

ROIs 1

Attending: Chris, Josh, Curtis, Lee, Dscho, Saalfeld, Tobias, Pavel

10:21

- Curtis: imglib
 - embedding, usage
 - interface RegionOfInterface (RealRandomAccessible, RealInterval)
 - Lee: is a point in or out
 - Chris: winding?
 - Curtis: implementation dependent. Cursor determines steps through order. Average all the values, you don't care. For interpolation, you care a lot. You would want RandomAccessible.
 - Josh: (different topic) way to make suggestions to order; "strategies". Curtis: middle ground
 - Chris: less to do about enumerating windings, and more about we understand the constraints.
 - Curtis: ... ArrayImg..., PlanarImg...
- Lee: translation between OME and imglib
 - ... get to that

10:28 - Stephan and Tobias joined

- Curtis:
 - type mechanism
 - ...overflow...
 - Dscho: cut-off
 - Saalfeld: sum and divide
 - Tobias: type converters
- Curtis:
 - problem with two interfaces AND'ed in Javac
 - resolution of generics
 - Dscho: big cast doesn't always work
- Curtis:
 - current interface is NDim and a Mask (BooleanContinuousRegion)
 - RealInterval so must have bounds
 - Saalfeld: extend it to make it more general
 - noninterval
 - Lee: -Inf to Inf?
 - Saalfeld: ROI that includes all uneven integers (checkerboard)
 - That would be a RealRandomAccessible
 - nonboolean
 - probability function
 - Then specialization for NDims or boolean
 - Chris: how can we persist this?
 - Curtis: impl. dependent
 - Represent this in XML?
 - Saalfeld: you'd only ever express the concrete representation
 - Josh: cF. Roger's calipers v. angles (all just 2 lines)
 - Chris: difficult to affect types in the middle of the hierarchy

- Curtis: also persistable in a particular way
- Chris: want something persistable for exchange
- Saalfeld: decide to only store hierarchy leaves
- Josh: turn it around to concrete types with “contexts”
 - System of deprecation and translation
- Lee: ontology and data model. Names are clearly associated
- Dscho: not just Java, have to define persistent data model first
 - that should be hard to change
 - Chris: yeah, half a dozen concrete implementations (CellProfiler, ITK)
- Curtis: concrete number of things that would make it in and there would still be things that you can only do in imglib
- Josh: should still be able to support the storage of extra
- Dscho: hierarchy of interfaces are also generally useful (Interval, Renderable...)
- Chris: or not Renderable by a given implementation. If we don't come up with a standard way of displaying on screen, then some of this doesn't matter (cF. jHotDraw)
 - Dscho: why?
 - Josh: depends on how much is in the storage. Choice of points, how to draw curves. Otherwise we get variance.
 - Chris: should take constraints of display into account for storage.
 - Dscho: separate problems: display and persistence.
 - Curtis: subset of it.
- Saalfeld/Lee: 10:57
 - image data that is chosen/displayed is not the same that came from the scope
 - Curtis: non-invertible coordinates.
 - Saalfeld: attach a transformation used to generate display
 - Curtis: another aspect: unioning regions of interest becomes difficult
 - deform projection to 2D plane and then another one
 - union the two becomes difficult
 - Saalfeld: can only union them in the transform domain (in “ViewSpace”)
 - Lee: didn't like the Oval that I wrote
 - Chris: all transform matrices work on defined plane
 - Saalfeld: don't want to constraint to affine trans. Interesting in neuro-science. 1 mio. fly brains, completely arbitrary
 - Josh: cF. Richard Baldon
 - Chris: even useful to store the transform?
 - Saalfeld: roi -> rotate -> roi -> rotate
 - Each has a separate uniform. Want to store each of them separately, even if a single union
 - TrackEM has a very ...
 - name/type/data string; contract is outside of the XML
 - Chris: have to focus on the things that we can exchange
 - Saalfeld: can have an “Unknown” transformation
 - Lee: useful to have model for affine, etc.

11:09 Pavel joined

- con't
 - Dscho: do you have to store the transformation with the ROI?
 - Lee (via Curtis): 2 different things we're trying to model
 - what are the spaces and what are the objects in the space?
 - ROI model covers what are the objects but also need what are the spaces/transformations

- Lee / What we want to store in OME
 - Generating Labelling
 - Scale is about 100K images
 - ~1000 cells/cellular components per image
 - up to 1000 measurements per component
 - ~1 Billion measurements
 - cram that into database to learn phenotypes of cells
 - combination of features
 - ImgLib2 implementation is [net.imglib2.labeling](#) package
 - store segmentation and measurements
 - Chris: valid to say that all seg. are in 2D?
 - Lee: at the moment. project coming up with 3D segmentation. Can see generalizing it to NDim.
 - Chris: output is a mask? i.e. not translating to a vector?
 - Lee: one integer mask (conceptually a series of binary masks)
 - Chris: compared strategy to using masks and using run-length encoding
 - Lee: worried about the worst case storage
 - Lee: storing measurements
 - Chris: More concerned with decomposing the regions for attaching the measruements
 - Chris: not storing masks as above
 - 10 million of 500x500
 - Curtis: relationship between labelling and roi packages?
 - Lee: you can iterate over labeling to get rois
 - Curtis: labeling is a bigger than than rois
 - Curtis: if you could take ROIs and render them, it'd be solved.

- Curtis: common rendering 11:25
 - Some cases where we should do it; and some where we won't
 - also: e.g. Apache Pivot. Can't render outside of jhotdraw. Would have to implement mechanism for rectangles, etc. or Android.
 - Dscho: RnR
 - Saalfeld: in imglib is only grabbing pixels and transform. Rest is imglib independent.
 - Curtis: shouldn't target a drawing lib on top of imglib that's not jhotdraw.
 - Chris: Tried to interact with developers?
 - Curtis: one guy, fork. Kind of prickly. Wanted poms -- "I don't see the point" Everything is in netbeans.
 - Chris: bad experience with insight

- Chris: geometry?
 - Curtis: drawing is completely in Swing UI. Lee added sezpoz for jhotdraw adapters for ROIs. Looks for adapter for ROI implementation with jhotdraw shape.
 - Chris: decoupling between presentation model and persistence leads to duplicating everything
 - Curtis: can get around some of that at the imglib level...
 - Chris: make sense to porting AWT to use imglib primitives. If sticking point is AWT mess, then go in and hack it.
 - Dscho: would be nice
 - Curtis: if we have a body that we can't put a body on it. imglib2.rnr
 - Dscho: but it's rewriting jhotdraw
 - Josh: fork-the-fork. LGPL or CC (compatible with BSD)
 - Dscho: not big on the pattern style
 - Chris: will have the problem in all of our tools. Anything persisted from ImgLib will need to be shown on screen.
 - Dscho: and you want them to be manipulatable
 - Chris: write, read in, measure is the holy grail. Why we're discussing
 - Lee: worst case is points in the plane
 - ... some lost here
 - Curtis: imagej wraps ROIs with Overlays. OME used to have the distinction..
 - Chris: if want to take this on, that's fine.
 - Need a person sitting on it. ROI person is also RnR person.
 - Saalfeld: view must be outside
 - Dscho: call it a probability map, that's what it is.
 - Curtis: RnR will also deal with other things. Rois may be most complicated.
 - Chris: if we control the type hierarchy (jhotdraw) then we can do what we need.
 - Lee: might be nice to have (list of) control points in the overlays
 - Dscho: that package would have to provide that.
 - Curtis: parts of imagej2 will have to change or migrate into the RnR package
 - Chris: how much Java is CellProfiler using?
 - Lee: have translation layer for images and masks that we pipe into java and use imglib
 - Chris: just an extension point or core?
 - Lee: just one module out of many.
 - Chris: anything core that's in Java?
 - Lee: just bioformats
 - JNI...JNA
- Way forward (11:45)

- Josh: Need to back up
- Chris: need a straw man
- Dscho: can we do that now?
- Tobias: split into several packages
 - basic (fundamental) subsets of points, binary case etc.
 - in which space do they live, etc. transformations. separate from images.
 - display, manipulation (above and beyond the simple are pixels in)
 - ...invertible transforms, unions
 - persistence...
- Tobias:
 - one task is we need editor for polygons, shapes
 - what comes out are fundamental rois
 - then apply transforms
 - that can be fit into any representation
 - Dscho:
 - Curtis: no one has problem with imglib2 proposal, so let's do it today?
 - Tobias: interfaces are relatively simple. Everything is there. Doesn't mean, though, that we can do something useful with it. For us, ROI is just RandomAccessible.
 - Dscho: but it's also so much more. Also, Manipulatable, Renderable.
 - Josh: want to understand "Persistence space" and the possibly transform (at least one interface per class of transforms)
 - Curtis: Lee's package has concrete implementations already as straw man and there are limitations
 - Lee: for example adding a single bit to 2D polygon
 - Saalfeld: leave it out
 - Josh: don't want to have an explosion of concrete types
 - Lee: to get from space A to space B you have to have a transform
 - ...
 - Saalfeld: thickness in OME? No.
 - Lee: want 2 micron thick, etc. Having that in the transformation, you can control that arbitrarily.
 - Saalfeld: projecting or limiting transformations
 - Chris: think it's bad to be projecting into "Real" space by microscope frame reference
 - Dscho: we want to compare things...
 - Saalfeld: transformation included? Chris: only 2D affine.
 - Saalfeld: ITK has more. ND affine.
 - ImgLib has a general one with matrix
 - Curtis: java3d takes a 3x3 or 4x4
 - Chris: jogl? Presumably also (OpenGL projection matrix)
 - Tobias: one more thing to think about
 - with the above, you can define 2D roi and say "2 slices thick"
 - another concrete type with a transform

- same thing could be achieved by 1 2D roi, transform to make it infinite, then another roi which bounds in 3D, then take union
- would be nice if you could store hierarchy of rois
- Saalfeld: nested list of transformations. Tobias: “operations”
- Compound list of ROI.
- Nice to store all of that into XML, but would be good to simplify.
 - some things can’t simplify
- Chris: as implementor, don’t want to deal with that
 - Saalfeld: just deal with it literally
- Curtis: just how views work. Tobias has logic to simplify
 - Saalfeld: TrackEM have that. in Data string, “transformation list”
- Tobias: 2 things. Transformations and operations, and you have a tree of them
 - For looking up a pixel, it’s fine
 - For manipulating it, it’s another story
 - Chris: or drawing on screen
 - Tobias: always manipulating on leaves
 - Josh: basically a compiler
 - Lee: compiler or interpreter
 - Tobias: simplifications and reductions
 - To do anything you may have to simplify
 - Chris: turn it into a “result”
 - Dscho: this is why I say we need interfaces
 - ND hypersphere
 - can keep radius constant and just move center
 - keep center constant and change the radius
 - Chris: more complex the translation hierarchy, the more complicated it becomes to manipulate it
 - Saalfeld: first step is to not allow it. only manipulations on the leaves.
 - 2D polygon to 2D polygons, that’s fine.
 - Pavel: can you split ROIs?
 - Josh: yes. union of hyper-volumes
 - Curtis: definitely expressible
 - Saalfeld:
 - Many ROIs with Transforms
 - Dscho: implicit surfaces
 - Saalfeld: in TrackEM “Ontology term”, a “concept”, “Thing”, “Term”
 - Curtis: only in target space, not in source space
 - Josh: cF. Roger’s control points example
 - Chris: you can never invert the set.
 - OME: ROIs is a union of shapes

- ImgLib: Group is a union of ROIs
 - Saalfeld: Bridge into semantic space
 - Saalfeld: Entry point for CellProfiler to reason
 - Pavel: mixing two domains
 - one is that what user defines and something abstract (application domain)
 - Josh: still thinks that it's just a ROI
 - Chris: we get to this every time
 - 1) ROI is a set of shapes over time
 - 2) or regions in same time that make up a volume
 - Pavel: definition of group should be up to the user
 - Saalfeld: my problem with the group
 - Josh: concrete types could handle the representation; just a "Groupable" interface
 - Tobias: have to be careful that leaf of transform doesn't explode storage
 - Saalfeld: interesting point - iterate all points
 - Josh: also boundary
 - Saalfeld: doesn't exist in real space, perhaps boundary region
- Lee leaving soon (12:38)
 - Curtis: covered what you wanted?
 - Lee: think I did a good job
 - Curtis: can fill you in later.
 - Lee: may have suggestions
 - Curtis: most important thing is to get someone (ROI and RnR)
 - Plan for the next year
 - Saalfeld: graduate next month (maybe later)
 - stay for a while in Berlin, 1 year
 - Curtis: no specific plan of work?
 - Saalfeld: some science projects
 - Curtis: Catmaid branch, etc. Secondary. Nothing in core library that needs to be done other than ROIs?
 - Saalfeld: no, quite happy. More transformations.
 - Lee
 - Chris: plate full?
 - Lee: could propose initial XML model. That's within bounds.
 - Chris: but we agree that it's a concerted effort
 - Saalfeld: first draft comes out quickly
 - Pavel
 - User? (cF. bioformats)
 - Level of competence needed?
 - Does every IJ2 developer need to know about it.

- Chris: we will need to have good entry points
- Curtis: all that stuff is doing more and better math. But things will be similar to now.
- Pavel: tools for you to manage that?
- Chris: yes, and things can be saved in OMERO, re-shared, etc.
- Pavel: new GUI for grouping ROIs
- Saalfeld: IJ2 support would turn better into TrackEM implementation and XML is storable in OME.
- Pavel: most IJ2 bio-hackers won't be able to use it
- Dscho: they should be able to if we do our job right
- Pavel: if there are GUI elements, then it should be ok.
- Saalfeld: there'll also be a nice programming interface
- Chris: also just the 2D case, making it easy, will make it usable
- Dscho: "here's an image, a roi, and an algorithm, do it"
- Pavel: if that's a core feature, it would be so much better
- Saalfeld: in TrackEM it takes a week because the ROIs are stored in world-space,...
- Pavel: biologists would like this because they do work, but then someone says your data is ugly and needs to be mapped.
- Saalfeld: OME-XML is 5D, any plan to fix it? Definitely.
- Curtis:
 - guessing we could put some funds towards it
 - perhaps best to have someone paid from multiple sources
 - question of where they should be
 - Chris: moving toward people championing things
 - Curtis: OK for Chris & Josh to be on top of everything. But then need more people in the trenches.
- Tabled for Kevin
 - Chris: we have positions open
 - another Bio-Formats person?
 - perhaps a separate RnR person
 - LOCI wants a BF position
 - Mark has joined as staff through at least December (enjoyed the proj.)

12:54 Lee left.

Lunch

High-level list

1. Contexts/Interfaces

- a. **Display, ...**
 - 2. **Concrete implementations (non-breaking. “Leaves”)**
 - 3. **Storing spaces**
 - 4. **Other**
 - a. **Layers**
 - b. **Hierarchical**
 - c. **Statistics / Measurements**
 - 5. **Action items**
-

Action Items

1. Continue fleshing out and refining ImgLib2 RegionOfInterest interfaces and implementations.
 - a. RegionOfInterest not necessarily an Interval
 - b. RegionOfInterest can map to [0, 1] rather than boolean (i.e., BitType)
2. Develop XML model for spatial transformations?
 - a. Can express ROIs as objects in a particular space and/or with a particular transformation
3. Consider creation of a rendering framework (imglib2-rnr?) that is AWT-independent
 - a. Could be a fork of JHotDraw7 (which is dual LGPL or CC BY 1.5 licensed)
 - b. Would provide a library to paint—and maybe even manipulate—overlays/ROIs
 - c. Makes sense as a layer on top of ImgLib2
 - d. Would need to include the concept of “overlays” (i.e., rendering settings for ROIs)
 - e. Could be used by ImageJ2, OMERO.insight, and other Java software
 - f. Could even provide ROI visualization on Android ImageJ2 UI, etc.
 - g. Would unify the dichotomy between ImgLib2-generated image renderings, and JHotDraw-generated ROI renderings and ImageFigure scaling & translation

ROIs 2

Attending: Saalfeld, Tobias, Chris, Curtis, Josh, Dscho

Goal: Define example exchange format with interface examples

[Sketching via examples](#)

[Current OME Example:](#)

[Group Example:](#)

[Transformations](#)

[Notes](#)

[Complex example with hierarchy](#)

[Concrete types](#)

Sketching via examples**Current OME Example:**

```
<Roi>
  <ImageRef ID="1"/>
  <Union <!-- no intersection yet -->
    <Polyline z="1" t="6" z="5"/>
    <Polygon/> <!-- missing elements means entire extent -->
  </Union>
</Roi>
```

Group Example:

```
<GroupRoi>
  <Metadata>

  </Metadata>
  <Rois>
    <Polygon2D/>
    <Polygon2D/>
    <GroupRoi>
      <Rois>...</Rois>
    </GroupRoi>
  </Rois>
</GroupRoi>
```

Transformations

```

<Mapping ID="6"
  source="Image:6"
  target="Roi:6"
  TransformID="7"/>

<Mapping ID="7"
  source="Image:7"
  target="Roi:6"/>
  <!-- no transform == identity -->

<!-- is a source-space itself -->
<Rectangle2D X="5" Y="5" H="5" W="5"/>

<AffineTransform ID="6">
  <data>[0 1 0 0 1 0]</data>
  <!-- default is identity →
  <!-- other options: Projection, Unknown -->
</AffineTransform>

```

Notes

- For the moment only invertible transforms, or at least pseudo-invertible
- Josh: Possibly adding other metadata?
- Dscho: Image-Image example is SPIM with different angles
- Curtis: Better way than mis-using SPW for lots of images
- Chris: deprecate Project and Dataset
- IntervalRoi ("RectangleND") is the same as an interval target
- Dscho: model spaces even "RealSpace" with units (jscience)
 - Saalfeld: unit of each axis (all optional)
 - Dscho: also "lifetime", etc.

Complex example with hierarchy

```

<HierarchicalRoi>
  <Operation/>
  <Transformation/>
  <Roi/>
</HierarchicalRoi>

```

Coffee. 16:15-16:30

Things to do: examples, concrete types, interfaces, hierarchy, measurements, action items

Ordering:

1. concrete

Concrete types / examples

- Tobias: transform goes from coords to coords

- 2D circle to cylinder is not a transform
- transforms are always invertible
- Dscho: not injective
- Therefore this is an “Operation” since not an “Invertible coordinate transformation”
- Dscho: opposite of a projection (“Embedding”, “Extension”)
- For legacy of ij1, since all are 2D with infinite extent

<!-- Always appends to the end -->

```
<ExtendDimension ID= "Roi:6" roiID="Roi:5" min="1.0" max="2.0" />
```

```
<ExtendDimension ID= "Roi:7" roiID="Roi:6" min="5.0" max="4.0" />
```

```
<RemoveDimension ID= "Roi:6" roiID="Roi:5" d="3" position="2.0" />
```

```
<RemoveDimension ID= "Roi:7" roiID="Roi:6" d="2" position="1.0" />
```

- Primitives
 - OME
 - NBox: point, line, rectangle (incl. 3-point def. 1 is rotation)
 - polyline
 - ellipse (incl. 3-point definition along with centers)
 - Other
 - Bezier curves (but what’s inside)
 - Masks (images that are RL-encoded)
 - Voronoi graph (set of coordinates with a sample value [0,1] each)
 - hyper-spheres, lines, planes, ovals, ...
 - chain-codes (same as RL-encoded)
 - Kevin E.: get Kevin McCormick/ITK’s medical ROIs
 - Point-Sets: extents/intervals and logical conditions
 - “ $x^2+y^2<5$ ” i.e. “FormulaRoi” (maybe “ImplicitRoi”?)
 - Or “RelationRoi”?)
- Non-primitives
 - compound ROIs