

# ITK - The Next Generation

## Welcoming and training new ITK Community Members

The University of Wisconsin-Madison

January 2013

This presentation is copyrighted by  
**The Insight Software Consortium**

distributed under the  
**Creative Commons by Attribution License 3.0**  
<http://creativecommons.org/licenses/by/3.0>

1 Virtual Machines Preparation I

2 Introduction to ITK

3 Virtual Machines Preparation II

4 SimpleITK

5 Level-Sets

6 Registration

7 How To Contribute

8 Modularization

# Virtual Machine Preparation I

# Virtual Machines Preparation

- Get USB Memory Stick
- Install VirtualBox from it
- Import the VirtualMachine file
- Boot the Virtual Machine
- Log in
- Get familiar with directories

# Media Content

## Directories and Files

- VirtualBoxInstallers

- VirtualBox-4.2.6-82870-OSX.dmg (Mac)
- VirtualBox-4.2.6-82870-Win.exe (Windows)
- VirtualBox-4.2.6-82870-Linux\_amd64.run (Linux-64bit)
- VirtualBox-4.2.6-82870-Linux\_x86.run (Linux-32bit)

# Media Content

## Directories and Files

- VirtualBoxInstallers
  - VirtualBox-4.2.6-82870-OSX.dmg (Mac)
  - VirtualBox-4.2.6-82870-Win.exe (Windows)
  - VirtualBox-4.2.6-82870-Linux\_amd64.run (Linux-64bit)
  - VirtualBox-4.2.6-82870-Linux\_x86.run (Linux-32bit)
- VirtualMachine
  - ITK-Madison-Jan-2013.ova

# Install VirtualBox

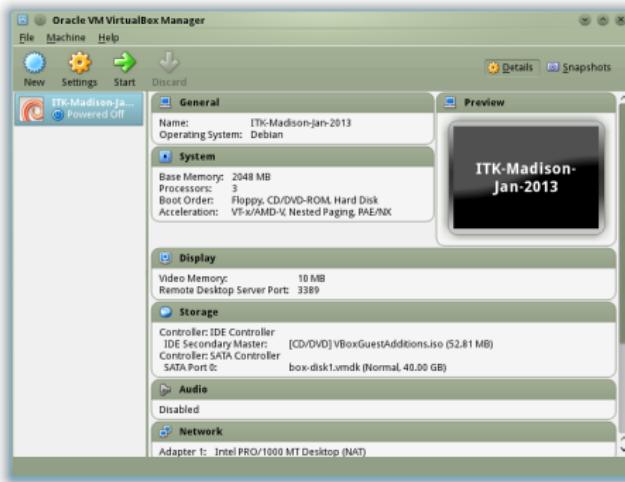
- Select the installer for your platform
- Run it

# Alternative Linux Installation

- You can also install VirtualBox by doing:
- `sudo apt-get install virtualbox-ose-qt`

# Importing the Virtual Machine

- Run VirtualBox
- In “File” Menu select “Import Appliance”
- Provide the filename in the USB stick  
“VirtualMachine/ITK-Madison-2013.ova”
- A progress bar will appear, and when it finishes you should see:

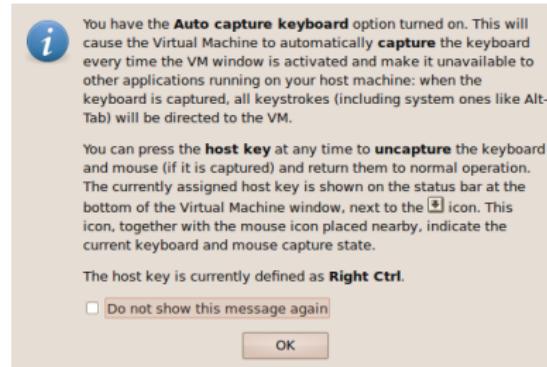


# Introduction to ITK...

# Virtual Machine Preparation II

# Booting the Virtual Machine

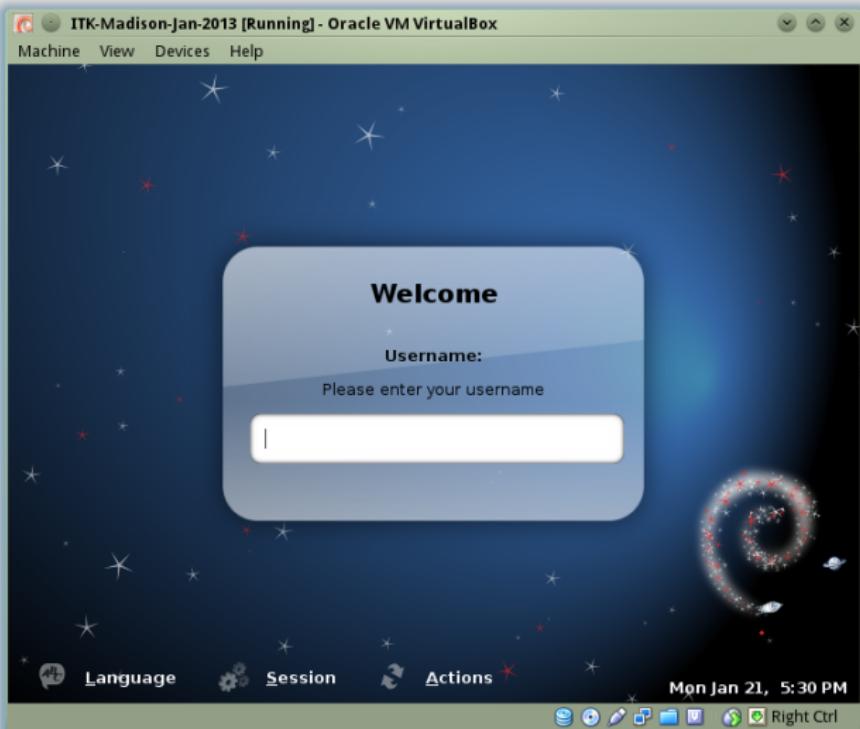
- Click on the “ITK-Madison-Jan-2013” icon on the left, to select it.
- Click on the Green Arrow at the top “Show”.
- The VM will start to boot and you will see the warning:



- Click “OK”

# Booting the Virtual Machine

- The boot sequence should continue and you should see:



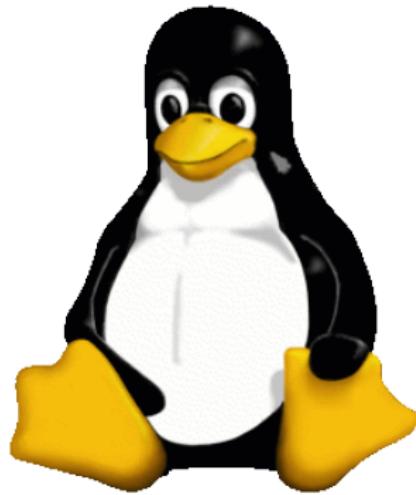
# Booting the Virtual Machine

- Username: itk
- Password: itk

Your Virtual Machine  
is Ready !

# Software Environment

# Welcome to Debian GNU/Linux

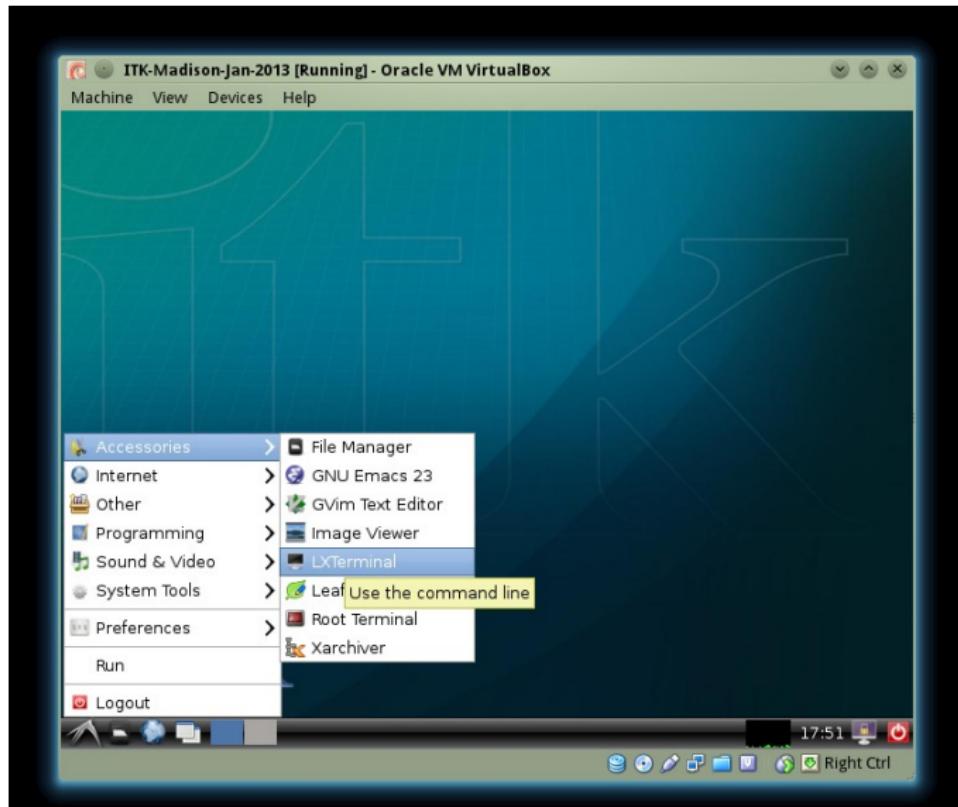


# How to take the mouse out of the Virtual Machine

- Hit the **RIGHT CTRL** Key on Windows / Linux Host
- Hit the **LEFT APPLE** Key on Mac Host

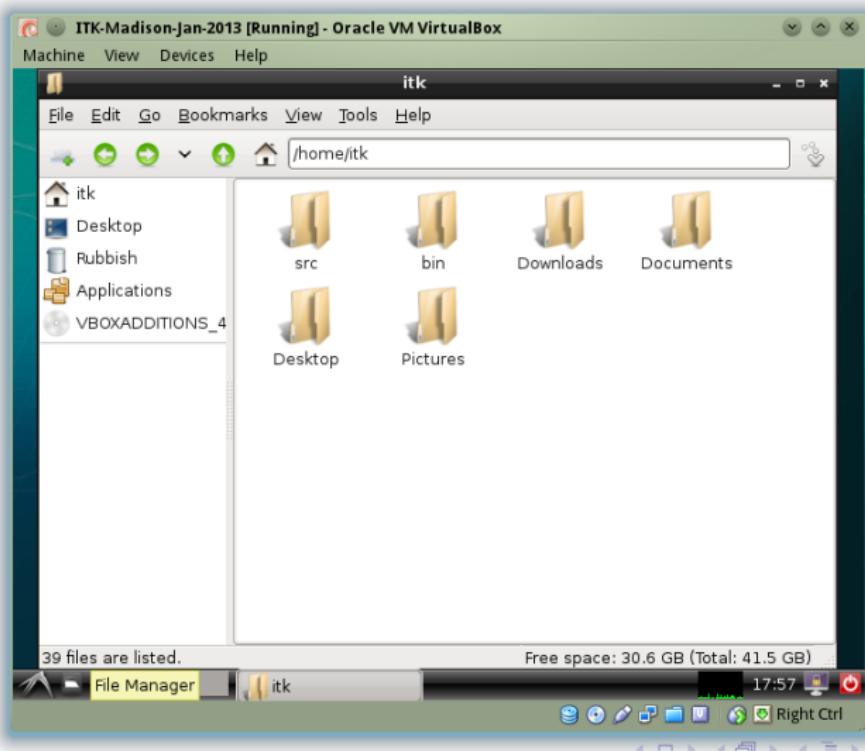
# How to Open a Terminal - Menu in Lower Left Corner

To type your command line instructions



# How to Navigate Directories

Double Click in Folder Icons in the Desktop



# Walk through the directories

- Find source code of exercises

```
cd ~/src/ITKv4-TheNextGeneration-Tutorial/Exercises  
pwd  
ls  
pcmanfm .
```

# Walk through the directories

- Find source code of exercises

```
cd ~/src/ITKv4-TheNextGeneration-Tutorial/Exercises  
pwd  
ls  
pcmanfm .
```

- Find binary build of exercises

```
cd ~/bin/ITKv4-TheNextGeneration-Tutorial/Exercises  
pwd  
ls
```

# The TAB Key is your Friend

- When writing filenames
- Use the TAB key for completions
- No need to type full filenames

# How to View Images

- Go to the directory
- Invoke “ImageViewer” application

```
cd ~/data
```

```
ImageViewer BrainProtonDensitySlice.png
```

# How to View Images

- Go to the directory
- Invoke “ImageViewer” application

```
cd ~/data
```

```
ImageViewer BrainProtonDensitySlice.png
```

- Hit “+” key to zoom in

# How to View Images

- Go to the directory
- Invoke “ImageViewer” application

```
cd ~/data
```

```
ImageViewer BrainProtonDensitySlice.png
```

- Hit “+” key to zoom in
- Hit “-” key to zoom out

# How to View Images

- Go to the directory
- Invoke “ImageViewer” application

```
cd ~/data
```

```
ImageViewer BrainProtonDensitySlice.png
```

- Hit “+” key to zoom in
- Hit “-” key to zoom out
- Hit ESC key to quit the application

# SimpleITK

- Hides templates

- Hides templates
- Smarter IO

- Hides templates
- Smarter IO
- Both procedural and object-oriented interfaces

- Hides templates
- Smarter IO
- Both procedural and object-oriented interfaces
- Easier wrapping

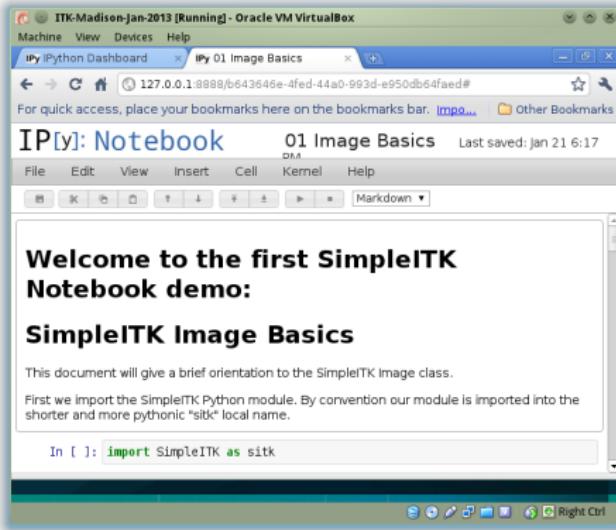
- Hides templates
- Smarter IO
- Both procedural and object-oriented interfaces
- Easier wrapping
- Binary distribution  
(`easy_install SimpleITK`)

# How to run the example?

```
cd ~/src/SimpleITK-Notebooks  
ipython notebook --pylab=inline
```

# IPython Notebook Quickstart

- Shift-Enter to run a cell
- Ctrl-m h to show keyboard shortcuts



## More information on SimpleITK

- <http://simpleitk.org/>
- [http://www.itk.org/Wiki/ITK\\_Release\\_4/SimpleITK](http://www.itk.org/Wiki/ITK_Release_4/SimpleITK)

# Refactored Level Sets

Insight Software Consortium  
Megason Lab, Department of Systems Biology  
Harvard Medical School

Arnaud Gelas

Kishore Mosaliganti

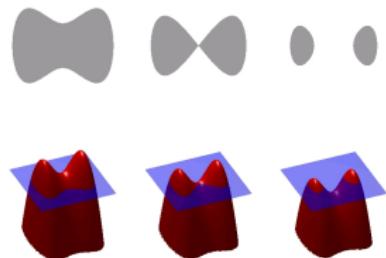
Sean Megason

# Introduction

# What is a level-set function?

## Definition

- Implicit function  $\phi : \Omega \rightarrow \mathbb{R}$
- If  $\phi(p) = 0$ ,  $p$  is on the interface  $\Gamma$
- If  $\phi(p) < 0$ ,  $p$  is inside
- Else  $p$  is outside



# Level-Set Evolution

- Deforms the level-set function
  - Driven by given PDE

# Level-Set Evolution

- Deforms the level-set function
  - Driven by given PDE
  - Regularized Advection

$$\frac{\partial \phi}{\partial \tau} = \alpha \cdot \vec{A}(p) \bullet \vec{\nabla} \phi + \gamma \cdot \operatorname{div} \left( \frac{\vec{\nabla} \phi}{\|\vec{\nabla} \phi\|} \right) \cdot \|\vec{\nabla} \phi\|$$

## Advection

$$\alpha \cdot \vec{A}(p) \bullet \vec{\nabla} \phi$$

$\alpha$ : coefficient

$A(p)$ : Advection field

## Curvature

$$\gamma \cdot \|\vec{\nabla} \phi\| \cdot \operatorname{div} \frac{\vec{\nabla} \phi}{\|\vec{\nabla} \phi\|}$$

$\gamma$ : coefficient

# Level-Set Evolution

- Deforms the level-set function
  - Driven by given PDE
    - Regularized Advection

$$\frac{\partial \phi}{\partial \tau} = \alpha \cdot \vec{A}(p) \bullet \vec{\nabla} \phi + \gamma \cdot \operatorname{div} \left( \frac{\vec{\nabla} \phi}{\|\vec{\nabla} \phi\|} \right) \cdot \|\vec{\nabla} \phi\|$$

- Regularized Propagation

$$\frac{\partial \phi}{\partial \tau} = \beta \cdot P(p) \cdot \|\vec{\nabla} \phi\| + \gamma \cdot \operatorname{div} \left( \frac{\vec{\nabla} \phi}{\|\vec{\nabla} \phi\|} \right) \cdot \|\vec{\nabla} \phi\|$$

## Propagation

$$\beta \cdot P(p) \cdot \|\vec{\nabla} \phi\|$$

$\beta$ : coefficient

$P(p)$ : Propagation field

## Curvature

$$\gamma \cdot \|\vec{\nabla} \phi\| \cdot \operatorname{div} \frac{\vec{\nabla} \phi}{\|\vec{\nabla} \phi\|}$$

$\gamma$ : coefficient

# Level-Set Evolution

- Deforms the level-set function
  - Driven by given PDE
  - Regularized Advection

$$\frac{\partial \phi}{\partial \tau} = \alpha \cdot \vec{A}(p) \bullet \vec{\nabla} \phi + \gamma \cdot \operatorname{div} \left( \frac{\vec{\nabla} \phi}{\|\vec{\nabla} \phi\|} \right) \cdot \|\vec{\nabla} \phi\|$$

- Regularized Propagation

$$\frac{\partial \phi}{\partial \tau} = \beta \cdot P(p) \cdot \|\vec{\nabla} \phi\| + \gamma \cdot \operatorname{div} \left( \frac{\vec{\nabla} \phi}{\|\vec{\nabla} \phi\|} \right) \cdot \|\vec{\nabla} \phi\|$$

- Chan and Vese

$$\frac{\partial \phi}{\partial \tau} = \delta_\epsilon(\phi) (-\lambda_{in} (I - \mu_{in}) + \lambda_{out} (I - \mu_{out}))$$

## Chan And Vese Internal

$$\delta_\epsilon(\phi) (\lambda_{in} (I - \mu_{in}))$$

$\lambda_{in}$ : coefficient

$\mu_{in}$ : Internal Mean

## Chan And Vese External

$$\delta_\epsilon(\phi) (\lambda_{out} (I - \mu_{out}))$$

$\lambda_{out}$ : coefficient

$\mu_{out}$ : External Mean

# Level-Set Evolution

- Iterative computation
- Topological flexibility

# Level Sets: Challenges in Segmentation?

- PDE Term choice
  - Advection terms ?
  - Propagation terms ?
  - Region terms ?
  - Regularization terms ?

# Level Sets: Challenges in Segmentation?

- PDE Term choice
  - Advection terms ?
  - Propagation terms ?
  - Region terms ?
  - Regularization terms ?
- Stopping criterion
  - Number of Iterations ?
  - Variation of interface length / area ?
  - Variation of shape area / volume ?

# Level Sets: Challenges in Segmentation?

- PDE Term choice
  - Advection terms ?
  - Propagation terms ?
  - Region terms ?
  - Regularization terms ?
- Stopping criterion
  - Number of Iterations ?
  - Variation of interface length / area ?
  - Variation of shape area / volume ?
- PDE Parameters tuning

# Level Sets Representation

## Discrete

- Dense

# Level Sets Representation

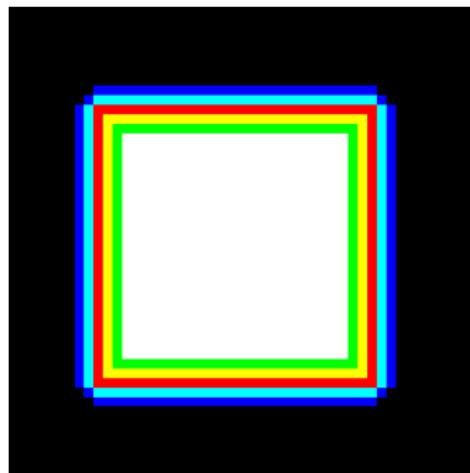
## Discrete

- Dense
- Sparse

# Level Sets Representation

## Discrete

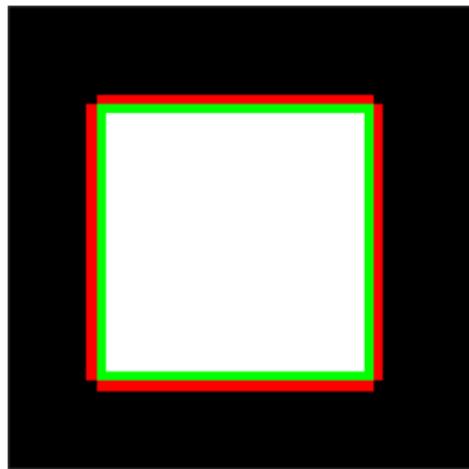
- Dense
  - Sparse
- ① Whitaker



# Level Sets Representation

## Discrete

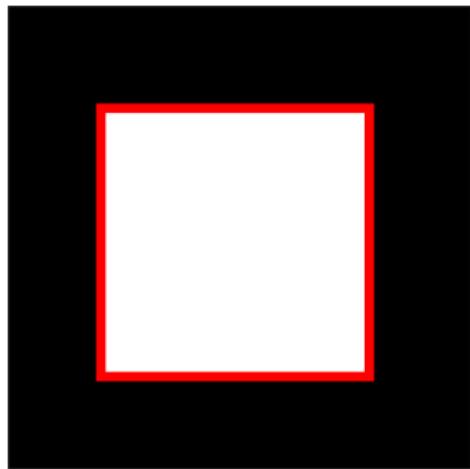
- Dense
- Sparse
  - ① Whitaker
  - ② Shi



# Level Sets Representation

## Discrete

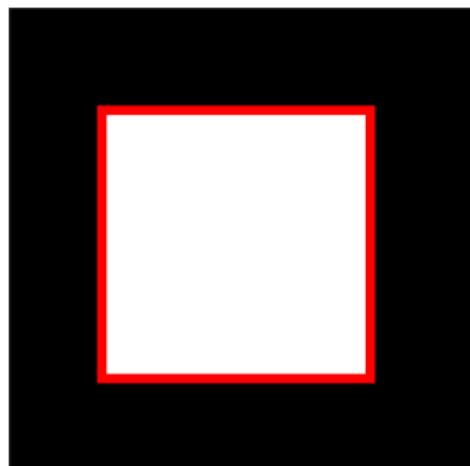
- Dense
- Sparse
  - ① Whitaker
  - ② Shi
  - ③ Malcolm



# Level Sets Representation

## Discrete

- Dense
- Sparse
  - ① Whitaker
  - ② Shi
  - ③ Malcolm



## Parametric

- Easy integration of new representation

# Level Sets Equation

## Term

- Contribution for  $\phi$  evolution
- Contribution for time step computation
- Coefficient

Easy to contribute new terms!

# Level Sets Equation

## Term

- Contribution for  $\phi$  evolution
- Contribution for time step computation
- Coefficient

Easy to contribute new terms!

# Level Sets Equation

## Term

- Contribution for  $\phi$  evolution
- Contribution for time step computation
- Coefficient

Easy to contribute new terms!

## TermContainer

- Represent a given PDEs
- Mix of any term
- Independent of the representation

Easy to contribute new PDEs!

# Level Sets Equation

## Term

- Contribution for  $\phi$  evolution
- Contribution for time step computation
- Coefficient

Easy to contribute new terms!

## TermContainer

- Represent a given PDEs
- Mix of any term
- Independent of the representation

Easy to contribute new PDEs!

# Other Features

- N Level-Sets function evolving at the same time
- Geometrical Constraints

# How to run the example?

```
cd ~/bin/ITKv4-TheNextGeneration-Tutorial/bin/  
  
./SingleLevelSetWhitaker  
  <InputImage>  
  <NumberOfIterations>  
  <Visualization (0 or 1)>  
  <OutputImage>  
  
. ./SingleLevelSetWhitaker  
  ~/data/cells.png  
  800  
  1  
  cells_segmented.mha
```

# Example Code Walk-Through

# Create a level-set function from binary mask

```
53 const unsigned int Dimension = 2;  
  
96 typedef float PixelType;  
97  
98 typedef itk::WhitakerSparseLevelSetImage<  
99   PixelType, Dimension > WhitakerSparseLevelSetImageType;  
100  
101 typedef itk::BinaryImageToLevelSetImageAdaptor<  
102   InputImageType, WhitakerSparseLevelSetImageType > BinaryToSparseAdaptorType;  
103  
104 BinaryToSparseAdaptorType::Pointer adaptor = BinaryToSparseAdaptorType::New();  
105 adaptor->SetInputImage( binaryImage );  
106 adaptor->Initialize();  
  
110  
111  
112 typedef BinaryToSparseAdaptorType::LevelSetType SparseLevelSetType;  
113 SparseLevelSetType::Pointer levelSet = adaptor->GetLevelSet();
```

# Create a domain for the level-set function

```
118  typedef itk::IdentifierType IdentifierType;
119  typedef std::list< IdentifierType > IdListType;
120
121  IdListType listIds;
122  listIds.push_back( 1 );
123
124
125
126  typedef itk::Image< IdListType, Dimension > IdListImageType;
127  IdListImageType::Pointer idImage = IdListImageType::New();
128  idImage->SetRegions( inputImage->GetLargestPossibleRegion() );
129  idImage->Allocate();
130  idImage->FillBuffer( listIds );
131
132  typedef itk::Image< short, Dimension > CacheImageType;
133  typedef itk::LevelSetDomainMapImageFilter< IdListImageType, CacheImageType >
134                                         DomainMapImageFilterType;
135  DomainMapImageFilterType::Pointer domainMapFilter = DomainMapImageFilterType::New();
136  domainMapFilter->SetInput( idImage );
137  domainMapFilter->Update();
```

# Setting up the level-set container

```
141  typedef SparseLevelSetType::OutputRealType LevelSetOutputRealType;
142
143  typedef itk::SigmoidRegularizedHeavisideStepFunction<
144      LevelSetOutputRealType, LevelSetOutputRealType > HeavisideFunctionBaseType;
145  HeavisideFunctionBaseType::Pointer heaviside = HeavisideFunctionBaseType::New();
146  heaviside->SetEpsilon( 1.0 );
147
148
149  typedef itk::LevelSetContainer<
150      IdentifierType, SparseLevelSetType > LevelSetContainerType;
151
152  LevelSetContainerType::Pointer lscontainer = LevelSetContainerType::New();
153  lscontainer->SetHeaviside( heaviside );
154  lscontainer->SetDomainMapFilter( domainMapFilter );
155
156  lscontainer->AddLevelSet( 0, levelSet );
```

# Creating PDE Terms

- Chan and Vese internal term

```
163  typedef itk::LevelSetEquationChanAndVeseInternalTerm<
164      InputImageType, LevelSetContainerType > InternalTermType;
165
166  InternalTermType::Pointer cvInternalTerm0 = InternalTermType::New();
167  cvInternalTerm0->SetInput( inputImage );
168  cvInternalTerm0->SetCoefficient( 1.0 );
169  cvInternalTerm0->SetCurrentLevelSetId( 0 );
170  cvInternalTerm0->SetLevelSetContainer( lscontainer );
```

- Chan and Vese external term

```
174  typedef itk::LevelSetEquationChanAndVeseExternalTerm<
175      InputImageType, LevelSetContainerType > ExternalTermType;
176
177  ExternalTermType::Pointer cvExternalTerm0 = ExternalTermType::New();
178  cvExternalTerm0->SetInput( inputImage );
179  cvExternalTerm0->SetCoefficient( 1.0 );
180  cvExternalTerm0->SetCurrentLevelSetId( 0 );
181  cvExternalTerm0->SetLevelSetContainer( lscontainer );
```

# Setting up PDE

```
187  typedef itk::LevelSetEquationTermContainer<
188    InputImageType, LevelSetContainerType > TermContainerType;

190  TermContainerType::Pointer termContainer0 = TermContainerType::New();
191  termContainer0->SetInput( inputImage );
192  termContainer0->SetLevelSetContainer( lscontainer );

194  termContainer0->AddTerm( 0, cvInternalTerm0 );
195  termContainer0->AddTerm( 1, cvExternalTerm0 );

199  typedef itk::LevelSetEquationContainer< TermContainerType > EquationContainerType;
200  EquationContainerType::Pointer equationContainer = EquationContainerType::New();
201  equationContainer->AddEquation( 0, termContainer0 );
202  equationContainer->SetLevelSetContainer( lscontainer );
```

# Stopping criterion

```
204  typedef itk::LevelSetEvolutionNumberOfIterationsStoppingCriterion<
205      LevelSetContainerType > StoppingCriterionType;
206
207  StoppingCriterionType::Pointer criterion = StoppingCriterionType::New();
208  criterion->SetNumberOfIterations( atoi( argv[2] ) );
```

# Starts the evolution

- Set a stopping criterion
- Evolve

```
226 IterationUpdateCommandType::Pointer iterationUpdateCommand = IterationUpdateCommand  
227 iterationUpdateCommand->SetFilterToUpdate( visualizer );  
228 iterationUpdateCommand->SetUpdatePeriod( 1 );  
  
230 if( atoi( argv[3] ) == 1 )  
231 {  
232 evolution->AddObserver( itk::IterationEvent(), iterationUpdateCommand );  
233 }  
  
235 evolution->SetStoppingCriterion( criterion );
```

Exercise: Add a  
curvature term to  
`LevelSetExercise1.cxx`

# Refactored Registration Framework

Insight Software Consortium

PICSL @ University of Pennsylvania

Brian Avants, Nicholas Tustison, Gang Song,  
Baohua Wu, Michael Stauffer, James C. Gee

# What is registration?

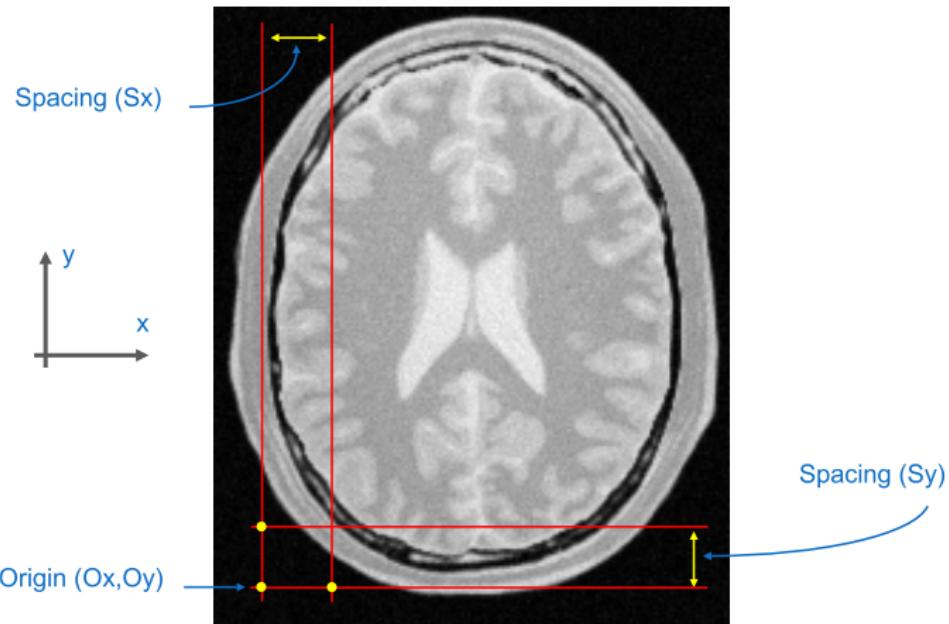


# What is registration?

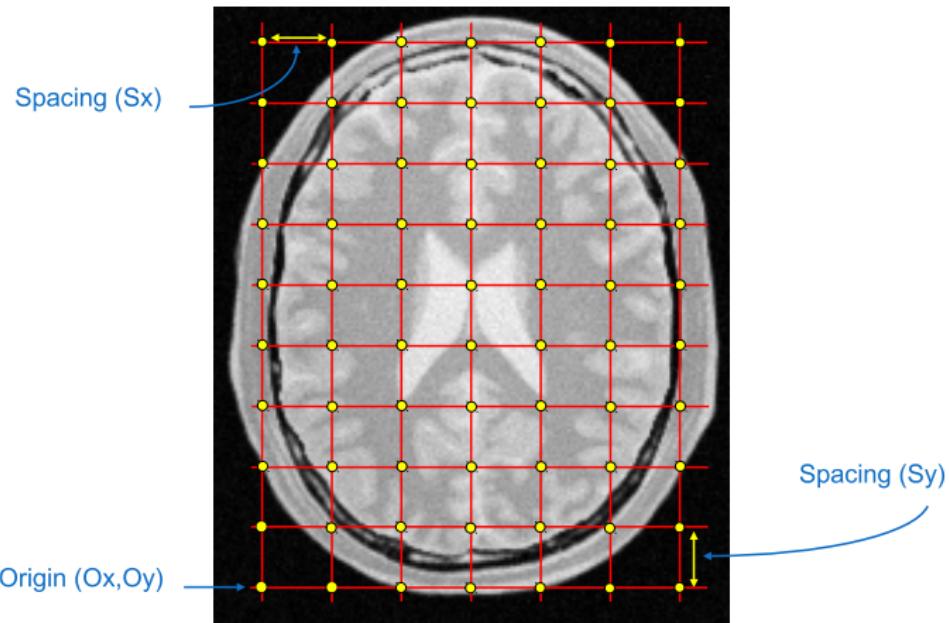


$$\|I(x) - J(\phi(x))\|^2 + R(\phi(x))$$

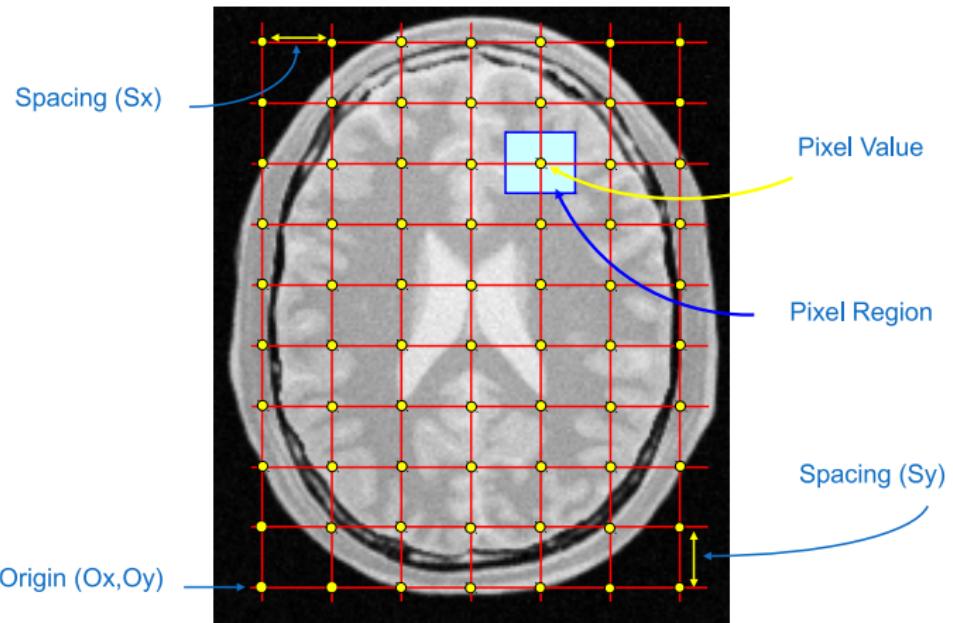
# Image Origin and Spacing



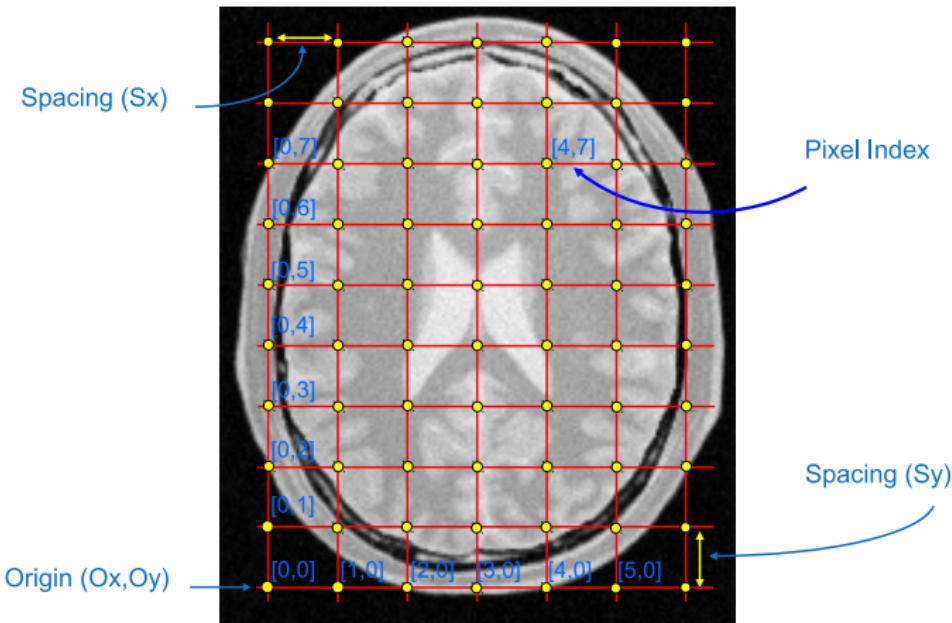
# Image Sampling Grid



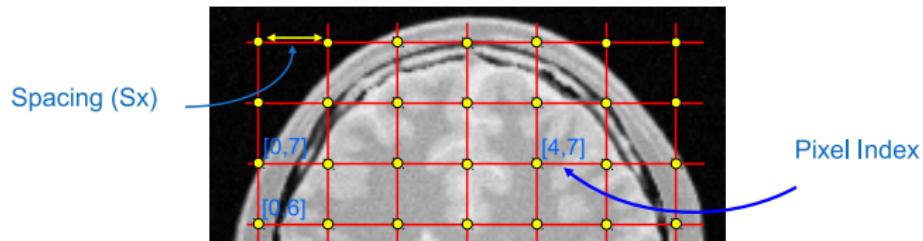
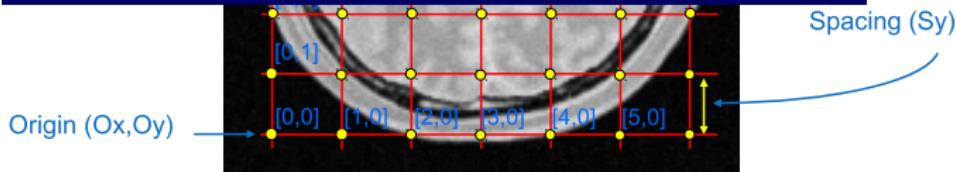
# Image Pixel



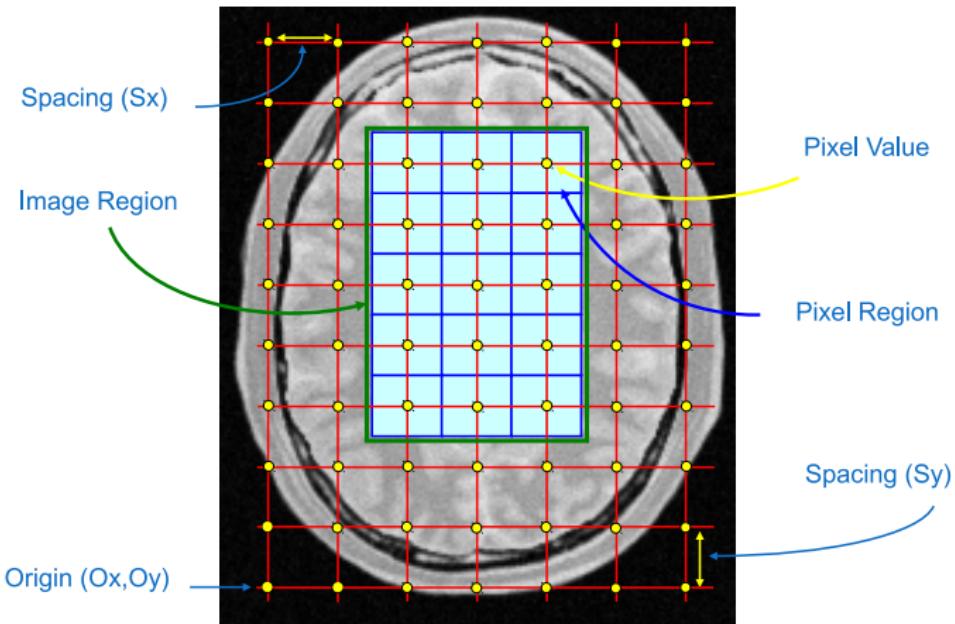
# Image Indices



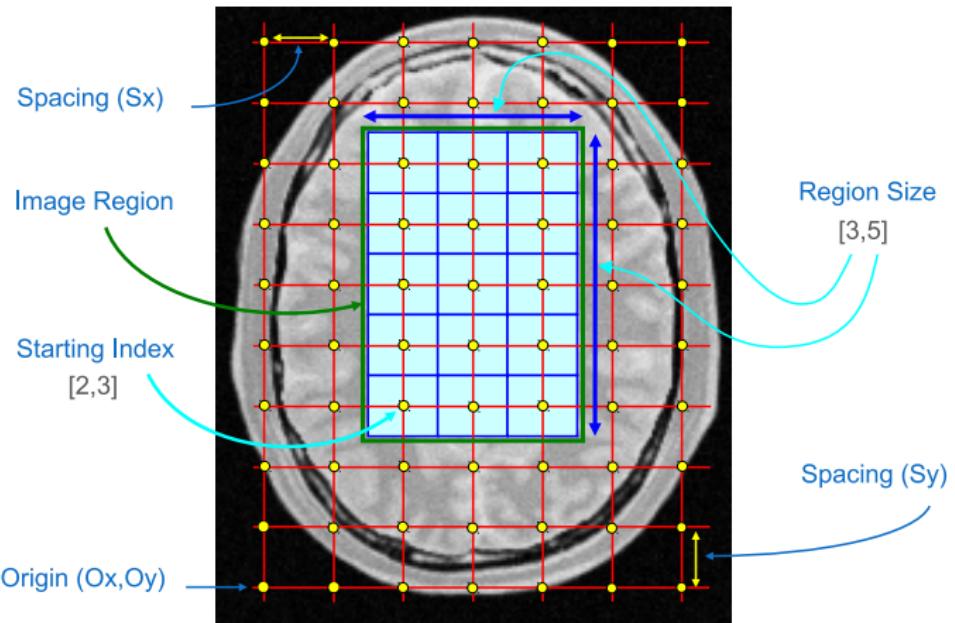
# Image To Physical Coordinates


$$P[0] = \text{Index}[0] \times \text{Spacing}[0] + \text{Origin}[0]$$
$$P[1] = \text{Index}[1] \times \text{Spacing}[1] + \text{Origin}[1]$$
$$\text{Index}[0] = \text{floor}((P[0] - \text{Origin}[0]) / \text{Spacing}[0] + 0.5)$$
$$\text{Index}[1] = \text{floor}((P[1] - \text{Origin}[1]) / \text{Spacing}[1] + 0.5)$$


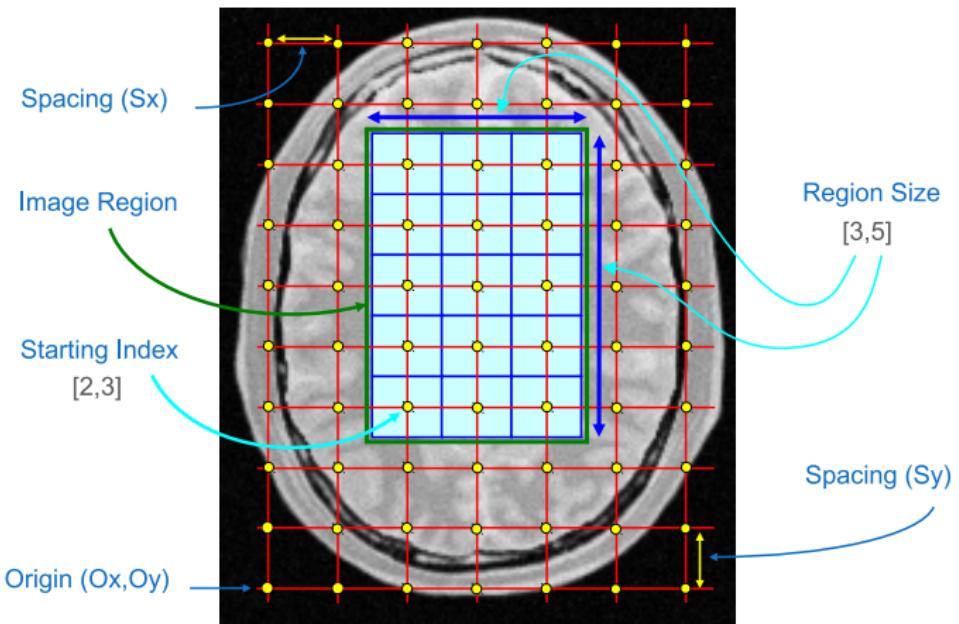
# Image Region



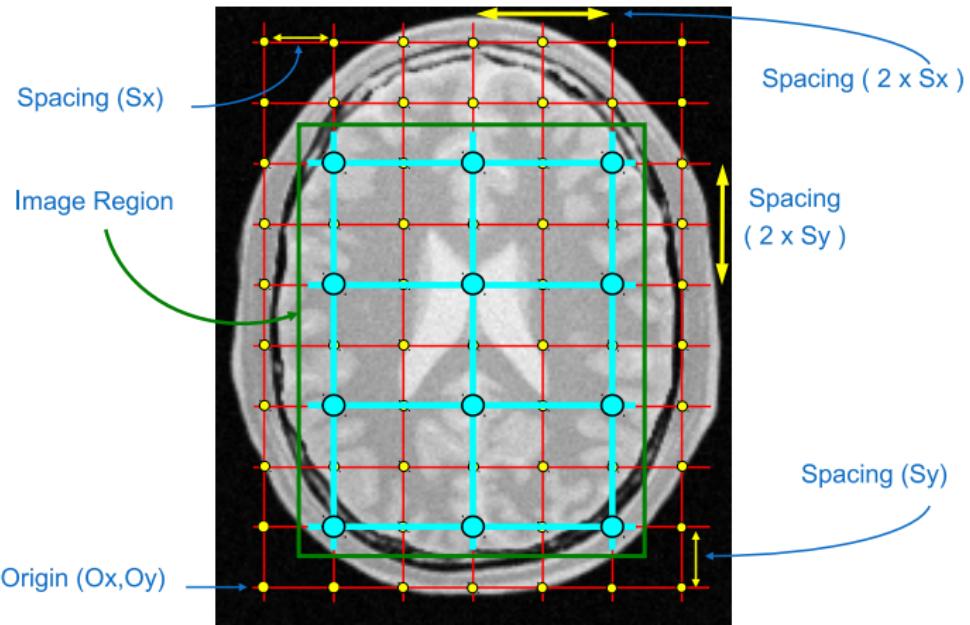
# Image Region: Size



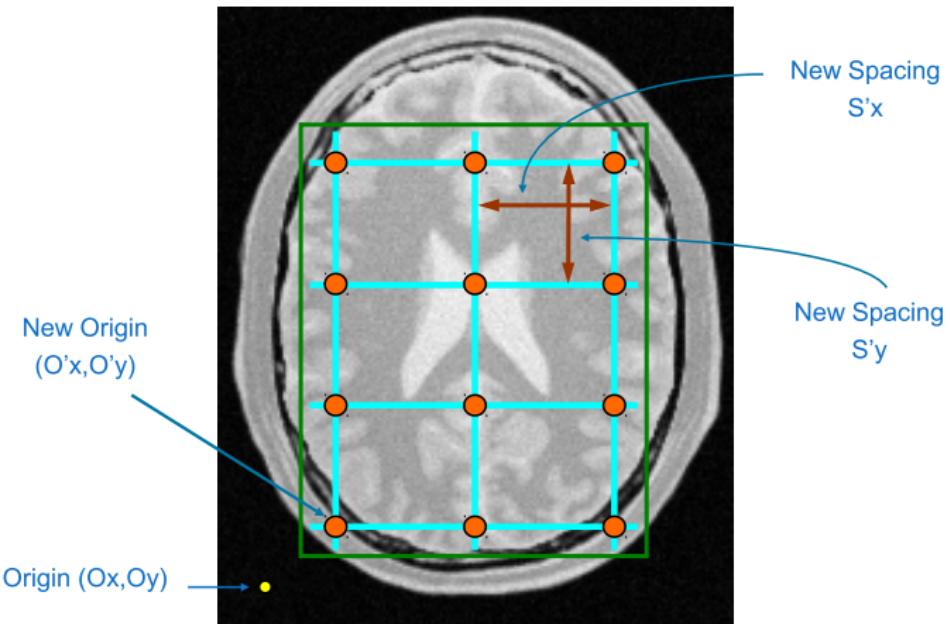
# Image Region: Starting Index



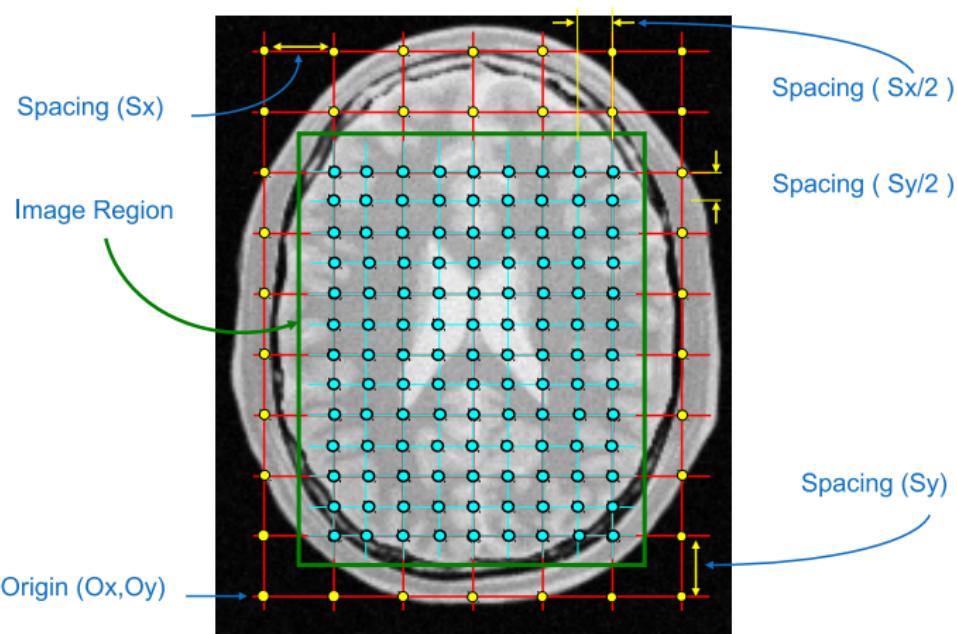
# Sub-sampling by Half



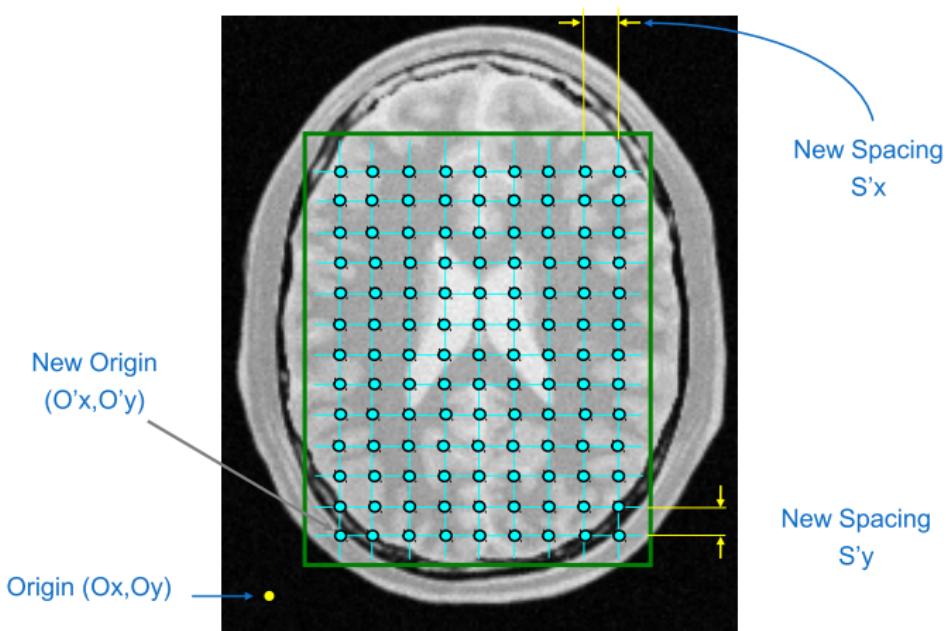
# Sub-sampling by Half



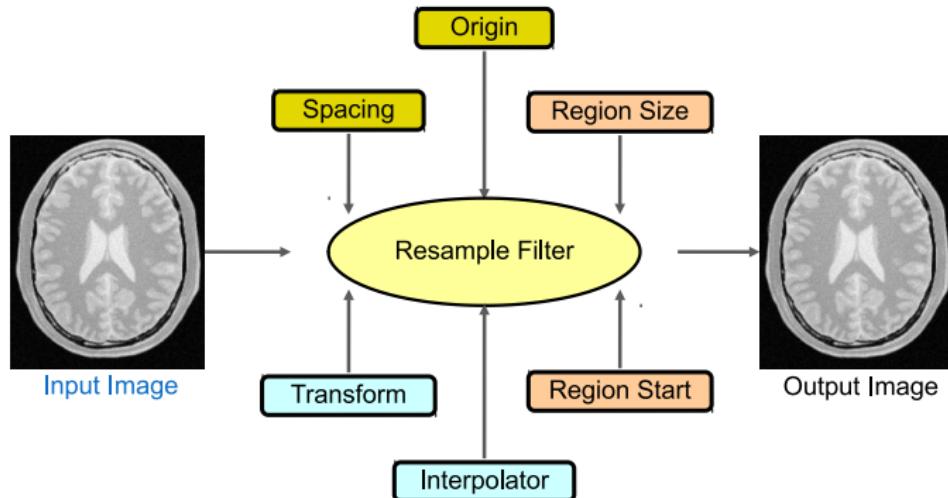
# Super-sampling by Double



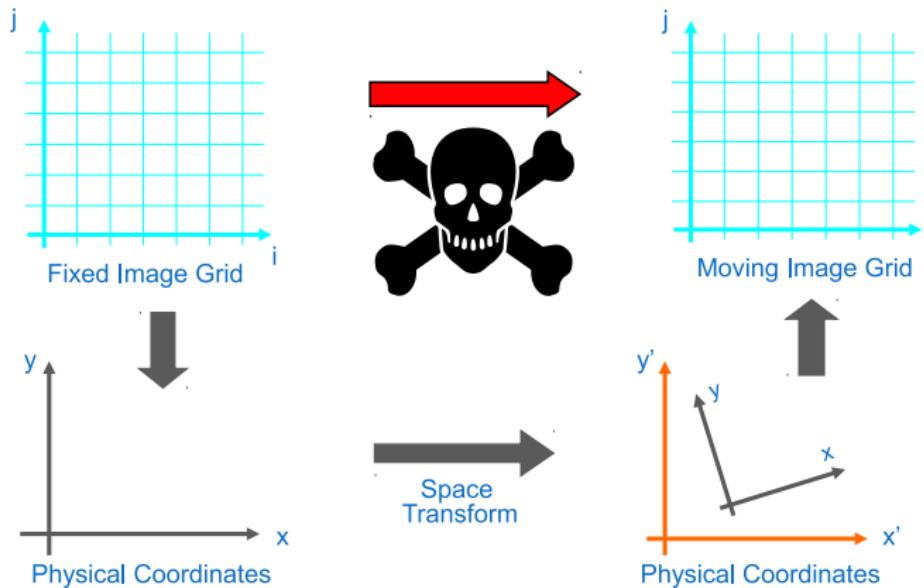
# Super-sampling by Double



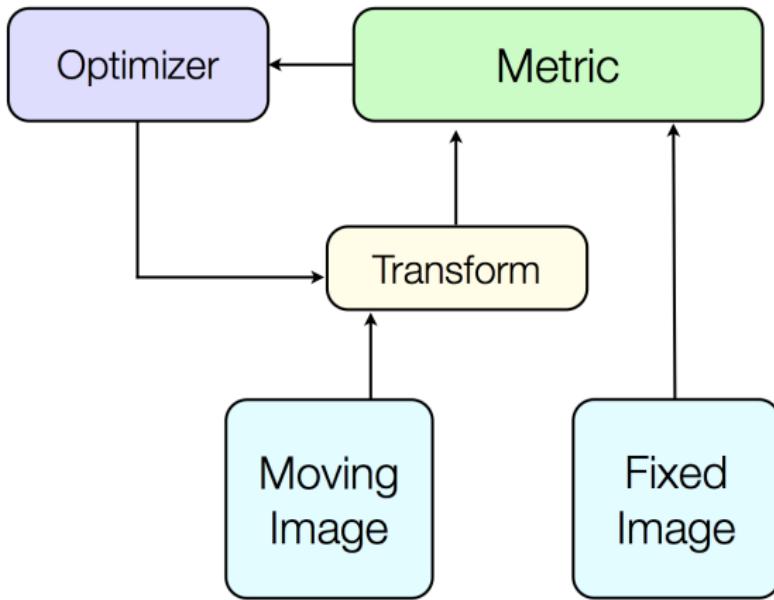
# Resampling In ITK



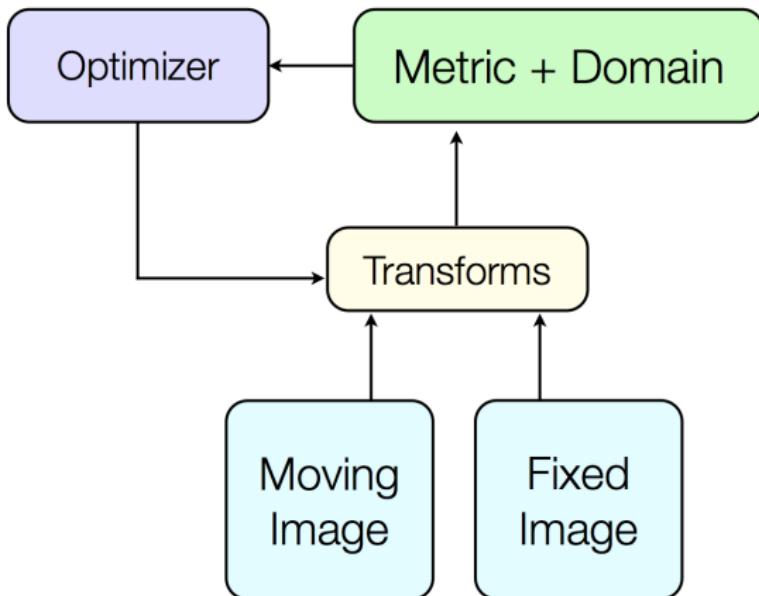
# Coordinate System Conversions



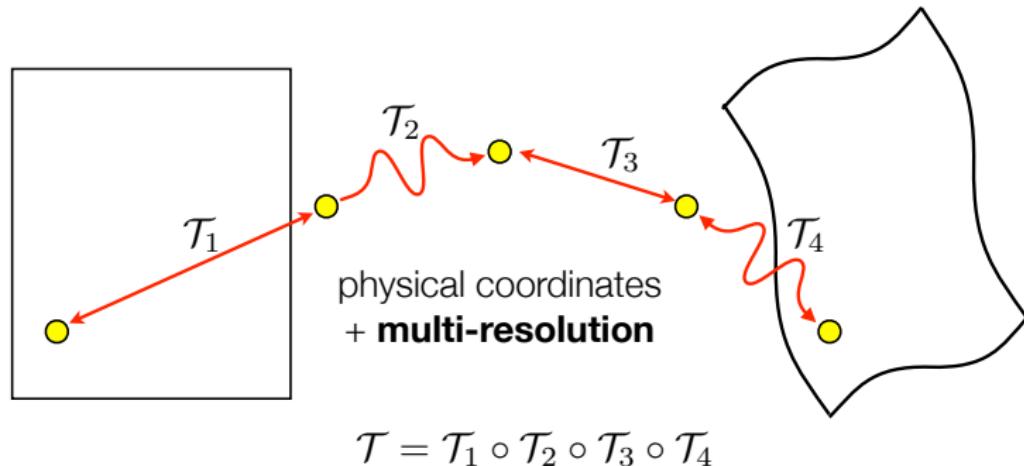
# ITK v3 framework



# ITK v4 framework



# Composite transformations



## Transformation Legend

- ↔ linear
- ↔ deformable
- ↔ symmetric deformable

To avoid compounding interpolation error with the concatenation of transformations, never use more than a single interpolation.

# Example Registration

## Example: Perform a simple registration

```
leafpad ~/src/ITKv4-TheNextGeneration-Tutorial/\
Exercises/Registration/RegistrationExercises.cxx &

ImageViewer ~/data/BrainProtonDensitySliceBorder20.png &
ImageViewer ~/data/BrainProtonDensitySliceR10X13Y17.png &
```

## Example: Perform a simple registration

```
./bin/RegistrationExercises ~/data/BrainProtonDensitySliceBor  
~/data/BrainProtonDensitySliceR10X13Y17.png \  
./Registered.png \  
3000  
  
ImageViewer ./Registered.png &
```

## Exercise: Do not scale the optimizer parameters

- In RegistrationExercises.cxx, **line 163**, turn off automatic optimizer parameter scaling.
- What happens and why?

# Insight Journal

The screenshot shows a web browser window for the Insight Journal website. The title bar reads "Insight Journal - Home - Chromium". The address bar shows the URL "insight-journal.org". The page content includes a logo for "Insight Journal" with a stylized eye icon, navigation links for "About | Register", "Home", "Journals", "Browse", "Submit", "Help", and "Blog", and a search bar for "Email". A main text block describes the journal as an Open Access online publication covering medical image processing and visualization. It lists unique characteristics such as open-access articles, peer-review, reproducible science, and continuous revision. A call-to-action encourages subscribing to the Kitware's newsletter. Below this, a news item is displayed about an OpenCL implementation of the Gaussian pyramid and the resampler, featuring logos for elastix, OpenCV, and ITK, and a "No Opinion" button.

**The Insight Journal** is an Open Access on-line publication covering the domain of medical image processing and visualization.

The unique characteristics of the Insight Journal include:

- Open-access to articles, data, code, and reviews
- Open peer-review that invites discussion between reviewers and authors
- Emphasis on reproducible science via automated code compilation and testing
- Support for continuous revision of articles, code, and reviews

Subscribe to the [Kitware's newsletter](#) to receive news about open-source on your desk, it's free!

**An OpenCL implementation of the Gaussian pyramid and the resampler**  
Shamonin D., Staring M.  
Nonrigid image registration is an important, but resource demanding and time-consuming task in medical image analysis. This limits its application in time-critical clinical routines. In this report we explore acceleration of two time-consuming parts of a [...] downloaded 88 times, viewed 683 times and no review.

## • Git Backend

- Git Backend
- Virtual Appliances

- Git Backend
- Virtual Appliances
- CDE

- Git Backend
- Virtual Appliances
- CDE
- Reproducible Research

- Submit patches
- Review patches
- <http://www.itk.org/Wiki/ITK/Git/Develop>
- <http://review.source.kitware.com>

status:open project:ITK | review.source Code Review - Chromium

status:open project:ITK | review.source.kitware.com/#/status:open+project:ITK,n,z

Home Docs Forums Lists Bugs Blogs Other Bookmarks

All Topics Changes Open Merged Abandoned Register Sign In Search status:open project:ITK

### Search for status:open project:ITK

ID	Subject	Owner	Project	Branch	Updated	V	R	BS
Ic67f8b76	BUG: Avoid using def file when BUILD_SHARED_LIBS is off	David Cole	ITK	master (avoid-def-file-for-static-builds)	12:19 AM	✓		
Ie9ed4151	WIP: Adding libmrc and updating MINC IO support.	Vladimir Fonov	ITK	master (MINC_IO_Modules)	Jan 22			
I153ca6c5	BUG: GradientVectorFlow should calculate timestep based on image dimension	Ho Cheung	ITK	master (GradientVectorFlowTimestepBug)	Jan 21			
I75d8c8d3	ENH: Add MaskingValue to MaskImageFilter	Julien Finet	ITK	master (2993-add-maskingvalue)	Jan 21			
I475ccc97	ENH: Add ThresholdInside and Negated to itkThresholdImageFilter	Julien Finet	ITK	master (2991-threshold-inside)	Jan 21	✗		
Ibdc58c38	ENH: Add unit test to itkThresholdImageFilterTest	Julien Finet	ITK	master (2991-threshold-inside)	Jan 20			
I02b2f4e4	COMP: Fix TIFF on big endian systems	Paul Novotny	ITK	master (tiff-big-endian)	Jan 19			
Ia15d8f2e	ENH: Get function accessible from const objects	Hans J. Johnson	ITK	master (AddGetConstObjectMacro)	Jan 18			
I6815f5c8	COMP: Update DCMTK version to incorporate new fixes.	kent williams	ITK	master (UPDATE_DCMTK_816a4060)	Jan 16			
I3459854e	WIP: Adding SubdivisionQuadEdgeMeshFilters	wanlin zhu	ITK	master (SubdivisionQuadEdgeMeshFilter)	Jan 16			

# Mailing Lists

- <http://www.itk.org/mailman/listinfo/insight-users>
- <http://www.itk.org/mailman/listinfo/insight-developers>

# Edit the Doxygen Documentation in the Browser

The screenshot shows a web browser window displaying the Doxygen documentation for the `itk::AbsImageAdaptor< TImage, TOutputPixelType >` class. The browser is identified as Chromium. The URL in the address bar is `www.itk.org/Doxygen/html/classitk_1_1AbsImageAdaptor.html`. The page header includes the ITK logo and the text "ITK 4.4.0 Insight Segmentation and Registration Toolkit". The navigation menu at the top has tabs for "Main Page", "Related Pages", "Modules", "Namespaces", "Classes" (which is selected), and "Search". Below the menu, there are links for "Class List", "Class Index", "Class Hierarchy", and "Class Members". The main content area shows the class template reference with the title `itk::AbsImageAdaptor< TImage, TOutputPixelType > Class Template Reference`. A red circle highlights the "Edit comments" link in the breadcrumb navigation bar. Other visible links include "Public Types", "Public Member Functions", "Static Public Member Functions", "Protected Member Functions", "Private Member Functions", and "List of all members". The bottom of the page features a "Detailed Description" section and standard browser navigation buttons.

# Edit the Doxygen Documentation in the Browser

ITK: itkAbsImageAdaptor.h - Chromium

ITK: itkAbsImageAdaptor.h

www.itk.org/editdoc/editcomments.php?file=itkAbsImageAdaptor.h

Home Docs Forums Lists Bugs Blogs Other Bookmarks

## itkAbsImageAdaptor.h

This page allow you to modify the comments of the library. Your request will create a request of patch which will be reviewed by Kitware.

To modify the comment, just click on what you want to modify. You can also add or delete comment.

When you are done, please submit this form to send your request.

Your email: \*

Comments:

Verify that you are a human, please choose lemon

Send >>

```
1. //=====
2. *
3. * Copyright Insight Software Consortium
4. *
5. * Licensed under the Apache License, Version
6. * you may not use this file except in compliance
7. * You may obtain a copy of the License at
8. *
9. * http://www.apache.org/licenses/LICENSE-2.0
10. *
11. * Unless required by applicable law or agreed
12. * distributed under the License is distributed
13. * WITHOUT WARRANTIES OR CONDITIONS OF ANY
14. * See the License for the specific language
15. * limitations under the License.
16. *
17. *=====
18. #ifndef __itkAbsImageAdaptor_h
19. #define __itkAbsImageAdaptor_h
```

The screenshot shows a web browser window for the ITK Examples v4.3.0 documentation. The title bar reads "Overview — ITK Examples v4.3.0 - Chromium". The address bar shows the URL "itk.org/ITKExamples/". The page content is titled "Welcome to the Insight Toolkit Examples!". It features a sidebar on the left with a logo, a quick search bar, and a "Go" button. The main content area is divided into several sections: "Learn" (with a sub-section "the extensive library of ITK examples"), "Do" (with a sub-section "how to locally build and run the examples"), "Download" (with a sub-section "download the examples or this documentation"), "Contribute" (with a sub-section "make corrections and add new examples"), "Search" (with a sub-section "quickly search the documentation"), "Lookup" (with a sub-section "index of all functions, classes, terms"), "Credits" (with a sub-section "who wrote the code"), and "Dashboard" (with a sub-section "software quality dashboard").

# Modularization

# Why Modularization

- Growth management

# Why Modularization

- Growth management
- Software quality improvement

# Why Modularization

- Growth management
- Software quality improvement
- Facilitating add-ons

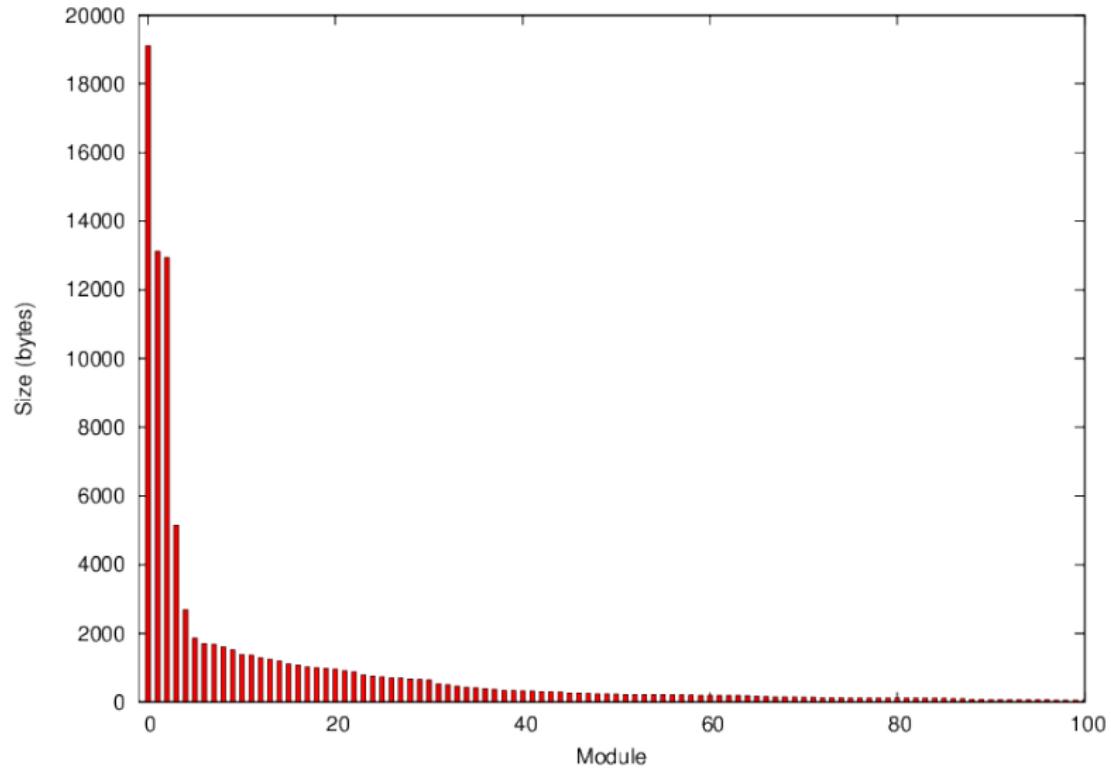
# Why Modularization

- Growth management
- Software quality improvement
- Facilitating add-ons
- Separate third party libraries

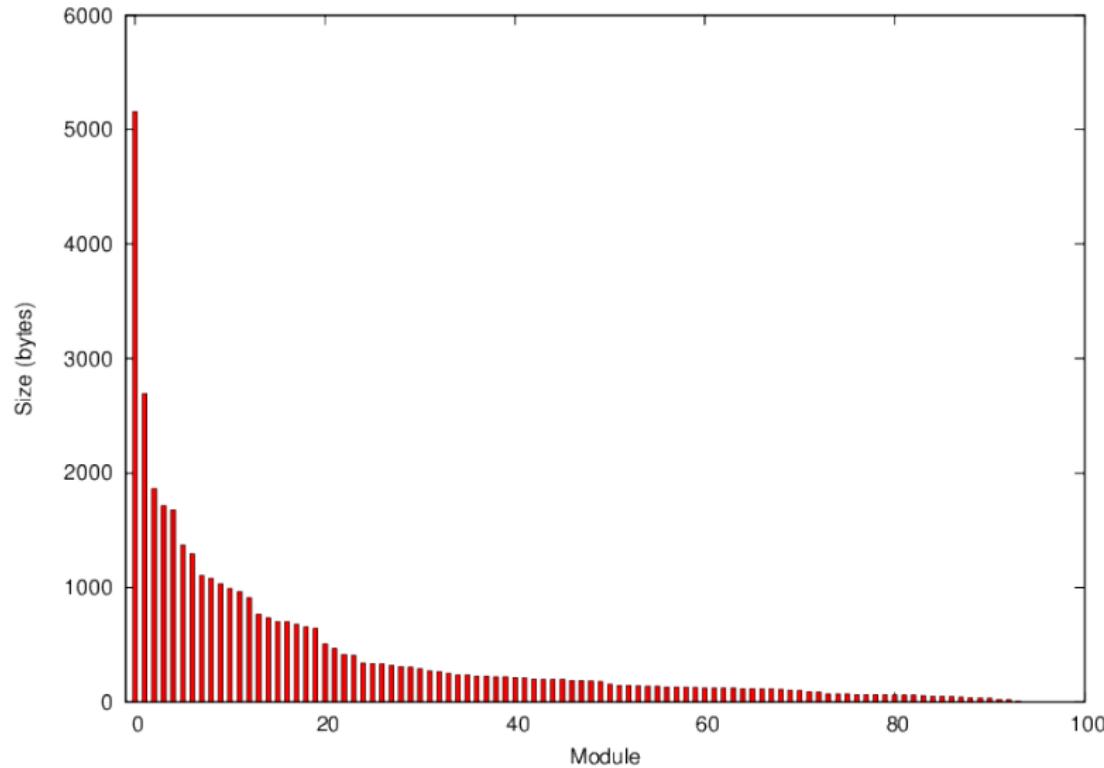
# Why Modularization

- Growth management
- Software quality improvement
- Facilitating add-ons
- Separate third party libraries
- Optional Components

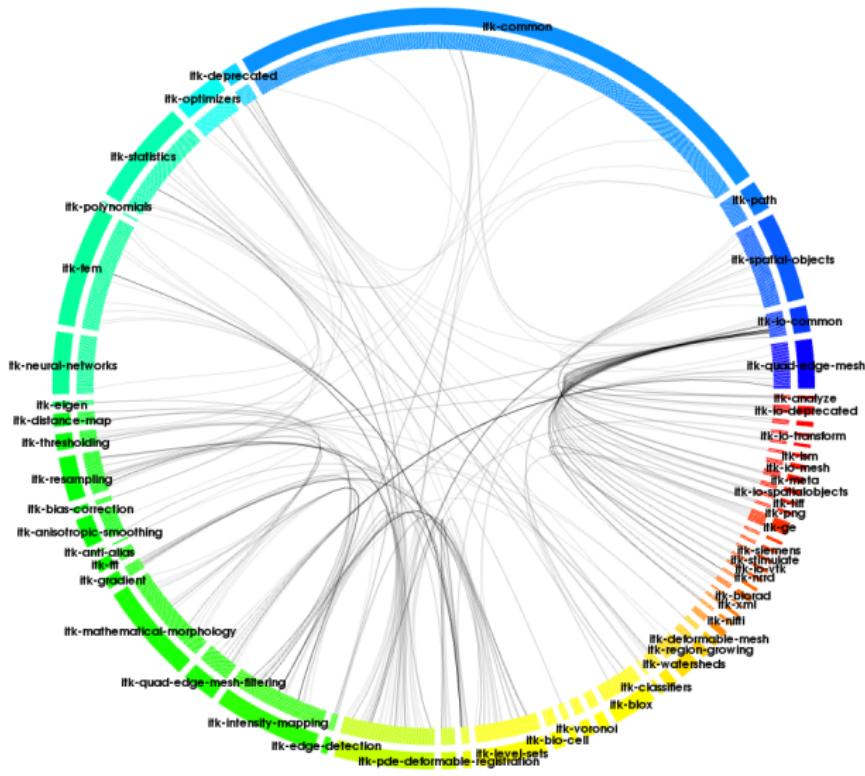
# Module sizes



# Module sizes (no third party modules)



# Visualize dependencies among modules



# ITK Module Grouping

- 12 groups, 107 modules and counting

# ITK Module Grouping

- 12 groups, 107 modules and counting
- Grouped by module functionality

# ITK Module Grouping

- 12 groups, 107 modules and counting
- Grouped by module functionality

Bridge	External	IO	Registration
Compatibility	Filtering	Nonunit	Segmentation
Core	GPU	Numerics	ThirdParty

# How to add a module into ITK

- ① Decide where to put the module

# How to add a module into ITK

- ① Decide where to put the module
- ② Create the CMake setup

# How to add a module into ITK

- ① Decide where to put the module
- ② Create the CMake setup
- ③ Add its content

# Modularization checklist

- **Module categorization:** module group, module name

# Modularization checklist

- **Module categorization:**
  - Group is specified by the directory

# Modularization checklist

- **Module categorization:**

- Group is specified by the directory
- Module name is specified by directory, `itk-module.cmake`

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:** include, src, test

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:**
  - **include:** Headers defining the API, \*.hxx template implementations

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:**
  - **include:** Headers defining the API, \*.hxx template implementations
  - **src:** \*.cxx implementations, create a library

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:**
  - **include:** Headers defining the API, \*.hxx template implementations
  - **src:** \*.cxx implementations, create a library
  - **test:** Module unit tests

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:** include, src, test
- **CMakeLists.txt** top of module, src, test

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:** include, src, test
- **CMakeLists.txt**
  - **top of module:** project(), itk\_module\_impl(),  
set(ITKFoo\_LIBRARIES ITKFoo)

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:** include, src, test
- **CMakeLists.txt**
  - **top of module:** project(), itk\_module\_impl(),  
set(ITKFoo\_LIBRARIES ITKFoo)
  - **src:** add\_library(),  
target\_link\_library(),itk\_module\_target()

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:** include, src, test
- **CMakeLists.txt**
  - **top of module:** project(), itk\_module\_impl(),  
set(ITKFoo\_LIBRARIES ITKFoo)
  - **src:** add\_library(),  
target\_link\_libraries(),itk\_module\_target()
  - **test:** itk\_module\_test(), itk\_add\_test()

# Modularization checklist

- **Module categorization:** module group, module name
- **Directory hierarchy:** include, src, test
- **CMakeLists.txt** top of module, src, test
- **Module dependencies and documentation** `itk-module.cmake`

# External Modules

- Externally distributed ITK modules.
- Share the same modular structure as other internal modules.
- Needs to be manually downloaded to be used as a module.
- Facilitate code sharing and diverse development.
- Refer to: Lesion Sizing Toolkit Module  
(Modules/External/LesionSizingToolkit)

# Remote Modules

- Modules distributed remotely.
- Can be fetched through cmake automatically(Super build).
- With version control.
- Should have the same code quality standards as other internal ITK modules.

# Exercise: ITKRAT

- Insight Journal article on `itkRobustAutomaticThresholdImageFilter(RAT)`
- `/Documents/ITKv4-TheNextGeneration-Tutorial/Exercises/Modularization/IJ-submission-rat`
- Make into an External module: ITKRAT

## More information on modularization

- Details about ITK Modularization can be found at [Wiki Page](#).

Live Long  
and  
Prosper !