Summary of the RTK Course

A Little History

From Kitware

- First there was VTK - visualisation toolkit
  General purpose visualisation

- For this they invented Cmake - which became more popular that anything else they made :-)

- Then there was ITK - insight toolkit
  For segmentation and registration
  (of medical data like the visible human)

- Building on top is RTK - reconstruction toolkit
  Circular CT reconstruction
  not directly from Kitware - might become a module in the future

Registration & Segmentation

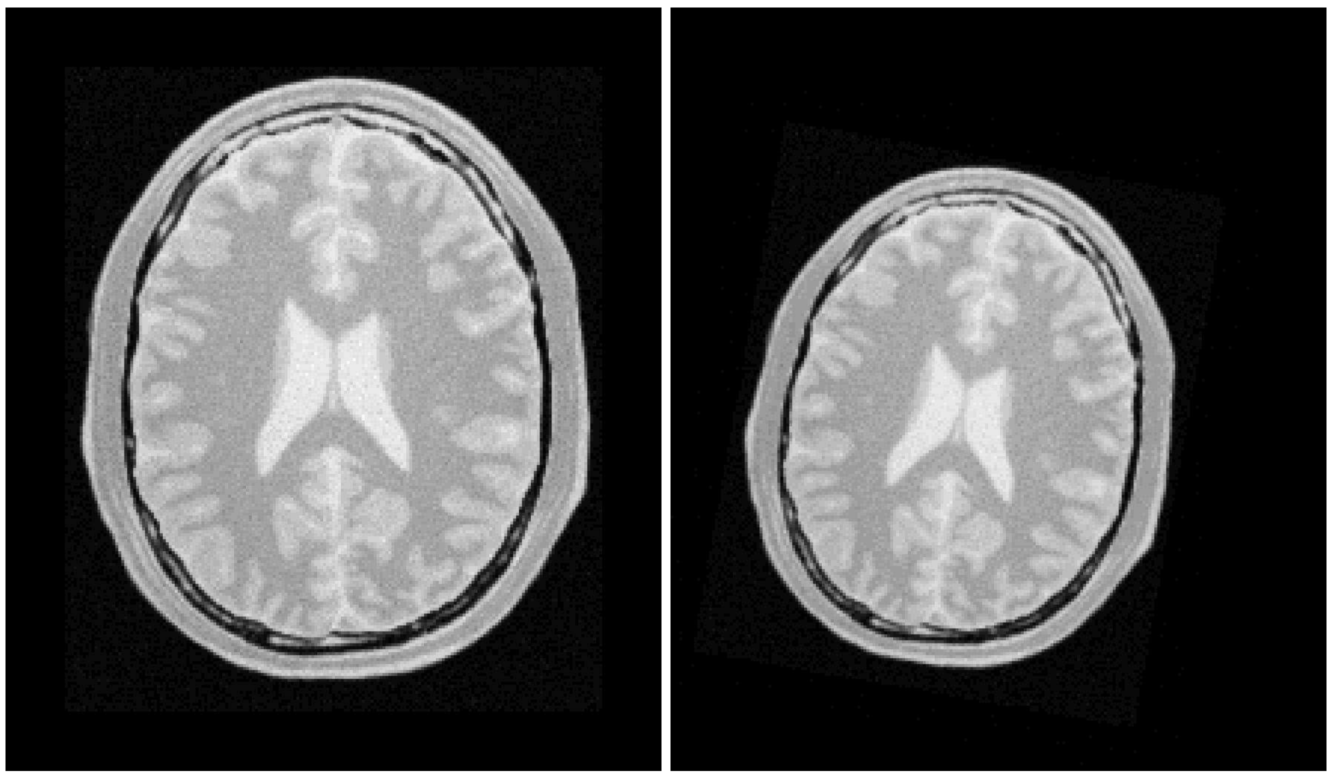


Figure 3.24: Fixed and Moving image provided as input to the registration method using the Similarity2D transform.

ITK

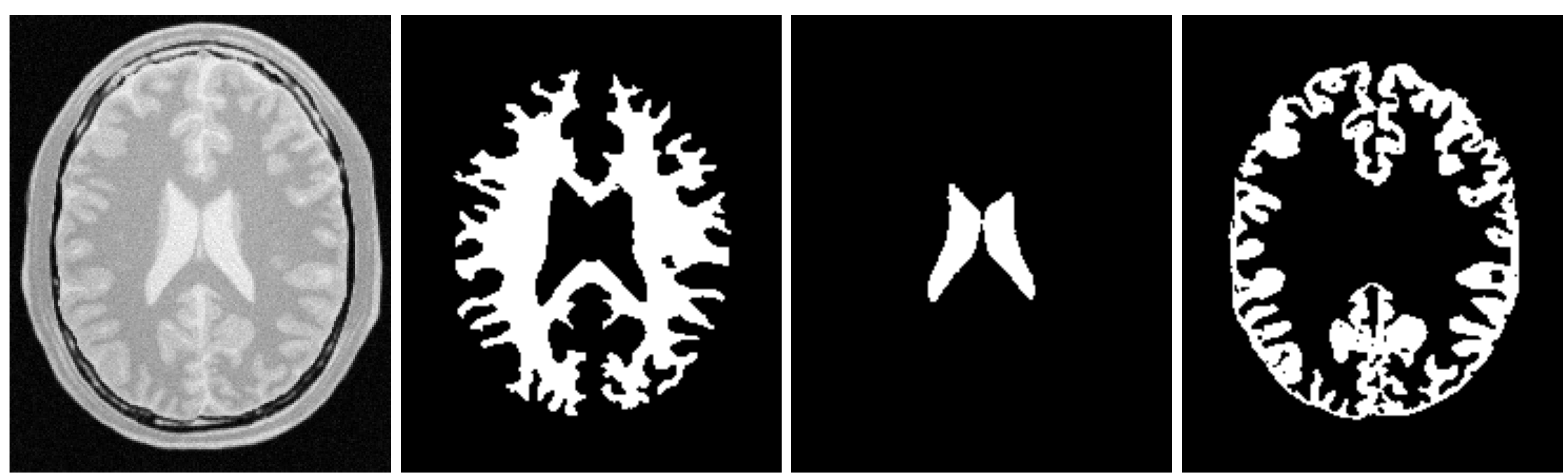


Figure 4.1: Segmentation results for the ConnectedThreshold filter for various seed points.

Coordinate systems

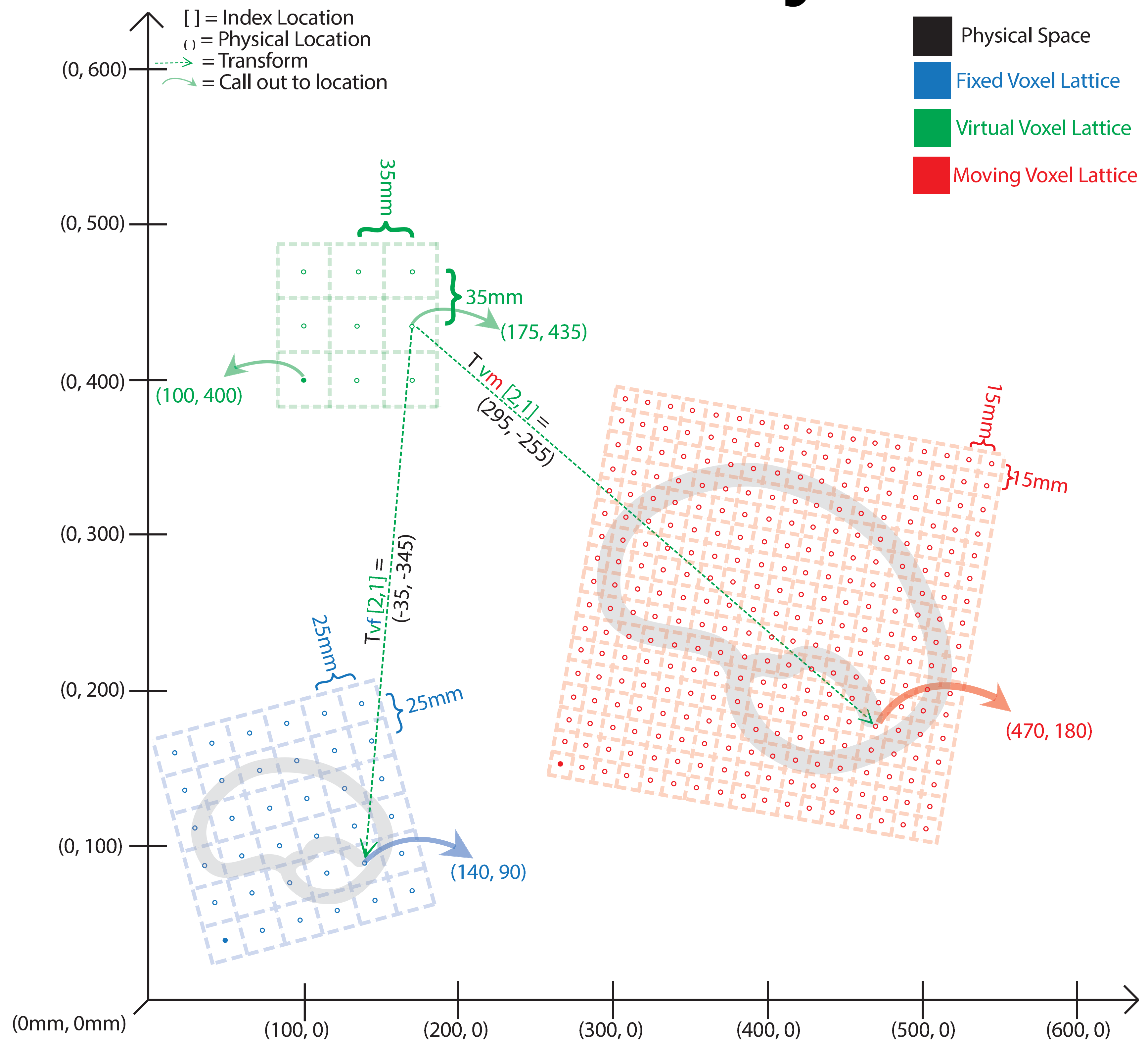


Figure 3.8: Different coordinate systems involved in the image registration process. Note that the transform being optimized is the one mapping from the physical space of the **virtual** image into the physical space of the **moving** image.

Spatial Objects

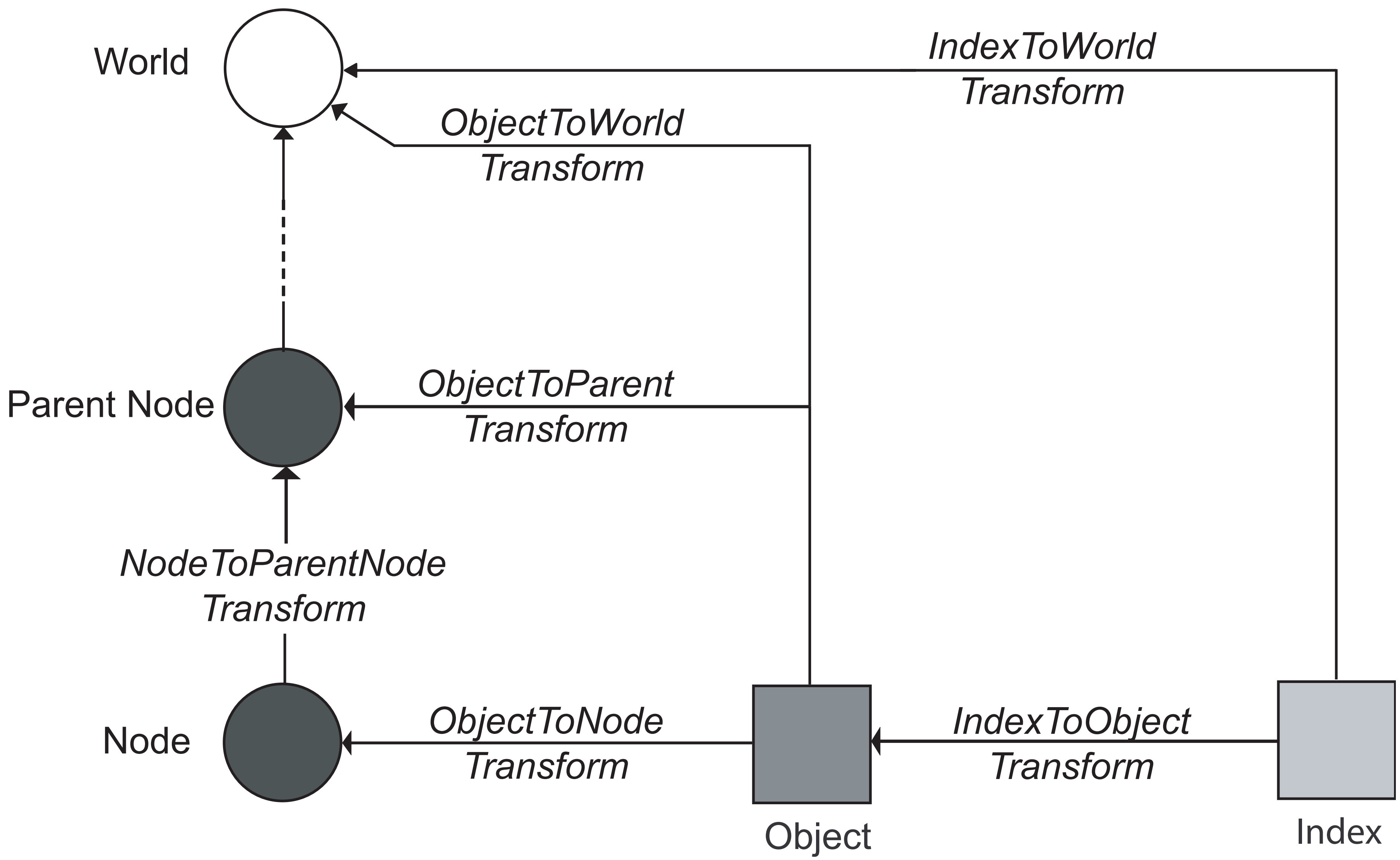


Figure 5.1: Set of transformations associated with a Spatial Object

Image Segmentation

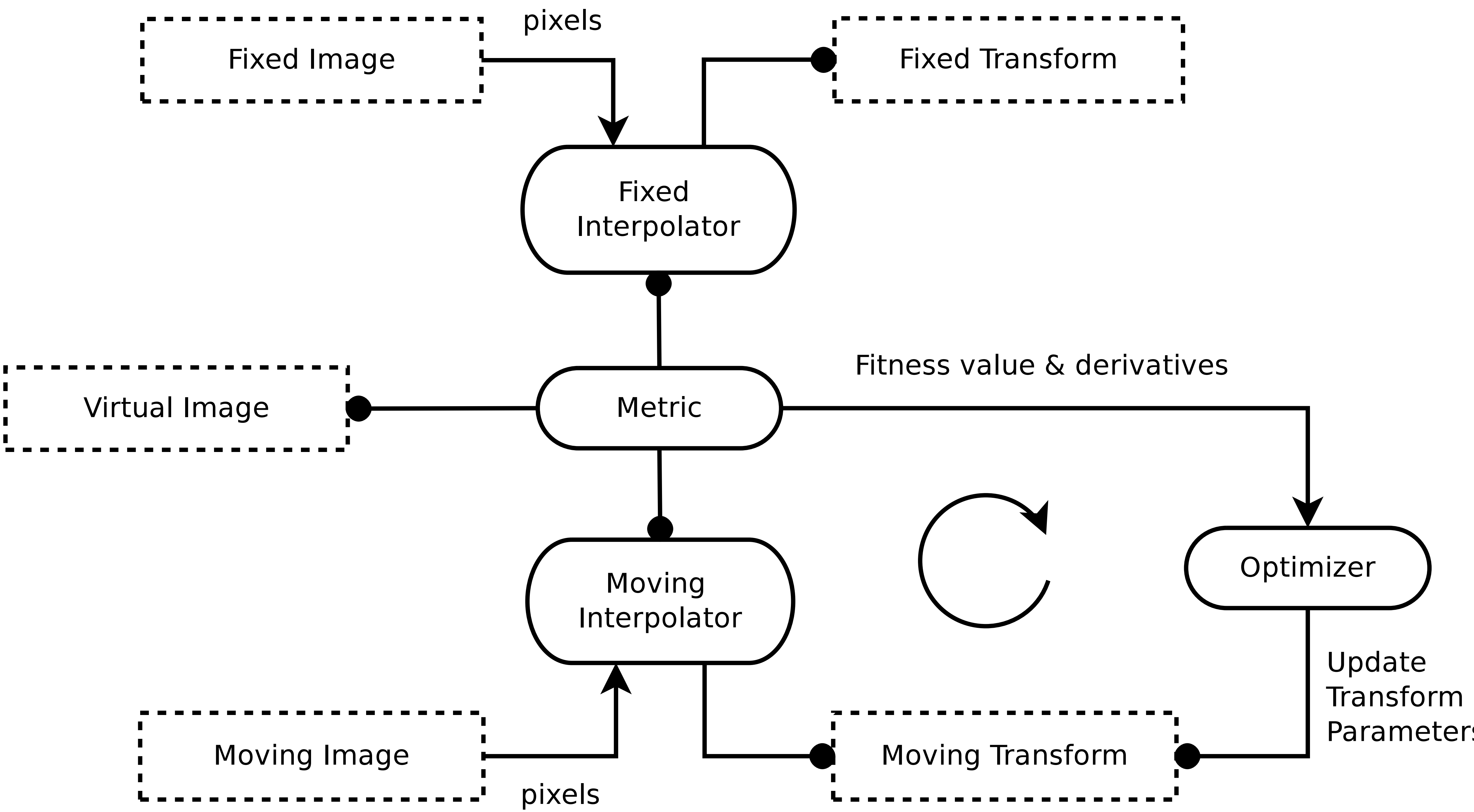


Figure 3.3: The basic components of the ITKv4 registration framework.

Some Computing Concepts

- It's C++ (and some Python bindings)
- Generic programming = heavy use of templates
  For optimal performance
- Object Factories = no constructors or destructors,
  but every class has a New() function
  E.g. for instantiation of hardware based filters
- Smart pointers - for memory management via reference counting
  (own implementation predating C++11)
- Error handling via Exceptions - not so hip anymore ;-)
- Command/Observer pattern - like Qt signal/slot mechanism
  multiple objects can watch and act on a certain event or action

Basic Data Structure

- itk::Image (and itk::Mesh - not used here)
- ”represents an *n*-dimensional, regular sampling of data. The sampling direction is parallel to direction matrix axes, and the origin of the sampling, inter-pixel spacing, and the number of samples in each direction (i.e., image dimension) can be specified. The sample, or pixel, type in ITK is arbitrary ”

itk::Image physical information

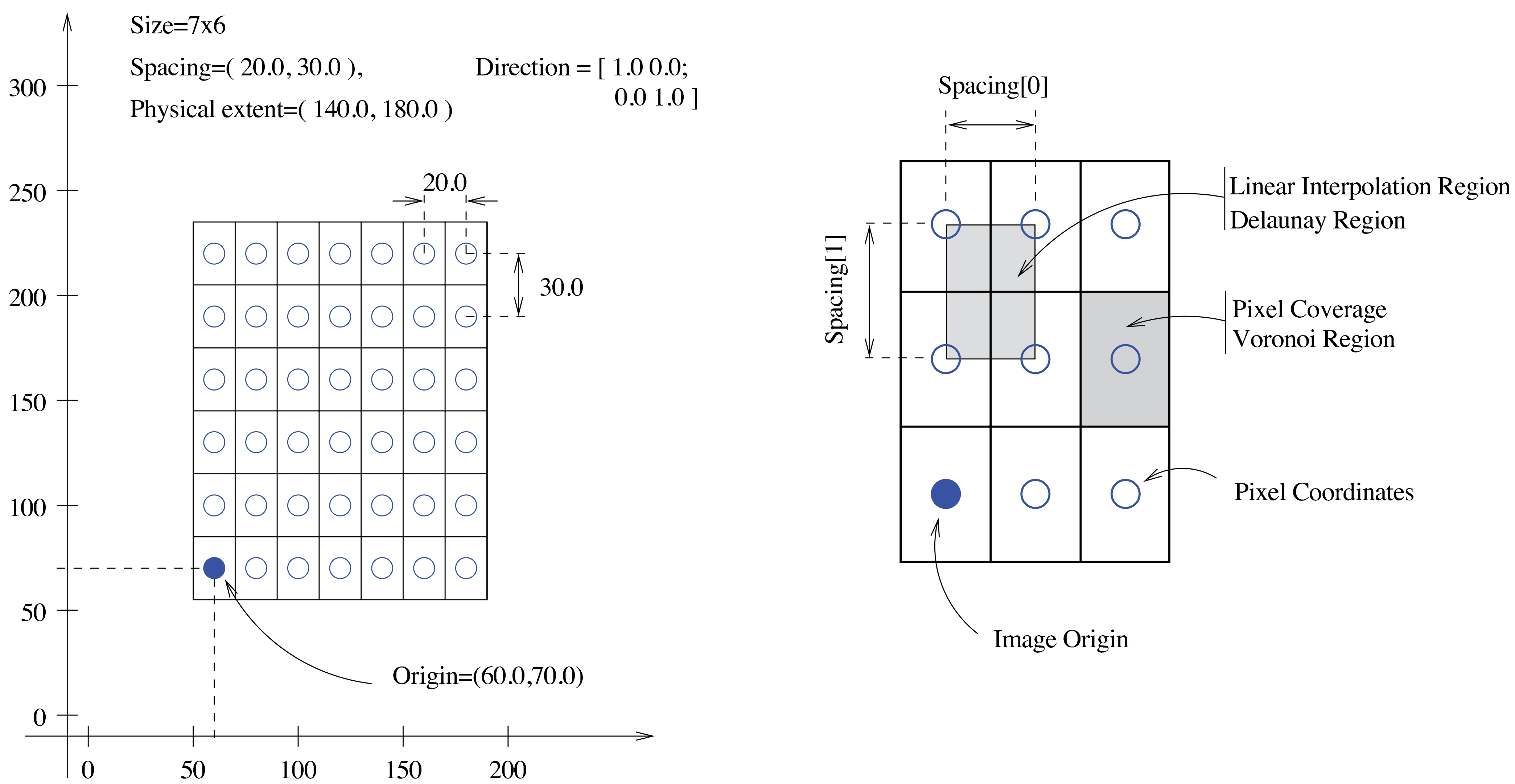


Figure 4.1: Geometrical concepts associated with the ITK image.

itk::Image regions

- LargestPossibleRegion: the image in its entirety.
- BufferedRegion: the portion of the image retained in memory.
- RequestedRegion: the portion of the region requested by a filter or other class when operating on the image.

Basic Workflow

- Because using variable regions and multithreading and GPUs, it’s a streaming model

- There are data objects (images)

- And processing objects (reader, writer, filter) applied to the data objects

itk::ImageSource

A filter that creates (generates or reads from disk) an itk::Image

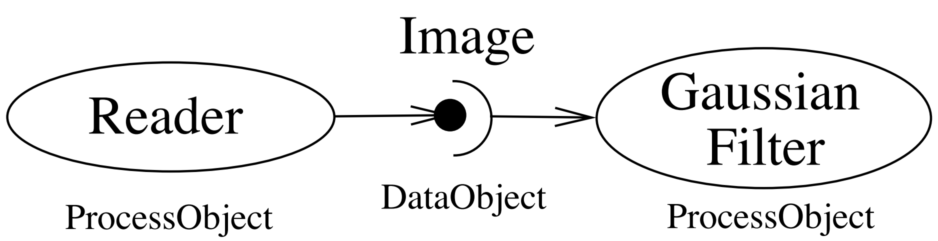


Figure 8.1: Relationship between DataObject and ProcessObject.

http://www.itk.org/Doxygen/html/classitk_1_1ImageSource.html

Example Workflow

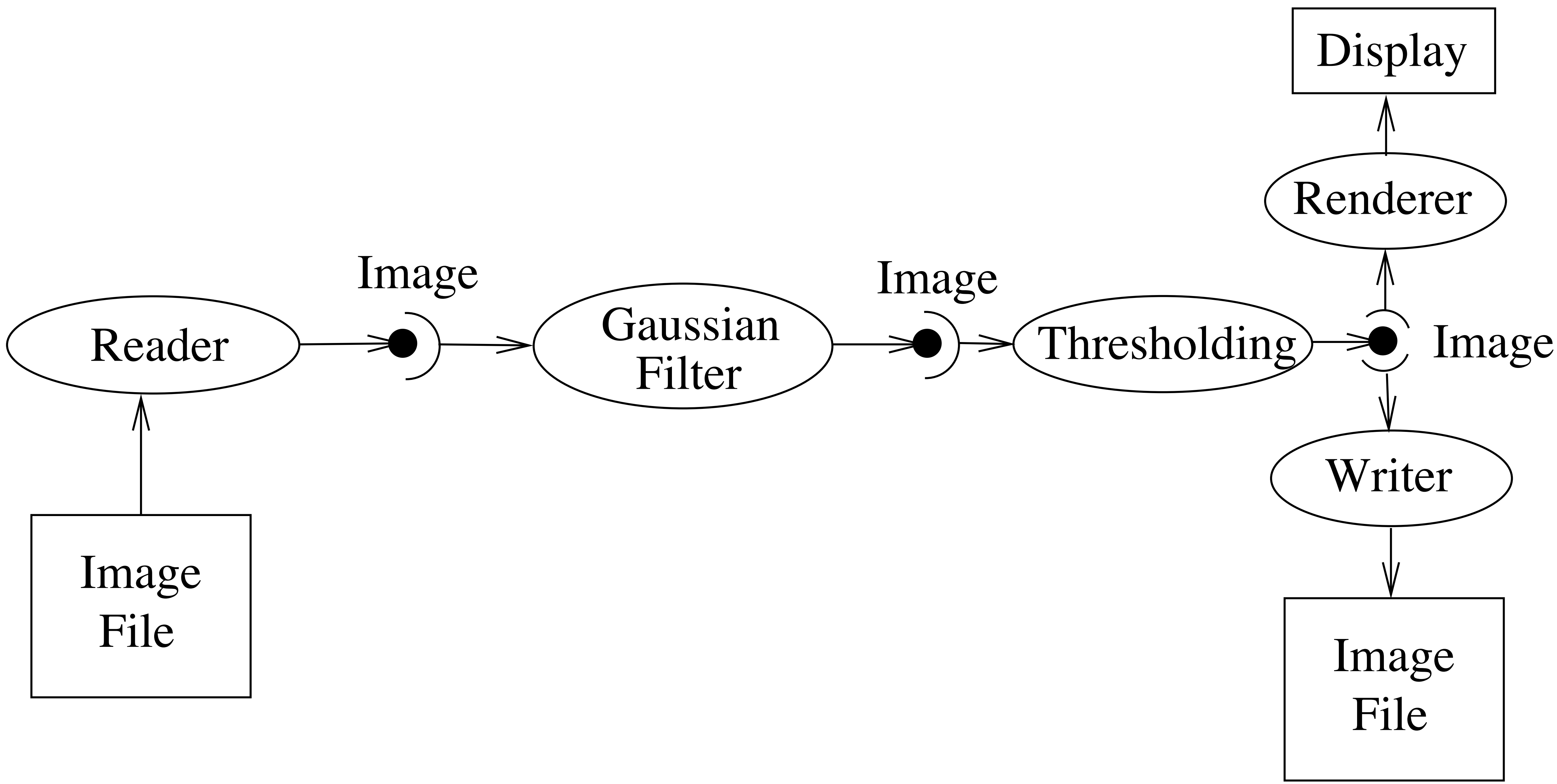


Figure 8.2: The Data Pipeline

Update mechanism

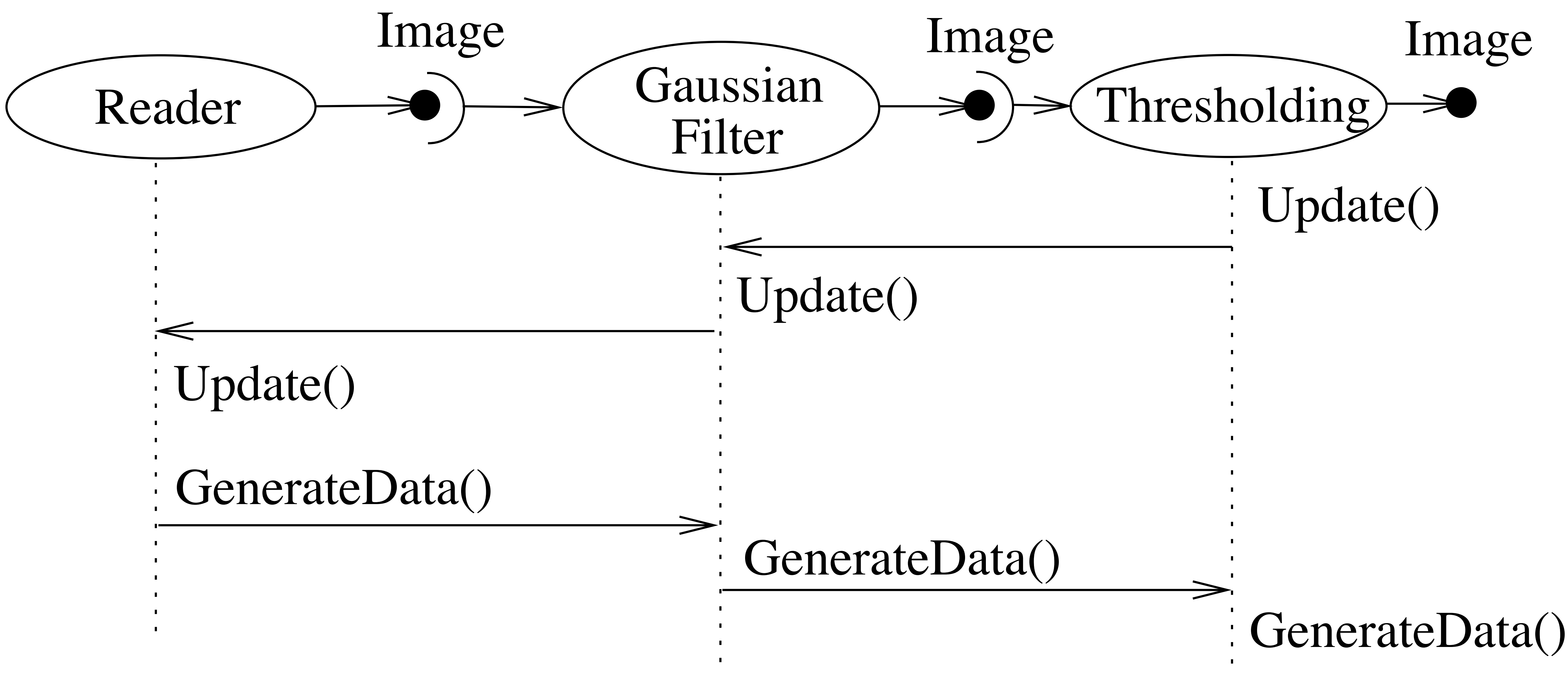
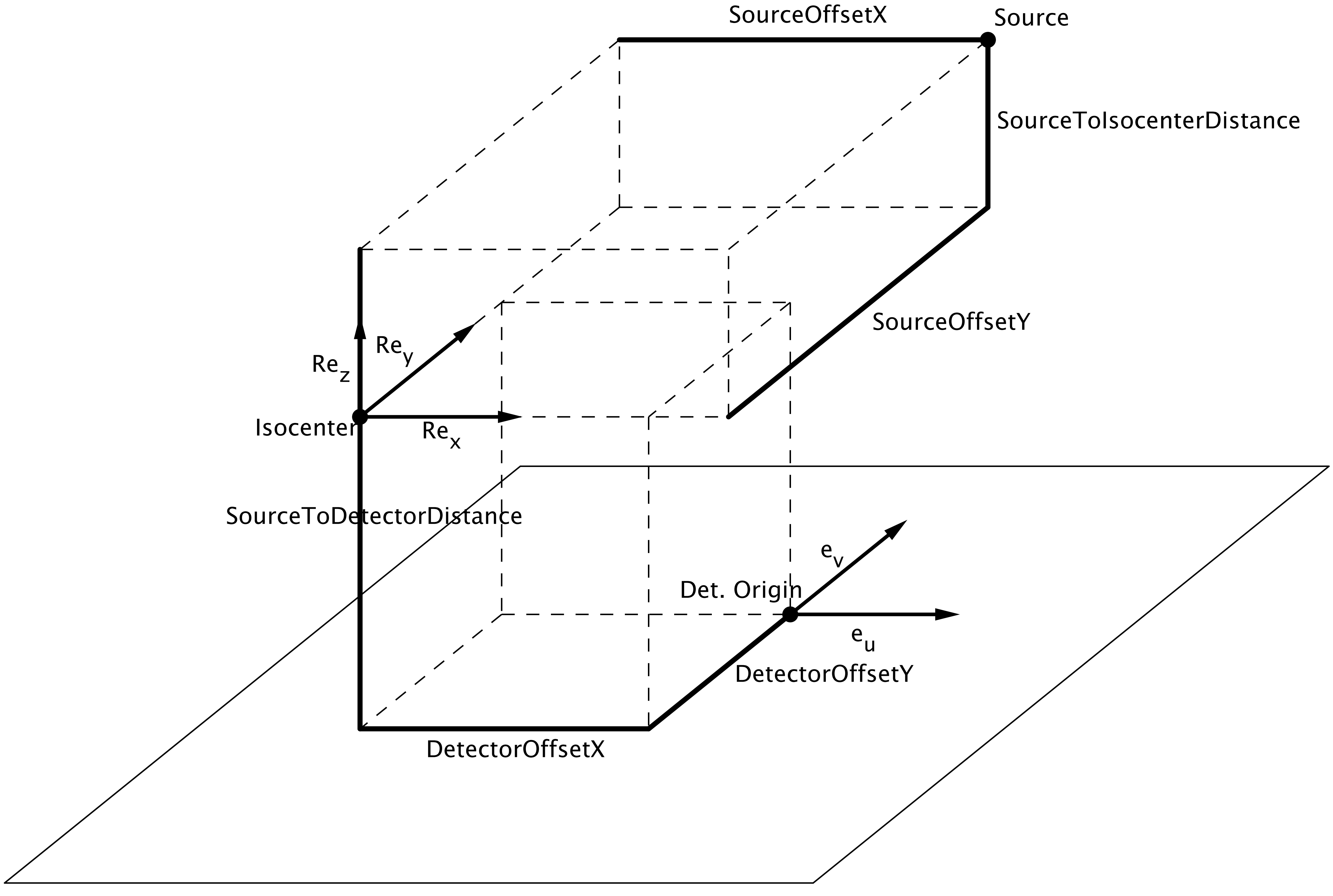


Figure 8.3: Sequence of the Data Pipeline updating mechanism

What is RTK?

`http://www.openrtk.org/`

An open-source and cross-platform toolkit for fast circular cone-beam CT reconstruction based on the Insight Toolkit (ITK)

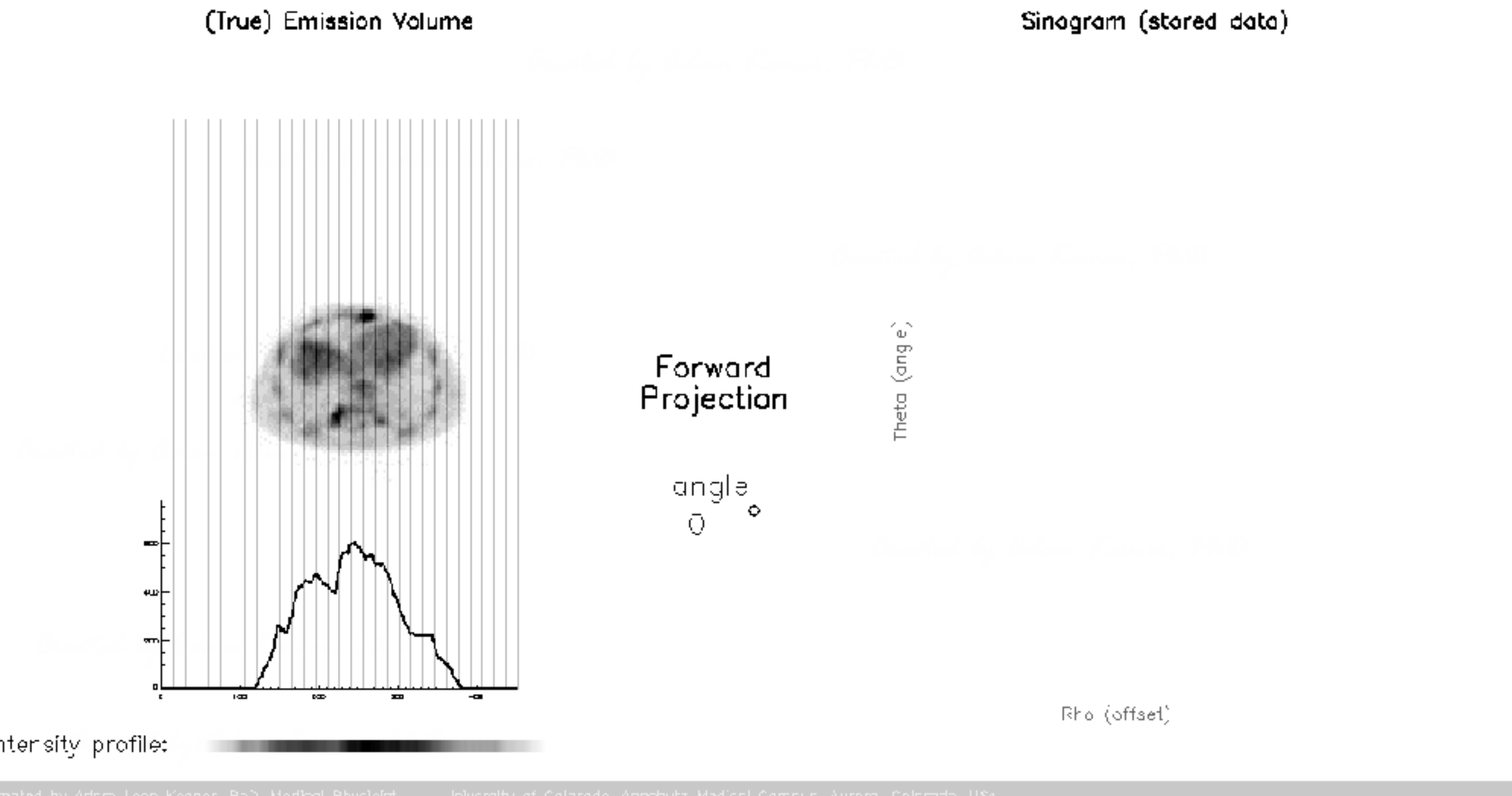

SourceOffsetX
Source
SourceToIsocenterDistance
SourceOffsetY
Re$_y$
Re$_z$
Isocenter
Re$_x$
SourceToDetectorDistance
e$_v$
Det. Origin
e$_u$
DetectorOffsetY
DetectorOffsetX

Ingredients of Reconstruction

- Geometry - description of your source-detector setup
- Projectors - forward e.g. for iterative methods
- Pre-processing - e.g. for scanner properties
- Filtered backprojection - standard reconstruction
- Iterative reconstruction - specialized reconstruction
- Composite Filters - custom combinations

Forward Projection

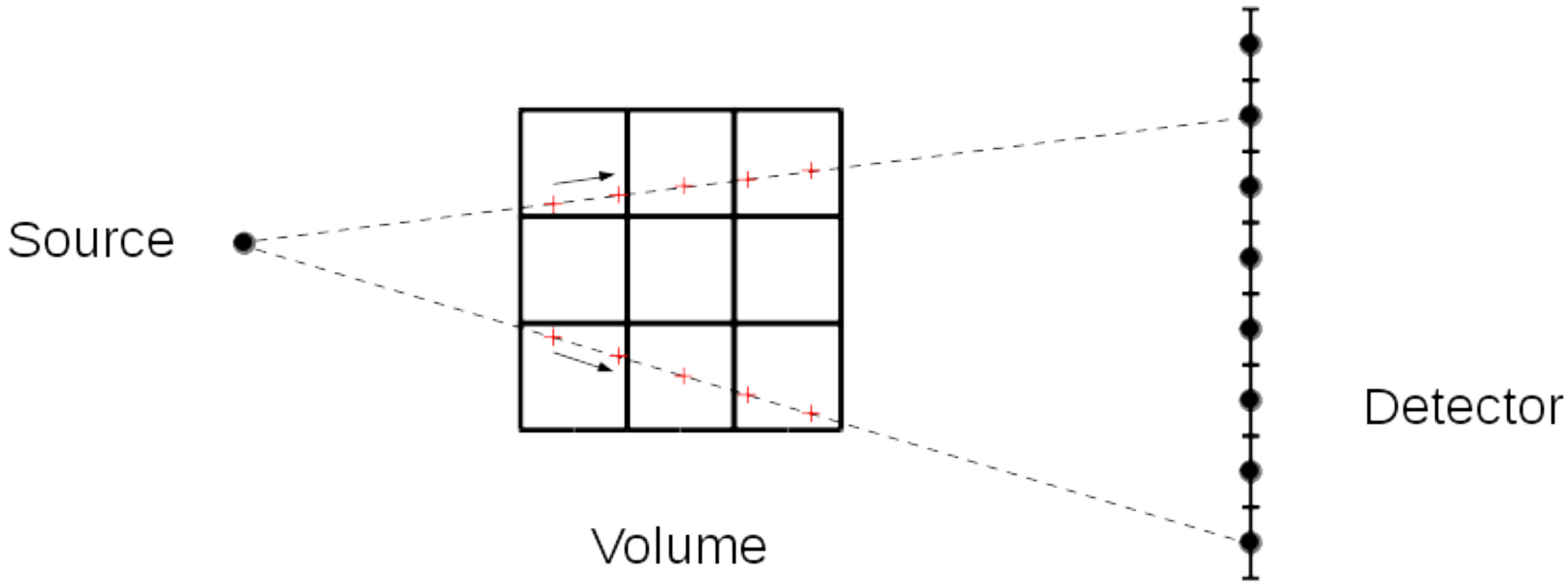
collection of attenuation line integrals



http://www.ucdenver.edu/academics/colleges/medicalschool/departments/Radiology/About%20Us/Faculty/education-research-portal/Pages/filtbackproj.aspx

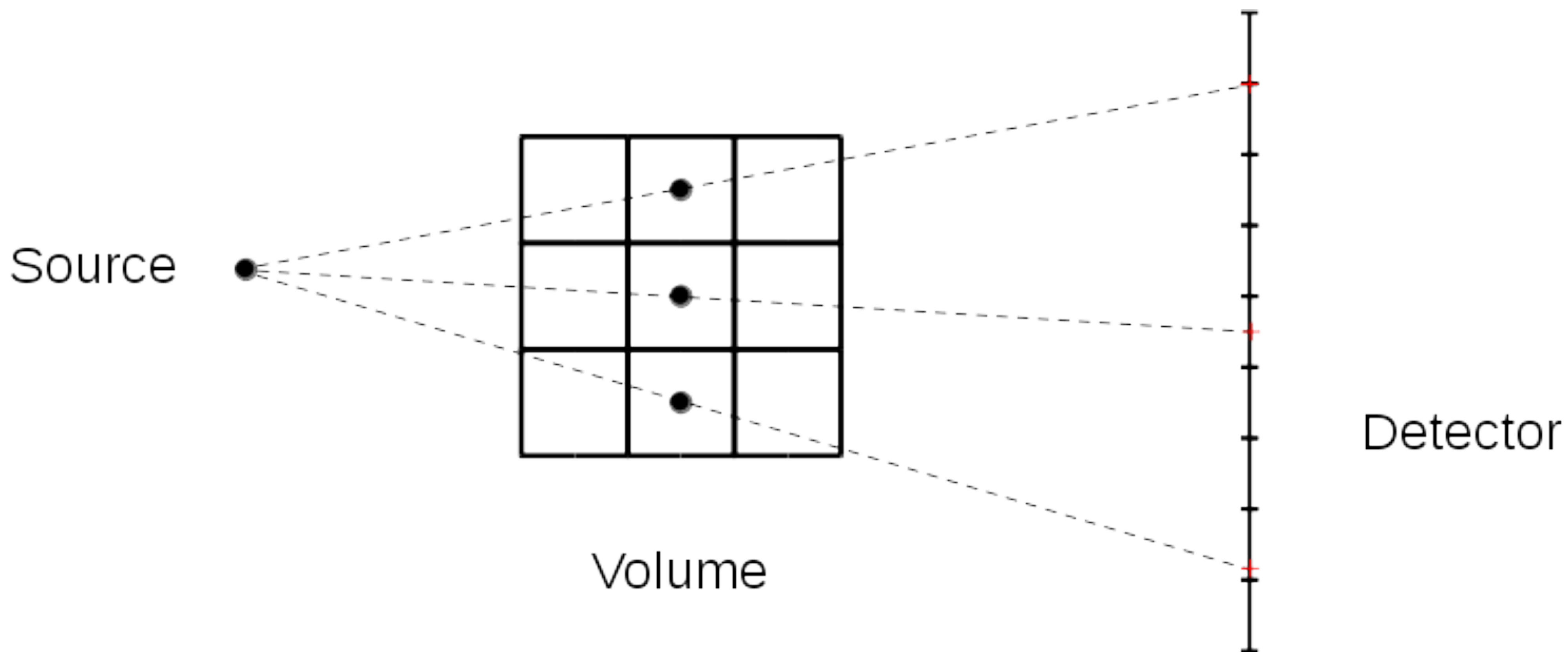
Ray cast forward projector (rtk::CudaForwardProjectionImageFilter)

- Choose a step length
- For each detector pixel of each projection, start from the source, and until you reach the pixel
  - Move one step towards the pixel
  - Interpolate in the volume at the current position
  - Add step $\times$ interpolated value to the pixel



Voxel based back projector (rtk::BackProjectionImageFilter and rtk::CudaBackProjectionImageFilter)

- For each voxel of the volume
  - For each projection
    - Project the center of the voxel onto the detector
    - Interpolate at that position on the detector
    - Add the interpolated value to the voxel



Available in RTK

- rtkforwardprojections
  - –fp Joseph, RayCastInterpolator, CudaRayCast
- rtkbackprojections
  - –bp VoxelBasedBackProjection, Joseph, NormalizedJoseph, CudaRayCast, FDKBackProjection, CudaFDKBackProjection
- Motion-compensated operators
  - CudaWarpForwardProjection
  - CudaWarpBackProjection

Rationale

Pre-processing

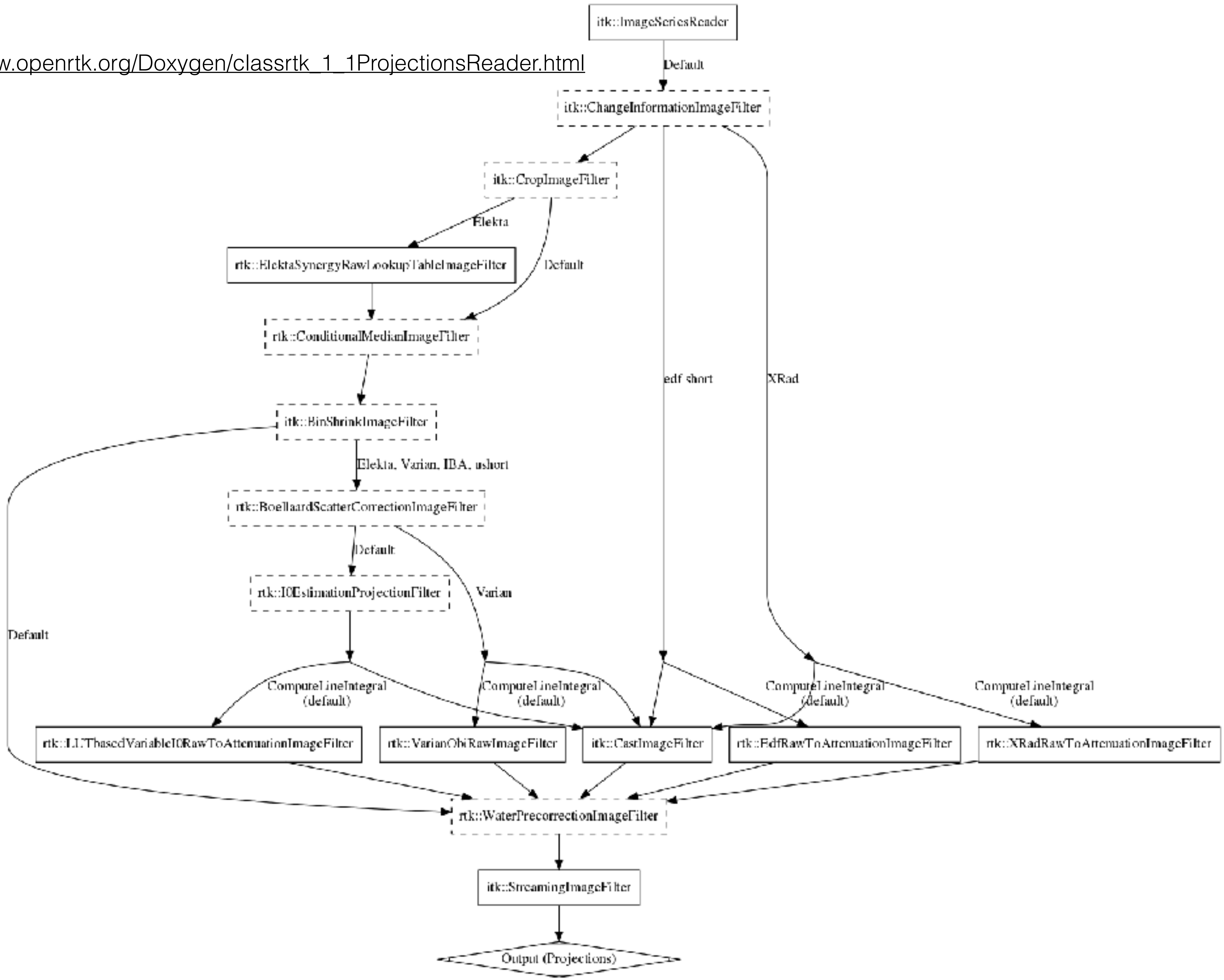
- Real data often need to be pre-processed before they can be used.
- For example, taking the log of the ratio between x-ray projection without object I_0 and x-ray projections with the object I to get line integrals g:

$$g = \ln \frac{I_0}{I}$$

RTK pre-processing

- Gathered in class `rtk::ProjectionsReader` with scanner dependent branches, see online diagram
- All applications using projections have a section `Input projections and their pre-processing` Common options are defined in file `applications/rtkinputprojections_section.ggo`
- `rtkprojections` just read, pre-process and write projections

http://www.openrtk.org/Doxygen/classrtk_1_1ProjectionsReader.html



Available pre-processing

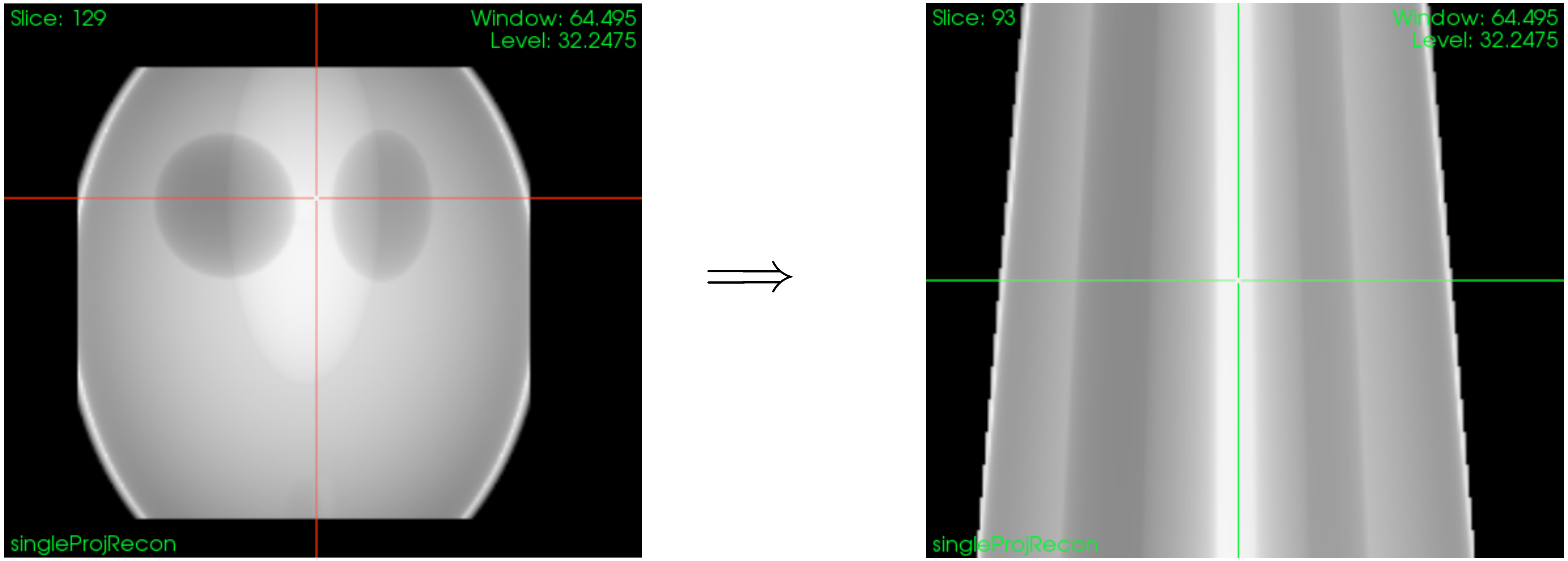
In `rtk::ProjectionsReader`

- Changing meta-information, cropping, binning
- I0 constant value (0 means auto-detection)
- Water pre-correction [Kachelriess et al., 2006]
- Scatter correction [Boellaard et al., 1997]
- And others under developement

Some are not yet in `rtk::ProjectionsReader`, e.g.,

- `rtk::LookupTableImageFilter` for a simple beam-hardening correction (also accessible via the `rtklut` application)
- `rtk::MedianImageFilter` (also accessible via the `rtkmedian` application)

Ideal continuous back projection



[Feldkamp et al., 1984]

- This is not a course on tomography...
- Known as the FDK or Feldkamp algorithm
- Typical algorithm for 2π circular cone-beam CT
- Missing data ([Tuy, 1983]'s condition is not satisfied), approximate algorithm

RTK includes several versions
Also understands <2π scans
overlapping scans
non-centered detector geometries

3D iterative reconstruction methods

- Given measured projections p and the forward projection operator R, find f such that $Rf = p$
- Usually, R cannot be inversed. We seek an approximate solution, as close as possible to $\hat{f} = \arg\min_f \|Rf - p\|_2^2$
- Depending on the method used to minimize the cost function, the resulting algorithms take different names:
  - Gradient descent $\Longrightarrow$ Simultaneous Iterative Reconstruction Technique
  - Kaczmarz method $\Longrightarrow$ Algebraic Reconstruction Technique
  - Block-Kaczmarz method $\Longrightarrow$ (Ordered subsets) Simultaneous Algebraic Reconstruction Technique
  - Conjugate gradient $\Longrightarrow$ . . . Conjugate gradient

This is what we want to use - most likely path algo is iterative

4D iterative reconstruction methods

This is the research topic of the two presenters

- Periodic motion during the acquisition (heart or lungs)
  $\Longrightarrow$ Extract a periodic signal, and its phase
- Each of the N projections has been acquired at a given phase $\phi(n)$
- The cost function reads: $\sum_{n=1}^{N} \|R_n S_{\phi(n)} f - p_n\|_2^2$, where
  - f is the 3D + time sequence of volumes
  - $S_{\phi(n)}$ is a linear interpolator
  - $S_{\phi(n)} f$ is a 3D volume
  - $R_n S_{\phi(n)} f$ is a 2D projections calculated through
  - $R_n S_{\phi(n)} f - p_n$ is the difference with the measured projection

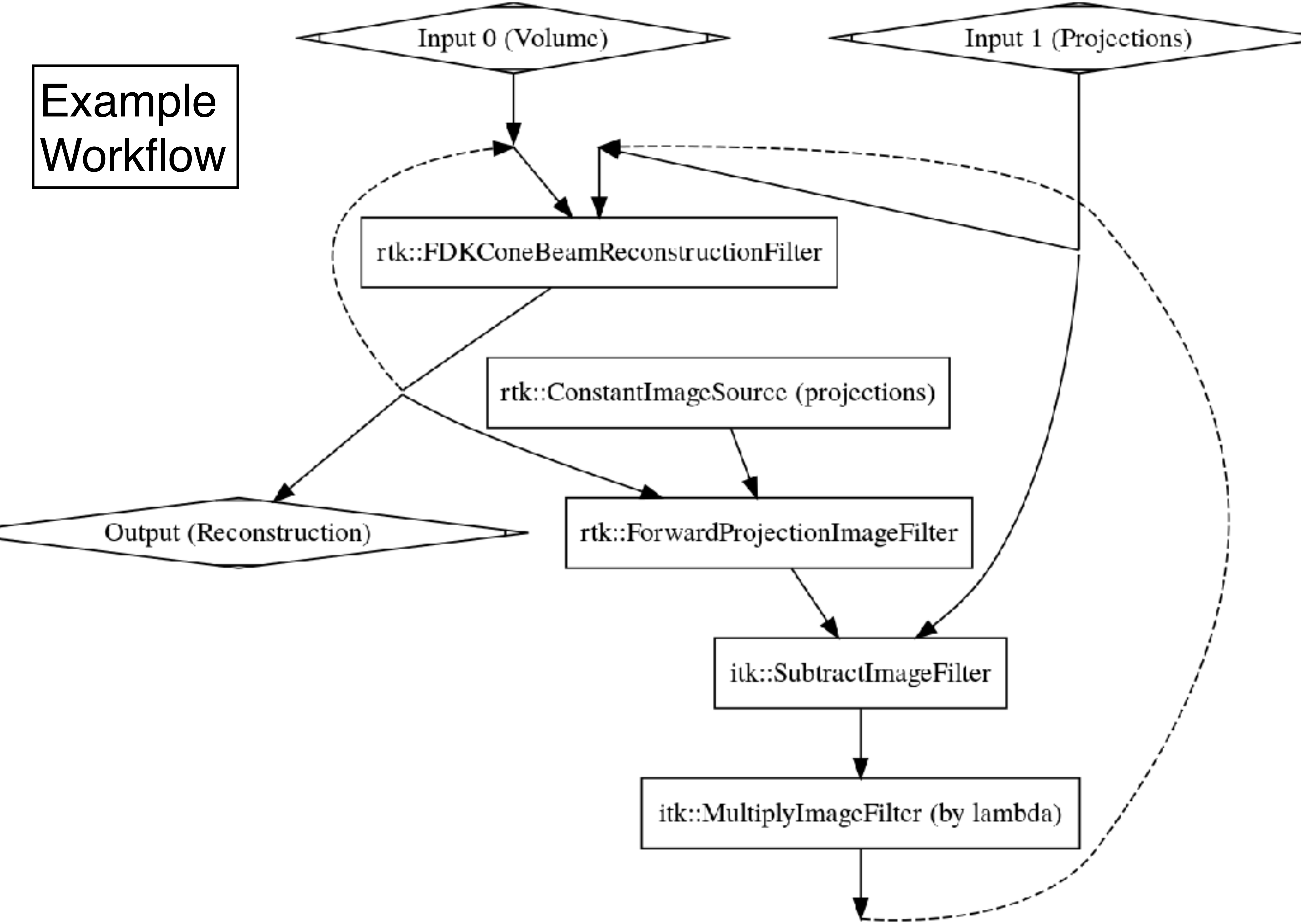
Writing a new composite filter in RTK

- Do the math
- Turn your equations into a pipeline, AND DRAW IT (really, with a pen and a paper). Keep the drawing, you will need it later
- If some of the basic operations you need are not available in RTK or ITK, write a new filter for each
- Now that you have all the pieces, let us start
- Please open
  - code/rtkConjugateGradientConeBeam. . . .h
  - code/rtkConjugateGradientConeBeam. . . .hxx

3D iterative FDK algorithm

- Perform an FDK reconstruction. You get “the current result”
- Forward project the current result. You get “the simulated projections”
- Subtract the simulated projections to the measured projections (the input). You get “difference projections”
- Multiply the difference by λ
- Perform the FDK reconstruction of these difference projections
- Add it to the current result
- Forward project the current result
- ...

Example Workflow



https://cernbox.cern.ch/index.php/s/qBOeD2Tfw3Ev141

Material from the course (~1.4GB)