GroIMP 1.5

A closer look at the new features

Michael Henke

Department Ecoinformatics, Biometrics and Forest Growth,
University of Göttingen, Germany

Tutorial and Workshop
"Functional-Structural Plan Modelling with GroIMP and XL"
combined with the 7th GroIMP user and developer meeting

Angers, 2015-05-06

GroIMP 1.5 - Part I

- added export to file function to Dataset and DatasetRef
- added *leaf3d, getMesh, triangulate* and *getAreaOfTriangulation* functions
- added snapshot function to 2d graph view
- added changes tab to help panel
- added check for updates button / auto check at program start
- added *list* and *help* command to the console
- added Help → Function Browser
- added Panel → Graph → Textual overview
- added Help → System Info
- added Command History to XL Console
- added *getSurfaceArea, getVolume* functions to all primitives
- added *getSurfaceArea, getVolume* functions to the library class to calculate the values for a subgraph

GroIMP 1.5 - Part II

- added functions to control view3d repaint
- added Library functions to calculate and visualize the convex hull
- added Library functions to calculate the XY-projection
- added references for spectra and light distributions
- added functions to visualize the physical light distribution
- added concepts for multi-scaled modelling and level-of-detail visualization
- added STL export
- added stereo image (anaglyph) view for Wireframe View3D
- added split view for Wireframe View3D
- added copy protection for models
- extended ”Show Examples” gallery
- general bug fixes

GroIMP 1.5 - Part II

- added functions to control view3d repaint
- added Library functions to calculate and visualize the convex hull
- added Library functions to calculate the XY-projection
- added references for spectra and light distributions
- added functions to visualize the physical light distribution
- added concepts for multi-scaled modelling and level-of-detail visualization
- added STL export
- added stereo image (anaglyph) view for Wireframe View3D
- added split view for Wireframe View3D
- added copy protection for models
- extended ”Show Examples” gallery
- general bug fixes

→ ’daily build’
http://ufgb966.forst.uni-goettingen.de/GroIMP/

added export to file function to Dataset and DatasetRef

- Example: Technics → ChartsDemo2.gsz

```
DatasetRef myChartXY = new DatasetRef("Scatter");
...

myChartXY.clear().setColumnKey(0,"Value");
...

public void exportData() {
    myChartXY.export(new FileWriter("../test.txt"));
}
```

added export to file function to Dataset and DatasetRef

- Example: Technics → ChartsDemo2.gsz

```
1 DatasetRef myChartXY = new DatasetRef("Scatter");
  ...
3
  myChartXY.clear().setColumnKey(0,"Value");
5 ...

7 public void exportData() {
      myChartXY.export(new FileWriter("../test
9 }
```

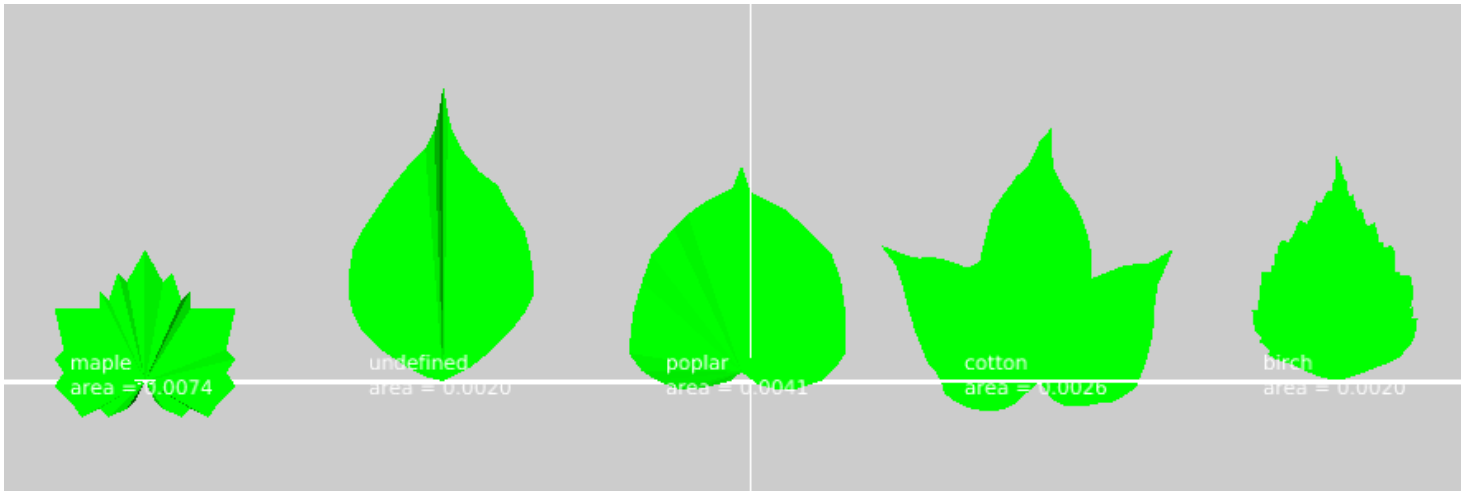
```
1   1.0345631
2   0.7027052
3   1.3539617
4   0.6167236
5   0.6495002
6   1.3293481
7   0.5251097
8   1.2287854
9   1.3880142
...
```

added *leaf3d, getMesh, triangulate* and *getAreaOfTriangulation* functions

- Show Examples → leaf3dDemo

1. hard coded 3-d leaves: *leaf3d, getAreaOfTriangulation*

```
for(int i : (0:DEFAULT_LEAF3D.length-1)) (
  Null(0.33*i,0,0) leaf3d(i)
  TextLabel(DEFAULT_LEAF3D_NAME[i]+"\narea = "+
      getAreaOfTriangulation(DEFAULT_LEAF3D[i])), ^
)
```



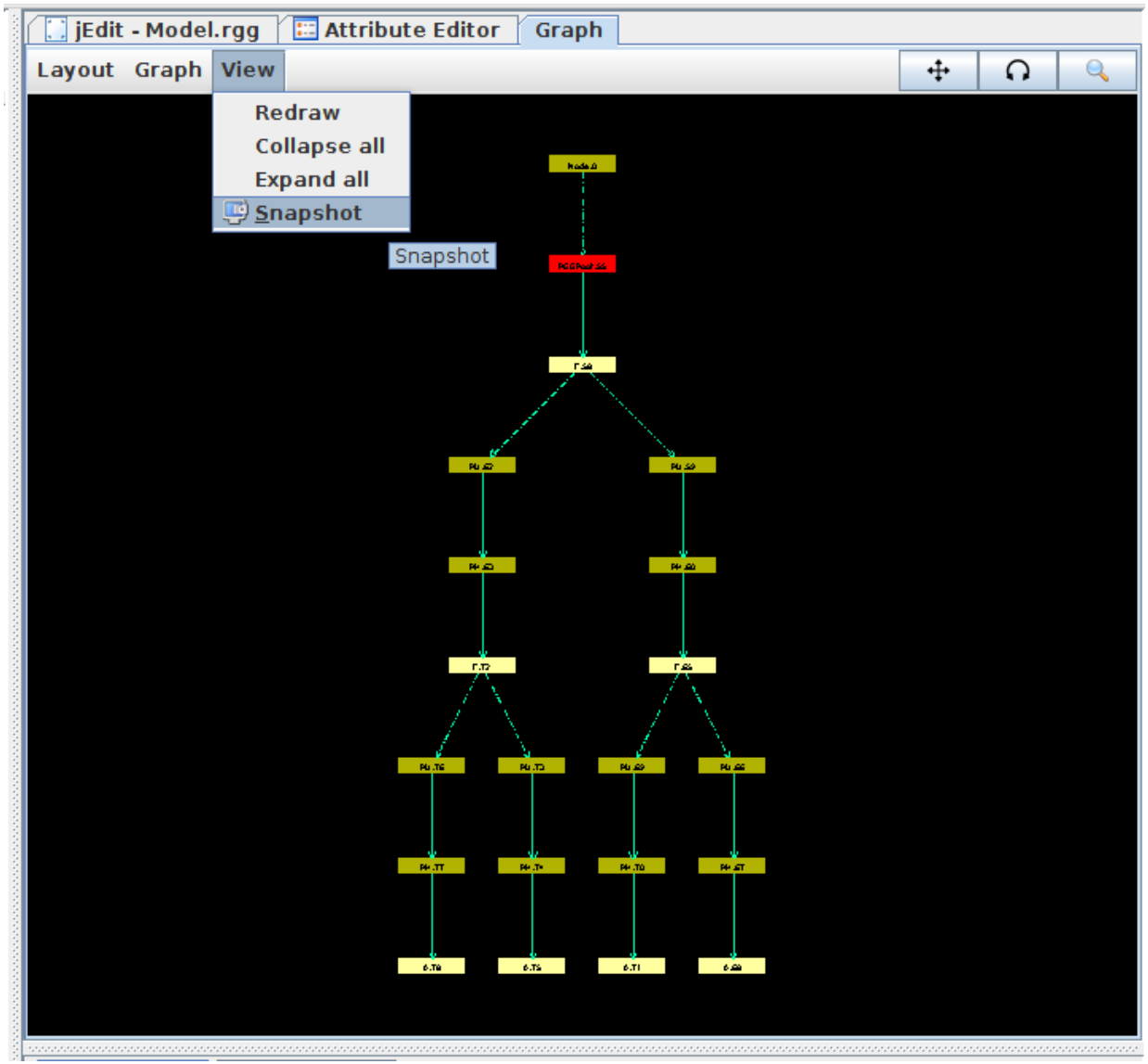
added *leaf3d, getMesh, triangulate* and *getAreaOfTriangulation* functions

- Show Examples → leaf3dDemo

2. object generation with and without triangulation: *triangulate, getMesh*

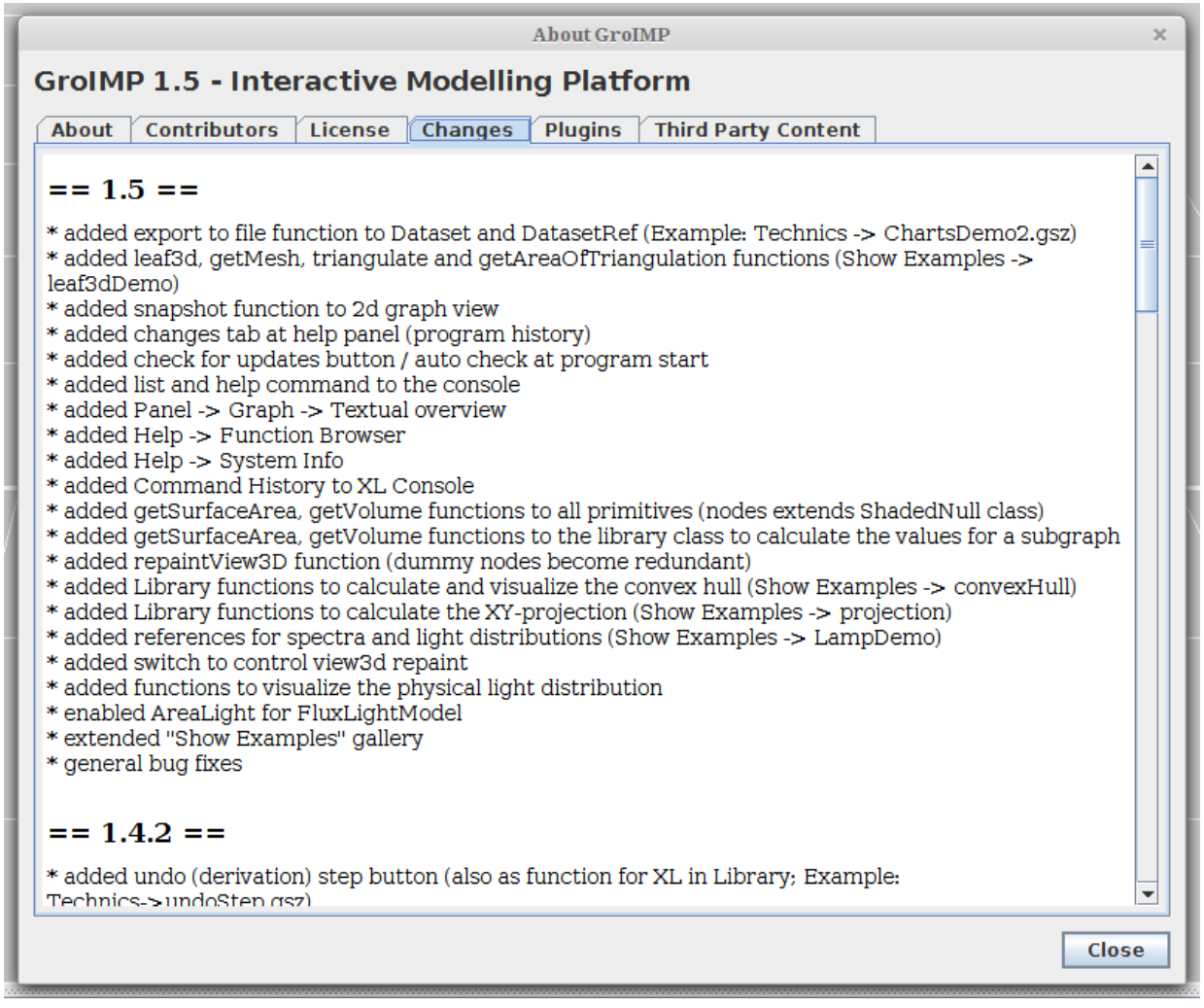
```
// direct input of points
{
  float[] points = {0,0,0, 1,0,0, 1,1,0};
  println("area of datapoint="+getAreaOfTriangulation(points));
}

// the points will be triangulated before applied to a MeshNode
triangulate(points).(setShader(new RGBAShader(1,0,0)))
```

added snapshot function to 2d graph view



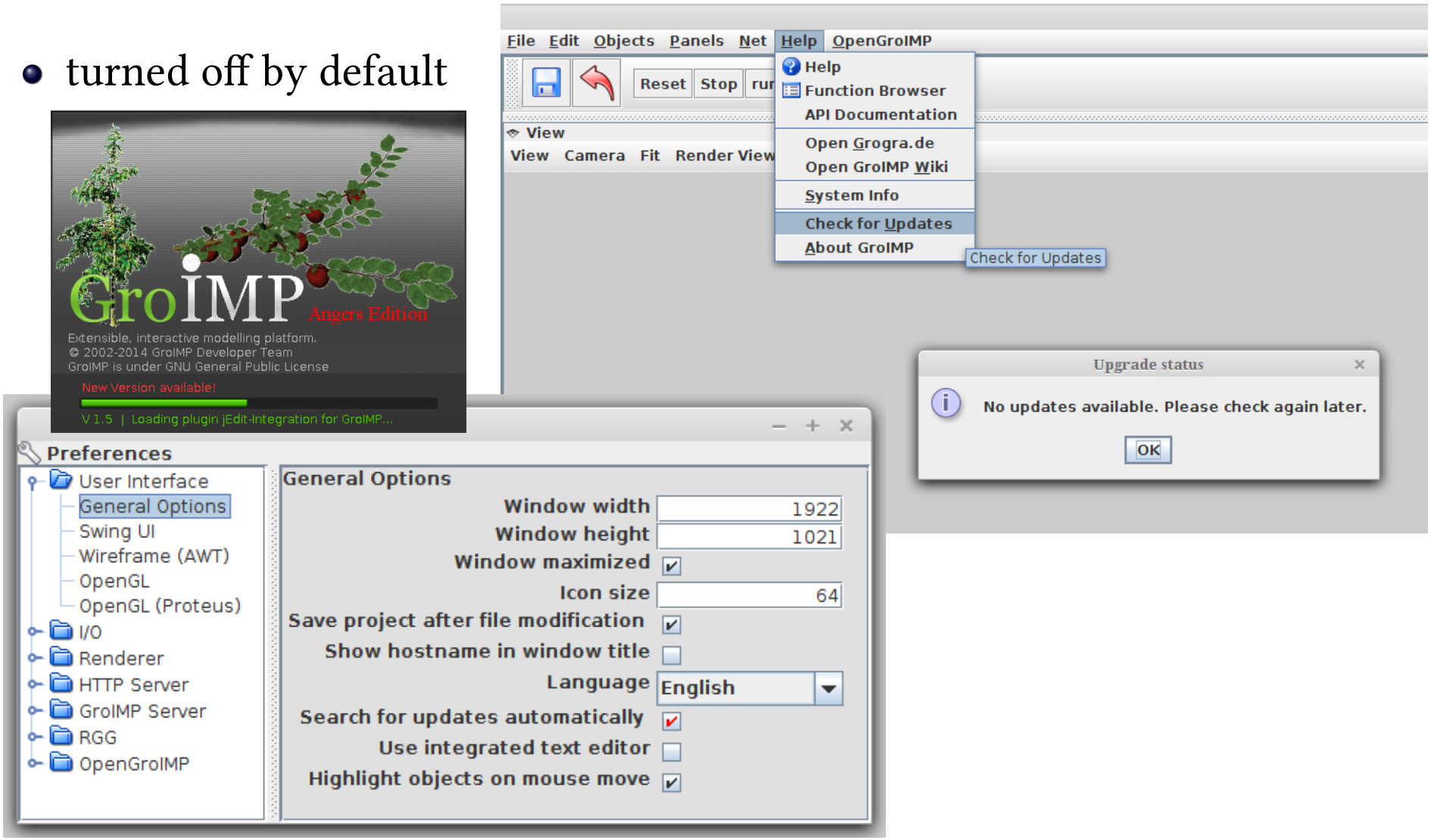
added changes tab to help panel

- program history, list of features



added check for updates button / auto check at program start

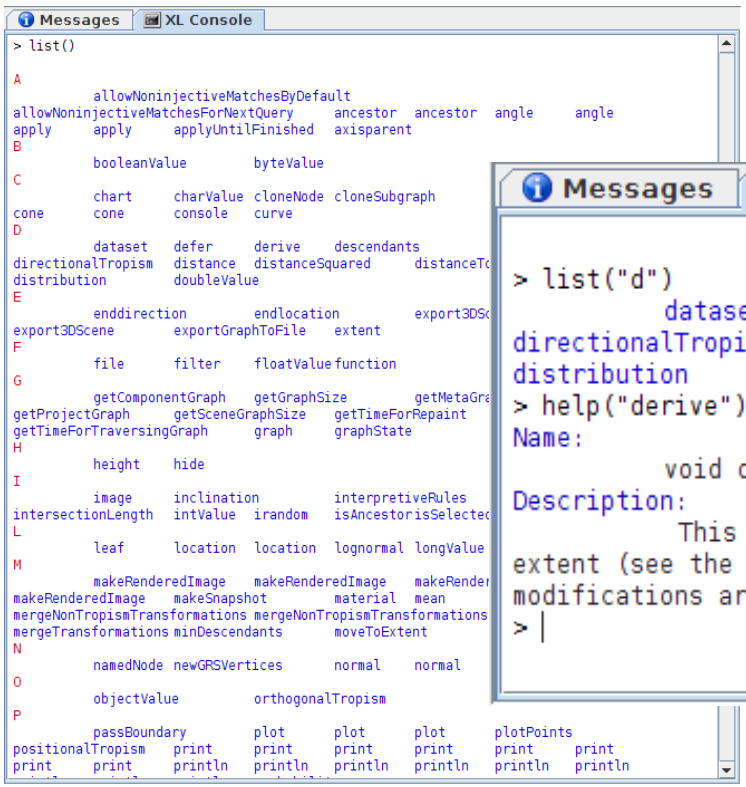
- turned off by default

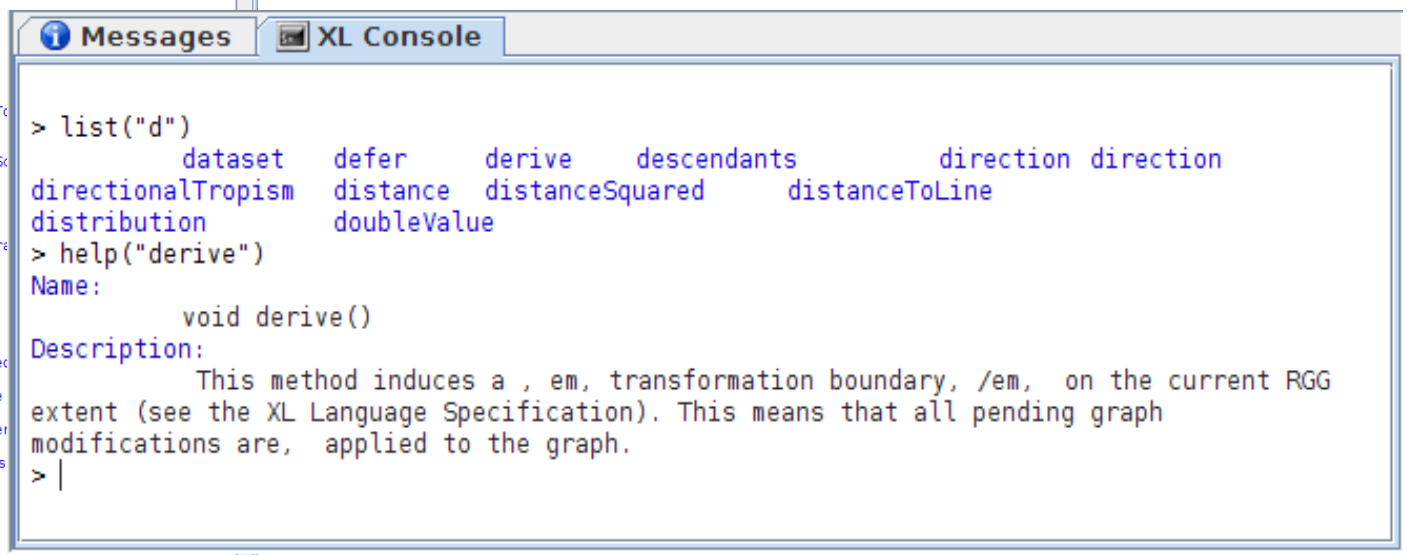


added *list* and *help* command to the console

- Overview and description of GroIMP commands via XL Console

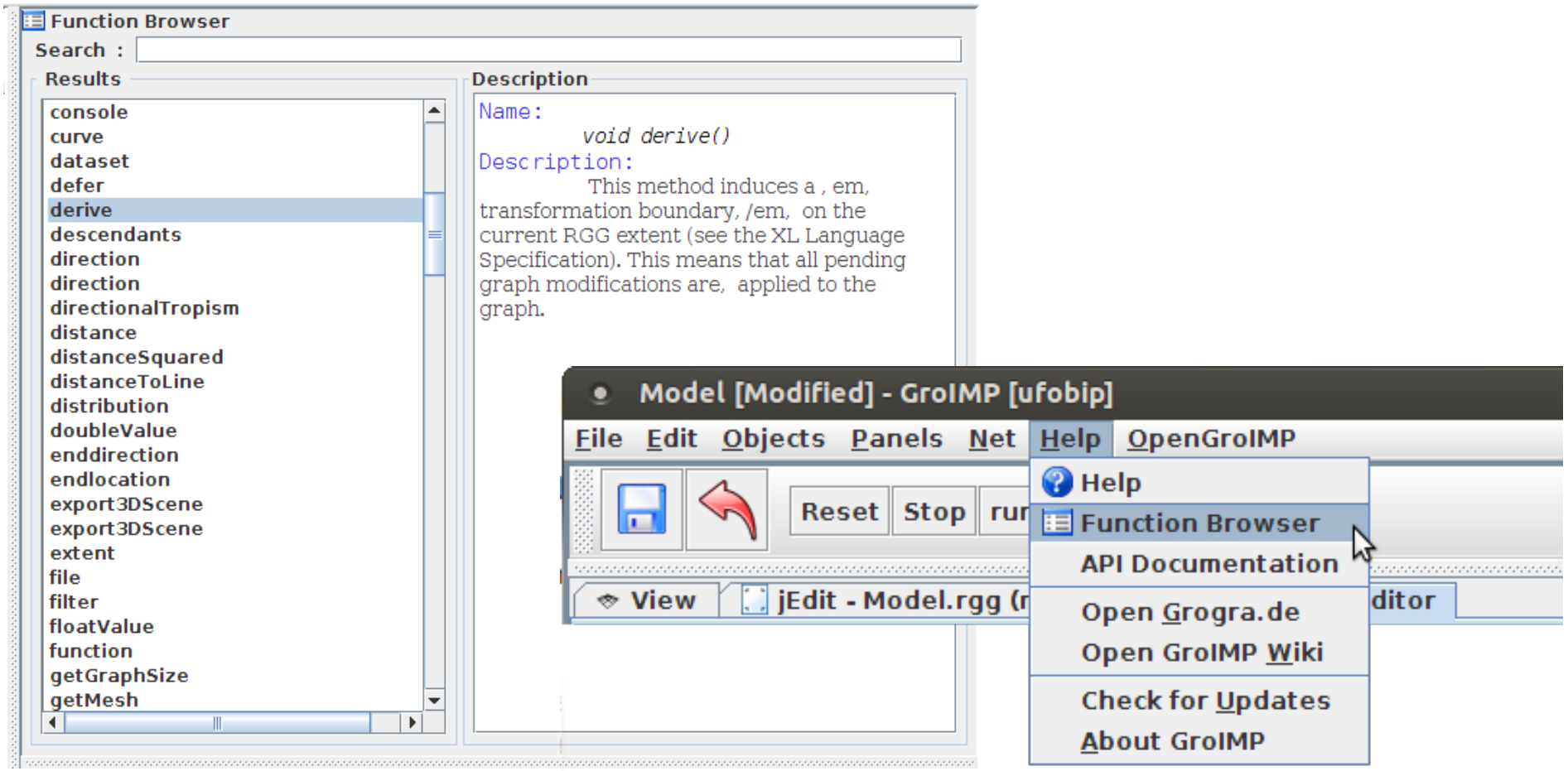
list() lists all available commands
list("key") lists all available commands starts with the specified key
help("key") lists the help of all command starts with the specified key





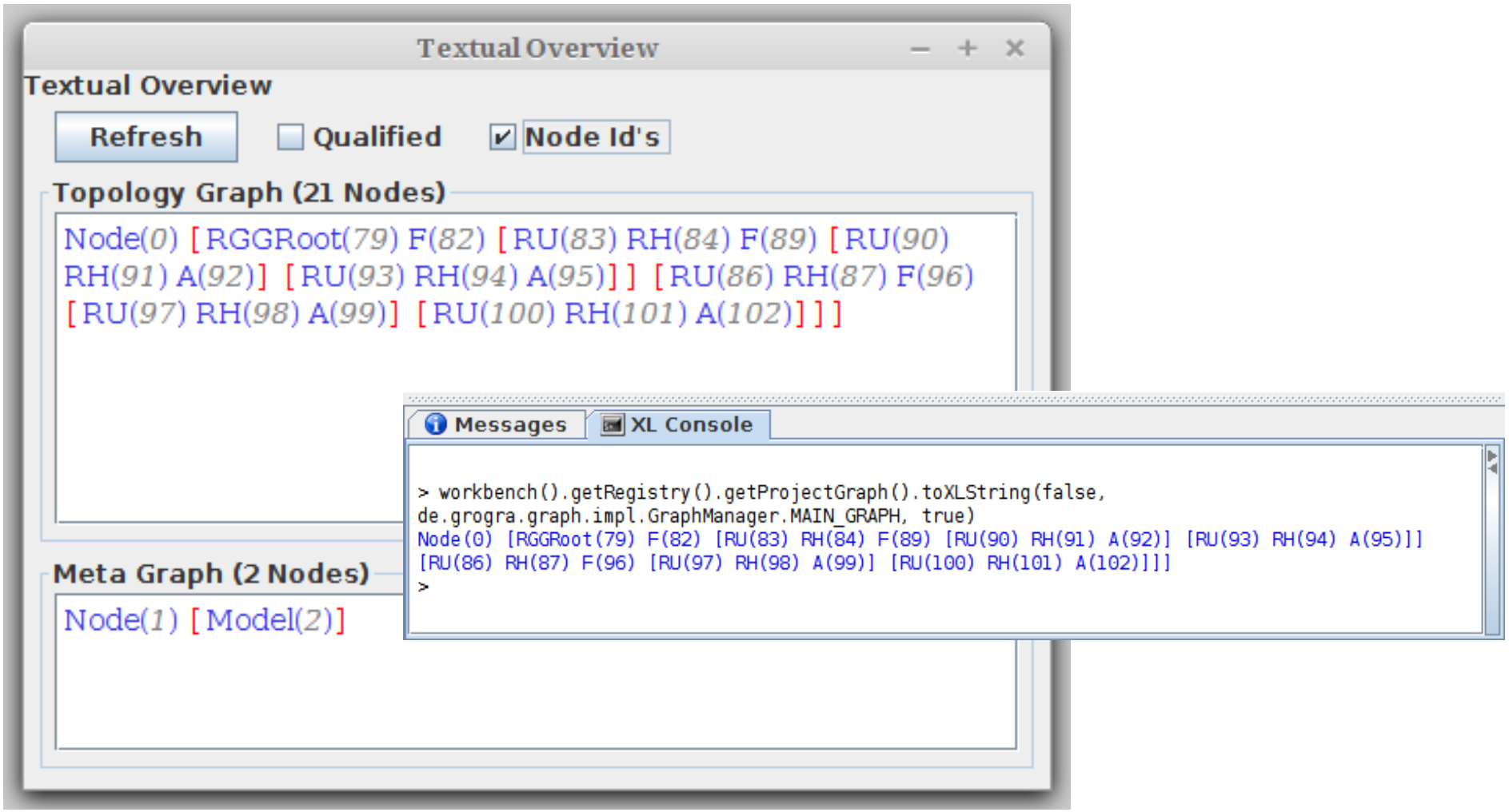
added Help → Function Browser

- … or use the Function Browser
- integrated search function



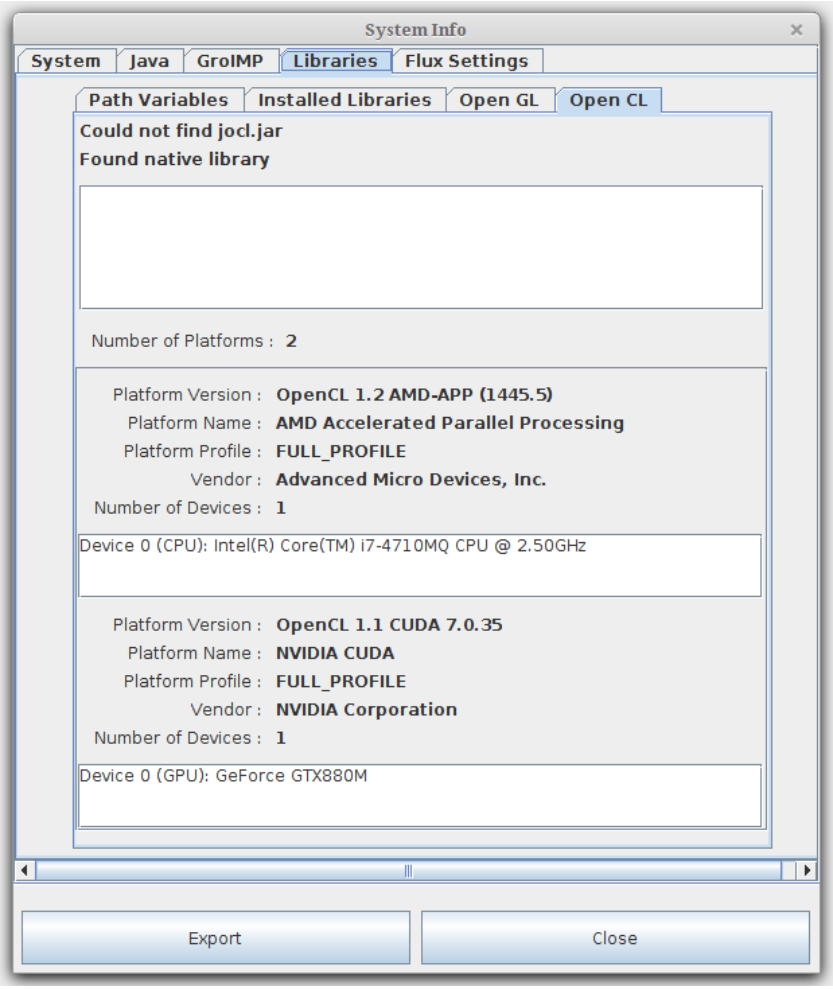
added Panel → Graph → Textual overview

- textual overview of internal graphs



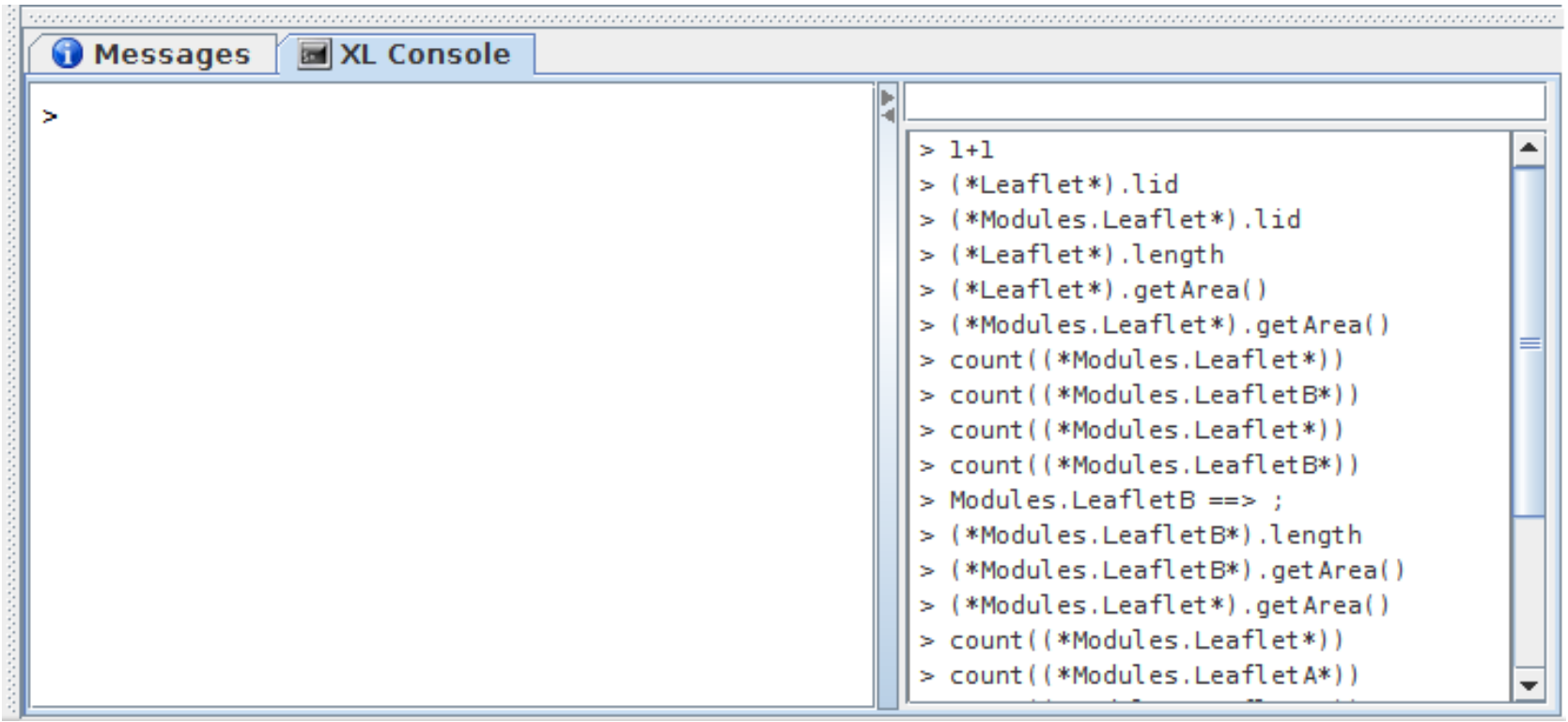
added Help → System Info

- infos about: system, installed libraries, HW, settings
- export function → error diagnosis



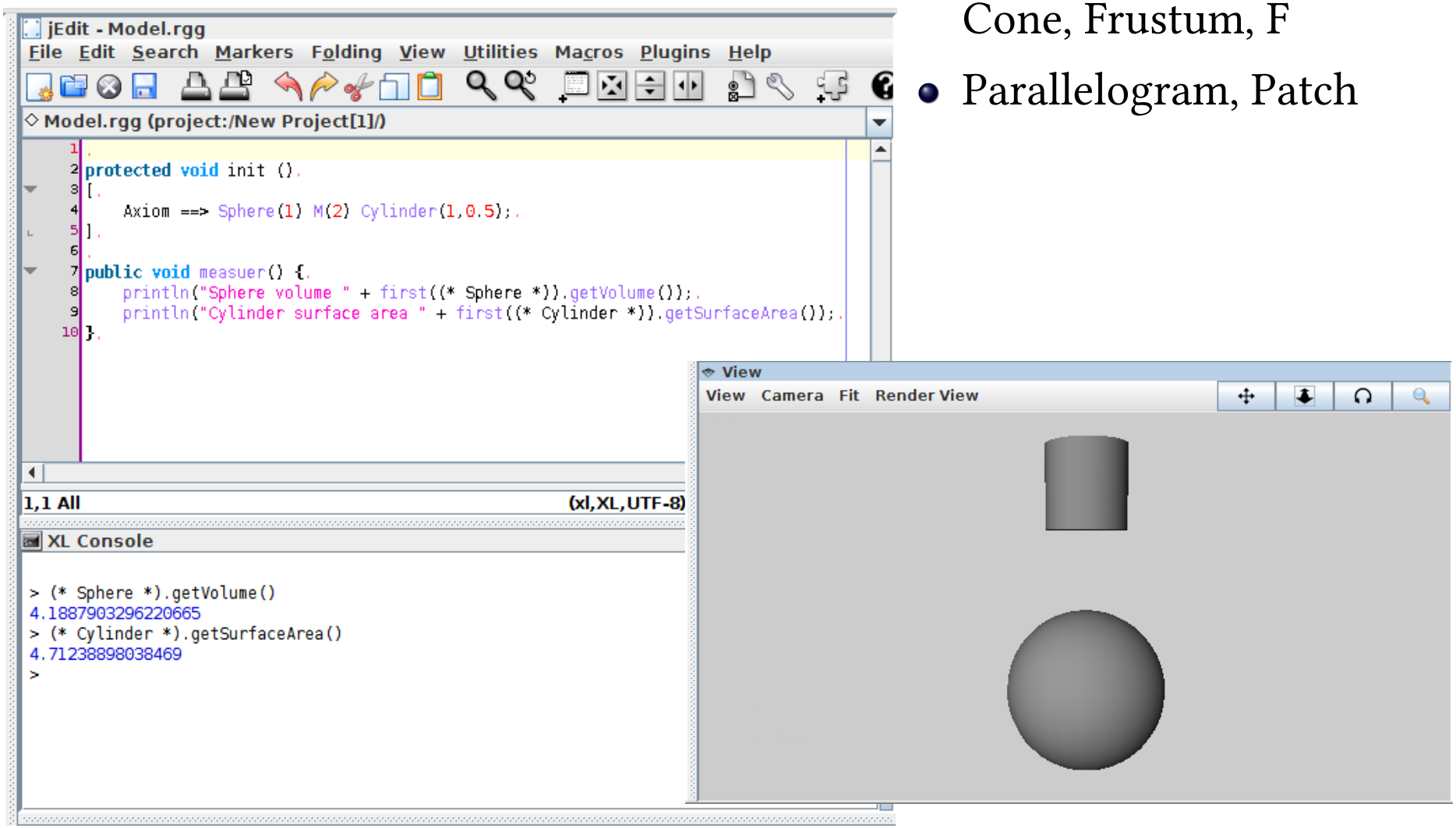
added Command History to XL Console

- history of all commands ever entered
- integrated search function
- can be hidden



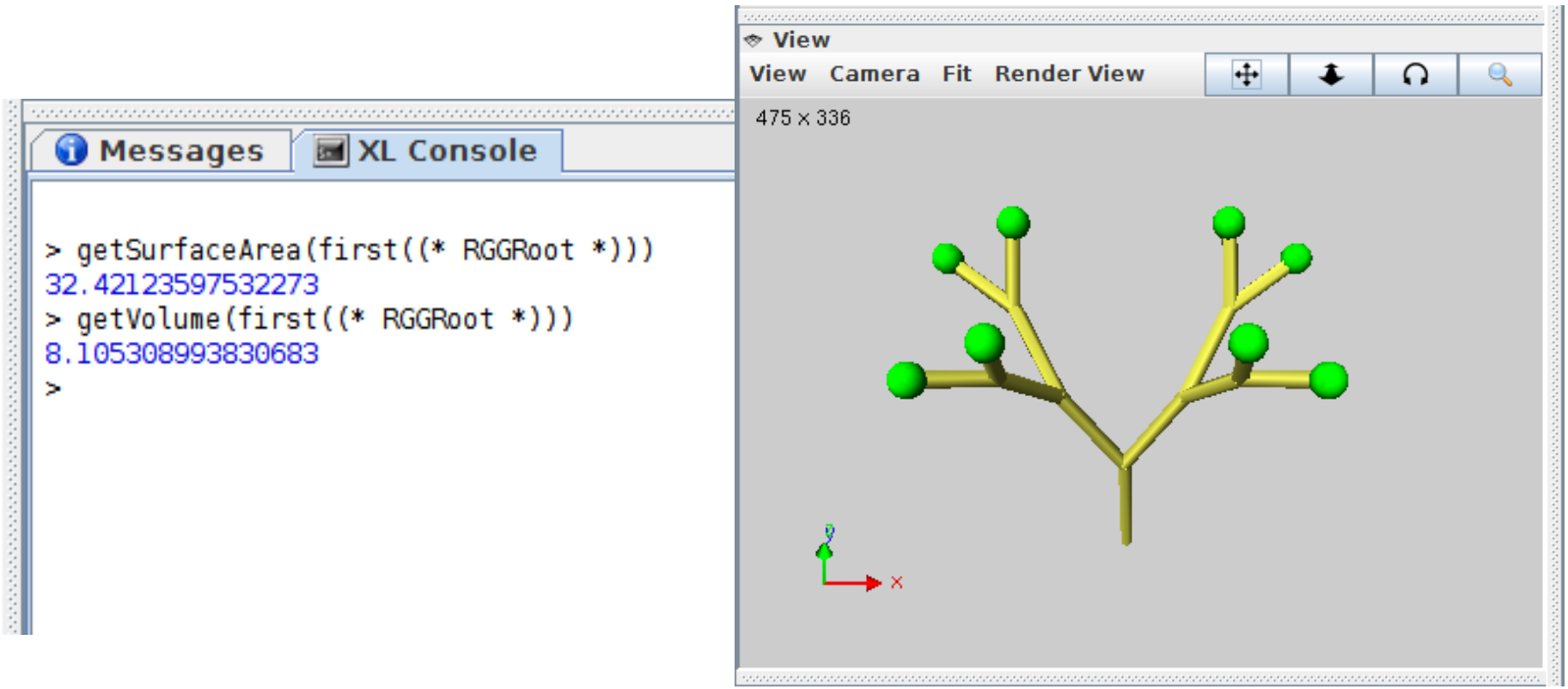
added *getSurfaceArea*, *getVolume* functions to all primitives

- implemented only for basic primitives ⟶
  - Sphere, Box, Cylinder, Cone, Frustum, F
  - Parallelogram, Patch



added *getSurfaceArea*, *getVolume* functions to the library class to calculate the values for a subgraph

- total volume of all nodes of the subgraph with the node *root* as root node
- intersection with other object are not considered



added functions to control view3d repaint

- turn repaint of View3D panel on and off
- dummy nodes become redundant for updating the 3-d view

```
int i = 0;

protected void init () {
    i = 0;
    disableView3DRepaint();

    [  Axiom ==> F;  ]
}

public void run () [
  F ==> F RU(120) F RU(-120) F;
  {
    if(i%2==0) repaintView3D();
    i++;
    if(i==7) enableView3DRepaint();
  }
]
```

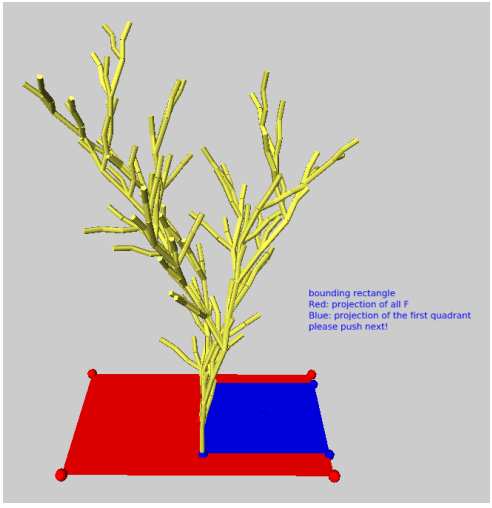
added Library functions to calculate the XY-projection

- Show Examples $\rightarrow$ projection
- $type \in \{BOUNDING\_RECTANGLE,\ CONVEXHULL,\ ALPHA\_SHAPE\}$
- alpha shape: $alpha \in [0,\ 1]$

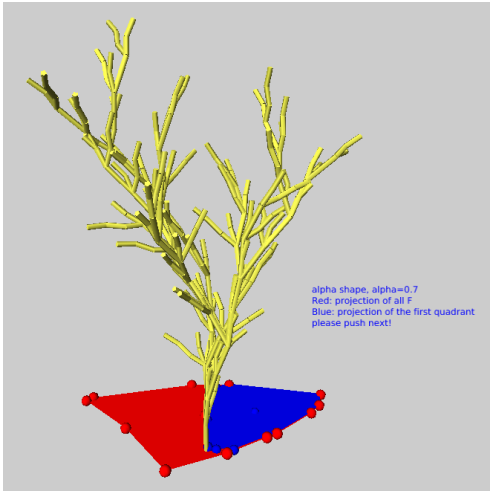
```
1  //this will use the location of all nodes that can be reached
       from Start
   Point3d[] points=getXYProjection(first((*Start*)), type, alpha);
3
   print("#points = "); println(points.length);
5  print("area = "); println(getAreaOfNonTriangulation(points));

7  //mark points
   markPoints(points, 0.035, RED);
9
   //draw surface
11 makeGraph ==> first((*Start*)) triangulate(points).(setShader(
       RED));
```

added Library functions to calculate the XY-projection

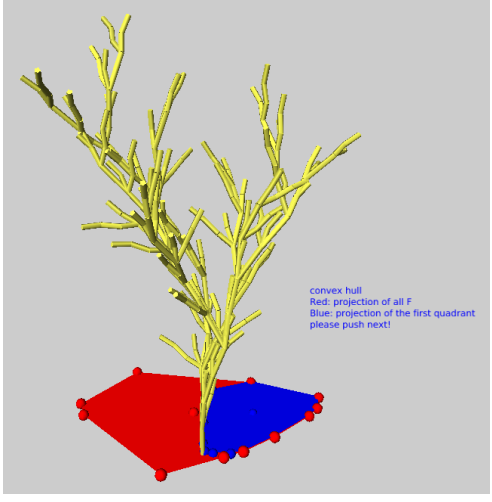
bounding rectangle:



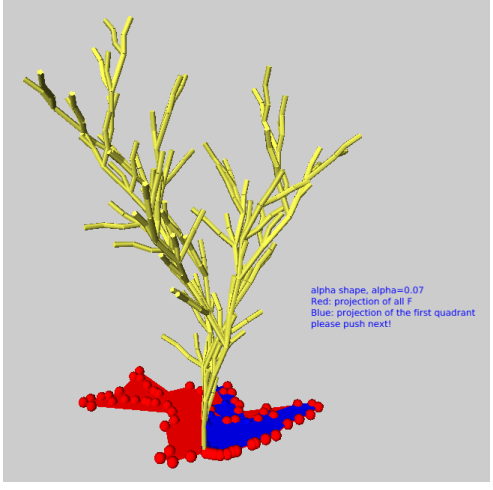
alpha = 0.7



convex hull:

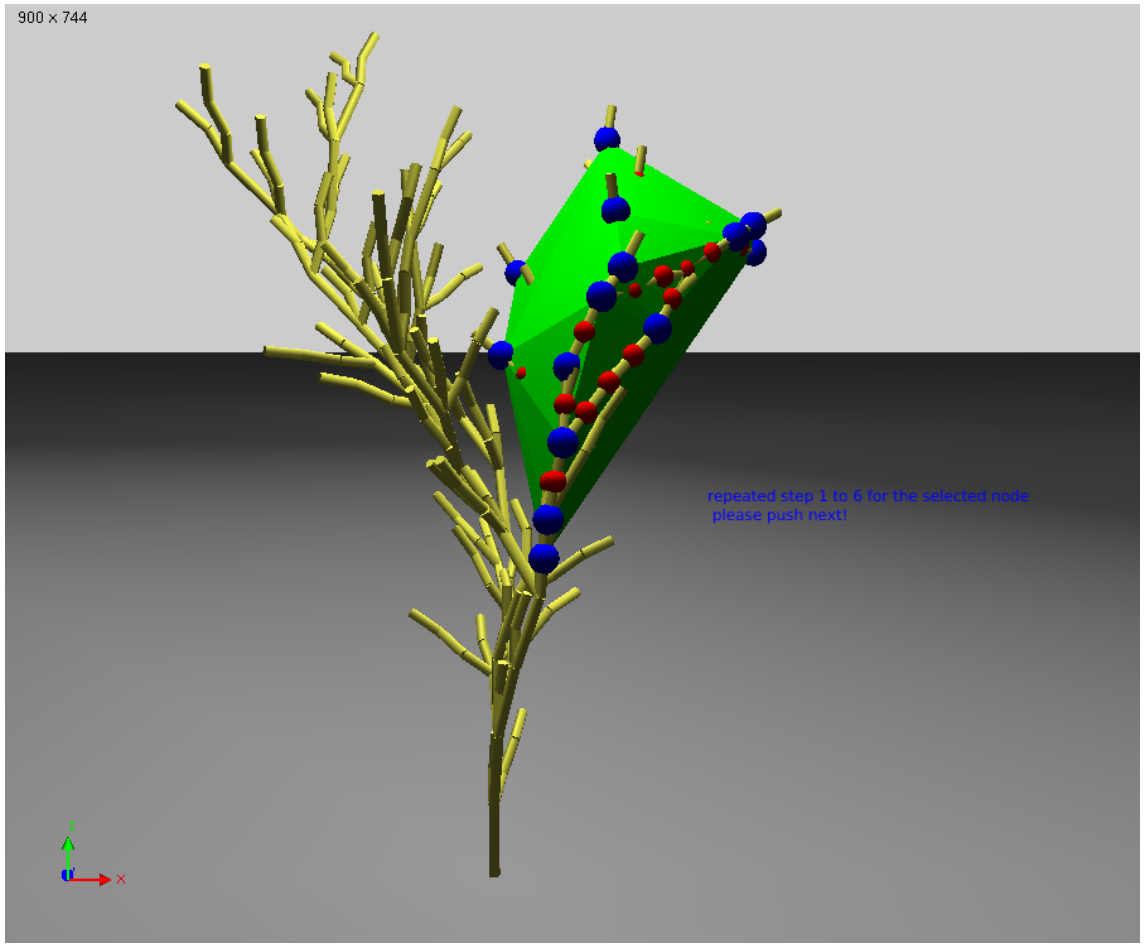


alpha = 0.07



added Library functions to calculate and visualize the convex hull

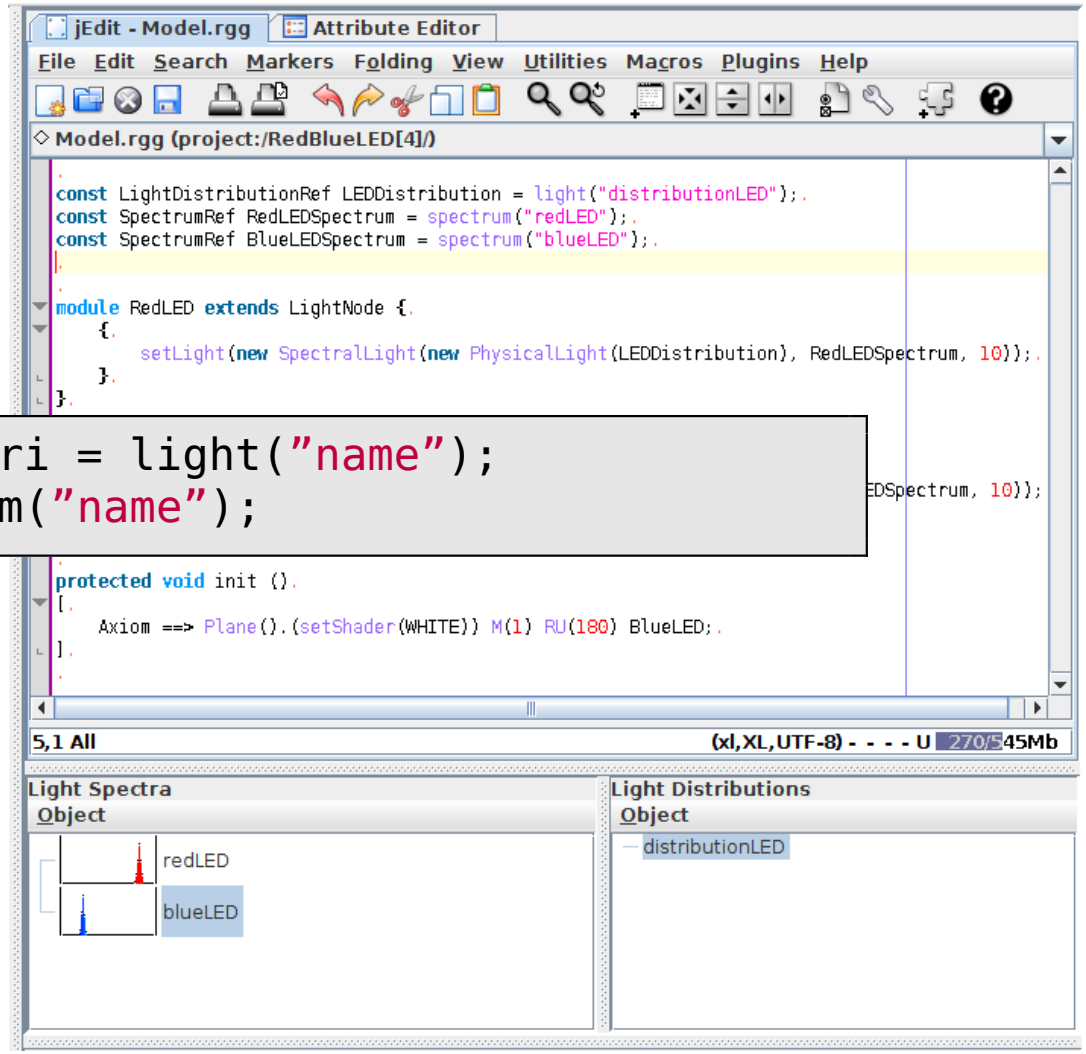
- Show Examples → convexHull



added references for spectra and light distributions

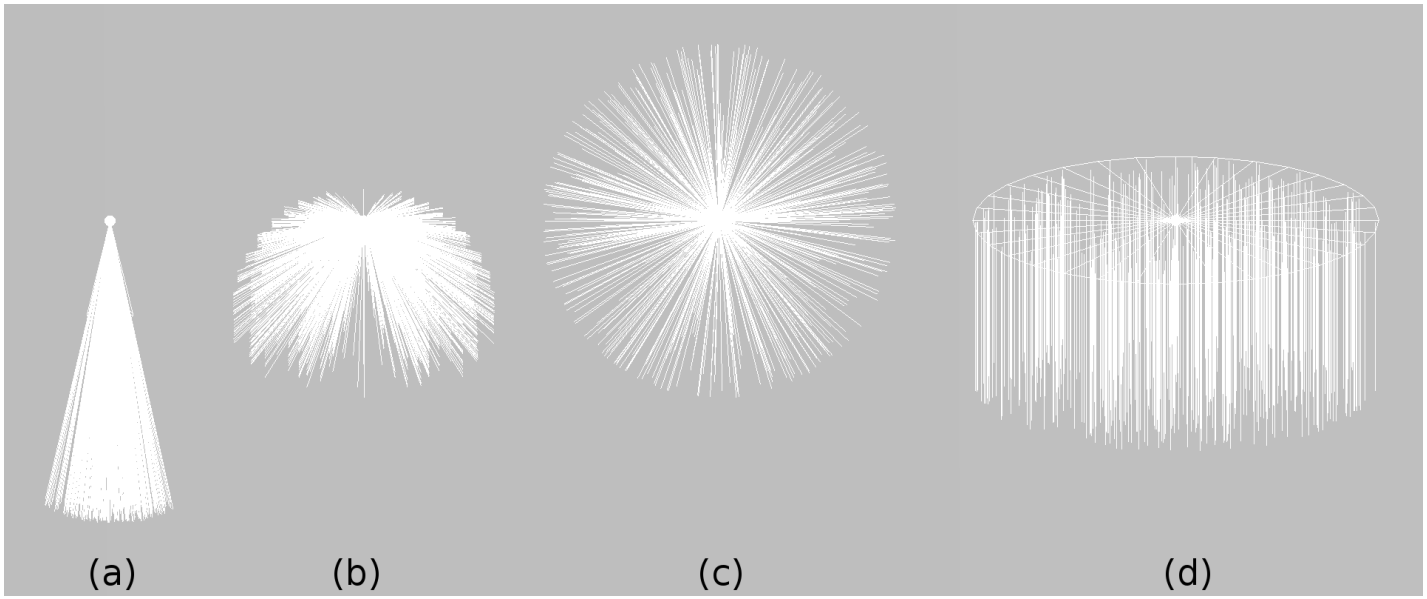
- Show Examples → LampDemo
- two new panels:
  - Light Spectra
  - Light Distributions
- same mechanism as *ShaderRef* and *shader*

```
1 LightDistributionRef distri = light("name");
  SpectrumRef spd = spectrum("name");
```



added functions to visualize the physical light distribution

- variable number of rays and length



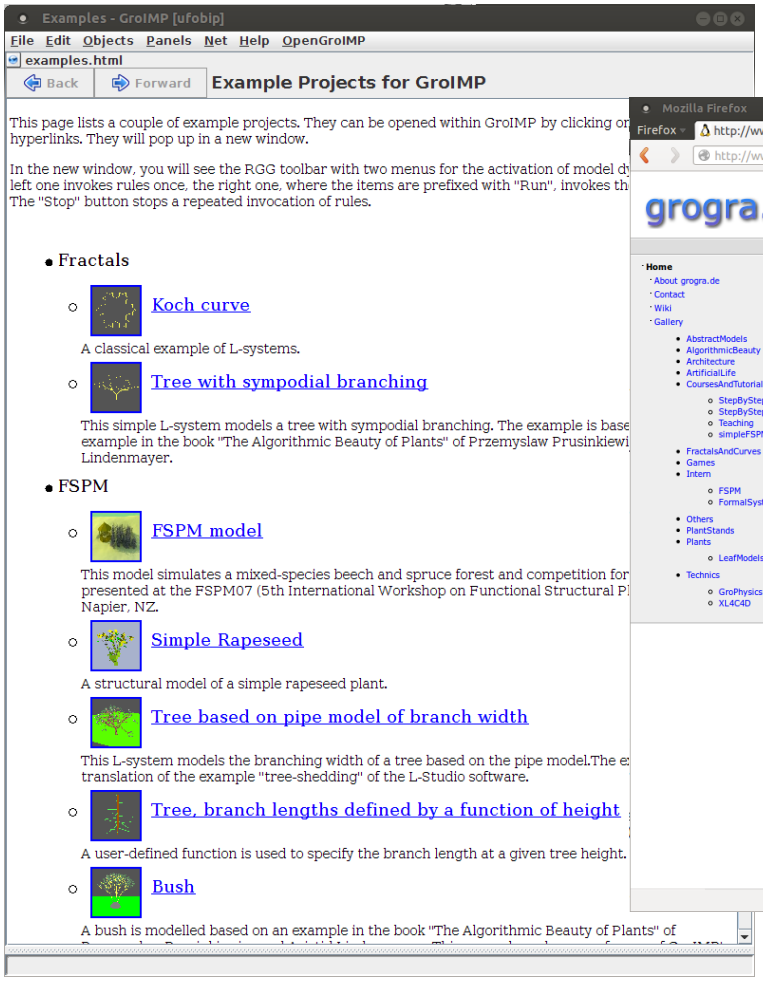
spot spectral point directed

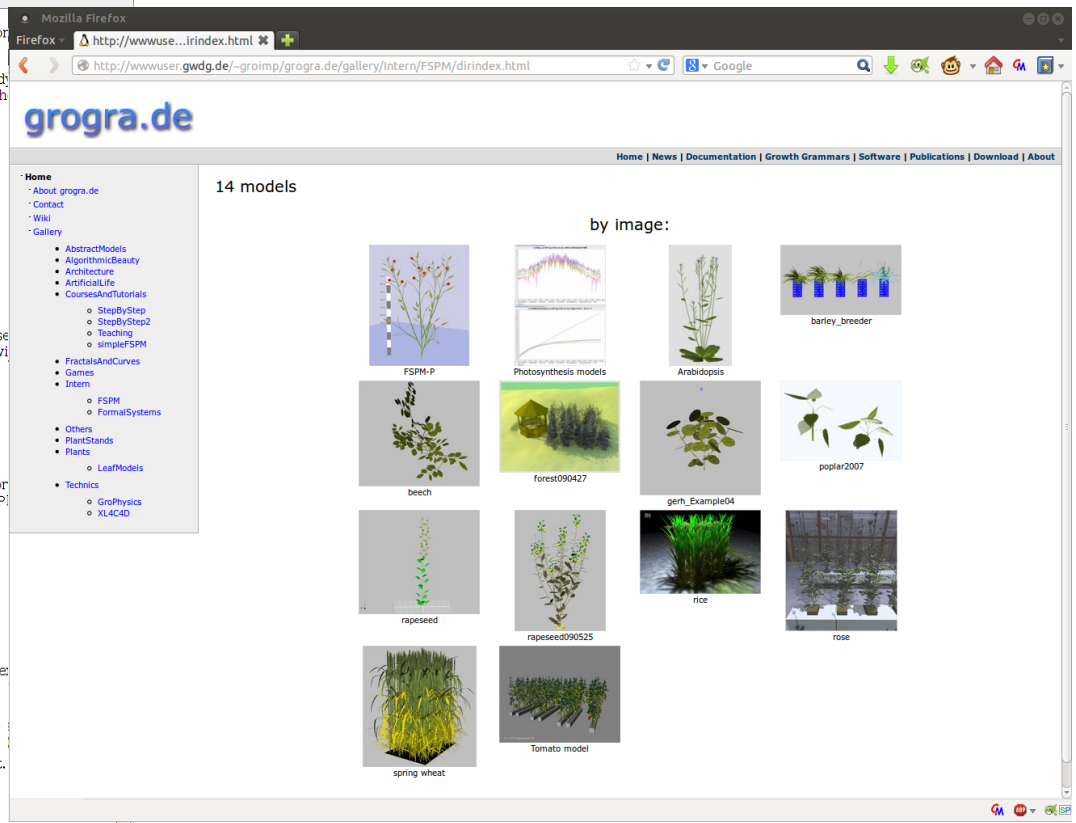
```
LightNode().(
    setLight(new DirectionalLight().(
        setVisualize(true),
        setRaylength(1.75)
    ))
);
```

extended ”Show Examples” gallery

- added new examples

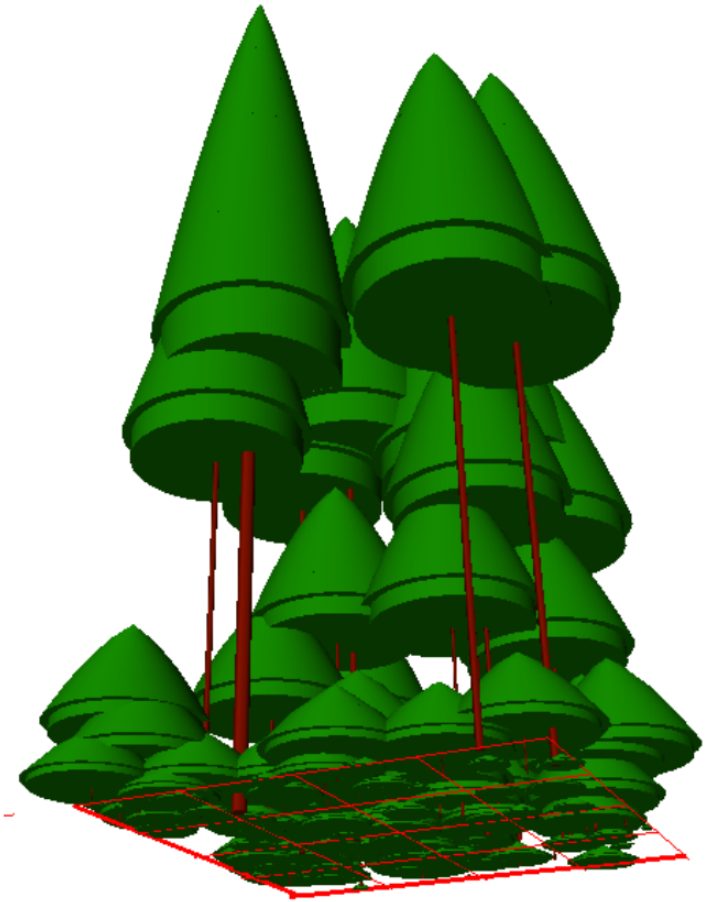
nearly 400 examples included ...





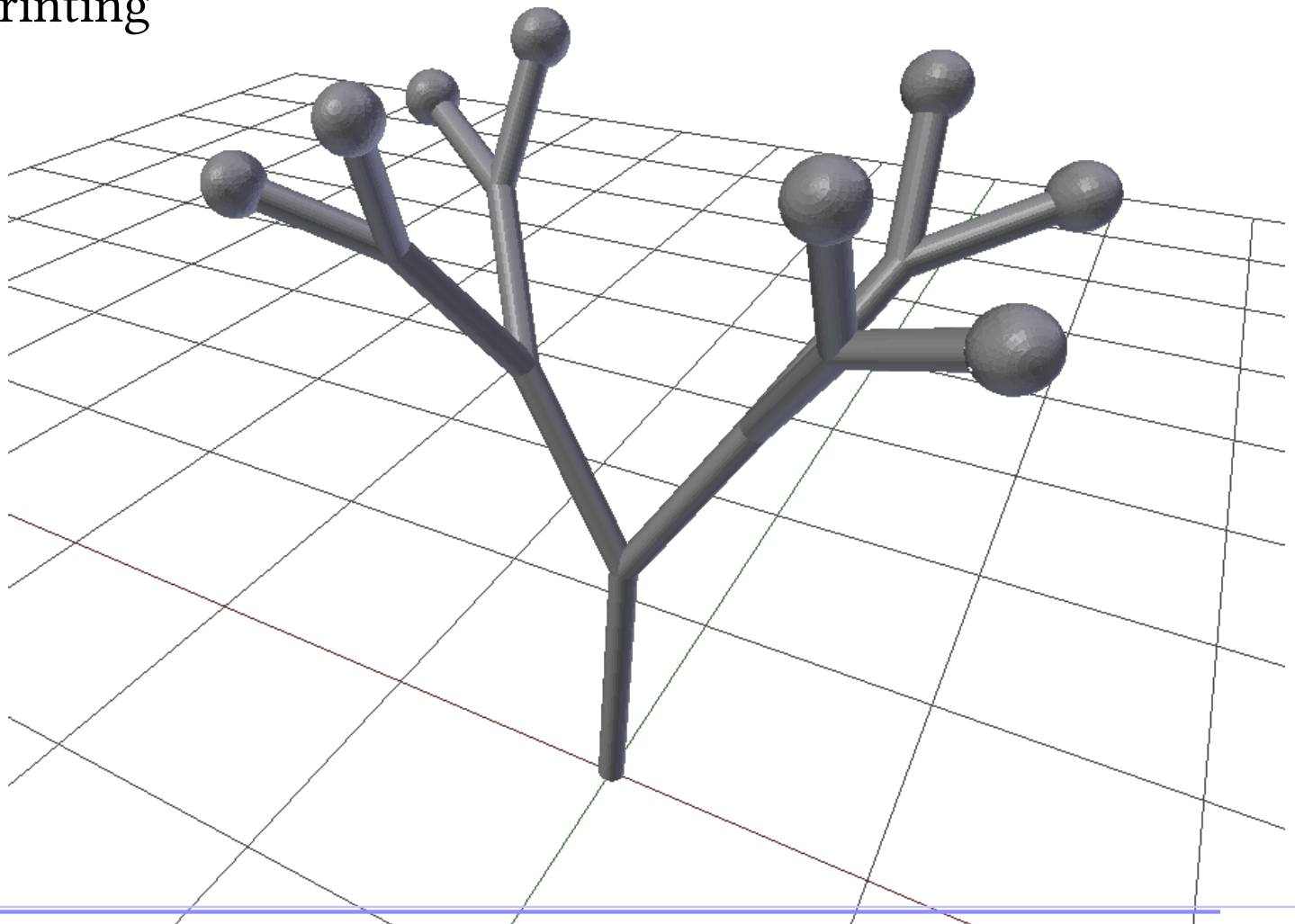
added concepts for multi-scaled modelling and level-of-detail visualization

- multi-scale typed graph



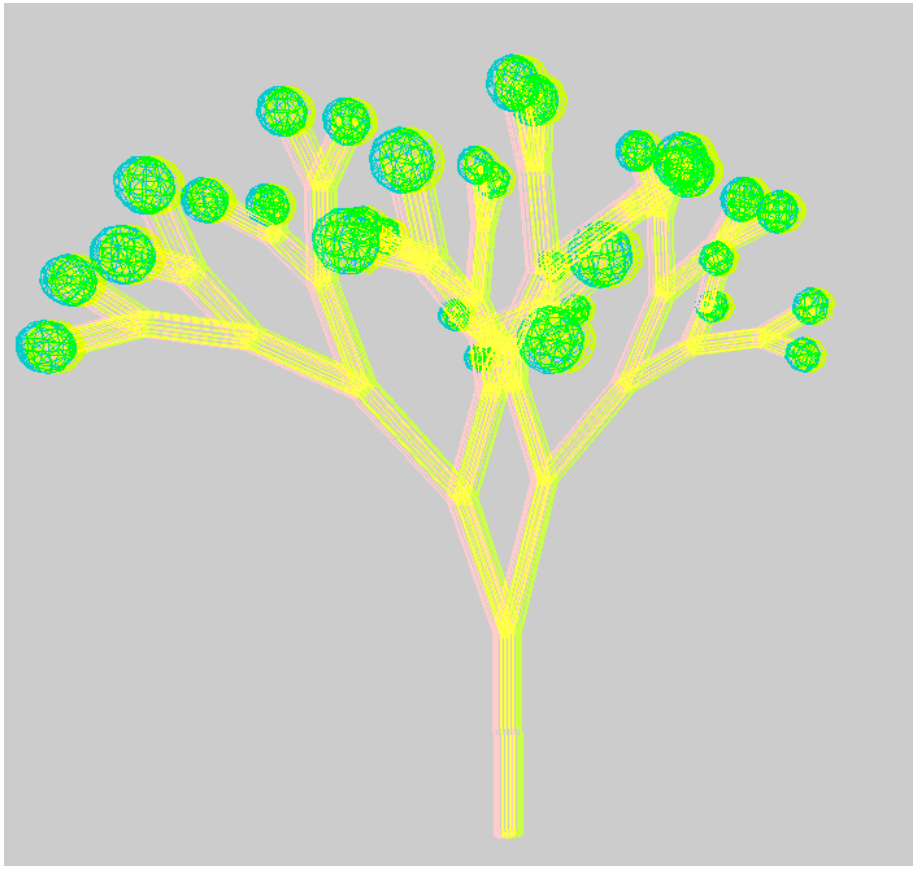
added STL export

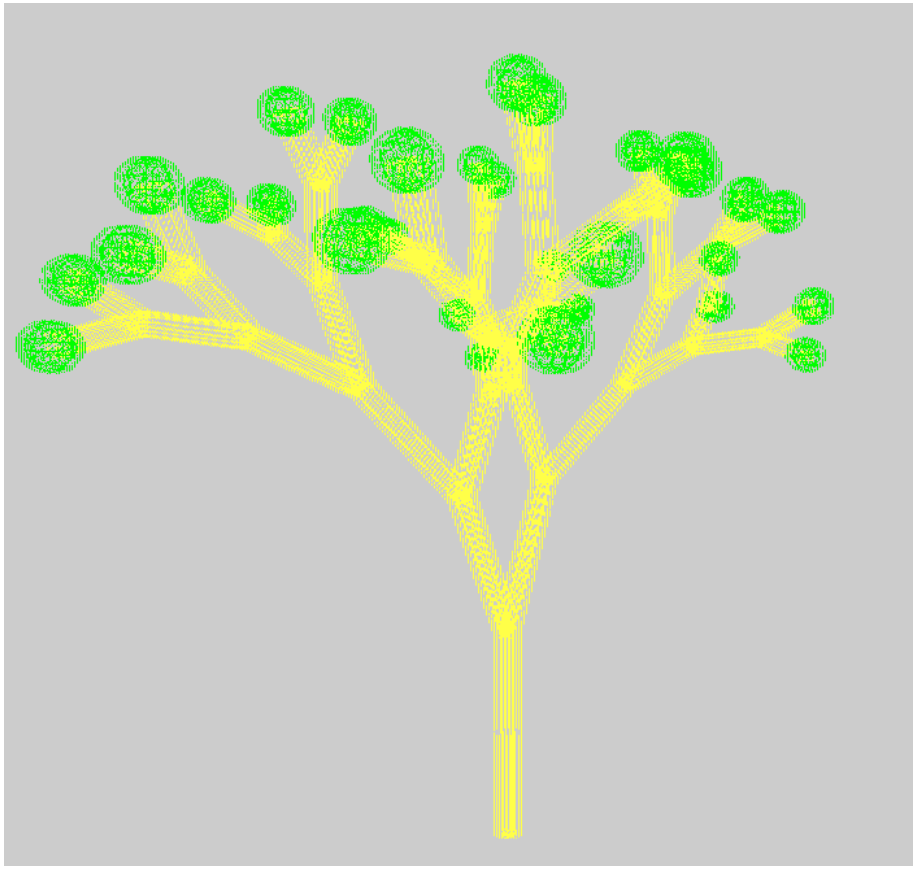
- STL (STereoLithography)
- native to common CAD software
- 3d printing



added stereo image (anaglyph) view for Wireframe View3D

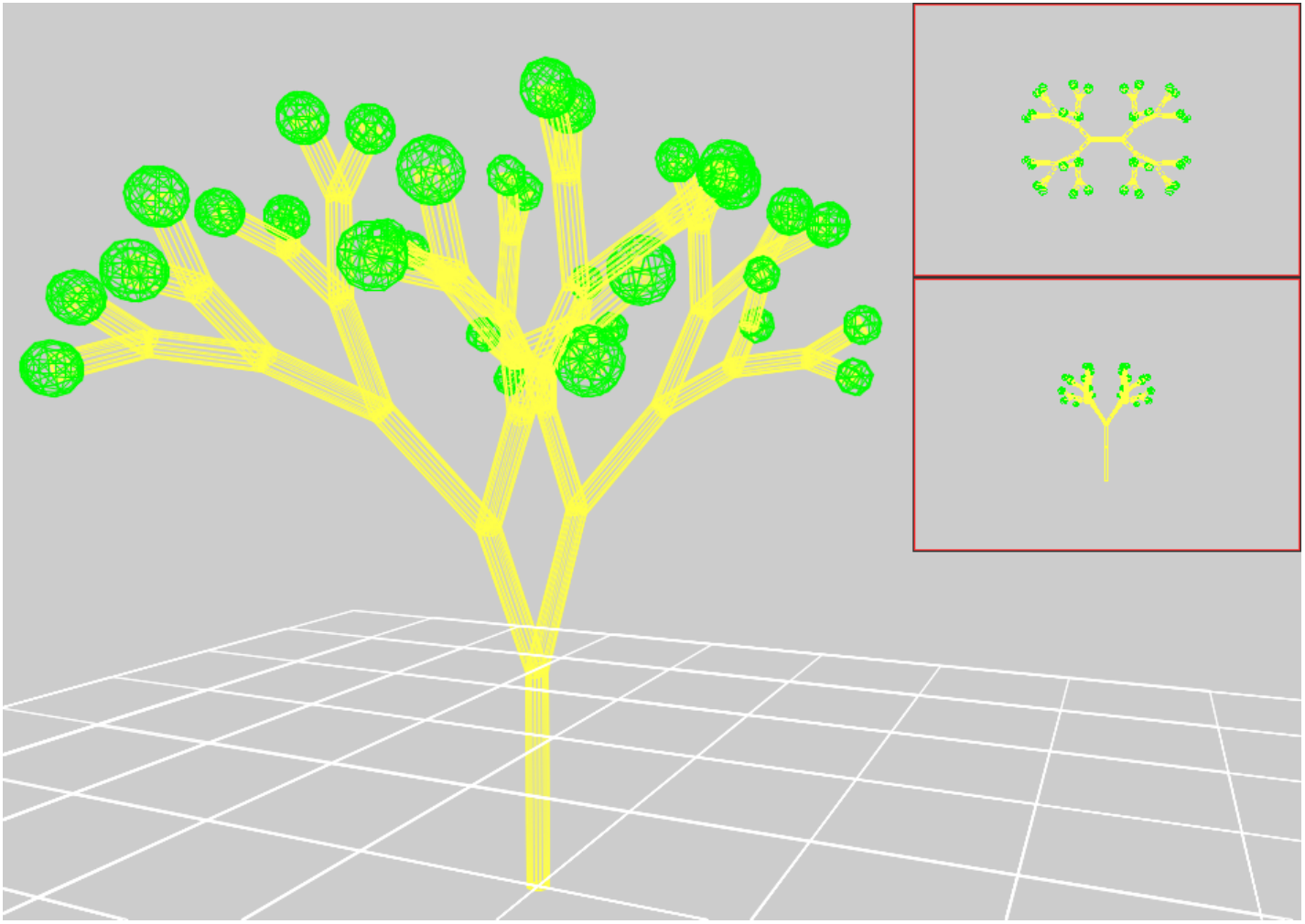
- stereo images and anaglyphs





added split view for Wireframe View3D

- top-, botton-, side-view



added copy protection for models

- protected code and parameter
- keyword: ***scrambled***
- individually for each file


```
//scrambled

import ...;

module A(float len) extends Sphere(0.1).(setShader(GREEN));

protected void init () [
    Axiom ==> A(1);
]
```

general bug fixes / extensions

fixes

- shader mapping
- compatibility to Java 8
- spectral rendering without dispersion (Flux)
- deactivated debug file
- *setPower* problems when used with CPU raytracer
- fixed memory-problem when closing a project
- ...

extensions

- implemented *setSeed* for FluxLightModel and FluxRenderer
- enabled AreaLight for FluxLightModel
- added errror messages
- ...

Thank you for your attention!

www.grogra.de