Wichita State University

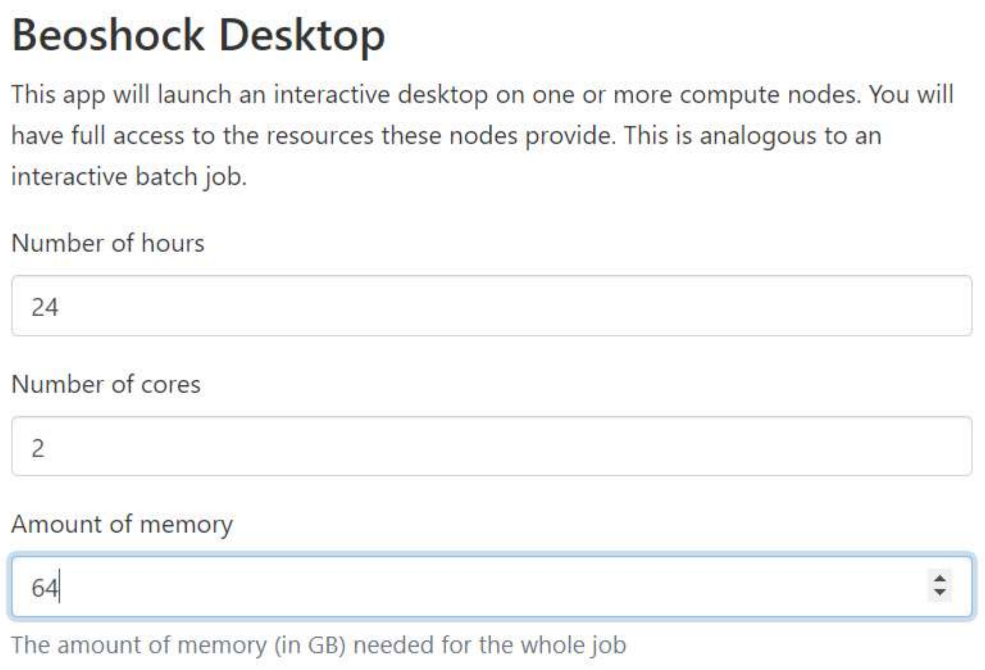
STL generation using ImageJ & ParaView

Marcus Ang
Department of Mechanical Engineering
Wichita State University

1/30/2023

1. Setting Up the Workspace

- In order to access the software with a GUI, graphic properties must be enabled (on Windows, X server is required e.g. MobaXterm), however the most convenient method is by setting up a Beoshock Desktop session.
- Amount of memory needs to be at least 8gb. Number of cores and amount of memory can help with processing time, however, it is dependent on the process in ImageJ as some are designed to utilize multiple cores but others are not



Beoshock Desktop

This app will launch an interactive desktop on one or more compute nodes. You will have full access to the resources these nodes provide. This is analogous to an interactive batch job.

Number of hours: 24

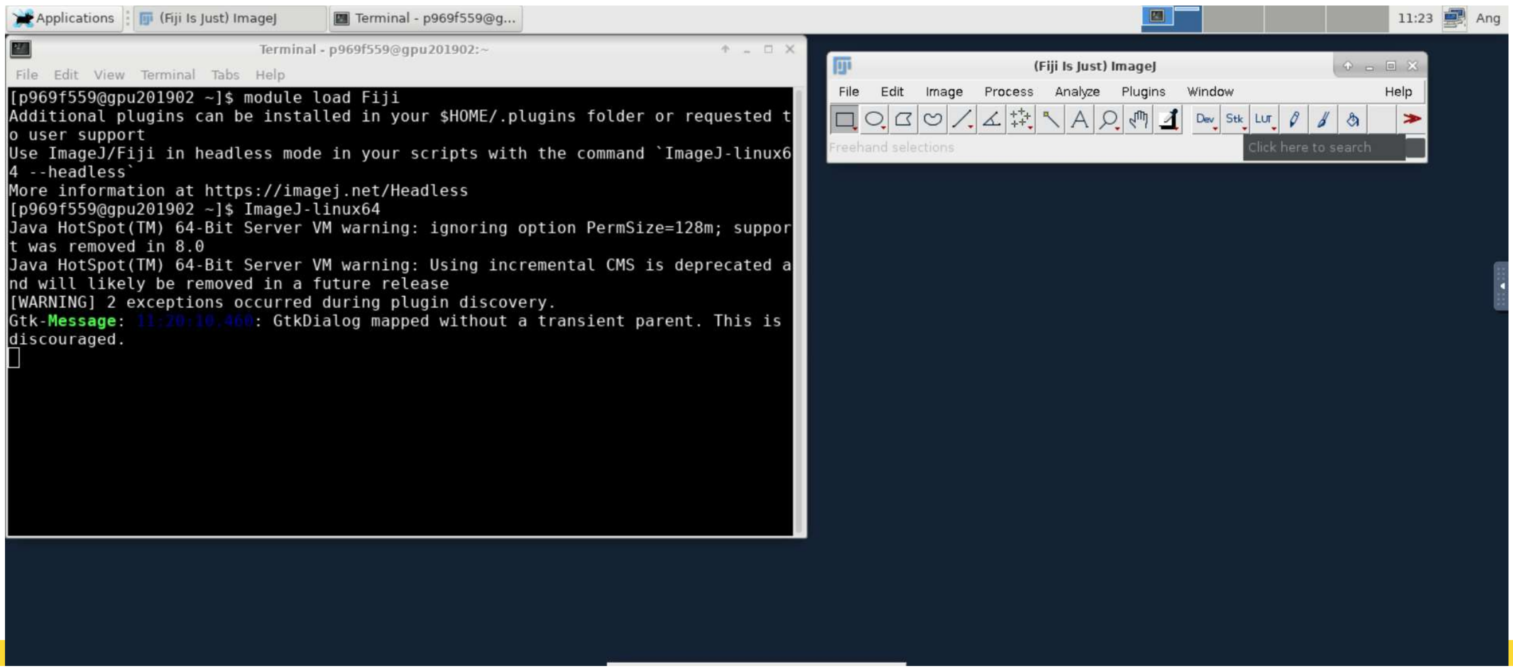
Number of cores: 2

Amount of memory: 64

The amount of memory (in GB) needed for the whole job

1. Setting Up the Workspace

1. Open Beoshock Desktop session and open terminal
2. Load the module: module load Fiji
3. Open the software. ImageJ-linux64
   Note that it is normal to see Java server error messages in terminal and auto-updater error.
4. On the top right of the Desktop Session, you will notice there are 4 windows. These windows are workspaces that can be utilized to run multiple macros on different ImageJ



```
[p969f559@gpu201902 ~]$ module load Fiji
Additional plugins can be installed in your $HOME/.plugins folder or requested t
o user support
Use ImageJ/Fiji in headless mode in your scripts with the command `ImageJ-linux6
4 --headless`
More information at https://imagej.net/Headless
[p969f559@gpu201902 ~]$ ImageJ-linux64
Java HotSpot(TM) 64-Bit Server VM warning: ignoring option PermSize=128m; suppor
t was removed in 8.0
Java HotSpot(TM) 64-Bit Server VM warning: Using incremental CMS is deprecated a
nd will likely be removed in a future release
[WARNING] 2 exceptions occurred during plugin discovery.
Gtk-Message: 11:20:10.460: GtkDialog mapped without a transient parent. This is
discouraged.
```

2. Filtering the Images

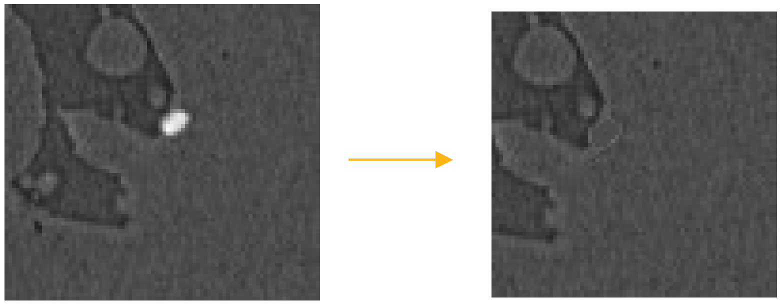
Note: Based on my experience, processes seem to run faster when images are processed one at a time instead of in an image sequence. The rest of the instructions are based on one single images. These instructions can easily be coded as a Macro for automatic image segmentation.

1. If the images are not in numerical order, an easy way to rename them is using the following command in terminal (in the directory with images)
   ```
   counter=0; for file in *; do [[ -f $file ]] && mv -i "$file" $((counter+1)).tiff &&
   ((counter++)); done
   ```

2. File>Open> “image”

3. If there are any defects, we want to remove them first before converting the image to 8-bit. This is necessary before applying filters and thresholds as the defect will add or subtract nonsensical values to the algorithm. The following code replaces any values more than 25,000 (bright white) and replaces them with 20,000 (average value)
   Process>Math>Macro
   ```
   if (v>25000) v=20000
   ```

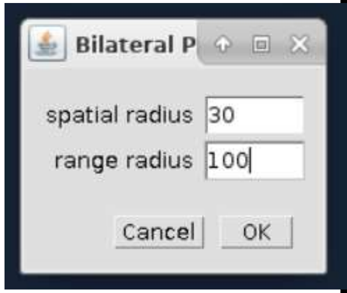
4. Convert the image to 8-bit
   Image>Type>8-bit



2. Filtering the Images

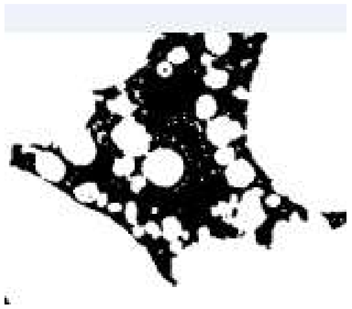
5. Crop the image so only the region of interest is filtered. This is necessary as black blank space (values=0) will affect the filtering and threshold algorithm. If the test specimen is rectangular, we can take advantage of the geometry and apply Auto Crop filter that will guess the background (black) and crop the image to a rectangle.
Image>Adjust>Auto Crop (Guess the background)

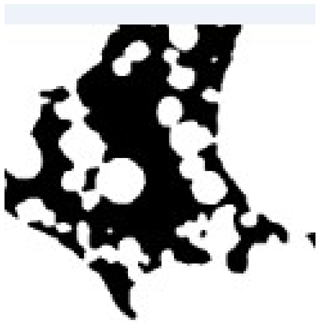
6. Now we have images that are suitable to apply filters. Generally, we want to have filters that blurs (reduce noise) but edge-preserving (preserves geometry). A good option is Bilateral Filter. Suggested parameters are provided.
Plugins>Process>Bilateral Filter



spatial radius 30
range radius 100

7. To improve image segmentation, a Gaussian Blur is suggested to further reduce background noise and small holes in pores. Suggested sigma value: 2 or 3.
Process>Filter>Gaussian Blur


Sigma 2


Sigma 3

3. Image Segmentation

1. Thresholding can be done manually, however, we can also do it automatically if the previous pre-processing steps were done. Ostu methods seems to work the best however percentile and RenyiEntropy works good too.
   Image>Adjust>Threshold
2. Click on auto and enable dark background. Apply the filter and the image should be segmented. If the image is not segmented well, the levels can be manually adjusted however, for automation, it is better to go back to Stage 2 and adjust some parameter.
3. Since the image was cropped, we need to modify the image such that the size of the segmented image matches the original image
   Image>Adjust>Canvas Size
4. Enter the width/height of the original image and ensure position is center
5. Now the image can be saved
6. Note that these are saved individually, we can reopen these later and save it as an image sequence or visualize it.

Sample Macro

```
for  (i = 1; i <2034; i+=1){
open("/home/p969f559/foamRun/porousMedia/images/images/HRES/"+i+".tiff");
run("Macro...", "code=[if (v>25000) v=20000]");
setOption("ScaleConversions", true);
run("8-bit");
run("Auto Crop (guess background color)");
run("Bilateral Filter", "spatial=20 range=80");
run("Gaussian Blur...", "sigma=2");
setAutoThreshold("Default dark");
setAutoThreshold("Otsu dark");
setOption("BlackBackground", true);
run("Convert to Mask");
run("Canvas Size...", "width=1988 height=1994 position=Center");
saveAs("Tiff", "/home/p969f559/foamRun/porousMedia/images/images/filterTest/cropTest/"+i+".tiff");
close();
}
```

4. Creating the STL

- Depending on the operation, ParaView may require a large number of resources. However, note that increasing cores and memory does not guarantee improved performance.
- All post-processing of the STL may be done in a separate software, however for simplicity and file management, I suggest perfoming all operation using ParaView

1. Open Image Sequence in ParaView
2. Apply threshold filter, minimum and maximum set to 255
3. Apply extract surface filter
4. Save surface to STL
5. Reset session and import STL to ParaView
6. Decimate up to 90% and select preserve topology. Save as new STL

Notes of Memory and Core requirement

- Image segmentation 1 Core 32 GB (Utilized 6.11gb)
- Stack construction 1 Core 32 GB (Utilized 12.5gb)