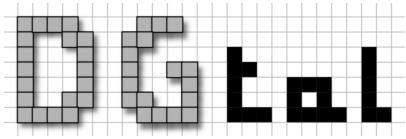

DGtal: Introduction to DGtal Kernel

http://liris.cnrs.fr/dgtal

D. Coeurjolly

Package description

Should contain

- Fundamental objects and methods to define a topological and geometric structure on $\mathbb{Z}^d$

Examples

- Digital space and domains definitions
- Integer types (unitary ring)
- Point & Vector
- Linear Algebra
- Digital sets
- ...

Location

- `{DGtal}\src\DGtal\kernel\`
- `{DGtal}\tests\DGtal\kernel\`

DGtal code skeleton

Things to do

1. Fix the dimension
2. Fix the Integer type (commutative ring (+,-,*))
3. Define the digital space DGtal::SpaceND

```
1    #include "DGtal/base/Common.h"
2    #include "DGtal/kernel/SpaceND.h"
3    {...}
4    typedef DGtal::int32_t Integer;
5    typedef DGtal::SpaceND<6, Integer> Space6;
6
7    typedef mpz_class IntegerGMP;
8    typedef DGtal::SpaceND<6, IntegerGMP> Space6GMP;
```

Q: what's wrong with ?

```
1  typedef DGtal::SpaceND<2, unsigned char> MySpaceUChar;
```

[DETAILS] Concept & Models

Answer

`unsigned char` does not define a ring !

Constraints on types and template parameters are defined with **Concepts**

`Integer` in `SpaceND` should be a model of `DGtal::CCommutativeRing`.

Concept Checking with `boost`

```
1    ...
2    //Integer must be signed to characterize a ring.
3    BOOST_CONCEPT_ASSERT(( CCommutativeRing<TInteger> ) );
4    ...
```

Digital Space

Types

```
1          Integer
2          Point
3          Vector
4          RealPoint
5          RealVector
6          Subspace
7          Subcospace
```

+

```
1          static Dimension dimension
```

Point/Vector...

Point/Vector in a $d-$Dimensional DGtal space.

- arithmetic operators (*, -, ...)
- comparison operators (< ,>, ...)
- methods associate to the canonical lattice associated to points (inf, sup, isLower,...)
- methods to compute various norms of Points/Vectors.

E.g.

```
1   #include "DGtal/base/Common.h"
2   #include "DGtal/kernel/SpaceND.h"
3   {...}
4   typedef DGtal::int32_t Integer;
5   typedef DGtal::SpaceND<2, Integer> Space2;
6   typedef Space2::Point Point2;
7
8   Point2 p(12, -34);
9   Point2 q(2, -2);
10  if (p < q)
11   ...
```

StdDefs.h: “Standard” digital spaces

Shortcuts with `StdDefs.h` with namespaces `Z2i` and `Z3i`.

```
1       #include "DGtal/base/Common.h"
2       #include "DGtal/utils/StdDefs.h"
3       {...}
4
5       DGtal::Z2i::Point p(12, -34);
6       DGtal::Z2i::Point q(2, -2);
7       if (p < q)
8       ...
```

```
1       #include "DGtal/base/Common.h"
2       #include "DGtal/utils/StdDefs.h"
3       {...}
4
5       DGtal::Z3i::Point p(12, 2, -34);
6       DGtal::Z3i::Point q(2, 0, -2);
7       if (p < q)
8       ...
```

Domains

Short description

Defines a subset of $\mathbb{Z}^d$ which we are working on.

- a domain is parametrized by a specific SpaceND type
- must implement various `Iterators` to scan the domain points

Example:

```
1    #include "DGtal/base/Common.h"
2    #include "DGtal/helpers/StdDefs.h"
3    #include "DGtal/kernel/domains/HyperRectDomain.h"
4    {...}
5
6    typedef HyperRectDomain<Z2i::Space> MyDomain;
7    Z2i::Point a(-3,-4);
8    Z2i::Point b(10,4);
9    MyDomain domain(a,b);
```

More details later...

Digital Sets

Short description

Define sets points in a given domain.

- several types of container (STL vector, STL set,...) which can be selected via a `DigitalSetSelector`
- must implement methods to add/remove points
- must implement `Iterators` to scan the points
- ...

Example:

```
1    Z2i::Point p1(  -10, -10  );
2    Z2i::Point p2(  10, 10  );
3    Z2i::Domain domain( p1, p2 );
4    typedef DigitalSetSelector < Domain, BIG_DS + HIGH_ITER_DS + ↩
         HIGH_BEL_DS >::Type SpecificSet;
5    SpecificSet mySet( domain );
6
7    Z2i::Point c(  0, 0  );
8    mySet.insert( c );
9    Z2i::Point d(  5, 2  );
10   mySet.insert( d );
```

Images in DGtal

Idea

Mapping between Domain points and and values
Models are parametrized by a Domain type and a Value type
IO with readers, writers and SVG/PDF exports

Several image containers:

- ImageContainerBySTLVector: linearization of nD domains
- ImageContainerBySTLMap: (point,value) map
- ImageContainerByHashTree: hierarchical nD-tree with geometric hashing functions.

but also:

- ImageContainerByITKImage: use ITKImage in DGtal

Key ideas

Type Inclusion

$$\{\text{dimension, Integer}\} \rightarrow \text{SpaceND} \rightarrow \text{Domain} \rightarrow \text{DigitalSet}$$

Concept checking

StdDefs

Visualisation

Kernel objects are DGtal stream compliant (2D and 3D).

e.g.

```
1   Point a,b;...
2   Domain domain(a,b);...
3   DigitalSet set(....); ...
4
5   Board2D board;
6   board << domain
7         << set
8         << a
9         << b;
10
11  board.saveSVG("myset.svg")
```