LAMA
Laboratoire de Mathématiques
Université de Savoie

DGtal: Topology module

Jacques-Olivier Lachaud

Equipe LIMD
Laboratoire de Mathématiques - UMR CNRS 5127
Université de Savoie

DGtal meeting: Jan. 3rd, 2011

DGtal: topology module

Objectives

Basic types and operations for representing a cartesian digital space equipped with a digital topology, and objects lying in this space.

- Arbitrary adjacencies in $\mathbb{Z}^n$, but also in subdomains
- Digital topology = couple of adjacencies (Rosenfeld)
- Object = Topology + Set
- Operations: neighborhoods, border, connectedness and connected components, decomposition into digital layers, simple points

Adjacency

Genericity ⇒ concept `CAdjacency`

- Types: `Space`, `Point`, `Adjacency`
- Methods:
  - `isAdjacentTo`(p1, p2)
  - `isProperlyAdjacentTo`(p1, p2)
  - `writeNeighborhood`(p, outit)
  - `writeProperNeighborhood`(p, outit)
  - `writeNeighborhood`(p, outit, pred)
  - `writeProperNeighborhood`(p, outit, pred)
- Models:
  - `MetricAdjacency`: 4-, 8-, 6-, 18-, 26-, $2n$-, $3^n - 1$-adjacencies
  - `DomainAdjacency`: adjacency limited by a specified domain.

Digital topology

Digital topology = couple of instances of adjacencies

- template class `DigitalTopology`

```
typedef SpaceND< 3,int > Z3;
typedef MetricAdjacency< Z3, 1 > Adj6;
typedef MetricAdjacency< Z3, 2 > Adj18;
typedef DigitalTopology< Adj6, Adj18 > DT6_18;

Adj6 adj6;
Adj18 adj18;
DT6_18 dt6_18( adj6, adj18, JORDAN_DT );
```

- Jordan topologies may be specified (for future use)
- instances are necessary (e.g., adj may not be invariant by translation)
- reverse topology is the reversed couple

Digital Object

Digital object = topology + digital set

- template class `Object`

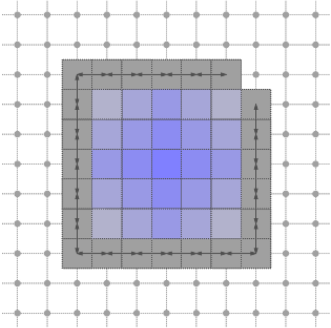
```
typedef HyperRectDomain< Z3 > Domain;
typedef DigitalSetSelector<Domain, BIG_DS+HIGH_BEL_DS>::Type DigitalSet;
typedef Object<DT6_18, DigitalSet> ObjectType;
Point p1( -50, -50, -50 ); Point p2( 50, 50, 50 );
Domain domain( p1, p2 );
// ball of radius 30
DigitalSet ball_set( domain );
Shapes<Domain>::addNorm2Ball( ball_set, Point( 0, 0 ), 30 );
ObjectType ball_object( dt6_18, ball_set );
```

- Objects may be passed by value and copied without cost
- Methods:
  - neighborhoods, border, geodesic neighborhoods are objects
  - (lazy) connectedness, connected components
  - simple points (in Z2 and Z3)

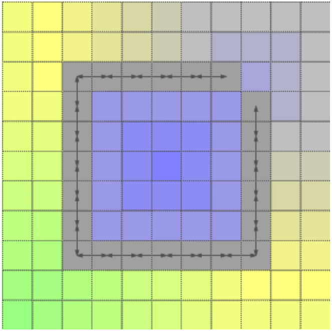
Expander: digital layers in an object

- Expansion layer by layer within an object, starting from an initial core
- core = a point or a pointset specified by iterators
- each new layer = the set of points of the object adjacent to the preceding layer
- each layer is iterable, has a digital distance to core
- finished when no more neighbor expansion is possible
- useful for connectedness

Expander: digital layers in an object



background in 4-adj



background in 8-adj

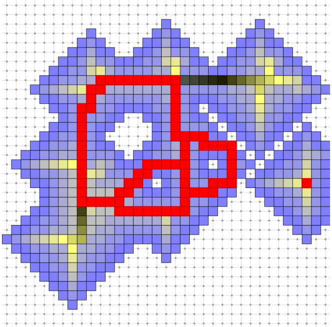
`tests/topology/testSimpleExpander.cpp`

Example: greedy homotopic thinning

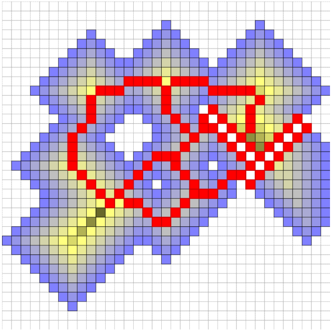
```
int layer = 0;
do {
    DigitalSet & S = shape.pointSet();
    std::queue<DigitalSet::Iterator> Q;
    for ( DigitalSet::Iterator it = S.begin(); it != S.end(); ++it )
      if ( shape.isSimple( *it ) )
        Q.push( it );
    nb_simple = 0;
    while ( ! Q.empty() ) {
      DigitalSet::Iterator it = Q.front();
      Q.pop();
      if ( shape.isSimple( *it ) ) {
        S.erase( *it );
        ++nb_simple;
      }
    }
    ++layer;
} while ( nb_simple != 0 );
```

See `testObject.cpp`

Example: greedy homotopic thinning



thinning in (4,8)



thinning in (8,4)

`tests/topology/testObject.cpp`

Conclusion and perspectives

- complete Rosenfeld's approach: curves and separation
- whole digital topology framework of Herman and Udupa
  - digital surface as a couple of ω-adjacent points
  - immediate interior and exterior, interior and exterior
  - $\kappa\lambda$-borders, $\kappa\lambda$-boundaries
  - digital pictures
- interpixel topology or cartesian cellular grid topology

See on-line doc.: Digital topology and digital objects