

Using pyFormex as preprocessor



UNIVERSITEIT
GENT

Benedict Verhegghe

What is pyFormex

- Free Open Source Software (FOSS):
<http://www.pyformex.org>
- Under development at bioMMeda
- Generation and transformation of complex 3D geometries
- Actions executed from a script, with GUI elements as support
- Aiming at minimal user interaction

Why pyFormex?

- Traditional CAD or FE preprocessor:
 - Expensive
 - Not available to everyone
 - Closed
 - Comprehensive
 - GUI based, scripting as an afterthought
 - Automation and parameterization are often cumbersome
 - Suboptimal FE meshes

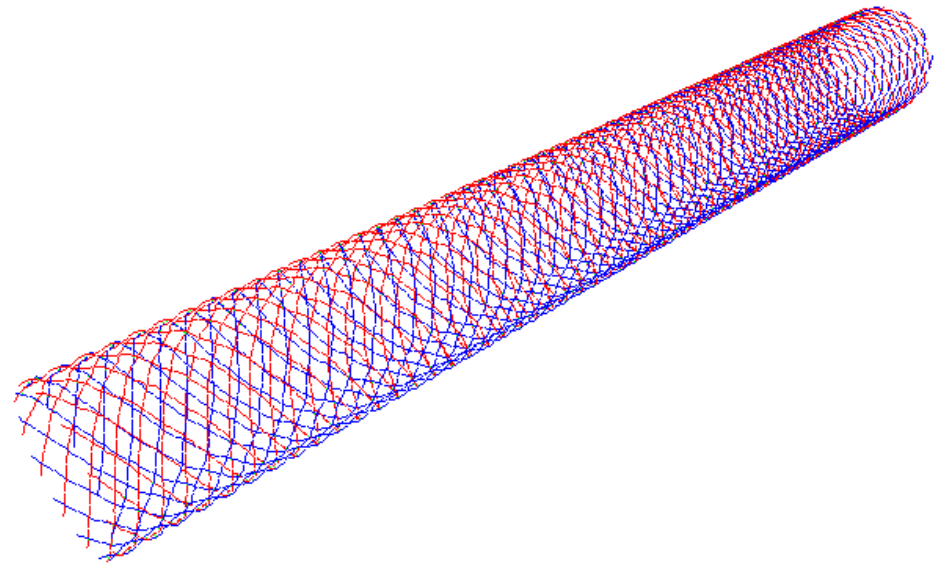
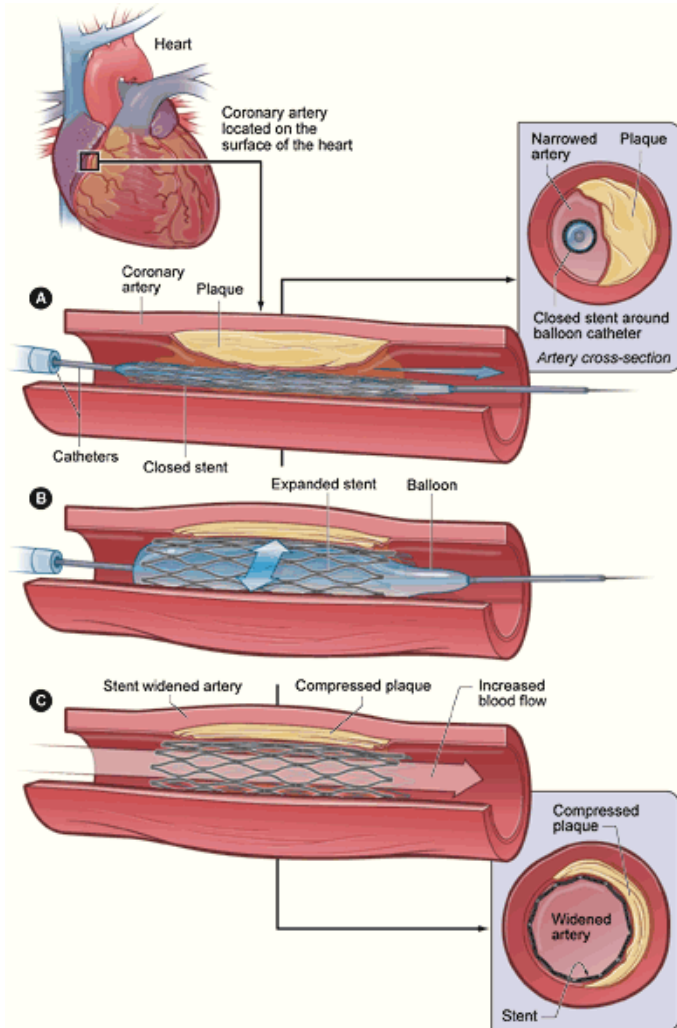
Why pyFormex?

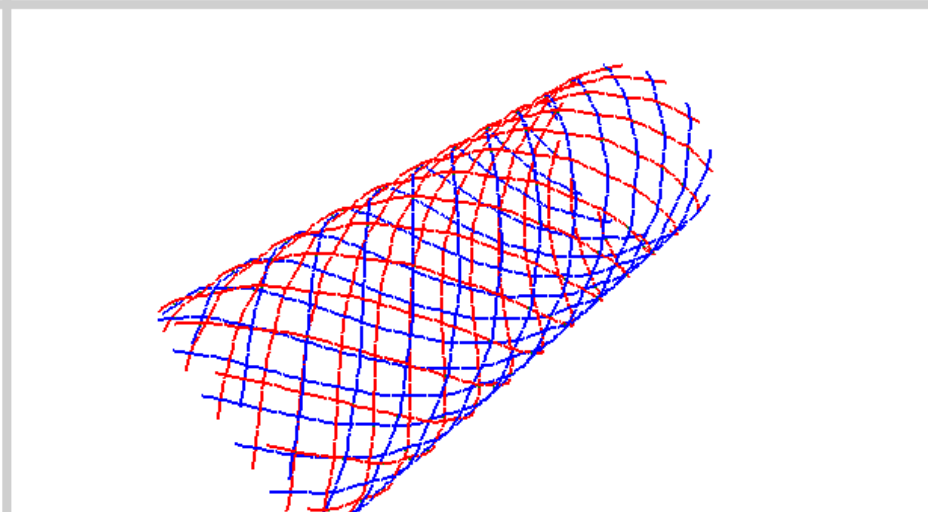
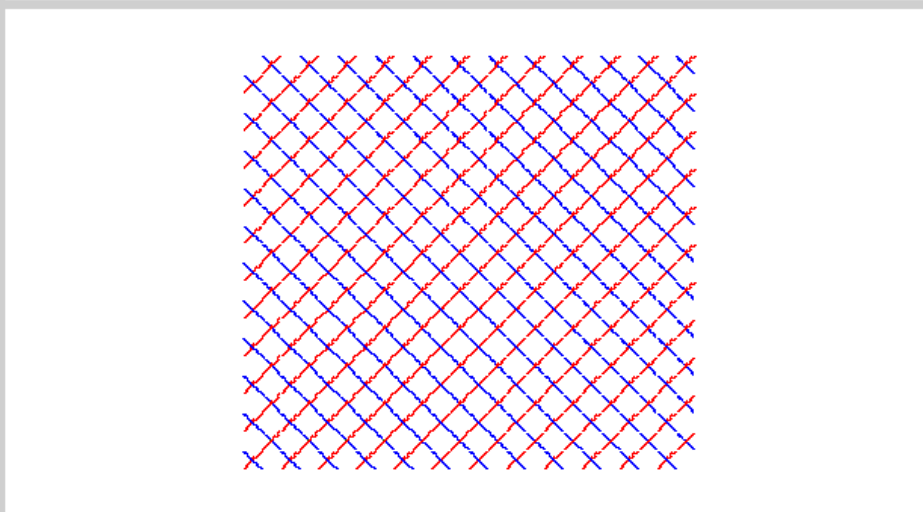
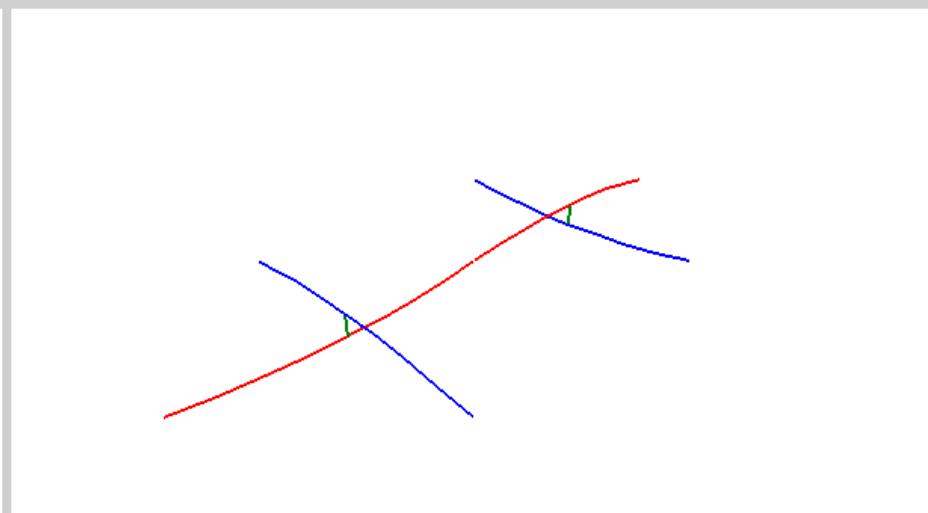
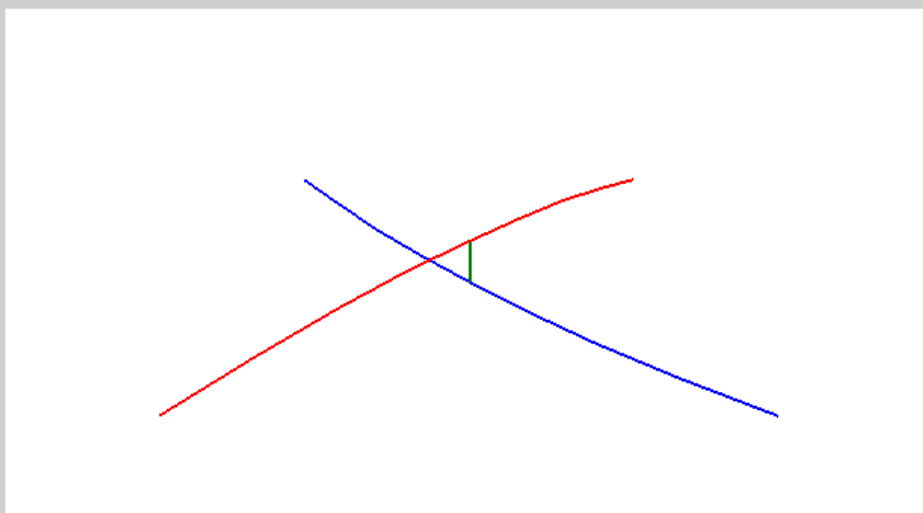
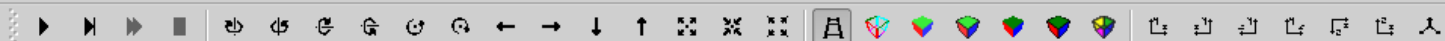
- pyFormex:
 - Free (no cost)
 - Freely distributable
 - Open: fully customizable, expandable
 - Small: only what is needed
 - Script based, supported by GUI elements
 - Automation, parametrization are natural
 - FE meshing can be completely guided

pyFormex project

- Home Page: <http://pyformex.org>
- Open Source:
<http://developer.berlios.de/projects/pyformex/>
- GNU General Public License v3 or higher:
 - Freedom to use, study, modify, distribute
 - Source is available to anyone
 - Changes, if distributed, available as source
 - All components need compatible licenses
- Linux (on Windows: BuMPix Linux Live USB)

pyFormex Examples





Step 13: Translate the full pattern over the stent radius in Z-direction

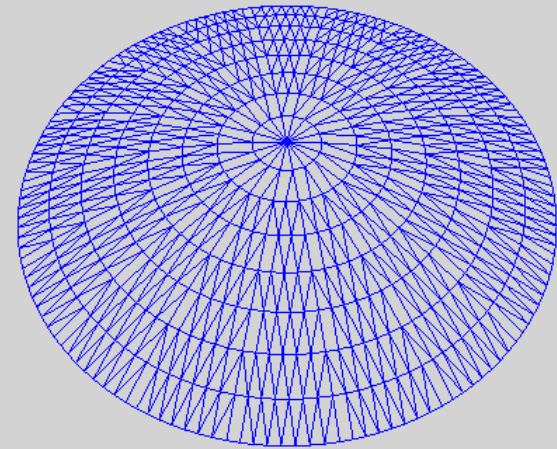
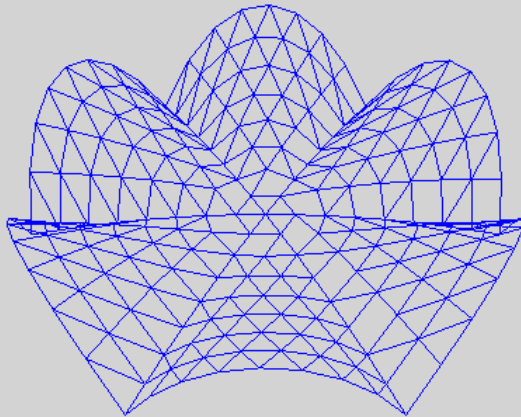
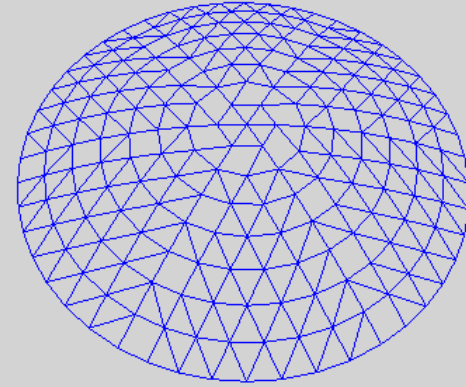
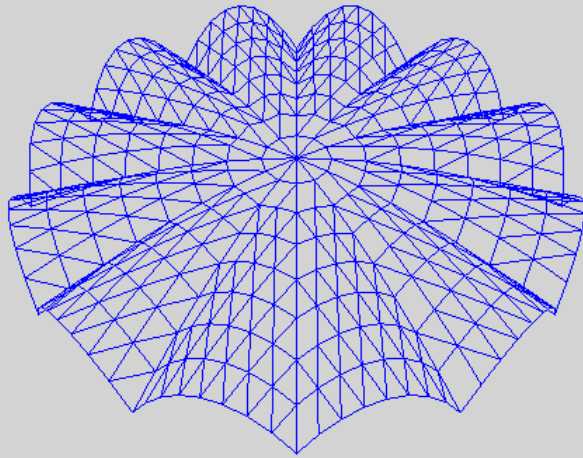
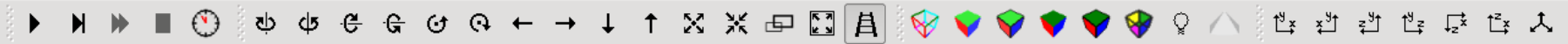
```
C=F.translate([0.,0.,r])
```

Step 14: Roll the nearly planar grid into a cylinder

```
self.F = C.cylindrical(dir=[2,0,1],scale=[1.,360./(nx*dx),p/nx/dy])
```

Script finished

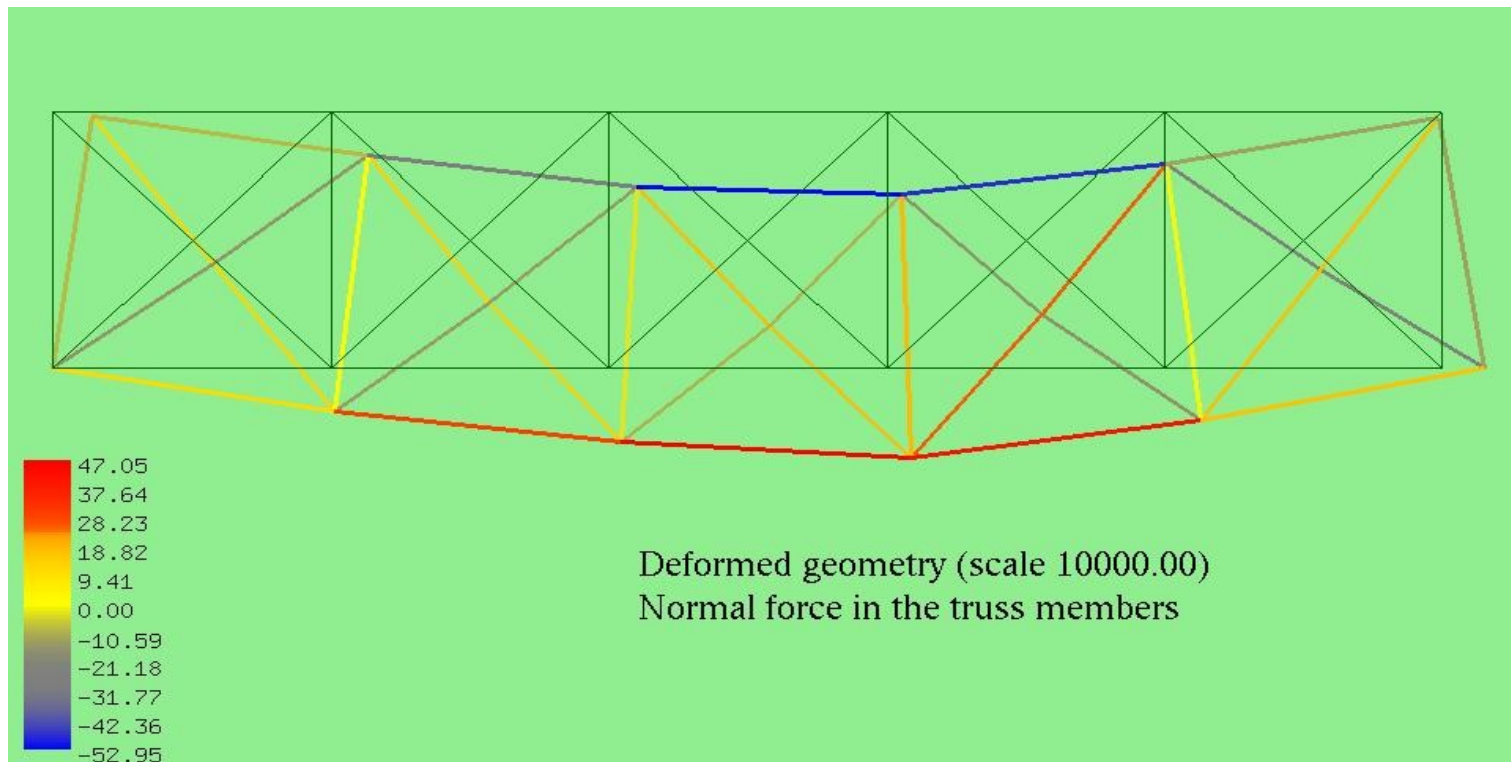
Running command: import -window "pyFormex 0.5.1a1 (2)" png:/home/matthieu/Figures/WS_DEMO.png



```
Dict({'qu': -1, 'do': False, 'wi': True, 'mu': False, 'nk': False, 'ad': False, 'rc': False, 'fm': 'From  
Extension', 'fn': '/home/bene/prj/pyformex/screenshots/scallops.png'})  
Setting workdir = /home/bene/prj/pyformex/screenshots  
[]  
SAVE: quality=-1  
Running command: import -window "pyFormex 0.8.4-a1" png:/home/bene/prj/pyformex/screenshots/scallops.png
```

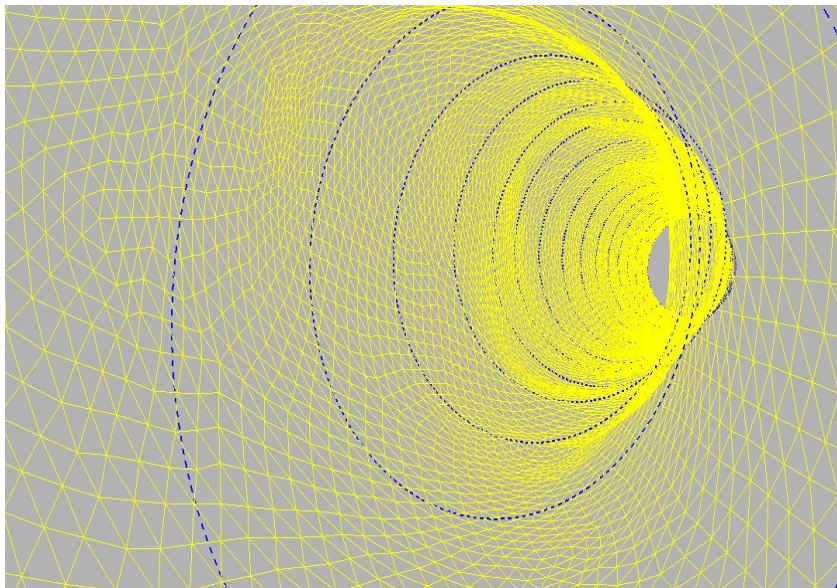

pyFormex examples...

- Structural Analysis

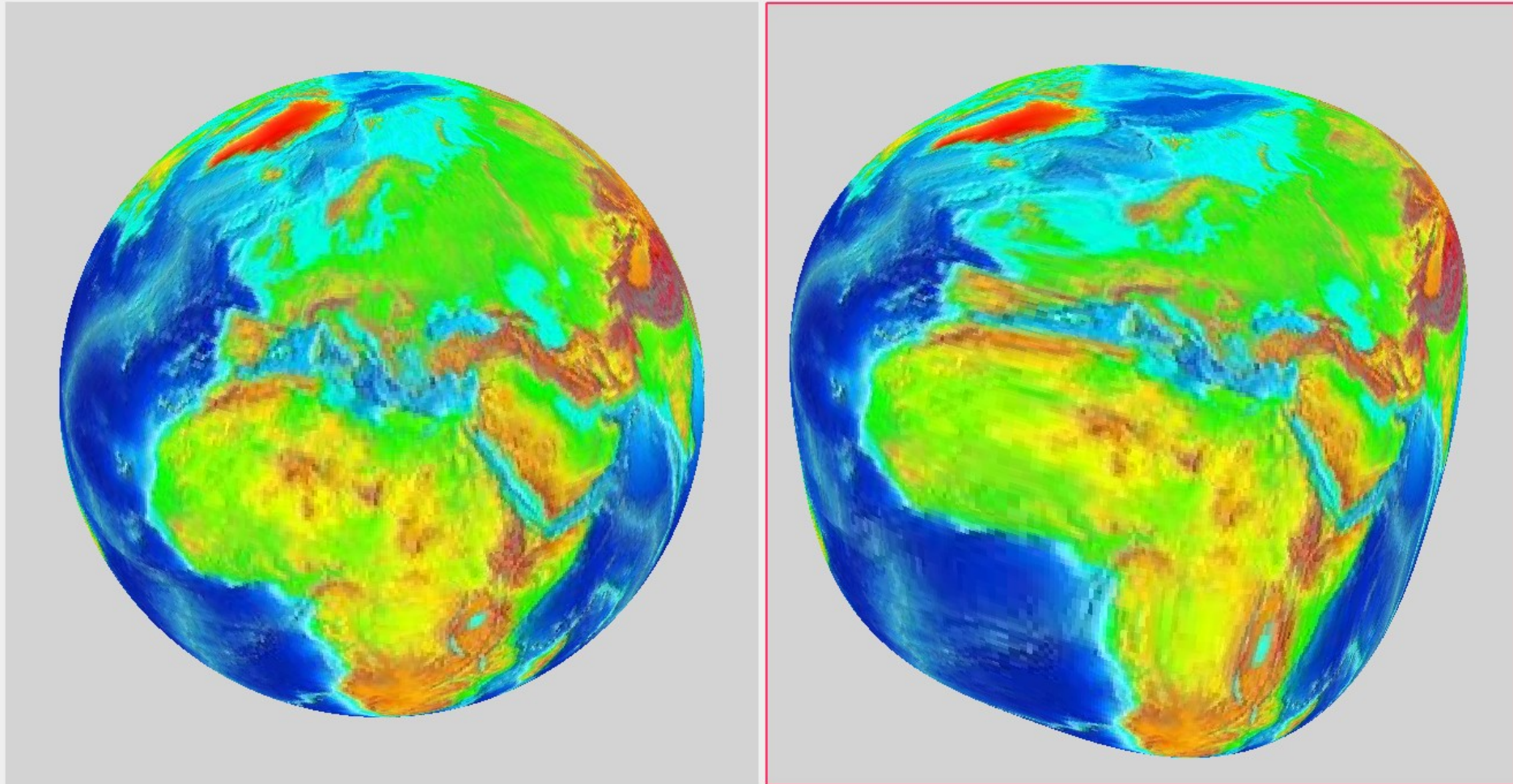


pyFormex examples...

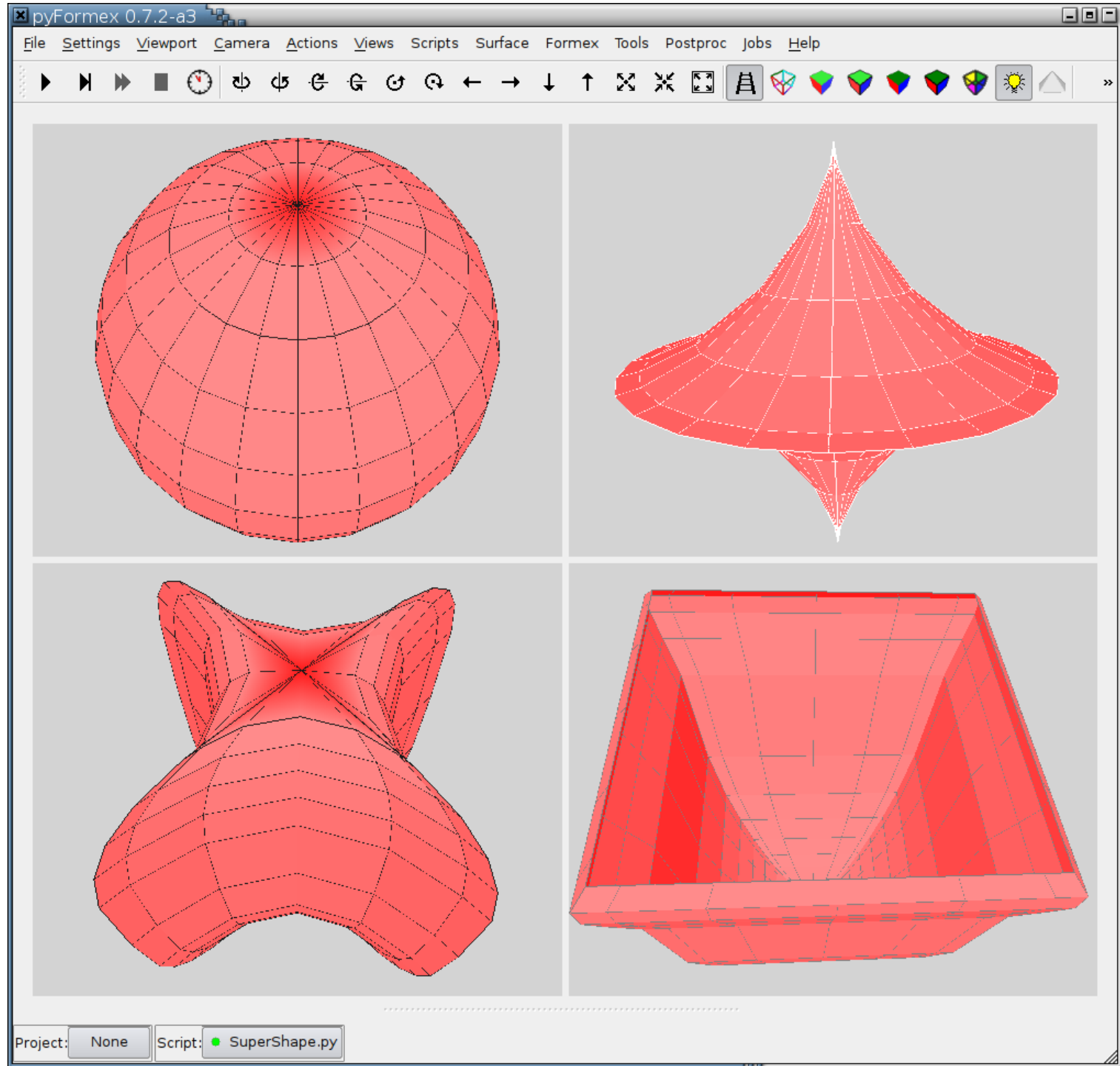
- Operations on surface models



File Settings Viewport Camera Actions Views Scripts Formex Surface Mesh Tools Postproc Jobs Help

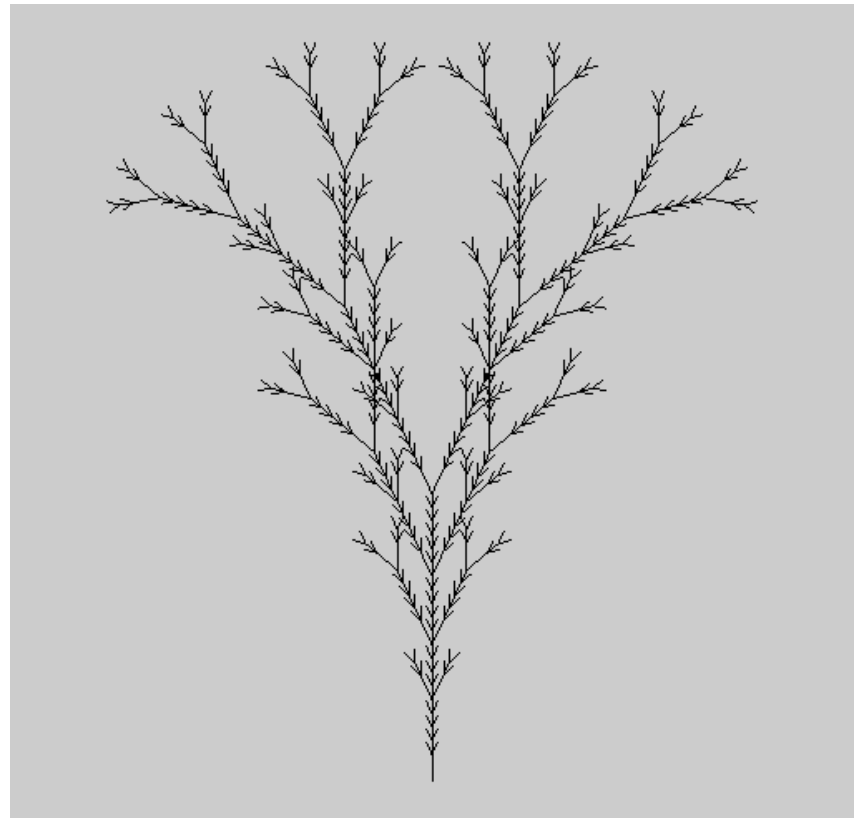
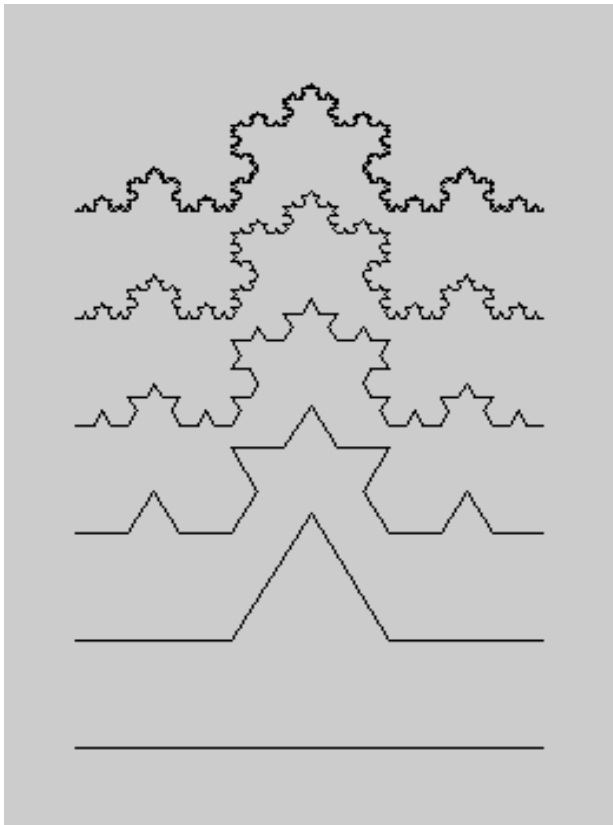


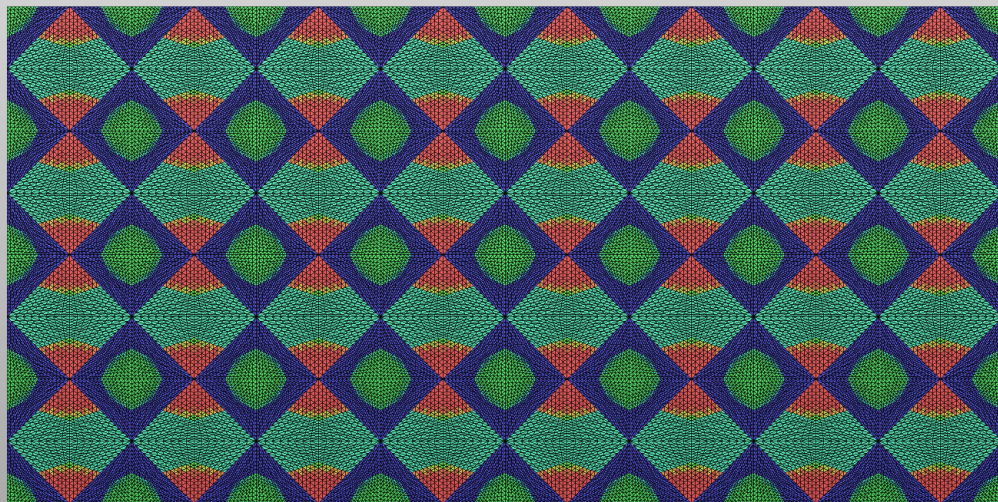
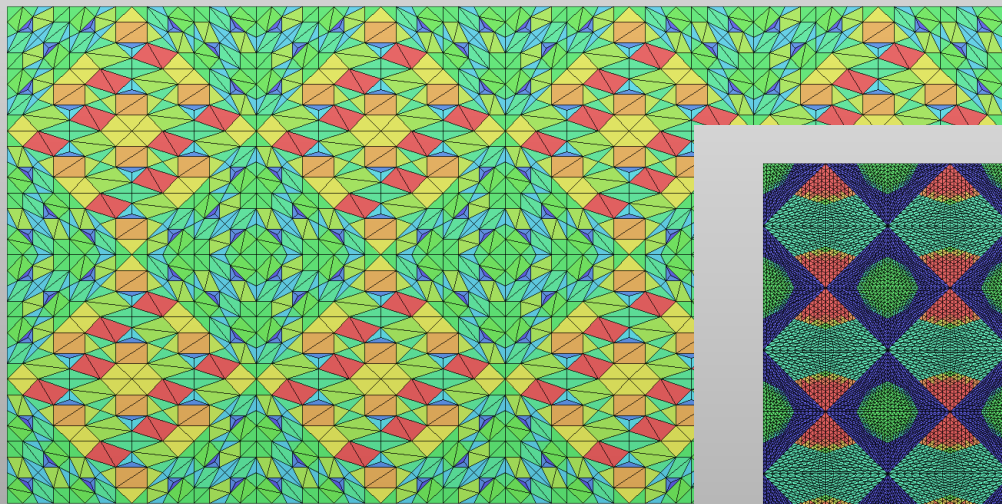
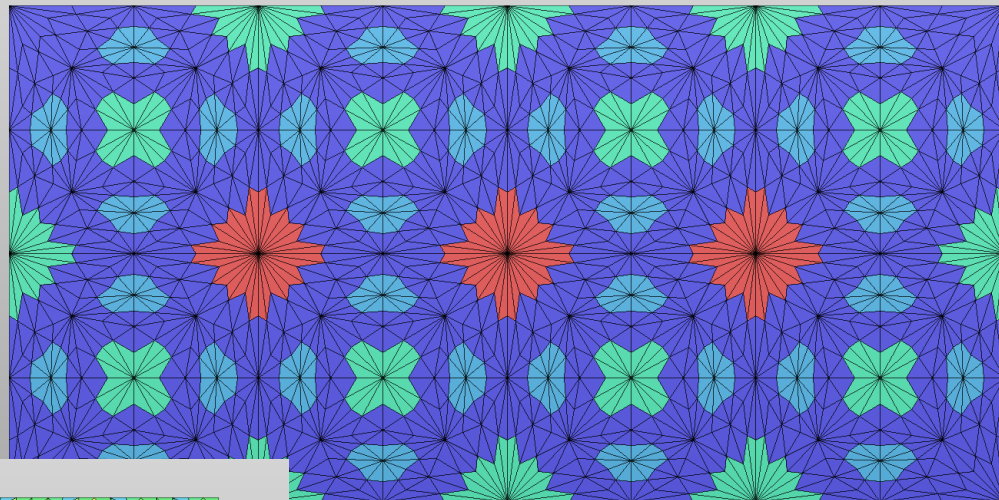
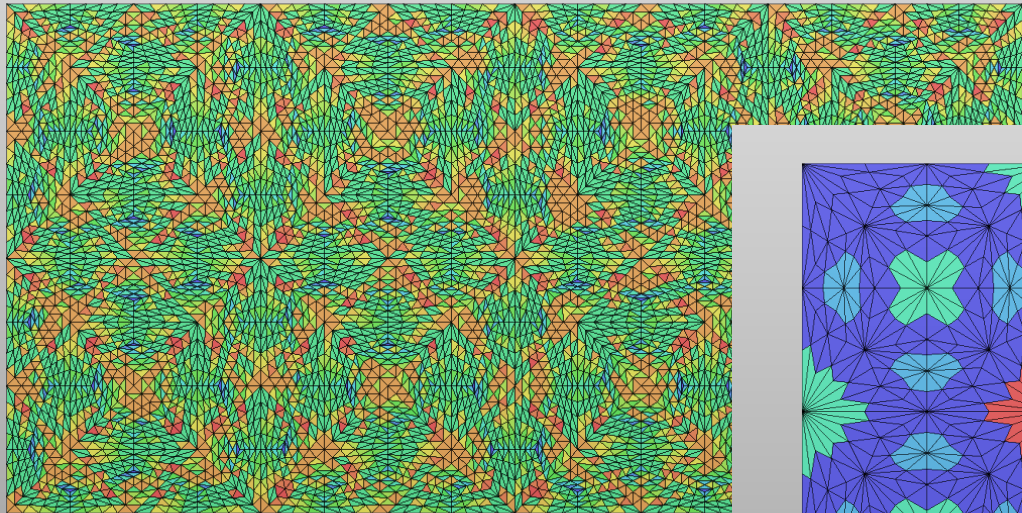
Project: None Script: ● SuperShape.py



pyFormex Examples...

- Illustrations





pyFormex Components

Formex

Mathematical/Geometrical
Transformations

NumPy

Fast Numerical Arrays

Python

Universal Glue

pyFormex Components

Formex
Mathematical/Geometrical
Transformations

NumPy
Fast Numerical Arrays

Python
Universal Glue

Scripting

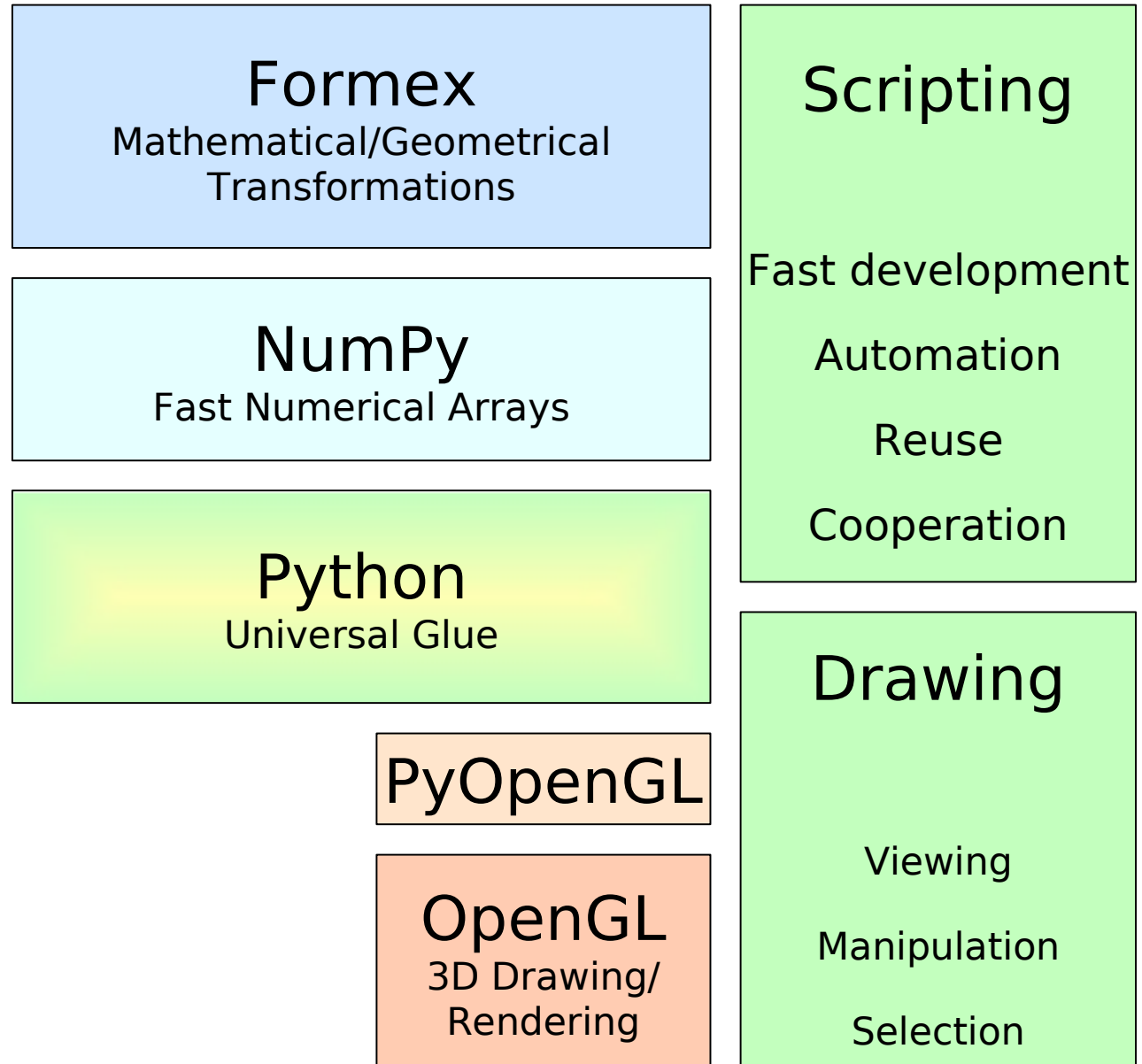
Fast development

Automation

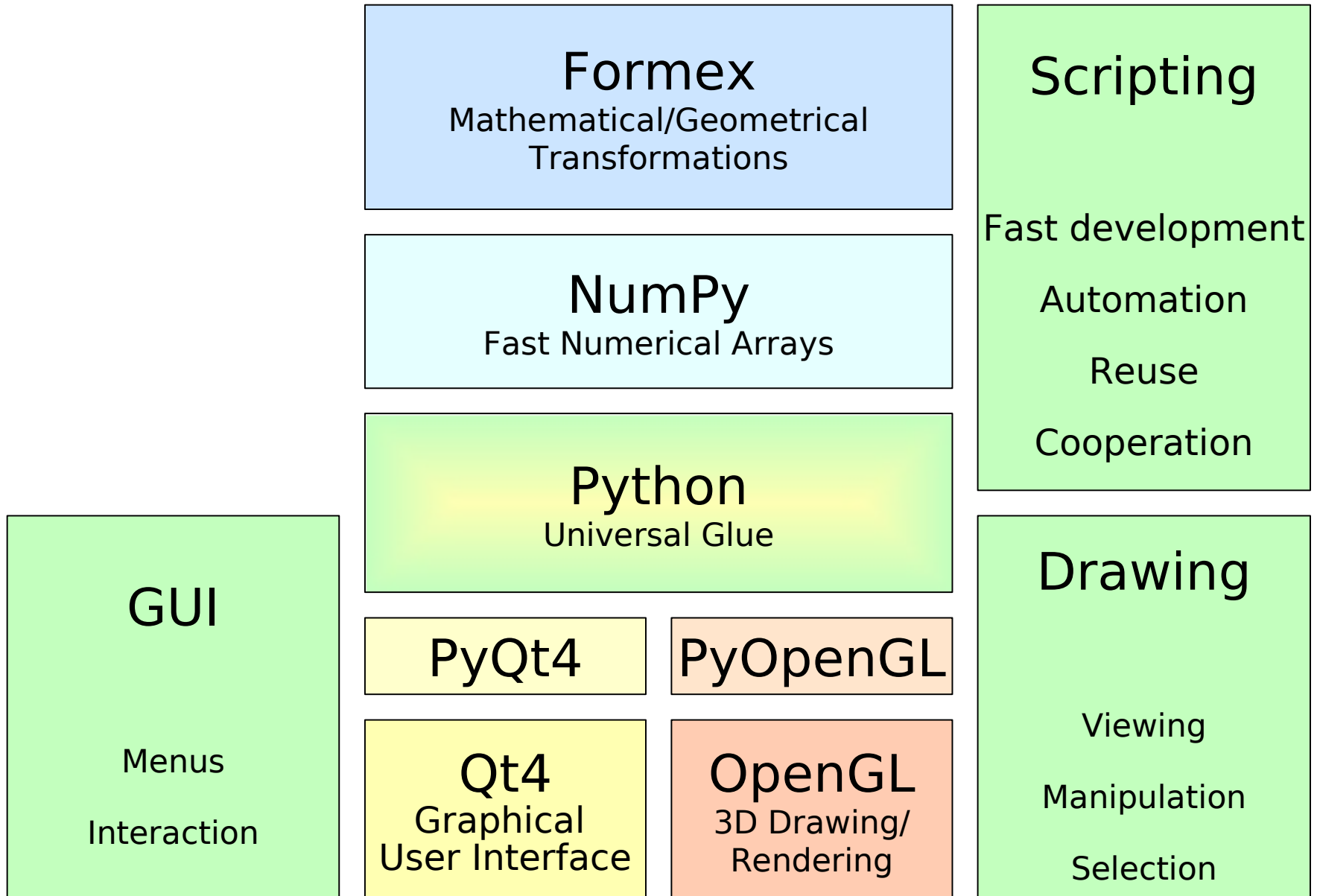
Reuse

Cooperation

pyFormex Components



pyFormex Components



pyFormex Components

Plugins

Formex interactive
Surfaces (STL/GTS)
Lindenmayer
Finite Elements
Interfaces

GUI

Menus
Interaction

Formex

Mathematical/Geometrical
Transformations

NumPy

Fast Numerical Arrays

Python

Universal Glue

PyQt4

Qt4
Graphical
User Interface

PyOpenGL

OpenGL
3D Drawing/
Rendering

Scripting

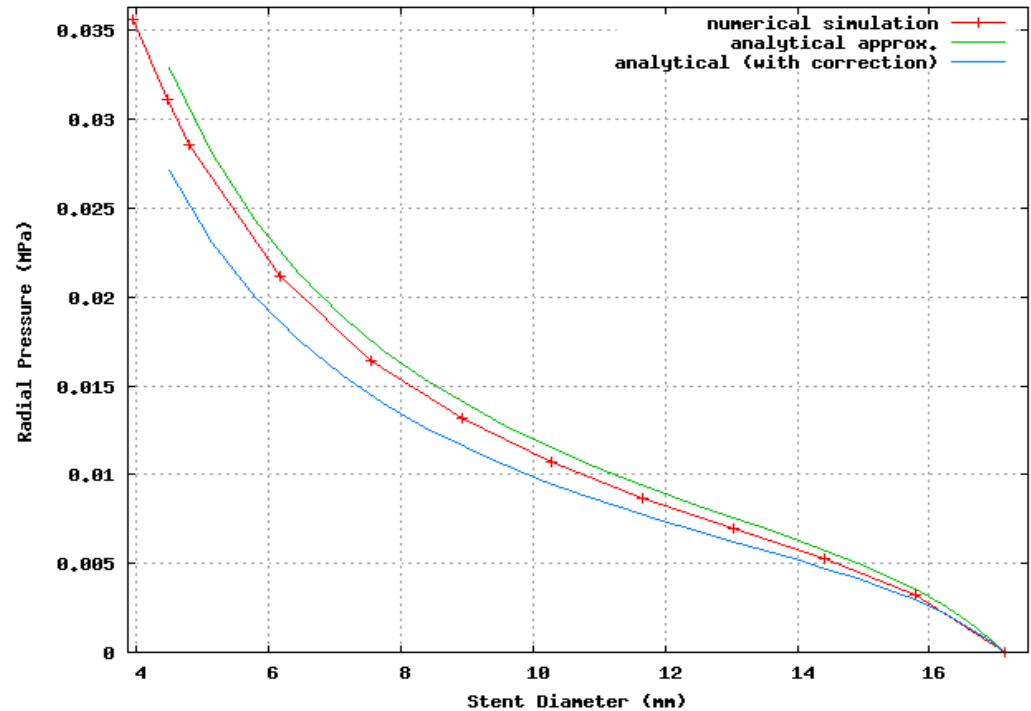
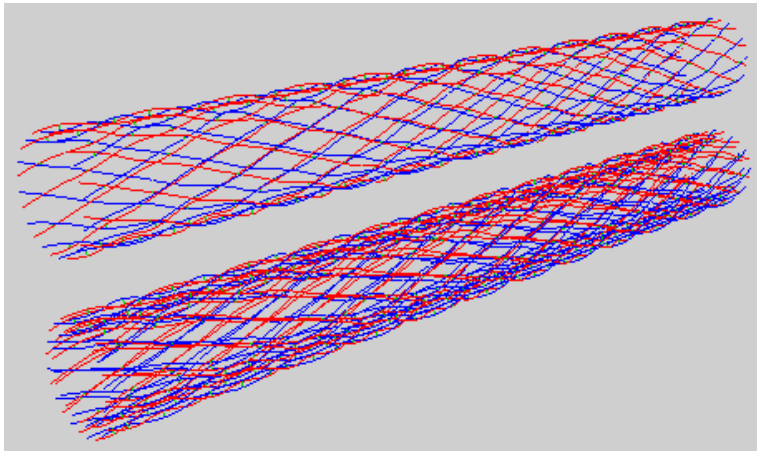
Fast development
Automation
Reuse
Cooperation

Drawing

Viewing
Manipulation
Selection

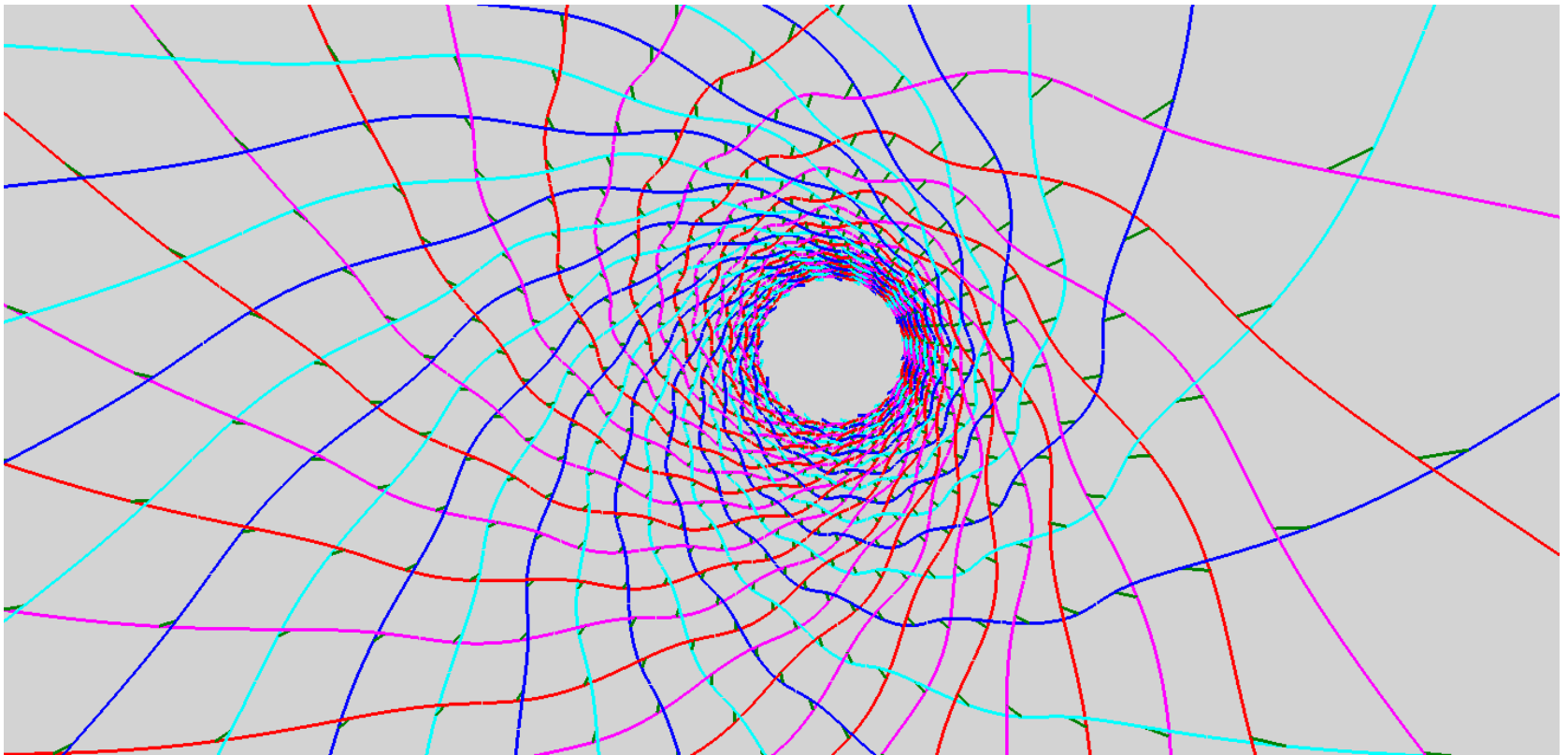
pyFormex in use

- Wirestent preprocessing, simulation, postprocessing, optimization



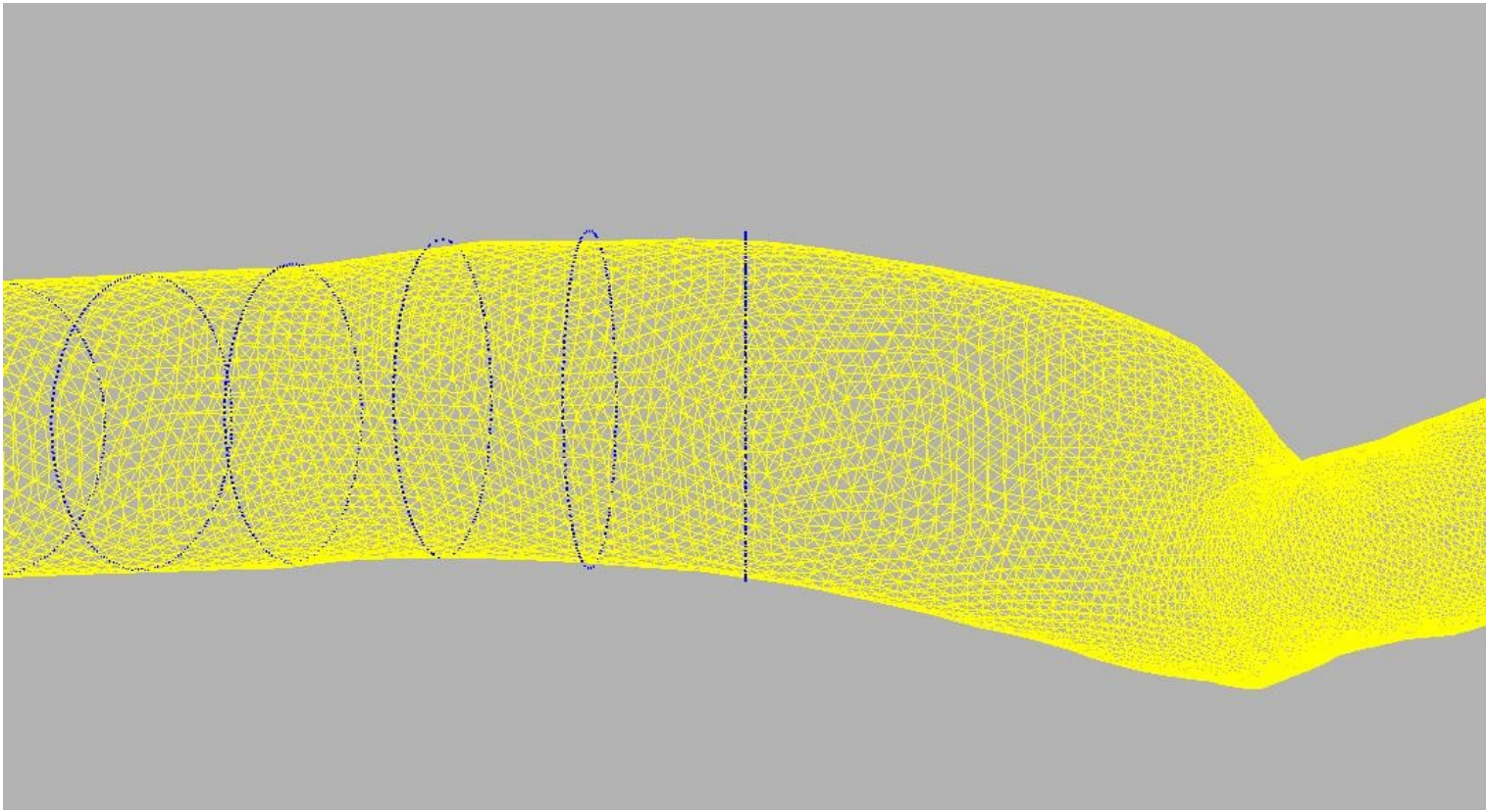
pyFormex in use

(courtesy of FEops)



pyFormex in use

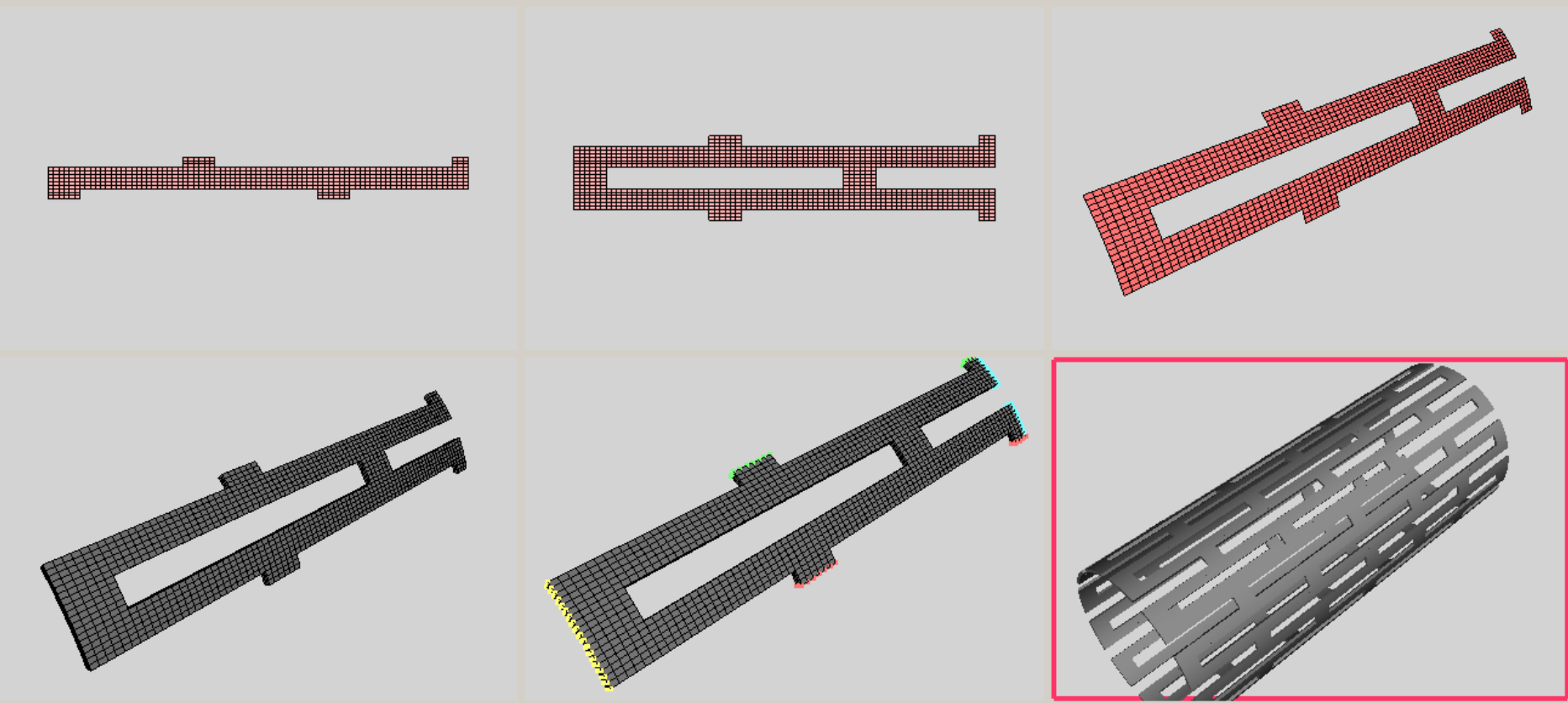
- Prototyping (Kink)



• Stent modelling

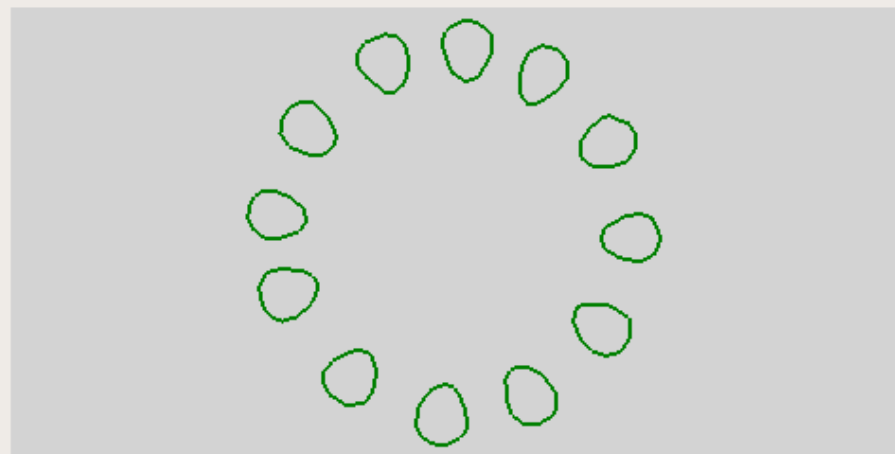
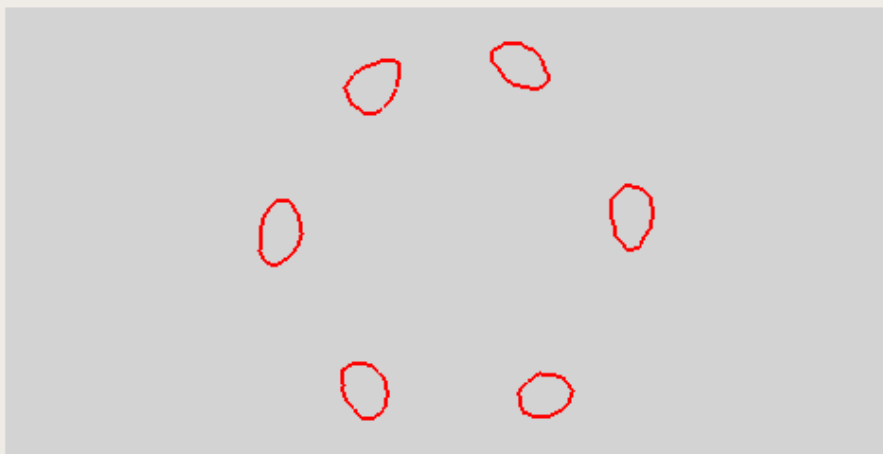
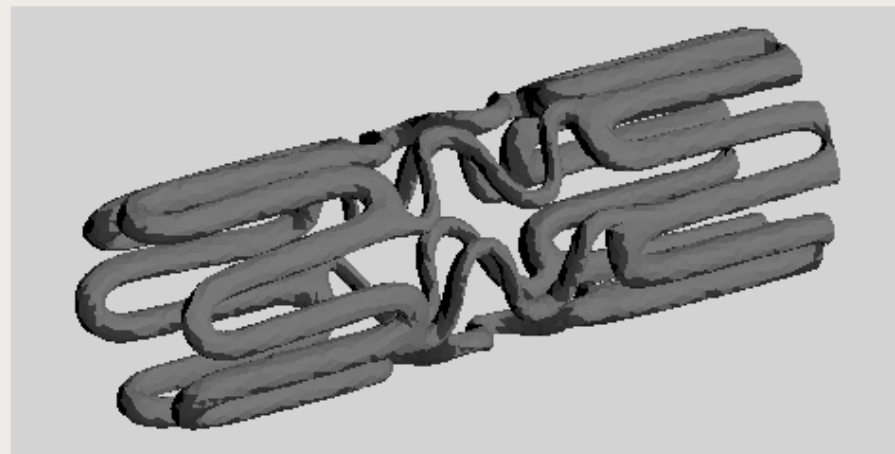
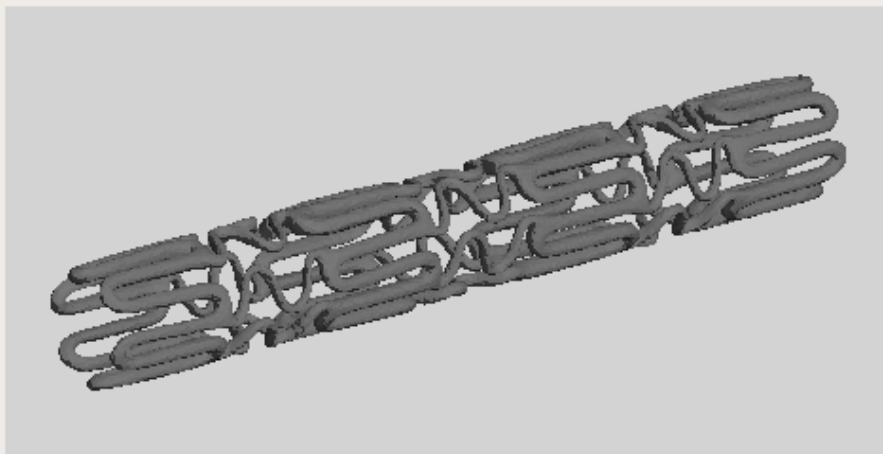
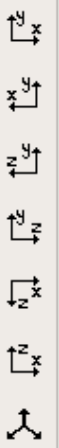
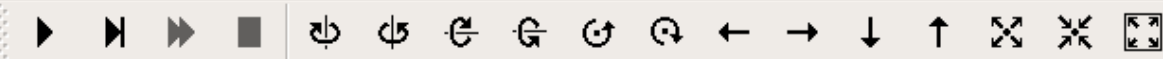
File Settings Viewport Camera Actions Scripts Geometry Formex Surface Mesh Tools Draw Jobs Postproc Help

▶ ▶▶ ■ ↺ ↻ ↶ ↷ ↸ ↹ ↵ ↶ ↷ ↸ ↹ ↵ ↶ ↷ ↸ ↹ ↵



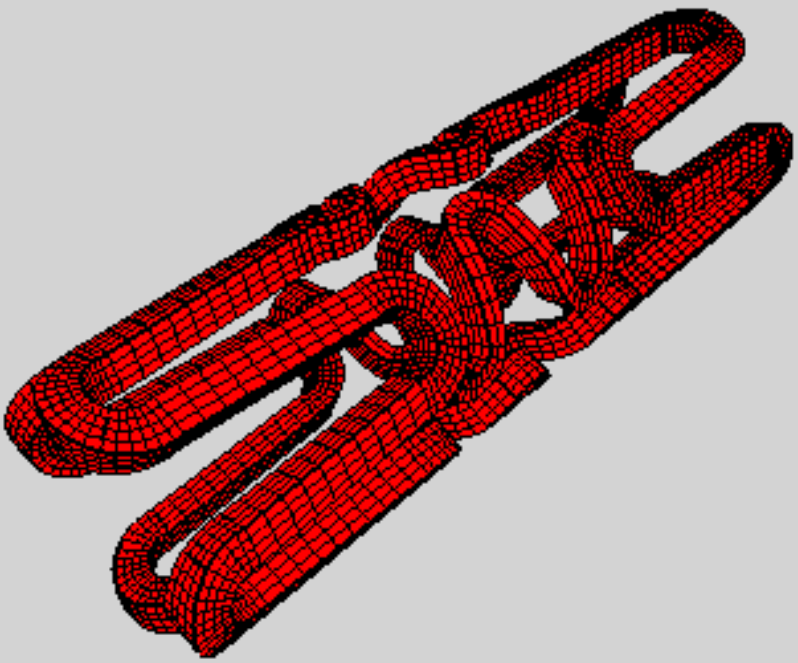
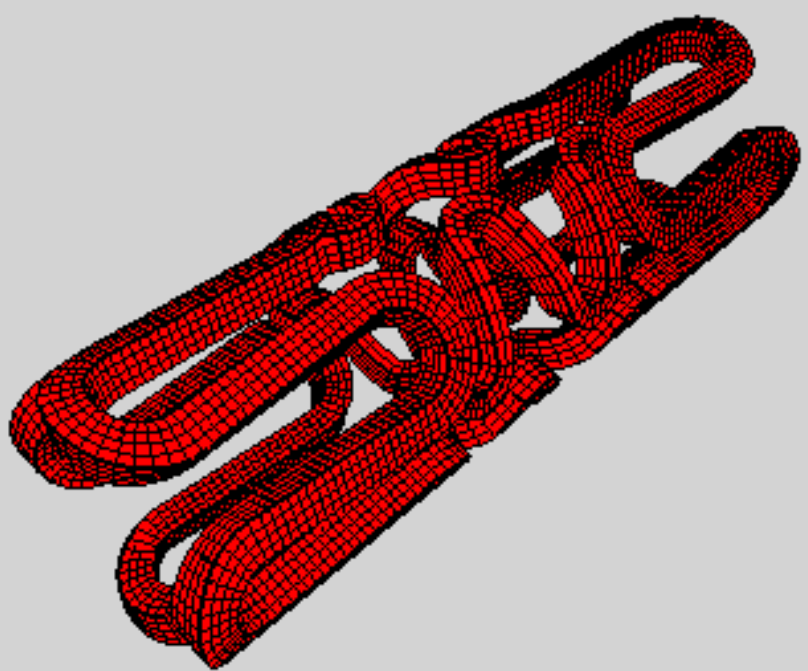
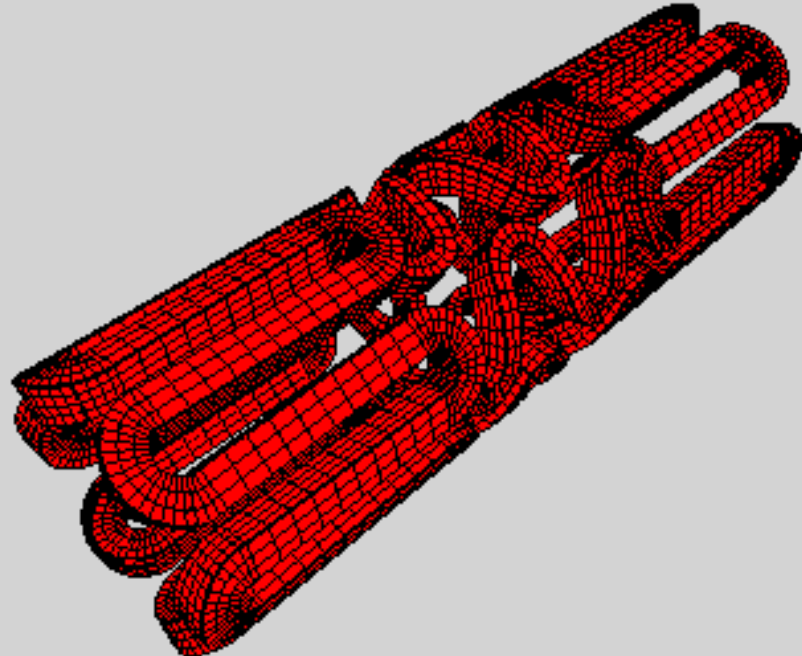
```
Running script (/home/bene/prj/pyformex/apps/giacomo/giacomo_stent.py) in 12 seconds
Finished script /home/bene/prj/pyformex/apps/giacomo/giacomo_stent.py in 12 seconds
GLINIT smoothwire True
GLINIT smooth True
Dict({'qu': -1, 'bo': False, 'wi': True, 'mu': False, 'hk': False, 'au': False, 'rc': False, 'fm': 'From Extension', 'fn': '/home/bene/prj/pyformex/apps/giacomo/giacomo_all.png'})
Setting workdir = /home/bene/prj/pyformex/apps/giacomo
[]
SAVE: quality=-1
```

Project: None Script: ● giacomo_stent.py Cwd: bene



float division
Running script (/home/pmortier/pyformex/pyformex/stent/cut3AtPlane.py)
Script finished
Running command: import -window "pyFormex 0.6a3" png:/home/pmortier/pyformex/pyformex/stent/intersection.png

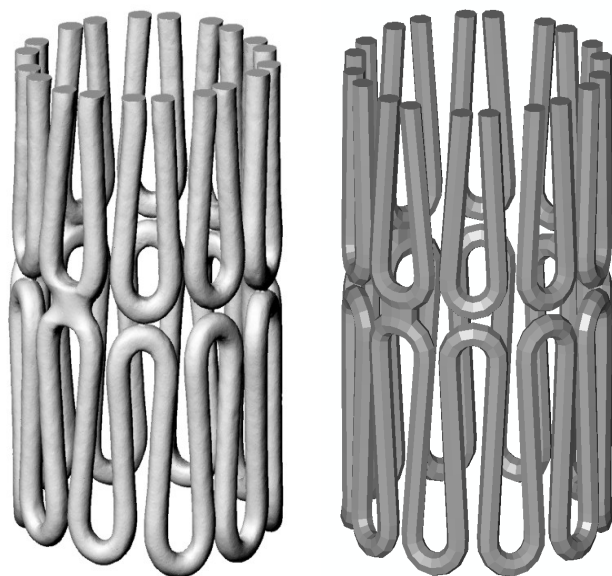




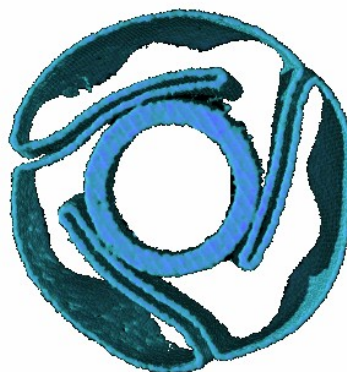
Basic ingredients of a computer simulation

Accurate geometrical models

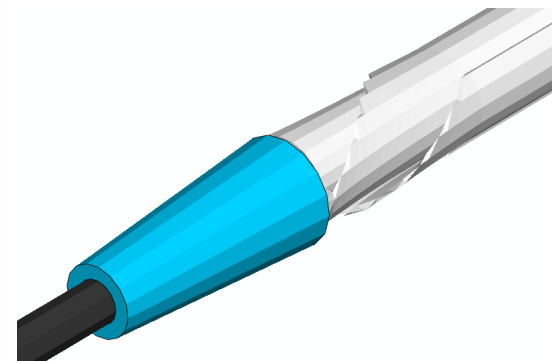
- Stent
- Folded balloon



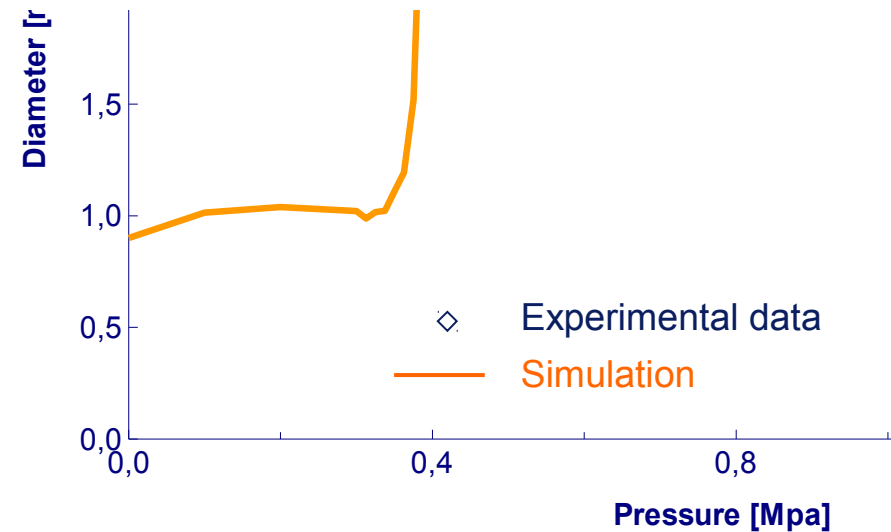
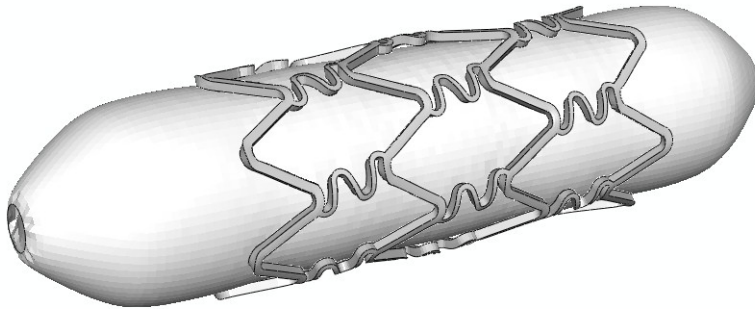
microCT → model



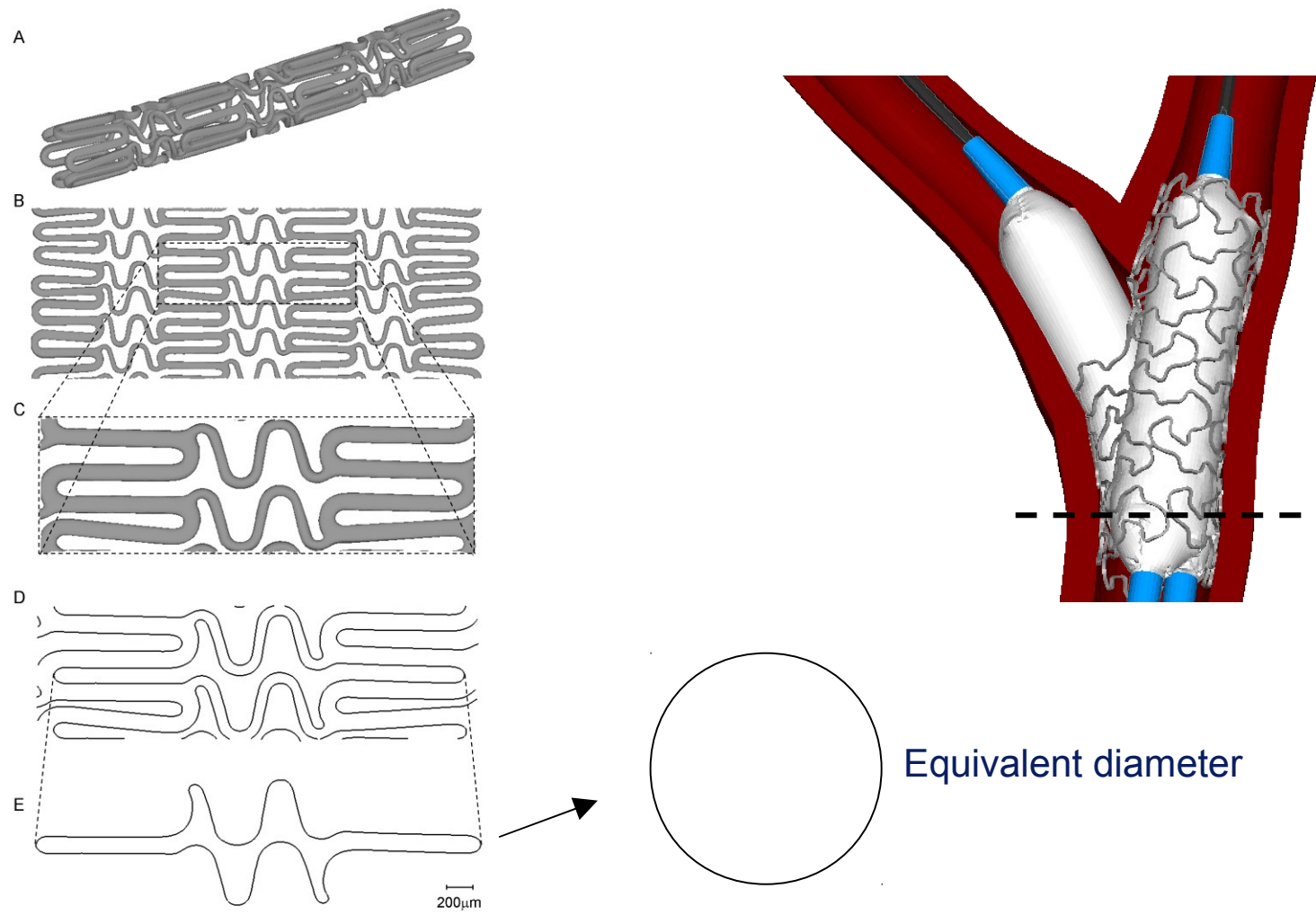
microCT → model

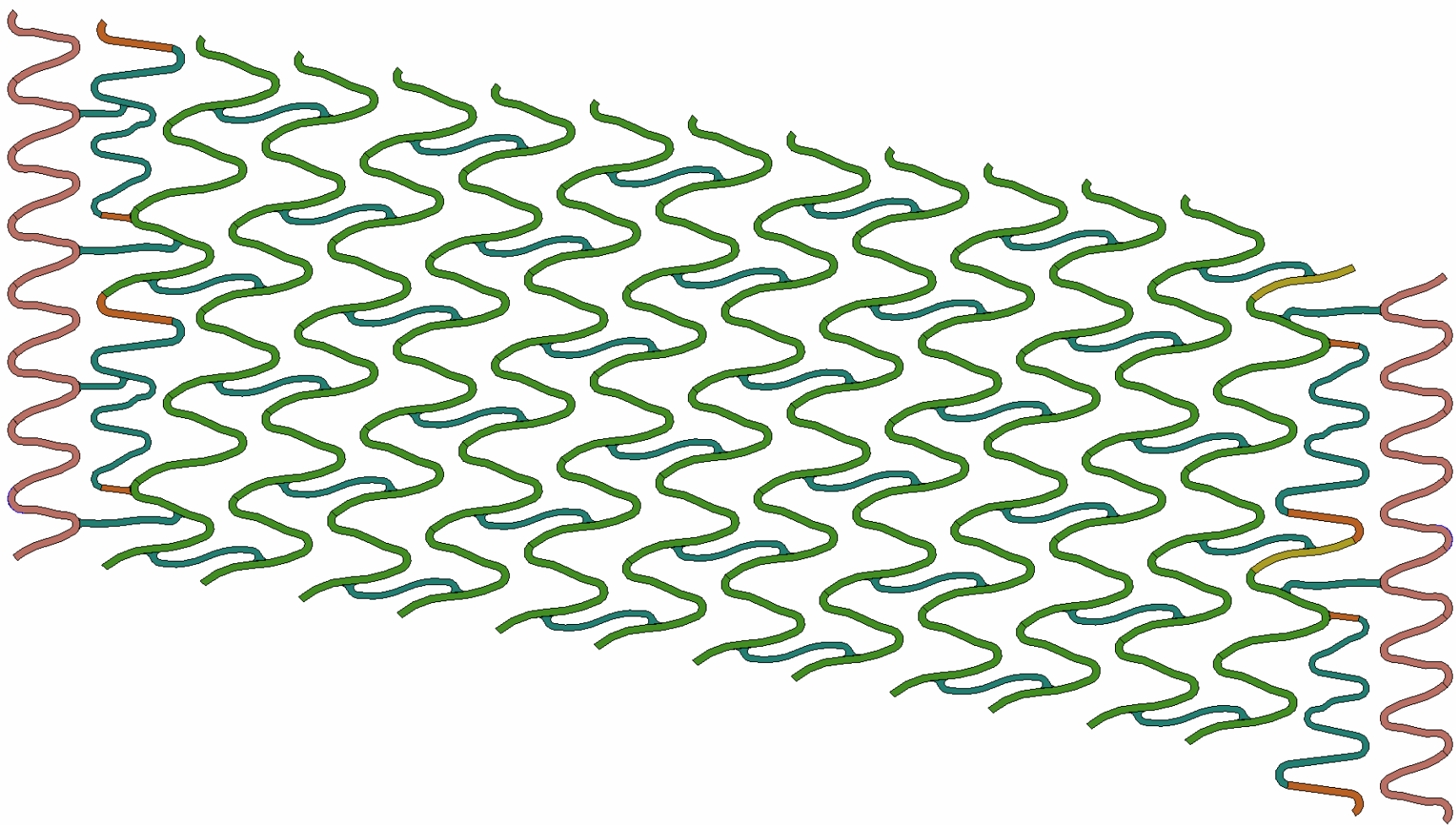


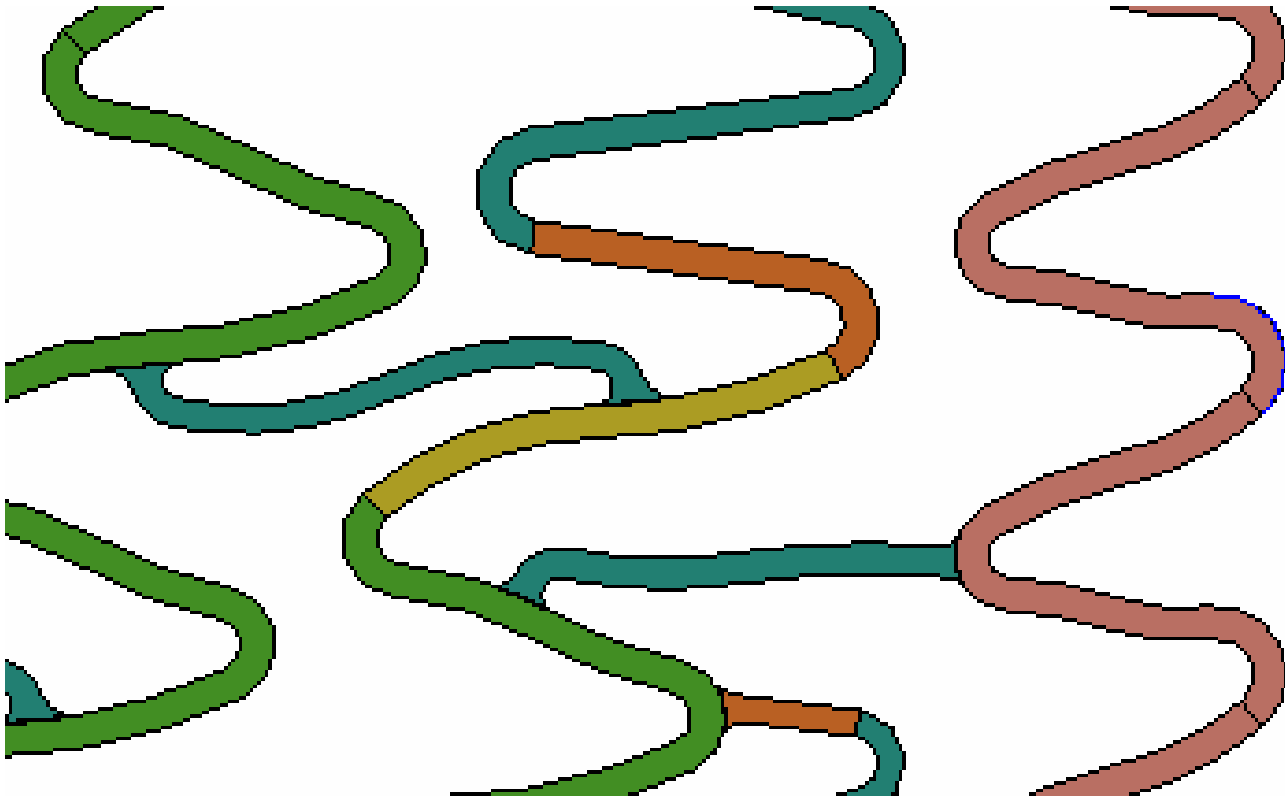
Model with folded balloon corresponds well with data provided by the manufacturer

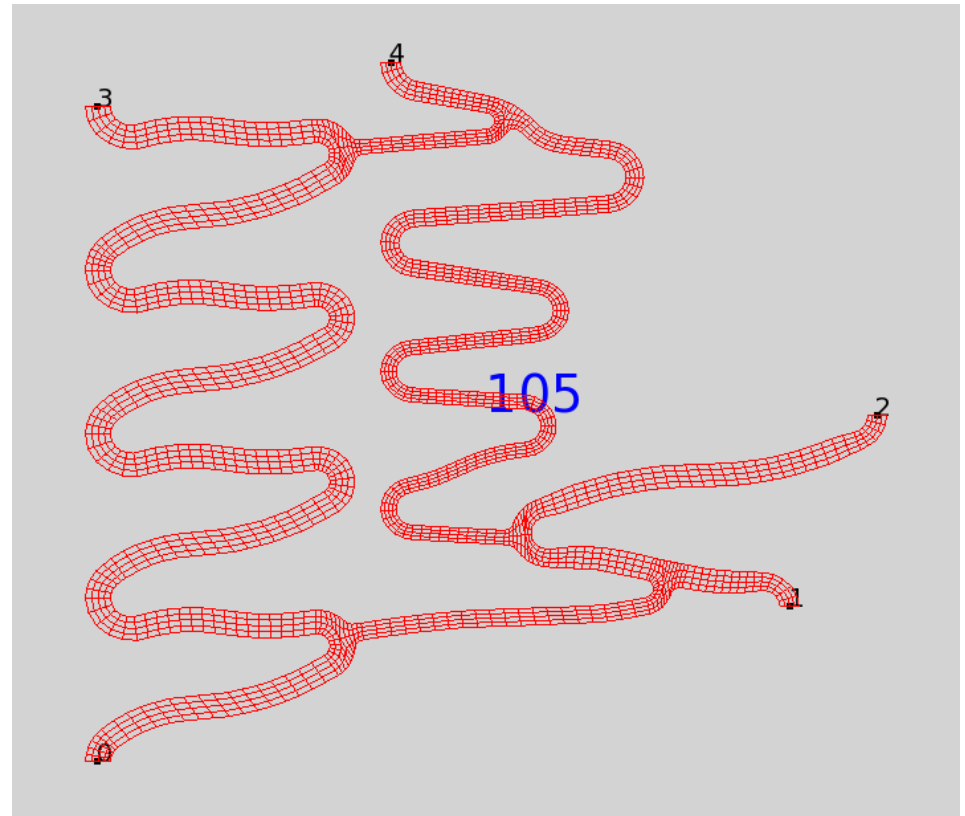
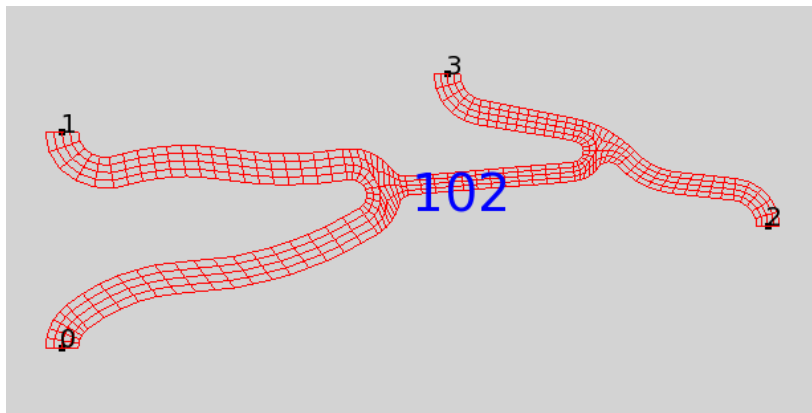
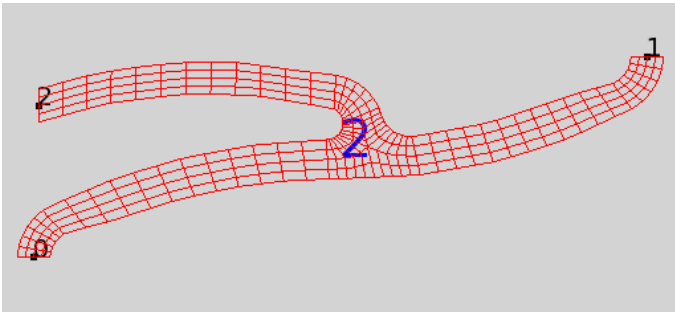
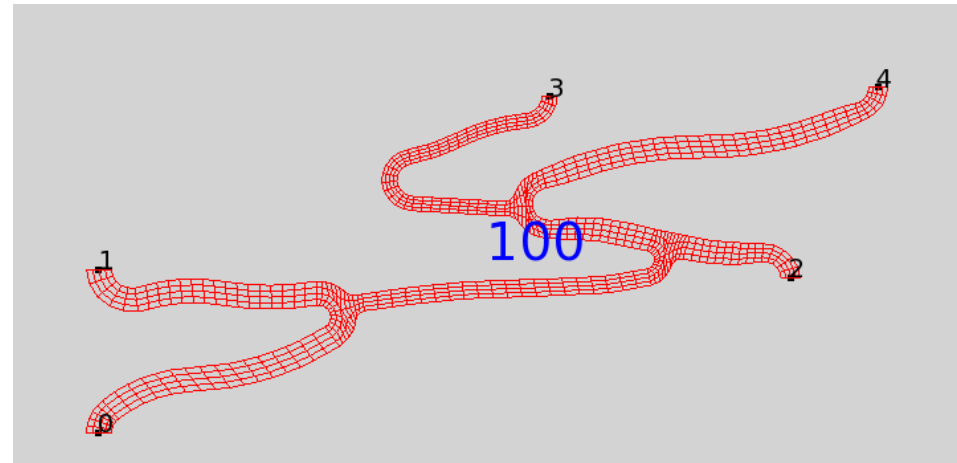
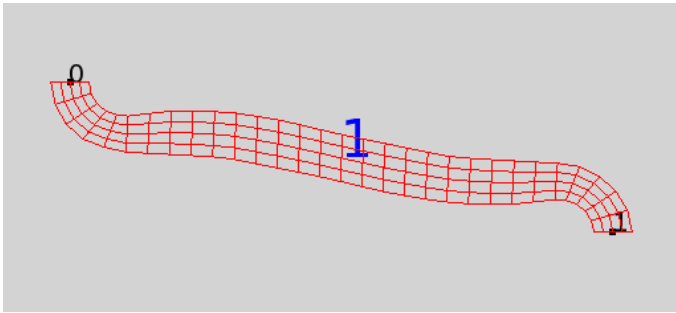


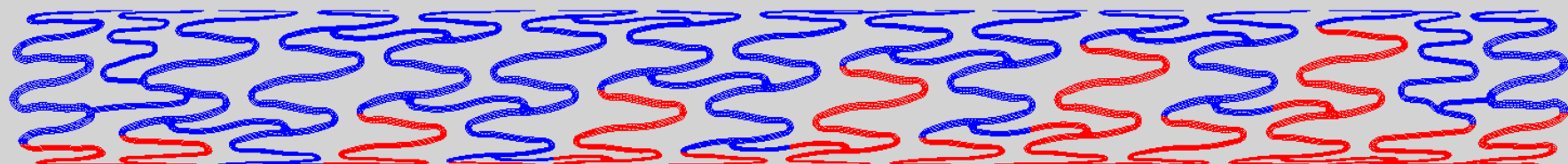
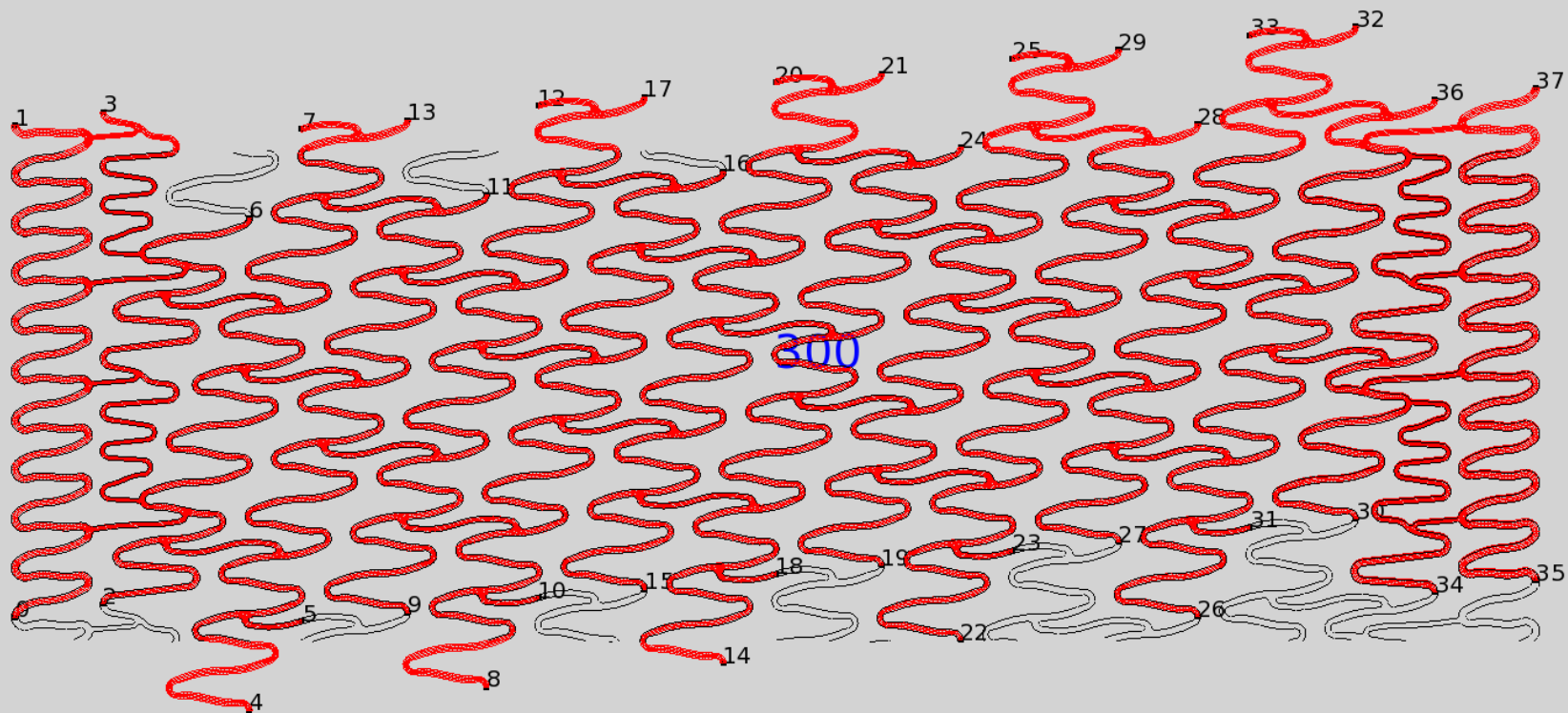
Assess maximal cell size

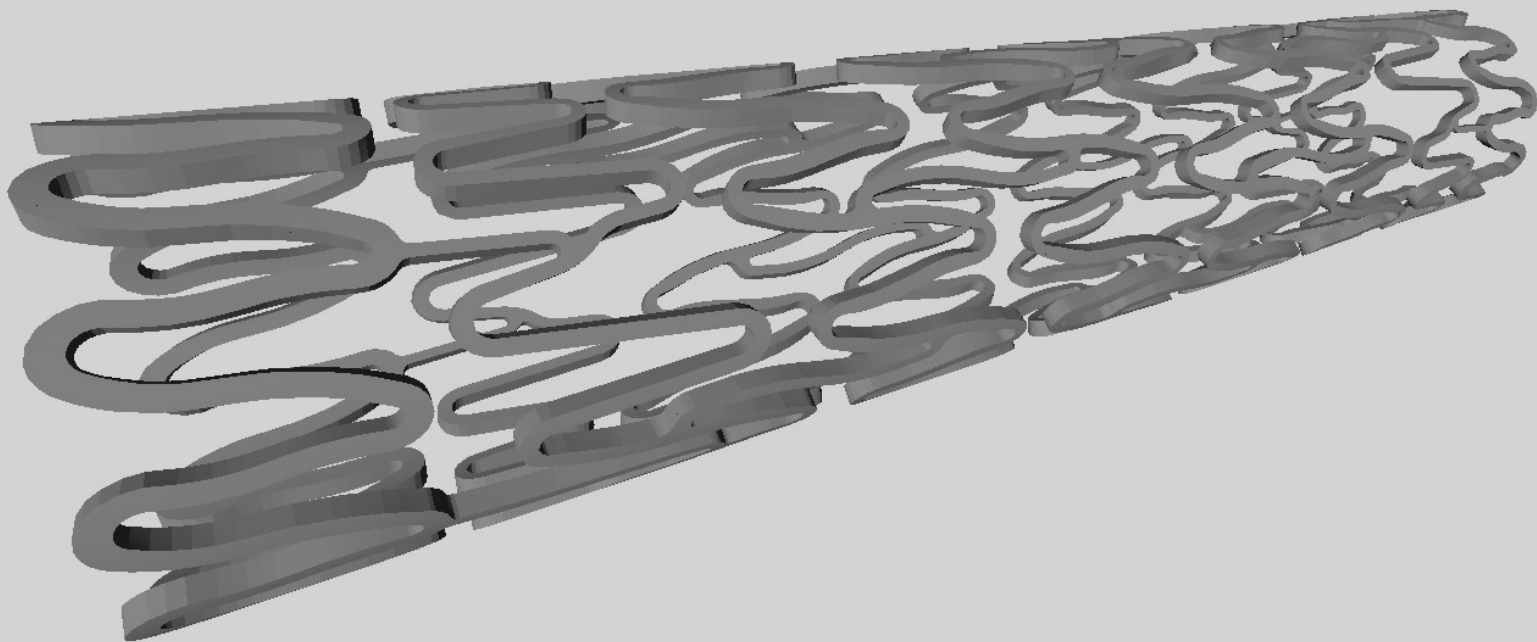
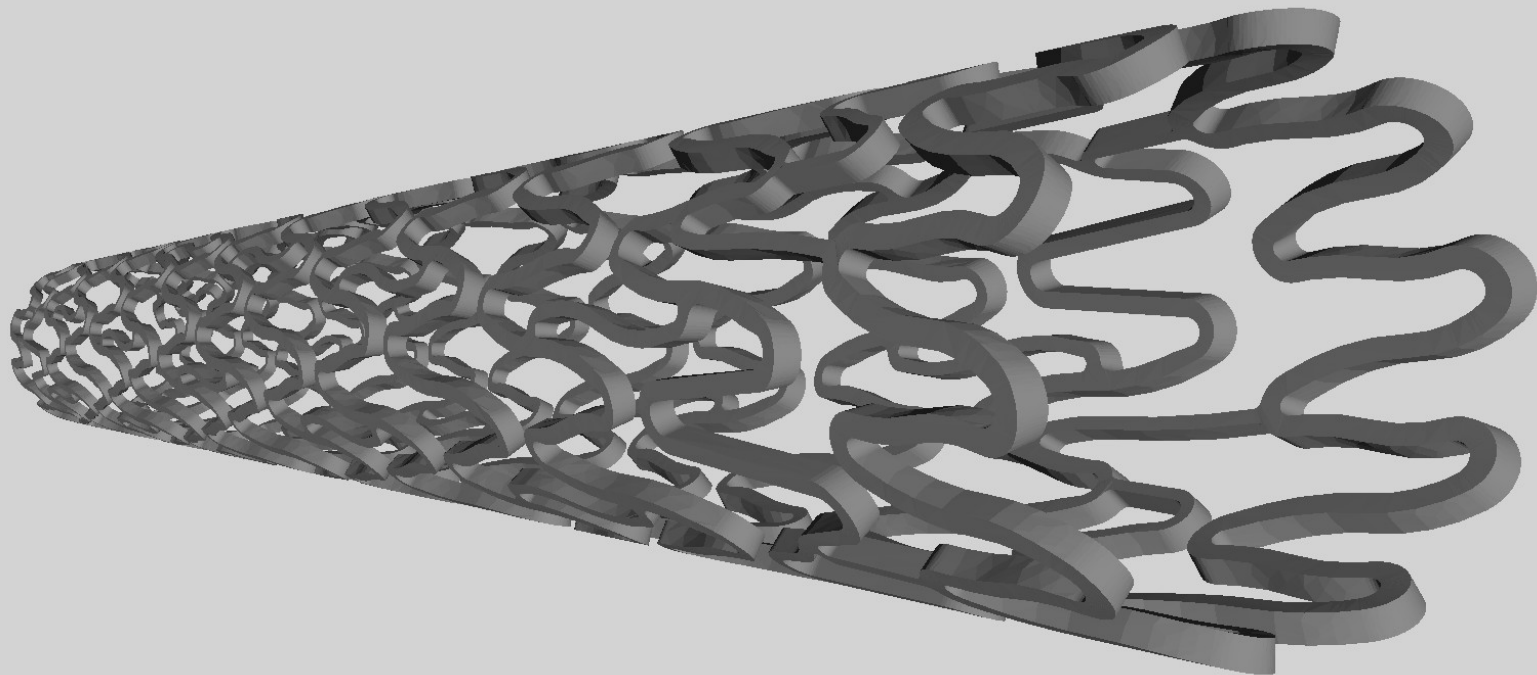








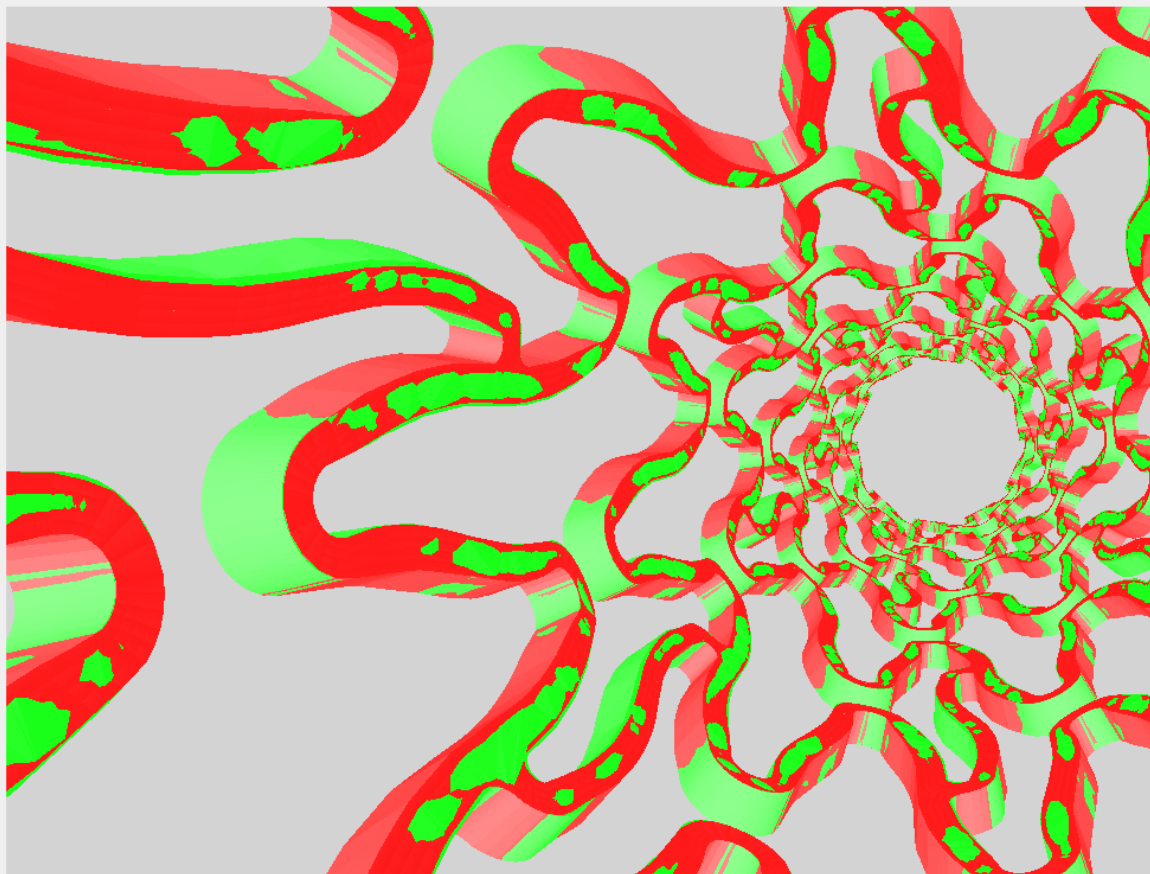




pyFormex 0.8-a1

File Settings Viewport Camera Actions Views Scripts Surface Formex Tools Postproc Jobs

Unroll Help



- Import a stent geometry
- Measure the stent geometry
- Unroll the stent geometry
- Create the midplane of the unrolled stent
- Clip a rough part from the midplane
- Cut cells from a part
- Split cell circumference
- Approximate curves with Splines
- Mesh Between Splines
- Set cell end points
- Show Cells
- Show Cells on midplane
- Show Cell Node numbers
- Print All Cells
- Merge some nodes from the cell
- Set Connection options
- Connect cells
- Report endpoint distances
- Fix inconsistencies in mesh 300
- Roll mesh into stent
- Show match between original and rolled
- Export 3D model to Abaqus
- Reload Menu
- Close Menu

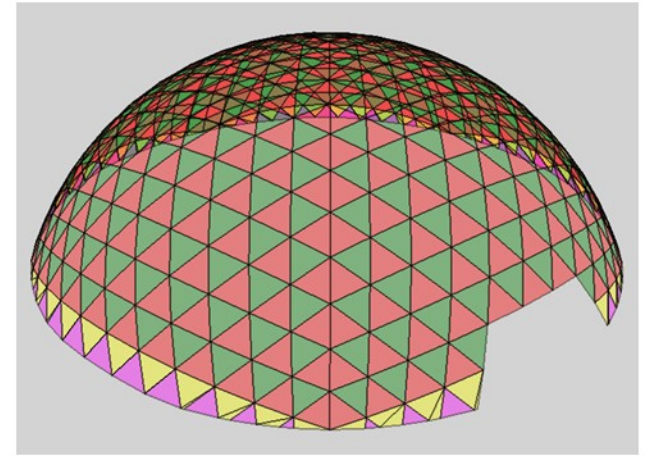
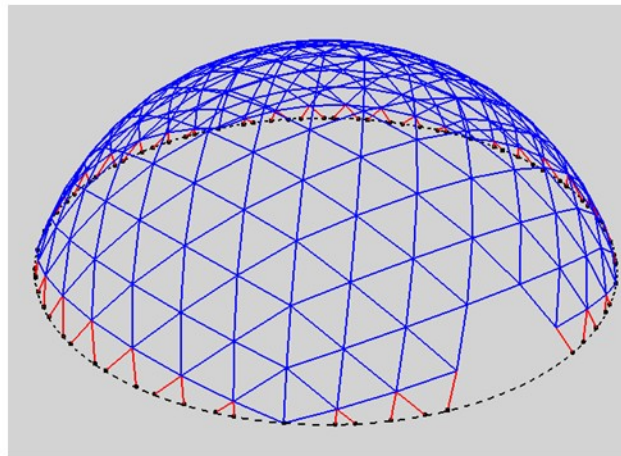
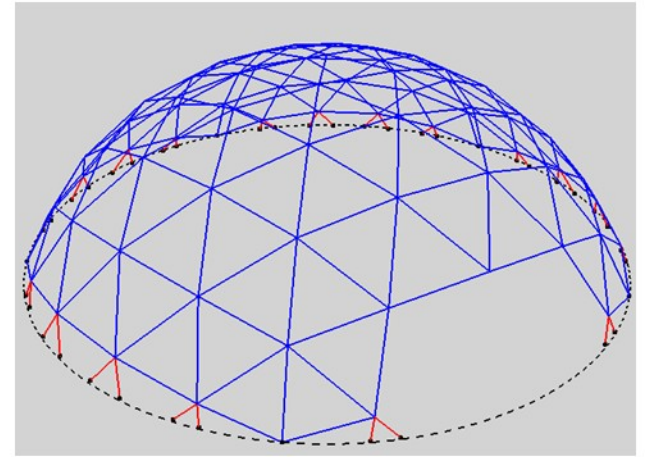
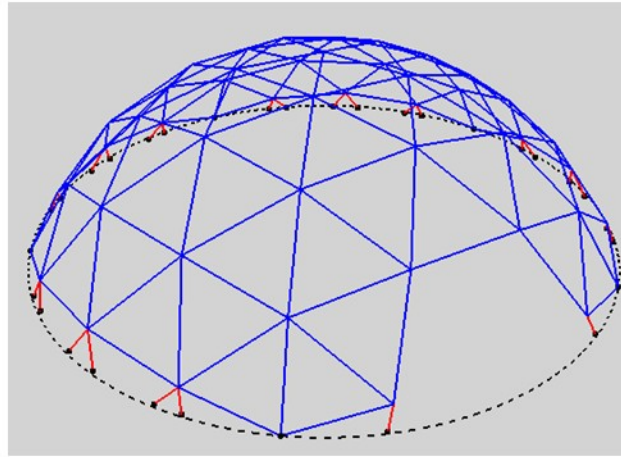
DEBUG: Formats available: ['bmp', 'ico', 'jpeg', 'jpg', 'png', 'ppm', 'tif', 'tiff', 'xbm', 'xpm', 'tex', 'ps', 'svg', 'pdf', 'eps', 'pgf', 'ppm', 'png', 'jpeg', 'rast', 'tiff', 'xwd', 'y4m']

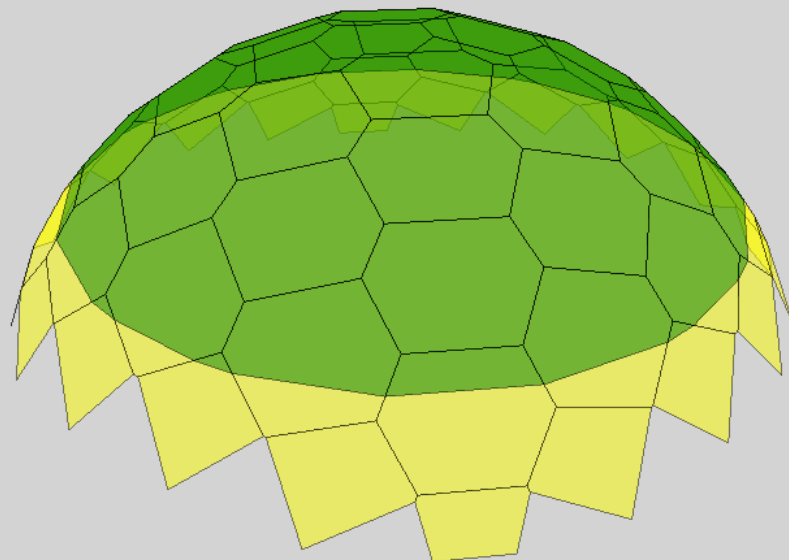
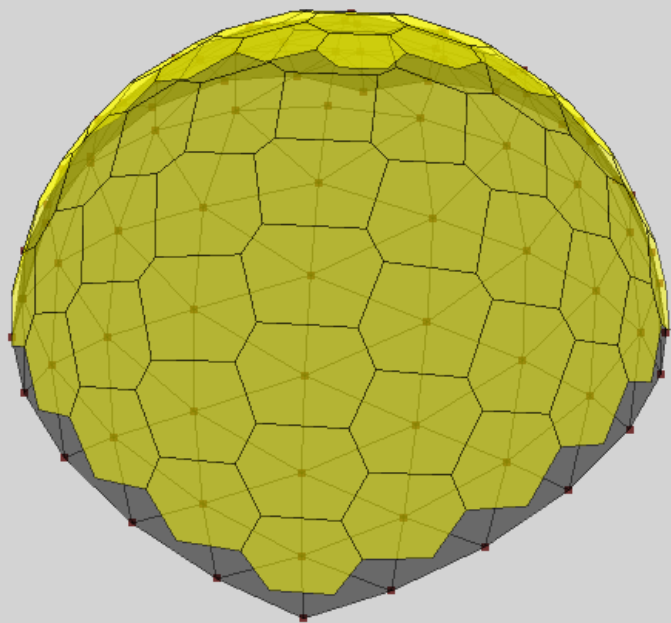
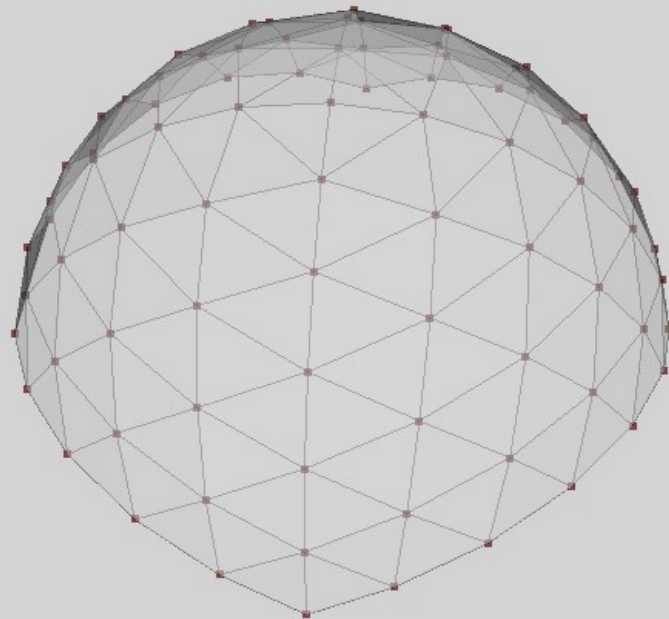
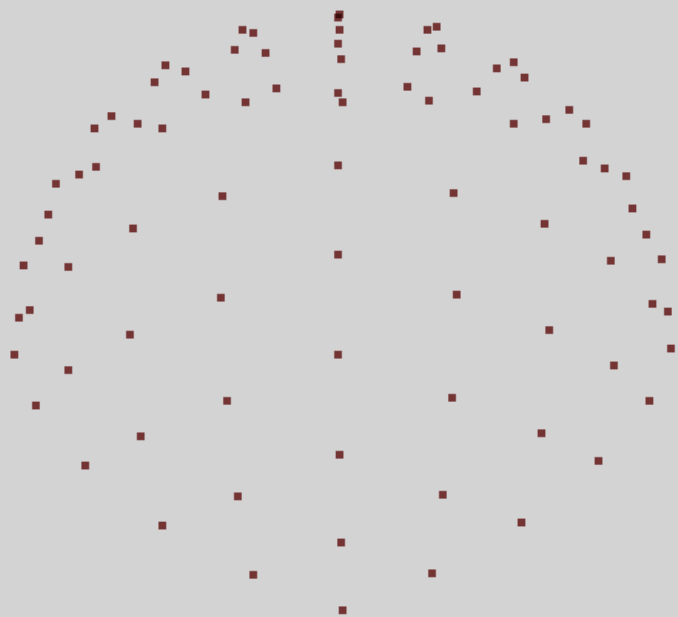
Start multisave mode to files: /home/bene/prj/pyformex/cortronik/gui-%03d.png (png)

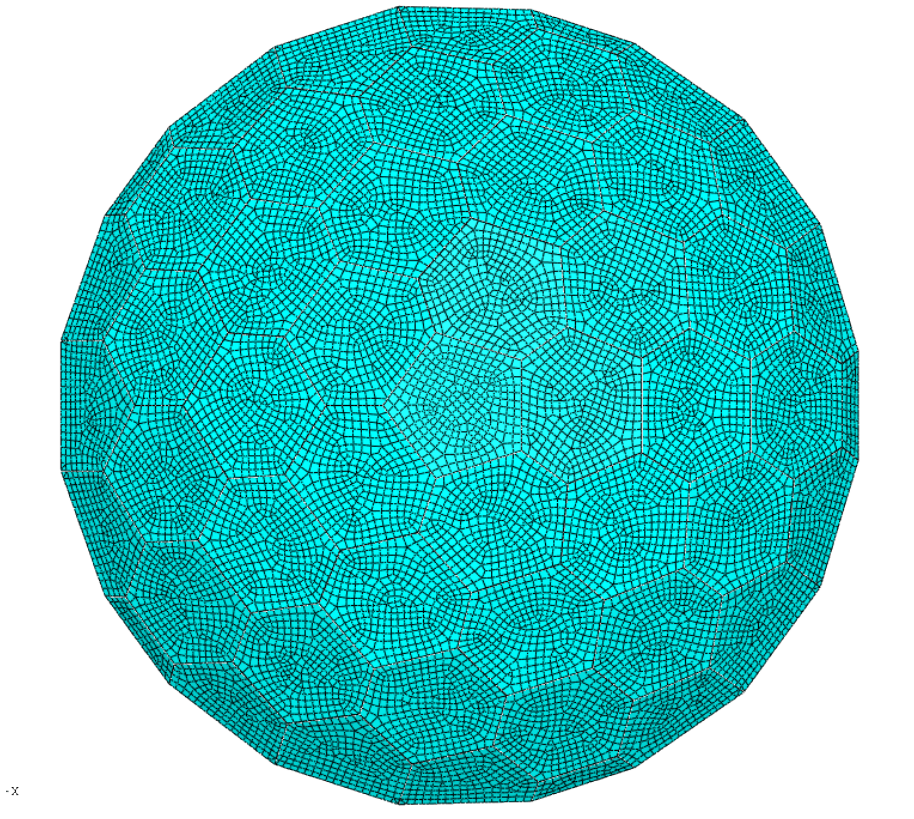
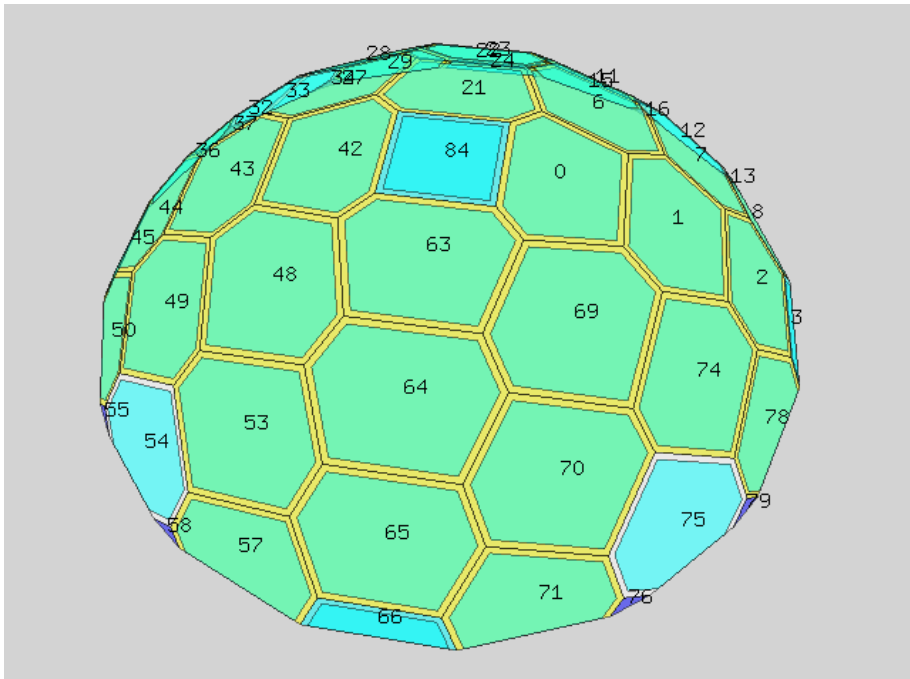
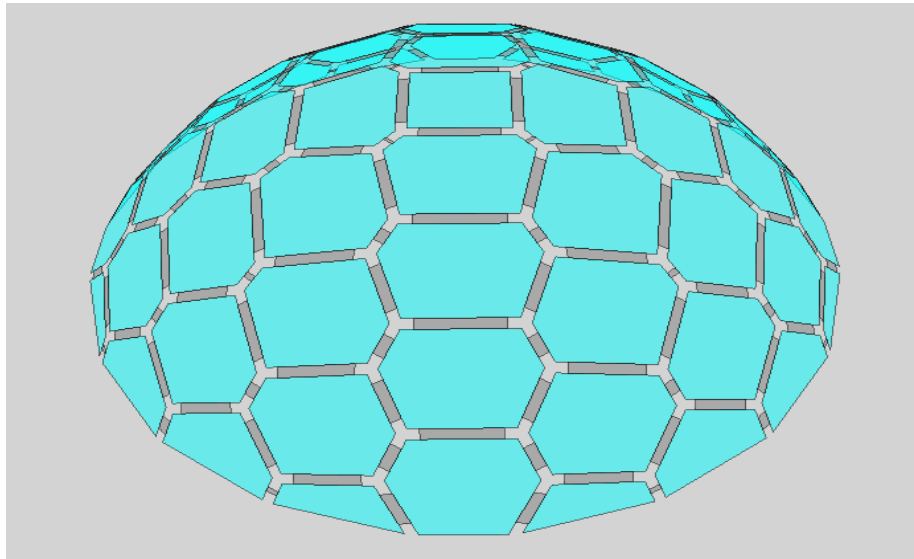
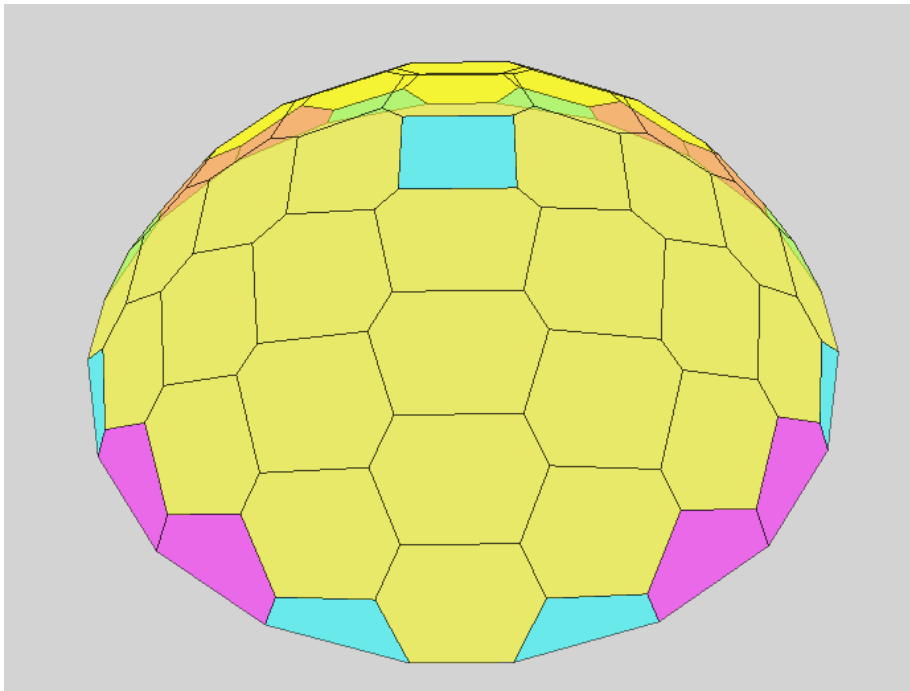
MULTISAVE (<utils.NameSequence object at 0xdc75950>, 'png', True, True, True, False, True)

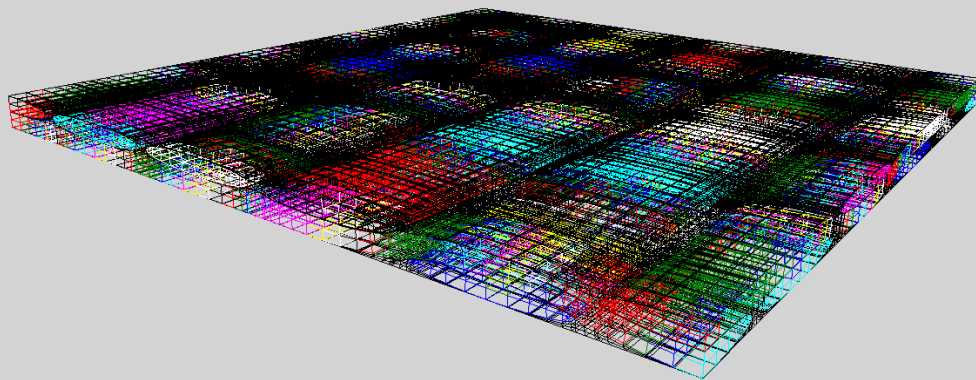
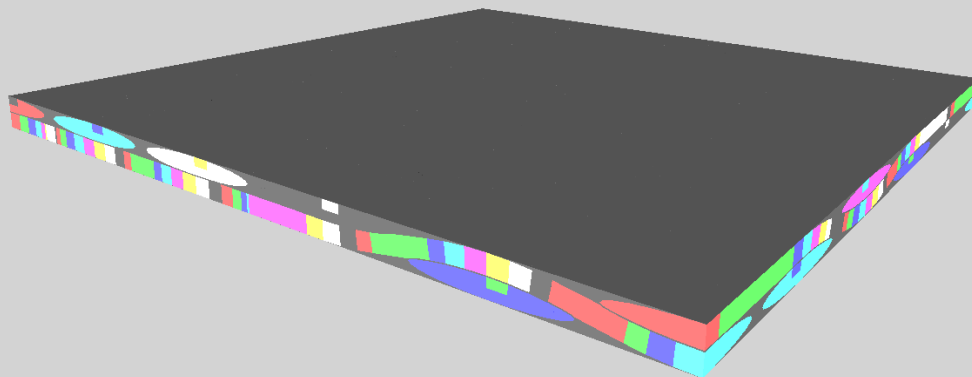
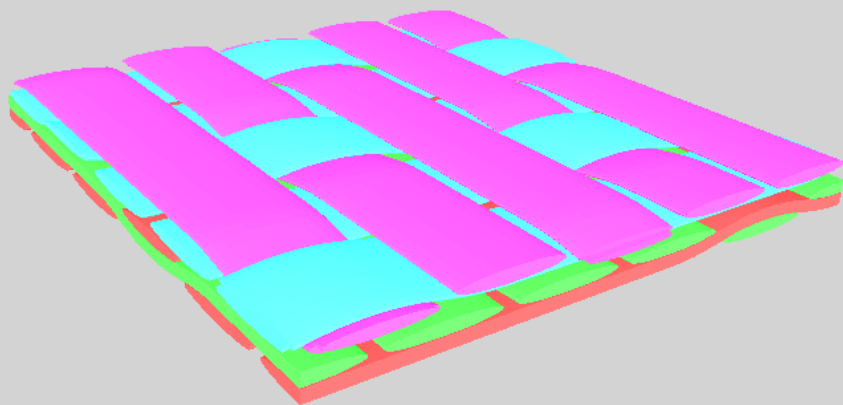
DEBUG: F2 pressed!

Project: t6l_20.pyf Script: unroll.py









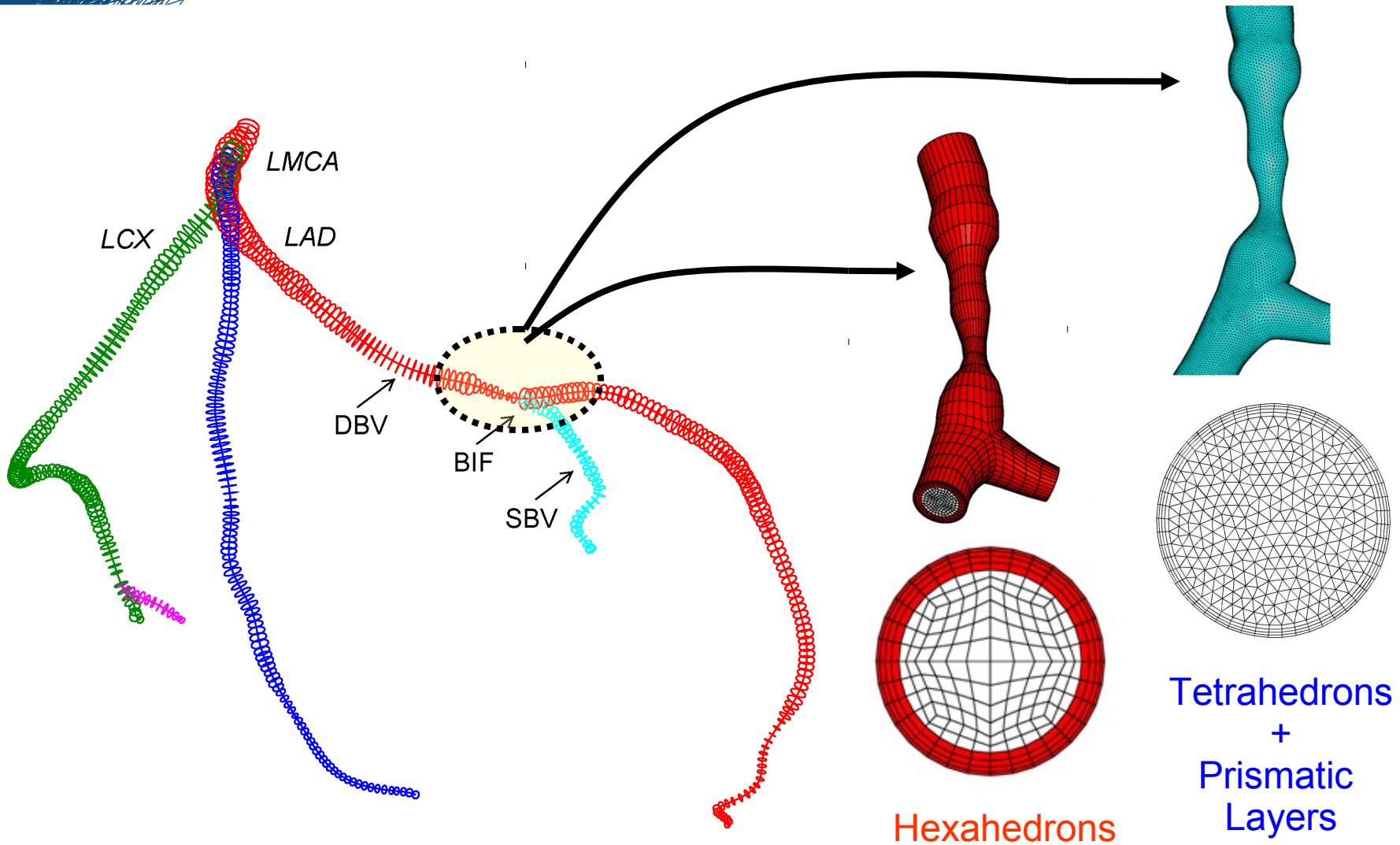
pyformex as HEX mesher for image-based CFD/FEA



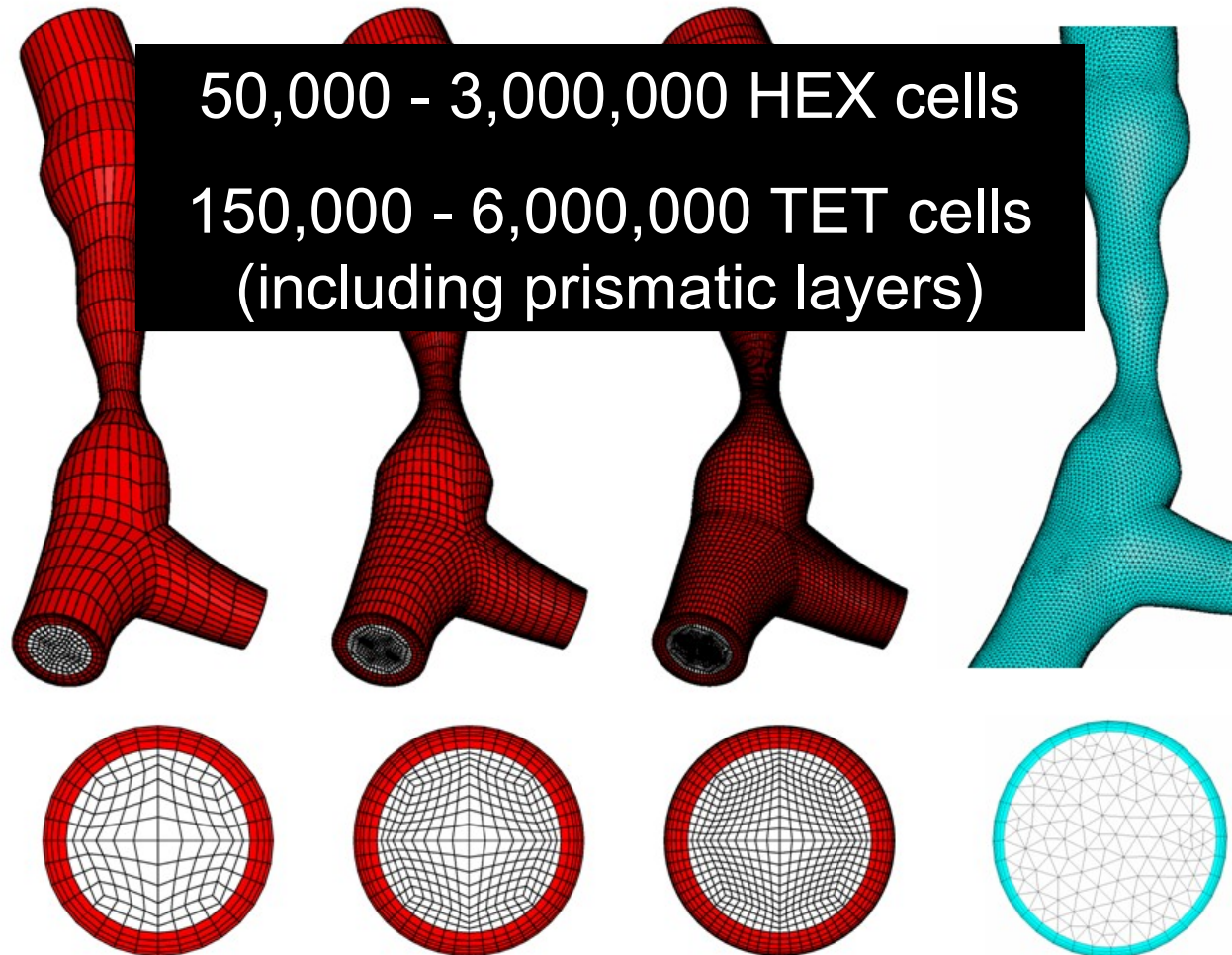
UNIVERSITEIT
GENT

Gianluca De Santis
Benedict Verhegghe

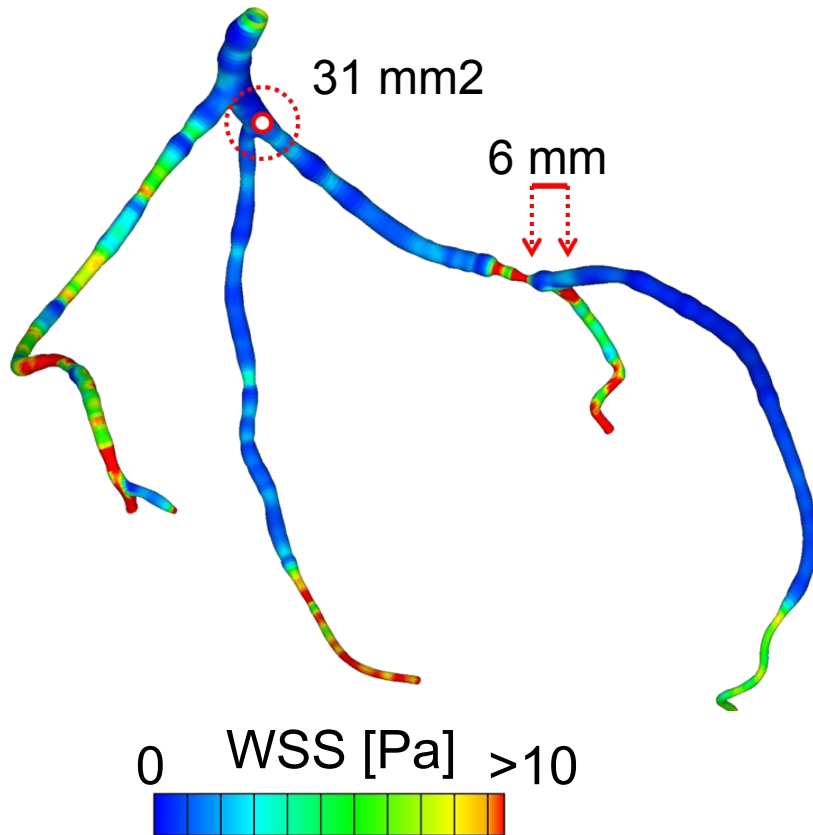
mesh-dependent ?



same LCA different meshes



WSS magnitude



- mean WSS

(Area-Averaged WSS on entire lumen surface)

- regional WSS

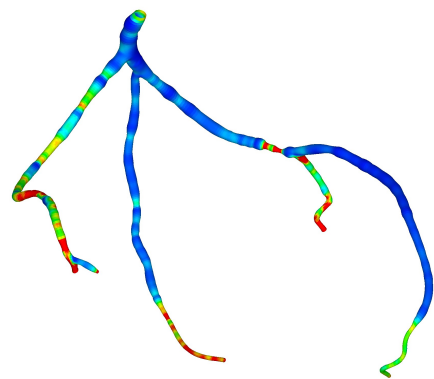
(Area-Averaged WSS on a small surface inside a bifurcation)

- local WSS

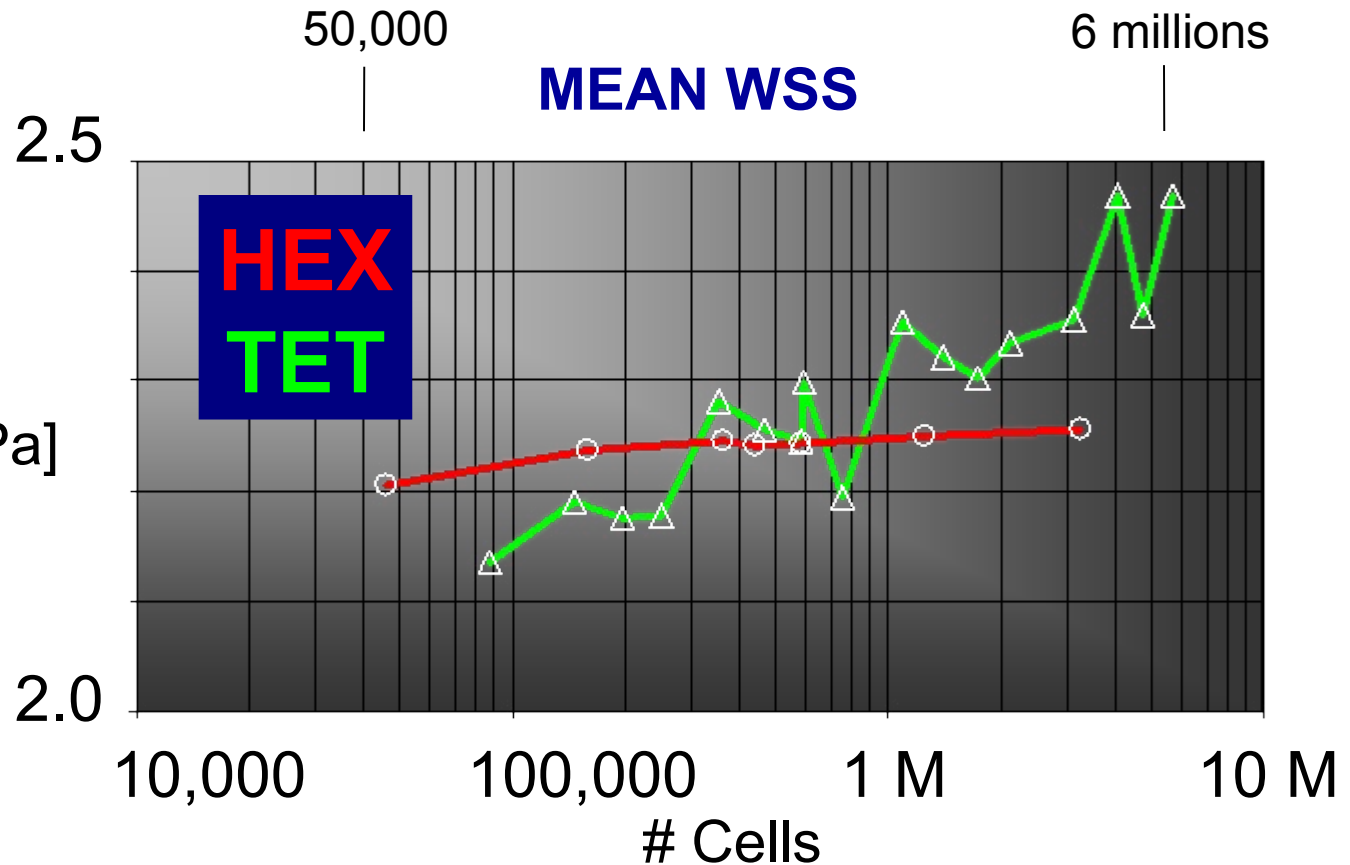
(nodal values of WSS along a line)

same post processing (with journal file)

area-averaged WSS on entire lumen surface

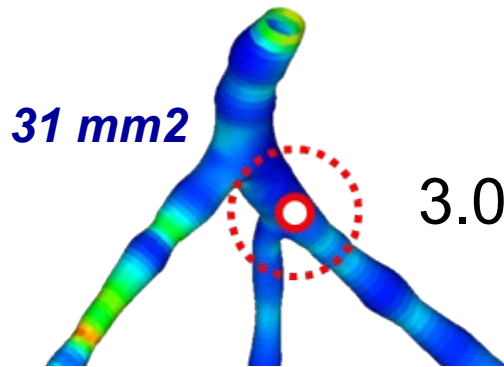


WSS [Pa]



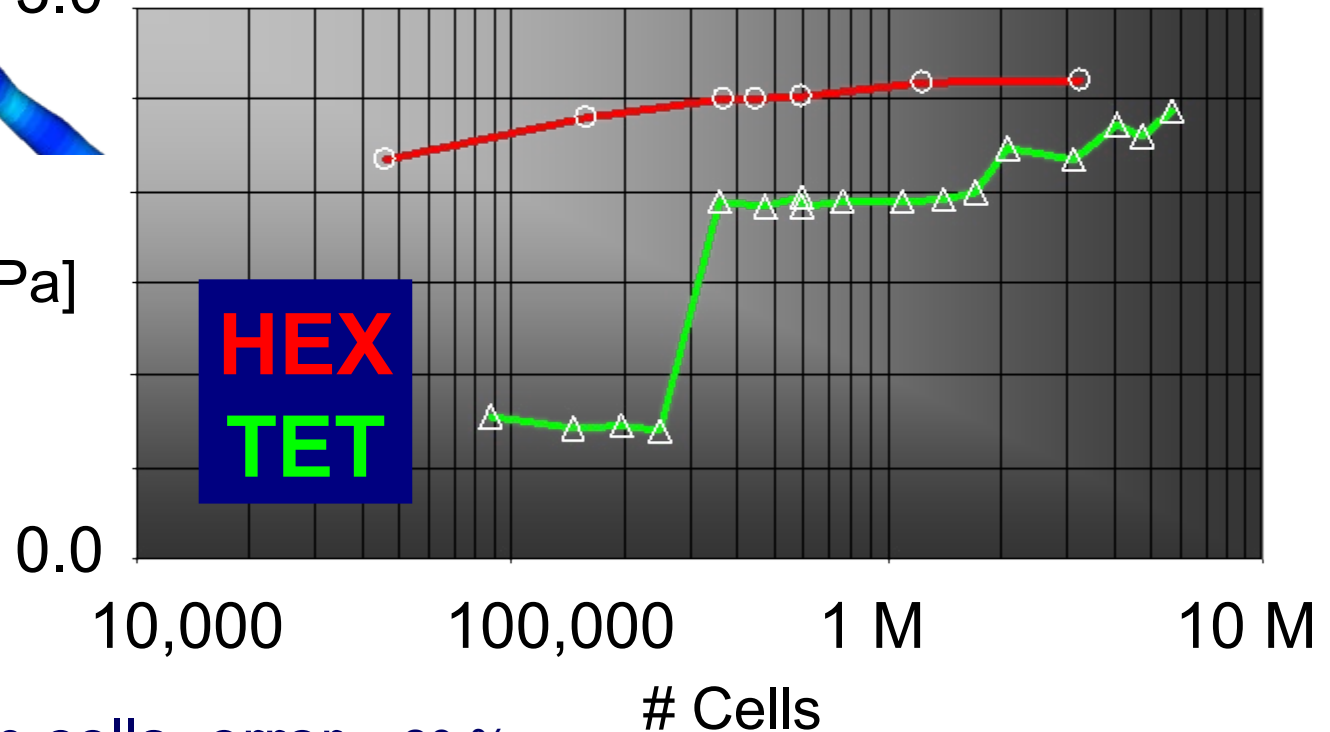
TET: after 1 m cells, error ~ 7 %, with oscillations

area-averaged WSS in a bifurcation



WSS [Pa]

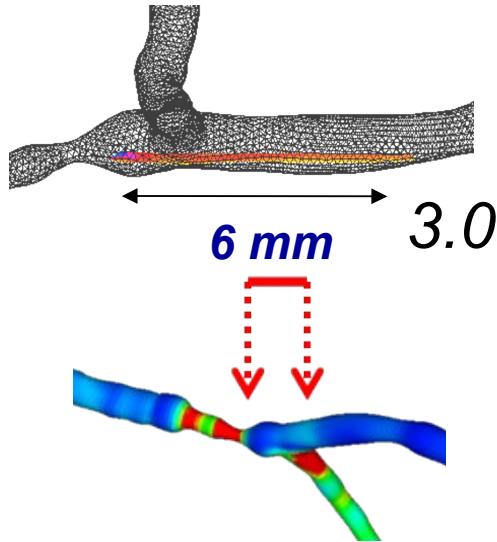
REGIONAL WSS



TET: after 1 m cells, error ~ 20 %.

Up to 300,000 does not approximate

WSS along a line



WSS [Pa]

0.0

0

6

mm

HEX

LOCAL WSS

TET

50,000
150,000
350,000
600,000
3,200,000

200,000
600,000
750,000
1,000,000
1,700,000
2,100,000
3,100,000
5,000,000

0

mm

6

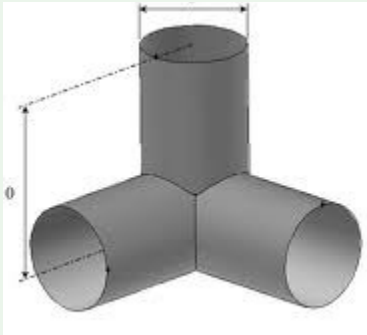
CPU time

- similar CPU time with similar number of cells (but Hex have ~ 3 times more nodes)
- same accuracy of the WSS (~ 5-10 %),
TET require 5x more cells = 14x longer CPU time
- TET mesh > 2 millions did not reduce the WSS error (WSS oscillates) but strongly increased the CPU time

structured hex mesh needs decomposition

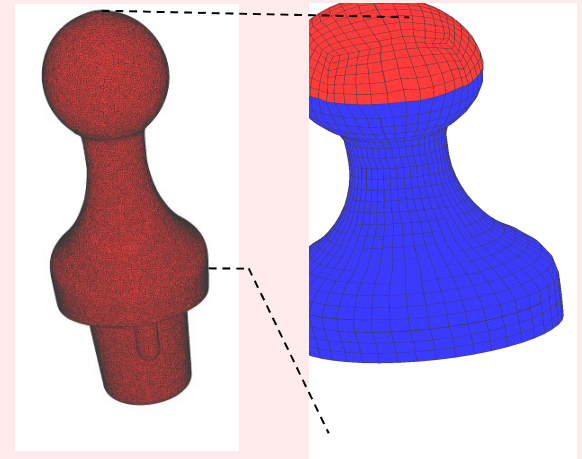
bottom-up

e.g. CAD, manufacturing
industry

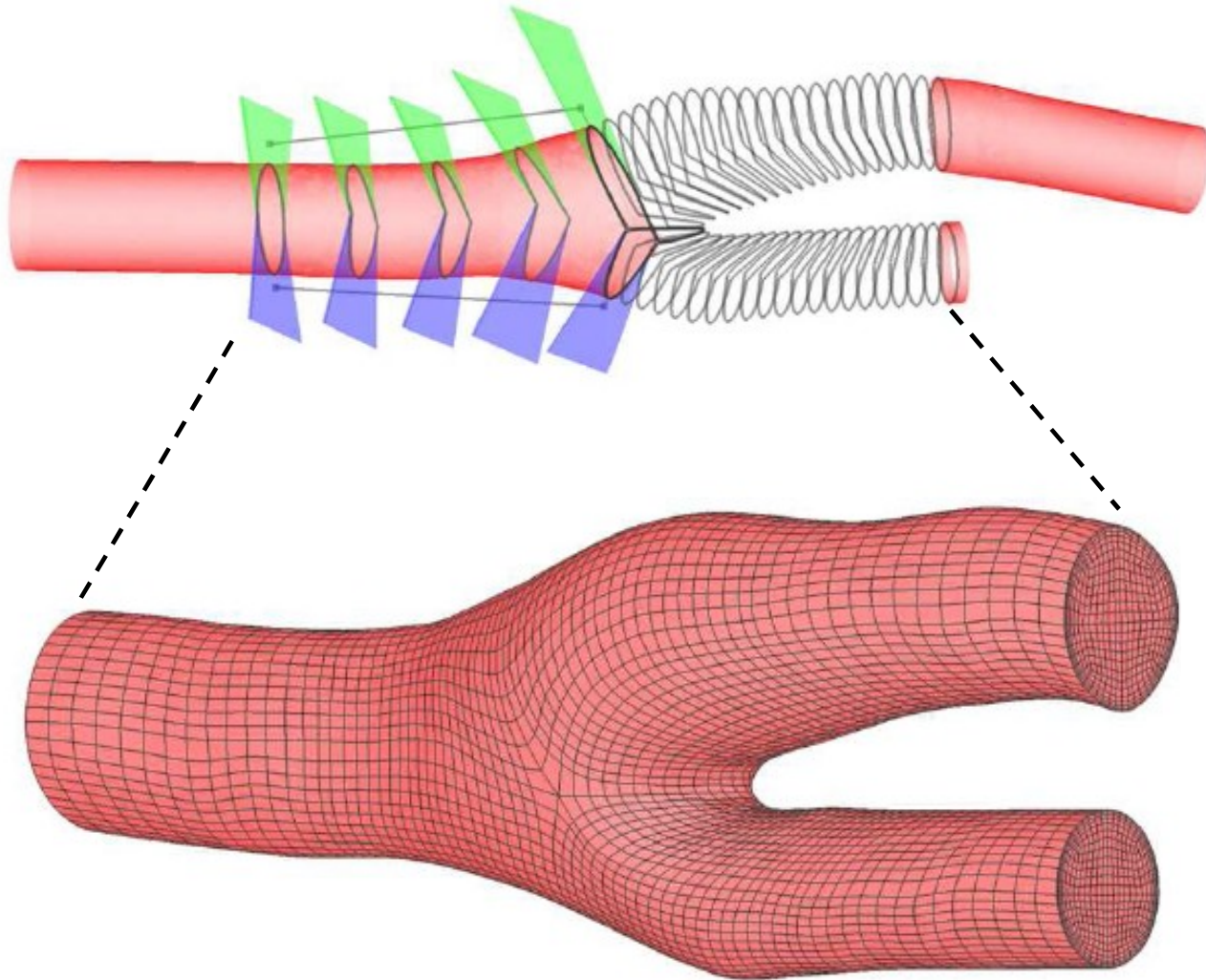


top-down

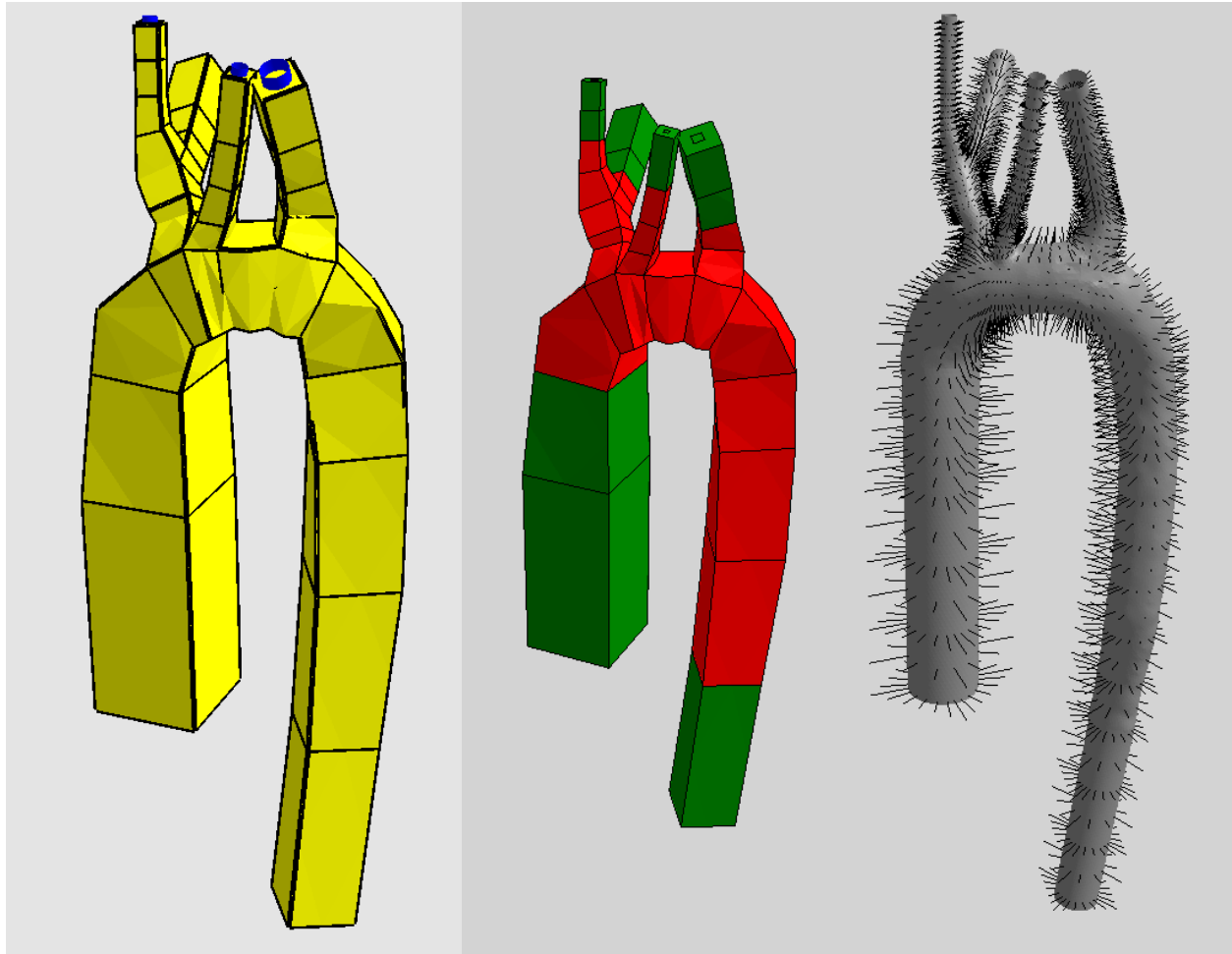
e.g. image-based,
from CT, MRI, US, μ CT



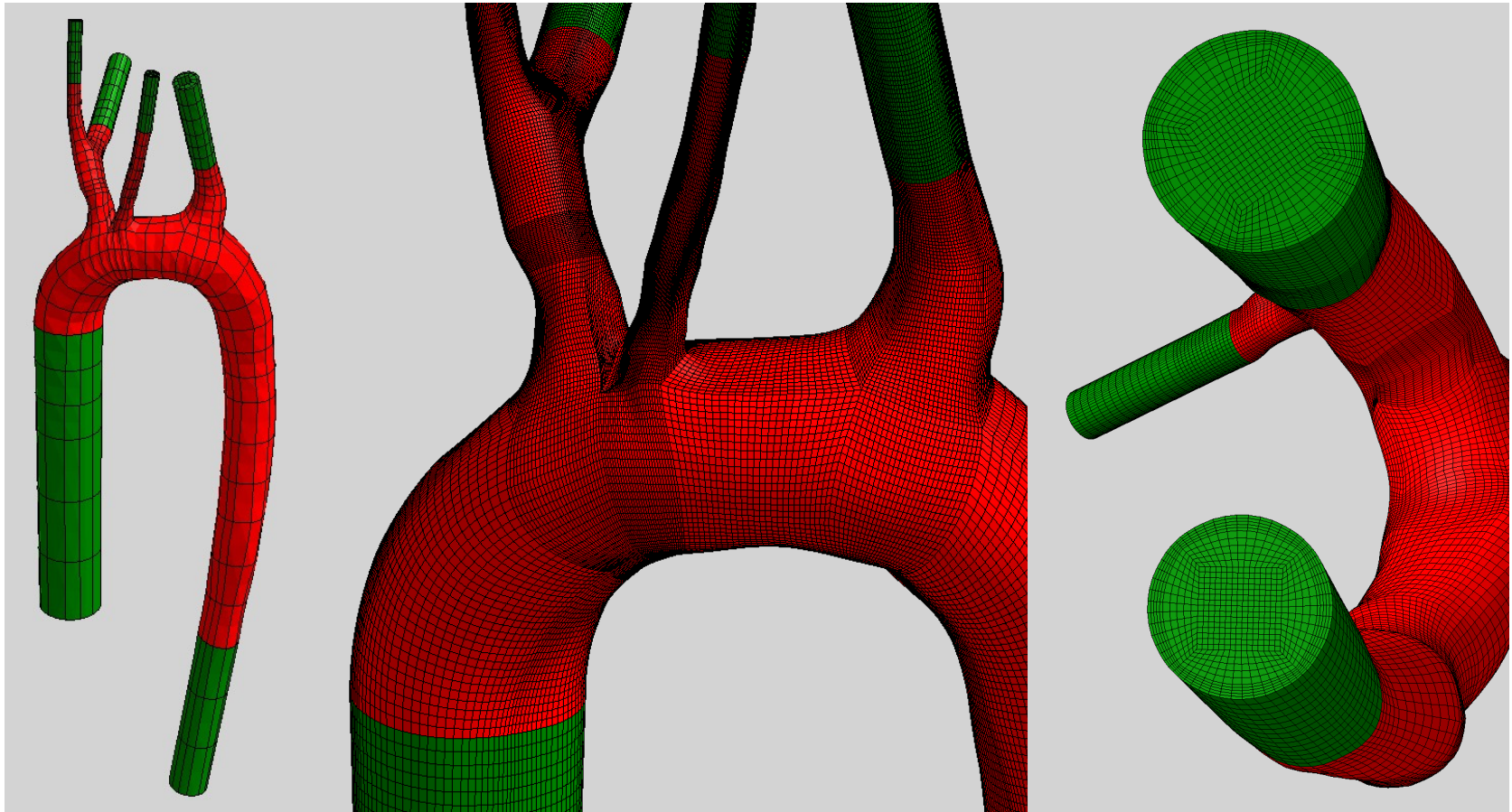
Single bifurcation

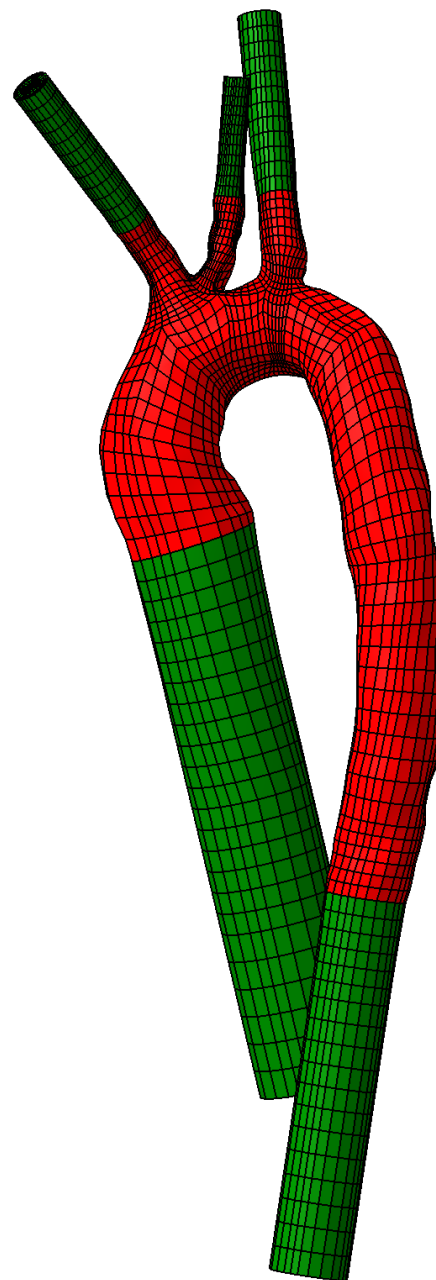
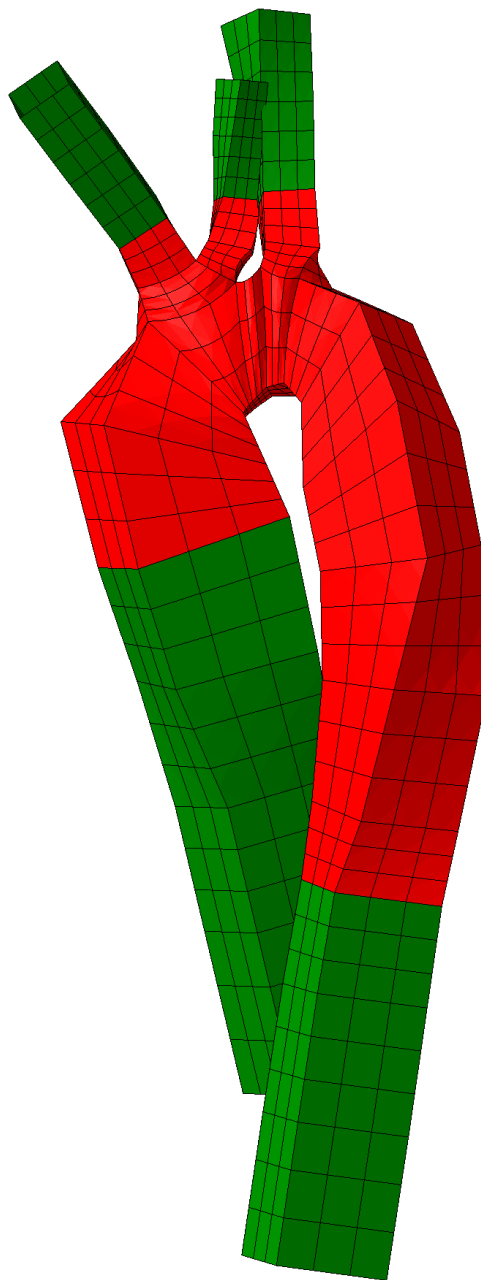
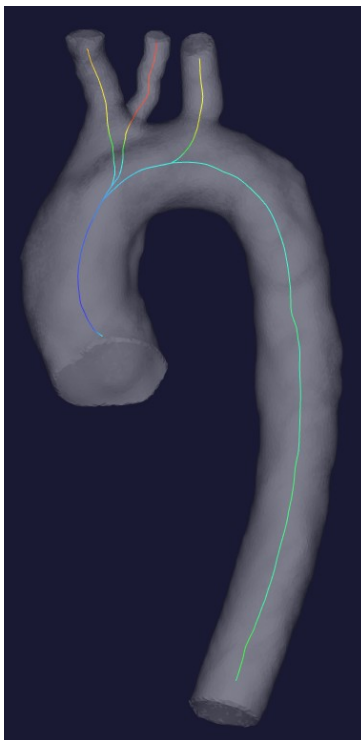


registered blocks around the vessel surface

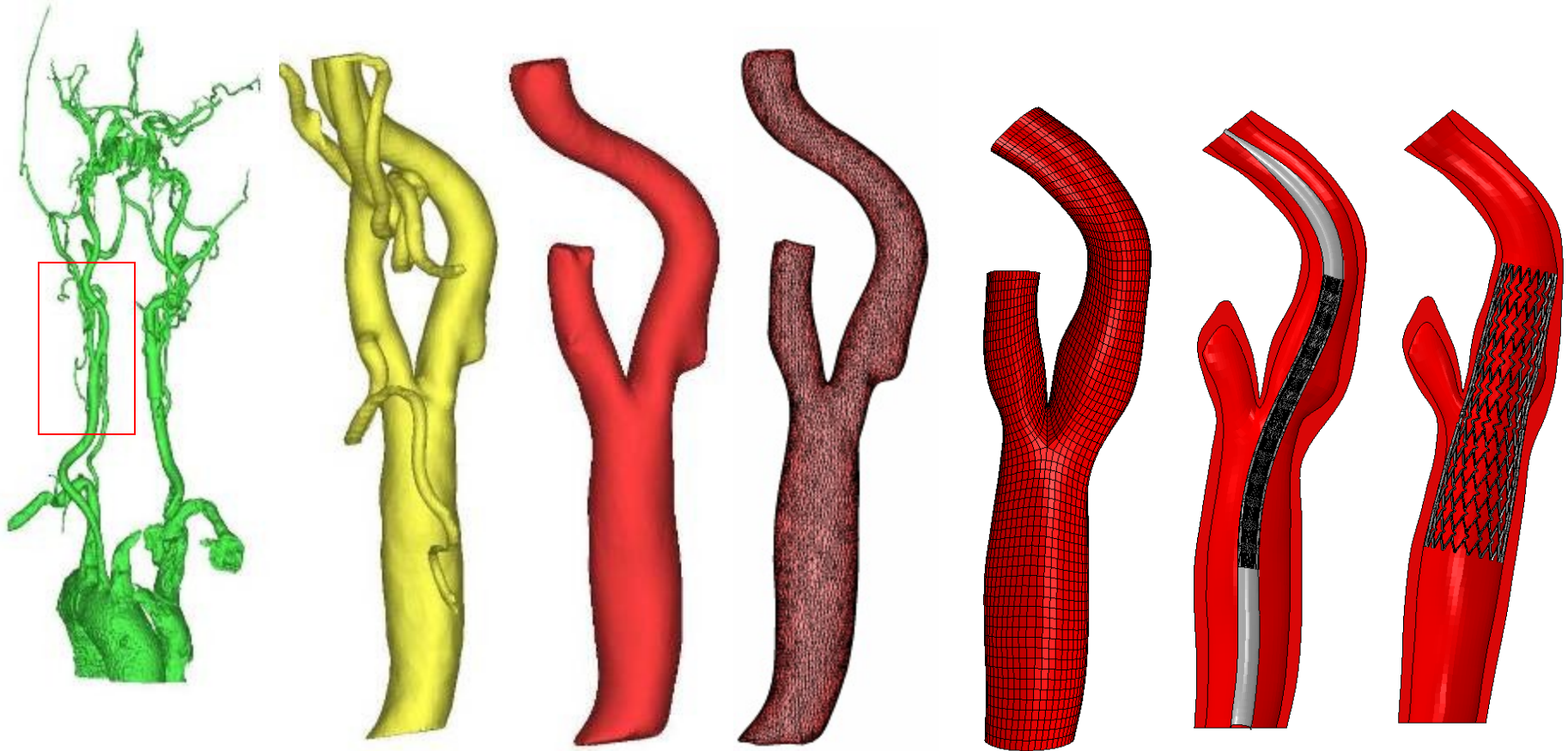


mapping after projection



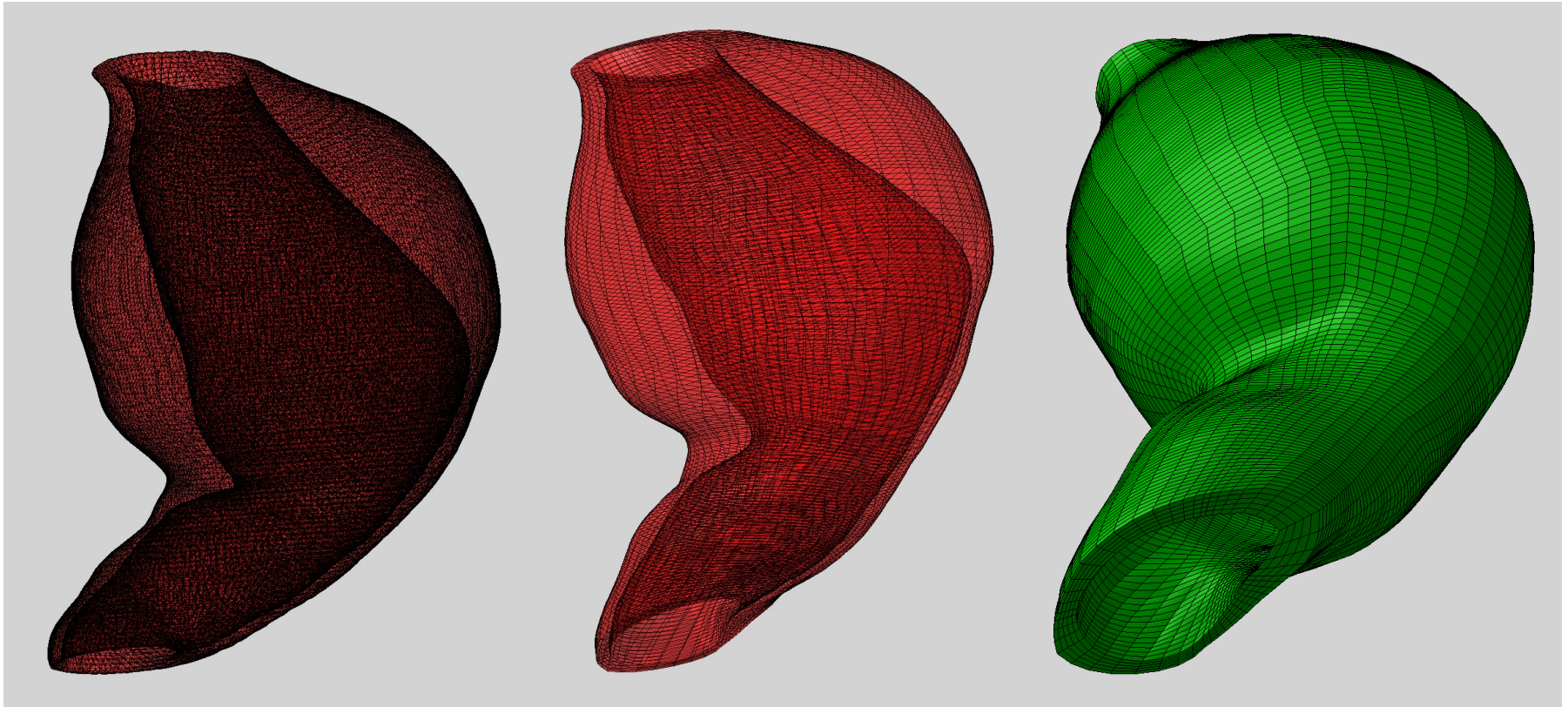


stent deployment FEA



mesh of a AAA thrombus

accurate computation of stress state due to arterial
pressure for rupture prediction



pyFormex Future

- Interactive Tools
- Surface and volume meshing
- Postprocessing
- Distribution and installation

pyFormex Future

Developers

Testers

Users

Welcome

<http://www.pyformex.org>