DGtal tutorial

DGMM 2022

D. Coeurjolly

Outline and objectives

- DGtal overview and features
- *Coffee break* ☕
- Getting started
    - Installation
    - Basic examples
- 5 practical works (Jacques-Olivier Lachaud, Tristan Roussillon, Bertrand Kerautret, D.C.)
- Open discussions

Set up your tutorial material

- **Requirements:**
  - Git client
  - C++ compiler (at least C++11 enabled)
  - Cmake (cmake.org)
  - Boost headers (boost.org)
  - Zlib (should be included by defaults in your OS)

  - for polyscope visualization, you may need some X/openGL headers. E.g. on ubuntu

    sudo apt-get install xorg-dev libglu1-mesa-dev freeglut3-dev mesa-common-dev

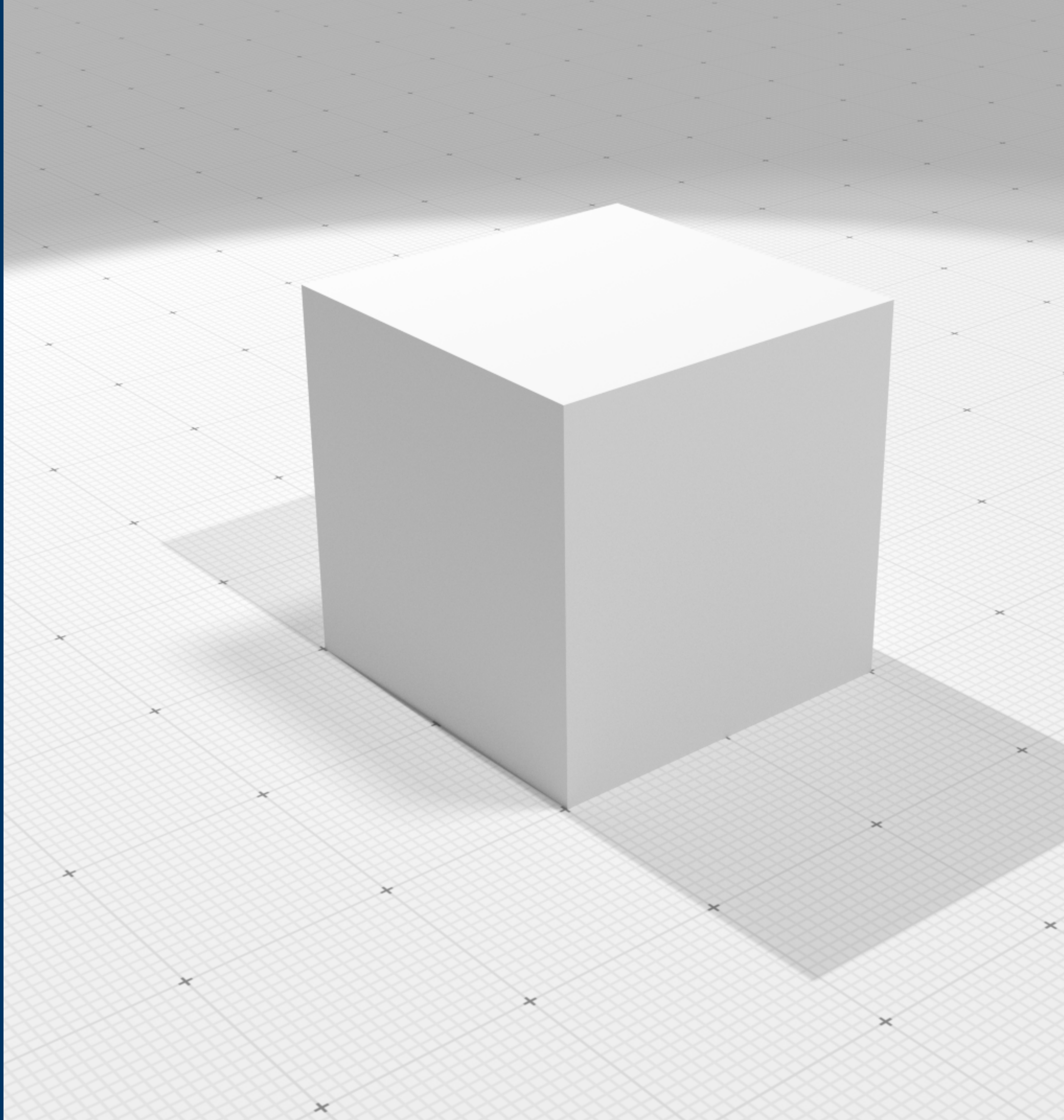
Discord server: https://discord.gg/PFBwRHMN

To get help during the practicals, post-tutorial discussions, to share your results / failures cases ...

Material: https://github.com/DGtal-team/DGtal-Tutorials-DGMM2022

Code + practicals

Context

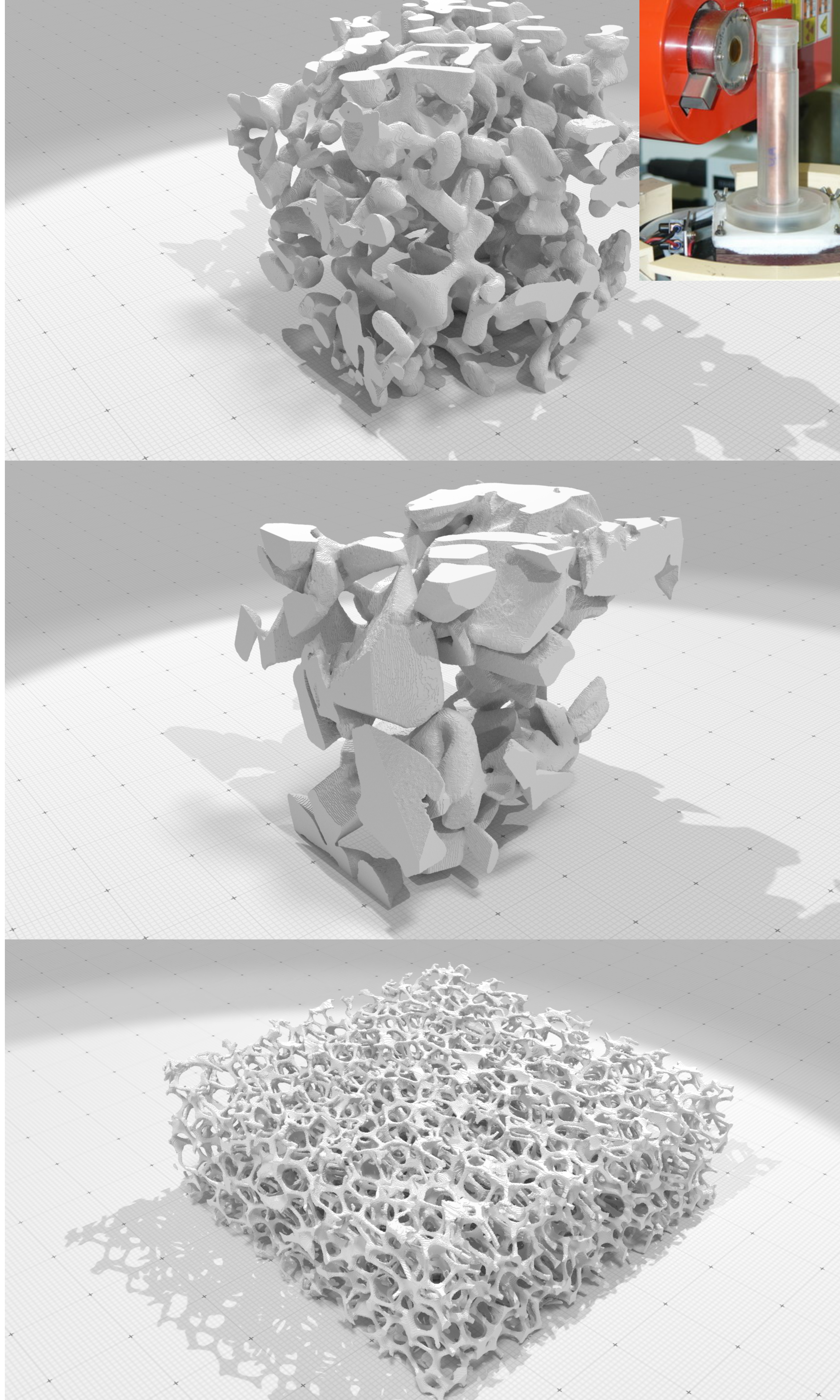


Use-cases (1): geometry processing

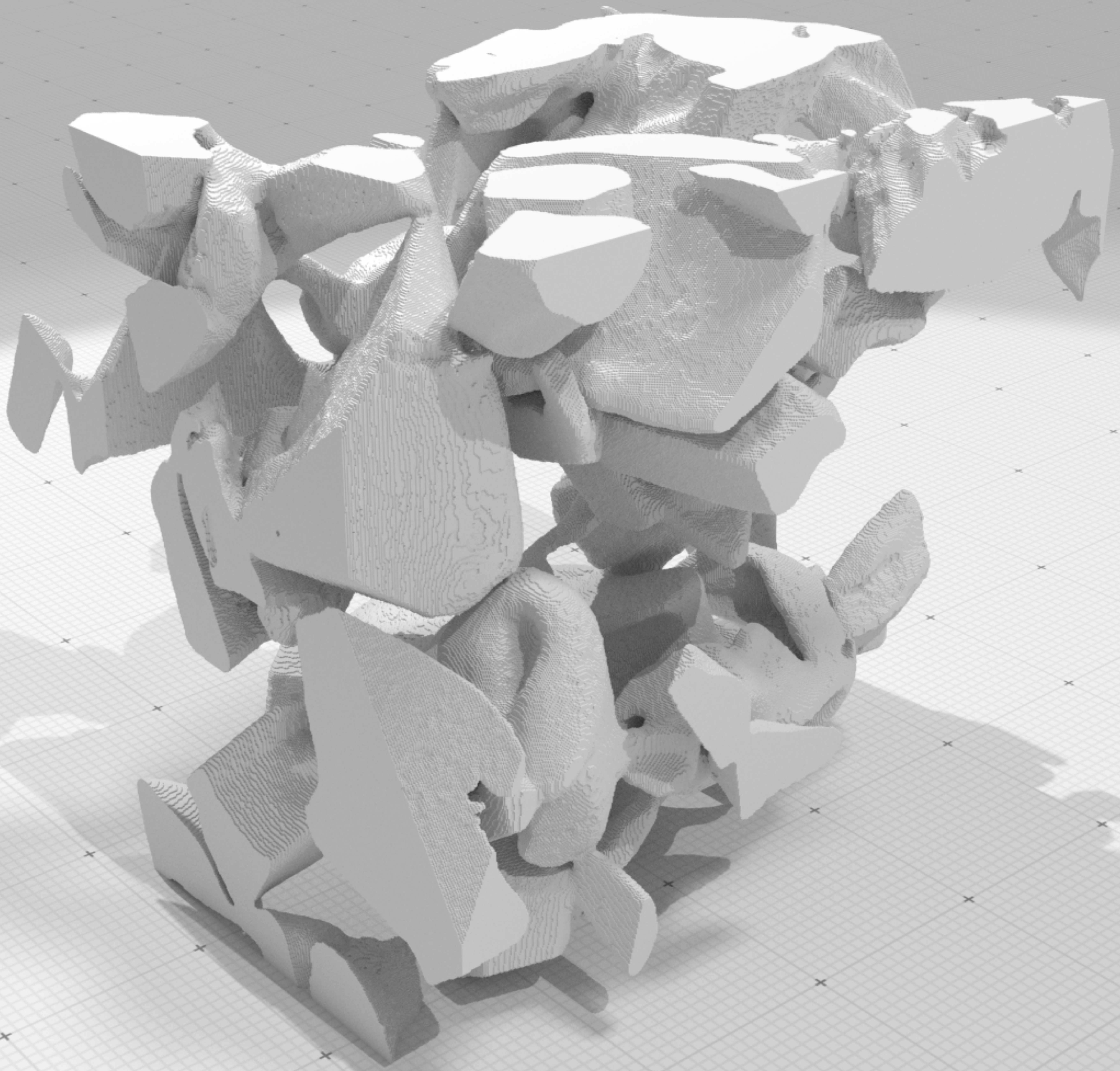
- Micro-tomographic images
  - material sciences
  - medical images

- Process geometry/topology of images partitions

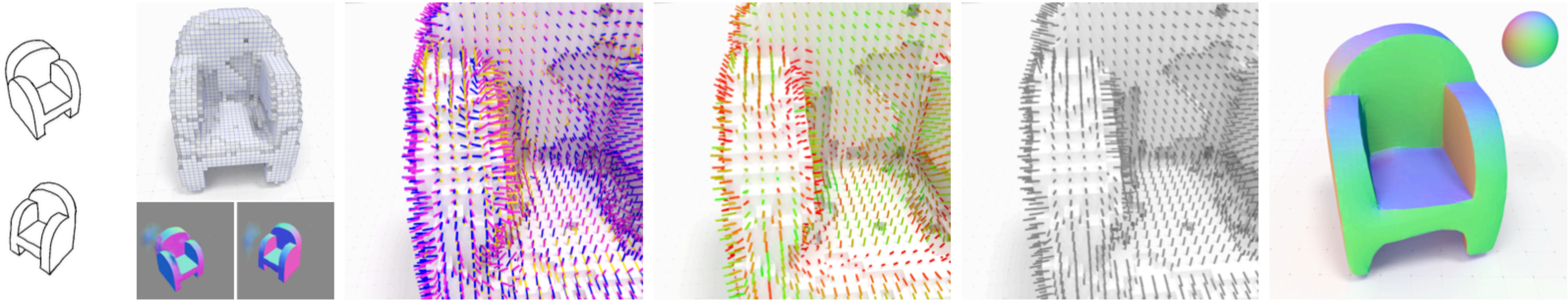
$\Rightarrow X \subset \mathbb{Z}^3$



Example 1



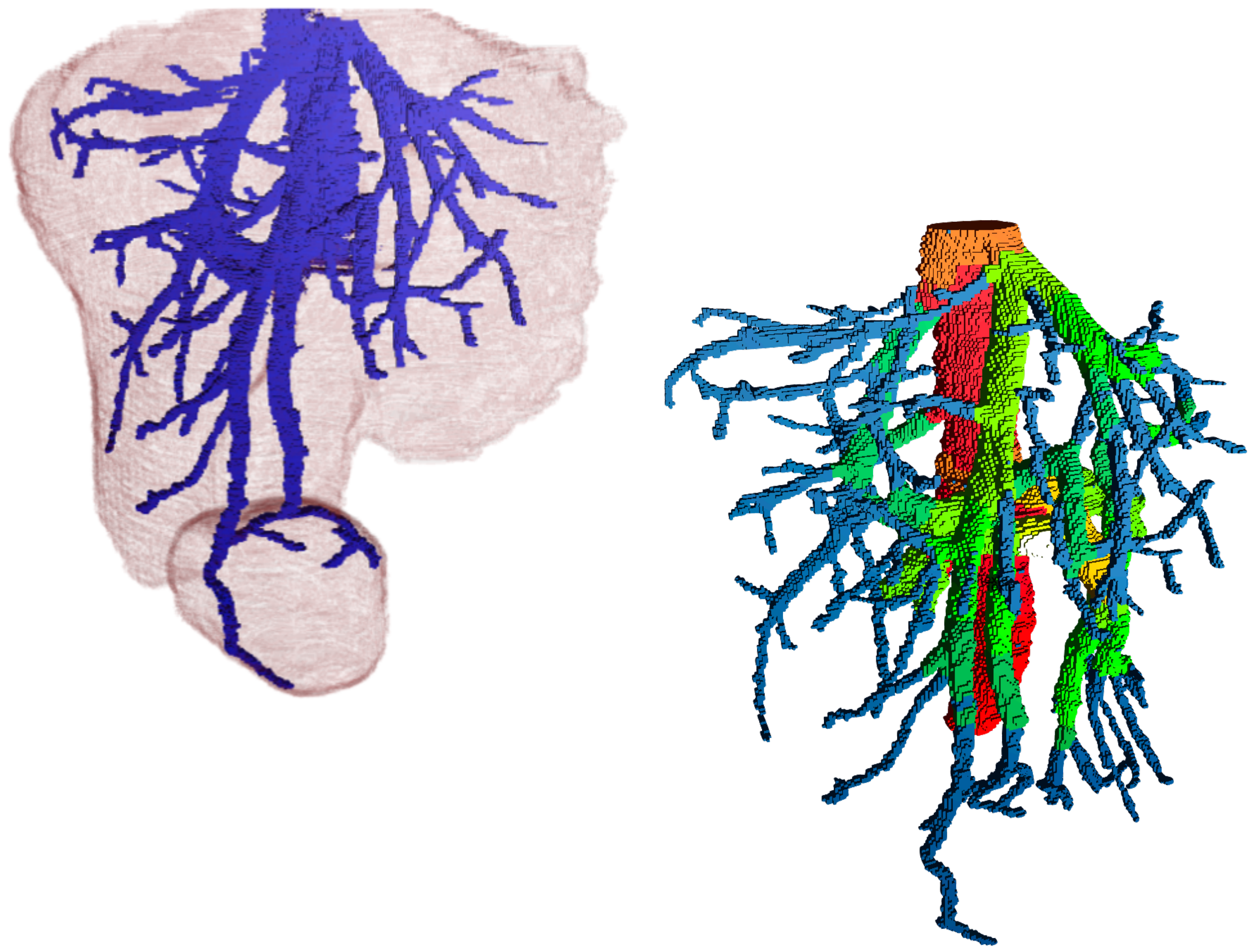
Example 2

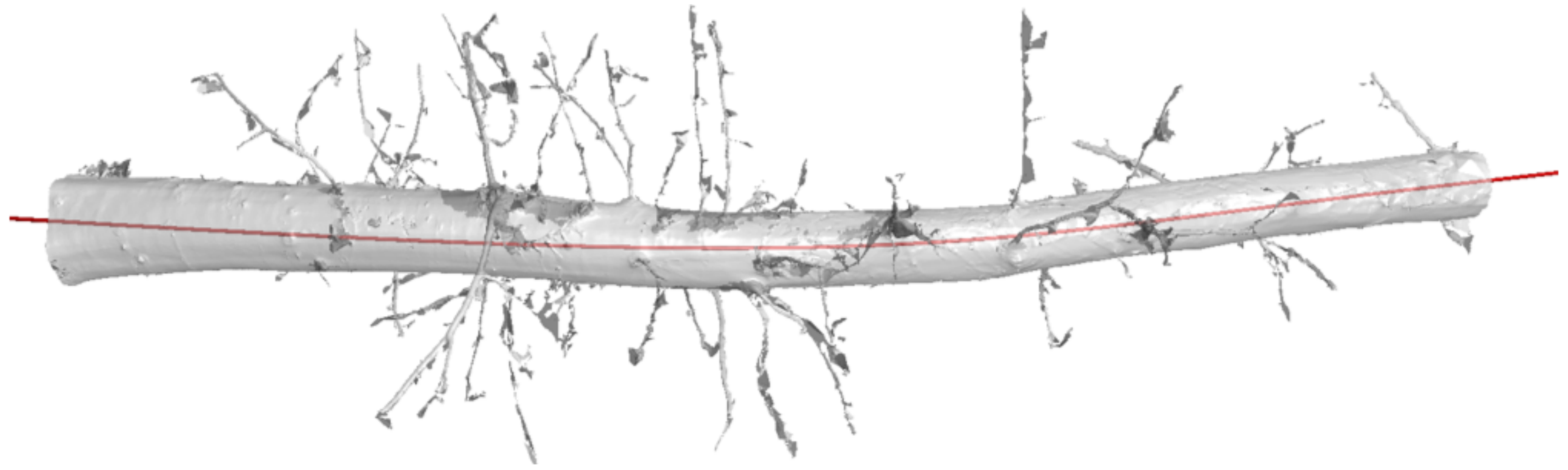


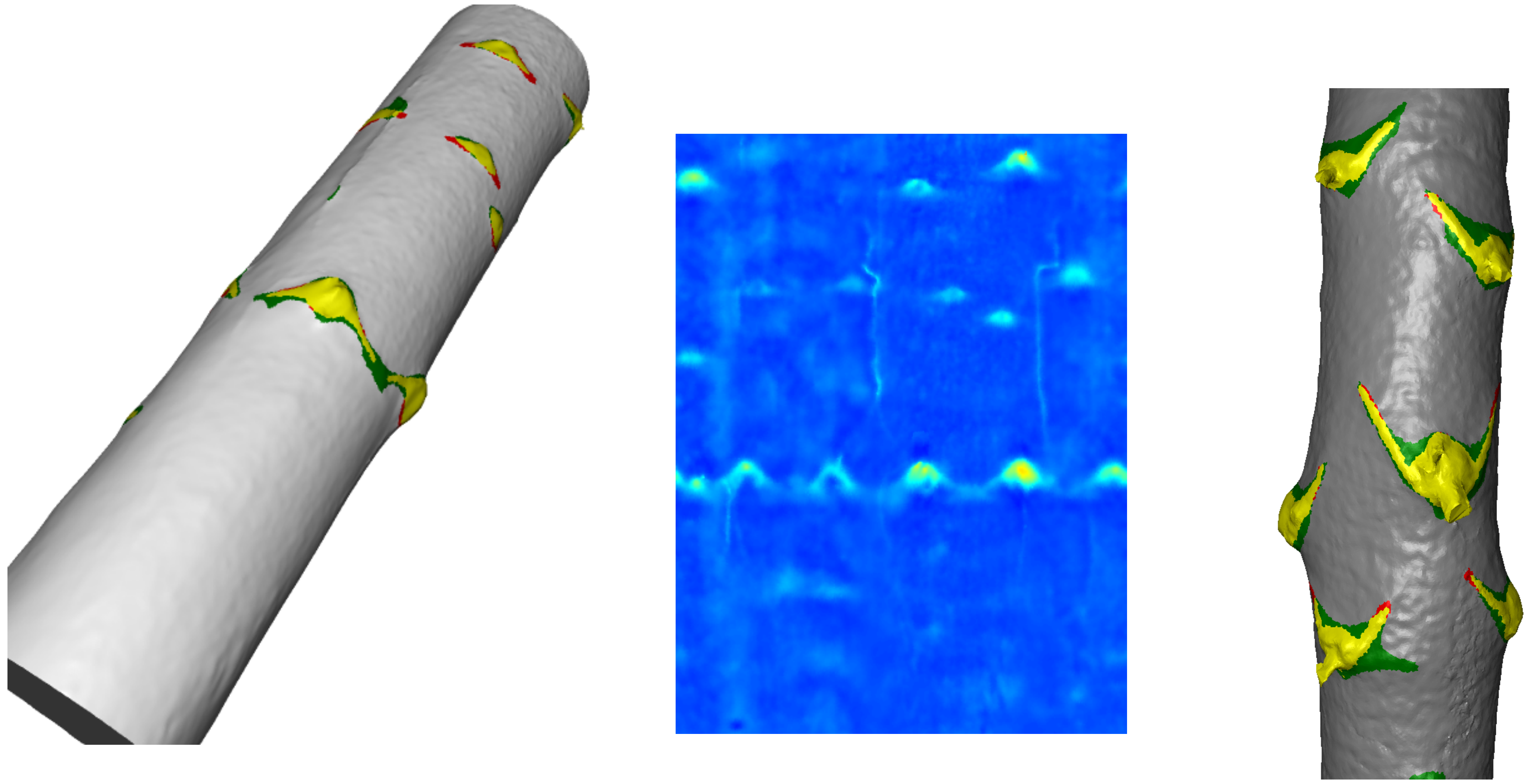
(a) Input (b) CNNs predictions (c) Candidate normals (d) Aggregated normals (e) Piecewise-smooth normals (f) Final surface

[Delanoy et al 19]

Examples 3: image processing





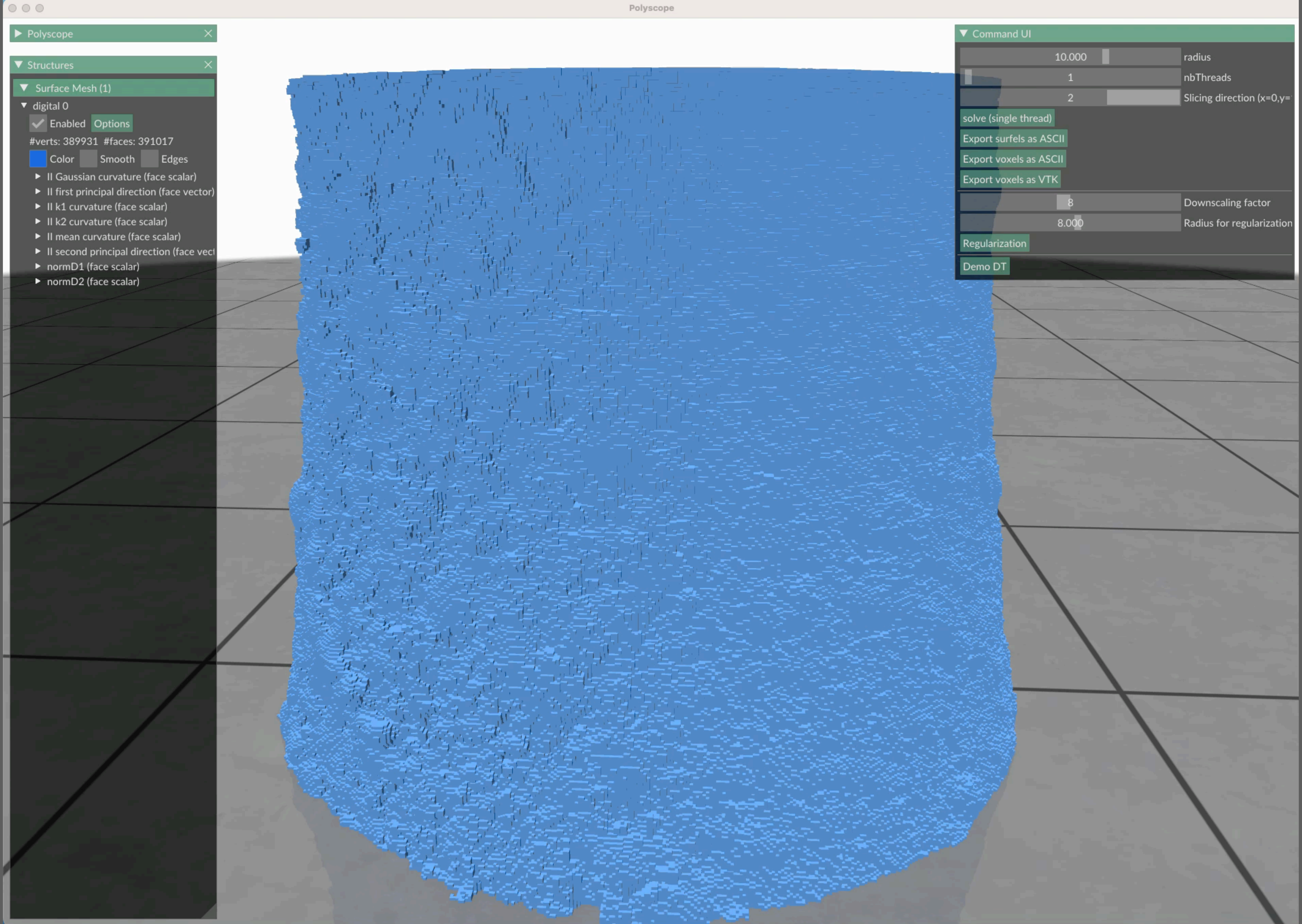


<table>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>Input image g</td><td>Approx. u with AT20 and AT01</td><td>u and v with AT20</td><td>u and v with AT01</td></tr>
<tr><td>Command line are:</td><td colspan="3"><code>./imageProcessing/at-u2-v0 -i ../imageProcessing/Images/barbara-cropped-b01.pgm -o barbara -a 1.0 --lambda 0.0054 --epsilon-1 2.0 --epsilon-2 0.25</code>
<code>./imageProcessing/at-u0-v1 -i ../imageProcessing/Images/barbara-cropped-b01.pgm -o barbara -a 0.69 --lambda 0.0065 --epsilon-1 2.0 --epsilon-2 0.25</code></td></tr>
</table>

DGtal

Principles

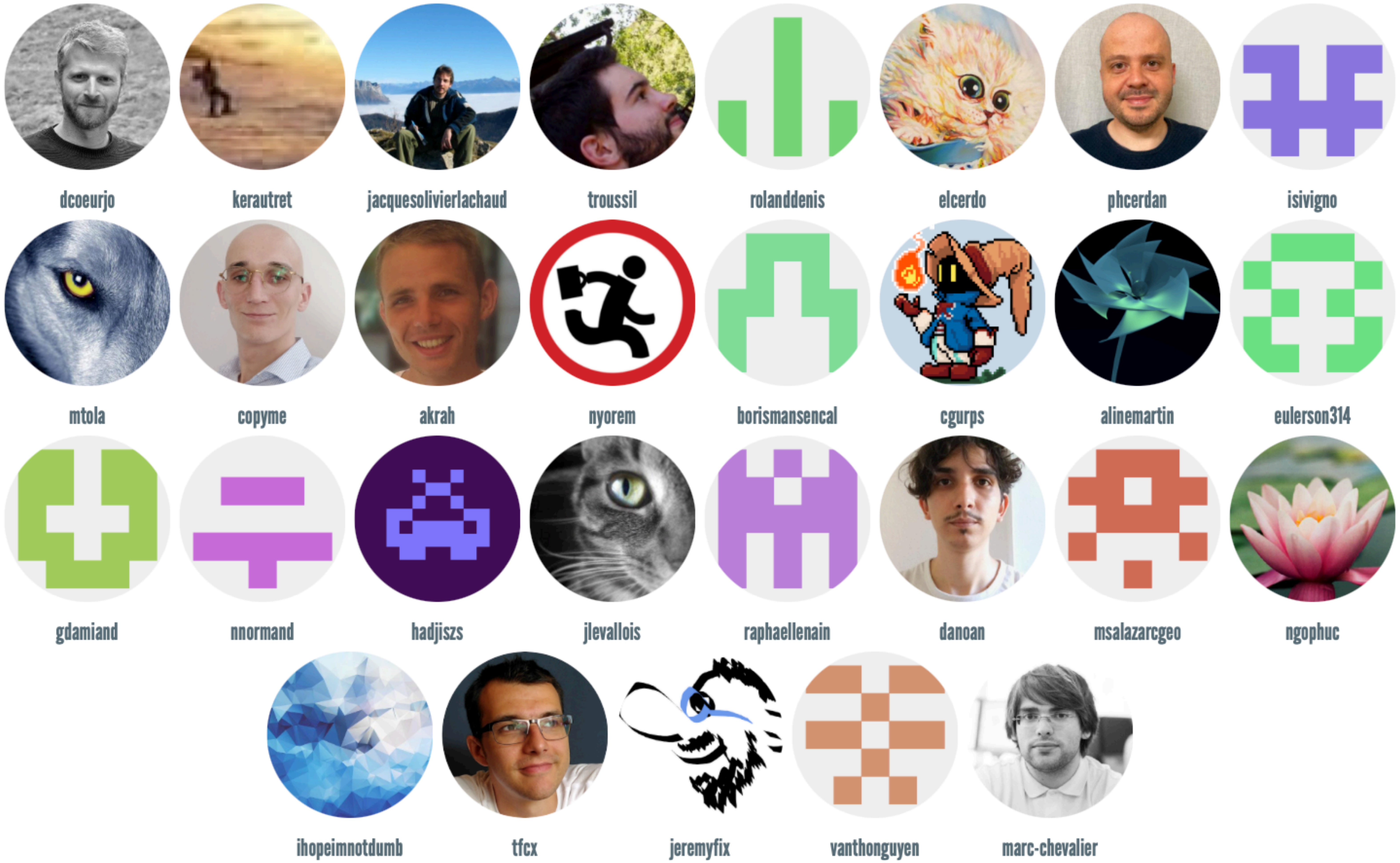
- Objectives: share state-of-the-art and cutting-edge algorithms from digital geometry community.
  - easy comparisons with the state-of-the-art
  - allows new-comers in the field to get started
  - fast prototyping of specific softwares (material sciences, medical imaging)
  - provides nice illustrations/outputs of data structures and algorithms


Polyscope
Polyscope
Structures
Surface Mesh (1)
digital 0
Enabled Options
#verts: 389931 #faces: 391017
Color Smooth Edges
II Gaussian curvature (face scalar)
II first principal direction (face vector)
II k1 curvature (face scalar)
II k2 curvature (face scalar)
II mean curvature (face scalar)
II second principal direction (face vect
normD1 (face scalar)
normD2 (face scalar)
Command UI
10.000 radius
1 nbThreads
2 Slicing direction (x=0,y=
solve (single thread)
Export surfels as ASCII
Export voxels as ASCII
Export voxels as VTK
8 Downscaling factor
8.000 Radius for regularization
Regularization
Demo DT

The project

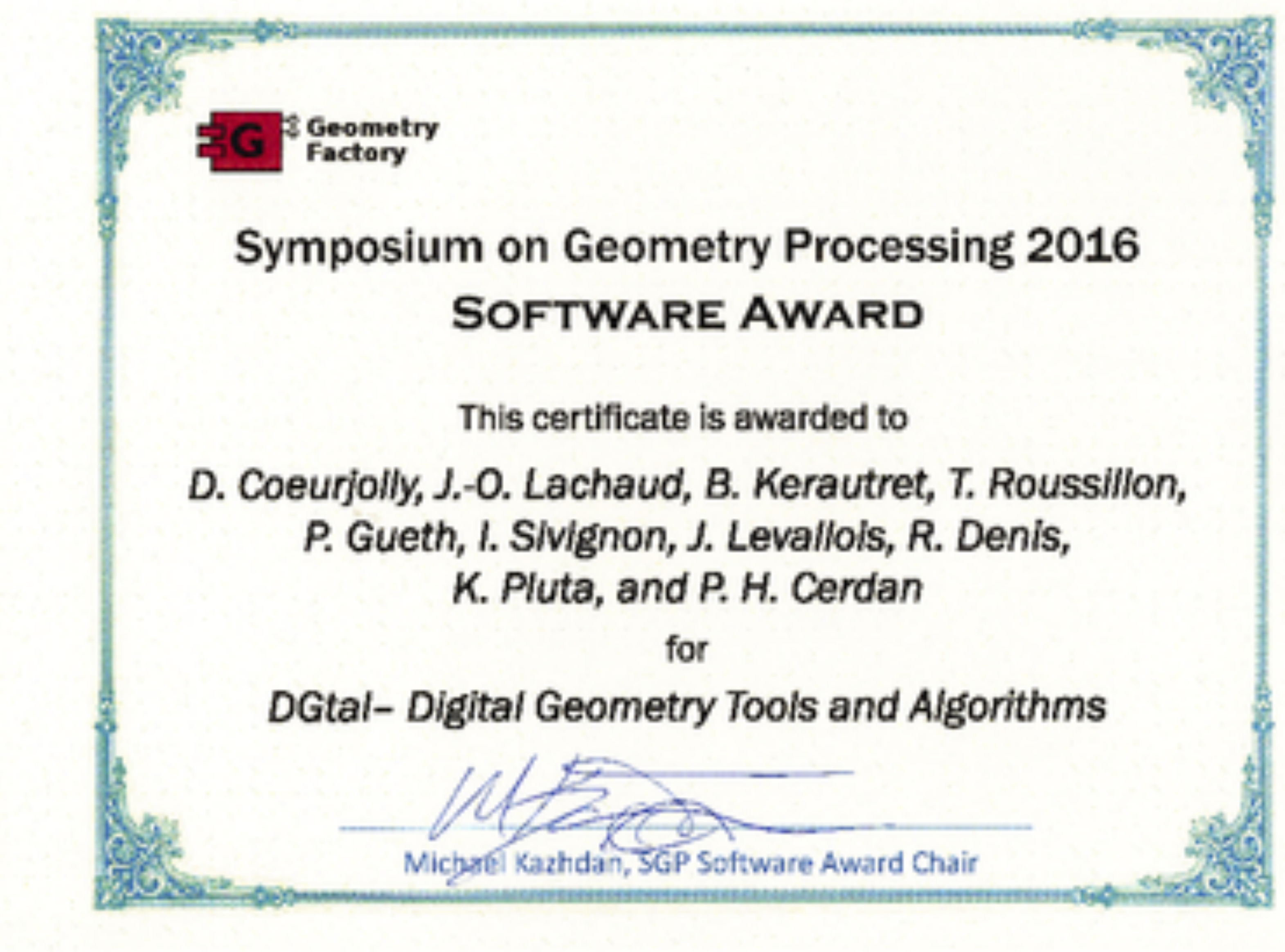
- C++ open-source library (**DGtal**)
- Preliminary **python** binding
- Collection of command-line tools (**DGtalTools**, **DGtalTools-contrib**) for the processing of images (2d, 3d), meshes or implicit shape
- Features
  - highly documented library
  - generic programming (data structure $\perp$ algorithms, nD kernel)
  - high performance tools (efficient containers, multithreading…)
  - quick visual feedback for interactive visual debugging
- *Support from:*

Community


dcoeurjo
kerautret
jacquesolivierlachaud
troussil
rolanddenis
elcerdo
phcerdan
isivigno
mtola
copyme
akrah
nyorem
borismansencal
cgurps
alinemartin
eulerson314
gdamiand
nnormand
hadjiszs
jlevallois
raphaellenain
danoan
msalazarcgeo
ngophuc
ihopeimnotdumb
tfcx
jeremyfix
vanthonguyen
marc-chevalier

History / Stats

- First git commit: Feb 28, 2010
- 12 425 commits
- ~30 github contributors
- 316 github stars
- 99 documentations HTML pages
- 3392 doxygen generated technical documentation pages
- many students involved (master, PhD...)
- countless number of related research papers

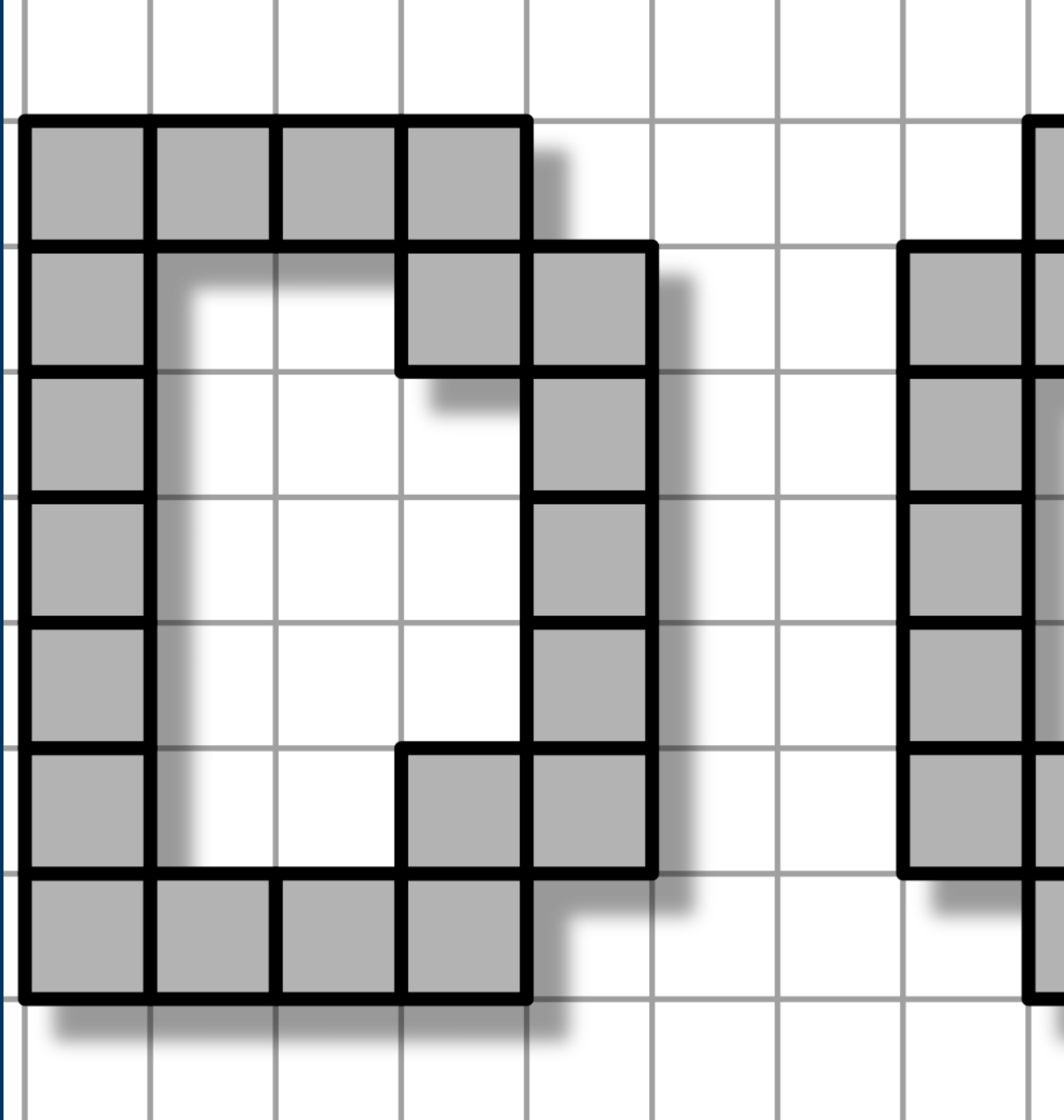


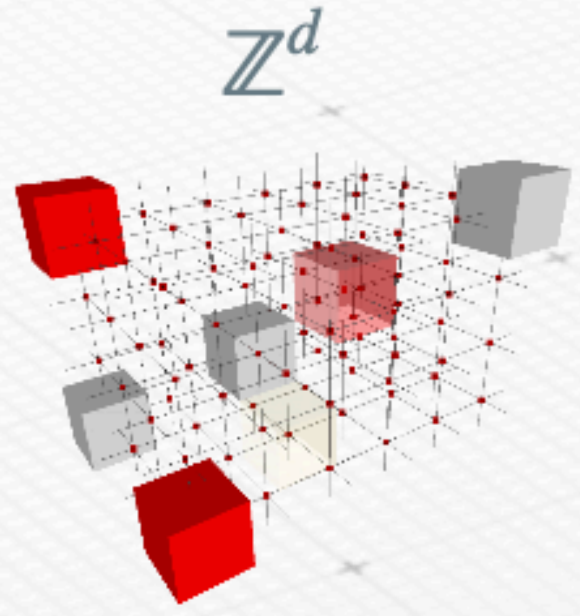
Current release: 1.3

- New features
  - Digital convexity and full convexity [DGMM2021] and [DGMM2022]
  - Differential calculus on polygonal surfaces and digital surfaces [DGMM2022]
  - Geodesics / vector field processing [DGMM2022]
  - Complete Voronoi map computation

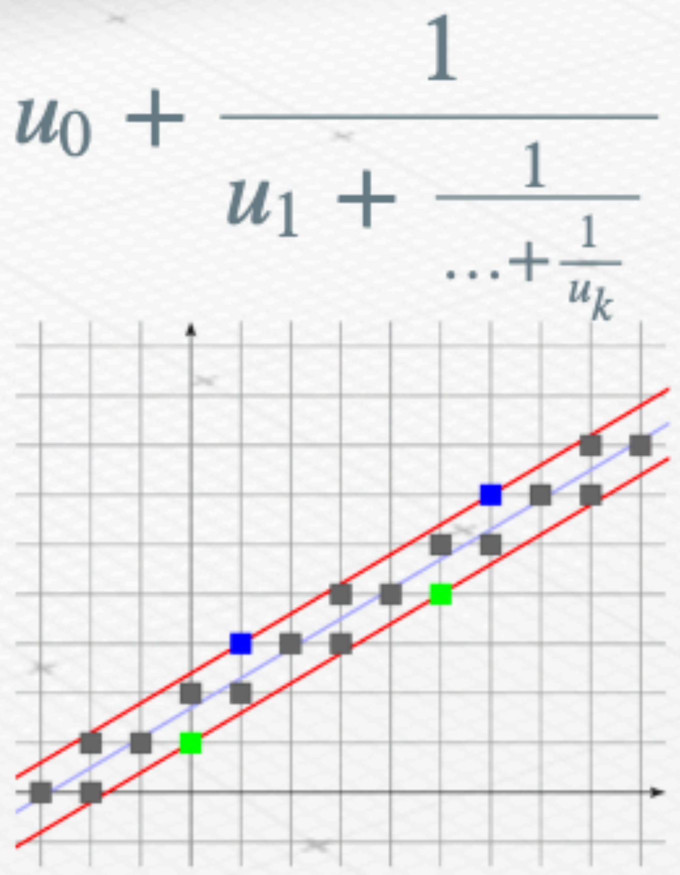
- Many bug fixes, improvements, documentation updates…

Package overview

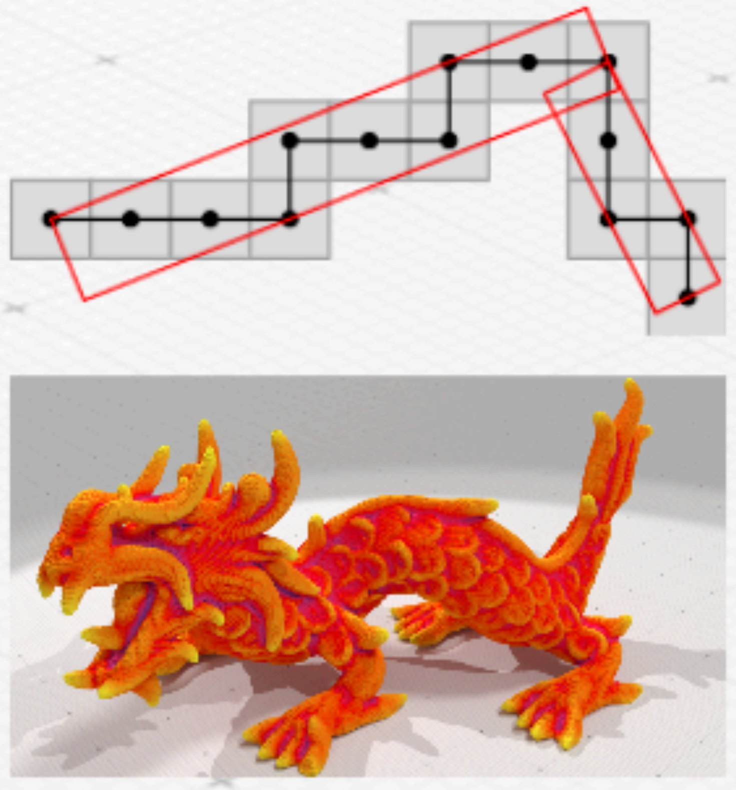




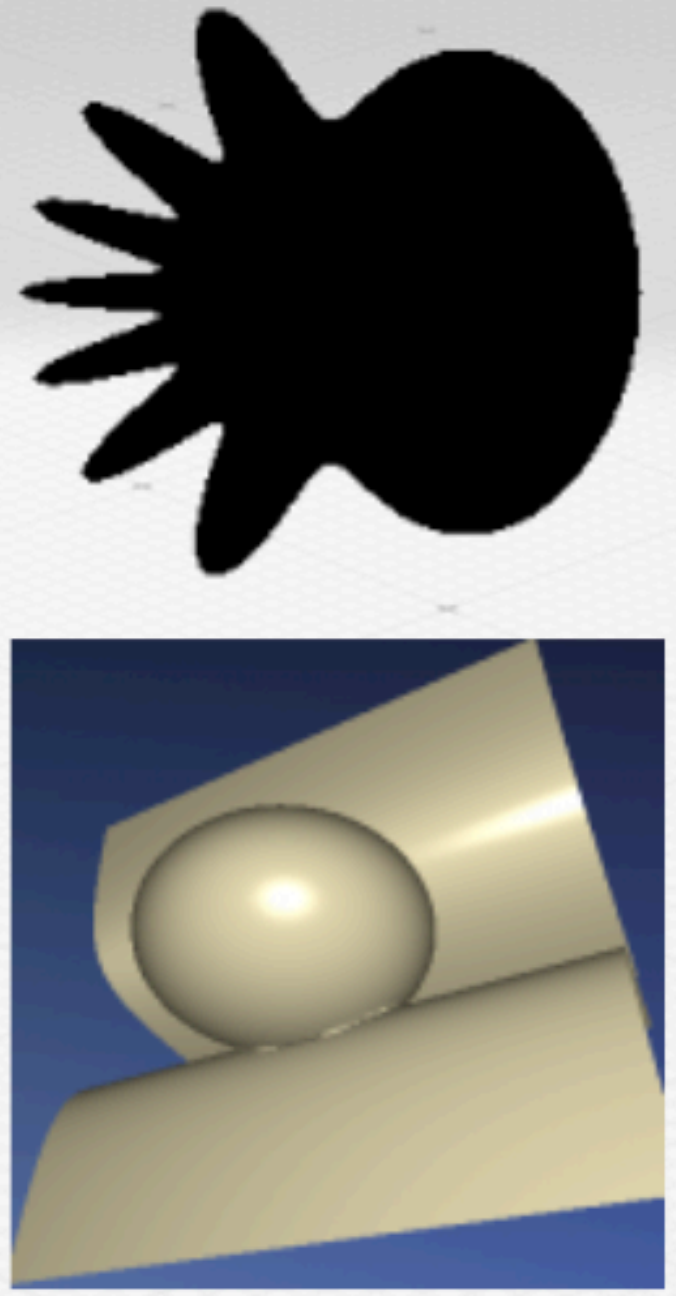
Kernel



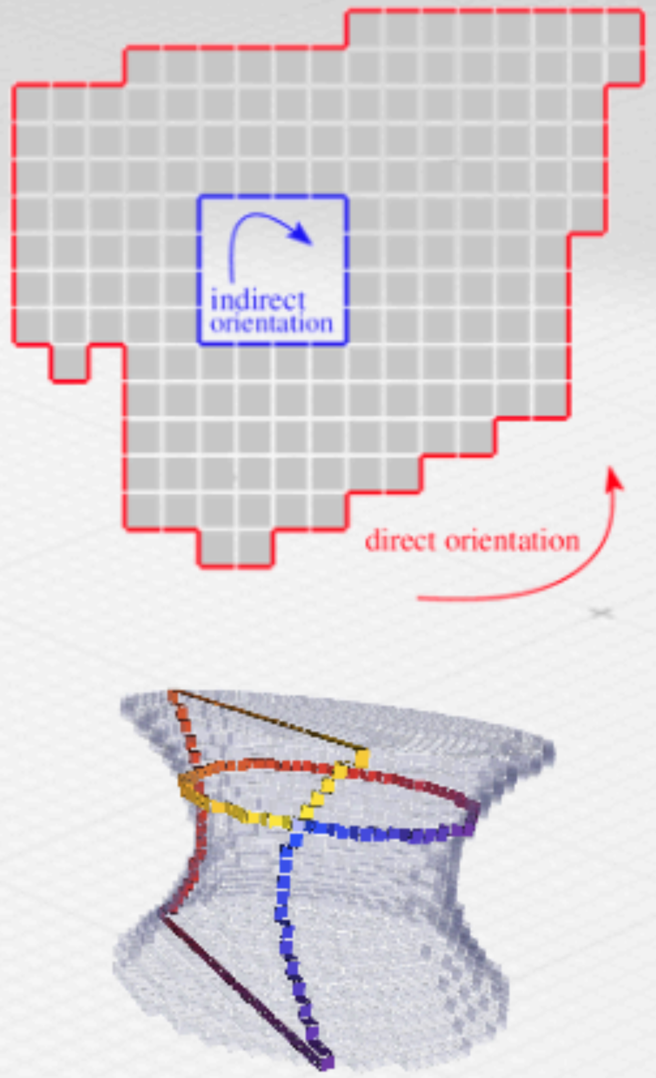
Arithmetic



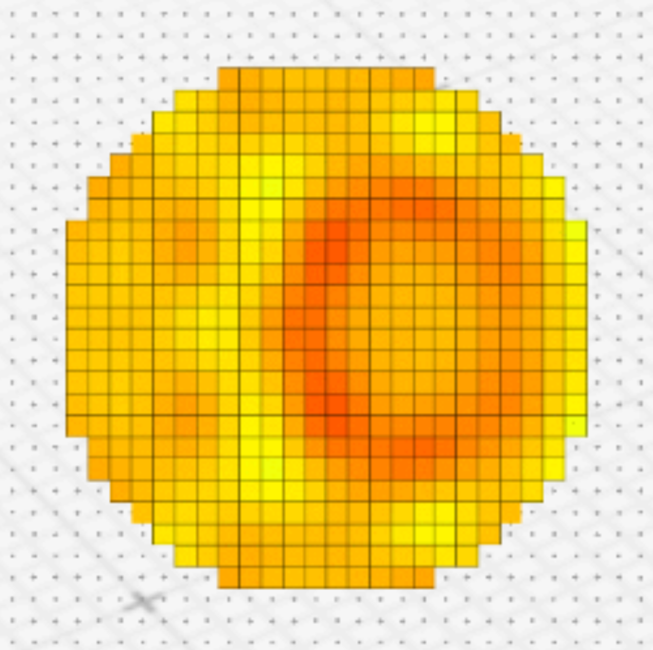
Geometry



Shapes

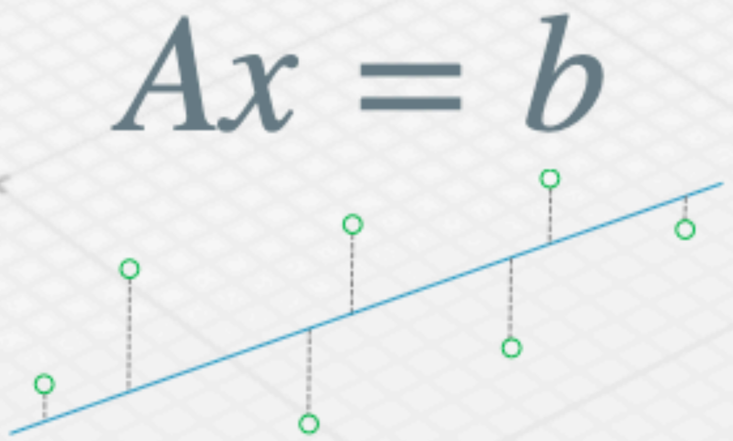


Topology

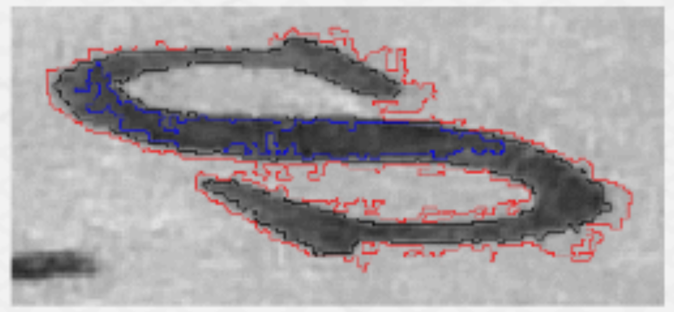


DEC

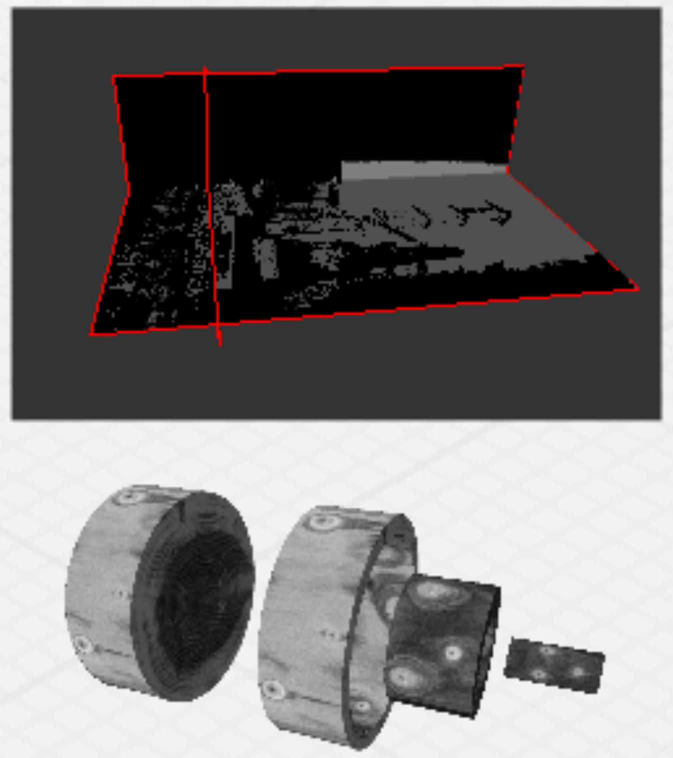
Graph



Mathematic

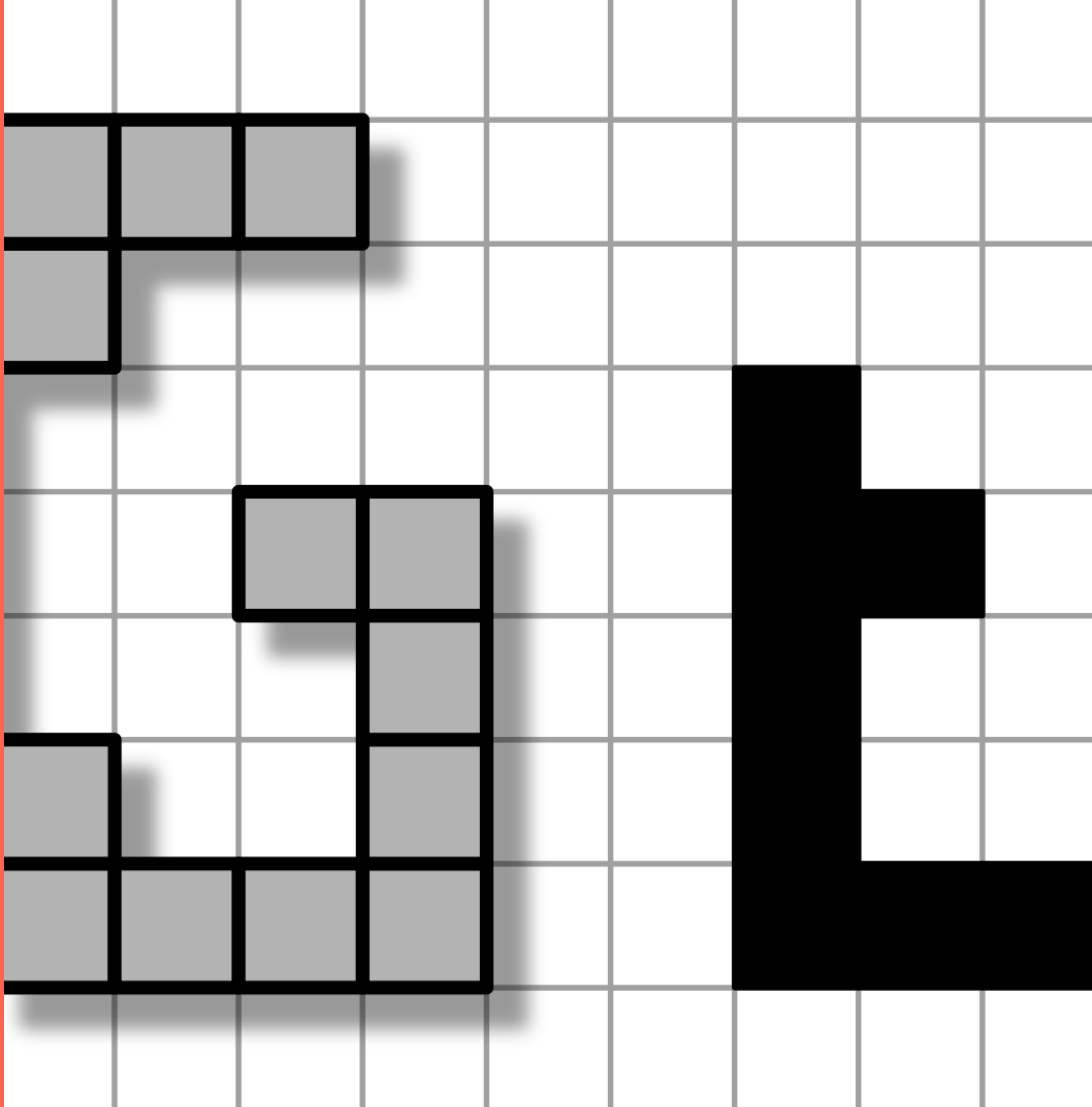


Image



IO

Kernel



Kernel package

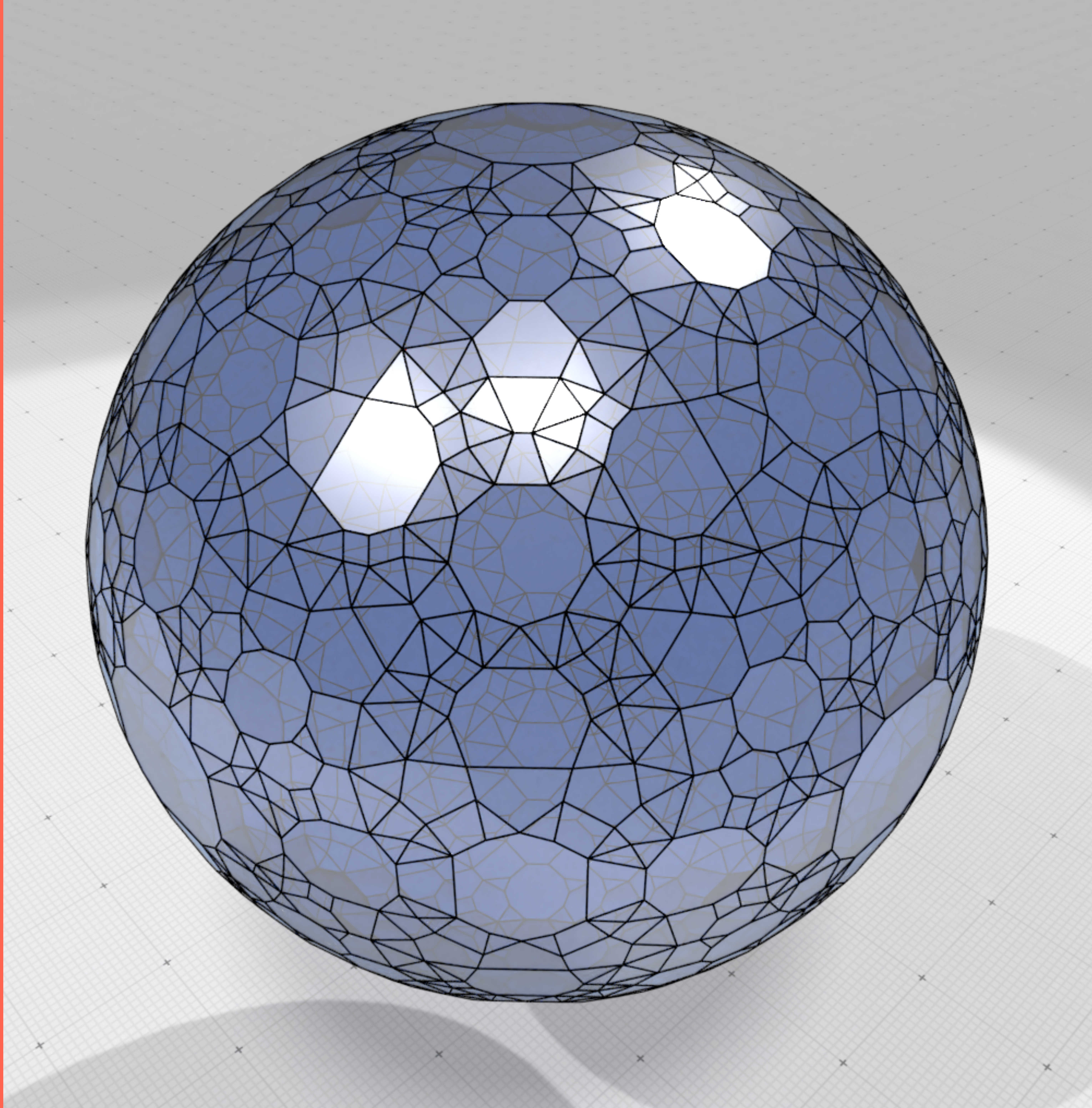
- Representation of Integers (with possible arbitrary precision arithmetic)
- Digital space, domains
- Digital sets
- Helper namespaces

```
typedef int32_t Integer;
typedef DGtal::SpaceND<3, Integer> Space;
typedef DGtal::HyperRectDomain<Space> Domain;
typedef Space::Point Point;
typedef DGtal::DigitalSetBySTLSet<Domain> Set;

Point a(1,2,3);
Point b(2,3,4);
Domain domain(a,b);
Set set(domain);
set.insert( Point(3,3,3) );
//....
```

$\Rightarrow$

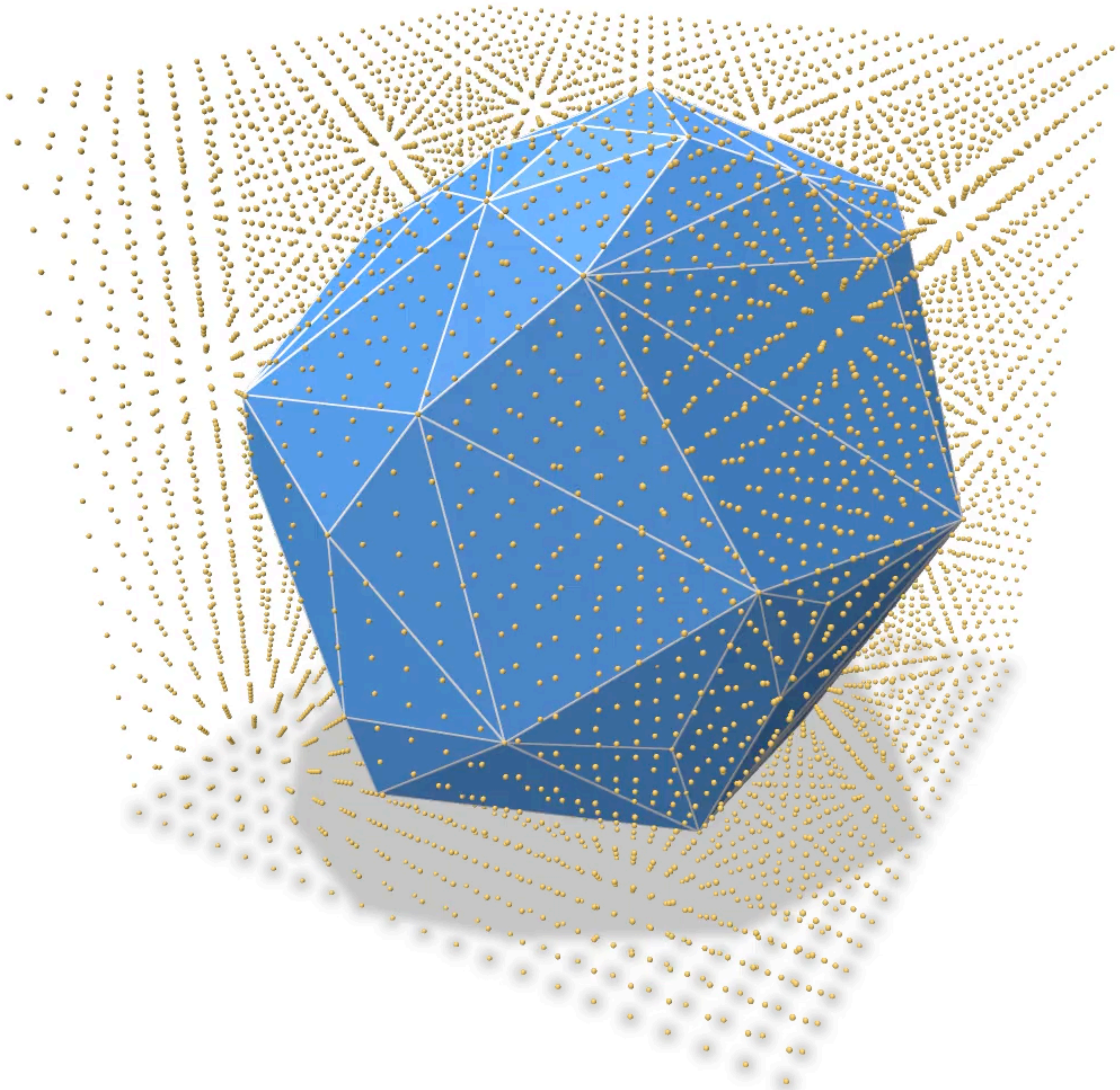
```
Z3i::Point a(1,2,3);
Z3i::Point b(2,3,4);
Z3i::Domain domain(a,b);
Z3i::DigitalSet set(domain);
set.insert( Point(3,3,3) );
//....
```

Arithmetic

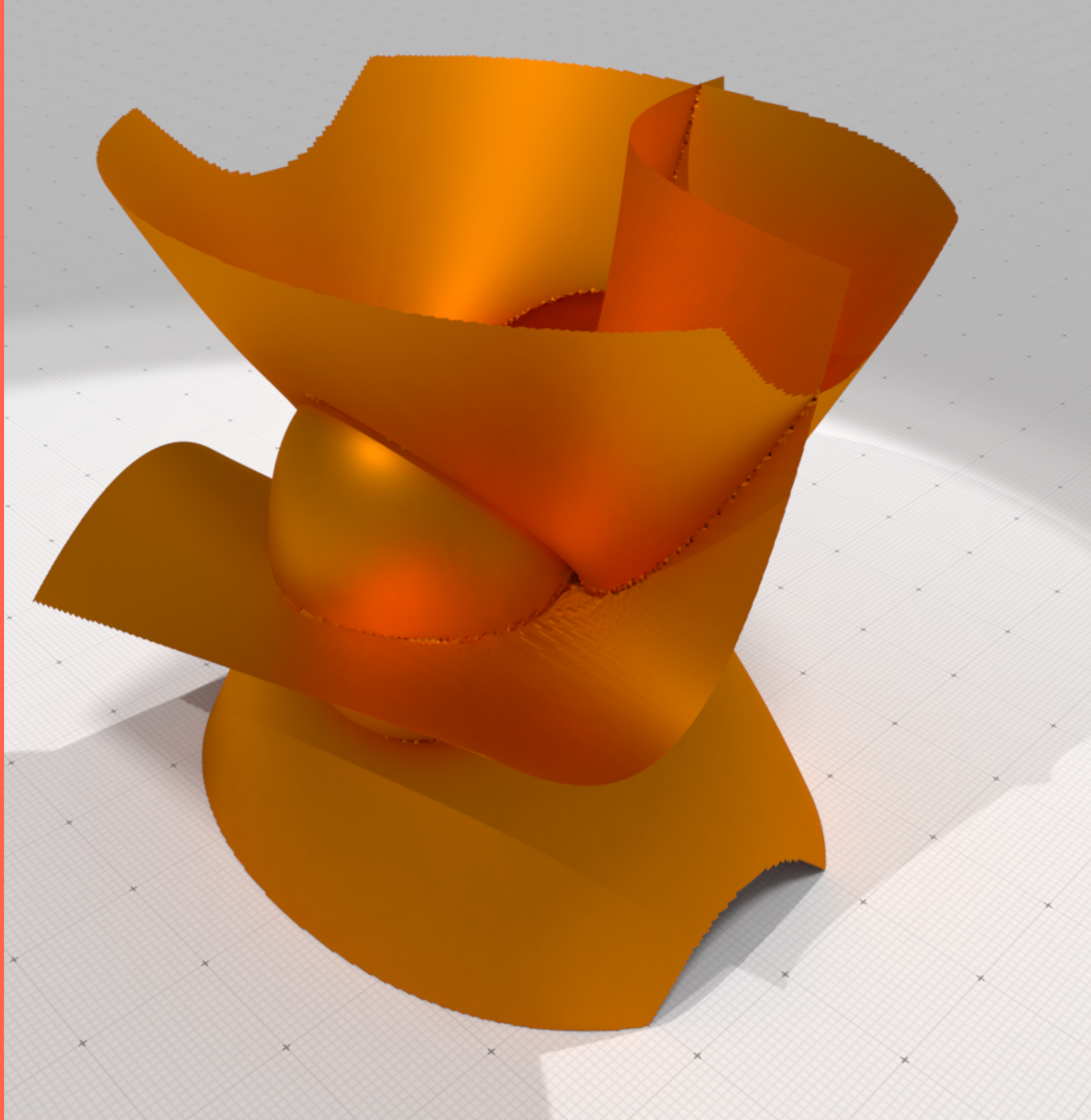


Arithmetic package

- Integers
- Fractions
- Patterns, DSS
- Lattice polytopes



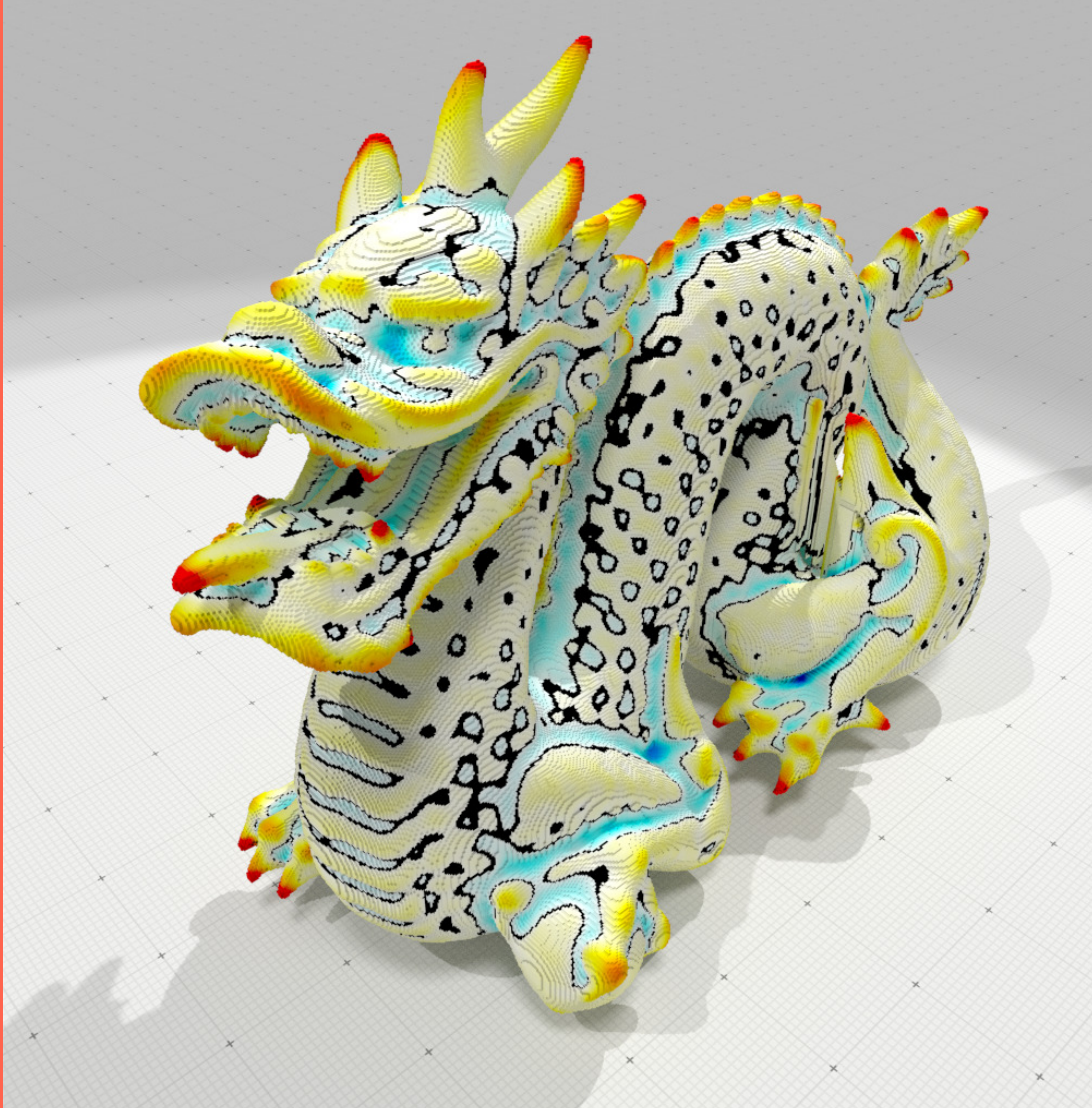
Topology



Topology package

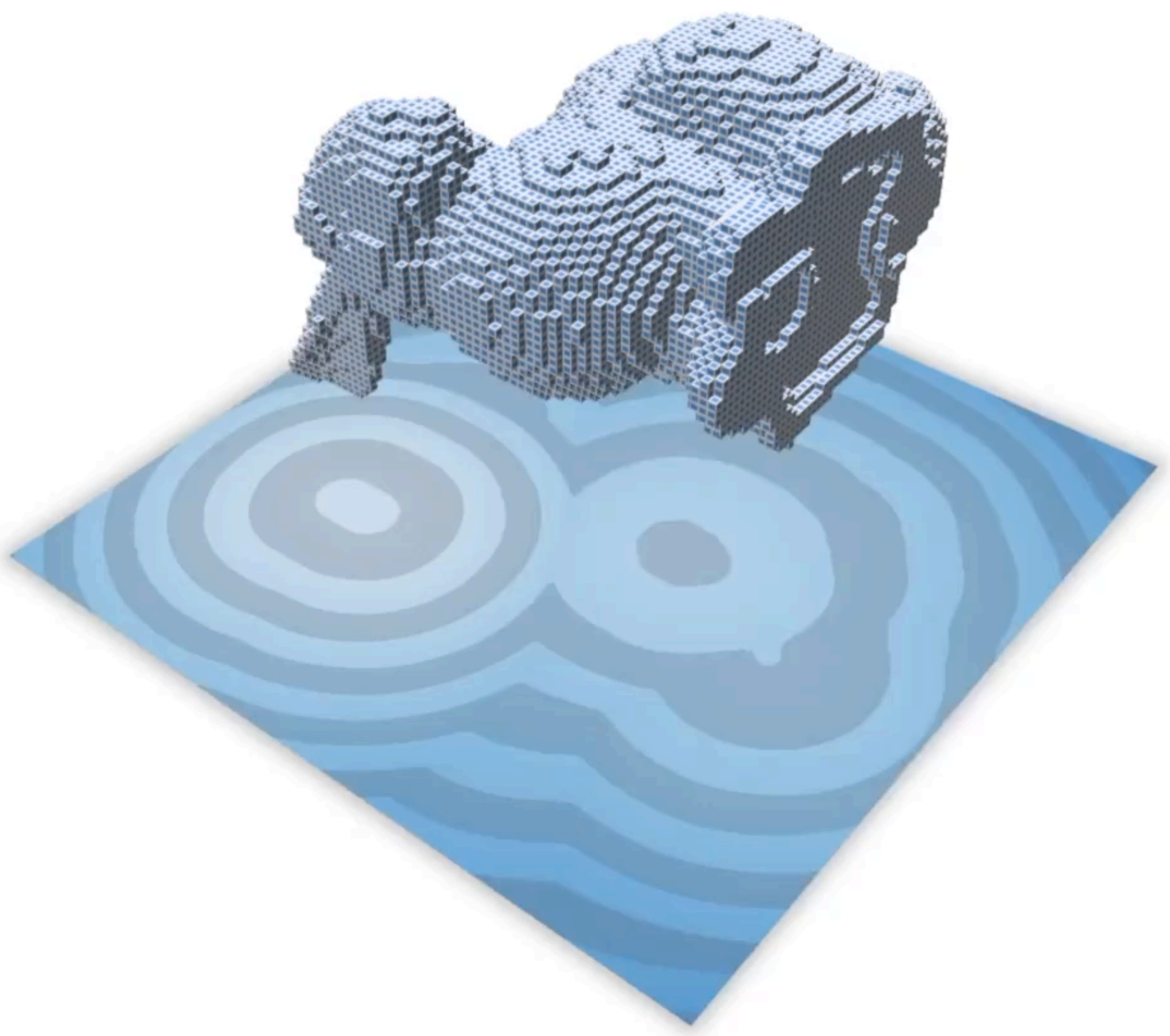
- À la Rosenfeld topology
  - *Adjacency relationships, Jordan pairs, object boundary, simplicity tests*
- Cellular Khalimsky spaces
  - *digital surfaces, tracking, dual surfaces*
- Cubical Complexes
  - Iterators, circulators, closure / star / link / collapse operators
- Voxel complexes
  - Isthmus, critical sets, advanced thinning algorithms

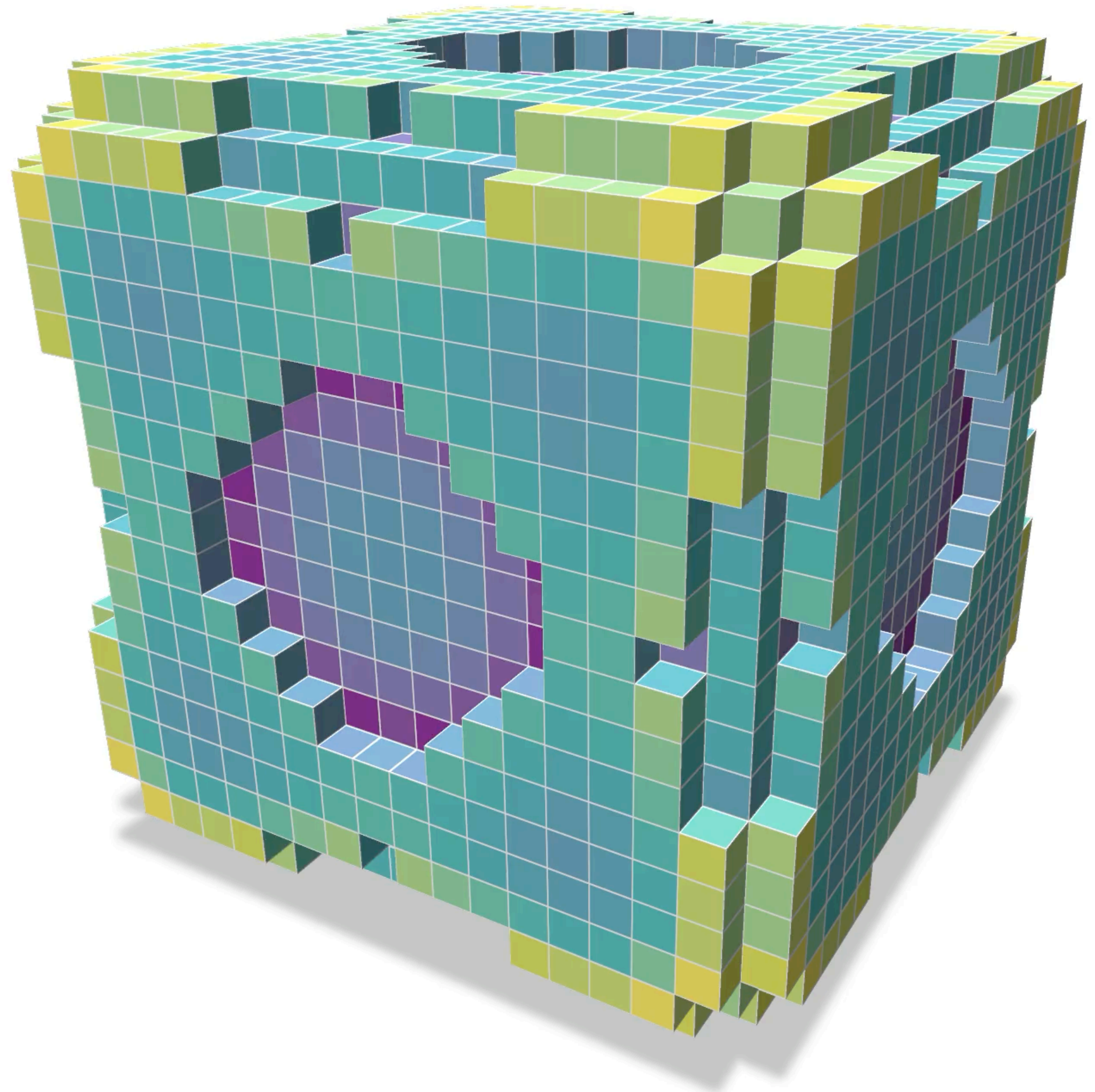
Geometry



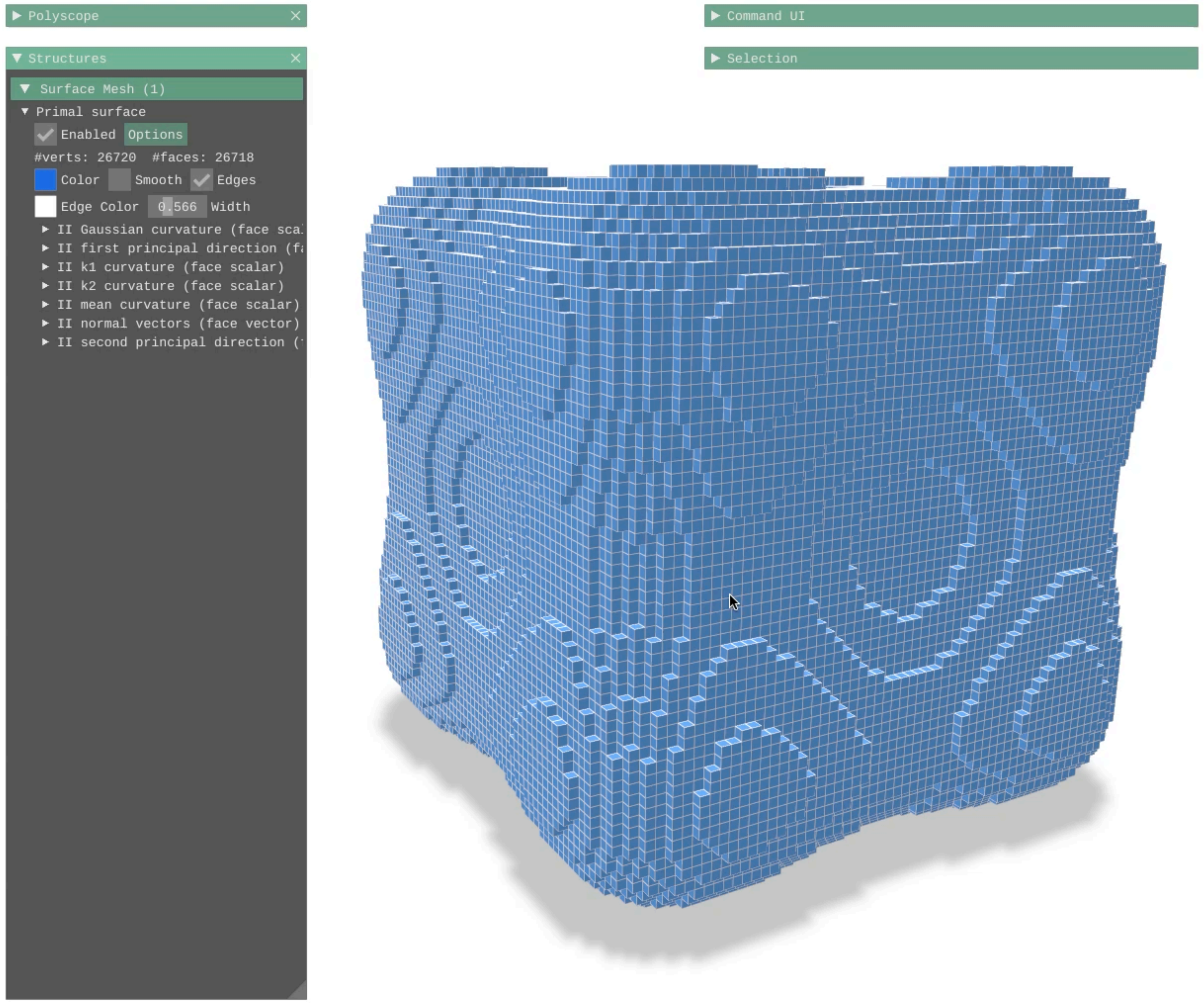
Geometry package

- Elementary objects reconstruction
    - recognition of arithmetical structures (straight lines/segments, circular arcs ,digital planes,…)
- Volumetric analysis
    - metric spaces, distance transformation, voronoi maps, medial axis…
    - convexity, full convexity
    - convex hulls
- Codimensional geometry processing (curves in 2d, surfaces in 3d)
    - differential estimators (length, area, tangent/normals, curvature tensor…)
    - digital surface regularization
    - meshes geometry processing

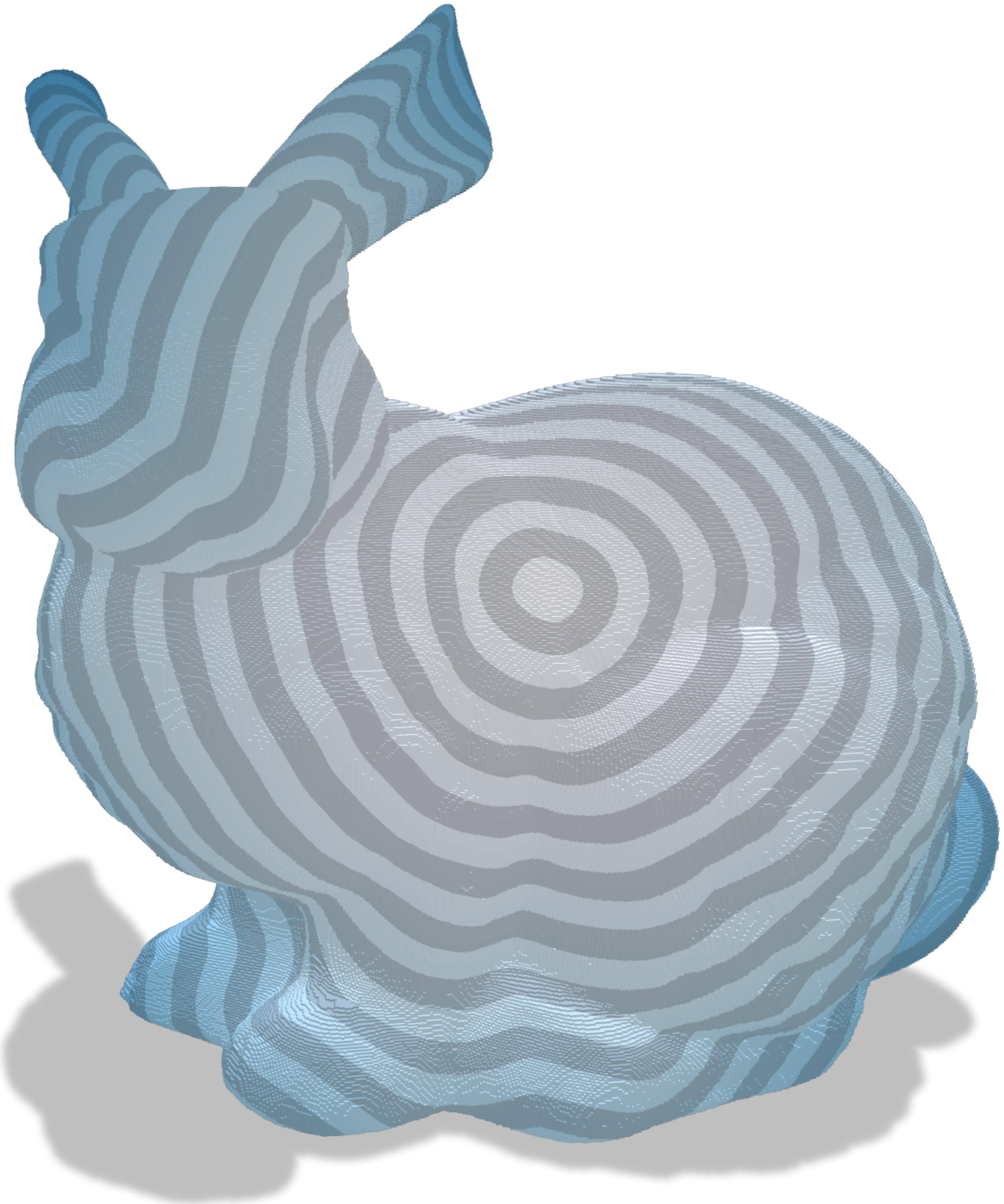




```cpp
void oneStepAll(double h)
{
  auto params = SH3::defaultParameters() | SHG3::defaultParameters() |  SHG3::parametersGeometryEstimation();
  params( "polynomial", "goursat" )( "gridstep", h );
  auto implicit_shape  = SH3::makeImplicitShape3D  ( params );
  auto digitized_shape = SH3::makeDigitizedImplicitShape3D( implicit_shape, params );
  auto K               = SH3::getKSpace( params );
  auto binary_image    = SH3::makeBinaryImage( digitized_shape, params );
  auto surface         = SH3::makeDigitalSurface( binary_image, K, params );
  auto embedder        = SH3::getCellEmbedder( K );
  SH3::Cell2Index c2i;
  auto surfels         = SH3::getSurfelRange( surface, params );
  auto primalSurface   = SH3::makePrimalPolygonalSurface(c2i, surface);

  //Need to convert the faces
  std::vector<std::vector<std::size_t>> faces;
  for(auto &face: primalSurface->allFaces())
    faces.push_back(primalSurface->verticesAroundFace( face ));
  auto digsurf = polyscope::registerSurfaceMesh("Primal surface", primalSurface->positions(), faces);
  digsurf->rescaleToUnit(); digsurf->setEdgeWidth(h*h);  digsurf->setEdgeColor({1.,1.,1.});

  //Computing some differential quantities
  params("r-radius", 5*std::pow(h,-2.0/3.0));
  auto Mcurv     = SHG3::getIIMeanCurvatures(binary_image, surfels, params);
  auto normalsII = SHG3::getIINormalVectors(binary_image, surfels, params);
  auto KTensor   = SHG3::getIIPrincipalCurvaturesAndDirections(binary_image, surfels, params);  //Recomputing...

  std::vector<double> Gcurv(surfels.size()),k1(surfels.size()),k2(surfels.size());
  std::vector<RealVector> d1(surfels.size()),d2(surfels.size());
  auto i=0;
  for(auto &t: KTensor) //AOS->SOA
  {
    k1[i]    = std::get<0>(t);
    k2[i]    = std::get<1>(t);
    d1[i]    = std::get<2>(t);
    d2[i]    = std::get<3>(t);
    Gcurv[i] = k1[i]*k2[i];
    ++i;
  }

  //Attaching quantities
  digsurf->addFaceVectorQuantity("II normal vectors", normalsII, polyscope::VectorType::AMBIENT);
  digsurf->addFaceScalarQuantity("II mean curvature", Mcurv);
  digsurf->addFaceScalarQuantity("II Gaussian curvature", Gcurv);
  digsurf->addFaceScalarQuantity("II k1 curvature", k1);
  digsurf->addFaceScalarQuantity("II k2 curvature", k2);
  digsurf->addFaceVectorQuantity("II first principal direction", d1, polyscope::VectorType::AMBIENT);
  digsurf->addFaceVectorQuantity("II second principal direction", d2, polyscope::VectorType::AMBIENT);
}
```

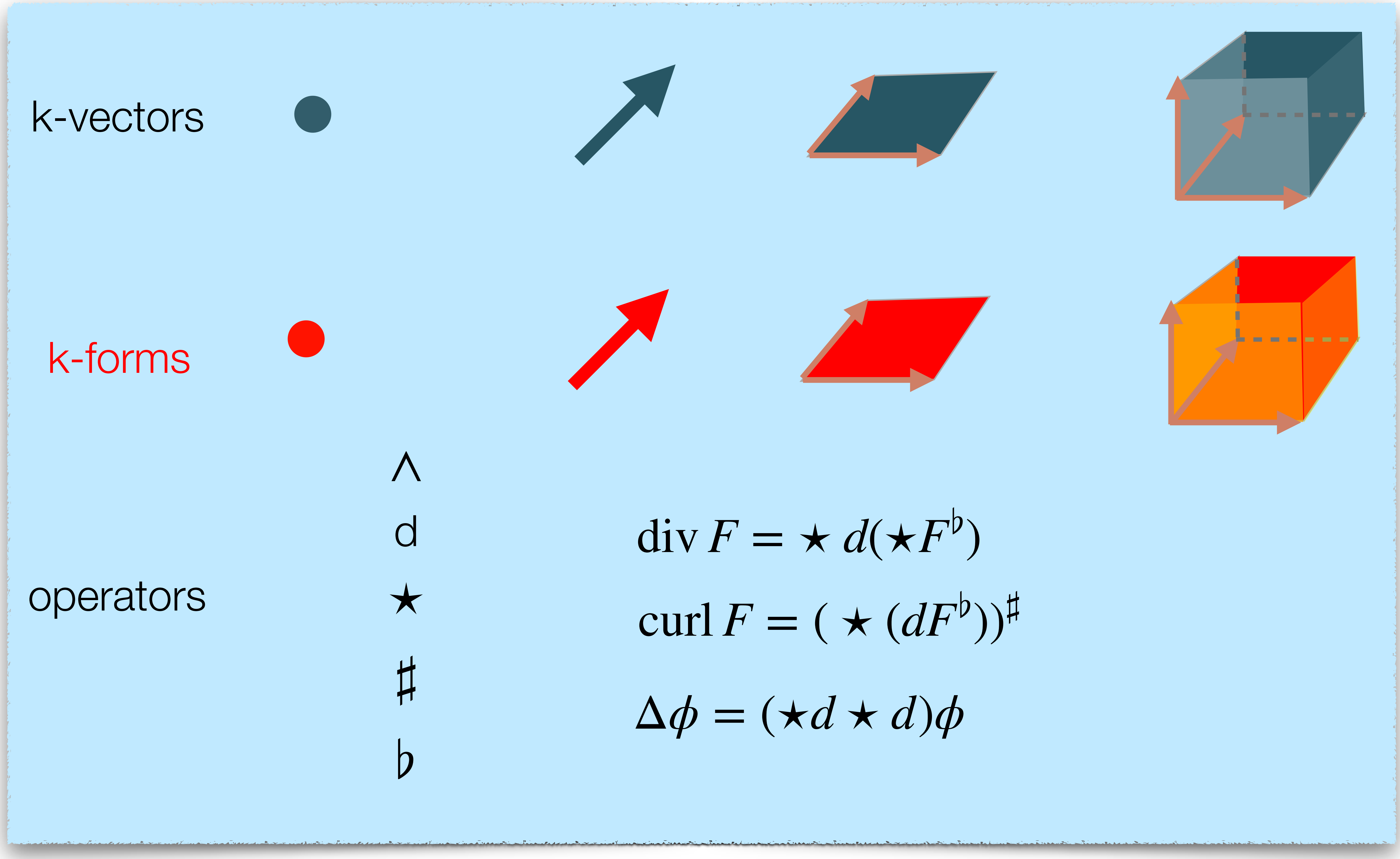


DEC



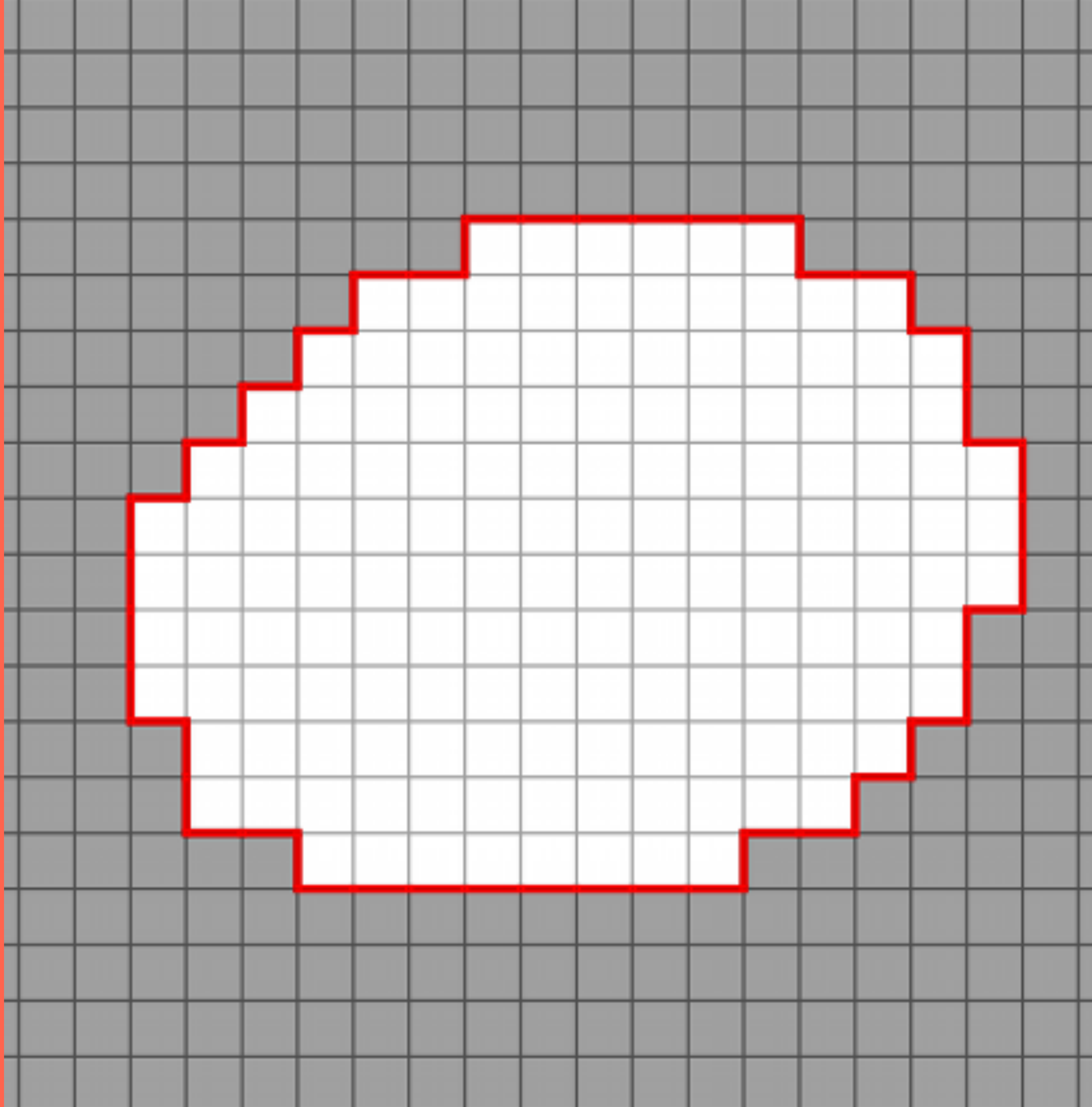
Calculus on digital surfaces

- Discrete Exterior Calculus
  - k-vectors, k-forms, operators
  - examples to solve PDE



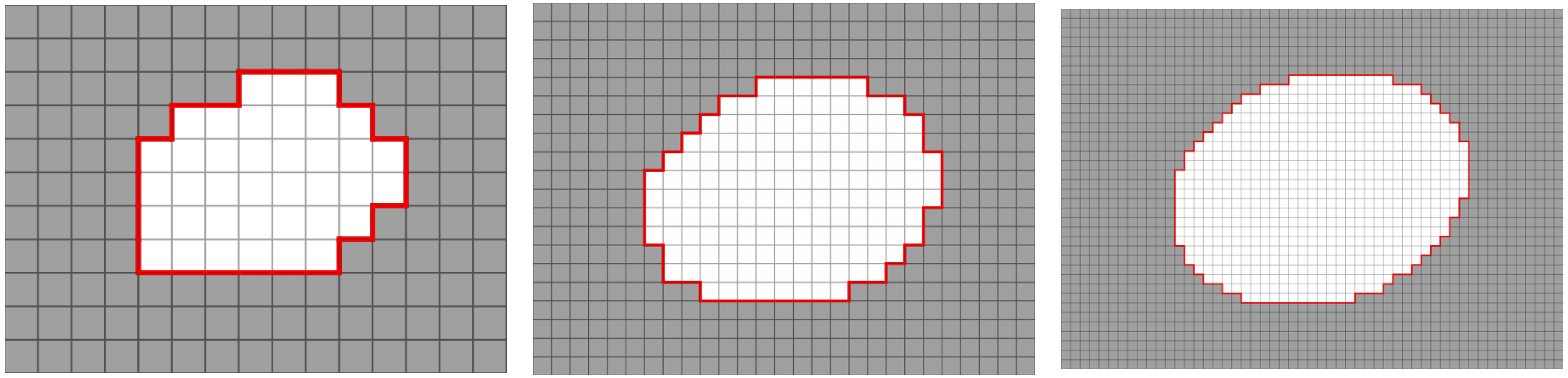
- Simple calculus on digital and polygonal surfaces (Wednesday!)

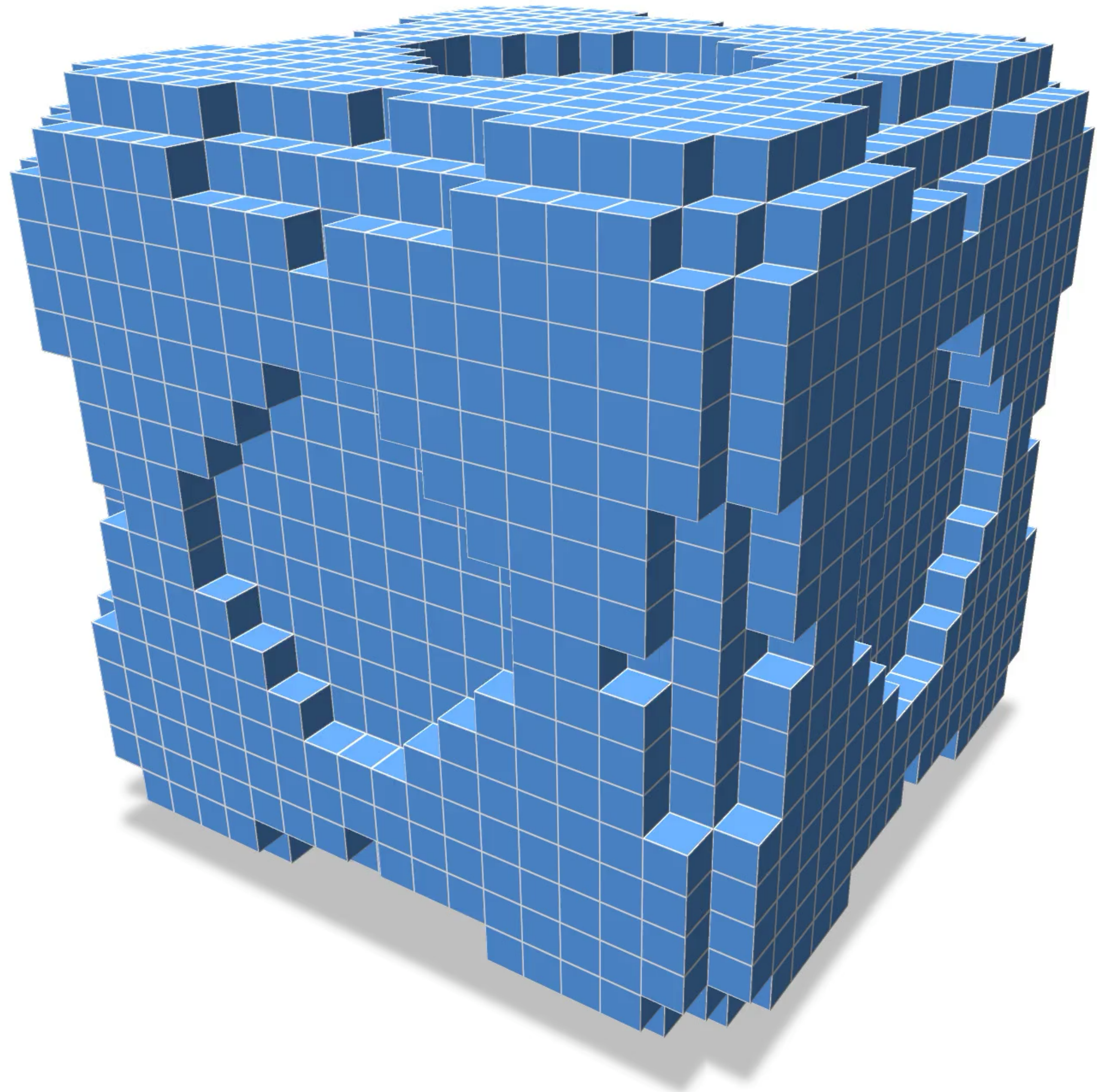
Shapes



Shapes package

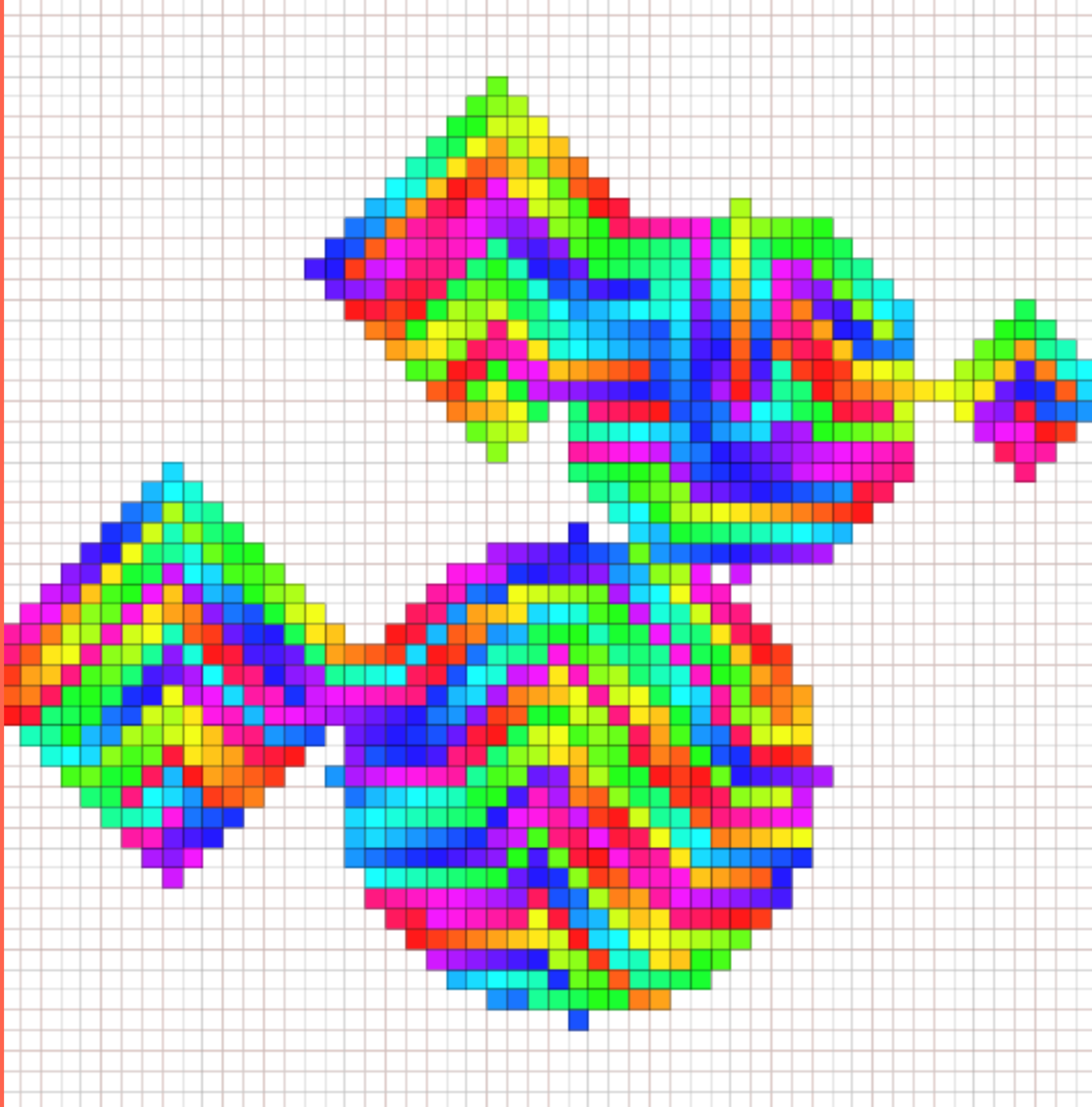
- Implicit shapes with *exact* quantities (length, area, curvatures…)
- Digitizer (Gauss digitization)





- half-edge / surface mesh data structures

Graph



Graph package

- Basic graphs and concepts
- Basic algorithm (traversal,...)
- Interface with boost::graph

Image

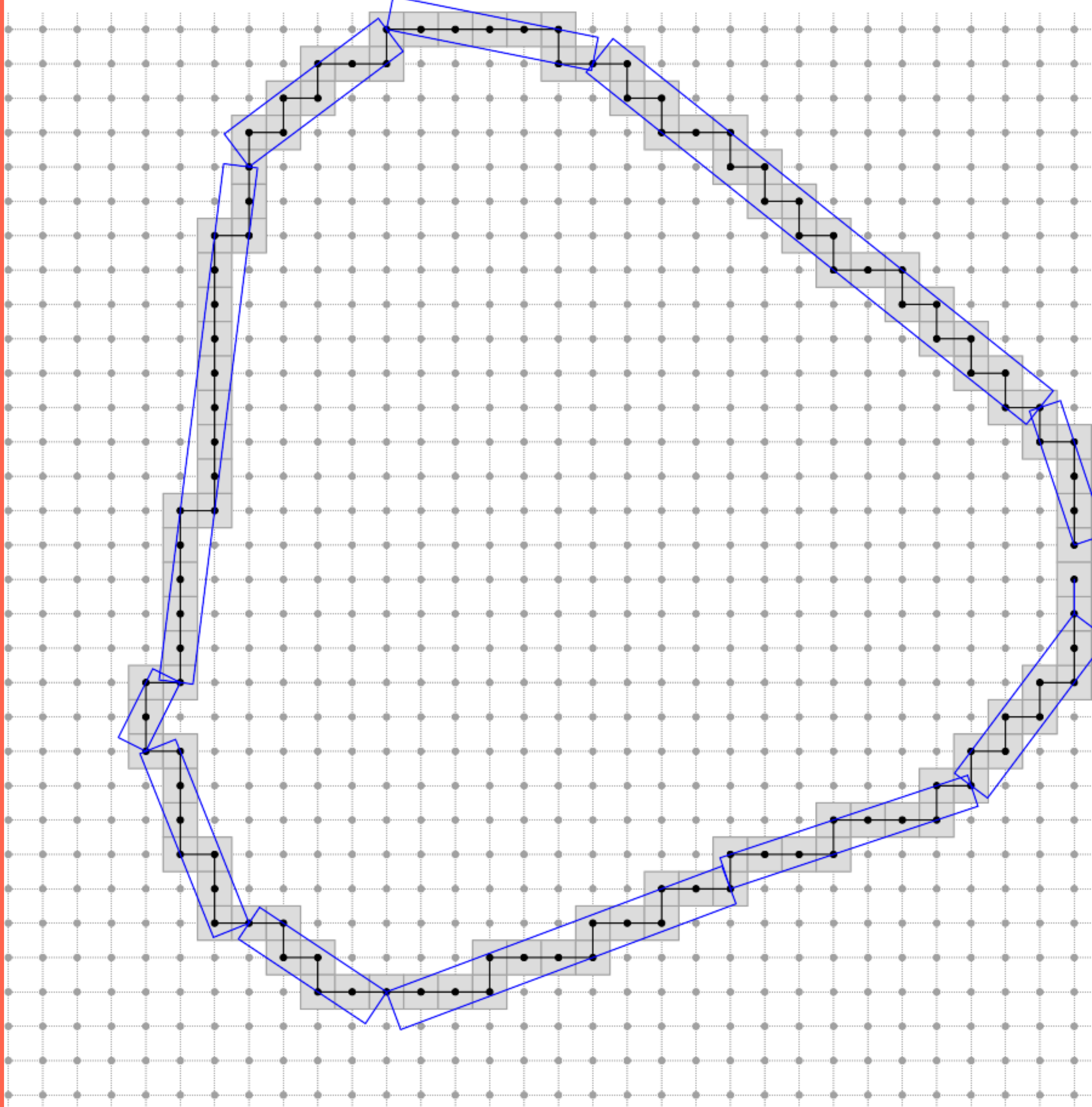

subSampledImage3D1x1x1.vol
subSampledImage3D2x2x2.vol
subSampledImage1x1.png
subSampledImage2x2.png

Image package

- Image containers
  - Generic containers (dimension, value type, accessors)
  - random access flat containers (eg. std::vector), associative container point↔ value
  - Tile based container
  - experimental Hashtree based container
  - *ITK* containers

- Many tools to convert, remap, or construct facades on images

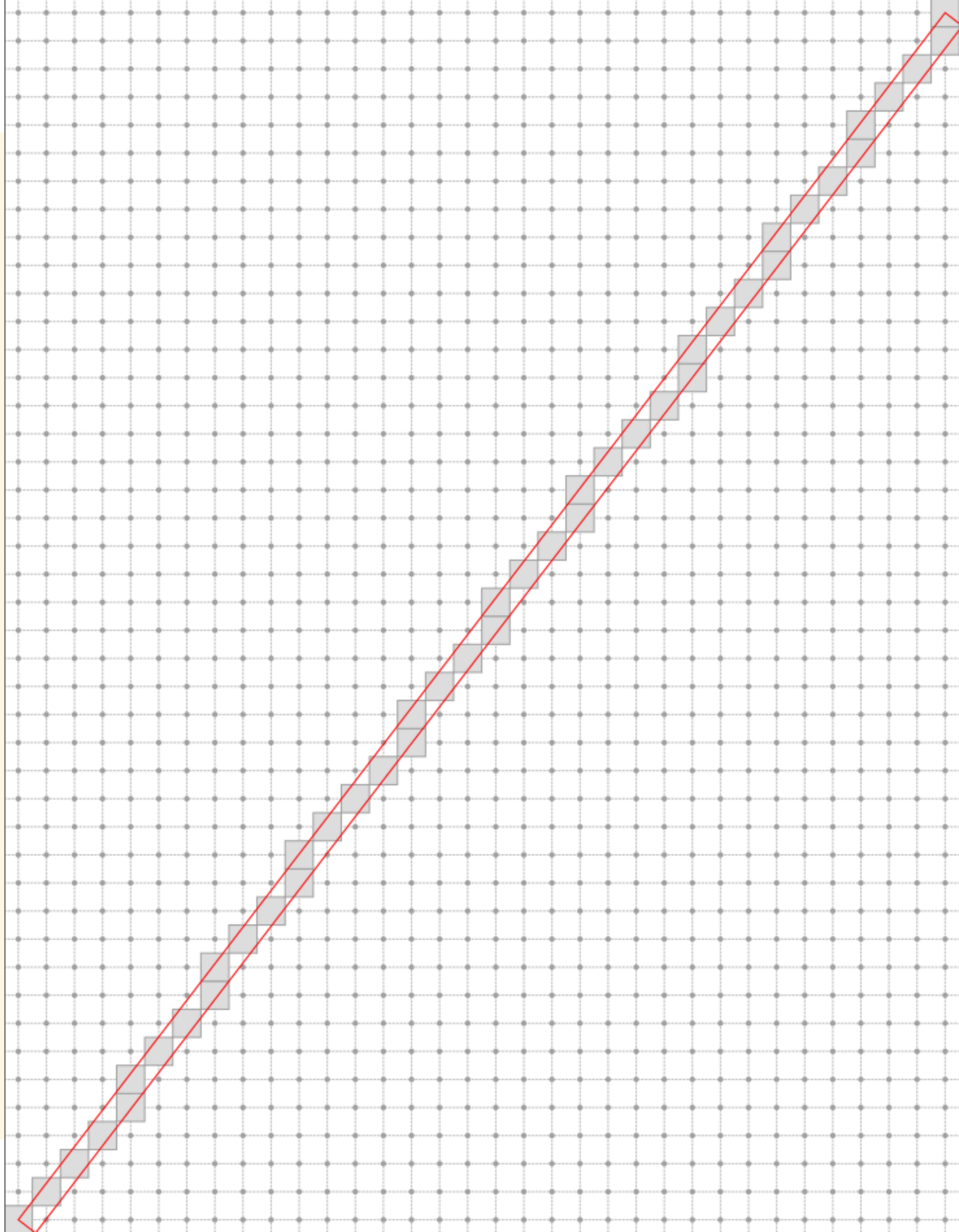
10



IO package

- Images/volume IO
  - Image file formats (png, jpg…)
  - Volumetric formats (vol, itk…)
- Boards
  - Export 2d and 3d structures to svg/eps figures
- Viewers
  - Interactive 3d viewers with libQGlviewers

```
#include <iostream>
#include <DGtal/base/Common.h>
#include <DGtal/helpers/StdDefs.h>
#include <DGtal/io/boards/Board2D.h>
#include <DGtal/geometry/curves/ArithmeticalDSS.h>

using namespace DGtal;

int main()
{
  Z2i::Point a(0,0);
  Z2i::Point b(33,43);

  Z2i::Domain domain(a,b);

  NaiveDSS8<Z2i::Integer> dss(a,b);

  Board2D board;
  board << domain;
  board << dss;

  board.saveSVG("simple-domain.svg");
  return 0;
}
```

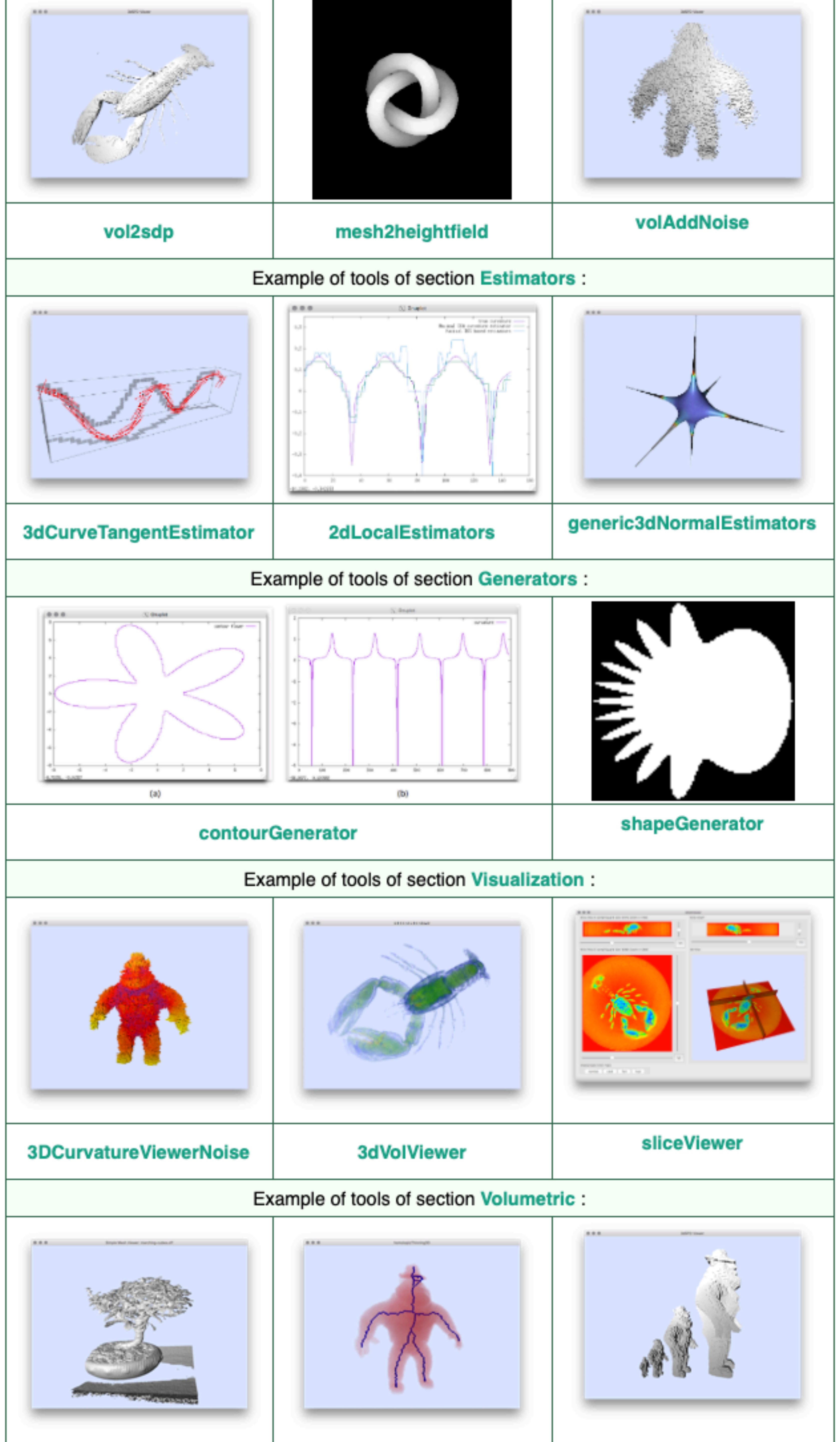


Math

Math package

- Linear algebra (facade to Eigen)
- Multivariate polynomials
- Lagrange polynomials / Lagrange interpolation

DGtalTools

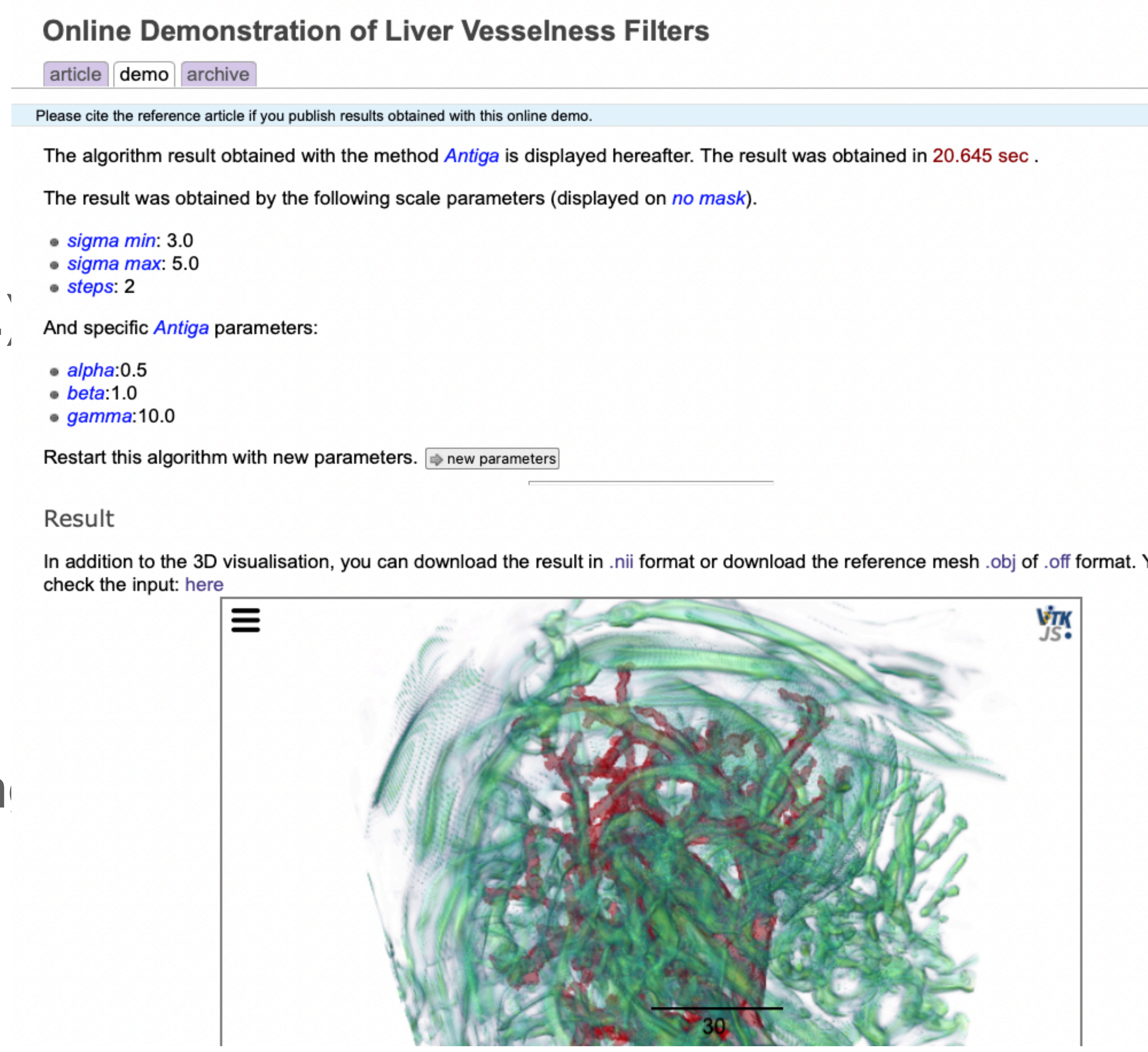

vol2sdp
mesh2heightfield
volAddNoise
Example of tools of section Estimators :
3dCurveTangentEstimator
2dLocalEstimators
generic3dNormalEstimators
Example of tools of section Generators :
(a)
(b)
contourGenerator
shapeGenerator
Example of tools of section Visualization :
3DCurvatureViewerNoise
3dVolViewer
sliceViewer
Example of tools of section Volumetric :

DGtalTools

Separate GitHub project: https://github.com/DGtal-team/DGtalTools

Light tools based on DGtal algorithms:

- Useful to share and apply results on various application domains.
- Make easier online demonstration (like IPOL).
- Provides simple independent tools for various domains:
  - Convertors: converts various file format (vol2raw, dicom2vol, mesh2heightfield …)
  - Estimators: apply different geometric estimator (tangent, curvature 2D/3D).
  - Generators: utilities to generate various contours/shape.
  - Visualization: visualize digital data (set of voxels, vol file, height map, mesh).
  - Volumetric: to manipulate volumetric files (marching-cubes, sub sampling, thinning
  - Image processing: tools to process images (image restoration, image inpainting)

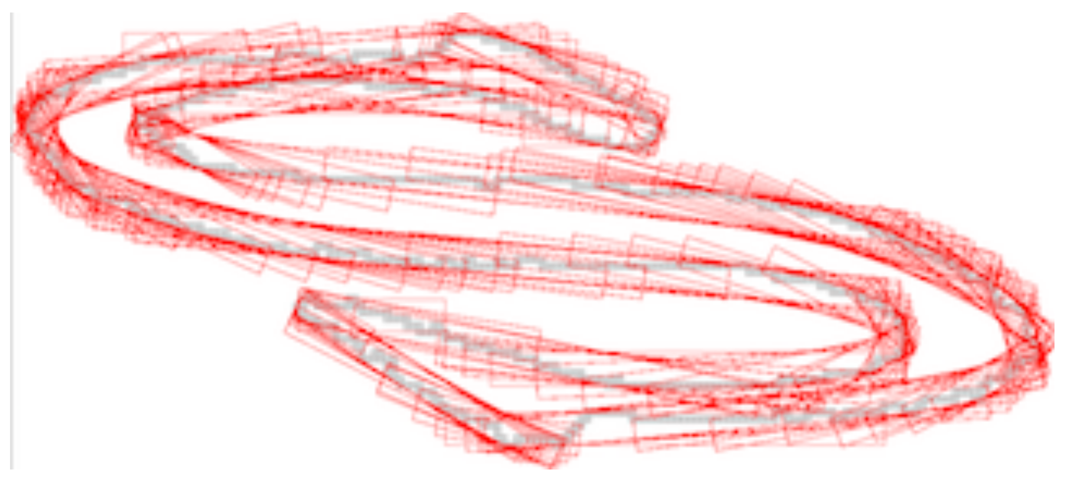


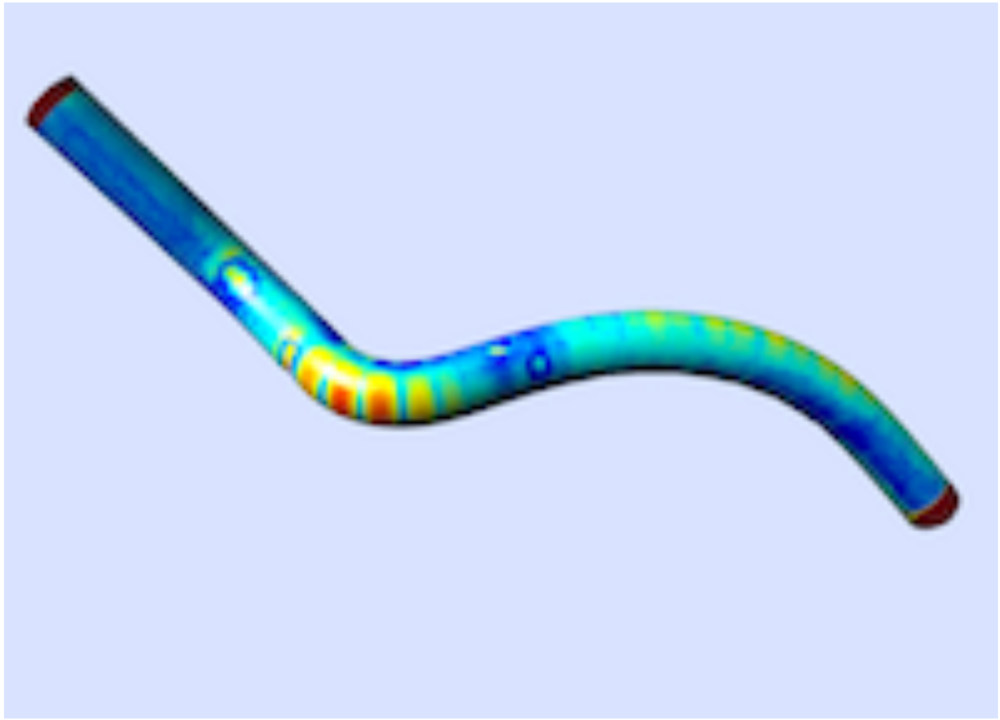
DGtalTools-contrib

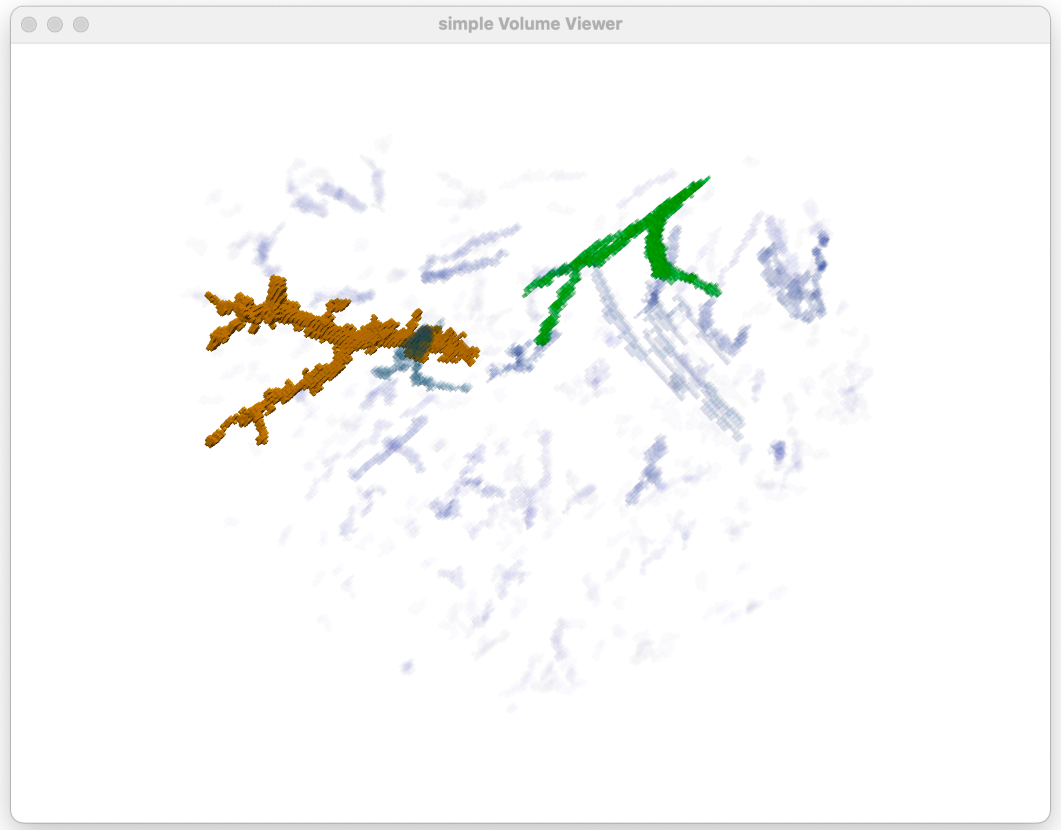
Separate GitHub project:

https://github.com/DGtal-team/DGtalTools-contrib

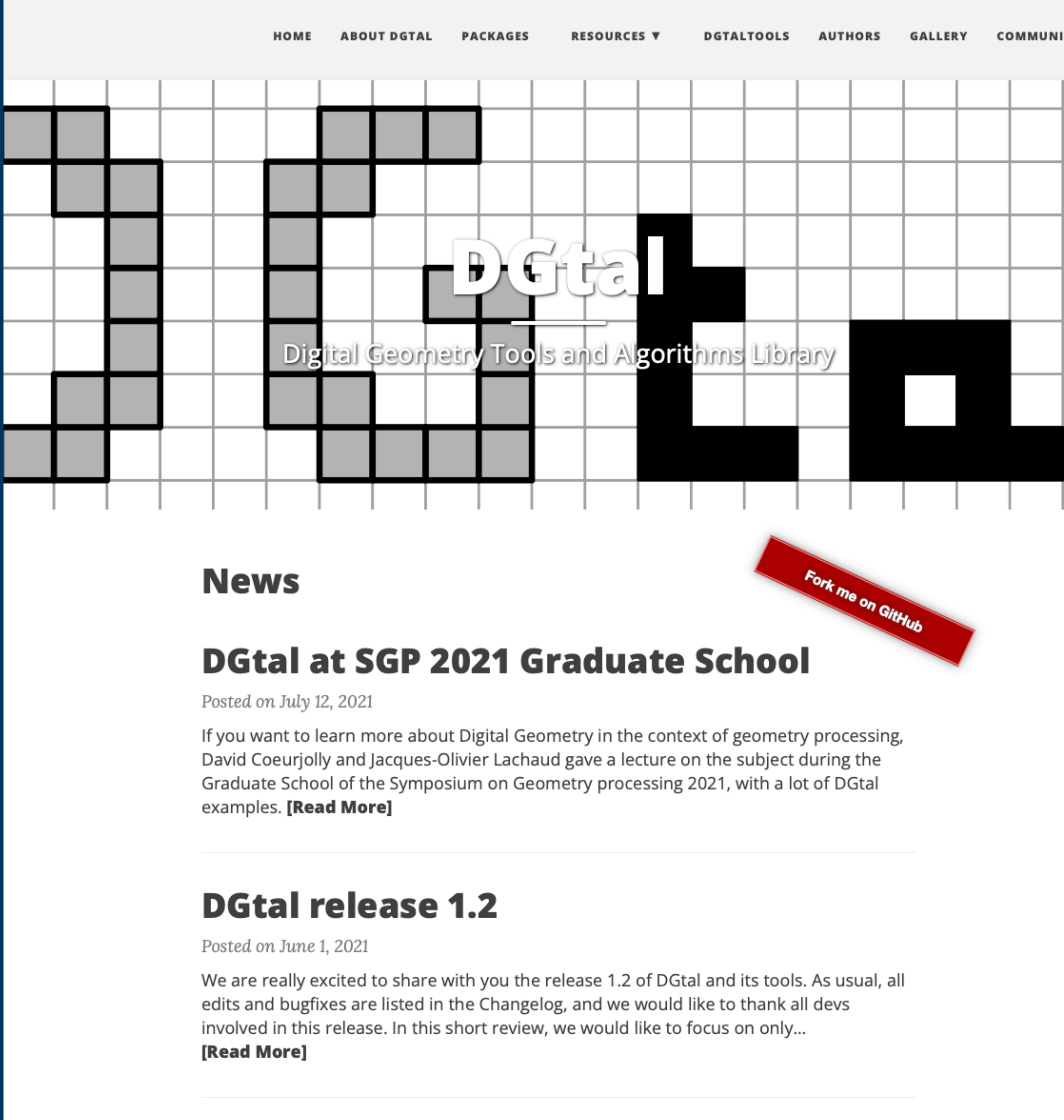
- Tools considered as development or prototype.
- Used to share tools useful in recent research in progress.





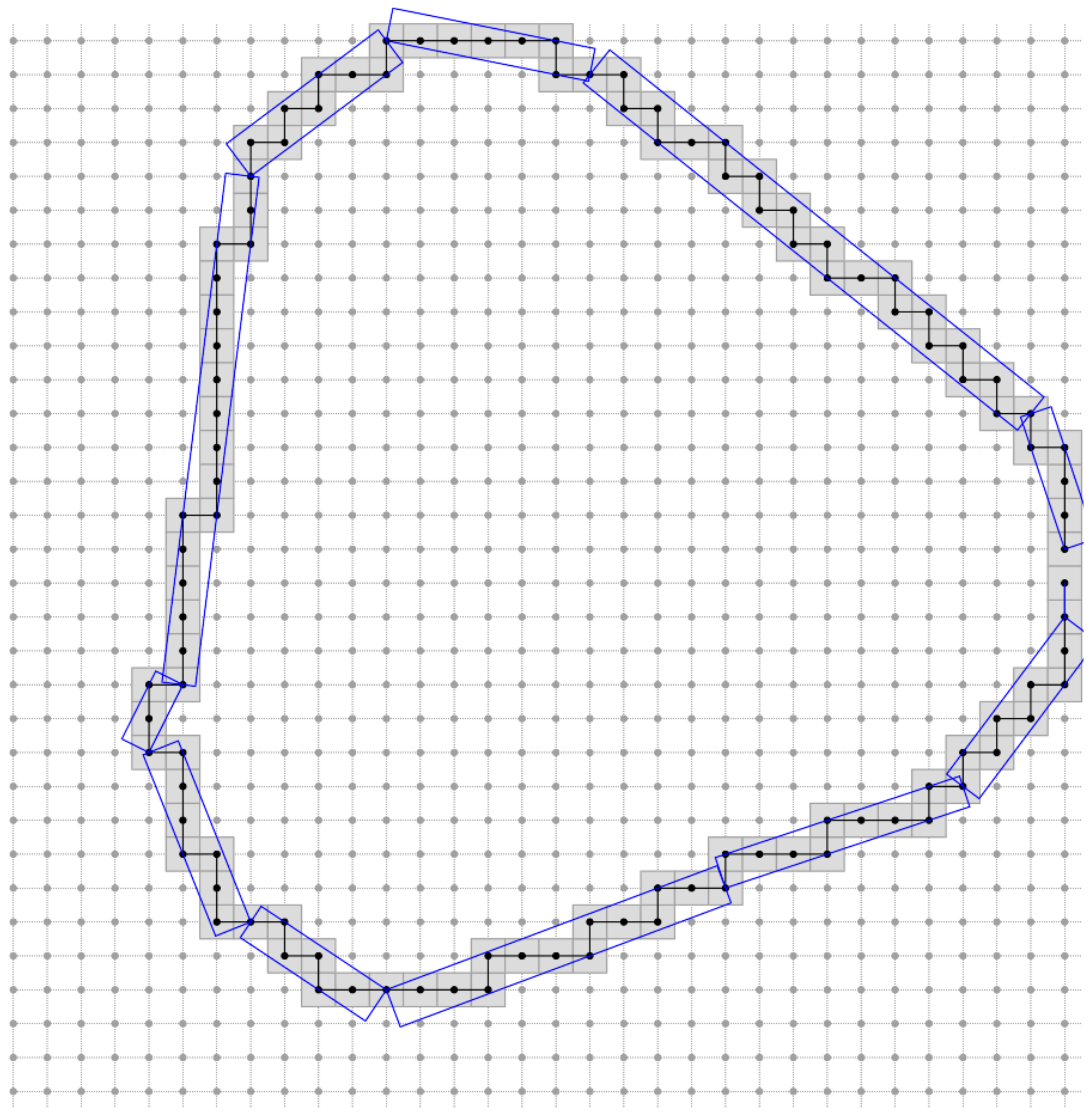


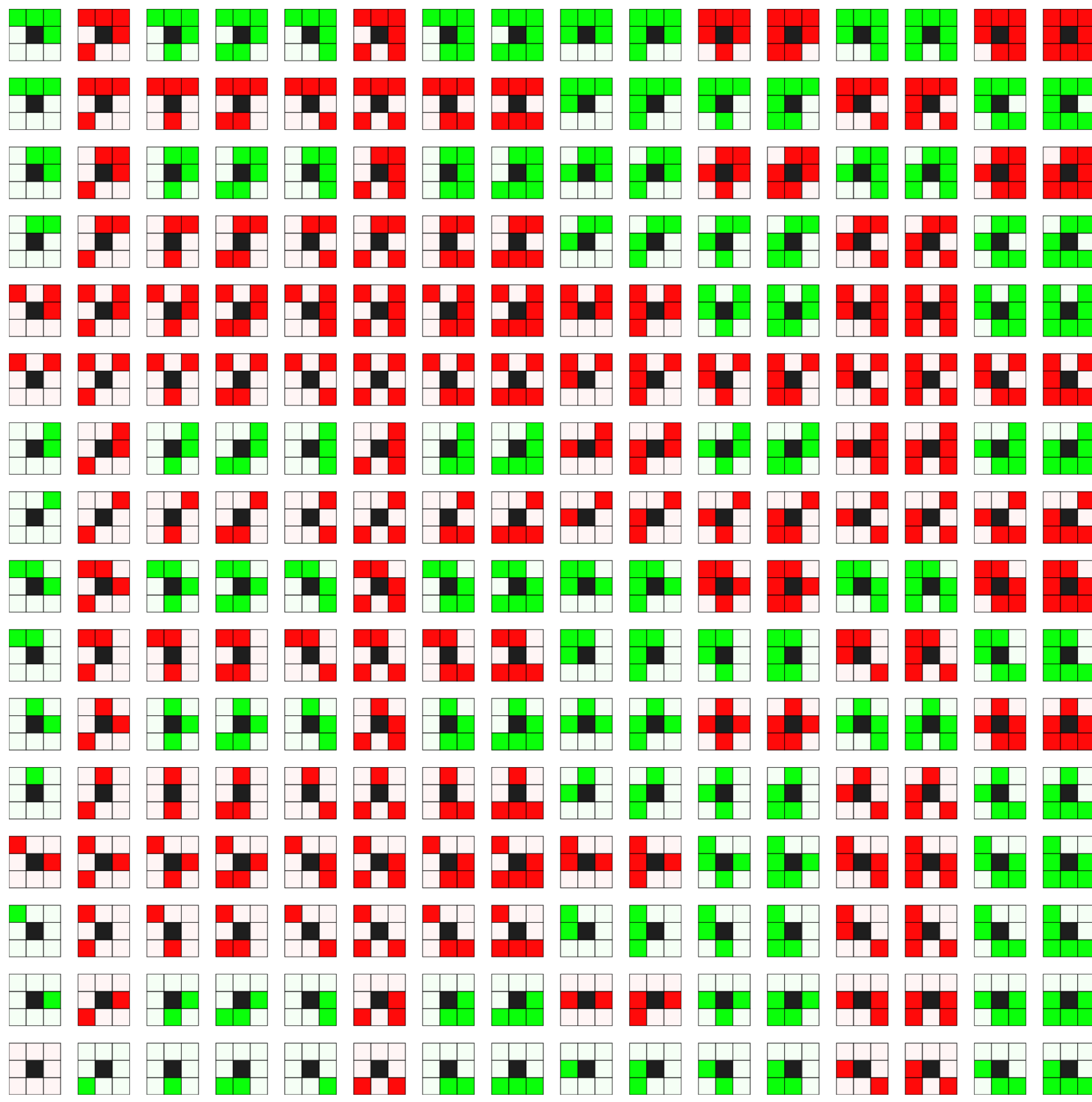
Website, documentation, GitHub...

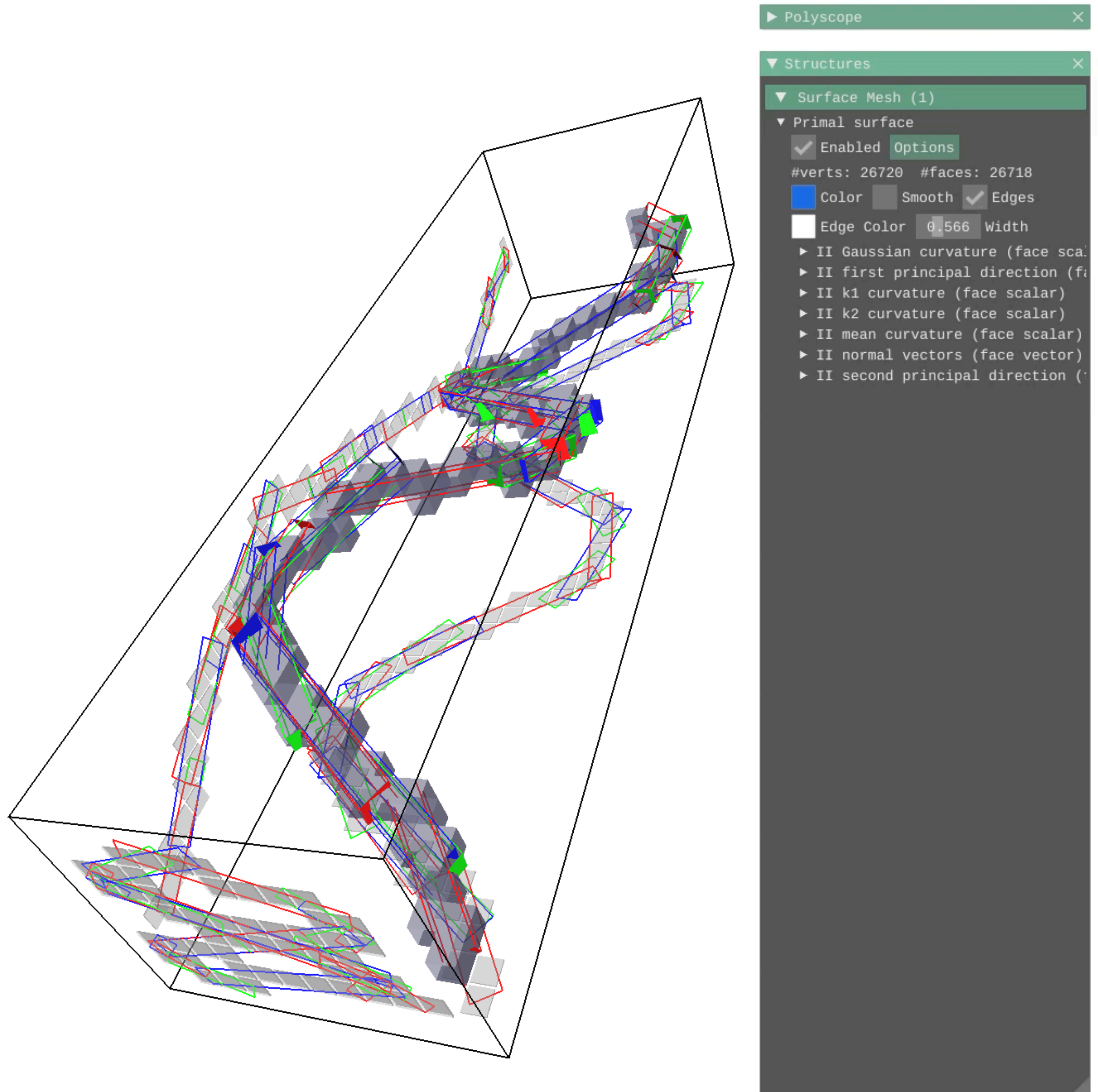


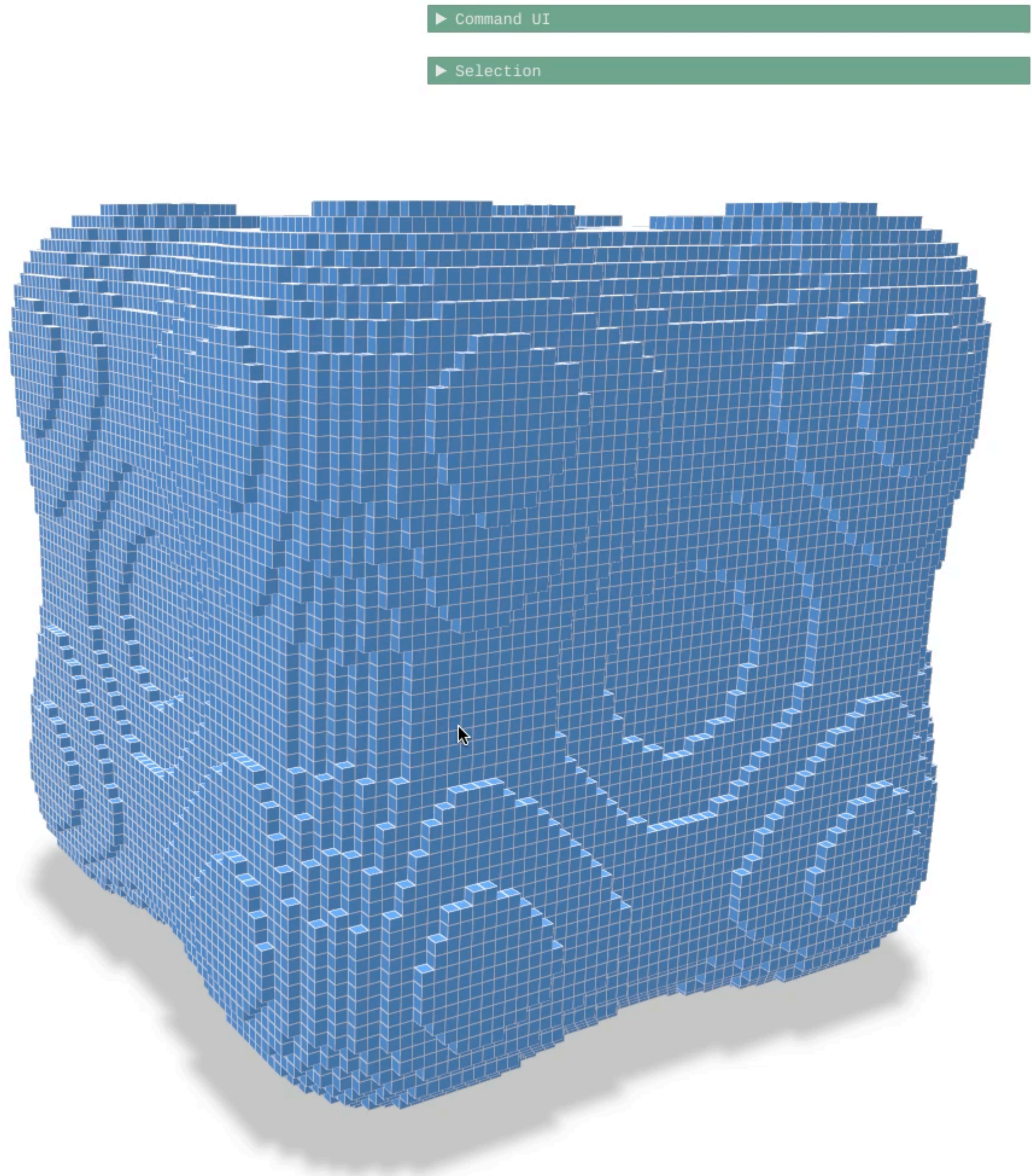
Focus:

IO/Interactions

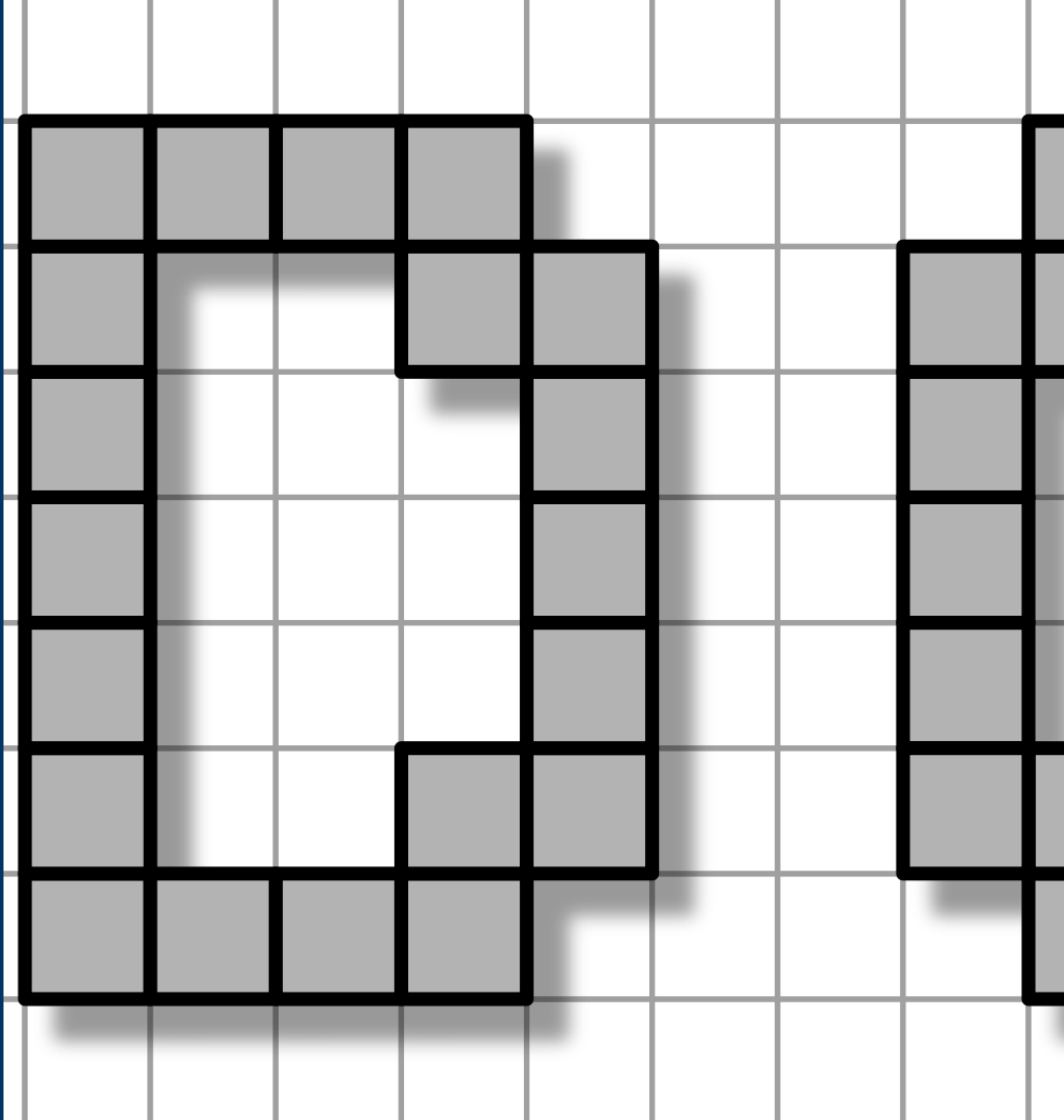








Integrate DGtal in your projects



CMake integration

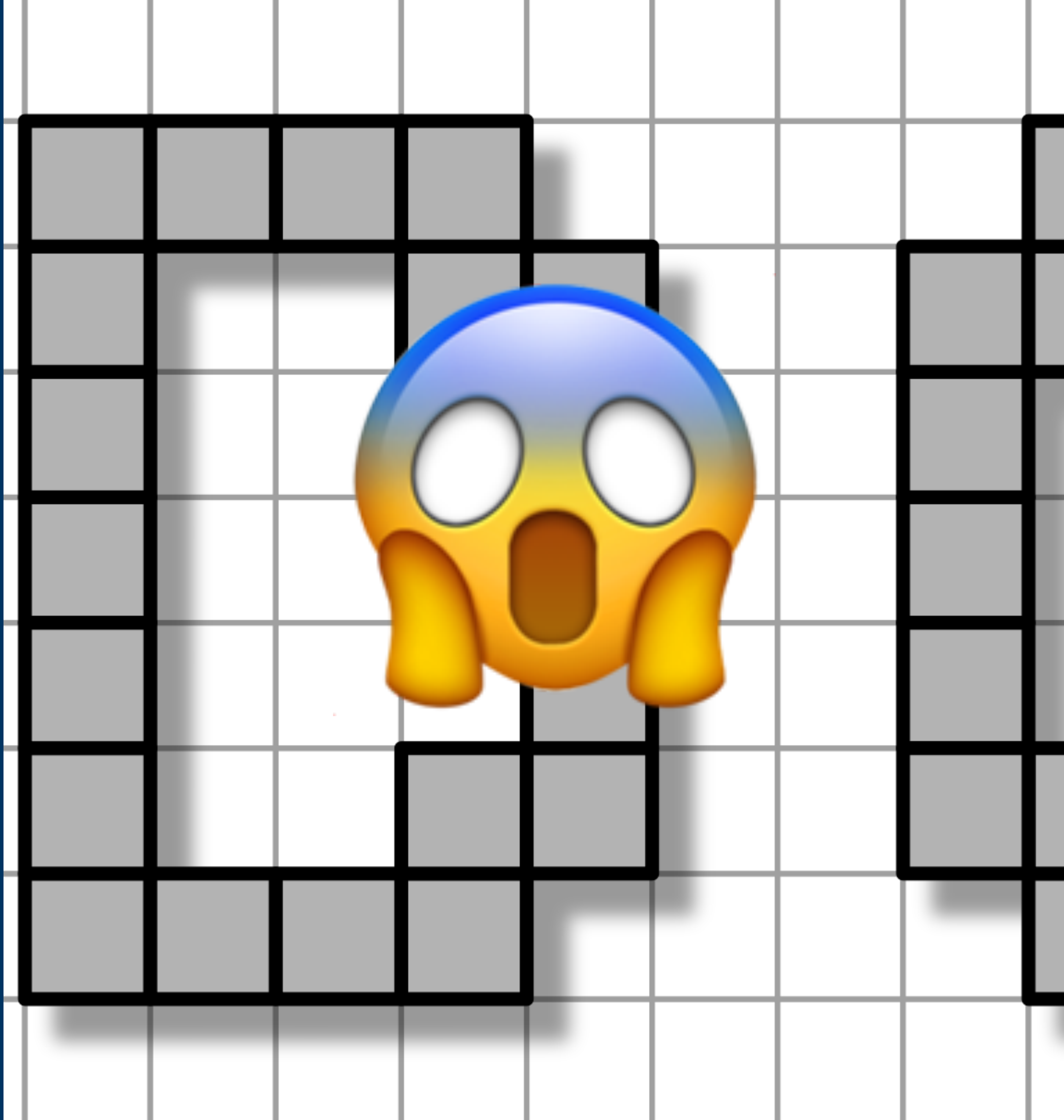
- 3 options:
  - Download DGtal, build the library and install it system-wide (`make install`)
  - Use cmake to fetch DGtal and to get a local build within your project
  - Just clone a DGtal template project

`https://github.com/DGtal-team/DGtal-template`

```
PROJECT(Helloworld)

### Required in DGtal
CMAKE_MINIMUM_REQUIRED(VERSION 3.11)
FIND_PACKAGE(DGtal REQUIRED)
INCLUDE_DIRECTORIES(${DGTAL_INCLUDE_DIRS})
LINK_DIRECTORIES(${DGTAL_LIBRARY_DIRS})

ADD_EXECUTABLE(helloworld helloworld)
TARGET_LINK_LIBRARIES(helloworld ${DGTAL_LIBRARIES})
```

```
project(DGtal-DGMM2022-tutorials)

cmake_minimum_required (VERSION 3.11)
list(APPEND CMAKE_MODULE_PATH ${PROJECT_SOURCE_DIR}/cmake)

set(CMAKE_CXX_STANDARD 11)
set(CMAKE_CXX_STANDARD_REQUIRED ON)

include(dgtal)

include_directories(${DGTAL_INCLUDE_DIRS})
include_directories(${PROJECT_SOURCE_DIR})

add_executable(helloworld helloworld.cpp)
target_link_libraries(helloworld ${DGTAL_LIBRARIES})
```

First steps…



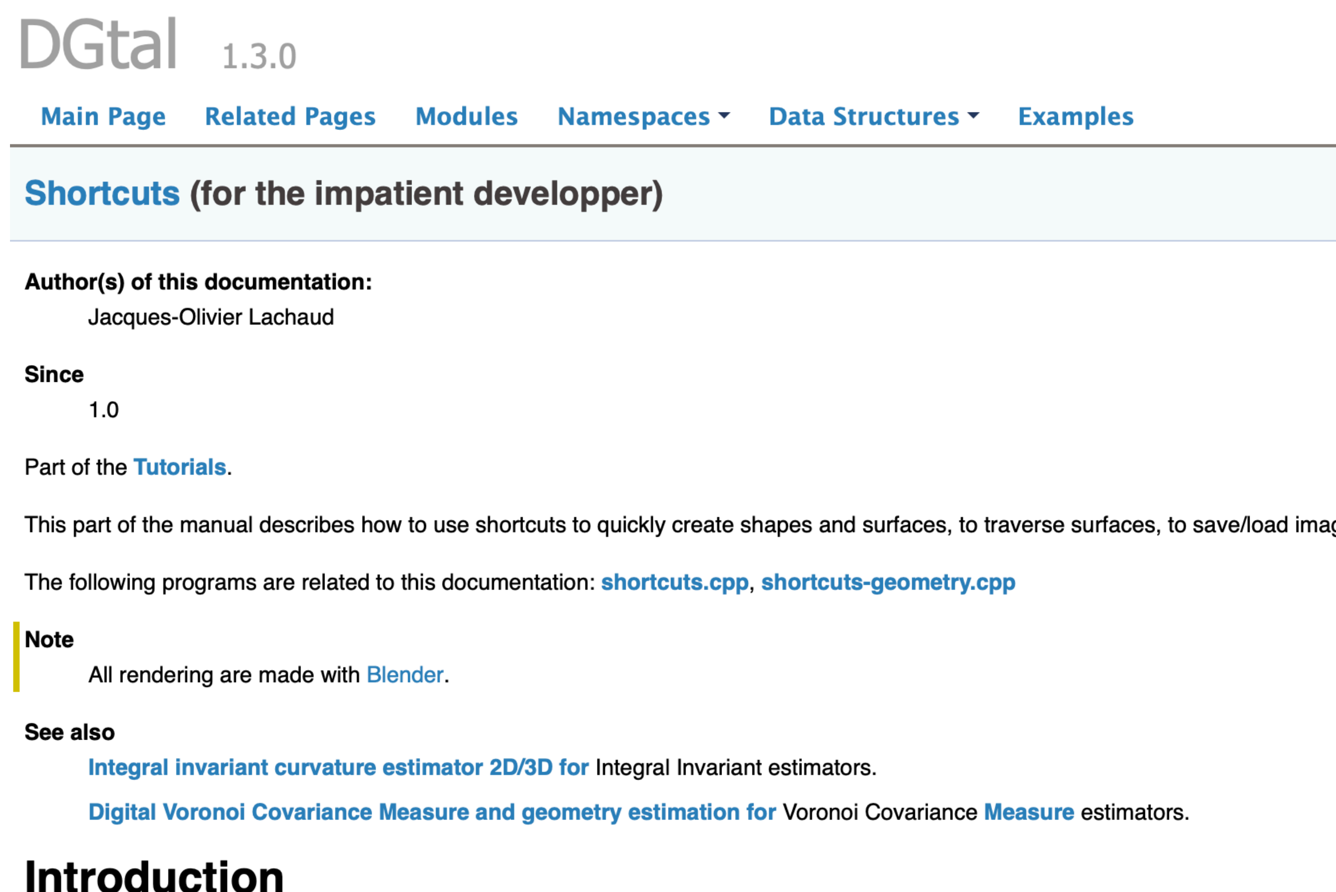
Checklist

- What is the dimension of the space for your problem ?
  - 2 or 3, with classical Integer types ⇒ `Z2i::` and `Z3i::` namespaces
  - Else creates the elementary types you need

fancy CIntegralNumber and CEuclideanRing model

```
typedef int32_t Integer;
typedef DGtal::SpaceND<3, Integer> Space;
typedef DGtal::HyperRectDomain<Space> Domain;
typedef Space::Point Point;
typedef DGtal::DigitalSetBySTLSet<Domain> Set;
```

- Check the shortcuts, the examples (and the related doc!)
- Interact with the authors (Github issues or discussion…)


DGtal 1.3.0

Main Page Related Pages Modules Namespaces Data Structures Examples

Shortcuts (for the impatient developper)

Author(s) of this documentation:
Jacques-Olivier Lachaud

Since
1.0

Part of the Tutorials.

This part of the manual describes how to use shortcuts to quickly create shapes and surfaces, to traverse surfaces, to save/load imag

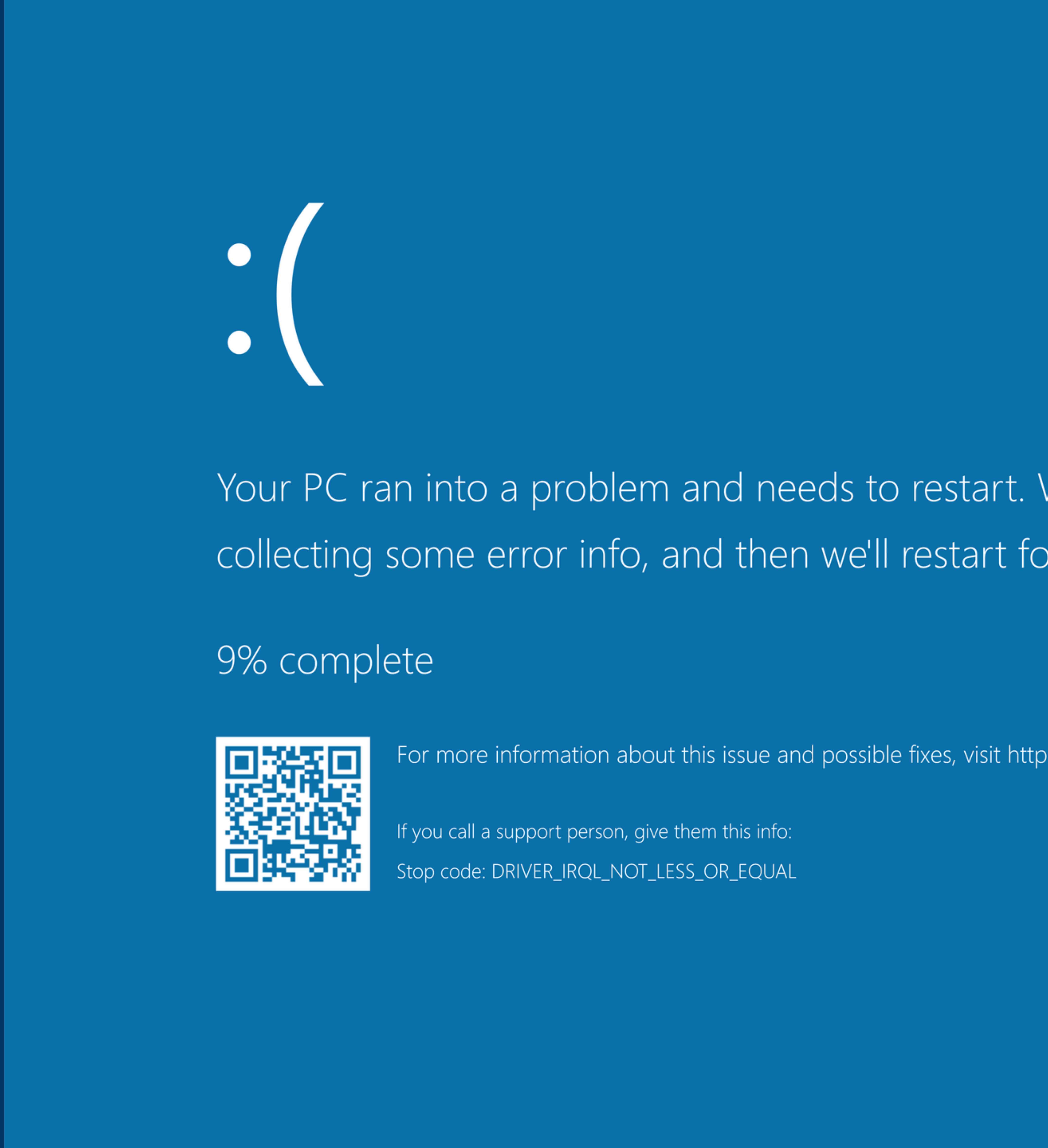
The following programs are related to this documentation: shortcuts.cpp, shortcuts-geometry.cpp

Note
All rendering are made with Blender.

See also
Integral invariant curvature estimator 2D/3D for Integral Invariant estimators.
Digital Voronoi Covariance Measure and geometry estimation for Voronoi Covariance Measure estimators.

Introduction

Live demo


:(
Your PC ran into a problem and needs to restart.
collecting some error info, and then we'll restart
9% complete
For more information about this issue and possible fixes, visit
If you call a support person, give them this info:
Stop code: DRIVER_IRQL_NOT_LESS_OR_EQUAL

Was Gauss right?

Practicals