

IMAGE PROCESSING WORKSHOP

Companion eBook

PROCESSING IMAGES USING IMAGEJ

Presenter: Aya Takase

You'll learn:

You have probably applied some form of image processing to your CT data, such as denoising or smoothing, before. Have you ever wondered how it works or what is different between different denoising techniques?

In this workshop, you will learn how these image processing techniques work and how to choose the right one.

You can also download the open-source image processing program [ImageJ](#) and try different image processing on sample data to have hands-on experience and deepen your understanding of image processing.

Here is the [recording of the workshop](#).

CONTENTS

04 [1. Why do we process images?](#)

06 [2. How does denoising work?](#)

11 [3. How does edge detection work?](#)

14 [4. How does sharpening work?](#)

17 [5. ImageJ hands-on exercises](#)

24 [About the tool](#)

26 [Take-aways](#)

PRESENTERS



*Presenter: Aya Takase
Director of X-ray Imaging*

Aya holds a MA in physics from Tokyo University of Science and has been with Rigaku for 23 years. She started in the X-ray Diffraction Application Lab and transitioned to X-ray Imaging in 2017. Her goal: Help non-expert X-ray users achieve expert results with less time and effort. She has worked on many projects designing automated and user-friendly X-ray instruments and analysis software. She is very passionate about helping people learn more about X-rays and working with X-ray users to solve their specific problems.



*Co-presenter: Angela Criswell
Senior Scientist*

Angela holds a PhD from Rice University and has been with Rigaku for 19 years. She started in the Macromolecular Crystallography Applications lab focusing on X-ray techniques to study structural biology. She has gained expertise in a number of X-ray methods in her tenure at Rigaku, including small angle X-ray scattering and X-ray computed tomography. Angela likes working with customers to find the best fit for their samples while addressing their specific experimental questions.



*Host: Tom Concolino
Southeast Regional Account
Manager*

Tom holds a PhD in Chemistry from Mississippi State University and has been with Rigaku for 20 years. He started out in the Small Molecule Crystallography Applications Lab before transitioning to the sales team in 2002. He has been on the front lines helping clients save on time, cost, and effort while pushing forward to support the never-ending need to innovate and explore new materials and structures. From academia to mining to pharmaceutical research, Tom has learned the value of bringing a fresh perspective to each customer application while utilizing his vast experience to collaborate on the best fit solution for each and every customer.

Why do we process images?

The ultimate goal of X-ray CT ([computed tomography](#)) data analysis is to quantify the features we observe in the CT images. To quantify features, we need to [segment](#) the CT images. Segmentation is always the first step of data analysis. However, image segmentation can be challenging when the noise level is high, the contrast is low, or the material interfaces are blurred.

Image processing such as denoising can help us segment challenging images. Although [machine learning](#) and [deep learning](#) based segmentation is now more accessible and widely used, it often requires an expensive software tool and computer. A combination of image processing and simple thresholding segmentation is still an inexpensive, often free, alternative solution.

EXAMPLE

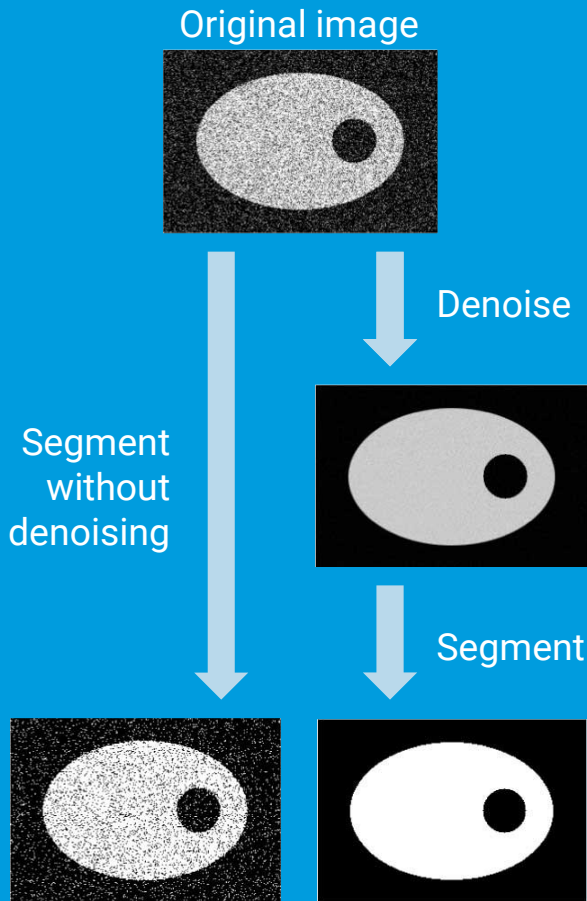


Figure 1: Comparison of segmentation with and without applying a denoising filter.

Resolution, contrast, and various [artifacts](#) often need to be optimized by adjusting the measurement conditions.

Contrast can be also simply enhanced by changing the gray level window setting so that we can distinguish different gray levels on a computer screen.

Noise, on the other hand, is always there and is one of the most common problems we face in CT image segmentation. It is also a relatively easy one to deal with by applying image processing.

Example

In Figure 1, we can see denoising minimizes the noise level and helps [thresholding](#) generate clean segmentation of the object (bright area) and the background (dark area). A median filter and Otsu binarization were in this example.

2

How does denoising work?

Denoising is a process that reduces the noise level of the original data. In case of X-ray analysis, the majority of the noise is shot noise, which reduces as the total photon count increase. So by combining multiple data points, we can increase the total photon count. And we use their average as a low noise value. In practice, we replace the value of a data point with an average of the nearby data points, as shown in Figure 2.

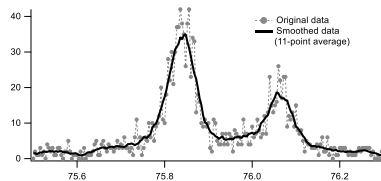


Figure 2: Smoothing of one-dimensional data by replacing each value by the average of -5 to +5 points (11-point average).

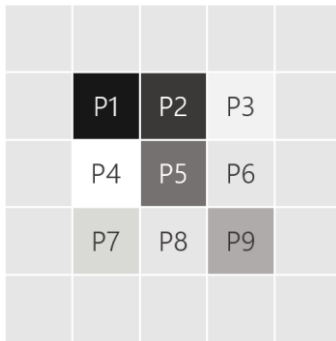
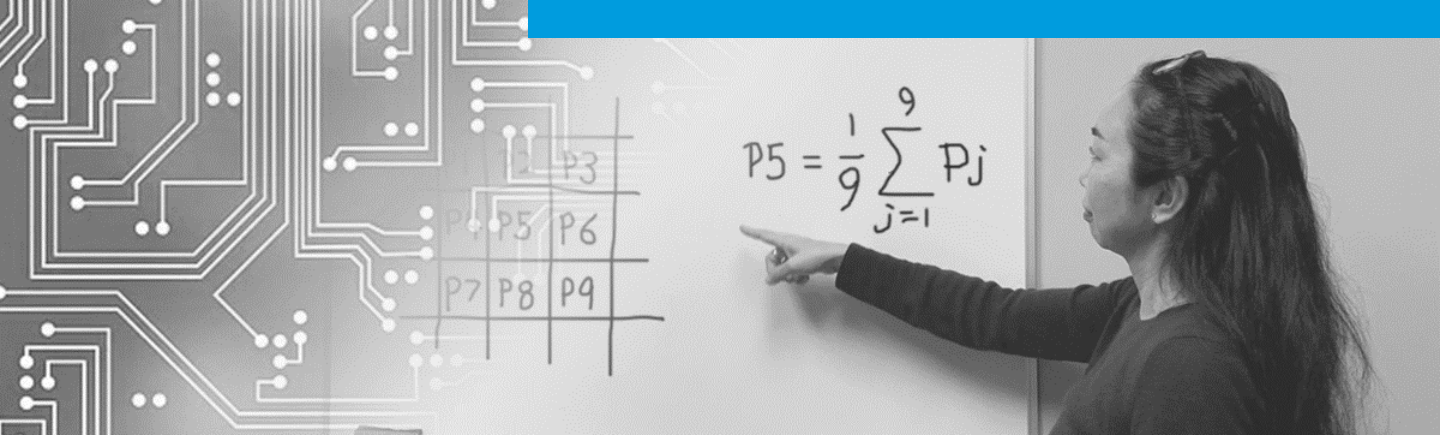


Figure 3: 5 x 5 pixels 2D grayscale image.

Denoising Filter

We can apply the same concept to two-dimensional data or images. Figure 3 is a representation of a 5 x 5 pixels 2D grayscale image. For example, by replacing the gray value of “P5” with the average of nine pixels, P1 to P9, we can denoise the image. This is called a *moving average filter*. The process can be expressed as follows:

$$P5 = \frac{1}{9} \sum_{j=1}^9 P_j$$

Mathematical Expression of Filters

It is mathematically convenient to express the filter, also called a kernel. Here is an example of the kernel of the moving average filter:

$$h_{ma}(m, n) = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

MOVING AVERAGE
FILTER

We can express an image with the gray value f of a point (x, y) . When the original and denoised images are $f(x, y)$ and $g(x, y)$, the denoising process using a $w \times w$ kernel can be expressed as the convolution of the original image and the kernel:

$$g(x, y) = \sum_{m=-w}^w \sum_{n=-w}^w h(m, n) f(x, y)$$

A moving average filter does not use weight based on the location of the center point and often blurs the interface between dark and light areas. To prevent this, we can put weight function. Here is a kernel of a *weighted average filter*:

MOVING AVERAGE
FILTER

$$h_{wa}(m, n) = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

We can also use Gaussian-shaped weight. This is called a *Gaussian filter*:

GAUSSIAN FILTER

$$h_{Gauss}(m, n) = \frac{1}{2\pi\sigma^2} \left(\frac{-m^2 + n^2}{2\sigma^2} \right)$$

The parameter σ changes the strength of a Gaussian filter. A greater σ has stronger denoising effects, though It blurs interfaces more.

Edge Preserving Filters

Although denoising is effective in reducing noise and making the segmentation process easier, blurring of the interfaces or “edges” can be a problem. A *median filter* and a *non-local means filter* are often used to preserve edges.

A median filter replaces the gray value of the center pixel by the median of the defined kernel size:

MEDIAN FILTER

$$h_{Median}(m, n) = \frac{1}{N} \text{Median}(x, y)$$

A median filter preserves edges better because it uses an actual gray value in the neighborhood instead of generating a processed value, such as an average.

Ideally, we want to calculate the average based on similar values. But the nearby pixel values are used in all of the denoising filters mentioned above. A non-local means filter assumes that similar pixel does not need to be close to the pixel of interest.

This filter widens the search area to find pixels of similar gray value and use them to calculate the mean value:

$$h_{NLM}(m, n) = \frac{1}{C} \exp \left(- \frac{\max(d^2 - 2\sigma^2, 0.0)}{h^2} \right)$$

d^2 : squared Euclidean distance
between Patches m and n

σ : noise level

h : damping factor

Further Reading

Toda, H. (2021). *X-ray CT: Hardware and Software Techniques*. Springer.
<https://doi.org/10.1007/978-981-16-0590-1>

Buades, A. et al. (2011). *Non-Local Means Denoising*. Image Processing On Line, 1, 208-212.
https://doi.org/10.5201/ipol.2011.bcm_nlm

3

How does edge detection work?

What we call an “edge” here is the interface between two materials with very different densities or two regions with very different gray levels. To detect or find the edges, we need to find where the gray level changes drastically.

We can use the first derivative of the gray level to identify the sudden change at the edges.

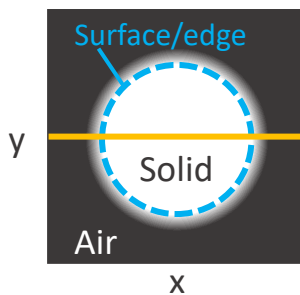
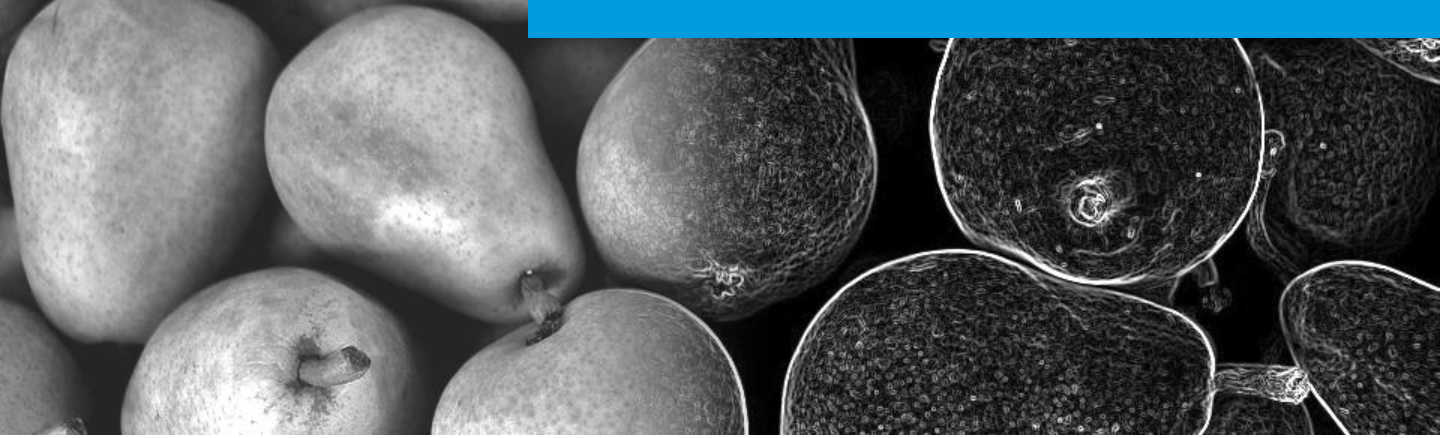


Figure 4: Schematic CT cross-section a round solid object

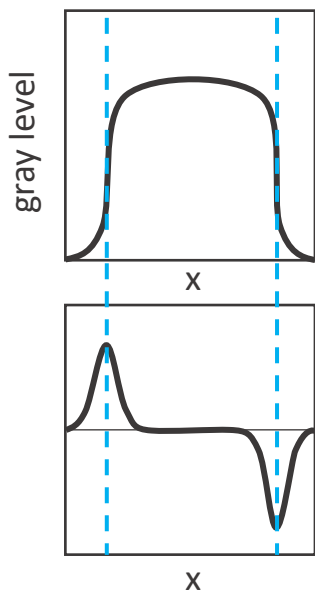


Figure 5: Profile at the center of the cross section in Figure 4 (top) and its first derivative

Edge Detection Filter

Figure 4 is a schematic CT cross-section of a round object (white) in the air (dark gray). The edge (surface of the object) is marked with a blue dotted line.

Figure 5, at the top, shows a profile of this CT cross-section at the yellow line in Figure 4. We can see the edges have a sudden rise and fall of the gray level. But these edges are not easy to define.

Figure 5, at the bottom, shows the first derivative of this profile. Now we see peaks and valleys at the edges. These points are easily identified by their maximum and minimum values.

Mathematical Expression of Edge Detection

The first order linear forward difference approximation for a first derivative is expressed as follows:

$$\frac{\partial f}{\partial x}(x, y) \approx f(x + 1, y) - f(x, y)$$

$$\frac{\partial f}{\partial y}(x, y) \approx f(x, y + 1) - f(x, y)$$

This calculation can be done by applying a linear forward difference filter;

$$h_{diff.x}(m, n) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$h_{diff.y}(m, n) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

LINEAR FORWARD
DIFFERENCE FILTER

This filter can recognize all the noise as edges. To prevent this problem, we can apply a weighted average. The result is called the *Sobel filter*:

$$h_{Sobel.x}(m, n) = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

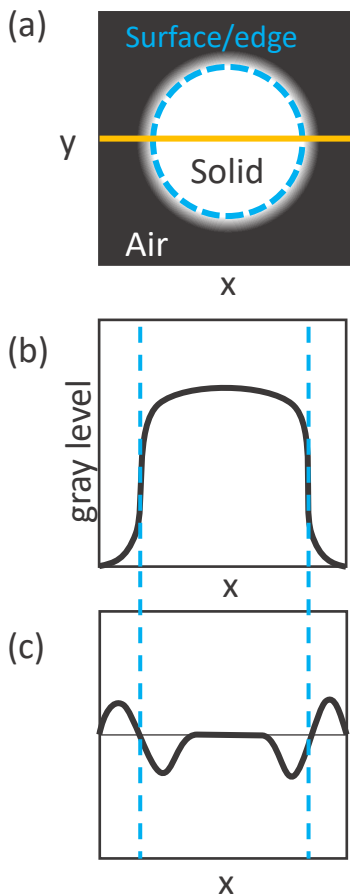
SOBEL FILTER

4

How does sharpening work?

A sharpening filter is often applied to enhance the edges. It reduces the blurring and gives an appearance of improved resolution. It is different from the contrast enhancement, which increases the gray level difference among pixels regardless of their locations, near the edge or not.

Because the sharpening should be applied to the edges, we can use derivatives of the gray level to identify the edges and use the second derivatives to “sharpen” that area.



Sharpening Filter

Figure 6(a) shows the same schematic CT cross-section of a round object (white) in the air (dark gray) we used in the previous section.

Figure 6(c) shows the second derivative of the profile (Figure 6(b)). The second derivative is zero at the edge and has a set of a small peak and valley next to the edge.

We can use these peaks and valleys to “sharpen” the edges.

Figure 6: (a) CT cross section, (b) a profile at the center of the cross section, and (c) its second derivative (bottom)

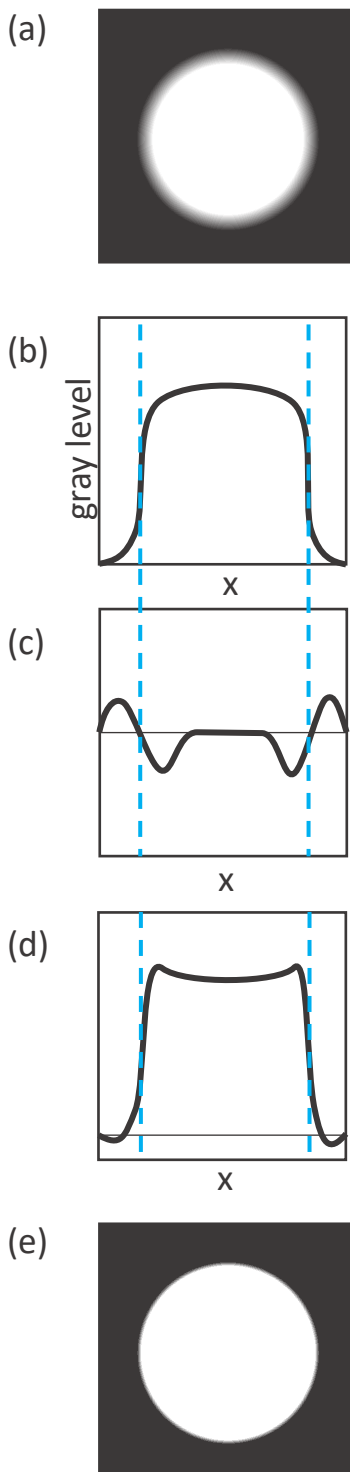


Figure 7(a) is the original cross-section. By subtracting the second derivative (Figure 7(c)) from the original profile (Figure 7(b)), we can sharpen the edges. Figures 7(d) and 7(e) are the result. We can see that the edges are “sharpened.” By applying this process, you can sharpen the blurred edges and obtain a cross-section image that looks like Figure 7(e).

Unsharp Mask

Unsharp masking subtracts a blurred, inverted, and scaled copy of the image and rescales the result to sharpen the image. This filter is called unsharp mask because it uses a blurred (unsharp) image.

Further Reading

Toda, H. (2021). *X-ray CT: Hardware and Software Techniques*. Springer.
<https://doi.org/10.1007/978-981-16-0590-1>

Figure 7: (a) CT cross section, (b) a profile at the center of the cross section, and (c) its second derivative (bottom)

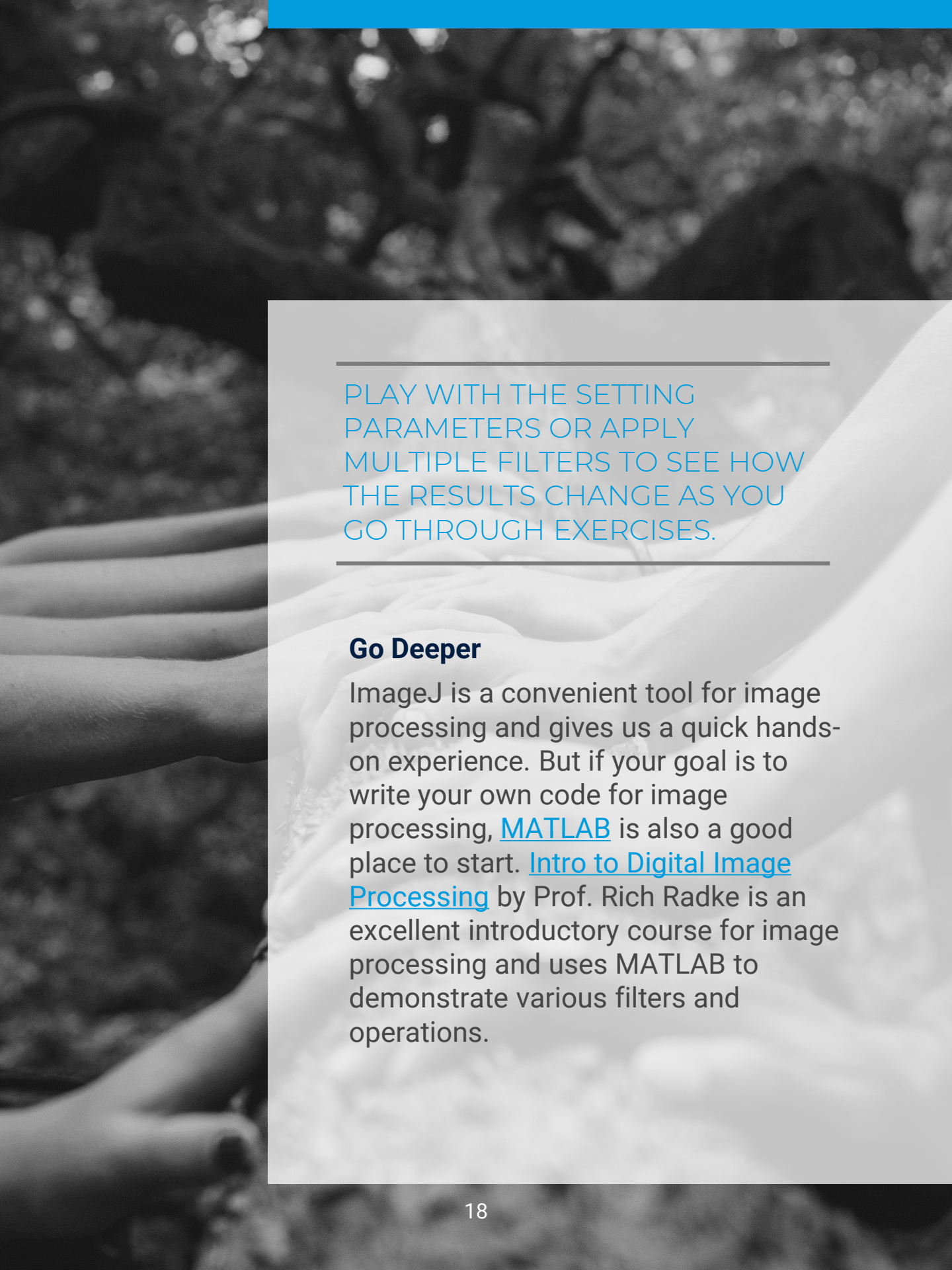
5

ImageJ hands-on exercises

Hands-on exercises help us understand how each processing filter affects images.

In this section, we will apply various filters we reviewed so far to [sample images](#) and observe their effects using an open-source [Fiji distribution of ImageJ](#).

If you are new to ImageJ, watch this [Mini Tutorial: ImageJ Getting Started Guide](#).



PLAY WITH THE SETTING
PARAMETERS OR APPLY
MULTIPLE FILTERS TO SEE HOW
THE RESULTS CHANGE AS YOU
GO THROUGH EXERCISES.

Go Deeper

ImageJ is a convenient tool for image processing and gives us a quick hands-on experience. But if your goal is to write your own code for image processing, [MATLAB](#) is also a good place to start. [Intro to Digital Image Processing](#) by Prof. Rich Radke is an excellent introductory course for image processing and uses MATLAB to demonstrate various filters and operations.

Playing with Filters

Let's make some custom filters and see how they change the original image.



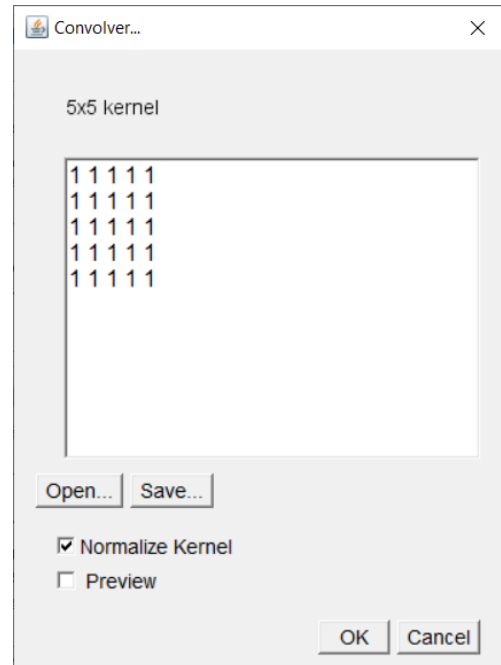
Original image: Sunflower
256 x 256 pixels, 8-bit grayscale

Applying custom filters

(Process Menu → Filters → Convolve)

In the Convolve... window, you can make your own filter (kernel).

Here is an example of a 5x5 moving average kernel.



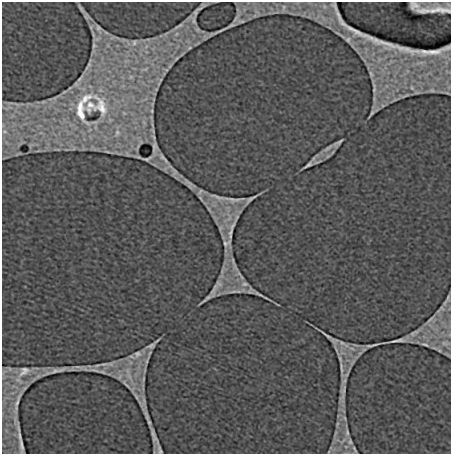
Turn on "Preview" to see the effect of the kernel.



Try different kernels to get a feel of how they work.

Applying Denoising Filters

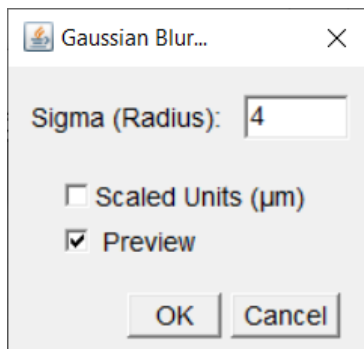
Let's apply different denoising filters to the CT cross-section of a foam material.



Original image: CT cross section of a foam material. 565 x 565 pixels, 1.06 microns/pixel, 8-bit grayscale

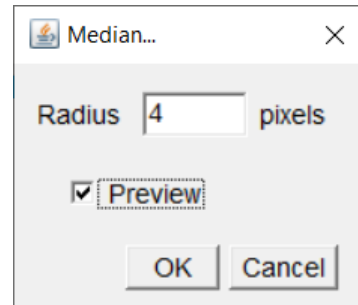
1. Gaussian filter

(Process Menu → Filters → Gaussian Blur)



2. Median filter

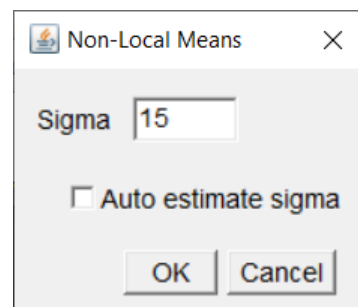
(Process Menu → Filters → Median)



3. Non-local means filter

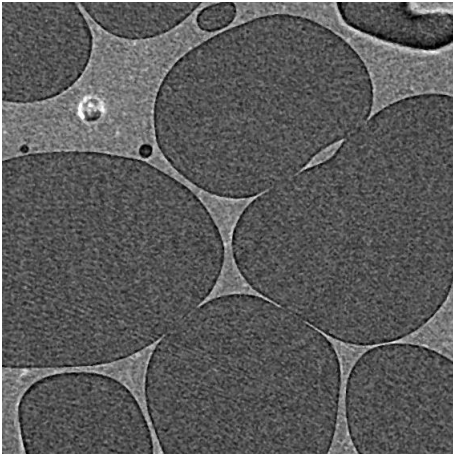
(Plugins Menu → Non-Local Means Denoising*)

*<https://imagej.net/plugins/non-local-means-denoise/>



Comparing Results

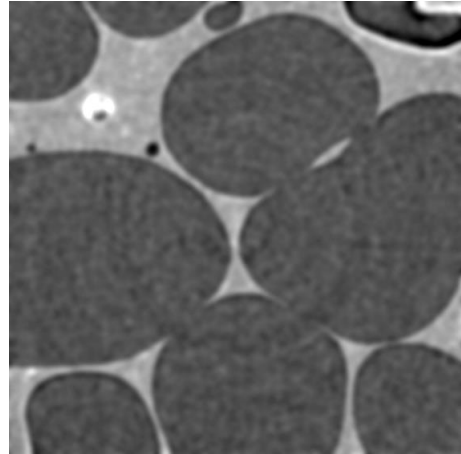
The original image was collected in 20 minutes and some noise.



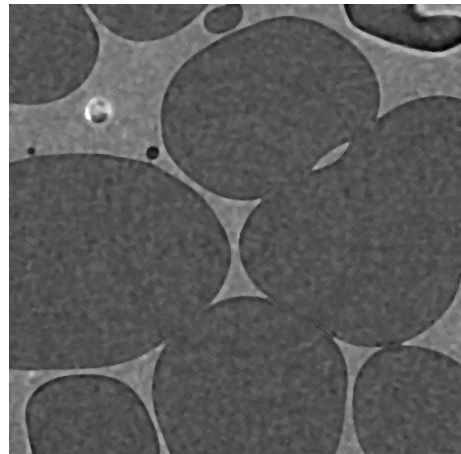
The results from the three filters show a significant difference.

A Gaussian filter reduces the noise but blurs the edges between polymer (light gray) and air (dark gray). A median filter preserves the edges well. A non-local means filter can reduce the noise even more than these two while preserving the edges very well.

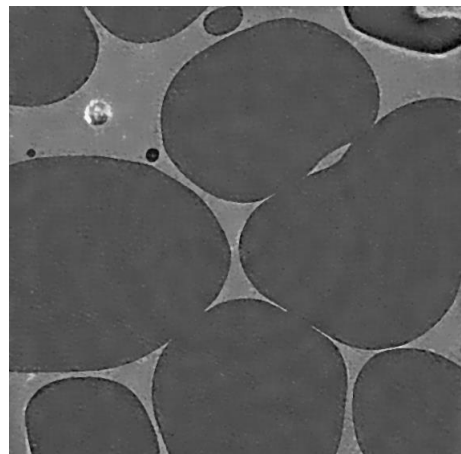
1. Gaussian filter, $\sigma = 4$



2. Median filter, radius = 4



3. Non-local means filter, $\sigma = 15$

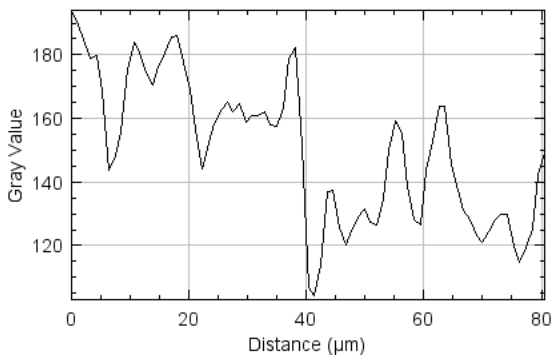
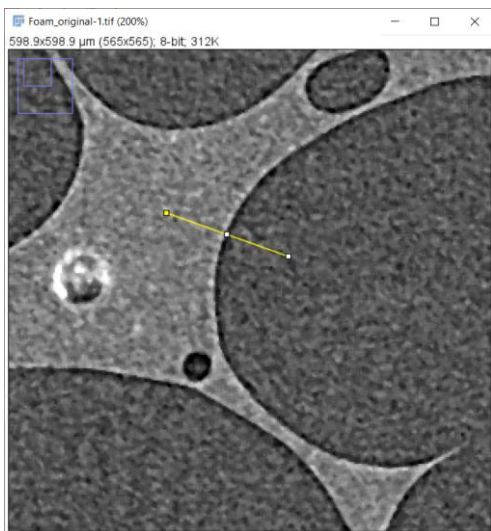


Quantitative Evaluation

Check the profile across the interface of polymer (light gray) and air (dark gray), standard deviation, and histogram of each image and quantitatively evaluate the effects of each filter.

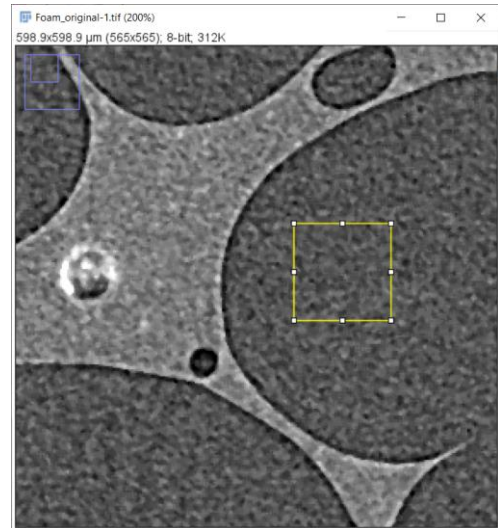
Plotting a profile

(Analyse Menu → Plot Profile | Ctrl + K)



Calculating standard deviation

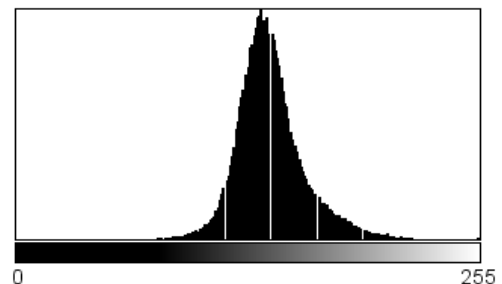
(Analyse Menu → Measure | Ctrl + M, with "Standard Deviation" selected in "Set Measurement...")



Standard deviation: 10.94

Comparing histograms

(Analyse Menu → Histogram | Ctrl + H)

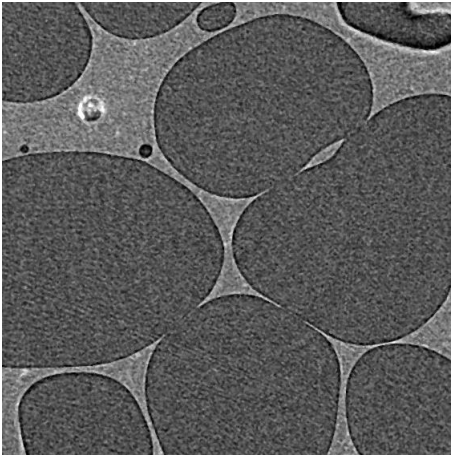


N: 319225
Mean: 140.228
StdDev: 19.515
Value: 143

Min: 43
Max: 255
Mode: 135 (9822)
Count: 8041

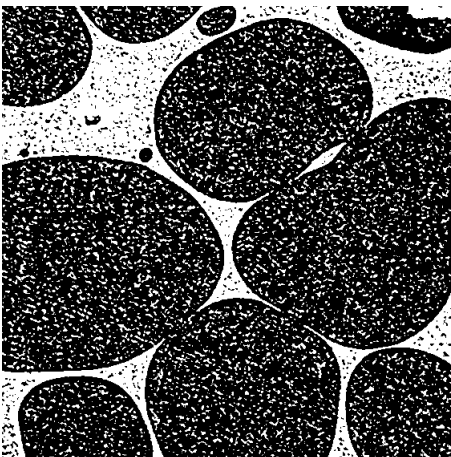
Compare Thresholding Segmentation Results

At each stage, apply thresholding segmentation (binarization) to see how each filter changes the segmentation results.

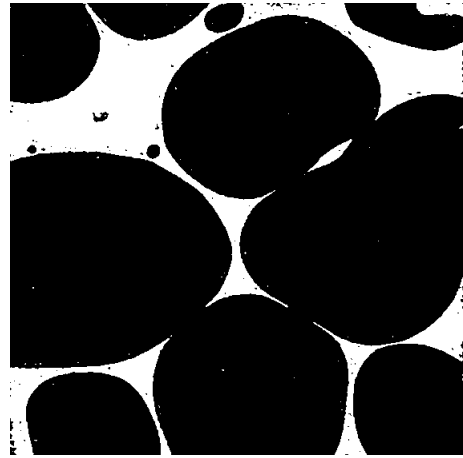


Original Image + Otsu Binarization

(Image Menu → Adjust → Threshold | Ctrl + Shift + T)



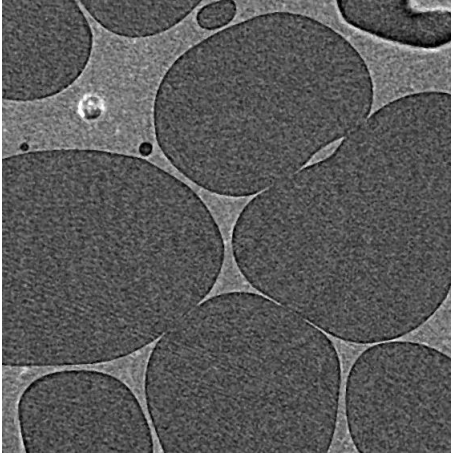
Non-local Means Filter + Otsu Binarization



In this example, we can see the benefit of denoising filters. The segmentation result is much cleaner and more suitable for the quantitative analysis that follows compared to that from the original image.

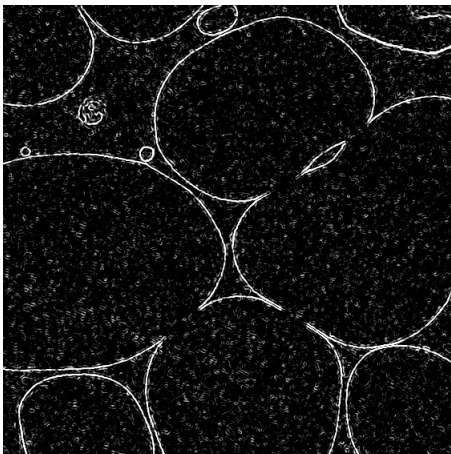
Applying Edge Detection Filter

Let's apply a edge detection filter (a Sobel filter) to this image:



Find edge filter

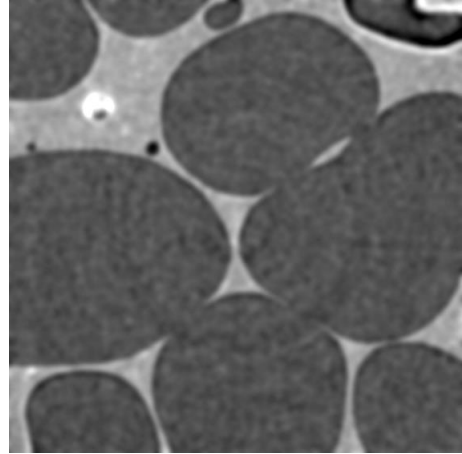
(Process Menu → Find Edges)



We can clearly see the edges between the polymer and air.

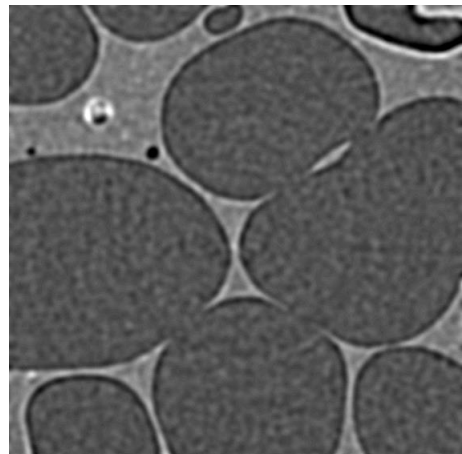
Applying Sharpening Filter

Let's apply a sharpening filter to the blurred version (Gaussian blur):



Unsharp mask

(Process Menu → Filters -> Unsharp mask)



(Radius = 4, mask weight = 0.8)



ABOUT THE TOOLS

Fiji: A distribution of ImageJ

ImageJ is an open-source image processing program for scientific multidimensional images. We used Fiji distribution in this workshop. You can download Fiji for Windows, Mac, or Linux from this link:

<https://imagej.net/software/fiji/downloads>

Non-local means denoise plugin

A non-local means denoise plugin used in this workshop is available from this link:

<https://imagej.net/plugins/non-local-means-denoise/>

Original work: Buades, A. et al. [doi:10.5201/ipol.2011.bcm_nlm](https://doi.org/10.5201/ipol.2011.bcm_nlm)
Darbon, J. et al. [doi:10.1109/isbi.2008.4541250](https://doi.org/10.1109/isbi.2008.4541250)

Takeaways

Segmentation is always the first step of data analysis. Image processing such as denoising can help us segment challenging images. Although [machine learning](#) and [deep learning](#) based segmentation is now more accessible, when we don't have access to these tools or they are too computationally taxing, a combination of image processing and simple thresholding segmentation is still an effective alternative solution.

Most commonly used filters, such as Gaussian and unsharp filters, can be expressed as a kernel in the form of a matrix. By convoluting the kernel and the original image, we can obtain a processed image. Depending on the desired outcome, you can also design your own kernels.

LET'S LEARN TOGETHER

Many people have learned what X-ray computed tomography (CT) is, how it works, and where it can be helpful in our webinar and workshop series. All recordings, application examples, a publication list, and blog articles are available at imaging.rigaku.com.

Subscribe to [the email updates](#) to stay informed about new articles, recommended publications and books, and upcoming learning events.

CONTACT US

imaging@rigaku.com