# ATHEY LAB

# Java Curation Modules for LONI Integration

# Contact: David Dilworth (dilworth@umich.edu)

**Introduction**

Runtime Java modules for integration with LONI.

# Compiling and Running

## Compiling the Source folders from command line

### Folders and files

* CurateCn, see section 2.2.1
* FilterC2Lst, see section 2.2.2
* MergeCn, see section 2.2.3
* ij.jar, ImageJ runtime file is run headless from some curation modules

### Compiling from command line

* “cd” into each folder
* Delete any existing .class files
* Run the “mk” script file

## Preparing the Runtime folder

* Delete all .class files from the Runtime folder
* Copy all .class files from three folders in Source to the root of Runtime
* Delete all files from the Results folder

## Executing in the Runtime folder from the command line

### Preparing the 0000169 folder

* Retain all files .tif and .txt files
* Delete all other files

### Execution

* “cd” into the Runtime folder
* Select Batch169 or Pipeline169 to process images in the 0000169 folder
  + Batch169 is the simplest and runs as single thread
  + Pipeline169 emulates pipeline and process images in parallel

Memory on current linux box limited processing to pairs

* Run Batch169 or Pipeline169
  + Images are read from the 0000169 folder
  + Curation setup is read from the Filters folder with settings dated 20151028
  + Results are written to the Images folder

### Analyzing results after Batch169 or Pipeline169 have completed

### Delete all files in the Results folder

### Run “MoveResults” (root of folder systems) from the command line Copies results from 0000169 to Results

### Results and Results\_Filter\_20151028, etc can be compared The log files have date and time and will be different

### The order of processed may be operating system dependent

### The OriginalResultsNAS folder has NAS results

# Data flow and processing

## Block Diagrams

### Square orange boxes are files

* Solid box means the file is read or written
* Dotted box means only the filename is accessed
* Incoming solid arrow means the filename is given on the command line
* Incoming dotted arrow means the filename is inferred

### Rounded green boxes are processing modules

### Filenames

* Filenames are the full path with appropriate linux (/) or windows (\) slash
* 2nd parameter in MergeCn is a part of a filename and doesn’t contain path

## Modules

### CurateCn

Inputs a segmented image file (1st parameter) and computes these voxel parameters for each segment ID and reported in the corresponding .log and .csv files.

* Centroid
* Voxel count
* Surface voxel count
* Spherical compactness
* Void count
* Edge count
* Bound true/false
* Adjacency matrix (only reported on .csv file)

Inputs the filter file (2nd parameter) and applies it to the computed voxel parameters. The filter file contains inclusive min/max values for these voxel parameters

* Voxel count
* Void count
* Edge count
* Bound true/false
* Spherical compactness

Outputs:

* name\_c0\_gGGG\_mask.log – log file
* name\_c0\_gGGG\_mask.lst – contains the filenames of image files that passed the filter test
* name\_c0\_gGGG\_mask.csv – spreadsheet readable file with voxel parameters

Notes:

* Processes a single .TIF file and can be executed in parallel with other .TIF images.
* For C2 images, the corresponding “name\_c0\_gGGG\_mask.lst” file must have been processed

### FilterC2Lst

Inputs these files given at parameters:

* name\_c2\_gGGG\_statistics.txt – statistics file created by an earlier process
* name\_c2\_gGGG\_connected\_prelim.lst file - created after CurateCn processed at C2 image.

Evaluates each nucleoli in the input .LST to see if it contains a non-zero amount of Fibrillarin (given in the .TXT tile)

Outputs:

* name\_c2\_gGGG\_connected.lst – contains filenames that passed the Fibrillarin test

Notes:

* Processes a single .LST file and can be executed in parallel with other images.
* The corresponding “name\_c2\_gGGG\_connected\_prelim.lst” file must have been processed

### MergeCn

Inputs all .LST files with names that match the 2nd command line parameter. The contents of all input .LST files is merged into a single file (name is the 1st parameter). Note the 1st parameter is traditionally the name of the working folder.

* **Example to merge files in the run 0169 data set:**
  + **java -cp . MergeCn ../0000169/0000169\_c0.lst mask.lst**
  + **java -cp . MergeCn ../0000169/0000169\_c2\_pre.lst prelim.lst**
  + **java -cp . MergeCn ../0000169/0000169\_c2.lst connected.lst**

name\_c0\_gGGG\_mask.tif

filter\_c0\_date.flt

CurateCn name\_c0\_gGGG\_mask.tif filter\_c0\_date.flt

name\_c0\_gGGG\_mask.log

name\_c0\_gGGG\_mask.lst

name\_c2\_gGGG\_connected.tif

filter\_c2\_date.flt

CurateCn name\_c2\_gGGG\_connected.tif filter\_c2\_date.flt

name\_c2\_gGGG\_connected.log

name\_c2\_gGGG\_connected\_prelim.lst

name\_c0\_gGGG\_mask-MMM-nucleoli-NNN.tif

name\_c2\_gGGG\_statistics.txt

FilterC2Lst name\_c2\_gGGG\_connected\_prelim.lst

name\_c2\_gGGG\_connected.lst

**Process C0 Imagery**

**Process C2 Imagery**

**Apply Fibrillarin Test to C2 Results**

name\_c0\_gGGG\_mask.csv

name\_c2\_gGGG\_connected.csv

MergeCn path/folder\_c0.lst mask.lst

name\_c0\_gGGG\_mask.lst

name\_c2\_gGGG\_connected\_prelim.lst

name\_c2\_gGGG\_connected.lst

folder\_c0.lst

MergeCn path/folder\_c2\_pre.lst prelim.lst

folder\_c2\_pre.lst

MergeCn path/folder\_c2.lst connected.lst

folder\_c2.lst

**Merge all “name\_c0\_gGGG\_mask.lst” files**

**files**

**Merge all “name\_c2\_gGGG\_connected\_prelim.lst” files**

**files**

**Merge all “name\_c2\_gGGG\_connected.lst” files**

**” files**

**files**