional' FDTD with all corresponding (dis)advantages
- Current and Voltages along with circuit elements form update equations
- Time domain method > broad band with one simulation (net response to gaussian pulse and FFT for freq. dom.)
- Computational resources scale linearly with problem size
- Integration of dispersive media via filter circuit (by e.g. multi-polar Drude-Lorentz model)
- Rigorous energy-based stability proof possible [2]



Cylindrical Coordinate System

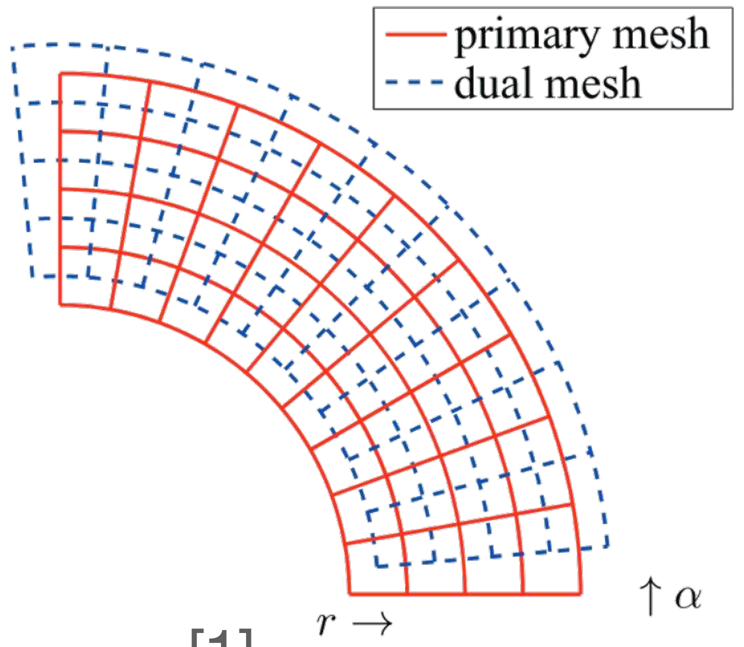
a few words



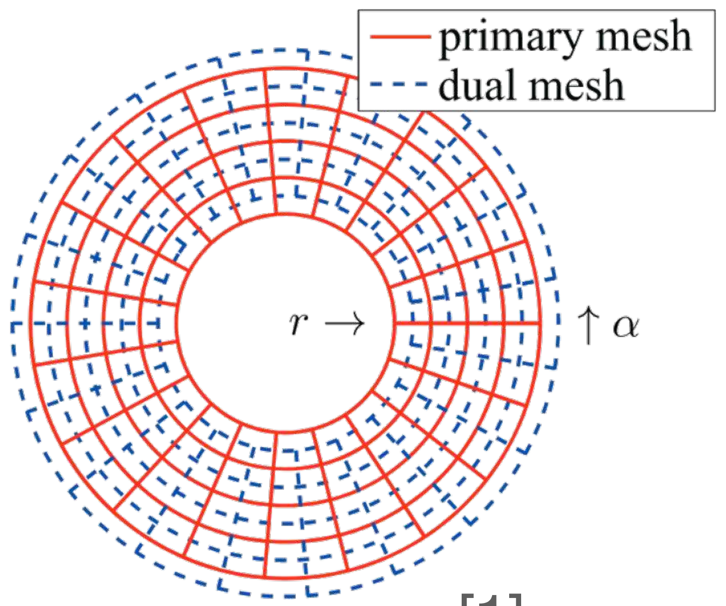
[1]

- *Identical* update equations (layout), coefficient format, calculation of EC parameters, support for e.g. dispersive materials
- Sole *difference* in calculation of edge length and surface areas

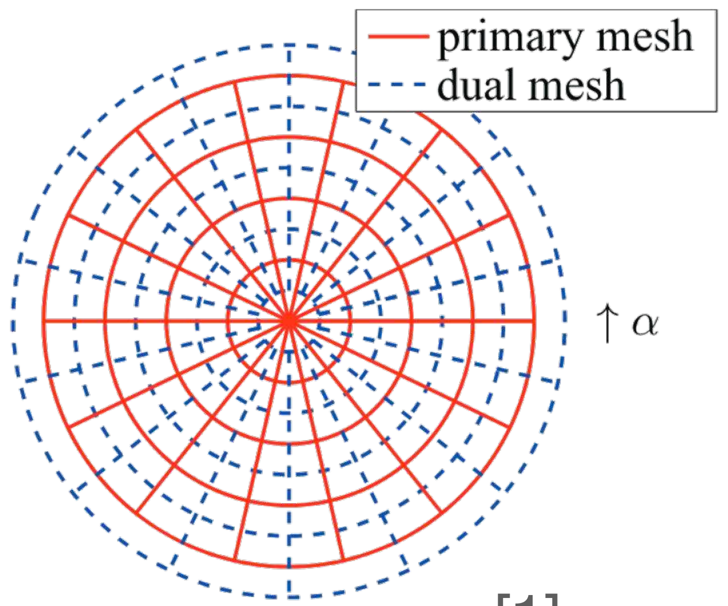
- Special boundary conditions needed: whole circle, origin
- Sub division of mesh grid (necessarily) useful



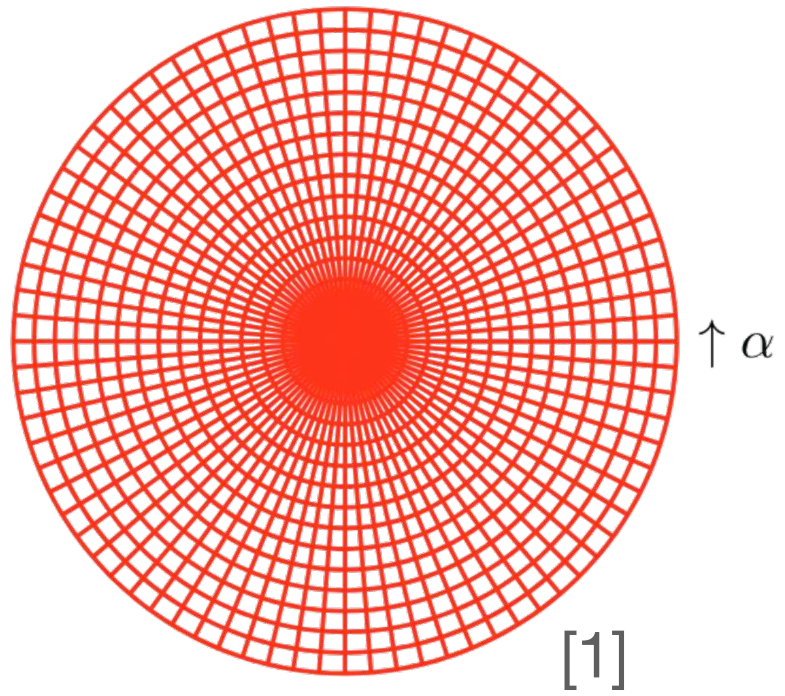
[1]



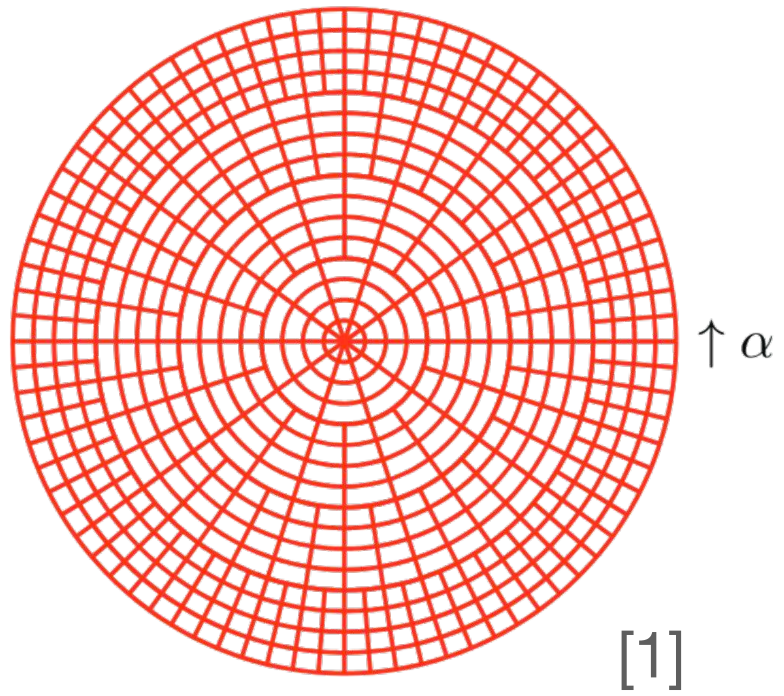
[1]



[1]



[1]



[1]

Structure of a Simulation

define simulation structure

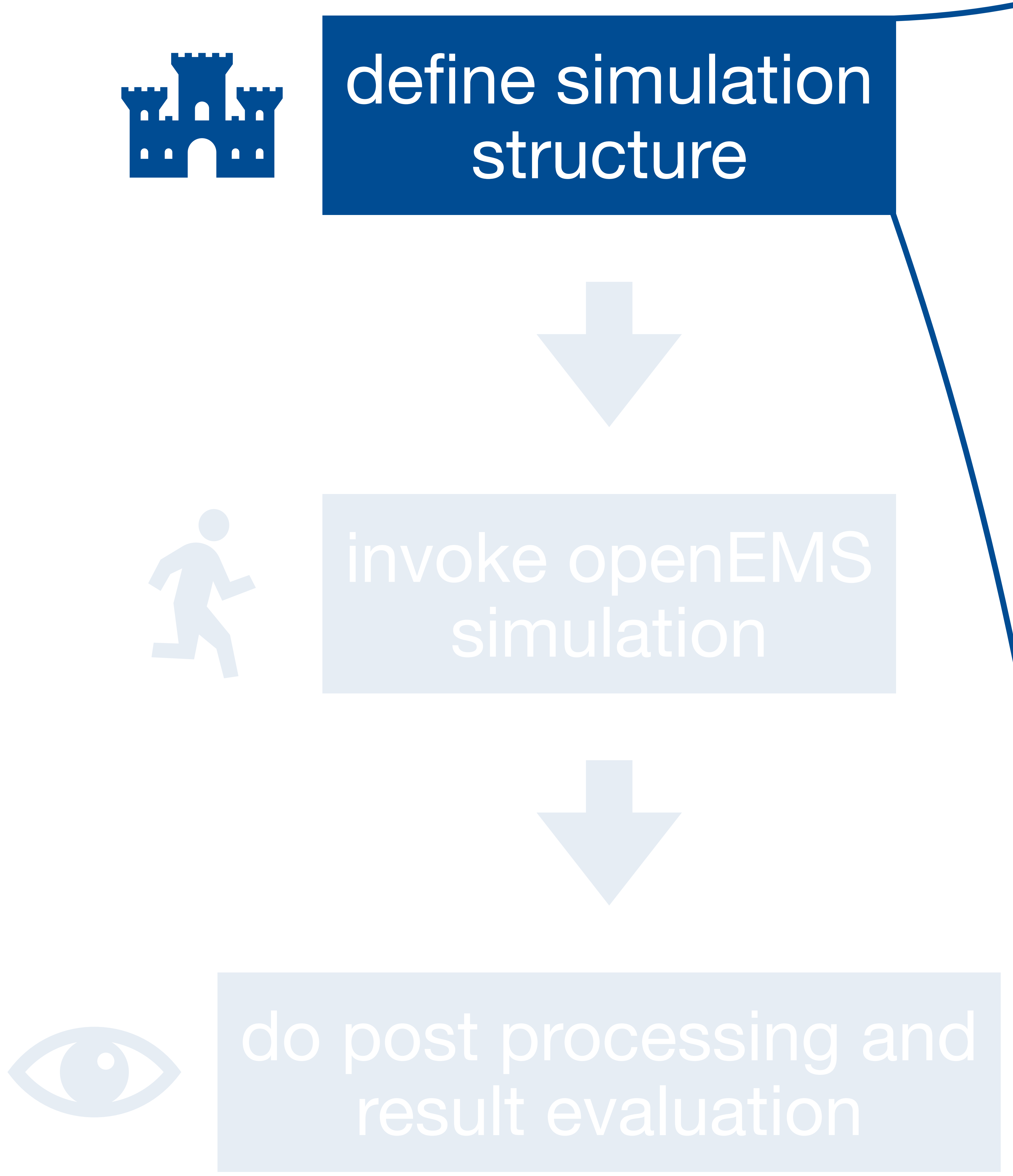
↓

invoke openEMS simulation

↓

do post processing and result evaluation

Structure of a Simulation



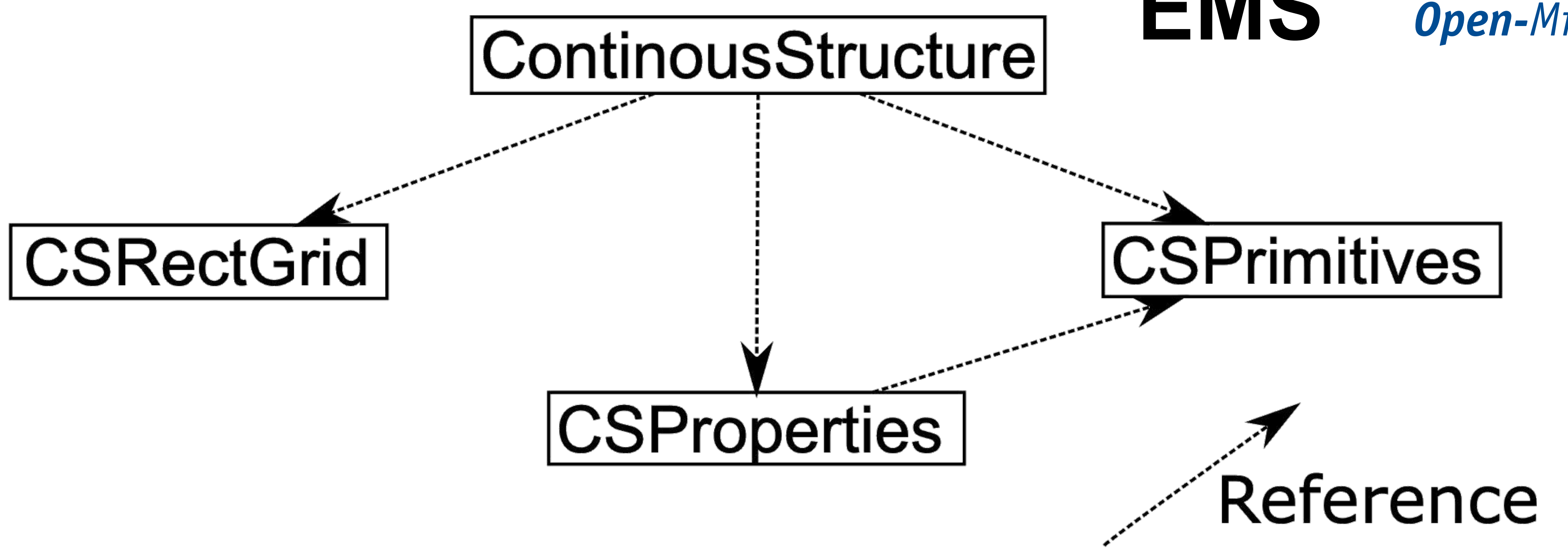
- Initialise CSXCAD
  - ➡ Continuous Structure XML CAD
  - open source library
  - description of geometrical objects along with physical properties
- Define parameters, functions, etc.
- Add material groups
- Add CSX specific primitive objects to material group
- (manually) define and smooth out mesh grid
- Inspect structure and mesh with AppCSXCAD

Structure of a Simulation

define simulation structure

invoke openEMS simulation

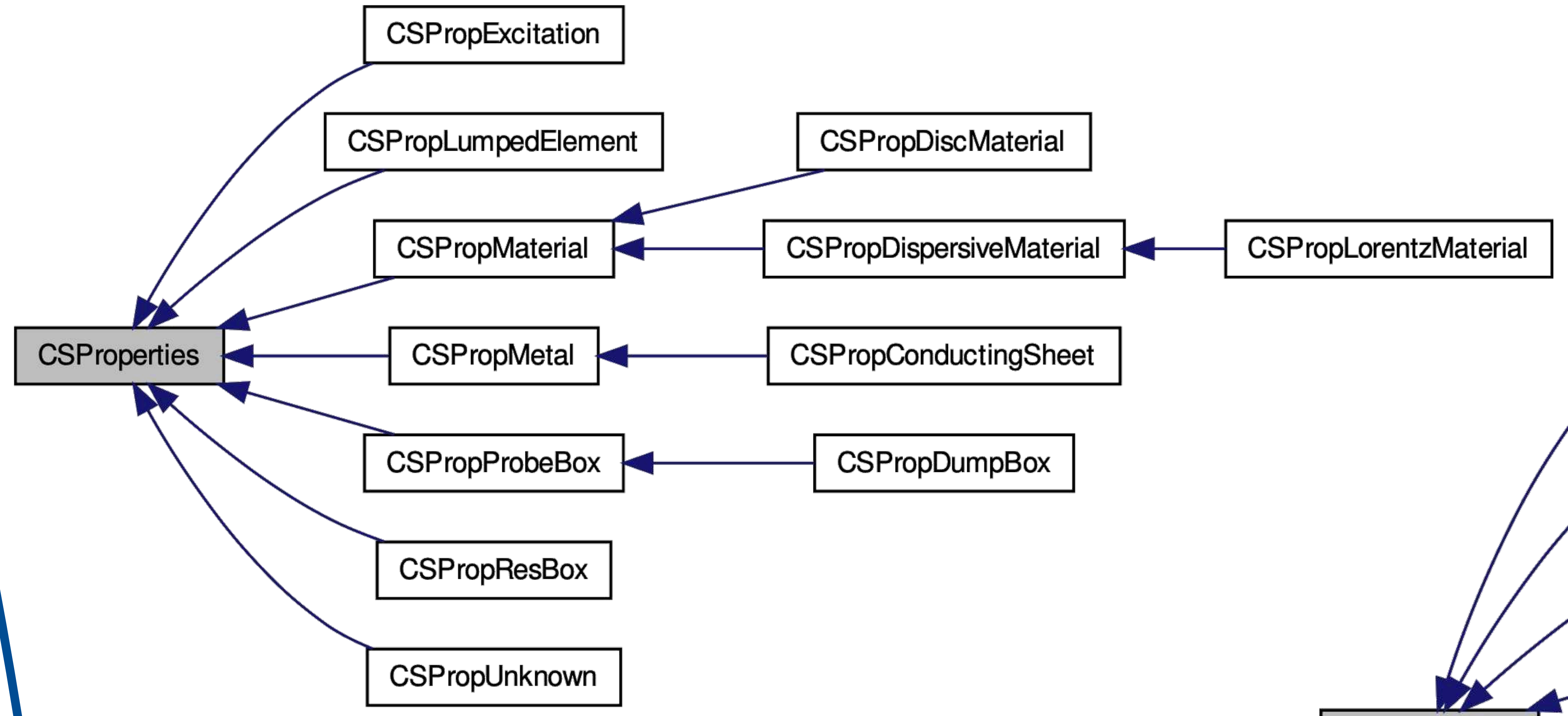
do post processing and result evaluation

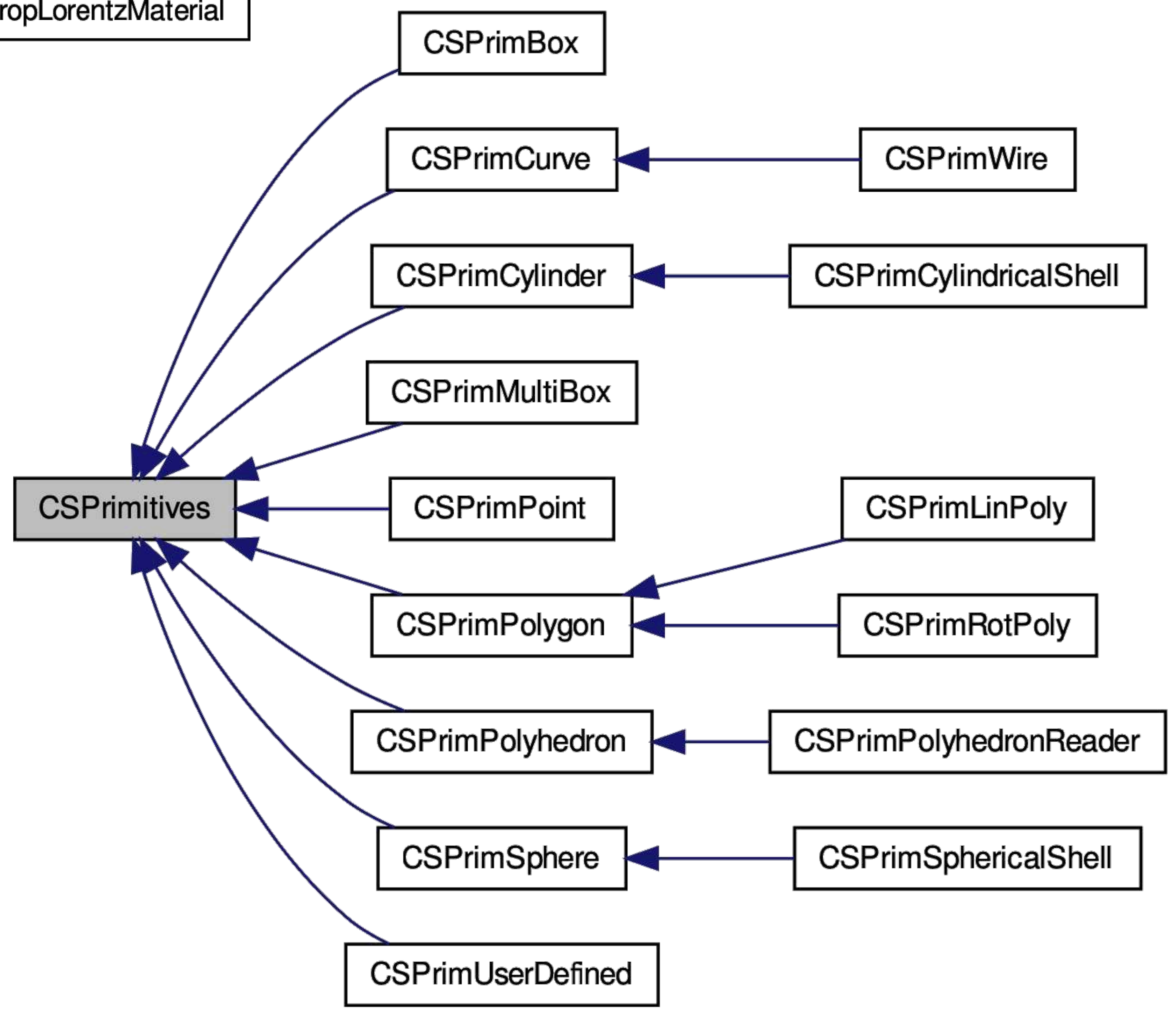


`CSRectGrid`: Class contains information for a (cartesian or cylindrical) mesh

`CSProperties`: Class contains information about physical and non-physical properties

`CSPrimitives`: Class contains description for geometrical primitives





Structure of a Simulation

define simulation structure

invoke openEMS simulation

do post processing and result evaluation

- Start openEMS by calling function within scripting interface
- Typical command line output:

```
 ----------------------------------------------------------------------
 | openEMS 64bit -- version v0.0.35-110-g782a738
 | (C) 2010-2023 Thorsten Liebig <thorsten.liebig@gmx.de>  GPL license
 ----------------------------------------------------------------------
        Used external libraries:
                CSXCAD -- Version: v0.6.2-124-gb5919a6
                hdf5   -- Version: 1.14.1
                          compiled against: HDF5 library version: 1.14.1-2
                tinyxml -- compiled against: 2.6.2
                fparser
                boost  -- compiled against: 1_81
                vtk -- Version: 9.2.6
                       compiled against: 9.2.6

Create FDTD operator (compressed SSE + multi-threading)
FDTD simulation size: 83x83x178 --> 1.22624e+06 FDTD cells
FDTD timestep is: 5.58967e-12 s; Nyquist rate: 30 timesteps @2.98169e+09 Hz
Excitation signal length is: 1026 timesteps (5.73501e-09s)
Max. number of timesteps: 1000000000 ( --> 974659 * Excitation signal length)
Create FDTD engine (compressed SSE + multi-threading)
Running FDTD engine... this may take a while... grab a cup of coffee?!?
[@        4s] Timestep:          203 || Speed:   61.8 MC/s (1.985e-02 s/TS) || Energy: ~1.03e-18 (- 0.00dB)
...
[@       52s] Timestep:         3157 || Speed:   74.7 MC/s (1.641e-02 s/TS) || Energy: ~4.52e-20 (-52.49dB)
Time for 3157 iterations with 1226242.00 cells : 52.75 sec
Speed: 73.39 MCells/s
```

Open-Minded

Structure of a Simulation

define simulation structure

invoke openEMS simulation

do post processing and result evaluation

- Start openEMS by calling function within scripting interface
- Typical command line output:

```
 ----------------------------------------------------------------------
 | openEMS 64bit -- version v0.0.35-110-g782a738
 | (C) 2010-2023 Thorsten Liebig <thorsten.liebig@gmx.de>  GPL license
 ----------------------------------------------------------------------
        Used external libraries:
                CSXCAD -- Version: v0.6.2-124-gb5919a6
                hdf5   -- Version: 1.14.1
                          compiled against: HDF5 library version: 1.14.1-2
                tinyxml -- compiled against: 2.6.2
                fparser
                boost  -- compiled against: 1_81
                vtk -- Version: 9.2.6
                       compiled against: 9.2.6

Create FDTD operator (compressed SSE + multi-threading)
FDTD simulation size: 83x83x178 --> 1.22624e+06 FDTD cells
FDTD timestep is: 5.58967e-12 s; Nyquist rate: 30 timesteps @2.98169e+09 Hz
Excitation signal length is: 1026 timesteps (5.73501e-09s)
Max. number of timesteps: 1000000000 ( --> 974659 * Excitation signal length)
Create FDTD engine (compressed SSE + multi-threading)
Running FDTD engine... this may take a while... grab a cup of coffee?!?
[@        4s] Timestep:          203 || Speed:   61.8 MC/s (1.985e-02 s/TS) || Energy: ~1.03e-18 (- 0.00dB)
...
[@       52s] Timestep:         3157 || Speed:   74.7 MC/s (1.641e-02 s/TS) || Energy: ~4.52e-20 (-52.49dB)
Time for 3157 iterations with 1226242.00 cells : 52.75 sec
Speed: 73.39 MCells/s
```

Structure of a Simulation

- define simulation structure
- invoke openEMS simulation
- **do post processing and result evaluation**

- Postprocessing
  - Calculate port parameters
    e.g. S-Parameters
  - Perform near-field to far-field (NF2FF) transformation
- Export dumped fields for use in other programs like e.g. ParaView
- Calculate complex weighted field superpositions

➡ Complex weights by currents from e.g. circuit simulations (freq. dom.)

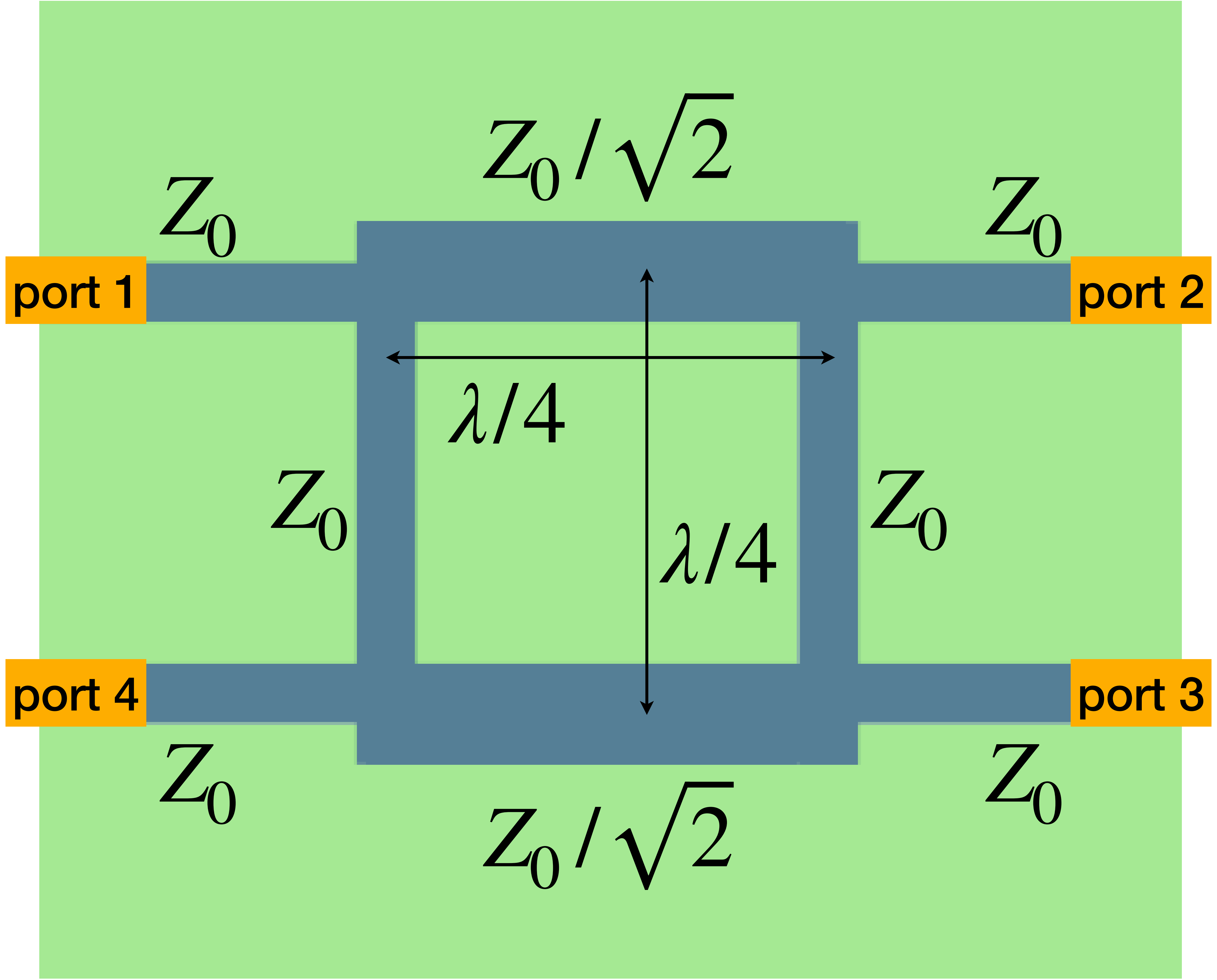
✓ openEMS comes with simple circuit simulator (CTB)

★ Quickly tune lumped elements replacing a port

Simple Example Simulation

the problem

- Problem definition:
  - hybrid coupler (power split, 90° phase) in microstrip strip line technoloy
  - Substrate:
    Rogers RO4350B
    1.5mm thickness
  - Frequency range 0.0 … 4.0 GHz
  - Operating frequency 2.0 GHz



Simple Example Simulation

define simulation structure

```
% Parameteres
addpath('~/opt/openEMS_build/share/openEMS/matlab');
addpath('~/opt/openEMS_build/share/CSXCAD/matlab');

const_c0 = 299792458;   const_mue0 = 1.2566370621219e-6;    const_eps0 = 1/(const_c0^2*const_mue0);
unit_len = 1e-6; % Lengths in unit um

freq_0 = 2e9;   freq_range = [0.0, 4e9];

sub_h = 1524;   sub_epsr = 3.66;    sub_tand = 0.0037;
msl_Zc = [50, 50/sqrt(2)];
for n = 1:2    msl_w(n) = calc_msl_width_by_book_of_pozar(msl_Zc(n), sub_epsr, sub_h);    end
eps_eff = (sub_epsr+1)/2;
lambda_0 = const_c0/freq_0 /unit_len;   lambda_range = const_c0./freq_range /unit_len;
mesh_max_delta = min(lambda_range) /sqrt(sub_epsr) /20;
feed_l = lambda_0 / 2;
```

Simple Example Simulation

define simulation structure

```
% Init FDTD and CSX
FDTD = InitFDTD('EndCriteria', 1e-6, 'Oversampling', 6);
FDTD = SetGaussExcite(FDTD, 0, max(freq_range));
FDTD = SetBoundaryCond(FDTD, {'PML_8', 'PML_8', 'MUR', 'MUR', 'PEC', 'PML_8'});
CSX = InitCSX();
% Define Structure
CSX = AddMetal(CSX, 'cond_top');
CSX = AddMaterial(CSX, 'RO4350B');
CSX = SetMaterialProperty(CSX, 'RO4350B', 'Epsilon', sub_epsr, 'Kappa', sub_tand*2*pi*10e9*sub_epsr*const_eps0);
```

(longer) code block not shown:

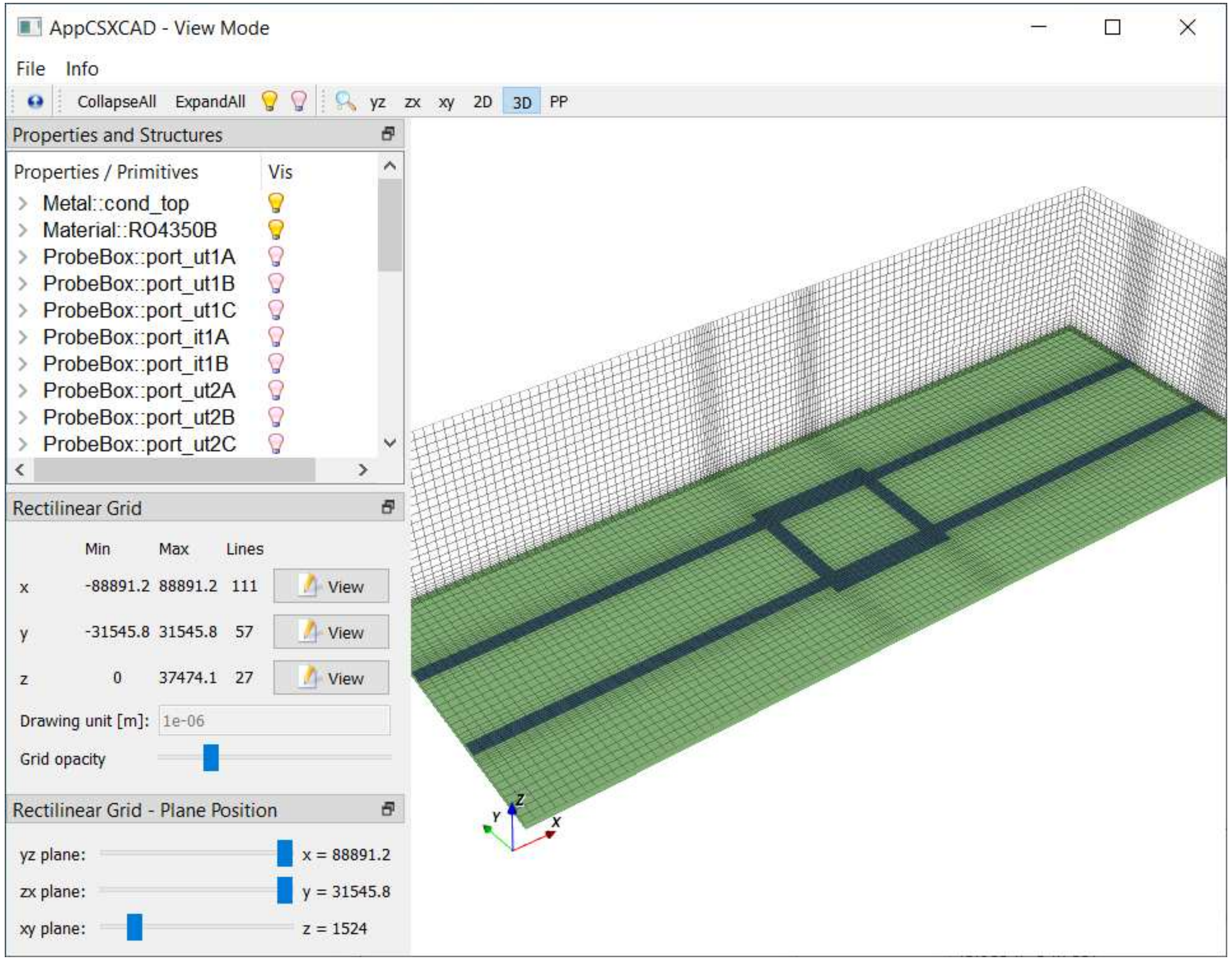
- calculating start and end points of boxes
- adding boxes to materials
- loop for similar parts
- add mesh lines of edges
- smooth mesh

```
start = [mesh.x(1),   mesh.y(1),   0];
stop  = [mesh.x(end), mesh.y(end), sub_h];
CSX = AddBox( CSX, 'RO4350B', 0, start, stop );

mesh.x = AutoSmoothMeshLines(mesh.x, mesh_max_delta, 1.25);
```

Simple Example Simulation

invoke openEMS simulation

```
% Save, view and run simulation
Sim_Path = 'MSL_Quad_Hybrid';
Sim_CSX = 'MSL_Quad_Hybrid.xml';

[~] = rmdir(Sim_Path, 's'); % clear previous directory
[~] = mkdir(Sim_Path); % create empty simulation folder

WriteOpenEMS(fullfile(Sim_Path, Sim_CSX), FDTD, CSX);

CSXGeomPlot(fullfile(Sim_Path, Sim_CSX));

Settings.LogFile = 'openEMS.log';
RunOpenEMS(Sim_Path, Sim_CSX, '', Settings)
```



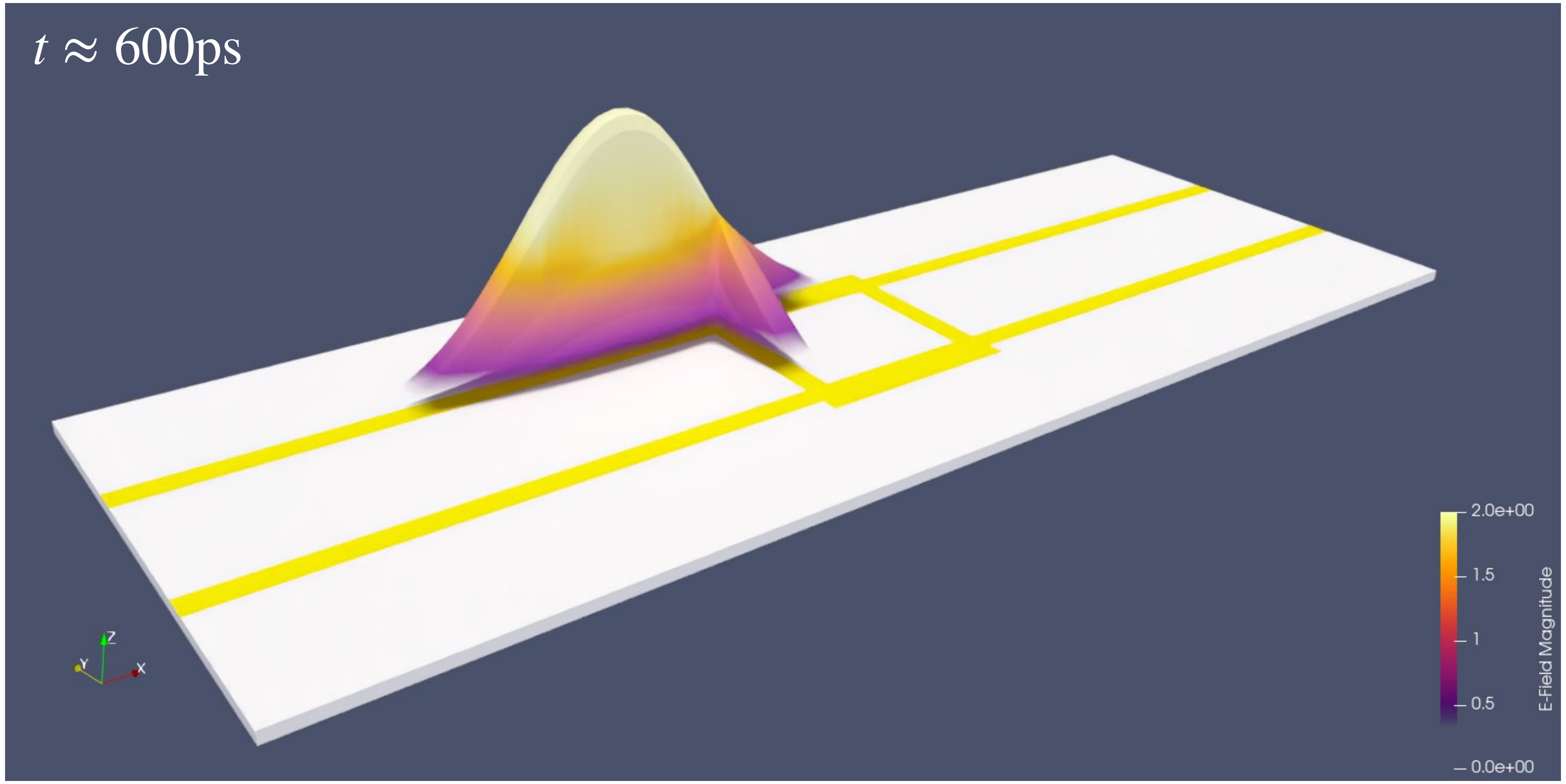
Simple Example Simulation

invoke openEMS simulation

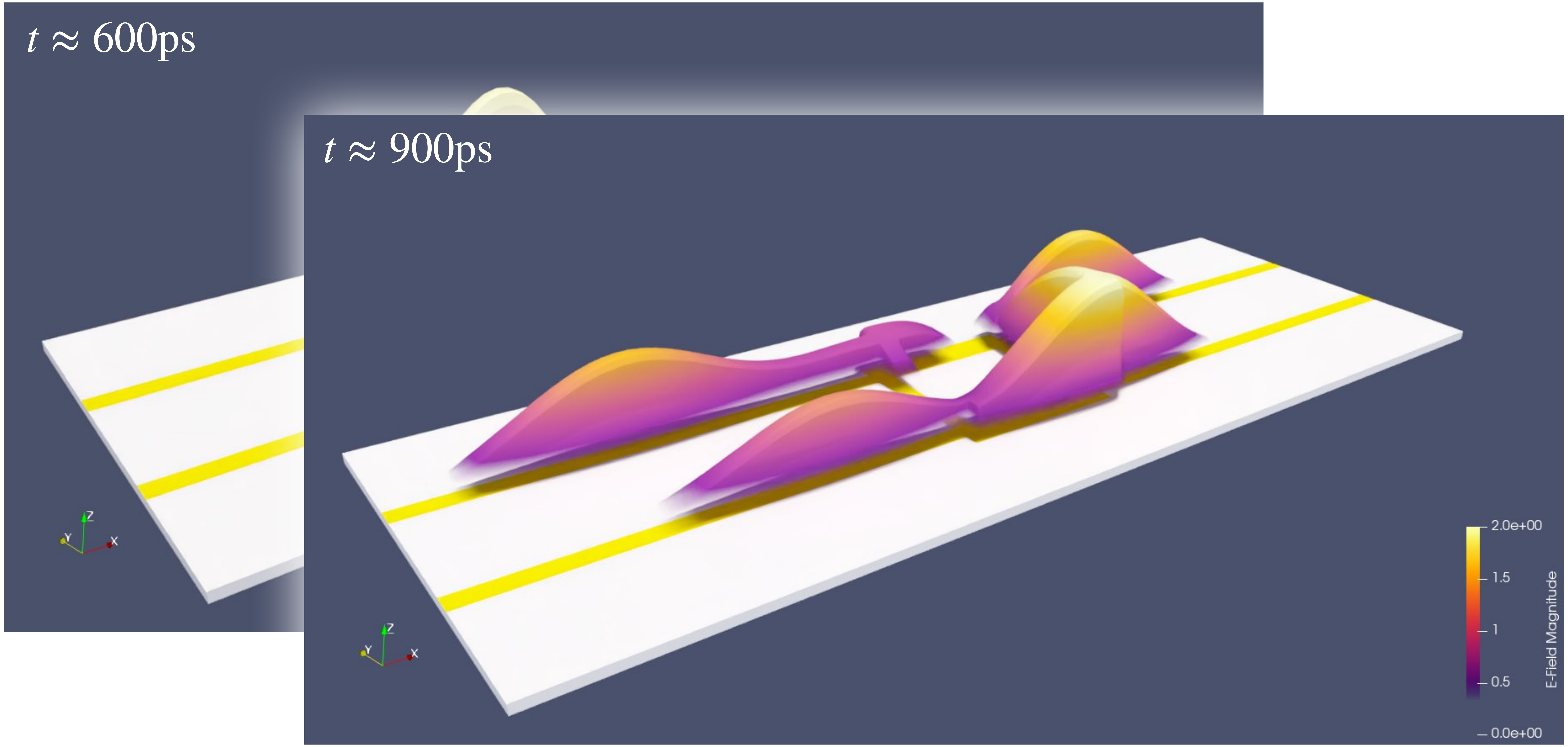
```
% Save, view and run simulation
Sim_Path = 'MSL_Quad_Hybrid';
Sim_CSX = 'MSL_Quad_Hybrid.xml';

[~] = rmdir(Sim_Path, 's'); % clear previous directory
[~] = mkdir(Sim_Path); % create empty simulation folder

WriteOpenEMS(fullfile(Sim_Path, Sim_CSX), FDTD, CSX);

CSXGeomPlot(fullfile(Sim_Path, Sim_CSX), ['--export-polydata-vtk=' Sim_Path]);

Settings.LogFile = 'openEMS.log';
RunOpenEMS(Sim_Path, Sim_CSX, '', Settings)
```

Running FDTD engine... this may take a while... grab a cup of coffee?!?

Magnitude of E-Field in Substrate

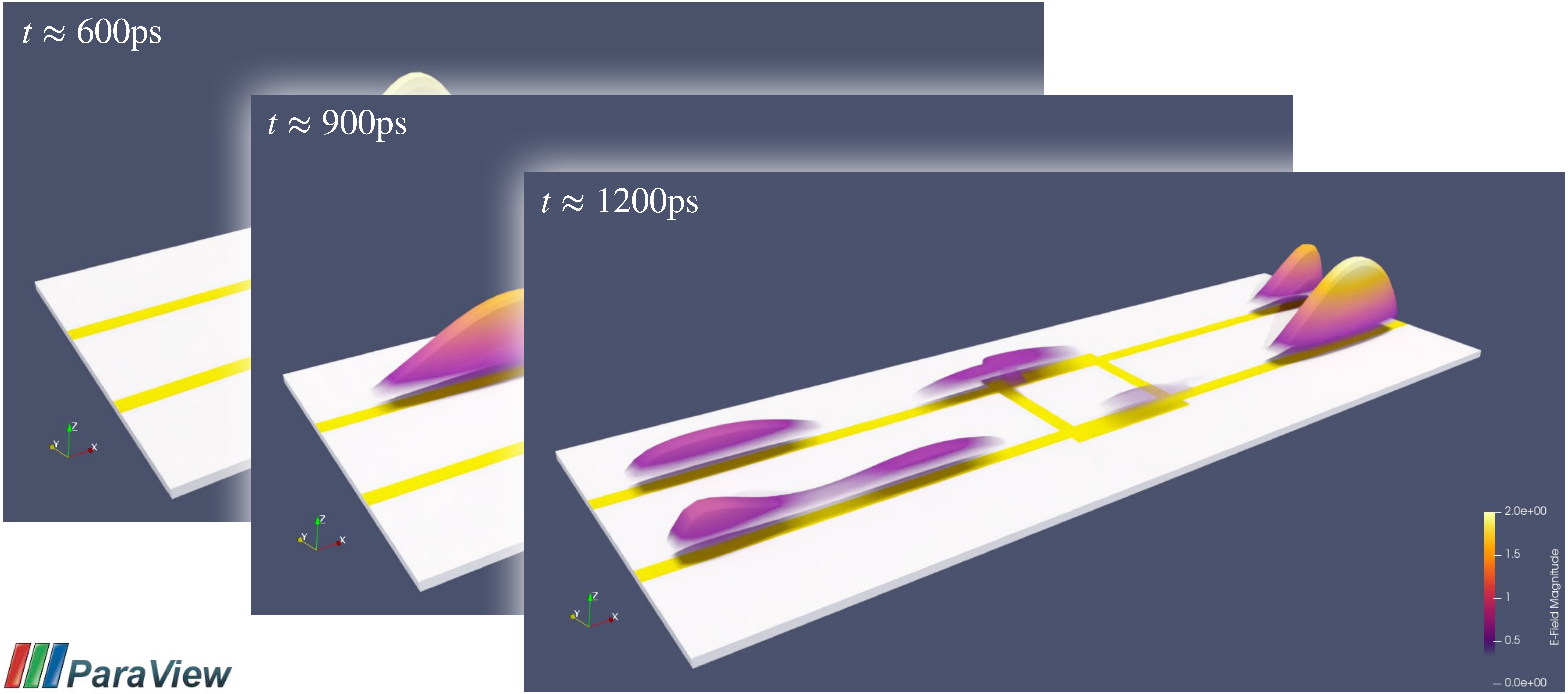


Magnitude of E-Field in Substrate



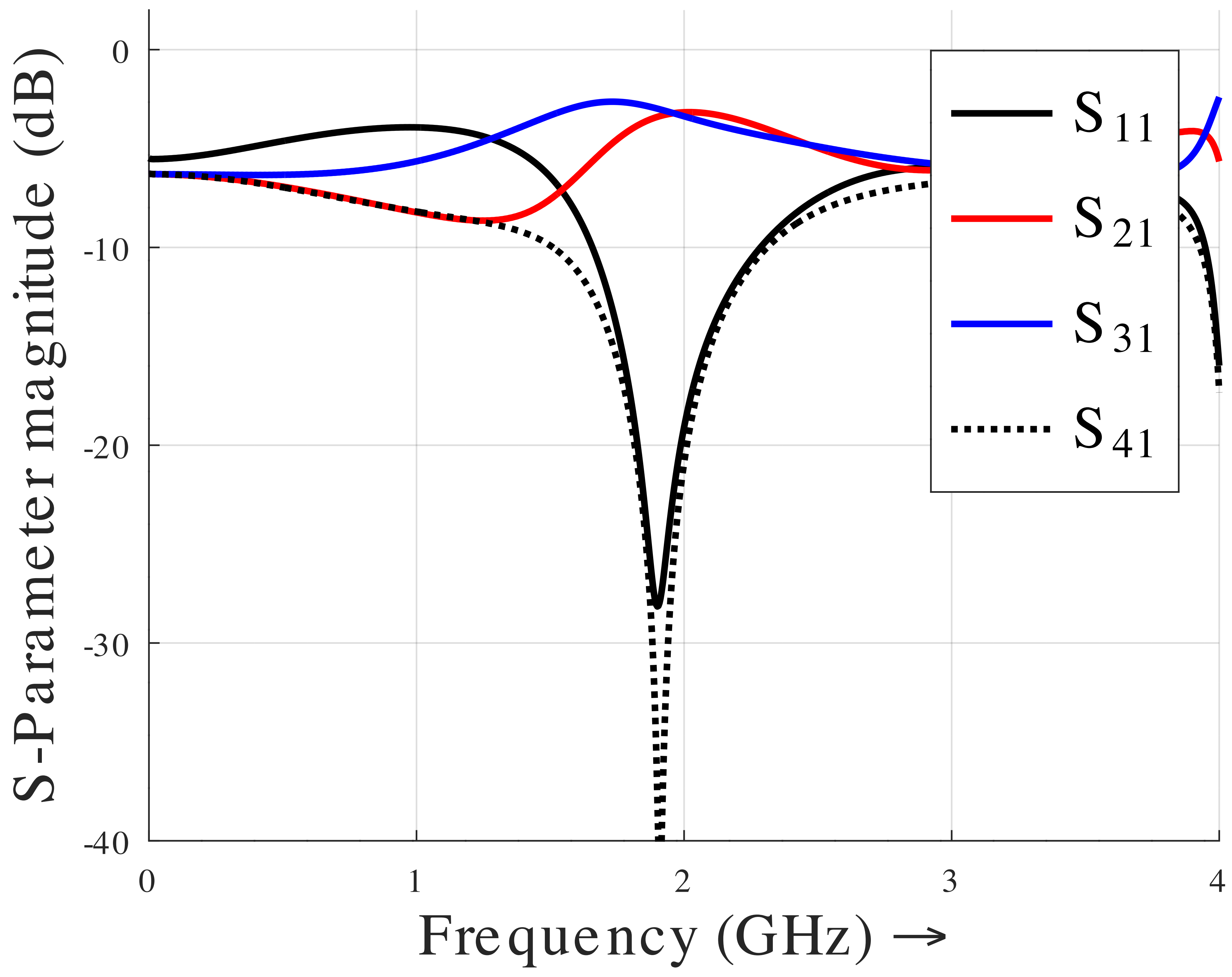
ParaView

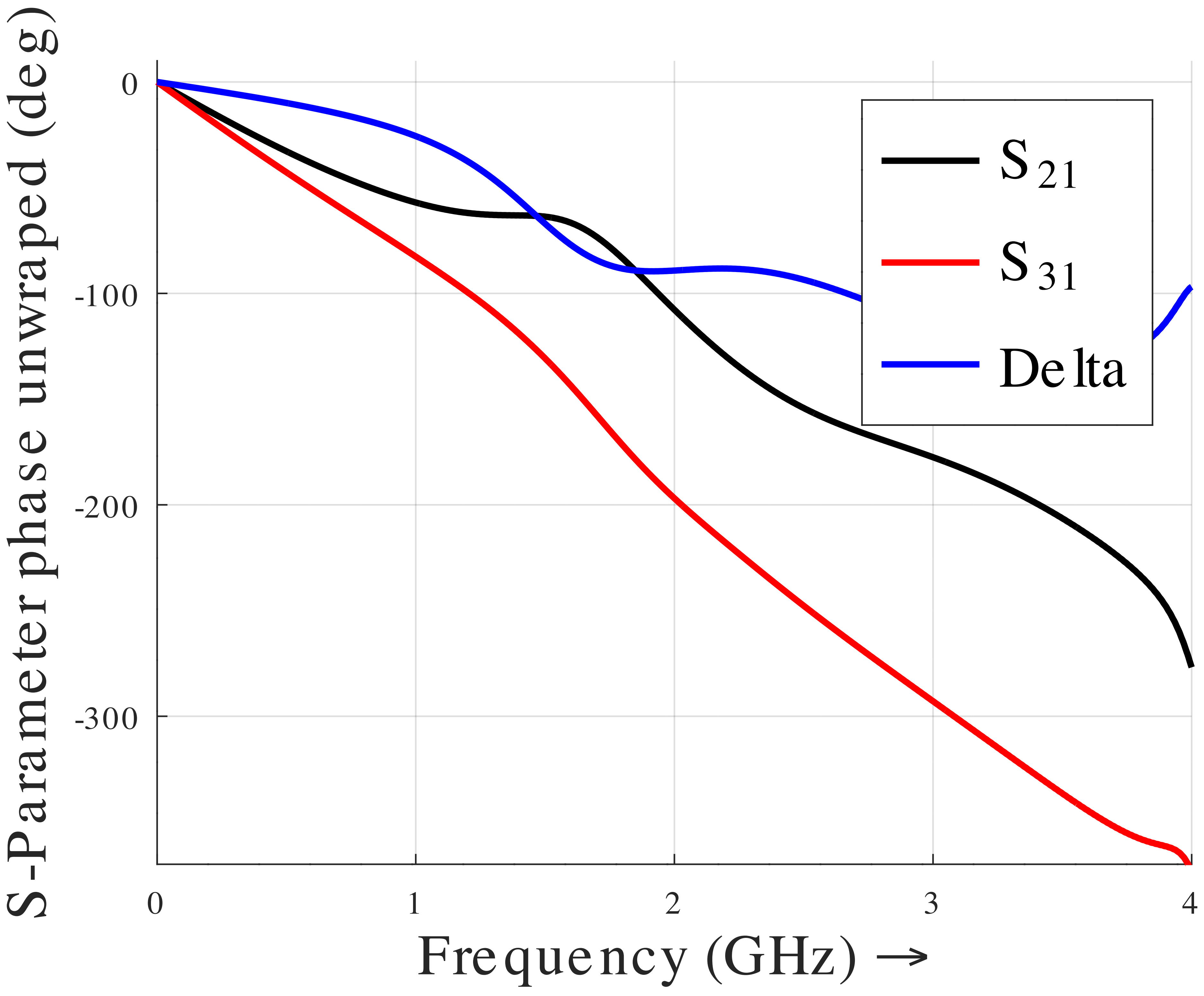
Magnitude of E-Field in Substrate



ParaView

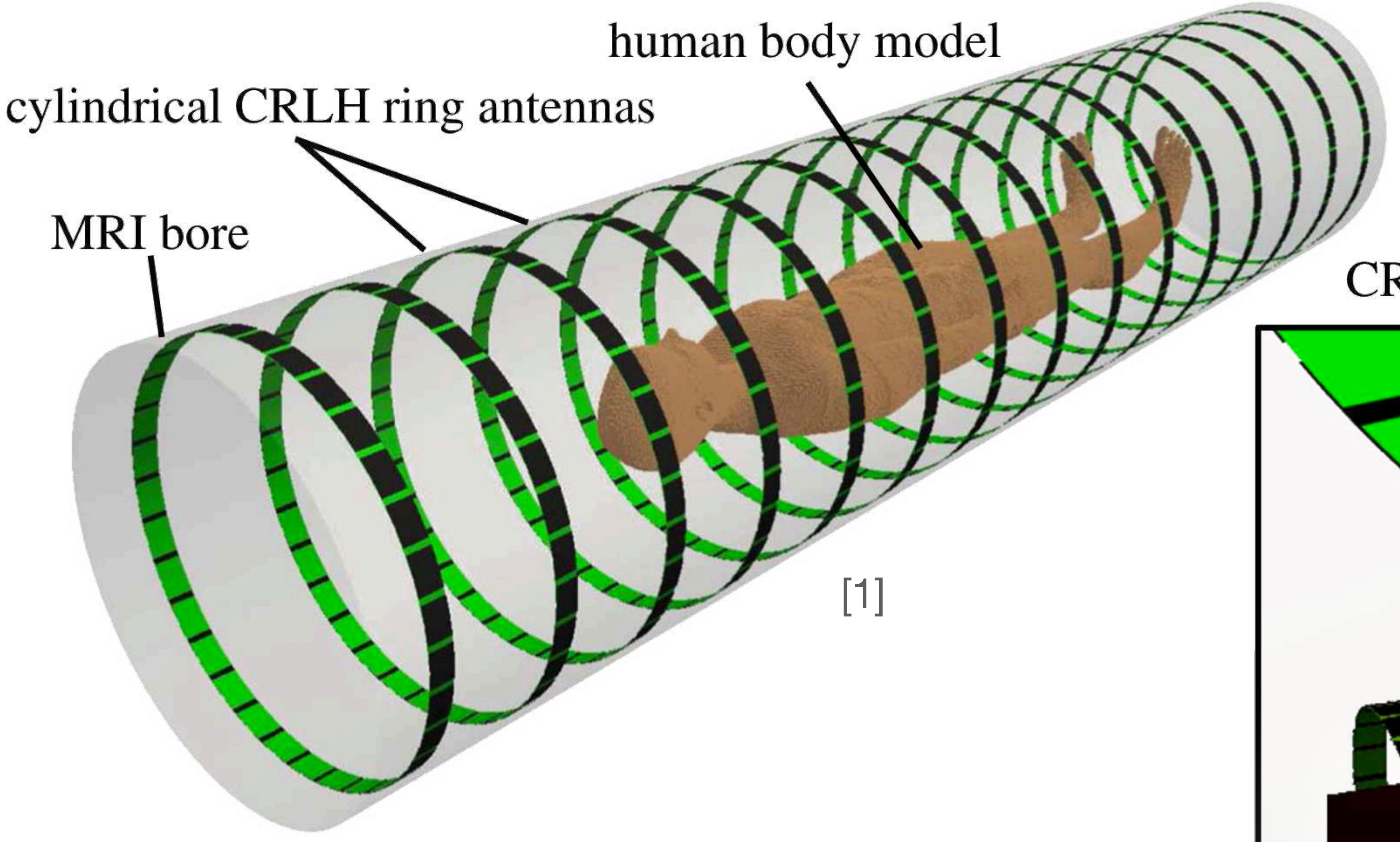
S-Parameter Results



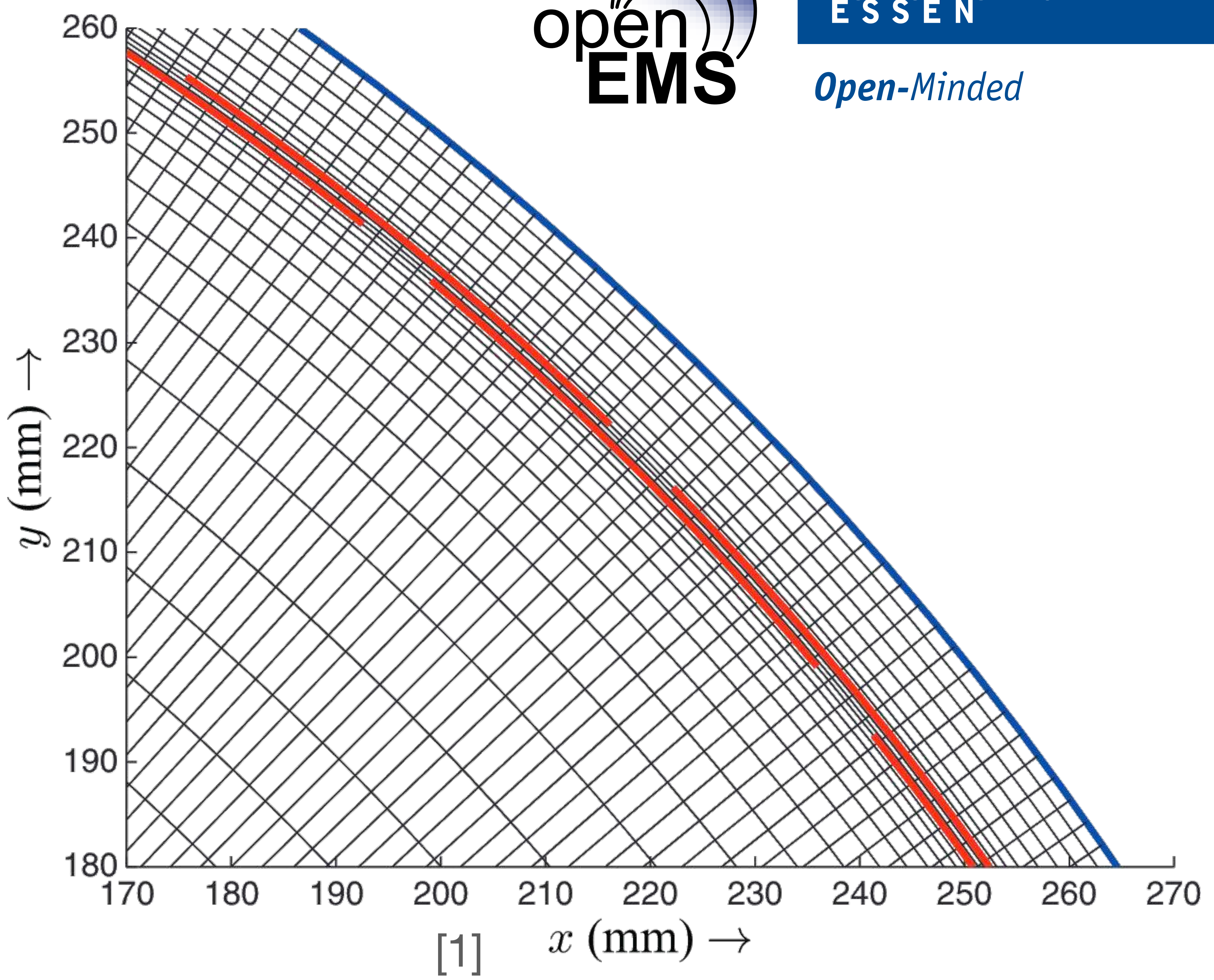


Showcases

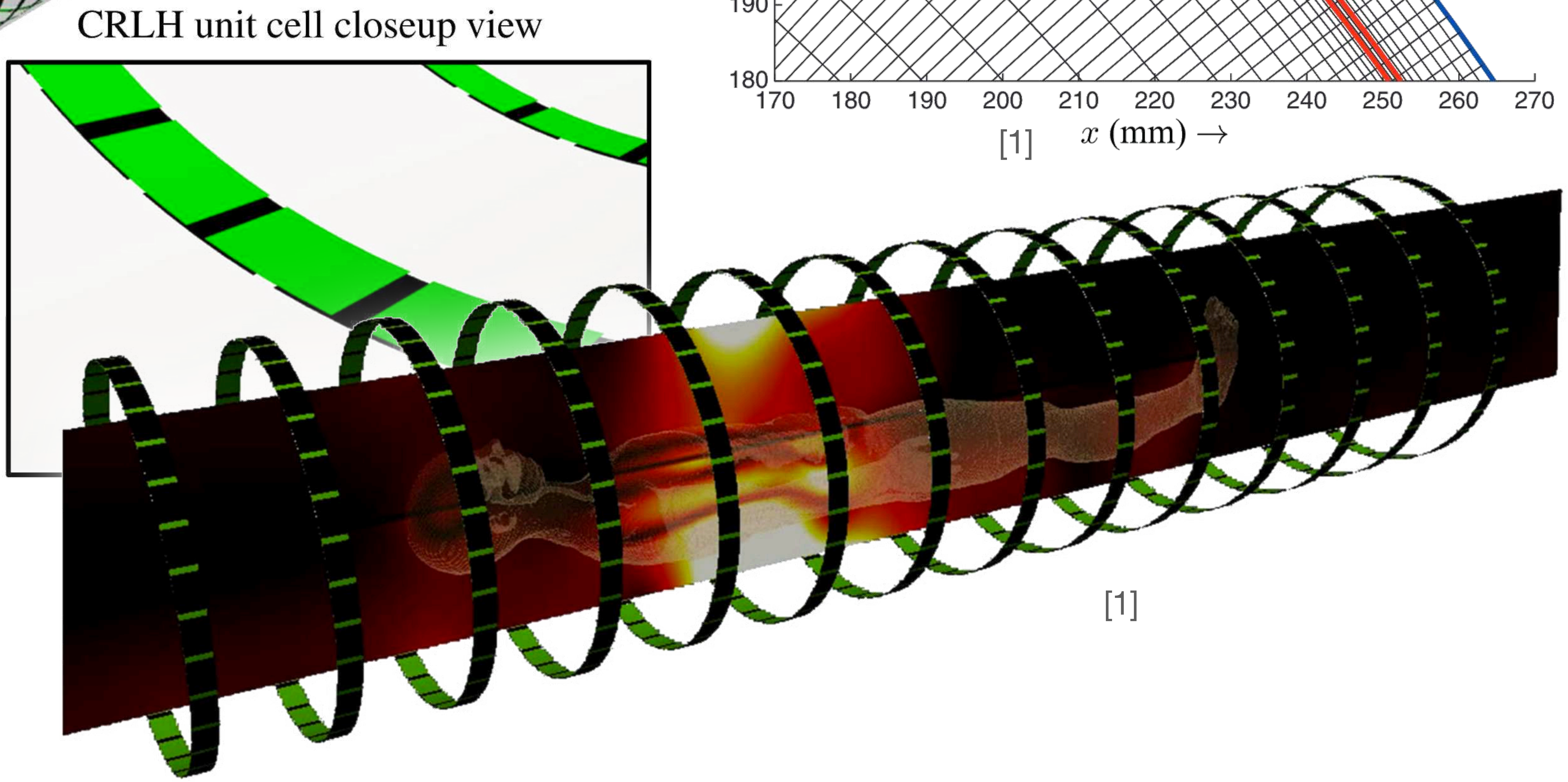
meta material resonators for 7T-MRI



[1]



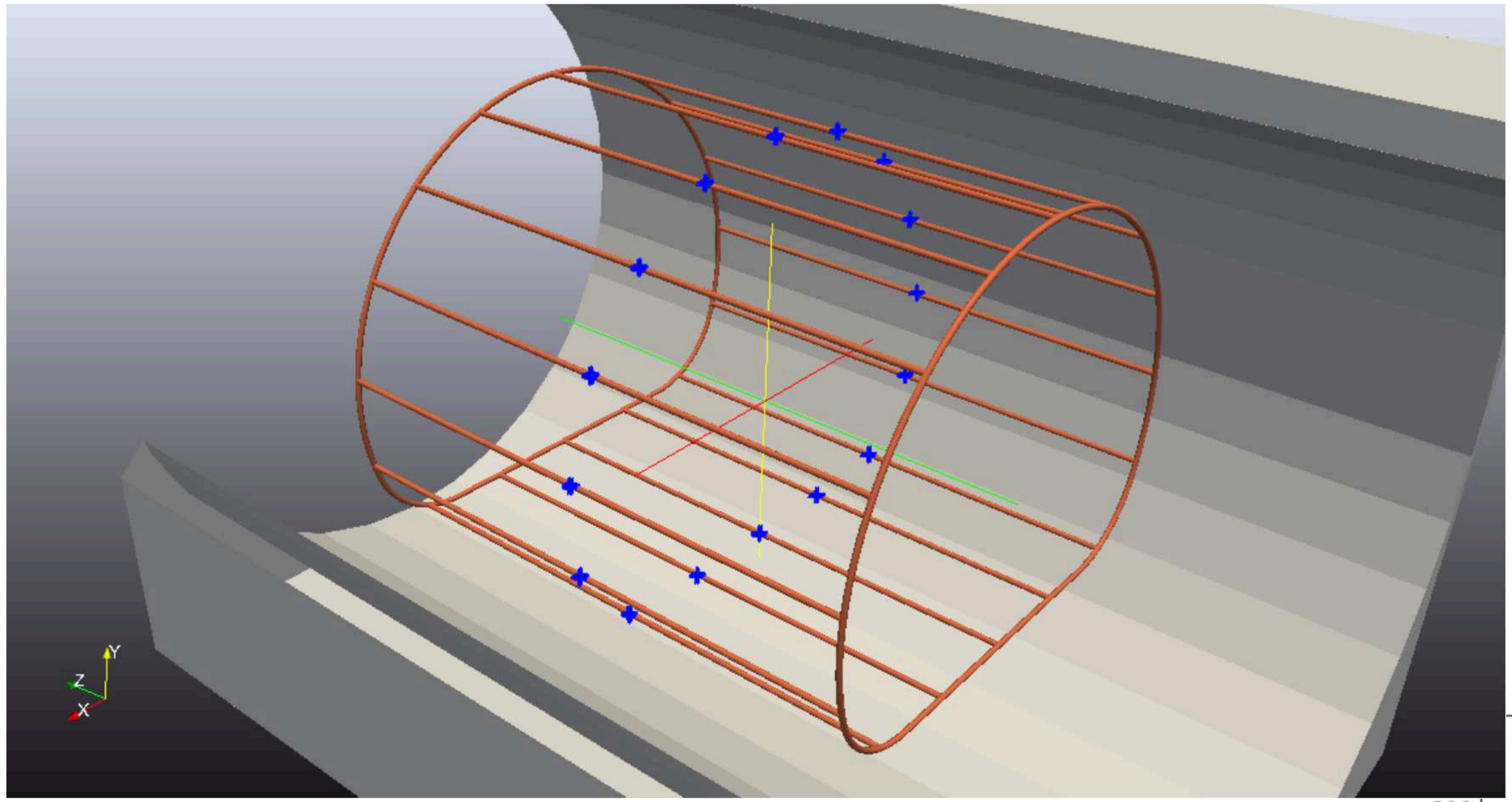
[1]

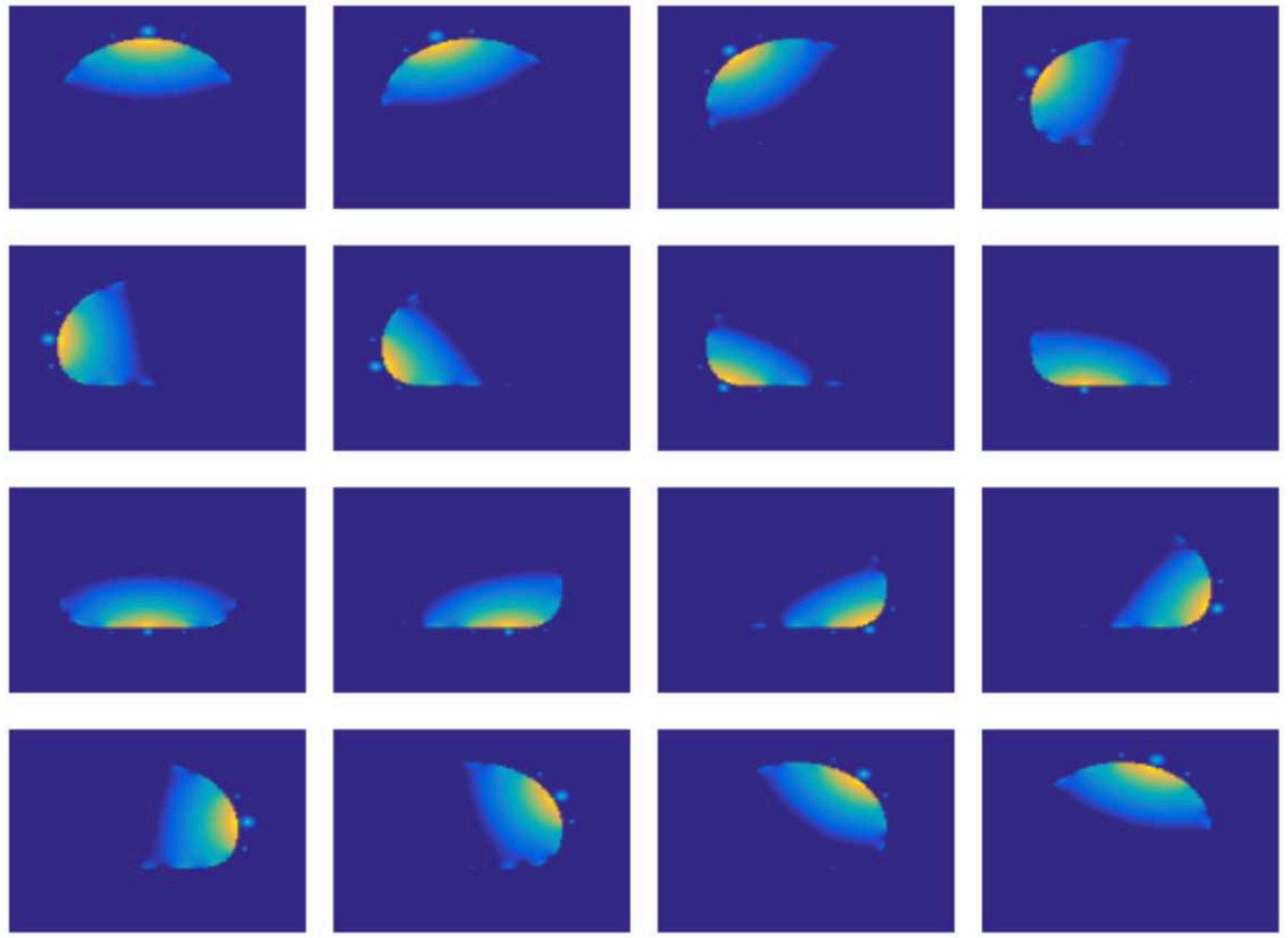


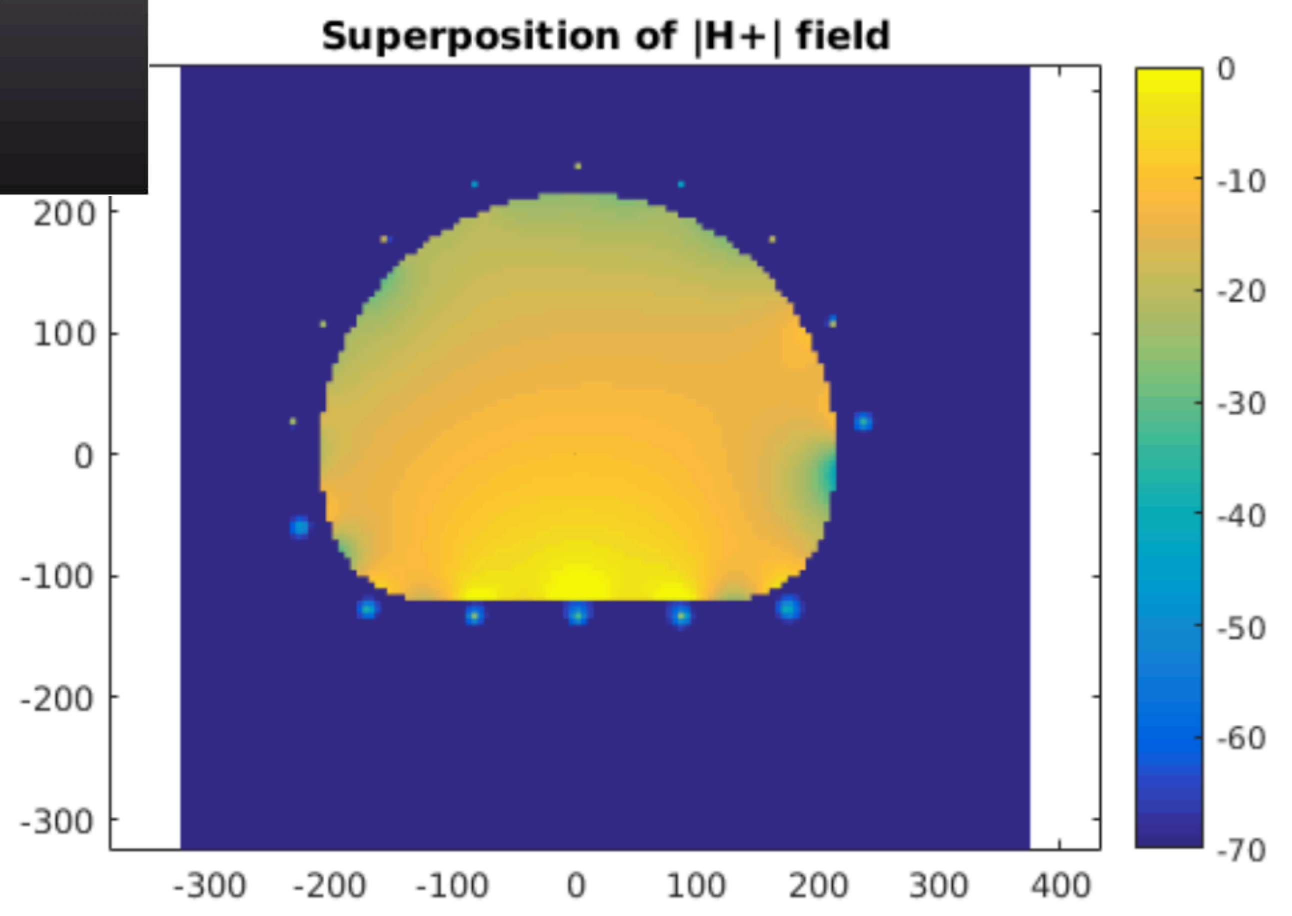
[1]

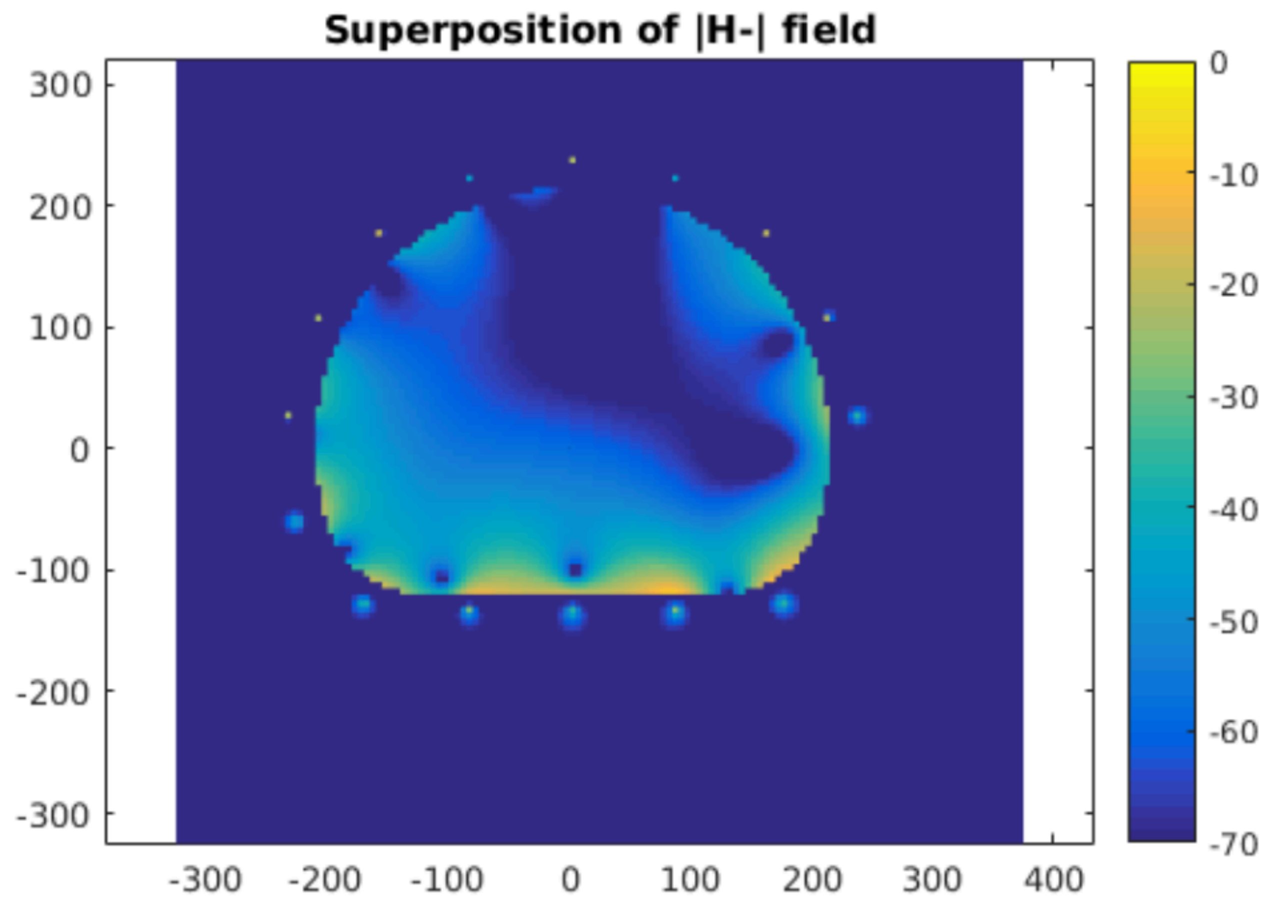
Showcases

flatened birdcage coil for 7T-MRI [4]

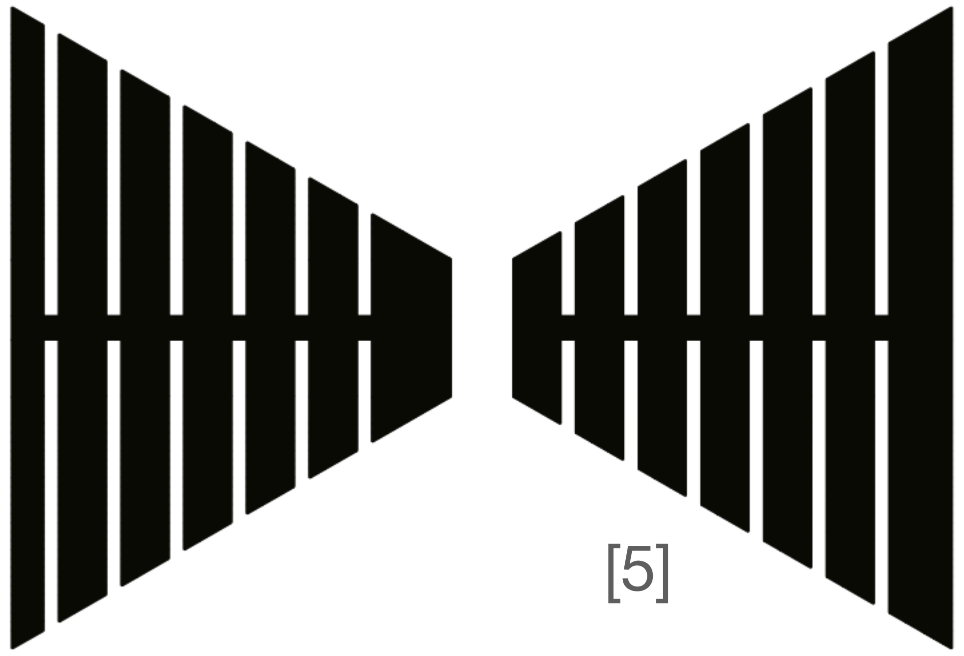




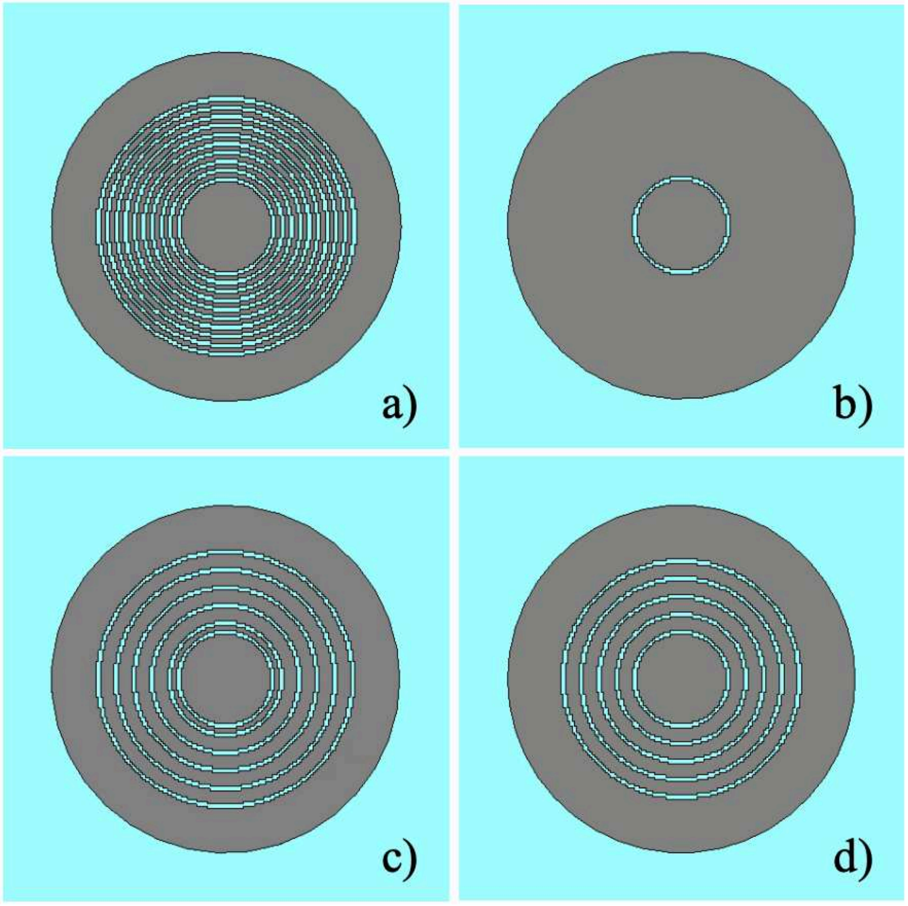




OpenEMS in Education at UDE

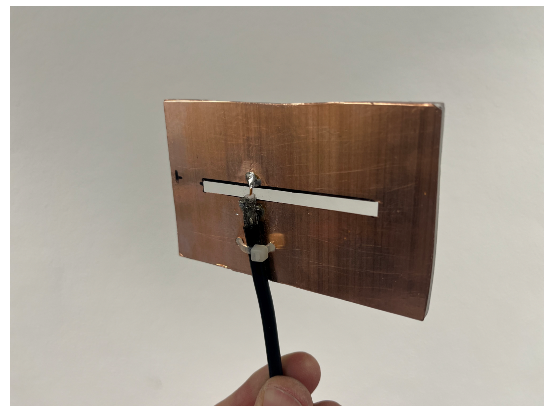


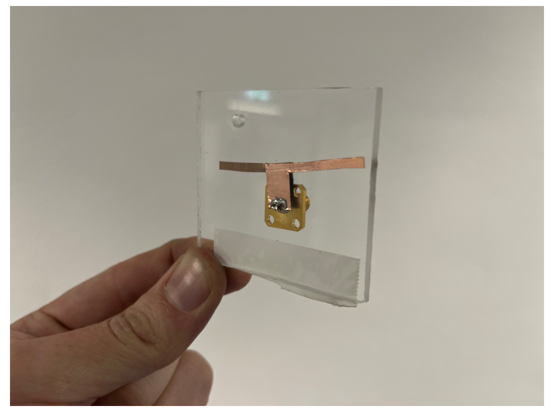
[5]

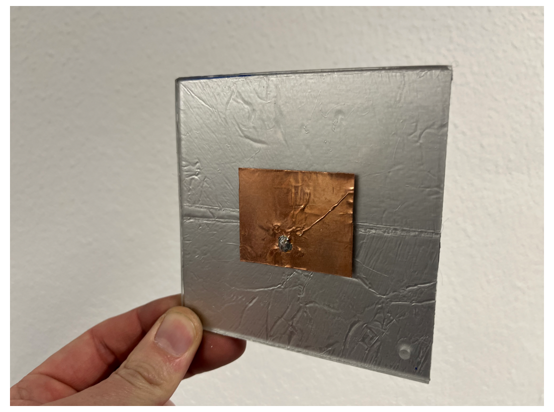


[6]

- Within remote Master degree program (*Hochfrequenz und Photonik Fernpraktikum*)
  - Students decide on Chipless RFID-Tag design from given list of papers
  - Task is to show functionality of at least three 6-Bit tags with varying codes
  - Simulations are done in openEMS (due to benefits of open-source)
- Included in Computational Electromagnetics course as an example FDTD solver
- Possible choice for antenna simulation during workshop part of Antennas for Communications course







openEMS | UNIVERSITÄT DUISBURG ESSEN | Open-Minded

Ways to Import | Export Structure

Export

- Structure is defined in CSXCAD
  - Export (currently) possible to
    - STL
    - Gerber (RS-274X)
    - KiCAD footprint
      via open source tool
      matthuszagh / pyems
- EM-Filed export to VTK, HDF5
- S-Parameter to e.g. touchstone

Import

- Inhomogenious materials
  - Import via CSXCAD as spacial dependent disc material
  - HDF5 File
- PCB-Design from KiCAD
  - Import via open source tool
    jcyrax / pcbmodelgen
- STL
- All imports need to fit CSXCAD

Summary

- Open EMS is a versatile EM solver
- Cartesian and cylindrical coordinate systems

➡ Advantageous:

✓ All benefits of open-source — freely available, no complicated licenses

✓ Several aspects of EM-simulations can be learned — meshing, boundary conditions, simulation domain

✓ Structure can easily be parameterised (

✓ FDTD enables broad band solutions with one simulation run

➡ Cons:

- Meshing is (for now?) to be done (primarily) by hand
- Single frequencies can not (easily) be simulated

★ Outlook

- Simulation speed improvement to more complex EC-FDTD implementation
- Interconnection with layout and circuit design tools

UNIVERSITÄT DUISBURG ESSEN

Open-Minded

Thank you

Feel free to clone, fork and contribute: https://www.openems.de

Jan Taro Svejda — jan.svejda@uni-due.de

General and Theoretical Electrical Engineering (ATE),
University of Duisburg-Essen,
CENIDE – Center for Nanointegration Duisburg-Essen,
D-47048 Duisburg, Germany

[1] Liebig Thorsten, Rennings Andreas, Held Sebastian, and Erni Daniel, "openEMS – a free and open source equivalent-circuit (EC) FDTD simulation platform supporting cylindrical coordinates suitable for the analysis of traveling wave MRI applications," Int. J. Numer. Model., vol. 26, no. 6, pp. 680–696, 2013, doi: 10.1002/jnm.1875.
[2] A. Rennings, "Elektromagnetische Zeitbereichssimulationen innovativer Antennen auf Basis von Metamaterialien," Dr.-Ing. Dissertation, Universität Duisburg-Essen, Fak. IW / ATE, Duisburg, DE, 2008.
[3] P. Russer, U. Siart, and W. J. R. Hoefer, Eds., Time domain methods in electrodynamics: a tribute to Wolfgang J.R. Hoefer. in Springer proceedings in physics, no. 121. Berlin: Springer, 2008.
[4] Gregor Dziuk, Entwurf und Aufbau von nicht-kreisförmigen Birdcage-Resonatoren für die X-Kern MRT bei 7 Tesla. Master thesis, (Master WiIng), Fachgebiet Allgemeine und Theoretische Elektrotechnik (ATE), Universität Duisburg-Essen, Jan. 8, 2016.
[5] L. Xu and K. Huang, "Design of Compact Trapezoidal Bow-Tie Chipless RFID Tag," International Journal of Antennas and Propagation, vol. 2015, pp. 1–7, 2015, doi: 10.1155/2015/502938.
[6] M. Martinez and D. Van Der Weide, "Compact slot-based chipless RFID tag," in 2014 IEEE RFID Technology and Applications Conference (RFID-TA), Tampere, Finland: IEEE, Sep. 2014, pp. 233–236. doi: 10.1109/RFID-TA.2014.6934234.