ITK Images

BugSquashing Seminar 21.06.16
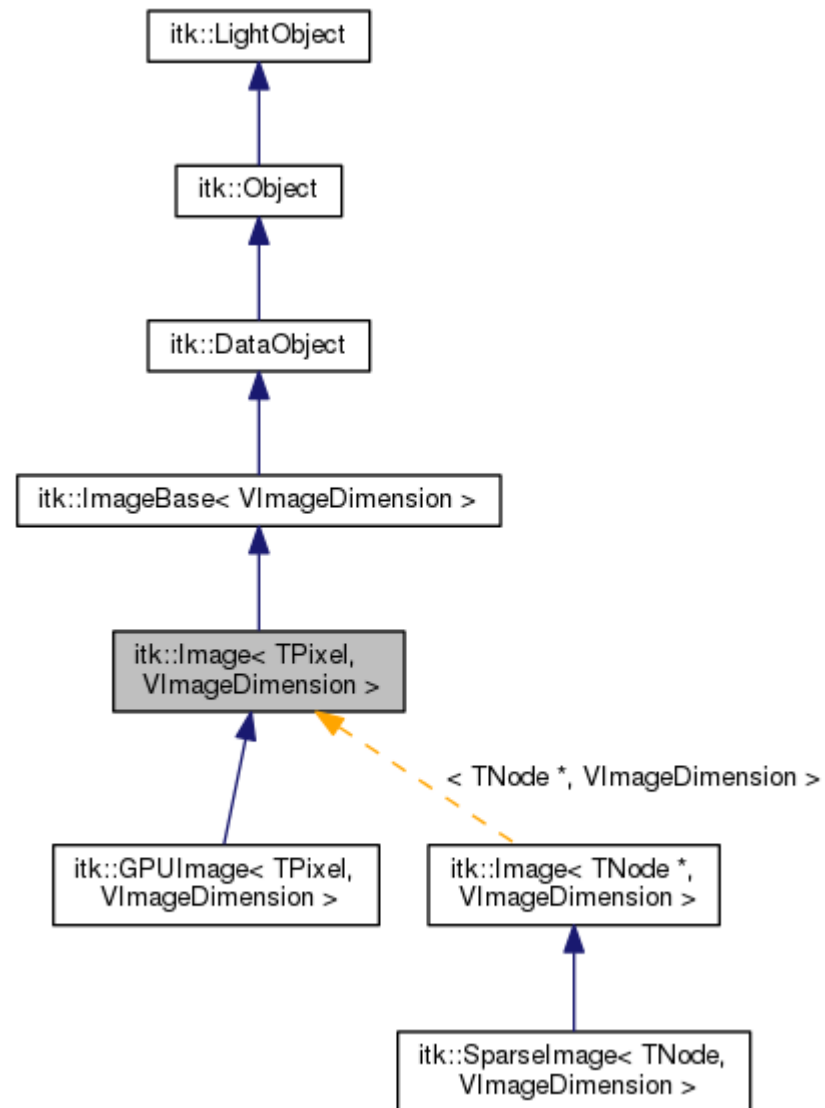
Tobias Norajitra

dkfz. GERMAN
CANCER RESEARCH CENTER
IN THE HELMHOLTZ ASSOCIATION

Today: some basics about itk::Image

dkfz.

1.   Relevant classes: Image, ImageBase, DataObject

2.   ImageBase: Geometry and coordinate transforms

3.   Image: access the pixel data

4.   Some use cases:
   - Initializing images
   - Access via iterator
   - World coordinates to image coordinates

- Transition to mitk::Image with the according adaptor classes (*see:* "*Using ITK images in MITK*" by Joseph Görres")
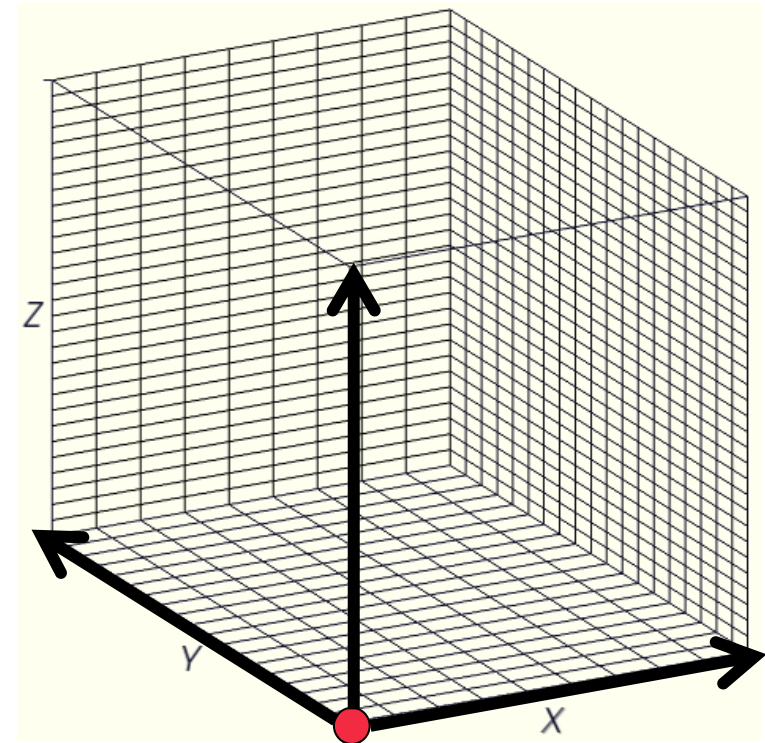
1. Relevant classes

- DataObject: used in itk-pipeline by ProcessObject

- ImageBase: geometry and coordinate transformations

- Image: access the pixel data

- ImageIterators: conveniently visit the pixel data

itk::LightObject

itk::Object

itk::DataObject

itk::ImageBase< VImageDimension >

itk::Image< TPixel, VImageDimension >

itk::GPUImage< TPixel, VImageDimension >

itk::Image< TNode *, VImageDimension >

< TNode *, VImageDimension >

itk::SparseImage< TNode, VImageDimension >

2. ImageBase class

- Origin (Point-Type)

- Direction (Matrix-Type)

- Spacing (Vector-Type)

- Regions (ImageRegion)
  - ImageRegion: Index and Size
  - LargestPossibleRegion
  - RequestedRegion

- Coordinate transformations:

```
bool   TransformPhysicalPointToContinuousIndex (const Point< TCoordRep, VImageDimension > &point, ContinuousIndex< TIndexRep, VImageDimension > &index) const
void   TransformContinuousIndexToPhysicalPoint (const ContinuousIndex< TIndexRep, VImageDimension > &index, Point< TCoordRep, VImageDimension > &point) const

bool   TransformPhysicalPointToIndex (const Point< TCoordRep, VImageDimension > &point, IndexType &index) const
void   TransformIndexToPhysicalPoint (const IndexType &index, Point< TCoordRep, VImageDimension > &point) const
```
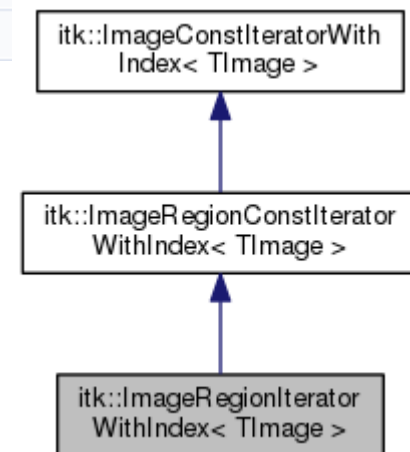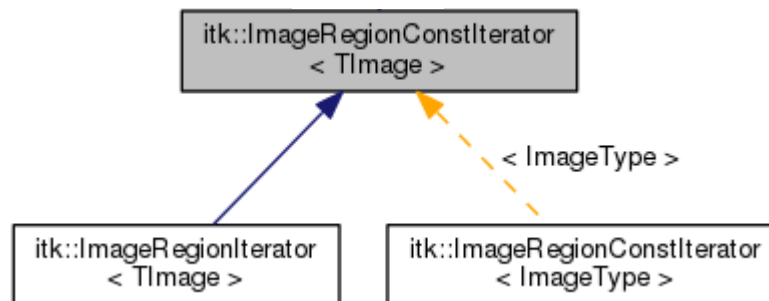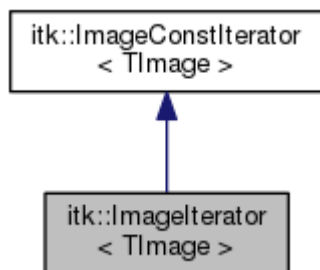
- itk::Image< PixelType, Dimension >

- Access data via: Pixel-Access, []-Operator, PixelContainer

const TPixel &	**GetPixel** (const **IndexType** &index) const
TPixel &	**GetPixel** (const **IndexType** &index)
AccessorType	**GetPixelAccessor** (void)
const **AccessorType**	**GetPixelAccessor** (void) const
PixelContainer *	**GetPixelContainer** ()
const **PixelContainer** *	**GetPixelContainer** () const
virtual void	**Graft** (const **DataObject** *data) override
virtual void	**Initialize** () override
TPixel &	**operator[]** (const **IndexType** &index)
const TPixel &	**operator[]** (const **IndexType** &index) const

- Or: Access via iterator

4. Some use cases

- Some type declarations first

```
typedef itk::Image<short, 3>                                      ImageType;
typedef itk::ImageFileReader< ImageType >                         ImageReaderType;
typedef itk::ImageFileWriter< ImageType >                         ImageWriterType;
typedef itk::Point< double, 3 >                                   PointType;
typedef itk::ImageRegionConstIteratorWithIndex< SegmentationType >   ConstIteratorType;
```

- Initialize

```
ImageType::Pointer map = ImageType::New();
map->SetOrigin( img->GetOrigin() );
map->SetSpacing( img->GetSpacing() );
map->SetDirection( img->GetDirection() );
map->SetRegions( img->GetLargestPossibleRegion() );
map->Allocate();
map->FillBuffer( 0 );
```

- Access via iterator

```
ConstIteratorType it( segmentation, segmentation->GetLargestPossibleRegion() );
while ( !it.IsAtEnd() )
{
        ImageType::PixelType val = it.Get();
        ImageType::IndexType ind = it.GetIndex();
        if ( val > 0 && map->GetLargestPossibleRegion().IsInside( ind ) )
        {
                ImageType::PixelType &count = map->GetPixel( it.GetIndex() );
                ++count;
        }
        ++it;
}
```

- Coordinate transformation

```
PointType pt;
m_Image->TransformIndexToPhysicalPoint( ind, pt );
pt += vec;
VotingMapType::IndexType id;
m_Image->TransformPhysicalPointToIndex( pt, id );
```

Questions?

dkfz. **GERMAN CANCER RESEARCH CENTER** IN THE HELMHOLTZ ASSOCIATION