Morph-M

Image Processing Library specialized in Mathematical Morphology

Centre de Morphologie Mathématique (CMM)
Mines ParisTech

Overview

- Founding Concept
- Software Development Layers
- Fundamental Notion
- Algorithm Decomposition
- Content
- Conclusion

Founding Concept

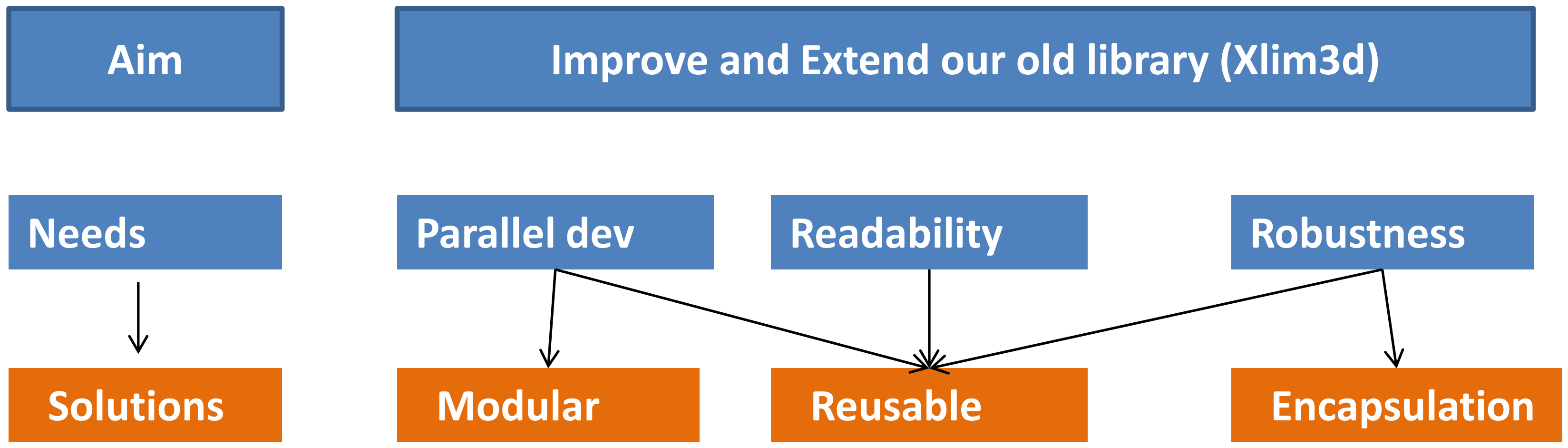
Aim	Improve and Extend our old library (Xlim3d)

Founding Concept

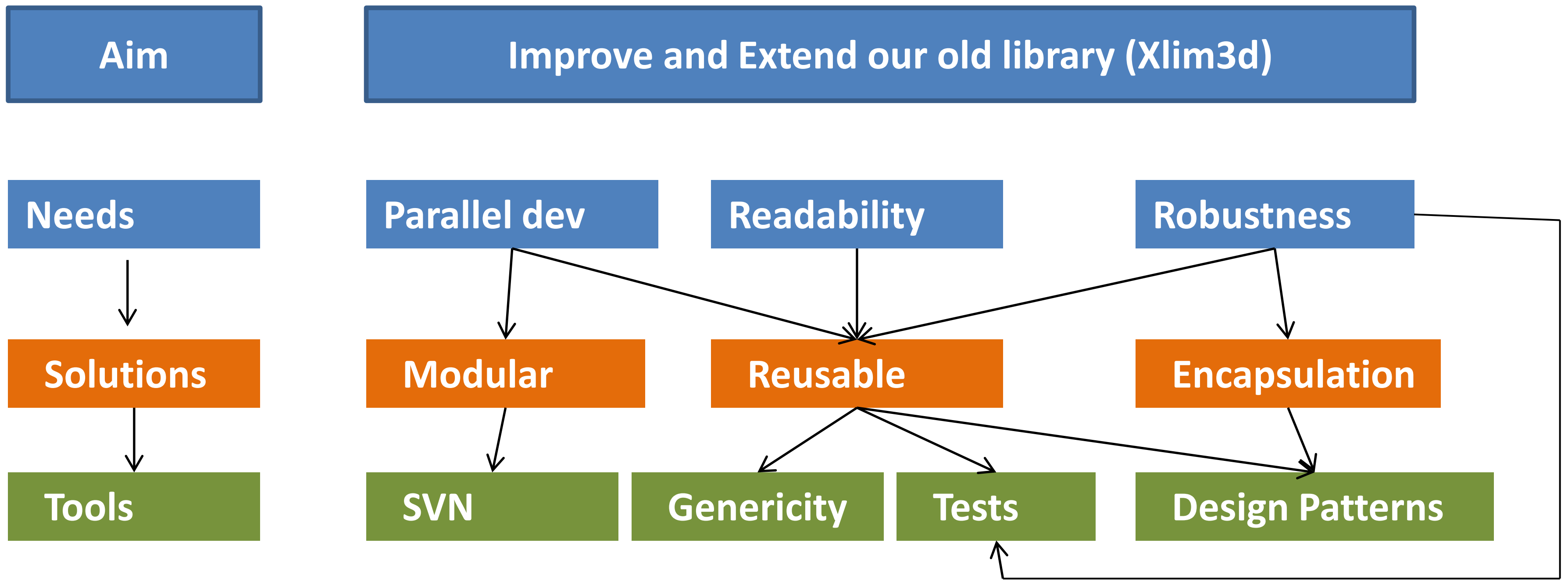
Aim	**Improve and Extend our old library (Xlim3d)**

Needs	**Parallel dev**	**Readability**	**Robustness**

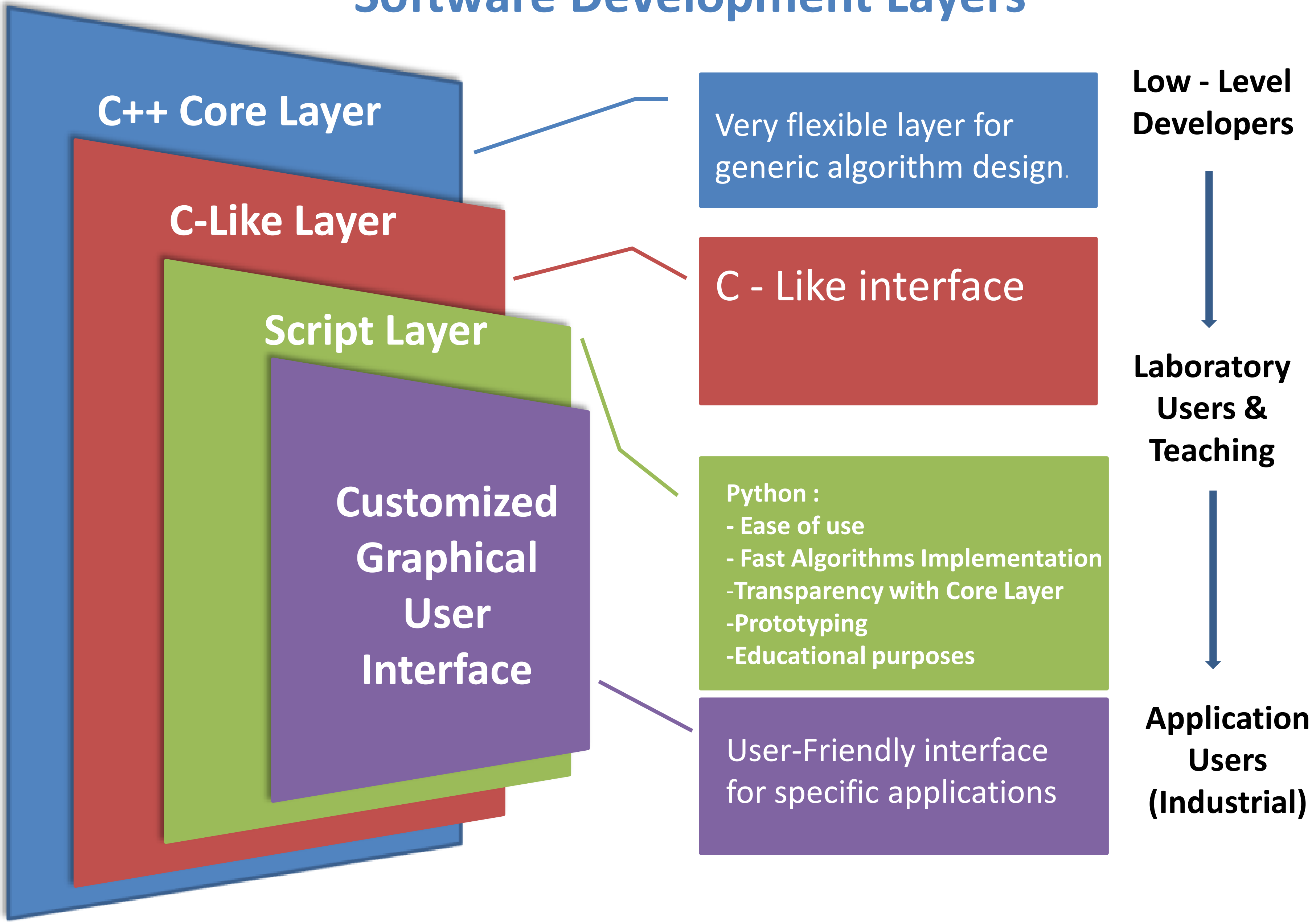
Founding Concept



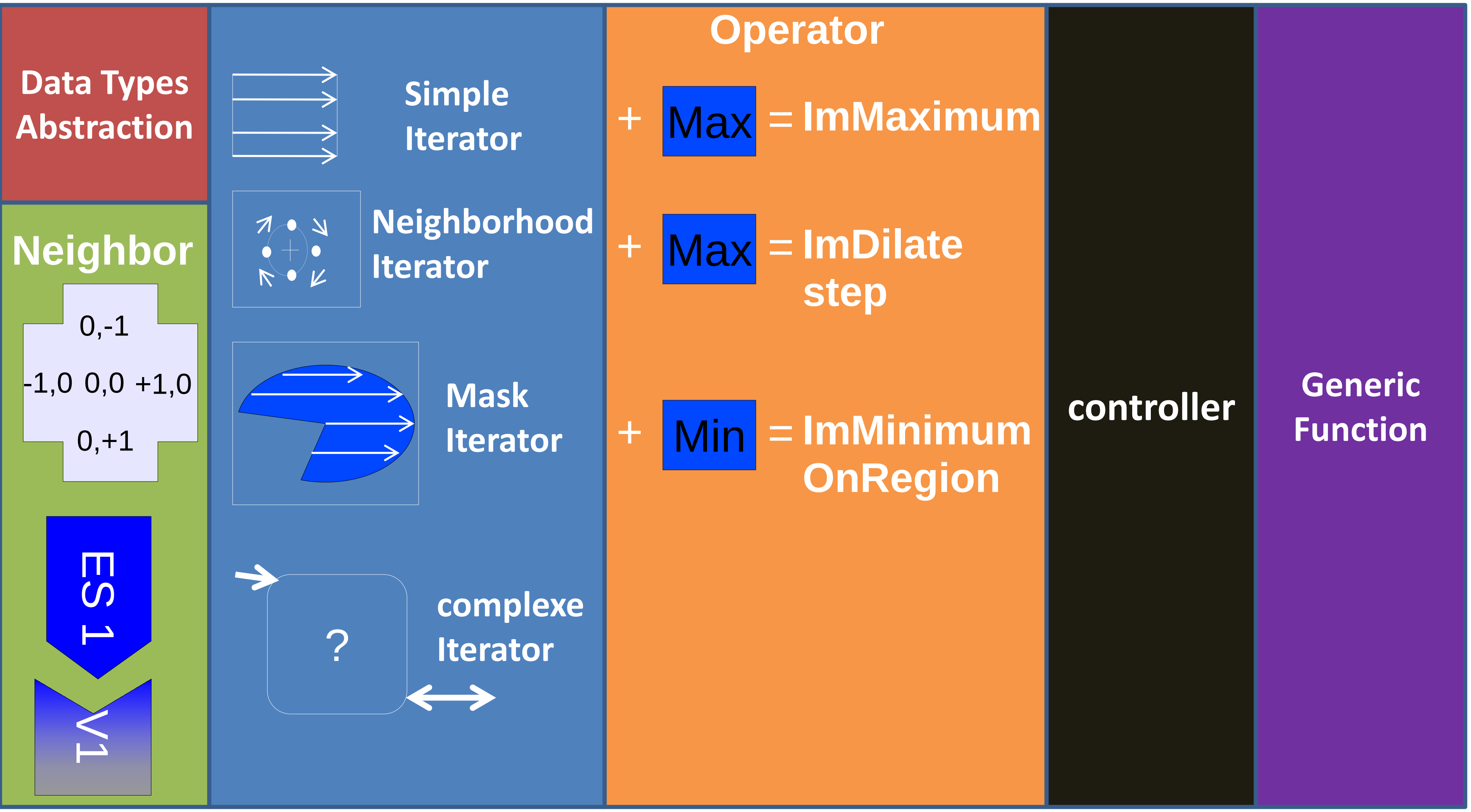
Founding Concept



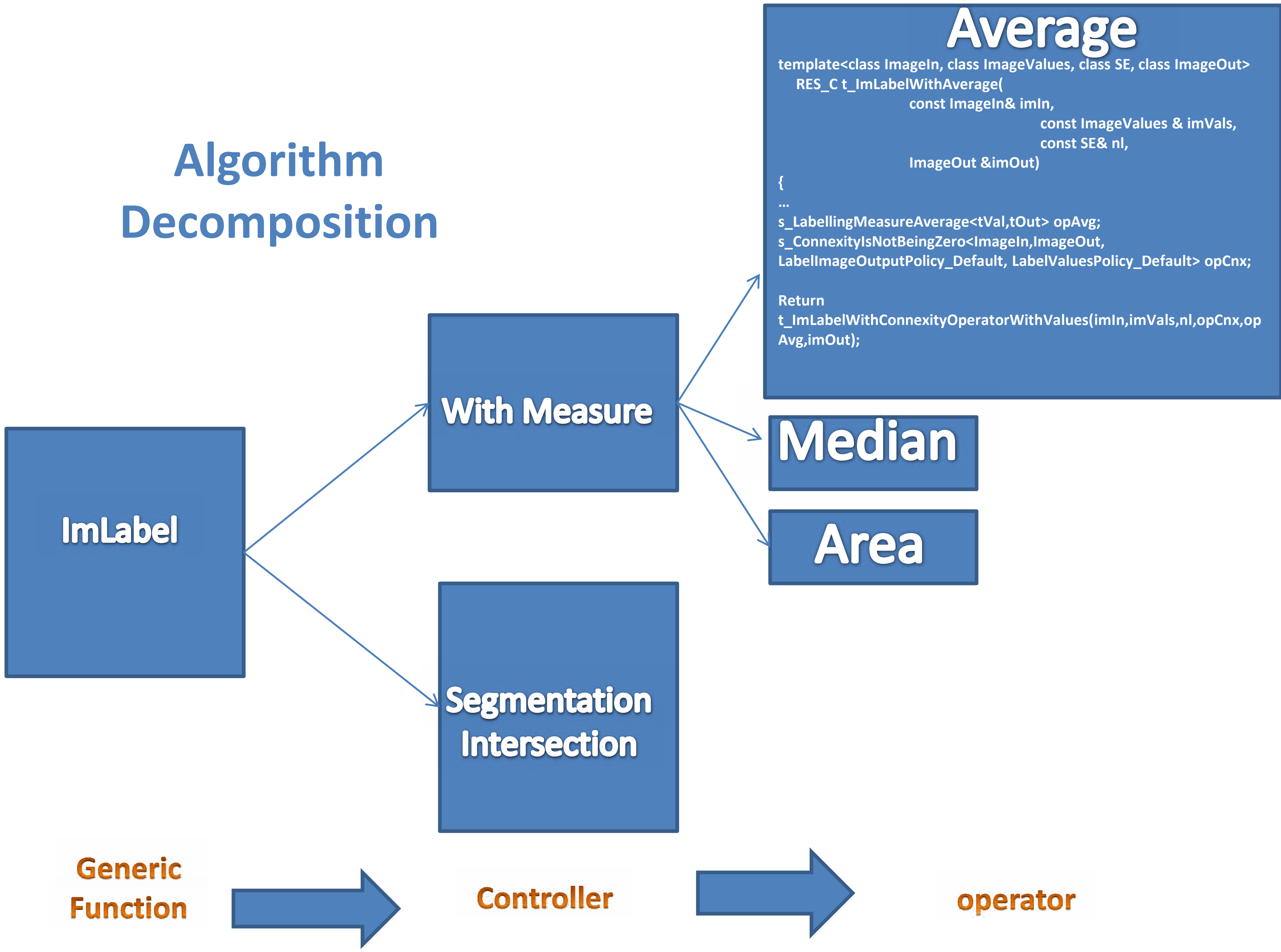
Software Development Layers


C++ Core Layer
C-Like Layer
Script Layer
Customized Graphical User Interface
Very flexible layer for generic algorithm design.
C - Like interface
Python :
- Ease of use
- Fast Algorithms Implementation
-Transparency with Core Layer
-Prototyping
-Educational purposes
User-Friendly interface for specific applications
Low - Level Developers
Laboratory Users & Teaching
Application Users (Industrial)

Fundamental Notion



Algorithm Decomposition



```
template<class ImageIn, class ImageValues, class SE, class ImageOut>
    RES_C t_ImLabelWithAverage(
                         const ImageIn& imIn,
                                              const ImageValues & imVals,
                                              const SE& nl,
                         ImageOut &imOut)
{
…
s_LabellingMeasureAverage<tVal,tOut> opAvg;
s_ConnexityIsNotBeingZero<ImageIn,ImageOut,
LabelImageOutputPolicy_Default, LabelValuesPolicy_Default> opCnx;

Return
t_ImLabelWithConnexityOperatorWithValues(imIn,imVals,nl,opCnx,op
Avg,imOut);
```

Python Example

```
def MyErode2(imIn,nl,imOut):
    itIn  = imIn.imageData()
    itOut = imOut.imageData()

    neighb = createNeighborhood( imIn, nl )

    while itIn.isNotFinished() and itOut.isNotFinished():
        neighb.setCenter( itIn )
        itOut.setPixel( min( neighb.imageData() ) )
        itIn.next()
        itOut.next()
```

```
def main():
  im=fileRead("./Gray/foreman.png")

  imEro = getSame(im)
  imEroRef = getSame(im)
  nl  = neighborsSquare2D

# C++ function
  ImErode( im, nl, imDilRef)
```

```
def MyErode1(imIn, nl, imOut):
   # lambda version:
   morphee.ImNeighborhoodUnaryOperation(imIn, nl, lambda l:min(l), imOut)
    # version using 'min' function:
    morphee.ImNeighborhoodUnaryOperation(imIn, nl, min, imOut)
```

Content

Images Structure :
- Multi-dimensional image data
- Templated image data structures for pixel type abstraction
- Several image le formats avalaible: PNG,TIFF,BMP,JPEG,VTK,...

Structuring Element :
-Myriad of predefined Structuring element
- Easy use and easy manipulation of SE Iterator
- Multi-dimentionnal structuring element
- Dynamic Structuring Element
- Image-based Structing Element
- Neighborhood based Generic operations

Morphological Operation:

-Criteria based morphology (AreaClosing,...)
-Basic morphological operators(Erode,...)
-Distance functions and Geodesic operators
-Lexicographical morphology
- Morphological lters and measures
- Labelling and Leveling
- Morphological Segmentation

Image Processing:
-Arithmetics and logics
- Color conversion and manipulations
- Geometrics transformations (Drawing,rotation,...)
- Pixel-wise generic operatior

Filters:
-Convolution Filters
- Diffusion Filters
- Noising Filters

Statistics Tools:
- Kriging
-Linear algebra
- Morphological Measures (Granulometry,...)
- Usual statistics (mean, variance, ...)
-Histograms and Counting (threshold intervariance class, ...

Graphs and Addons
- Morphology based on graph and Tree
- Graphs Cuts and Graph Manipulation
- FFT, Skeleton, ...

Conclusion

Features & Advantage

- ➡ Portability: 32bits or 64bits, Windows, Linux, OS X
- ➡ Genericity:
  - Modular and robustness project
  - Each Algorithms can be extended easily
  - Algorithms disconnected from data representation
  - Pixel Types abstraction (scalar, vector, matrix,…)
  - Robustness
  - Collaborative working
- ➡ A Lots of Addons:
  - Morphology on Graph
  - Morphology on Multi-Hyper spectale images
  - Skeleton
  - Maxtree, FFT,….

Conclusion

Features & Advantage

- ➡ Portability: 32bits or 64bits, Windows, Linux, OS X
- ➡ Genericity:
  - Modular and robustness project
  - Each Algorithms can be extended easily
  - Algorithms disconnected from data representation
  - Pixel Types abstraction (scalar, vector, matrix,...)
  - Robustness
  - Collaborative working
- ➡ A Lots of Addons:
  - Morphology on Graph
  - Morphology on Multi-Hyper spectale images
  - Skeleton
  - Maxtree, FFT,....

Drawback

- ➡ Coding time (heavy design, funny debug...)
- ➡ Performance (Not design for that !)
- ➡ Code Size (must be patient for browsing and compiling the code)
- ➡ Code Adaptation period (student or post-doc)

Conclusion

Features & Advantage

- ➡ Portability: 32bits or 64bits, Windows, Linux, OS X
- ➡ Genericity:
  - Modular and robustness project
  - Each Algorithms can be extended easily
  - Algorithms disconnected from data representation
  - Pixel Types abstraction (scalar, vector, matrix,…)
  - Robustness
  - Collaborative working
- ➡ A Lots of Addons:
  - Morphology on Graph
  - Morphology on Multi-Hyper spectale images
  - Skeleton
  - Maxtree, FFT,….

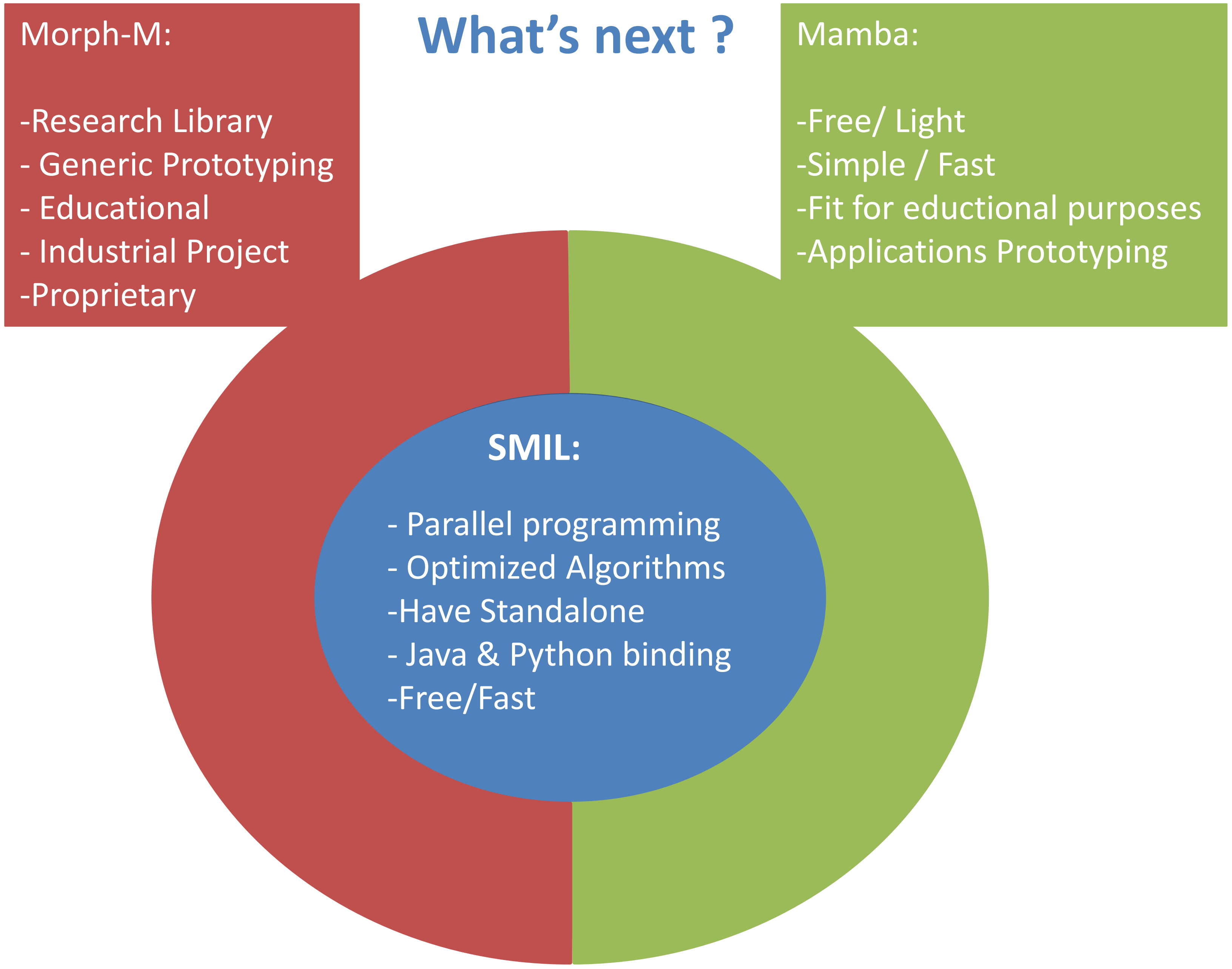
Drawback

- ➡ Coding time (heavy design, funny debug…)
- ➡ Performance (Not design for that !)
- ➡ Code Size (must be patient for browsing and compiling the code)
- ➡ Code Adaptation period (student or post-doc)

Solution: SMIL

- ➡ Light genericity / Generic-friendly
- ➡ Optimized algorithms as far as possible
- ➡ Parallel Programming / fast library
- ➡ Work on 2D and 3D images
- ➡ Predefined image Types (avoid exotic types)
- ➡ Design by Matthieu Faessel

What’s next ?



Morph-M:

-Research Library
- Generic Prototyping
- Educational
- Industrial Project
-Proprietary

Mamba:

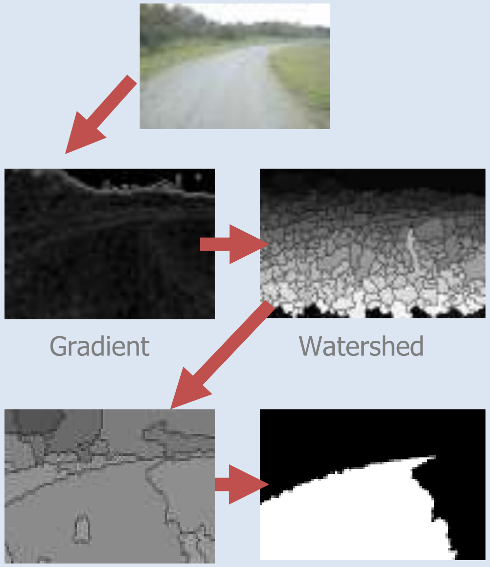
-Free/ Light
-Simple / Fast
-Fit for eductional purposes
-Applications Prototyping

SMIL:

- Parallel programming
- Optimized Algorithms
-Have Standalone
- Java & Python binding
-Free/Fast

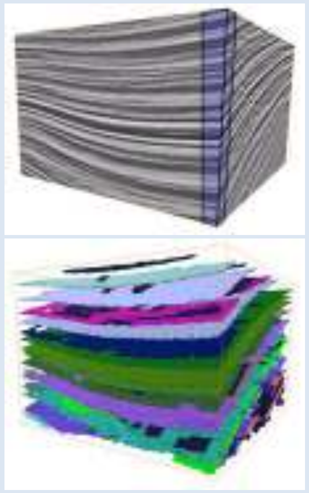
Some Applications

Computer Vision (2D+t)



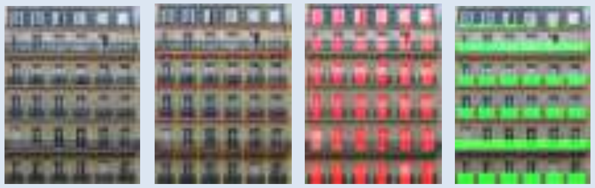
Road Detection & tracking based on hierarchical Segmentation

Segmentation of Seismic Data

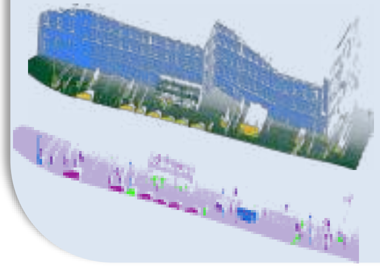


Urban Modeling

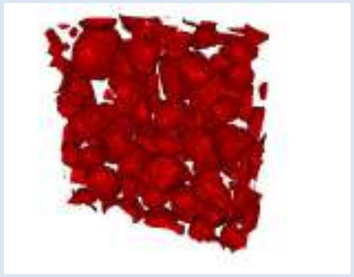
Haussmannian facades Analysis



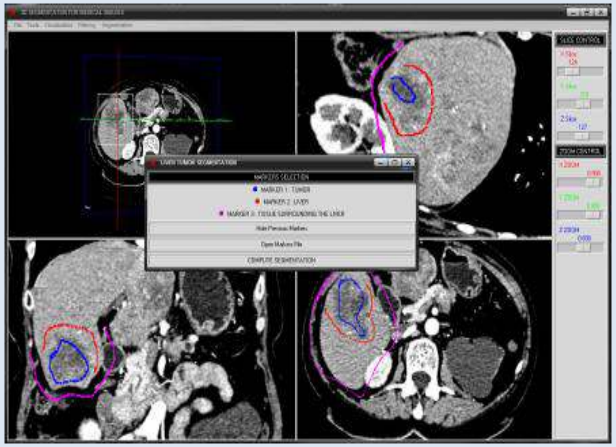
Point Clouds Semantic Analysis

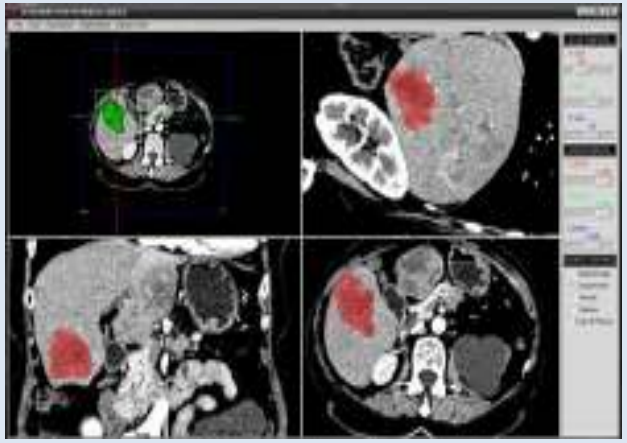


Microtomography Analysis



Interactive Segmentation of 3D Medical Images





Thanks for you attention

Question ?

Morph-M Image Processing Library Specialized in Mathematical Mophology

Mines ParisTech - Centre de Morphologie Mathématique - 35, rue Saint Honoré 77305 Fontainebleau CEDEX (France) – http://cmm.ensmp.fr

Morph-M is the result of the work of several researchers at the Centre for Mathematical Morphology. Morph-M provides a rich environment for the development of image processing algorithms.

Features

- **Portability**

  Windows, Linux & Mac ; 32 and 64 bits

- **Genericity**

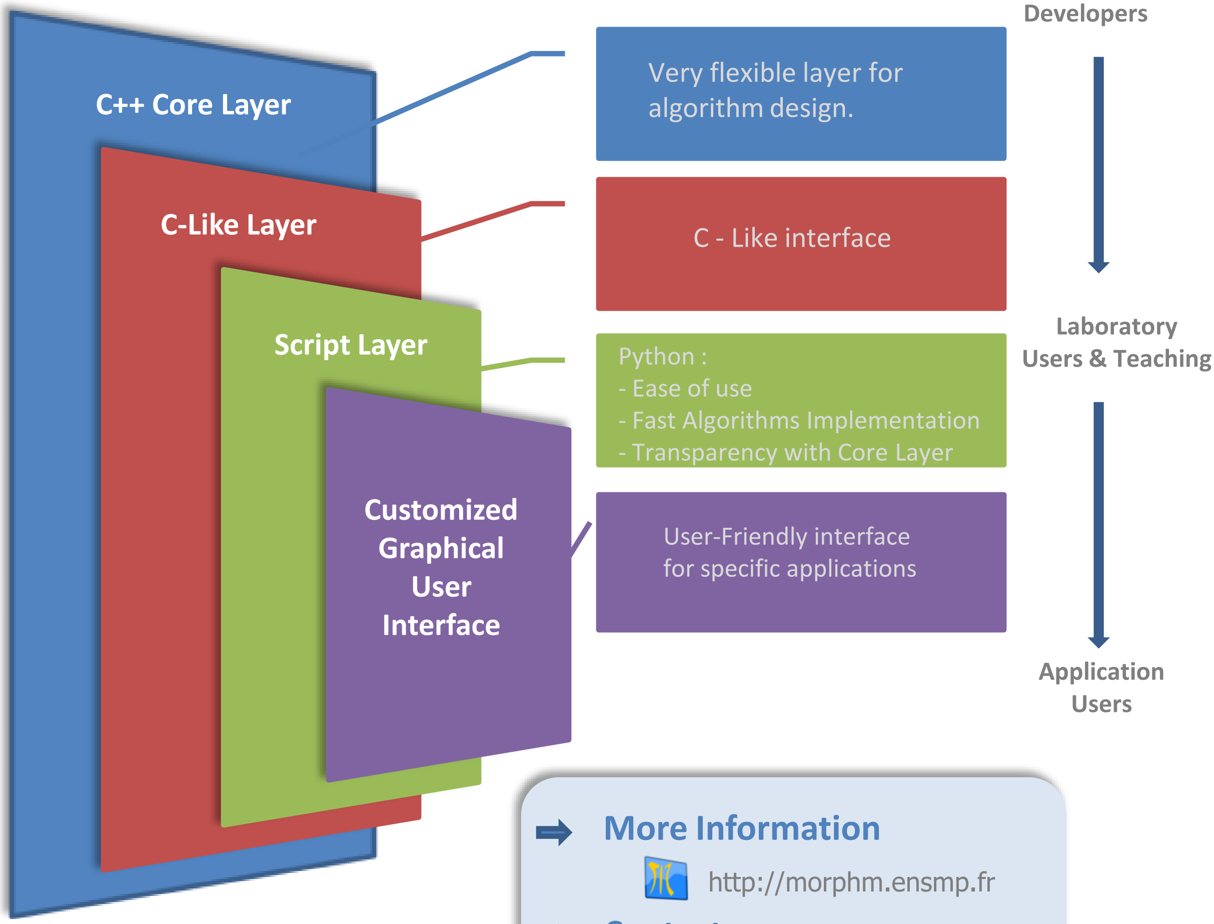
  Morph-M offers a large choice regarding image types and structuring elements

- **Extensible**
  - A myriad of addons
  - Connection with several libraries (vtk, opencv, …)

- **Professional Quality**
  - Nightly regression tests
  - Sources manager
  - Bug tracker
  - CMS

↳ Morph-M represents a reference in mathematical morphology.

Software Development Layers

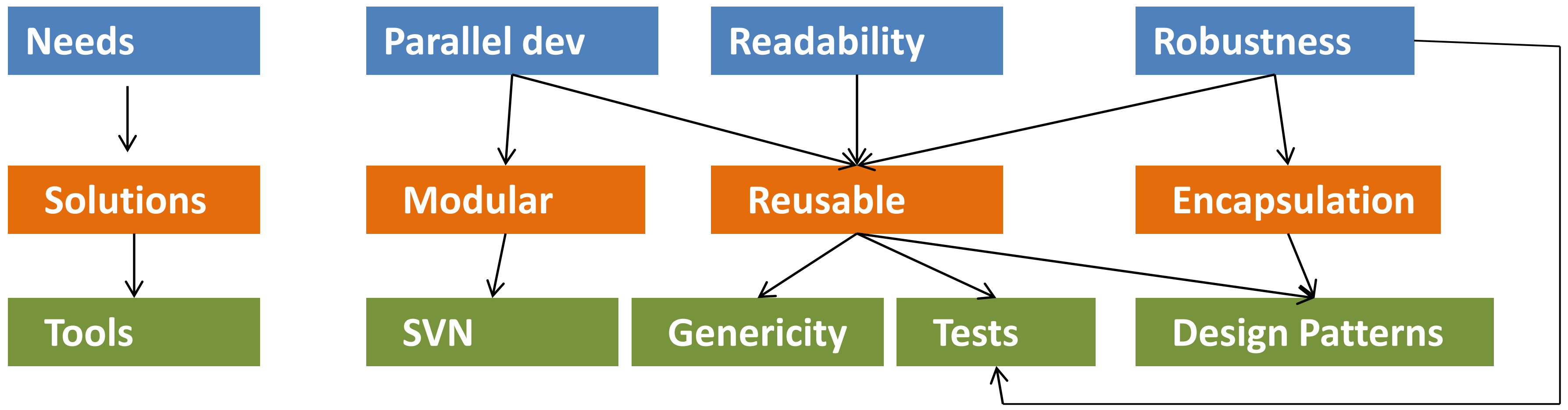

C++ Core Layer
C-Like Layer
Script Layer
Customized Graphical User Interface
Very flexible layer for algorithm design.
C - Like interface
Python :
- Ease of use
- Fast Algorithms Implementation
- Transparency with Core Layer
User-Friendly interface for specific applications
Low - Level Developers
Laboratory Users & Teaching
Application Users

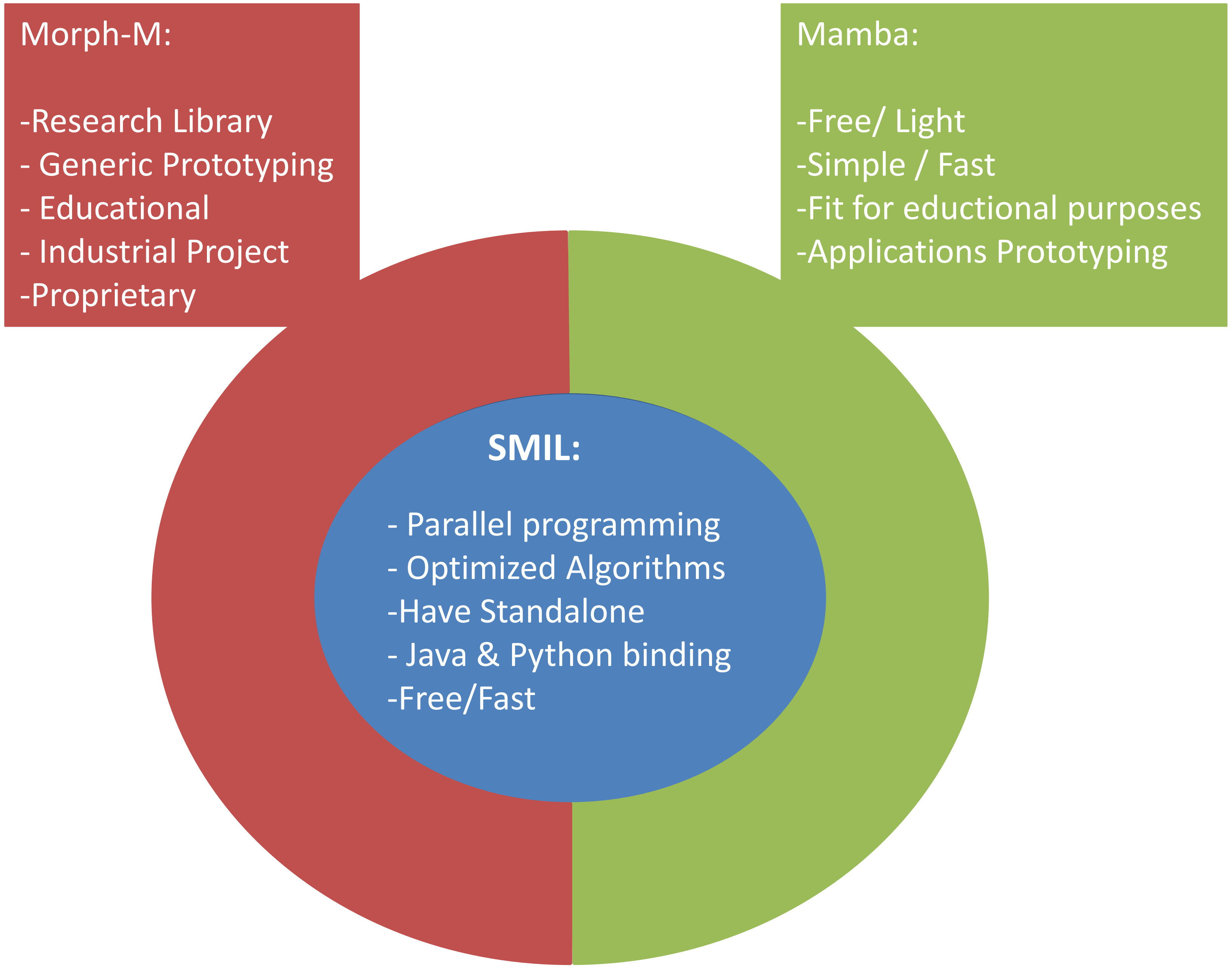
More Information

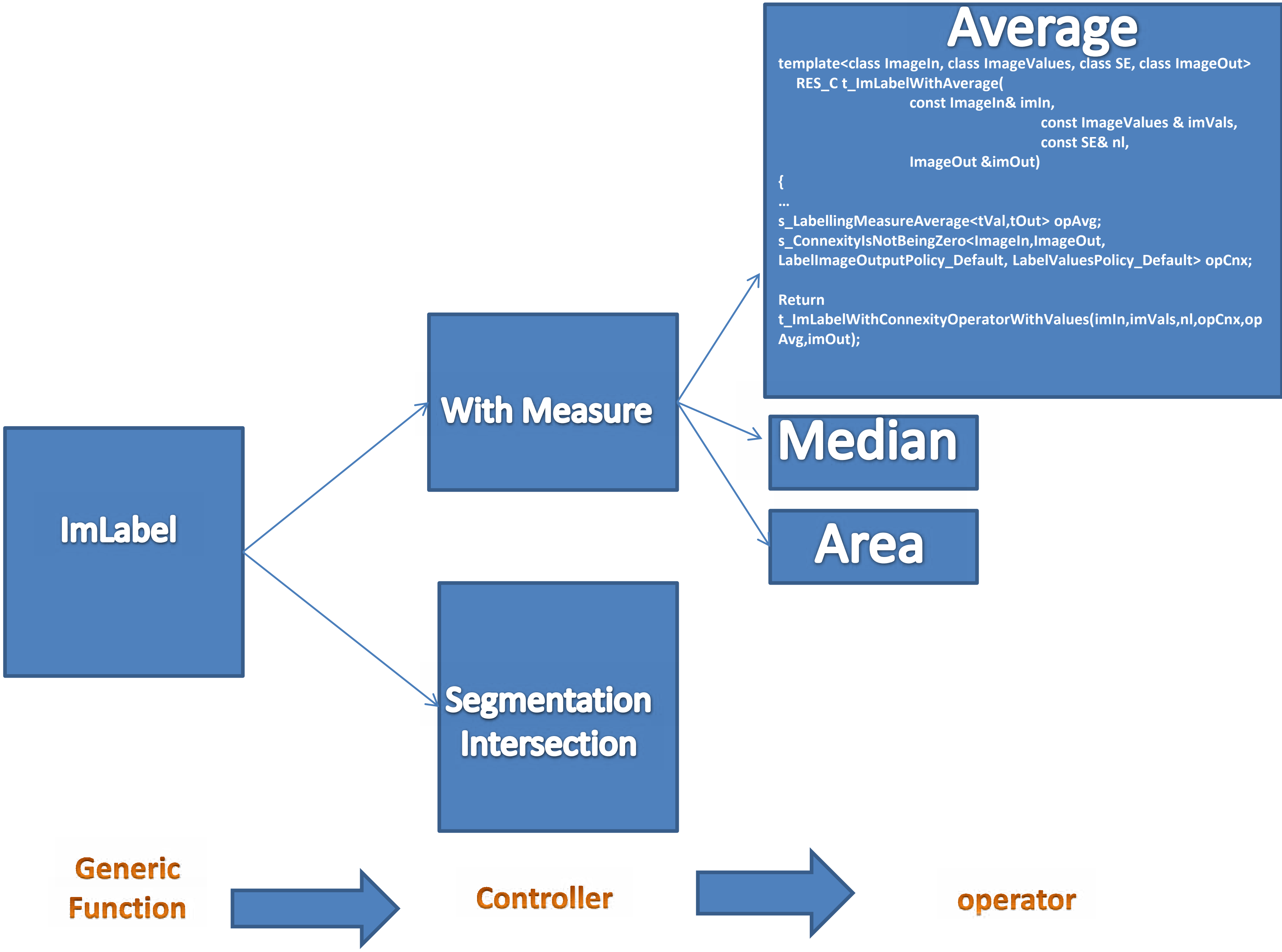
http://morphm.ensmp.fr

Contact

serge.koudoro@mines-paristech.fr


Needs
Solutions
Tools
Parallel dev
Readability
Robustness
Modular
Reusable
Encapsulation
SVN
Genericity
Tests
Design Patterns


Morph-M:
-Research Library
- Generic Prototyping
- Educational
- Industrial Project
-Proprietary
Mamba:
-Free/ Light
-Simple / Fast
-Fit for eductional purposes
-Applications Prototyping
SMIL:
- Parallel programming
- Optimized Algorithms
-Have Standalone
- Java & Python binding
-Free/Fast



```
template<class ImageIn, class ImageValues, class SE, class ImageOut>
    RES_C t_ImLabelWithAverage(
                        const ImageIn& imIn,
                                                const ImageValues & imVals,
                                                const SE& nl,
                        ImageOut &imOut)
{
…
s_LabellingMeasureAverage<tVal,tOut> opAvg;
s_ConnexityIsNotBeingZero<ImageIn,ImageOut,
LabelImageOutputPolicy_Default, LabelValuesPolicy_Default> opCnx;

Return
t_ImLabelWithConnexityOperatorWithValues(imIn,imVals,nl,opCnx,op
Avg,imOut);
```