Pandore 6

Tutorial

GREYC laboratory – IMAGE team
Caen , France

General features

GREYC IMAGE ENSICAEN

- Pandore is 2 purposes ...
  1. A collection of operators
  2. A programming environment
- ... related to 2 viewpoints
  1. Prototyping application
     - Application = sequence of tunable operators
  2. Programming application
     - Application = C++ program
- Independent from any other software or proprietary library

1- Prototyping

GREYC IMAGE ENSICAEN

- Operators = executable commands
  - Command interpreter (MsDOS, bash, ksh)
- Images = Pandore files
  - Image, Regions map, Graph, Collection
  - *Conversion tools from and to other image format*
    - *Ex: pbmp2pan, pgif2pan, ppan2bmp*
- Application = sequence of operators
  - *image* --operator--> *image* --operator--> *image* ...

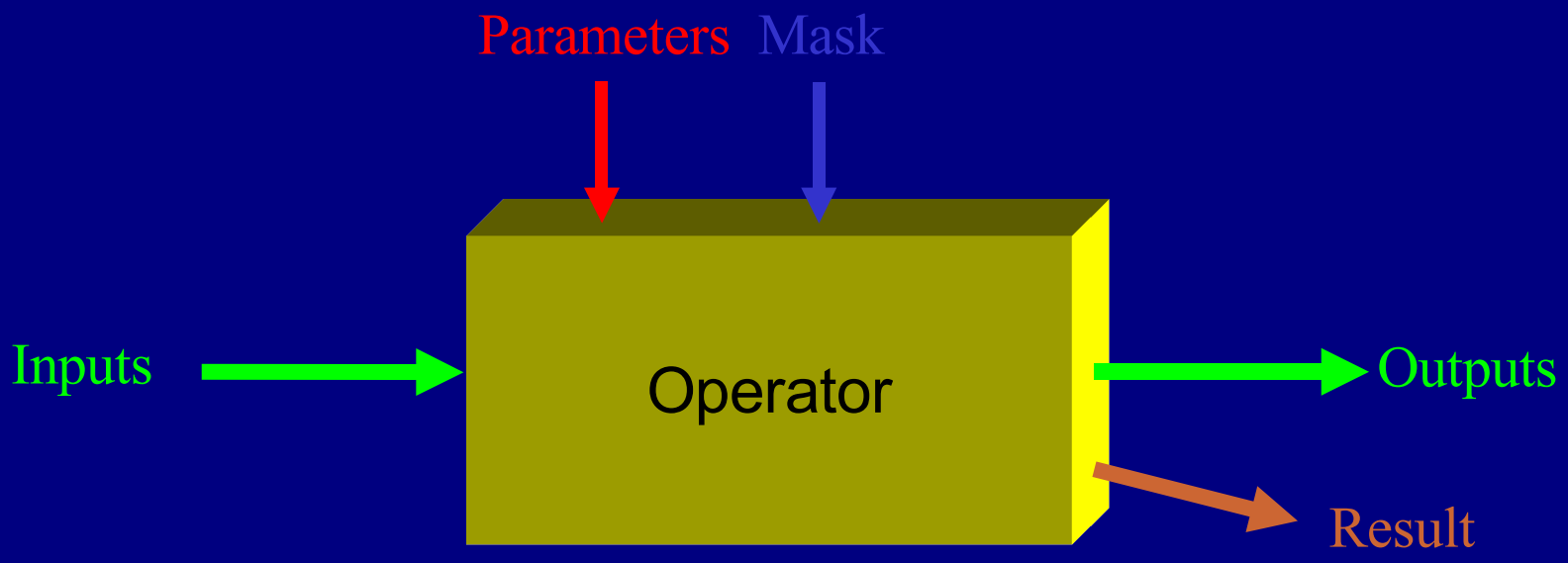
Model of operator

- Generic synopsis

```
operator parameter* [-m masque] [im_src|-]* [im_dest|-]*
```

- Examples

```
pdilatation 1 2 a.pan b.pan
```



a- Parameter

- Used to tune the operator
- Given in exact number defined by the operator even if some parameters are not used
- Types
  - Integer, Real, Character, String
- Examples

```
pdilatation 1 2 a.pan b.pan
pcolsetvalue length float 14.5 col.pan
```

b- Input and output

GREYC IMAGE ENSICAEN

- Files
  - Pandore files: Image, Region map, Graph, Collection
- Named or standard input ('-')
  - Allows sequence, pipe, redirection
- Example (black top hat)
  - `pdilatation 1 1 a.pan a1.pan;`
    `perosion 1 1 a1.pan a2.pan;`
    `pdif a2.pan a.pan b.pan`
  - `pdilatation 1 1 a.pan | perosion 1 1 | pdif - a.pan b.pan`
- Type of Pandore file: the command pfile'
  - `pfile  tangram.pan`

```
Creator : zeus
Date    : 2005/11/01
Type    : Img2duc (gray 2d image of char)
Size    : 256 rows x 256 cols
```

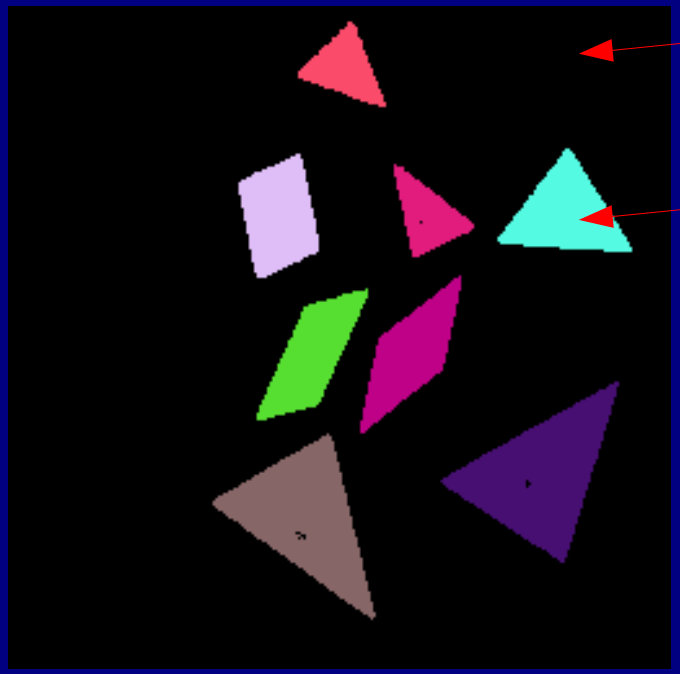
Image

GREYC IMAGE ENSICAEN

- Pixel image
- 27 image types
  - Band:
    - Gray level (number band =1)
    - Color (number band =3)
    - Multispectral (number band >=1)
  - Dimension
    - 1d, 2d and 3d.
  - Pixel type
    - Uchar (pixel values in [0 ; 255])
    - Slong (pixel values in [-2,147,483,648 ; 2,147,483,647])
    - Float (pixel values in [1.175494351E-38F ; 3.402823466E+38F])
  - Examples:
    - 3d, multispectral image of Uchar (named Imx3duc)
    - 2d, color image of Float (named Imc2dsf)

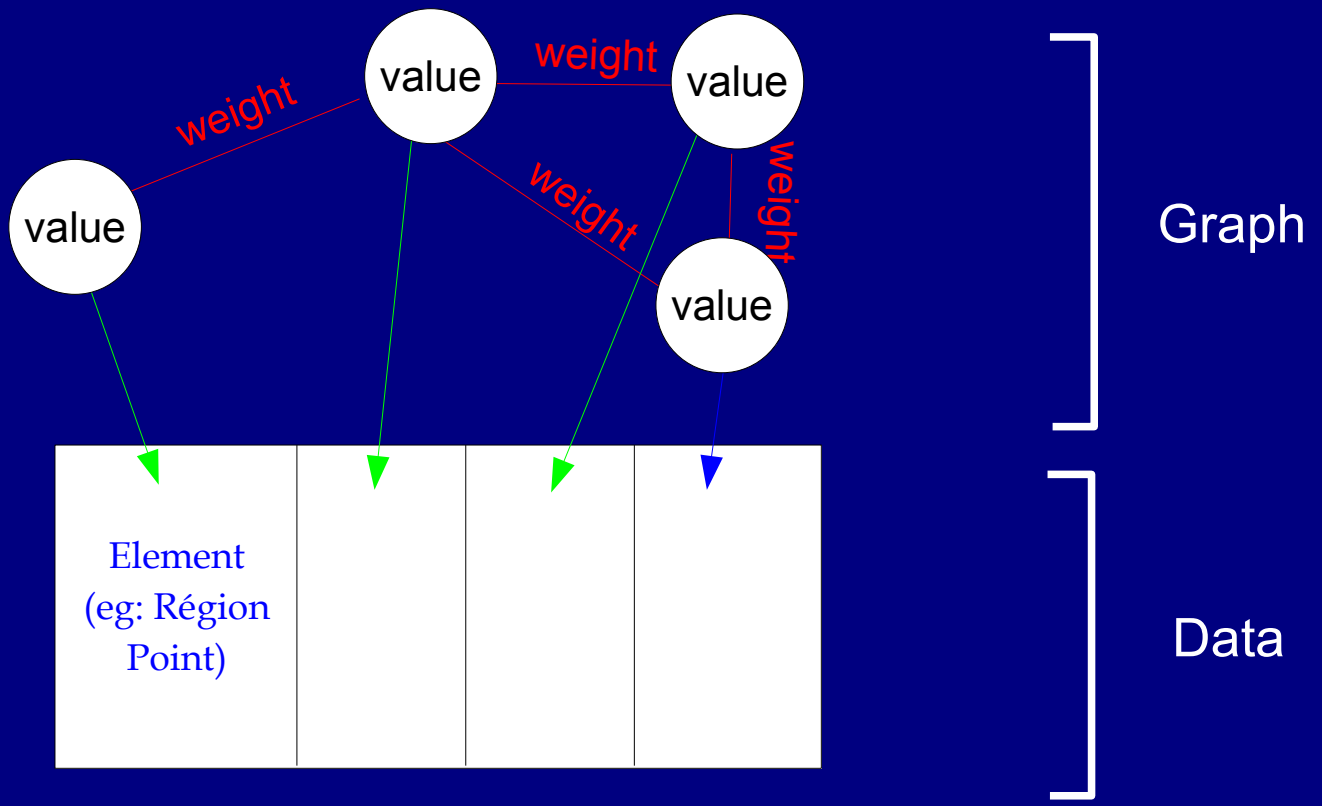
Regions map

- Labels image
  - Ulong (label values in [0 ; 4,294,967,295] regions
    - Label =0 : no region
    - Label > 0 : a region number
- 3 types
  - 1d, 2d and 3d region maps



Graph

- Weighted, non directed graph
- Separation between graph and data
  - Graph of everything (regions, points)
  - Need to attach the region map or collection with the graph



Collection

GREYC IMAGE ENSICAEN

- Container of heterogeneous data
  - Access by name
  - Available values
    - Primitive types: integer, real, string
    - Array of primitives types
    - Pandore types: point, dimension, image, regions map, graph, collection
    - Array of pandore type

name1	value1
name2	value2
name3	value3

c- Result

GREYC IMAGE ENSICAEN

- Type
  - Primitive value : Integer, Real, character.
  - SUCCESS or FAILURE
- Access: the command '`pstatus`'
  - Example (binarization from image mean value)
    - SHELL

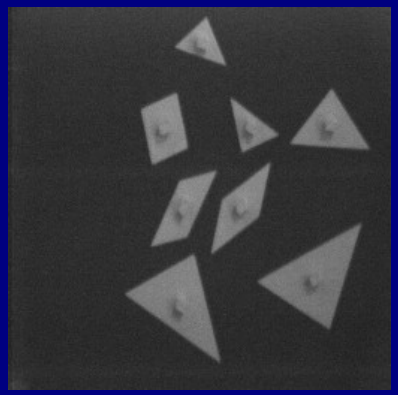
```
pmeanfiltering a.pan
x=`pstatus`
pbinarization $x 1e10 b.pan c.pan
```

    - MSDOS

```
pmeanfiltering a.pan
call pstatus
call pset x;
pbinarization %x% 1e30 b.pan c.pan
```

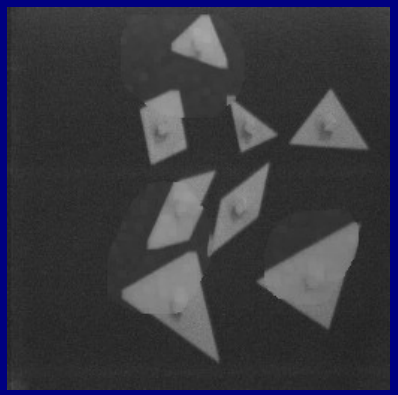
d- Mask

- Optional and introduced by -m <mask>
- Mask is a region map
  - Without mask, operator is applied on whole image;
  - With mask, operator is applied on pixel where region label > 0.
- Examples
  - `pim2rg a.pan r.pan`
  - `pdilatation 1 2 -m r.pan tangram.pan b.pan`



tangram.pan

r.pan



b.pan

Help

GREYC IMAGE ENSICAEN

- In line
  - <operator> -h: displays the usage.
    - usage: pdilatation num_se halfsize [-m mask] [im_in|-] [im_out|-]
  - <operator> -p: displays the prototype (name, number of parameter, input and output)
  - pman <operator>: prints out manual page for the specified operator (*a la* "man").
  - pman -k <keyword>+: prints out the list of operators that are related to the given keyword(s).
  - pman -M <path> <operator>: specifies an alternate search path for manual pages.
- Documentation
  - Manual of operators (doc/index_operatorsP0.html)

Encoding applications

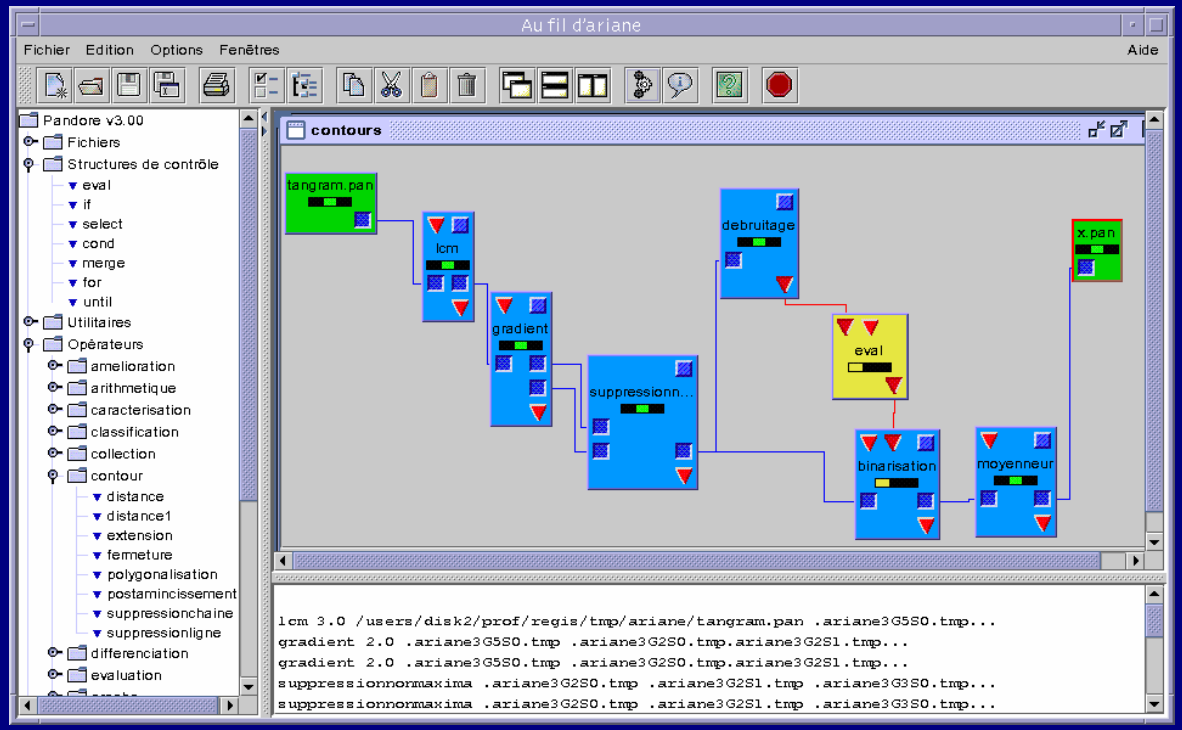
GREYC IMAGE ENSICAEN

```
#/bin/sh
#ASFrec: Alternate Sequential Filter by
#reconstruction type OC (Open-Close)
#parameters: se n in.pan out.pan

if [ $* -gt 4 ]; then exit 1; fi
cp $3 out.pan
se=$1; n=$2
i=1
while [ $i -le $n ]; do
  perosion $se $i out.pan i1.pan
  pdilatationreconstruction 8 i1.pan out.pan i2.pan
  pdilatation $se $i i2.pan i3.pan
  perosionreconstruction 8 i3.pan i2.pan out.pan
  i=`expr $i + 1`
done
cp out.pan $4
```

```
rem echo off
rem ASFrec: Alternate Sequential Filter by
rem reconstruction type OC (Open-Close)
rem parameters: se n in.pan out.pan

if "%4" == "" goto :ERROR
set n=%1
set se=%2
copy %3 out.pan
FOR /L %%i IN (1,1,%n%) DO (
  perosion %se% %%i out.pan a.pan
  pdilatationreconstruction 8 a.pan out.pan b.pan
  pdilatation %se% %%i b.pan c.pan
  perosionreconstruction 8 c.pan b.pan out.pan
)
copy out.pan %4
:ERROR
```

Using Ariane

- Visual Programming Interface
  - Automatic generation of Perl program



(www.greyc.ensicaen.fr/~regis/Ariane)

Basis paradigms

GREYC IMAGE ENSICAEN

- Atomic operator = reusing chunk
  - One operation;
  - Control problem solved inside the operator;
  - No interchangeable part.
- Operator = element of a language (*a la* C)
  - Conversion operators
    - pbmp2pan, ppan2d23d, im2txt
  - Casting operators
    - pim2uc, pimc2img, pim2rg
  - Arithmetic operators
  - Logical operators
  - Set operators
  - Image processing operators

Operators categories

GREYC IMAGE ENSICAEN

- Conversion
- Casting
- Color
- Arithmetic
- Logic
- Transformation
- Filtering
- Frequency
- Morphology
- PointOfInterest
- EdgeDetection
- Contour
- Thresholding
- Segmentation
- Region
- RegionFeatureExtraction
- ImageFeatureExtraction
- Evaluation
- Array
- Graph
- Collection
- Classification
- Visualization
- Information
- Motion
- Miscellaneous

2- Programming

GREYC IMAGE ENSICAEN

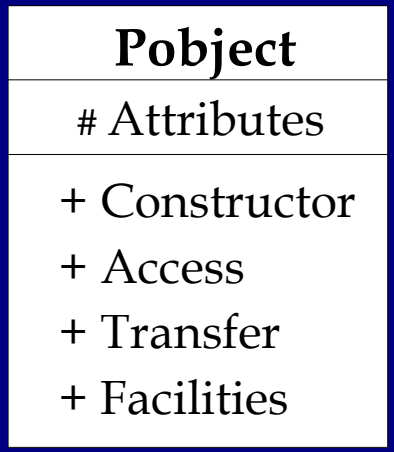
- Language C++
  - Pandore objects
  - Genericity
  - Basic programming
- Primitive types

  (*Redefined to be independent from any host architecture*)
  - Signed value: Char (1b), Short (2b), Long (4b), Float (4b), Double (>4b)
  - Unsigned value: Uchar, Ushort, Ulong
  - Error value: **Errc** (primitive types + FAILURE + SUCCESS)

```
Errc result=SUCCESS;
if (result == FAILURE) ...
result = (Uchar)12;
```

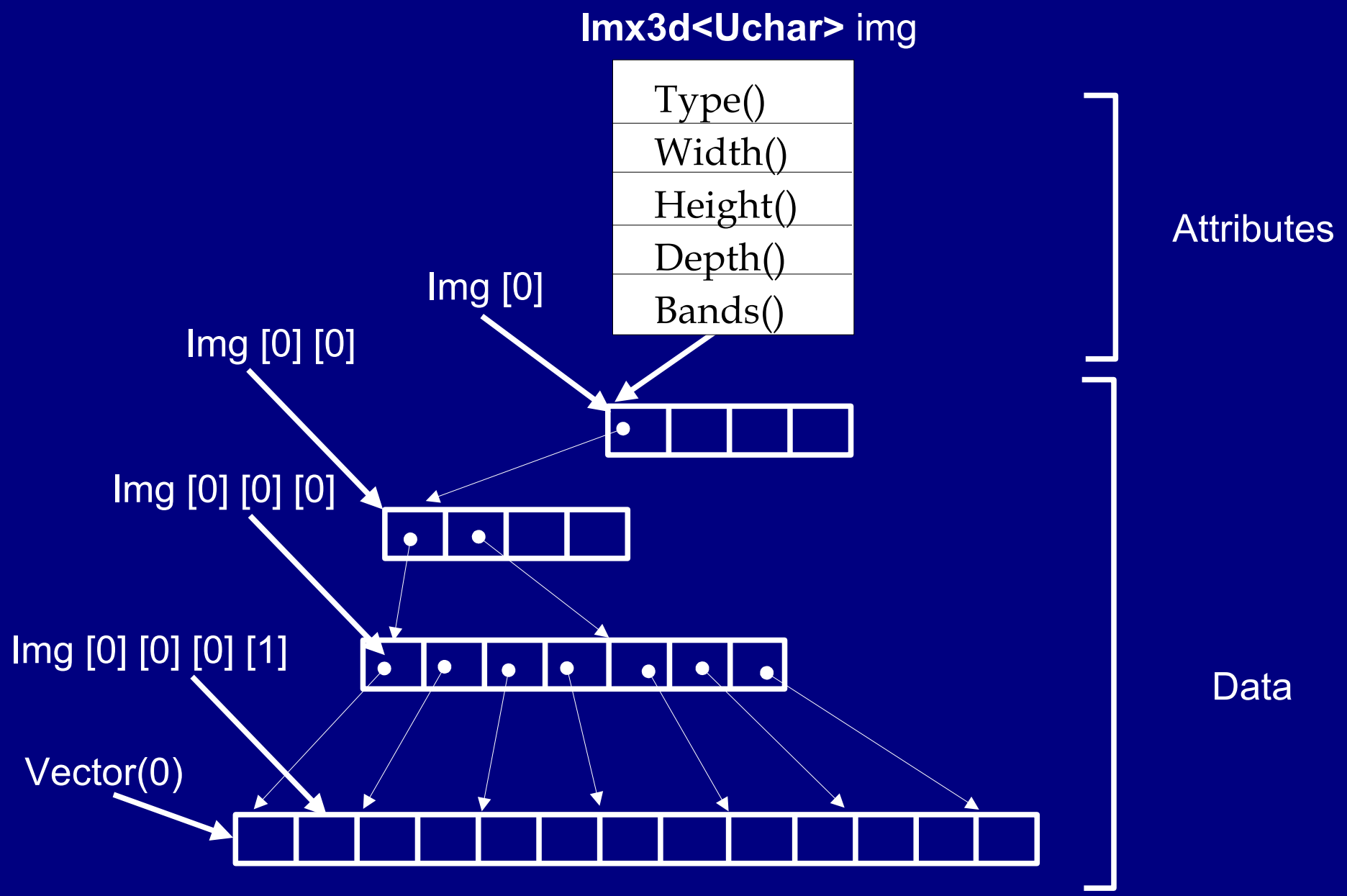
Abstract Pandore object

- Pobjects: the base class
  - Attributes: (protected)
  - Methods: (public)
    - Constructor
    - Access: data & attributes
    - File transfer
    - Facilities



- Pandore objects (inherit from Pobject)
  - Image
  - Graph
  - Point (Long x,y,z)
  - Region map
  - Collection
  - Dimension (Long w,h,d)

Image


Imx3d<Uchar> img
Type()
Width()
Height()
Depth()
Bands()
Attributes
Img [0]
Img [0] [0]
Img [0] [0] [0]
Img [0] [0] [0] [1]
Vector(0)
Data

Image

- Types
  - Name: Im[x,g,c][1d,2d,3d][uc, sl, sf]
    - x: multispectral, g: gray levels, c: color
    - uc: Uchar, sl: Slong, sf: Float
- Attributes
  - Width(), Height(), Depth(), Size(), Props()
  - Band()
  - ColorSpace() (for Imc only)
- Construction
  - Img2duc im(12,10); !*Caution: no initialisation to 0*.
  - Img2duc im; im.New(12,10); Img2duc *im = new Img2duc(12,10);
  - Uchar* *data*=malloc(12*10*sizeof(Uchar));
    Img2duc im(12,10,*data*): Allocation with a predefined vector

Image

- Remark
  - There is only one real type: `Imx3d<T>`
    - T in Uchar, Slong and Float.
  - All other types are rewriting of this type
    - Gray image = Imx with 1 band
    - Color image = Imx with 3 bands.
    - 2d image = Imx with 0 plane;
  - Operator: Only 1 function for the type Imx3d<T>
    - Example:

```
template <typename T>
Errc PAdd(Imx3d<T> &ims1, Imx3d<T> &ims2, Imx3d<T> &imd) {
   for (int b=0; b<ims1.Bands();b++) {
      T *ps1=ims1.Vector(b); T *ps2=ims2.Vector(b); T *pd=imd.Vector(b);
      for (;ps1<ims1.Vector(b)+ims1.VectorSize();ps1++,ps2++,pd++)
          *pd = (T)(((Float)*ps1+(Float)*ps2)/2.0F);
   }
   return SUCCESS;
}
```

Image

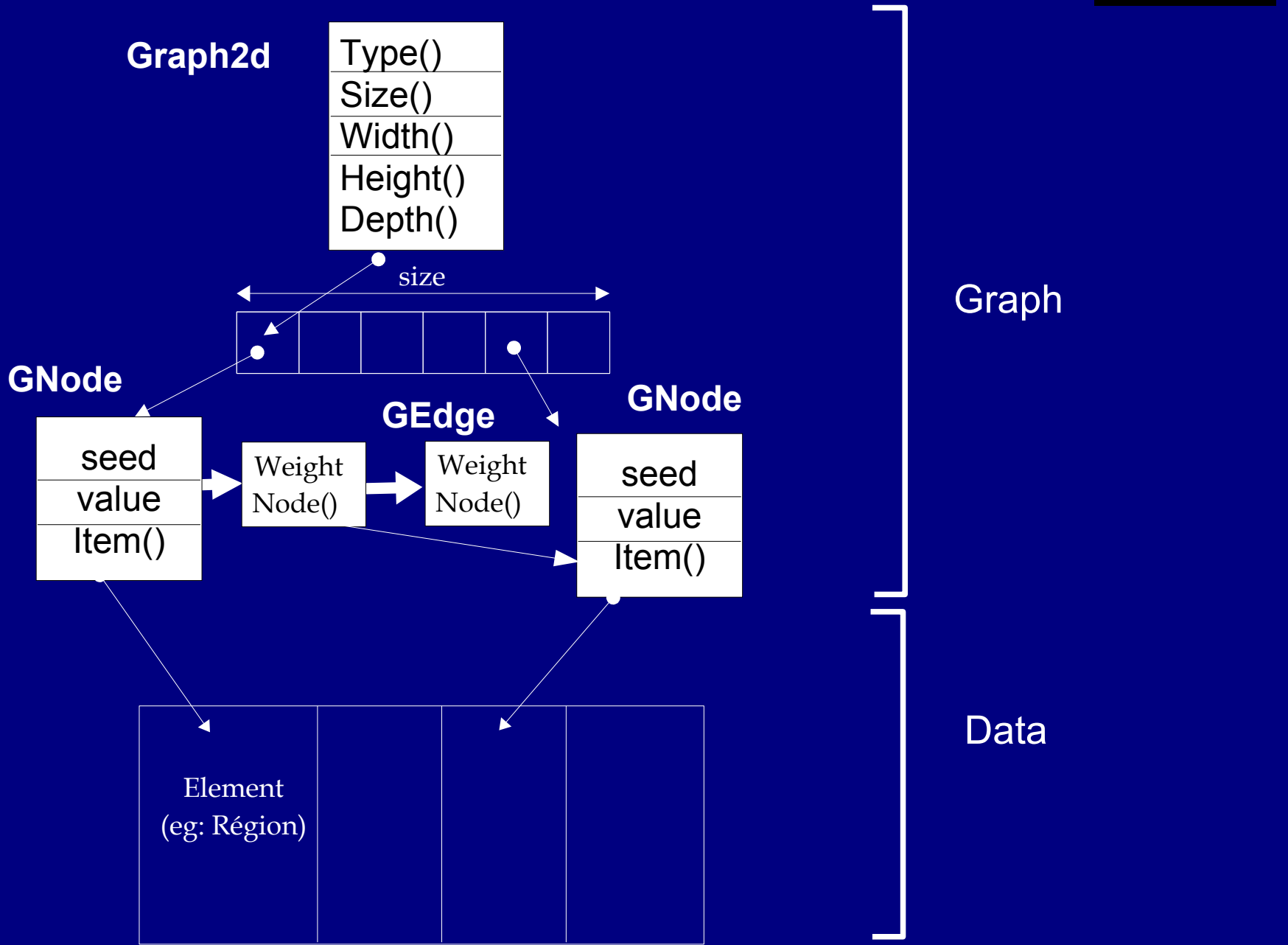
- Pixel access: operator[]
  - Long x,y; img2d: img[y][x], imc2d: imc.X[y][x], Imx2d: imx[b][y][x]
  - Point2d p; img2d: img[p], imc2d: imc.X[p], Imx2d: imx[b][p]
- Vector access: method Vector()
  - type* p=img.Vector(); type* p=imx.Vector(b);
- Neighbors access: predefined arrays v[]
  - Freeman codes (*2d: 4,8-connexity; 3d: 6,26-connexity*)
    - v4x[n], v8y[n] (x,y coordinates); v8[n] (Point2 p); v26[n] (Point3d)
  - Example

```
Point2d p;
for (p.y=1;p.y<im.Height()-1;p.y++)
   for (p.x=1;p.x<im.Width()-1;p.x++) {
       for (int n=0;n<8; n++)
            s+=im[p+v8[n]]
       im[p]=s/8.0F;
   }
```

Regions map

- Region map
  - Ulong image (~4 thousand regions)
- Types
  - Name : Reg(1d,2d,3d)
- Attributes (cf. Ulong image)
  - Labels(): Returns the maximum label.
  - Labels(x): sets and returns the maximum label.
- Construction
  - Reg2d rg(12,10); !*Caution: no initialization to 0*
  - Reg2d() rg; rg.New(12,10); rg.Labels(12);
- Access
  - (idem Ulong image)

Graph


Graph2d
Type()
Size()
Width()
Height()
Depth()
size
GNode
GEdge
GNode
seed
value
Item()
Weight
Node()
Weight
Node()
seed
value
Item()
Element
(eg: Région)
Graph
Data

Graph

GREYC IMAGE ENSICAEN

- Graph
  - Separation graph / elements (annexed array of elements)
- Types
  - Name : Graph(1d,2d,3d)
- Attributes
  - Width(), Height(), Depth(), ImageSize(), Props()
  - Size(): Number of nodes
- Construction
  - Graph2d gr(nodes); Graph2d gr(nodes, depth, height, width);
  - Init(Reg2d): Initialization from a regions map.
- Access
  - Add(s), Del(s): add/delete node s.
  - Link(s1,s2,weight), Unlink(s1,s2): Link 2 nodes s1 and s2.
  - gr[i].Neighbours(): get the list of neighbours

Collection

- Collection
  - Container of heterogeneous data (*a la* struct)
- Type
  - Name: Collection
- Available data type
    - Primitive types
    - Array of primitive types
    - Pandore objects
    - Array of Pandore objects

Collection

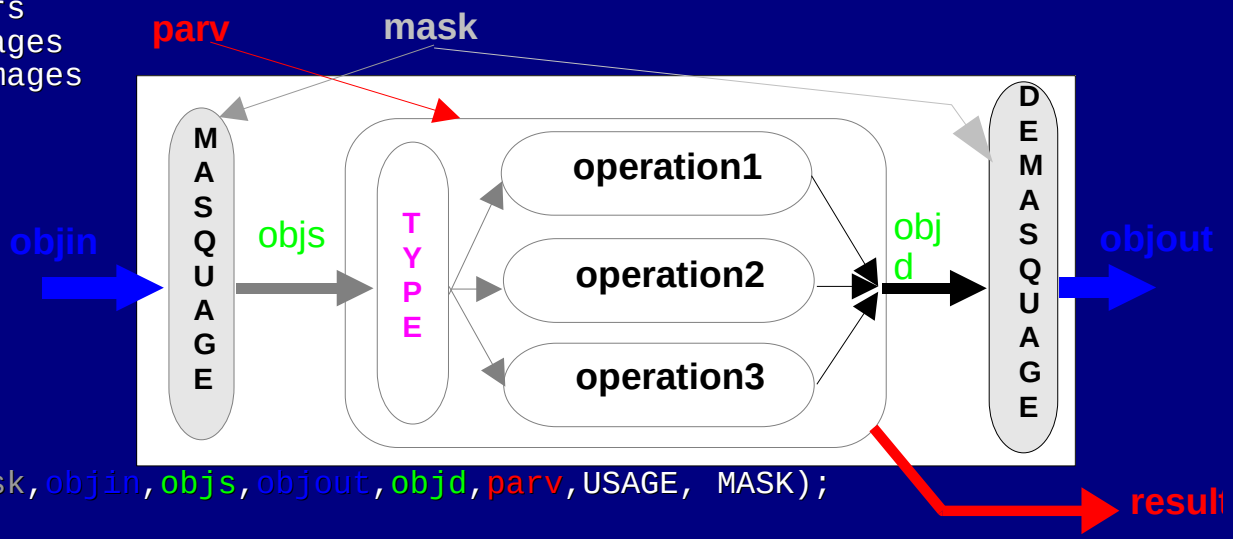
GREYC IMAGE ENSICAEN

- Access: Data are referenced by name.
  - SETVALUE(name, type, value);
  - SETARRAY(name, type, pointer_to_array, number_of_items);
    - *Caution: No copy of array*
  - SETPOBJECT(name,type,pointer_to_object);
    - *Caution: No copy of the object*
  - SETPARRAY(name,type,pointer_to_object,number_of_items).
    - *Caution: No copy of array*

  - GETVALUE(name,type);
  - GETARRAY(name,type) + GETARRAYSIZE(name,type);
  - GETPOBJECT(name,type);
  - GETPARRAY(name,type) + GETPARRAYSIZE(name,type).

Template: Main function


```
#ifdef MAIN
#define USAGE   "USAGE : %s parameter [-m mask] [im_in|-] [im_out|-]\n"
#define PARC    1  // Number of parameters
#define FINC    1  // Number of input images
#define FOUTC   1  // Number of output images
#define MASK    1  // Masking operation

int main(int argc,char* argv[]) {
  Errc result;
  Pobject *mask;
  Pobject *objin[FINC+1];
  Pobject *objs[FINC+1];
  Pobject *objd[FOUTC+1];
  Pobject *objout[FOUTC+1];
  char *parv[PARC+1];

  ReadArgs(argc,argv,PARC,FINC,FOUTC,&mask,objin,objs,objout,objd,parv,USAGE, MASK);

  Img2duc* const ims=(Img2duc*)objs[0];
  objd[0]=new Img2duc(ims->nrow,ims->ncol);Img2duc* const imd=(Img2duc*)objd[0];

  switch(objs[0]->Type()) {
      case Img2duc: result=Operation1((Img2duc)*(objs[0]),(Img2duc)*(objd[0]),(Short)parv[0]);break;
      case Img2dsl: result=Operation2((Img2dsl)*(objs[0]),(Img2dsl)*(objd[0]),(Short)parv[0]);break;
      case Img2dsf: result=Operation2((img2dsf)*(objs[0]),(Img2dsf)*(objd[0]),(Short)parv[0]);break;
      default: std::cerr<<"ERROR: Operator not available for this file"<<std::endl;
      result=FAILURE;break;
  }
  WriteArgs(argc,argv,PARC,FINC,FOUTC,&mask,objin,objs,objout,objd,MASK);
  Exit(result);
  return 0;
}
#endif
```



Template: operator function

GREYC IMAGE ENSICAEN

```
Errc Erosion(Img2duc &ims,Img2duc &imd,Short connexity = 4) {
   Point2d p;
   Img2duc::ValueType min,val;

   imd.Frame(0,1);
   if (connexity == 4){
      for (p.y=1;p.y<ims.Height()-1;p.y++)
          for (p.x=1;p.x<ims.Width()-1;p.x++) {
                    min=ims[p];
                    for (int v=0; v<4; v++){
                       if ((val=ims[p+v4[v]]) < min)
                              min = val;
                    }
                    imd[p] = min;
          }
   } else if (connexity == 8) { // connexity == 8.
               ... v8[]
   } else {
          std::err << "Error: bad connexity value: << connexity << <<std::endl;
          return FAILURE;
   }
   return SUCCESS;
}
```

Masking

GREYC IMAGE ENSICAEN

- The constant MASK:
  - MASK=0 : no masking allowed
    - When input and output are incompatible
  - MASK=1 : masking (input) + unmasking (output)
    - Apply operator only on specified regions and build result by adding input data on unprocessed regions
    - Example: pdilatation
  - MASK=2: masking (input)
    - Apply operator only on specified regions
    - Example: prectangularityselection
  - MASK=3: unmasking (output)
    - Apply operator on input image and build result with only specified regions
    - Example: pmeanfiltering

Genericity

- Write one code for several Pandore types:
- Fichiers .cct:

```
##begin PAdd < TIN1, TIN2, LOOPP, POINT>
Errc PAdd(TIN1 &ims1, TIN2 &ims2, Select<TIN1,TIN2>::Signed &imd) {
  POINT p;

  ##LOOPP(ims1,p)
     imd[p]=(Select<TIN1,TIN2>::Signed::ValueType)(((int)ims1[p]+(int)ims2[p])/2);
  return SUCCESS;
}

## append loadcases
if ((objs[0]->Type()==Po_$TIN1) && (objs[1]->Type()==Po_$TIN2)) {
  TIN1* const ims1=(TIN1*)objs[0];
  TIN2* const ims2=(TIN2*)objs[1];
  objd[0]=new Select<TIN1,TIN2>::Signed(ims1->Props());
  Select<TIN1,TIN2>::Signed* const imd=(Select<TIN1,TIN2>::Signed*)objd[0];
  result=PAdd(*ims1,*ims2,*imd);
} else
## end
##end

##forall(Add,/g2d/,/g2d/)
##forall(Add,/g3d/,/g3d/)
##main(0,2,1,1,"USAGE : %s [-m mask] [im_in1|-] [im_in2|-] [im_out|-]\n")
```

Conclusion

- Documentation, downloads, templates, Examples
  - http://www.greyc.ensicaen.fr/EquipeImage/Pandore
- User documentation
  - Operators: complete list and manual of operators
  - Procedures: Examples of some macro-operators
  - Scripts; Examples of some image processing applications
    - edge detection, region extraction,
- Programmer documentation
  - C++ doc: complete documentation
  - Material: some template files.