Shape Modeling and Geometry Processing

Exercise 1 - libigl "Hello World"

Michael Rabinovich-

michael.rabinovich@inf.ethz.ch

Acknowledgments: Olga Diamanti

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Exercise Sessions

- Friday from 09-10 at CAB, G 52
- Exercise discussion + Q&A
- Course website: http://crl.ethz.ch/teaching/shape-modeling-18/index.html

Rules of the Game (1)

- Total of 6 assignments
  - to be completed in 2 weeks' time
- Deliverables: Online Report + Code and data
  - Graded on all
  - Code must compile from git
- Submit until 9:00 on the day of the due date

Rules of the Game (2)

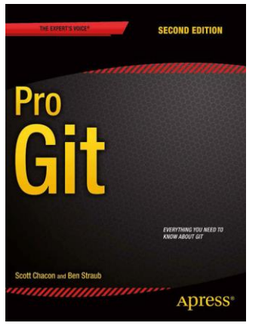
- Grading = Exam (20%) + Exercises (80%)
  - assignments are differently weighted
  - there will be bonus tasks (TBD)
- You can use STL but no non-standard dependencies
  - unless approved by us!

Questions? In this order, try:

- Exercise sessions
- Office hours:
  Wednesday, 13:00 - 14:00, CNB G108
- Email michael.rabinovich@inf.ethz.ch
- Appointment with me or Moritz

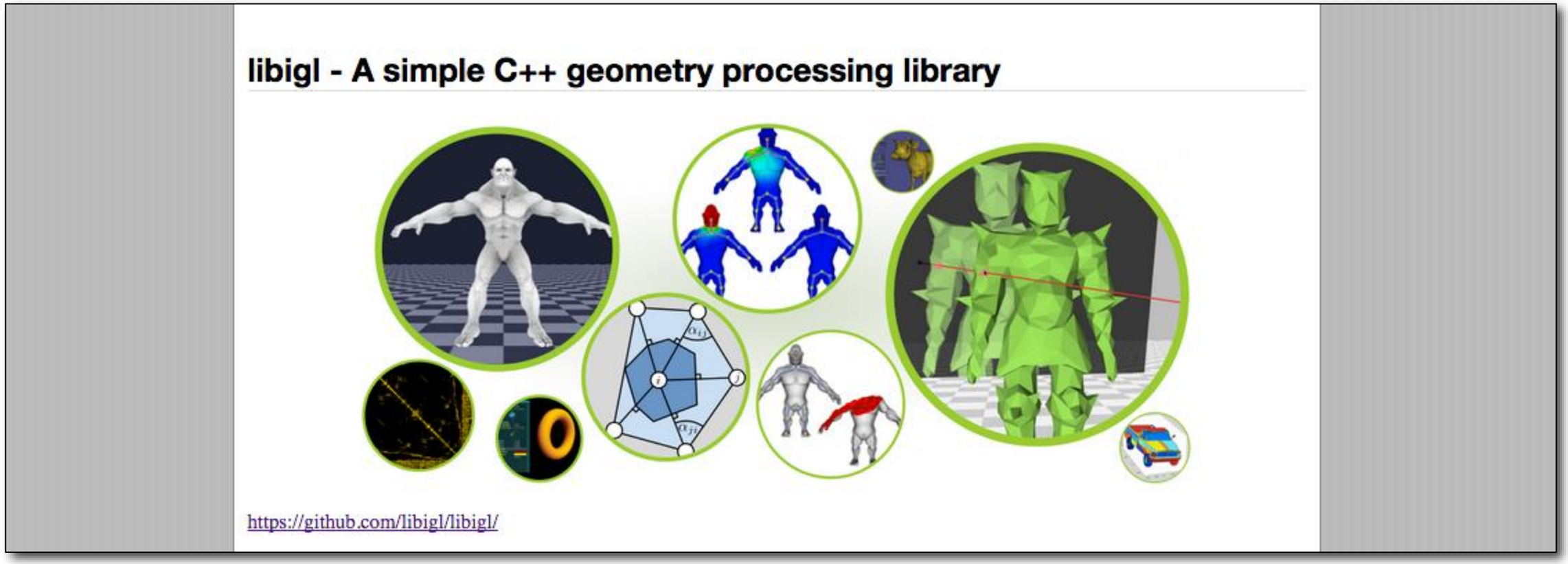
Assignments

- We will work with git
  (a distributed revision control system)
- More information: Pro Git book
- Our Repository:
  https://github.com/eth-igl/GP2018-Assignments



Exercise 1 - libigl “Hello World!”

- Experiment with the a geometry processing library

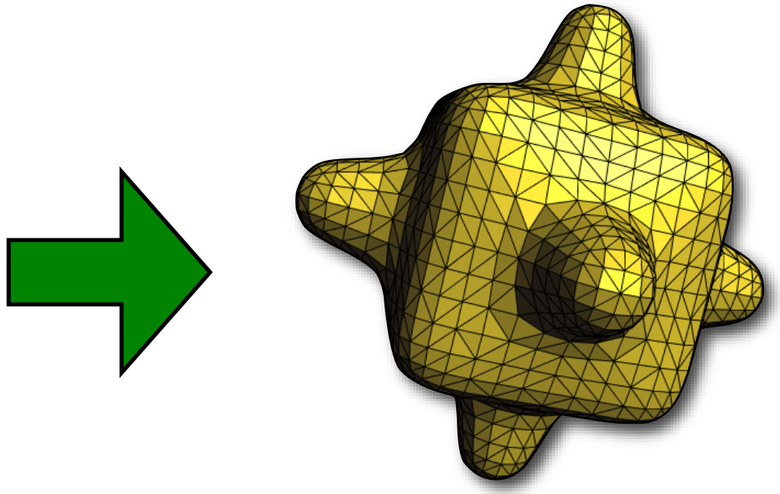


Exercise 1 - libigl “Hello World!”

- Read and visualize a mesh

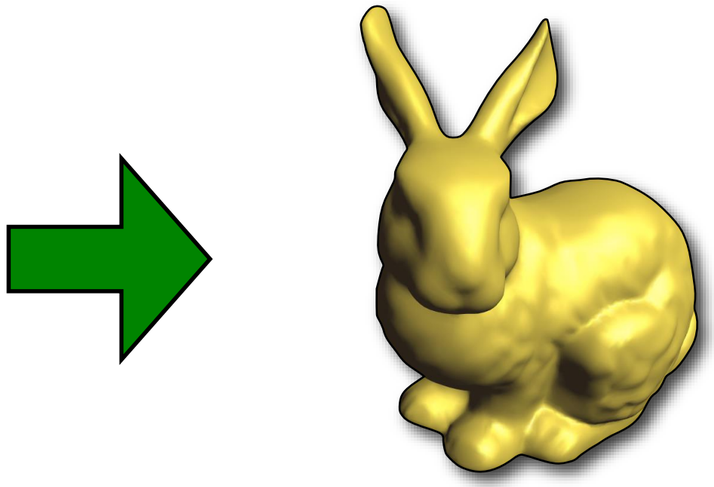
```
OFF
1250 2496 0
-2.09105 -2.09105 2.09105
-0.833333 -2.23958 2.23958
0.833333 -2.23958 2.23958
2.09105 -2.09105 2.09105
...
3 940 83 320
3 386 0 941
...
```

bumpy_cube.off



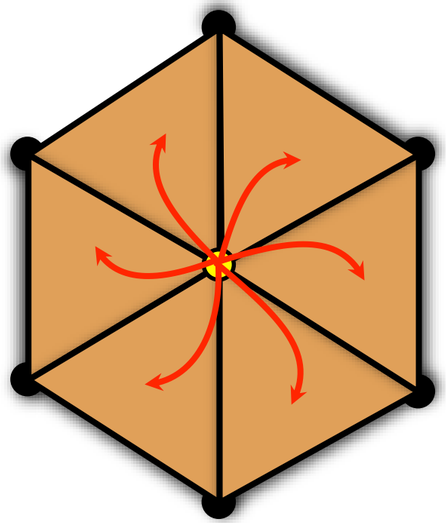
```
# Wavefront OBJ file
v 30.50959969 12.17459898 -15.84426970
v 30.49857998 11.87718728 -15.40759913
v 30.53679943 12.68500615 -14.82485356
v 30.67168999 11.71161003 -15.78844530
...
f  633/16706 11590/29979 4339/16704
f  11590/3161 633/16716 19901/16699
...
```

bunny.obj

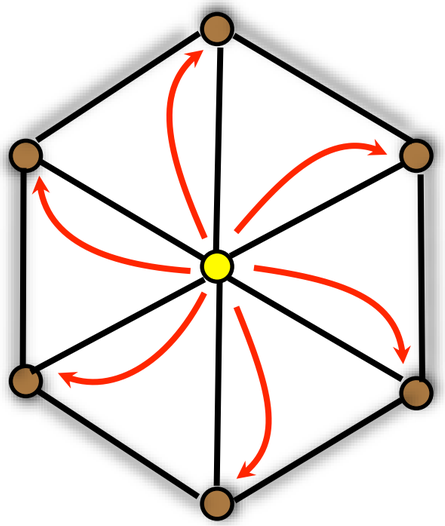


Exercise 1 - libigl “Hello World!”

- Perform simple neighborhood calculations



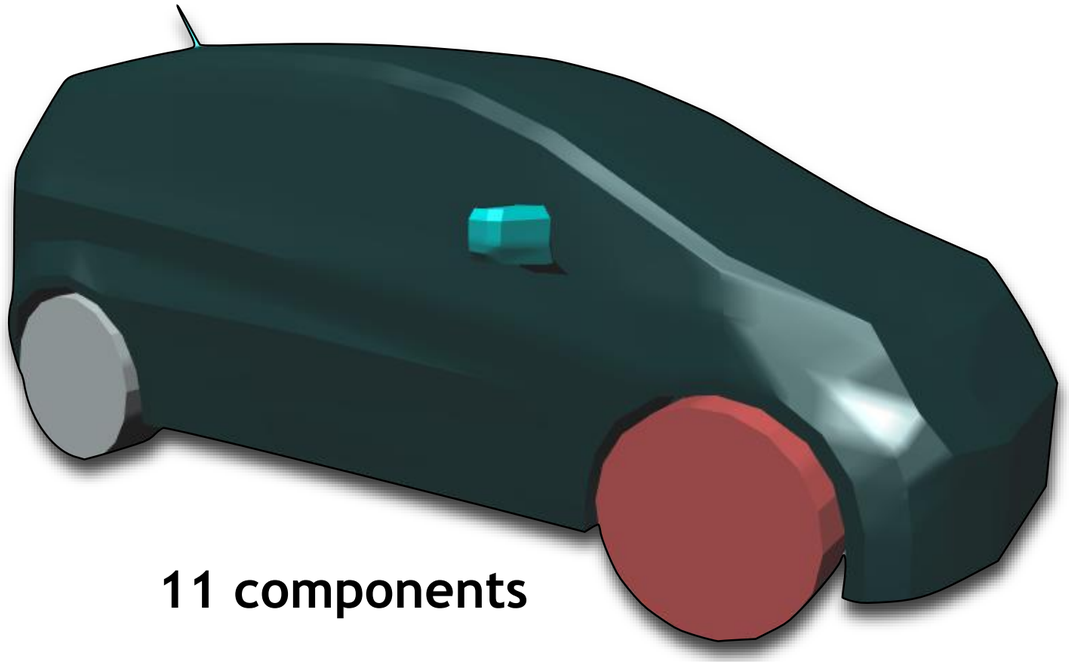
vertex-to-face



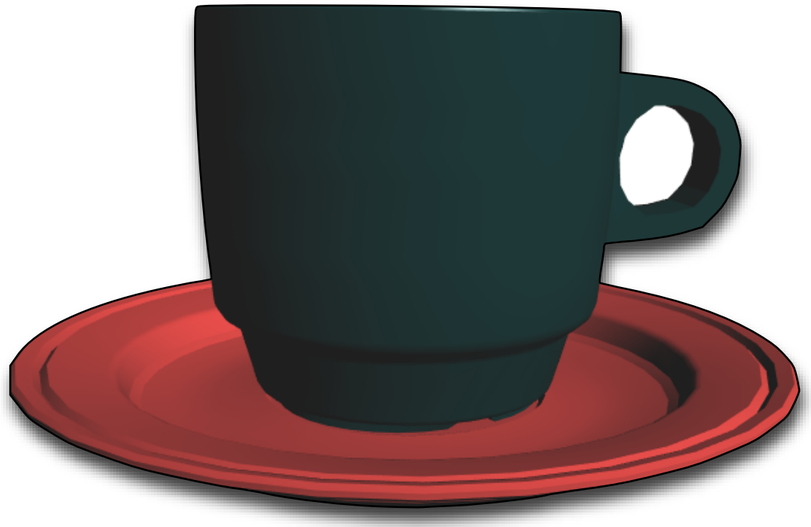
vertex-to-vertex

Exercise 1 - libigl “Hello World!”

- Connected Components



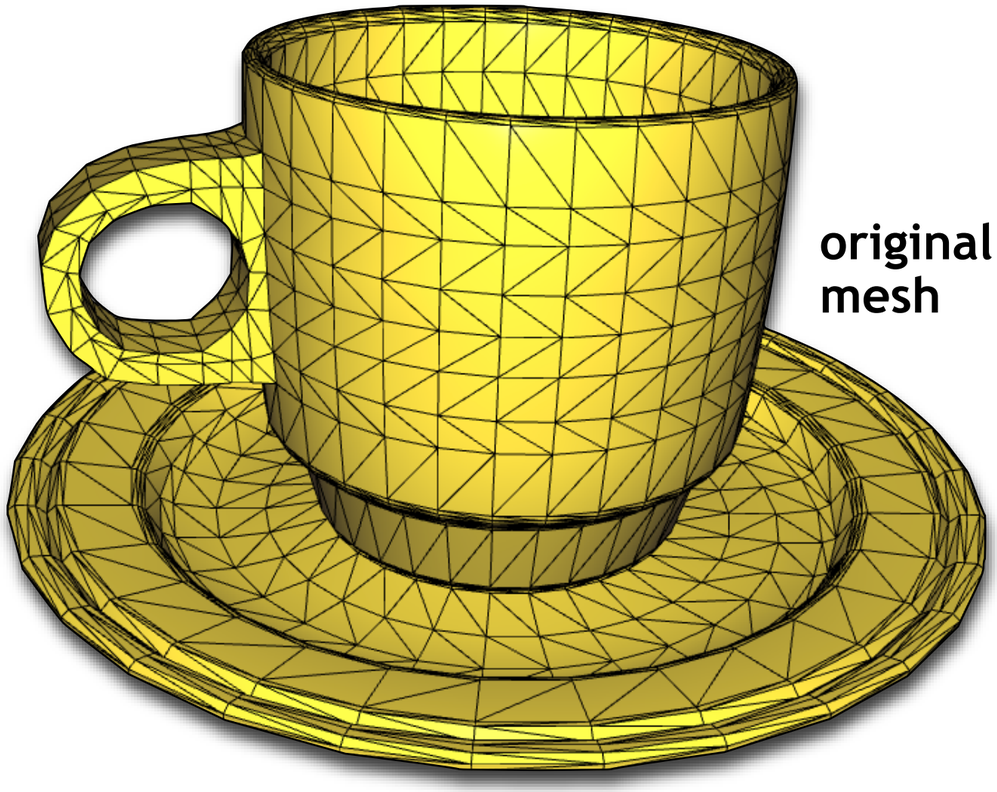
11 components



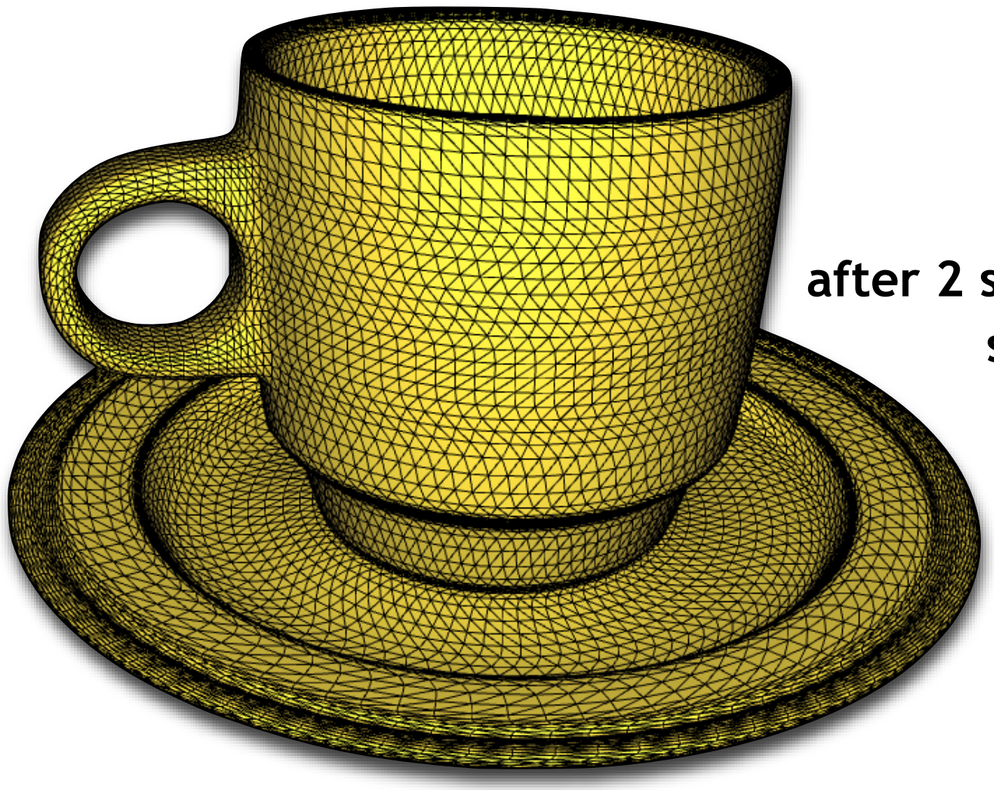
2 components

Exercise 1 - libigl “Hello World!”

- Sqrt(3) Subdivision



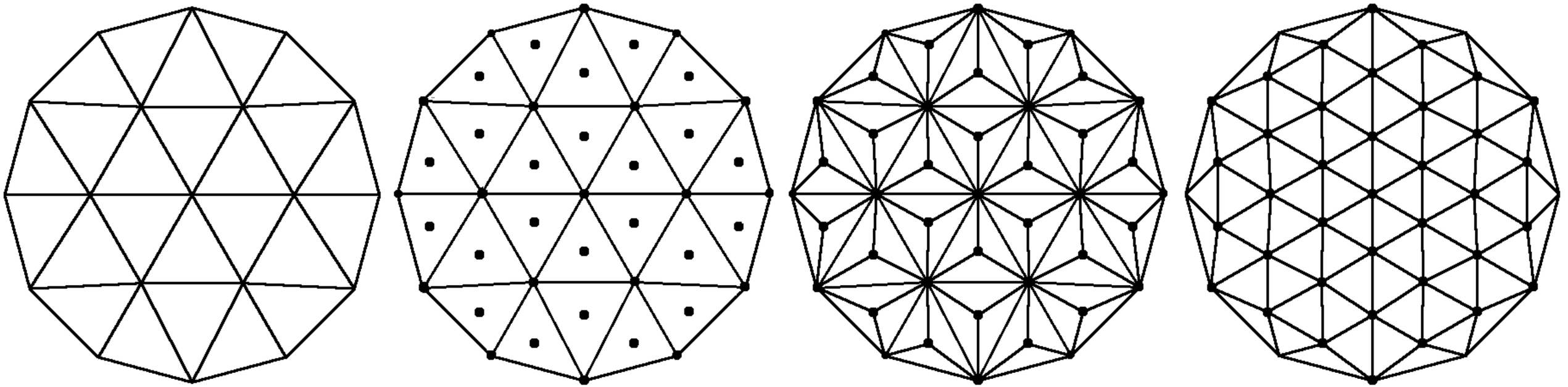
original mesh



after 2 subd. steps

Exercise 1 - libigl “Hello World!”

- Sqrt(3) Subdivision



original mesh

add vertices at face midpoints

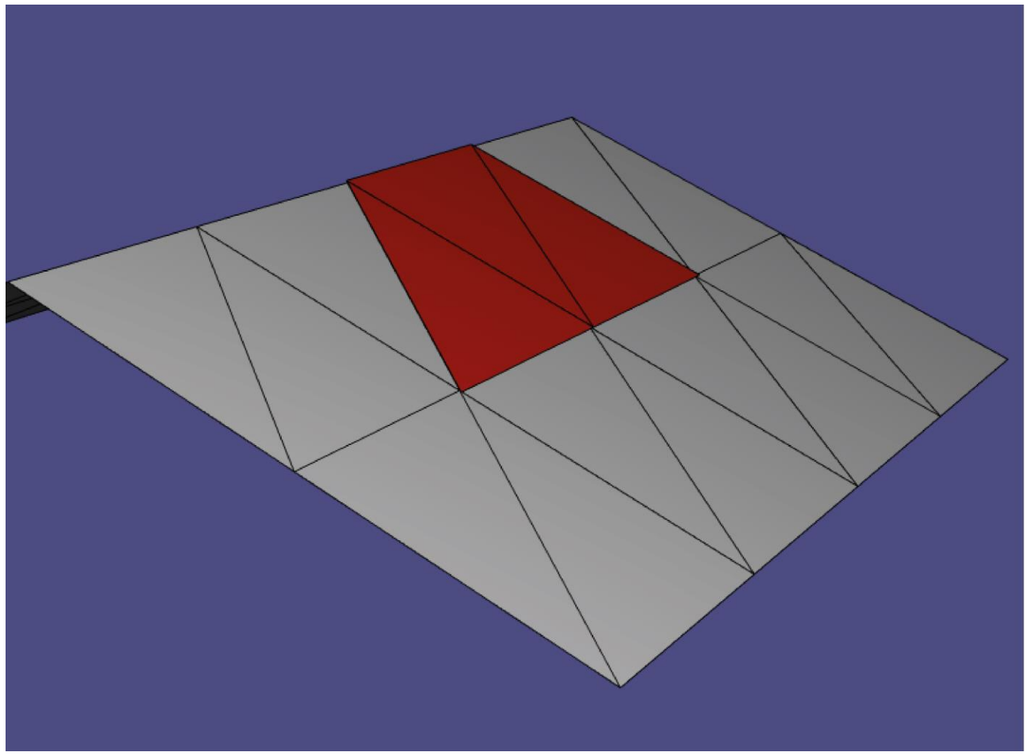
connect new vertices to face corners

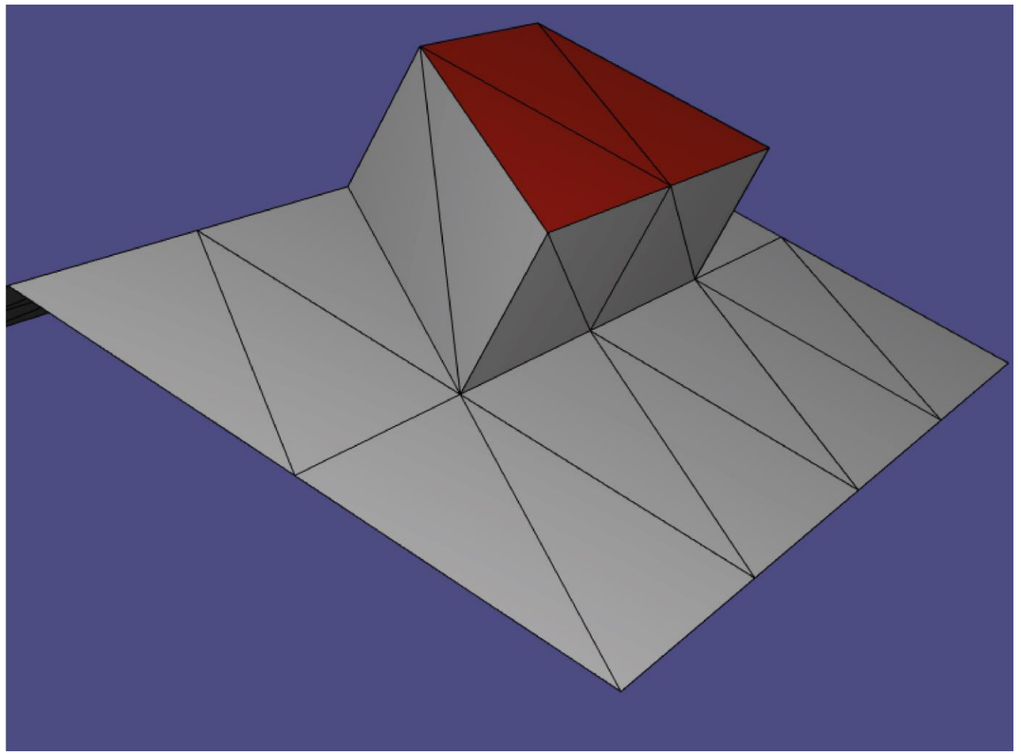
flip original edges

move old vertices by averaging in their one-ring

Exercise 1 - libigl “Hello World!”

- Triangular face extrusion





Eigen

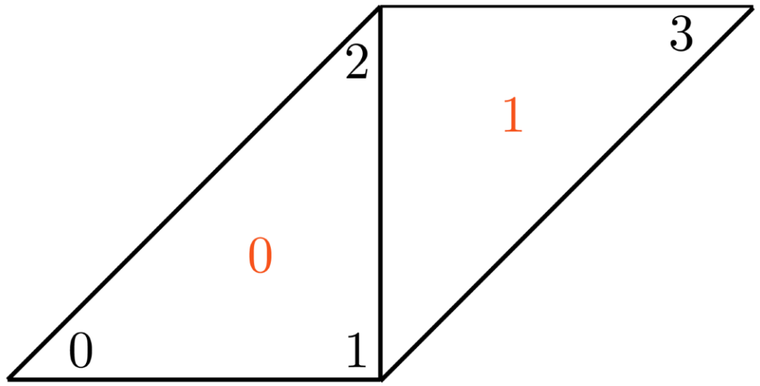
- *Eigen is a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms.*
  - *from* http://eigen.tuxfamily.org/index.php?title=Main_Page
- Header-only library
  - No compilation required!
- Tutorials:
  - http://eigen.tuxfamily.org/dox/TutorialMatrixClass.html
  - http://eigen.tuxfamily.org/dox/QuickRefPage.html

Mesh Representation with Eigen

- An Eigen matrix

```
Eigen::Matrix< type, #rows, #cols>
```

$$V = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 1 & 0 \end{pmatrix} \qquad F = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 3 & 2 \end{pmatrix}$$




```
Eigen::Matrix<double, Eigen::Dynamic, 3> V;
Eigen::Matrix<int,    Eigen::Dynamic, 3> F;
```

Have a look at: http://eigen.tuxfamily.org/dox/GettingStarted.html

Libigl

- Open source C++ library for geometry processing
  - Minimal dependencies
  - Header-only
  - No complex data types

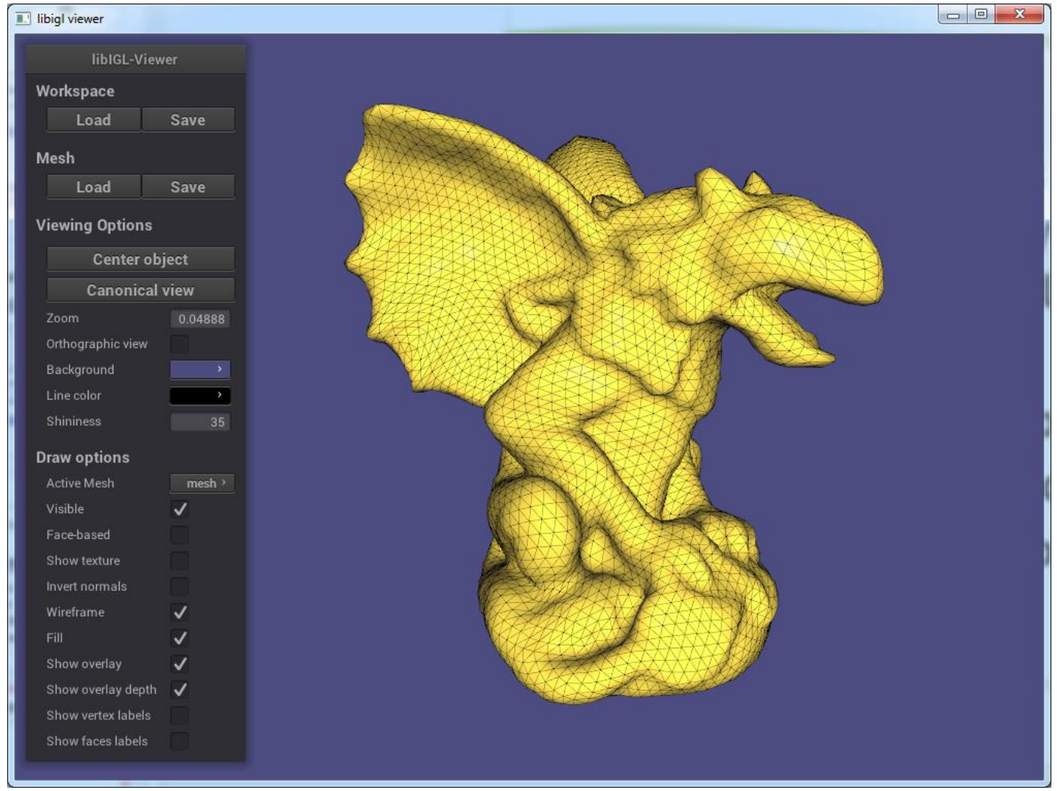
```
Eigen::MatrixXd V;
Eigen::MatrixXi F;
```

```
igl::readOFF("../shared/cube.off",V, F);
```

- Dependencies: Eigen, GLFW, nanogui, ...
  - All included as submodules in libigl
- Tutorials that explain the main functionality
  - Compilation instructions: see assignment sheet

Libigl - The Viewer

- Display mesh
- Very basic UI options
  - Rotate (left click) /translate (right click) /Zoom (scroll/‘a’-’s’)
  - Texture/Normals
  - Some material/color options
  - Show vertex/face id
- Other options available in source code



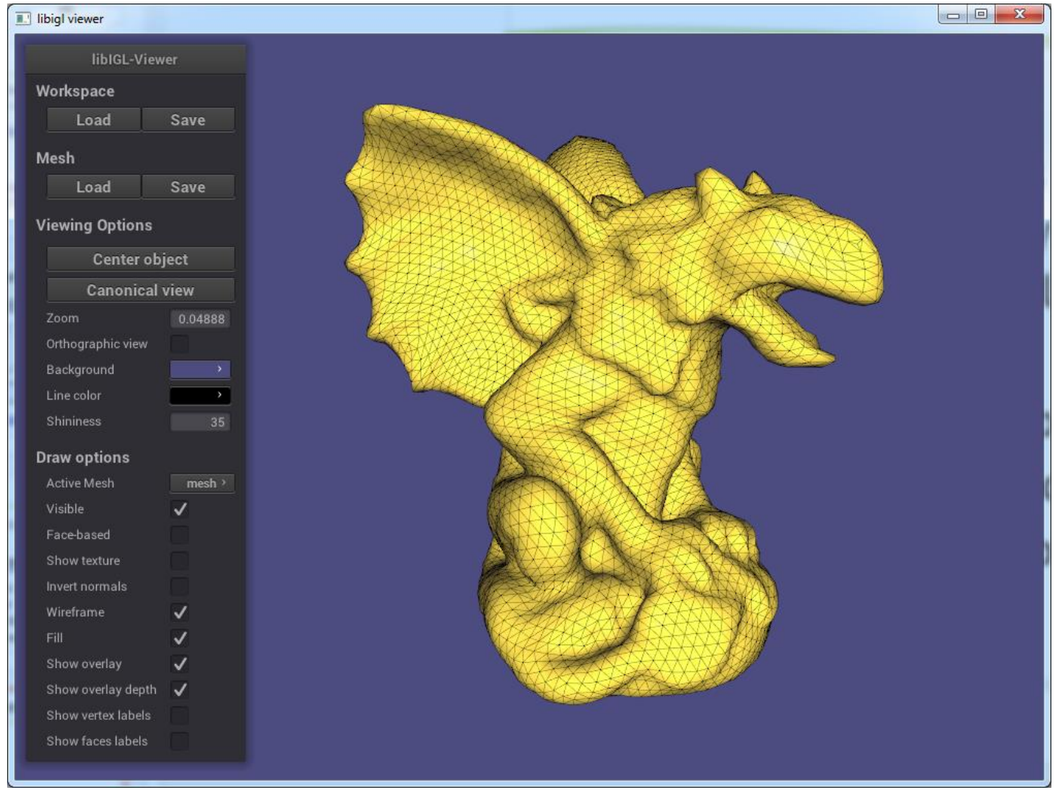
Basic program with libigl and viewer

```
#include <igl/readOFF.h>
#include <igl/viewer/Viewer.h>

Eigen::MatrixXd V;
Eigen::MatrixXi F;

int main(int argc, char *argv[])
{
  // Load a mesh in OFF format
  igl::readOFF("gargo.off", V, F);

  // Plot the mesh
  igl::Viewer viewer;
  viewer.data.set_mesh(V, F);
  viewer.launch();
}
```



Adding functionality to the viewer

- Custom callbacks for keyboard/mouse interactions supported
  - See tutorial 103_Events
- Also supported: Face/Vertex Colors, Overlays (points/lines)
  - See tutorials 104_Colors, 105_Overlays
- Read [LIBIGL_DIRECTORY]/tutorial/tutorial.html

CMake Project for exercise 1

- Compiles a single main.cpp that runs the viewer
  - Assignment tasks are to be implemented as key interactions
  - ‘1’ - ‘2’ : neighborhood relations
  - ‘3’: connected components
  - ‘4’: subdivision
  - Extrude button: Triangular face extrusion
- Compile

```
cd REPO_DIR/assignment1;
mkdir build; cd build
cmake -DCMAKE_BUILD_TYPE=Release ../; make
```

- Run

```
./assignment1_bin.exe <some mesh file>
```

Questions?

- Exercise sessions
- Office hours: ***Wednesday, 13:00 - 14:00, CNB G108***
- Email: michael.rabinovich@inf.ethz.ch
- Appointment with me or Moritz

- Bug reports/suggestions also welcome!