ElmerGUI

Mikko Lyly

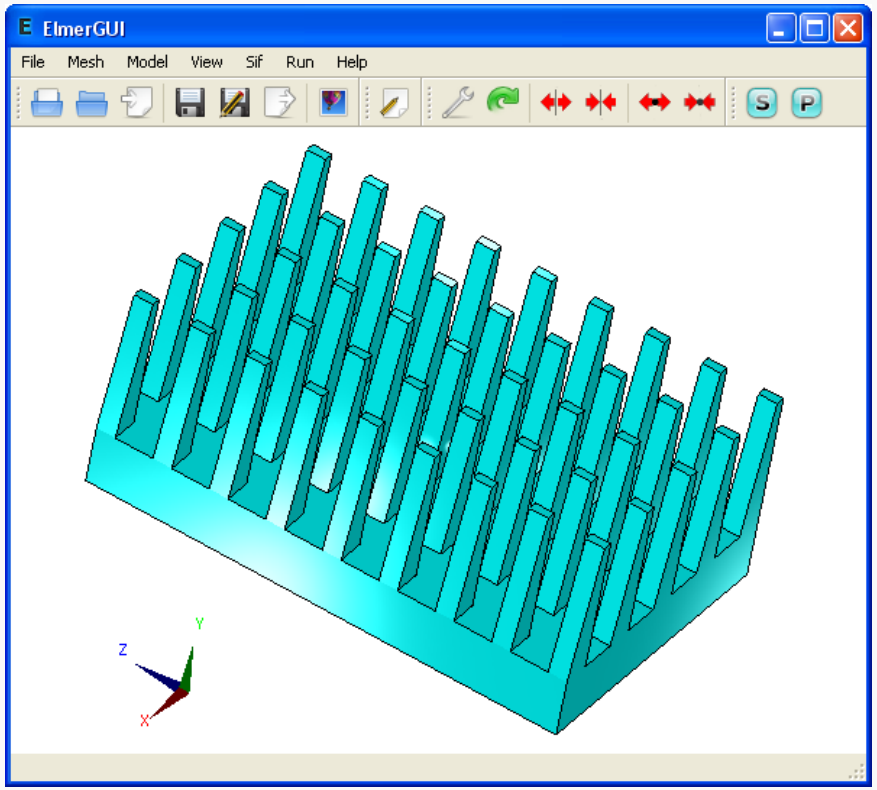
CSC - IT Center for Science
P.O. Box 405
FI-02101 Espoo, Finland

October 20, 2009

Outline

Introduction

ElmerGUI is a graphical user interface for the Elmer software suite. It is capable of importing finite element mesh files, generating finite element partitionings for CAD models, setting up PDE-systems to solve, and exporting model data for ElmerSolver and ElmerPost to visualize.



CSC

Getting ElmerGUI

ElmerGUI is available for all major desktop environments. There are precompiled binaries for Windows, Mac, and some Linux distributions. It is also possible to build the GUI from GPL licenced source:

- Windows (binaries): SourceForge (stable) or nic.funet.fi (latest)
- Linux (sources): Subversion repository at SourceForge.net
- Mac (binaries): Kindly maintained by Trueflaw Ltd.

The main download site is http://sourceforge.net/projects/elmerfem/

See also http://www.nic.funet.fi/pub/sci/physics/elmer/bin/windows/

Work flow

- Modeling the geometry with external tools:
  - GiD - powerful reasonably priced CAD program and mesh generator
  - Gmsh - open source tool for CAD modeling and mesh generation
  - Salome - GUI for OpenCASCADE, several built in mesh generators
- Geometry or mesh import followed by (re)meshing
- Setting up the PDE (equations, params, boundary conditions, ...)
- Generating and saving the Solver Input File
- Launching ElmerSolver as an external process
- Postprocessing the results (ElmerPost or VTK)

Demonstration

Demo.

Mesh generators

ElmerGUI has two built-in mesh generators: ElmerGrid and Netgen. It is also possible to use Tetgen as a plugin:

- ElmerGrid - structured mesh generator and file format converter
- Netgen - high quality Delaynay triangulations in 2D and 3D
- Tetgen - fast optional generator for 3D Delaunay meshing

Mesh generator is selected automatically based on the input file format.

Mesh and geometry input files

Mesh and geometry input files are handled by the menu item

File → *Open...*

The following file formats are supported:

- .grd - the native input file format for ElmerGrid
- .stp - STEP geometry file (needs OpenCASCADE)
- .igs - IGES geometry file (needs OpenCASCADE)
- .brep - BREP geometry file (needs OpenCASCADE)
- .msh - gmsh input file format (handled by ElmerGrid)
- .in2d - Netgen's 2D file format (handled by Netgen)

Mesh and geometry input files

- .mphtxt - Comsol multiphysics file format (ElmerGrid)
- .off - object file format (needs Tetgen plugin)
- .ply - polygon file format (needs Tetgen plugin)
- .poly - piecewise linear complex (needs Tetgen plugin)
- .smesh - native PLC format for Tetgen (plugin needed)
- .stl - stereo litography format (handled by Netgen or Tetgen)
- .fdneut - Fidap netral file format
- .unv - Ideas universal mesh file format

Mesh files

Native Elmer mesh files are handleded by the menu items

$$File \rightarrow Load\ mesh...$$

and

$$File \rightarrow Save\ (as)...$$

A native mesh consists of the following four text files located side-by-side in the same mesh directory (detailed description can be found from the SolverManuals):

- mesh.header
- mesh.nodes
- mesh.elements
- mesh.boundary

Project file format

An ElmerGUI-project consists of the geometry input file, native Elmer mesh files, Solver Input File (.sif), and an XML-file containing the state of the GUI:

- egproject.xml
- geom.file
- case.sif
- mesh.header
- mesh.nodes
- mesh.elements
- mesh.boundary

N.B.: ElmerGUI project files should not be edited by hand.

Postprocessing

The internal postprocessor of ElmerGUI is based on the Visualization Toolkit (VTK):

$$Run \rightarrow Postprocessor\ (VTK)...$$

The module is an alternative and complementary tool for ElmerPost. Most of its functionality is available also in Paraview, the graphical user interface for VTK.

One of the key features of the internal postprocessor is its scriptability. The module provides ECMAScript bindings to all classes and methods for which there is a menu item. That is, everything that can be done interactively with mouse, can also be automated by scripting. The scripting console is activated from the menu as

$$Edit \rightarrow ECMAScript\ console\ ...$$

Postprocessing

The base classes available for scripting are the following (refer to the docs for a detailed description):

```
egp // ElmerGUI post processor
matc // MATC language interpreter
preferences // controls for preferences
surfaces // controls for surface plots
vectors // controls for vector fields
isoContours // controls for isocontours
isoSurfaces // controls for isosurfaces
streamLines // controls for streamlines
colorBar // cotrols for the colorbar
timeStep // controls for transient results
text // text annotation
```

Postprocessing

Each class provides a number of methods for specific purposes (again, refer to the docs for more details). The class "egp" for example provides

```
bool ReadPostFile(QString); // read result file
...
void SetSurfaces(bool); // show/hide surfaces
...
```

The methods are accessed from the console as

```
qs> egp.ReadPostFile("MyResultFile.ep");
qs> egp.SetSurfaces(true);
```

The above two lines will read in the post file "MyResultFile.ep" and switch on the surface rendering mode.

Postprocessing

The whole command sequence can be stored in a script file (with file extension "qs"). For example "script.qs":

```
egp.ReadPostFile("case.ep")
egp.SetSurfaces(true)
for(var i = 0; i < 180; i++)
{
   egp.RotateY(2.0)
   egp.ResetCamera()
   egp.Render()
   egp.SavePngFile("frame" + i + ".png")
}
```

The script file is executed as

```
qs> egp.Execute("script.qs")
```

Definitions

The menu entries (equation, material, boundary conditions, ...) of ElmerGUI are customizable. The custom definitions for the so called "dynamic editors" can be found from the directory $ELMER_HOME/bin/edf/ The directory contains several XML-files, which describe the menu items related to different PDEs and instructions for the sif-generator to export the Solver Input File for ElmerSolver.

Definitions

An ElmerGUI-definition file has the following structure:

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE edf>
<edf version="1.0">
[PDE block]
[PDE block]
...
[PDE block]
</edf>
```

Usually, there is only one PDE block in a custom definition file.

Definitions

Each PDE block in the definition file has the following structure:

```
<PDE Name="My equation">
<Name>
My equation
</Name>
...
<Equation>
[Widget block]
</Equation>
...
<Material>
[Widget block]
</Material>
...
```

Definitions

```
<BodyForce>
[Widget block]
<BodyForce>
...
<InitialCondition>
[Widget block]
</InitialCondition>
...
<BoundaryCondition>
[Widget block]
</BoundaryCondition>
</PDE>
```

Definitions

Each Widget block in the PDE block has the following basic structure:

```
<Parameter Widget="Label">
<Name> My label </Name>
</Parameter>
...
<Parameter Widget="Edit">
<Name> My edit box </Name>
<Type> Integer </Type>
<Whatis> Meaning of my edit box </Whatis>
</Parameter>
...
```

Definitions

```
<Parameter Widget="CheckBox">
<Name> My check box </Name>
<Type> Logical </Type>
<Whatis> Meaning of my check box </Whatis>
</Parameter>
...
<Parameter Widget="Combo">
<Name> My combo box </Name>
<Type> String </Type>
<Item> <Name> My 1st item </Name> </Item>
<Item> <Name> My 2nd item </Name> </Item>
<Item> <Name> My 3rd item </Name> </Item>
<Whatis> Meaning of my combo box </Whatis>
</Parameter>
```

Materials

The file $ELMERGUI_HOME/edf/egmaterials.xml defines the material database for ElmerGUI. The format of this file is the following:

```
<!DOCTYPE egmaterials>
<materiallibrary>
<material name="Air (room temperature)" >
<parameter name="Density" >1.205</parameter>
<parameter name="Heat conductivity" >0.0257</parameter>
<parameter name="Heat capacity" >1005.0</parameter>
<parameter name="Heat expansion coeff." >3.43e-3</parameter>
<parameter name="Viscosity" >1.983e-5</parameter>
<parameter name="Turbulent Prandtl number" >0.713</parameter>
<parameter name="Sound speed" >343.0</parameter>
</material>
```

Materials

```
<material name="Water (room temperature)" >
<parameter name="Density" >998.3</parameter>
<parameter name="Heat conductivity" >0.58</parameter>
<parameter name="Heat capacity" >4183.0</parameter>
<parameter name="Heat expansion coeff." >0.207e-3</parameter>
<parameter name="Viscosity" >1.002e-3</parameter>
<parameter name="Turbulent Prandtl number" >7.01</parameter>
<parameter name="Sound speed" >1497.0</parameter>
</material>
...
</materiallibrary>
```