Digital Geometry Tools and Algorithms

http://libdgtal.org

A software library for the Digital Geometry community

Objectives

- Make **digital geometry easier** for neophyte (student, researcher from another field, ...)
- **Test new ideas**, with objective **comparisons** w.r.t. existing works
- Make the **implementation of demonstrators** easier
- **Spread our research** results to other domains
- Pursue a **federative project**

Main features

- Digital objects in arbitrary dimension
- Algorithms for topological and geometrical analysis
- Image analysis with data structures
- I/O mechanisms and visualization tools

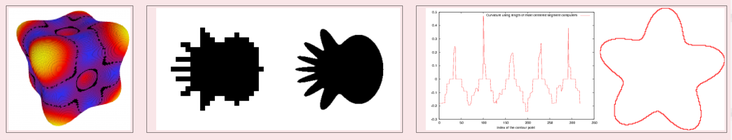
Philosophy

- **Genericity** and **efficiency**
- C++ library, concepts, generic programming with templates
- Open-source, LGPL
- Both user and developer oriented documentation

DGtalTools

Simple and useful tools exploiting the structures and algorithms defined in DGtal

- Converters: pgm2freeman, raw2vol, etc
- Estimators: 2D and 3D local tangent/curvature estimators, length estimators, etc
- Shape generator: multigrid shapes and contours
- Visualisation: 3D vol and mesh viewers, curvature viewer, etc
- Volumetric: marching cubes, ultimate skeleton, subsampling, etc



A collaborative effort

LIRIS — LAMA Laboratoire de Mathématiques Université de Savoie — Loria Laboratoire lorrain de recherche en informatique et ses applications — gipsa-lab — GREYC — IRCyN — cnrs

Lots of features

Kernel package

- Digital spaces, points, vectors, digital domains, digital sets

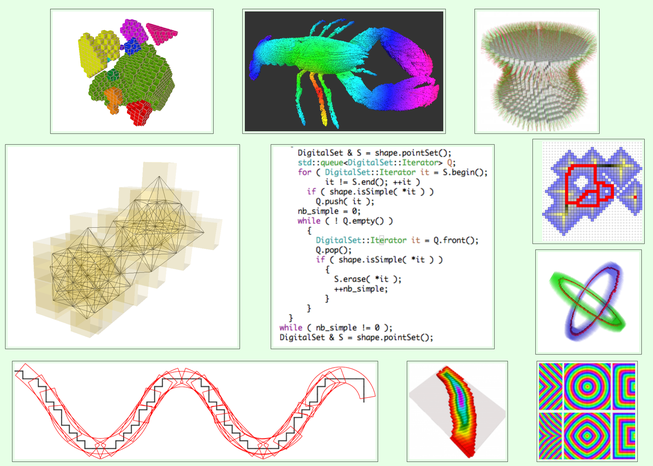
Base package

- Integer types, iterators, utilities, etc

I/O package

- Boards: export to illustrate 2D/3D objects/algorithms (eps,pdf,svg,png, ...)
- Viewers: interactive simple 3D viewer (Qt/QGLViewer)
- Readers/writers for various image formats

Examples



Arithmetic package

- Continued and irreducible fractions, Stern-Brocot tree, DSS patterns

Graph package

- Graph related structures and algorithms (visitors, graph concepts, ...)

Image package

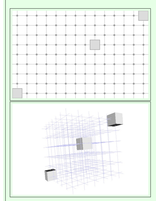
- Image by STL vector (linearized nD image), STL map
- HashTree image container (generalized octree with hashing functions)
- Image adapters

Mathematical package

- Multivariate polynomials

Examples

```
int main()
{
  // define digital space and domain
  typedef DGtal::SpaceND<2, DGtal::int32_t> MySpace;
  typedef MySpace::Point MyPoint;
  typedef HyperRectDomain<MySpace> MyDomain;
  // define points in this domain
  MyPoint p1(-3,-4);
  MyPoint p2(10,4);
  MyPoint p3(5,1);
  MyDomain domain(p1,p2);
  // 2D display
  Board2D board;
  board << domain;
  board << p1 << p2 << p3;
  board.saveSVG("demo-kernel-1.svg");
  board.saveEPS("demo-kernel-1.eps");
  return 0;
}
```



```
int main( int argc, char** argv )
{
  // define digital space, domain and points
  typedef DGtal::SpaceND<3, DGtal::int32_t> MySpace;
  typedef MySpace::Point MyPoint;
  typedef HyperRectDomain<MySpace> MyDomain;
  MyPoint p1( 0, 0, 0 );
  MyPoint p2( 5, 5 ,5 );
  MyPoint p3( 2, 3, 4 );
  MyDomain domain( p1, p2 );
  // 3D visualisation
  QApplication application(argc,argv);
  DGtalQGLViewer viewer;
  viewer.show();
  viewer << domain;
  viewer << p1 << p2 << p3;
  viewer<< DGtalQGLViewer::updateDisplay;
  return application.exec();
}
```

Shape package

- Implicit/parametric shape generator for multigrid analysis

Topology package

- Digital Topology: connectedness, border, simple points
- Cartesian Cellular Topology: cells, surfaces and contours, tracking algorithms

Geometry package

- Primitives: DSS, DCA, digital plane, etc
- Contour analysis: decomposition, convexity, estimators
- Volumetric analysis: area/volume, distance transforms, reverse distance transforms, fast-marching methods.

Examples

```
typedef DepthFirstVisitor<Graph, std::set<Vertex> > DFSVisitor;
typedef GraphVisitorRange<DFSVisitor> VisitorRange;
VisitorRange range( new DFSVisitor( g, Point( -2, -1 ) ) );
n = 0;
for ( VisitorRange::ConstIterator it = range.begin(), itEnd = range.end();
      it != itEnd; ++it, ++n )
  { // Vertex are colored according to their order (depth first order here).
    Vertex vtx = *it;
    board << CustomStyle( specificStyle,
                          new CustomColors( Color::Black,
                                            cmap_hue( n ) ) )
          << vtx;
  }
board.saveEPS("graphTraversal-dfs-range.eps");
```