From Image to Data Using Common Image-Processing Techniques

Laura R. Sysko[1] and Michael A. Davis[1]

[1]Nikon Instruments Inc., Melville, New York

ABSTRACT

A digital microscopy image is an array of number values, which with adequate contrast can be interpreted as spatial information. Through processing and analysis by mathematical means, using computer-assisted imaging software programs, raw image data contrast can be enhanced to improve the extraction of image features for measurement and analysis. This mathematical feature extraction (referred to as segmentation) provides the basis for general image processing. The methods discussed in this unit address common image analysis challenges such as object counting with touching objects, objects within other objects, and object identification in a field with uneven illumination or uneven brightness, along with step-by-step procedures for achieving these results. *Curr. Protoc. Cytom.* 54:12.21.1-12.21.17. © 2010 by John Wiley & Sons, Inc.

Keywords: image analysis • segmentation • image processing • image measurement • digital imaging • thresholding • automated counting

INTRODUCTION

This commentary provides a focused discussion on general microscopy image analysis methods for gathering data from the array of pixels within a digital image. There are a range of considerations to improve the digital representation and quantitation of the specimen. It is critical to understand sample preparation, sampling frequency, sources of noise, matching objectives with pixel size of the detector, illumination conditions, image registration, and image restoration through deconvolution (see *UNIT 12.19*; Zwier et al., 2004; Pawley, 2006a,b; Waters, 2009). However, these significant aspects of microscopy and imaging are outside of the scope of this commentary, but should be addressed and are included in the referenced articles. This commentary includes an overview of the digital image and concepts of image processing and contrast enhancement. In addition, three protocols are detailed in separate cases, demonstrating the techniques for addressing common image analysis challenges.

Nikon Instrument's NIS-Elements Advanced Research software was used for the screenshots and analysis techniques in this unit. Other commercially available or free image-processing programs, such as NIH's software ImageJ, allow similar techniques to some of the examples shown.

IMAGE ANATOMY

Digital images can be fundamentally broken down into a finite grid of squares called "pixels" (Inoué and Gliksman, 2003). When using either a charge-coupled device (CCD) camera or a photomultiplier tube (PMT) on a laser scanning confocal, digitization is the conversion of the signal from the image detector into intensity values (Pawley, 2006a).

The array of pixels holds the x, y location and their associated value of signal, which are referred to as gray-levels or gray shades (Inoue and Spring, 1997).The number of discrete gray shades is dependent on the bit number. For example, a pixel in an 8-bit resolution image (2^8) has a range of 256 gray shades (0 to 255), a pixel in a 12-bit image (2^{12}) has 4096 available gray shades, and a pixel within a 16-bit image (2^{16}) has 65,536 available gray shades. As the bit depth increases, so does the ability to represent finer detail at each pixel and its neighborhood (Hinchcliffe, 2003). It is generally accepted that images obtained by scientific-grade detectors exhibit linear intensities from minimum to maximum and therefore quantitative and comparison data can be derived from these images.

As shown in Figure 12.21.1, each pixel has a corresponding intensity/gray-scale value, which is the foundation for computer-assisted quantitative image analysis. Information that

Current Protocols in Cytometry 12.21.1-12.21.17, October 2010
Published online October 2010 in Wiley Online Library (wileyonlinelibrary.com).
DOI: 10.1002/0471142956.cy1221s54

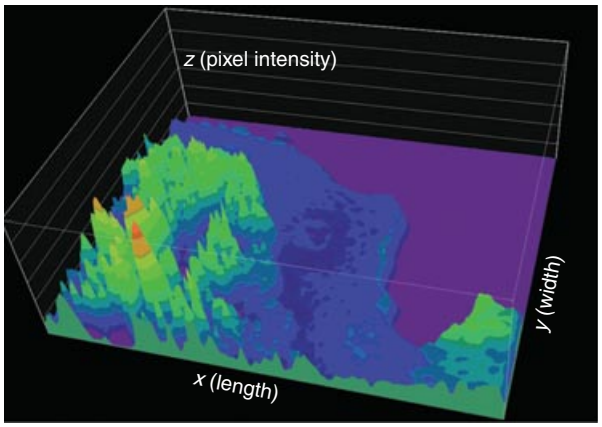


Figure 12.21.1 At each pixel, there is a corresponding gray-scale value, represented here in a 2D Intensity Surface Plot. The *x* axis represents the length dimension, the *y* axis represents the width dimension, and the *z* axis represents the pixel intensity. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

can be obtained from an image is dependent on how the specimen was prepared and acquired. If variables such as probe concentration/storage, transfection protocols, and camera exposure settings to name a few, are held constant across control and treated groups of images, then comparison and quantitation is theoretically possible.

Raw pixels are unitless in terms of distance and intensity. A common method of distance calibration consists of acquiring an image of a stage micrometer and creating specific calibration files per magnification setting of the microscope, or by calculating the pixel size in real units based upon the total magnification of the microscope. Next, this calibration can be assigned to the images acquired at the selected magnification. All measured pixels in the calibrated image will subsequently be reported in real distance units such as microns or millimeters. Once calibrated, measurements such as length and area are possible.

Intensity calibration can be much more challenging because oftentimes calibration standards are not available. Instead, users tend to try to minimize the number of variables (sample preparation, lamp intensity, exposure time, camera gain settings, etc.), such that these variables are held constant over the course of experiments. Then intensities can be compared between images and changes in intensity can be assessed.

Depending on the dimensions of data that are acquired, other measurements can be created. For example, volumetric measurements can be obtained by image stacks acquired as Z series, and velocity and tracking measurements can be obtained by time-lapse image acquisitions.

Most computer-assisted image analysis techniques, such as automated counting and morphology studies, require some type of "feature extraction" method to calculate data. Essentially, the user must help the computer software determine what is important to count versus unimportant by increasing the contrast/difference between items of interest and "background" information.

IMAGE PROCESSING

Image processing, commonly described as "segmentation," is a technique whose goal is to improve contrast in image data, in order to extract features of interest. Segmentation is also described as identification of one image feature from another (Inoué and Spring, 1997). Once these features have been extracted, they can more easily be quantified by means of intensity, distance measurement, or other analyses.

Thresholding

Segmentation techniques oftentimes begin with a process known as "thresholding," or creating a "binary mask," which is a quick method to identify, with the computer software, areas of an image to include and areas of an image to ignore. With sufficient contrast, objects of interest may then be "detected," resulting in masking binary image components, where each pixel is either "on" or "off" (Cox et al., 1998). Thresholding is a binary technique (meaning, area inclusion can be either "yes" or "no") and can be applied to images

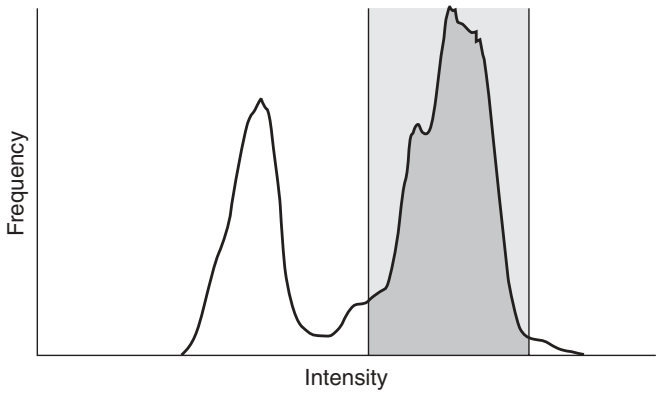


Figure 12.21.2 The portion of the histogram shown in gray is the area that relates to parts/signal of the image that are of interest.

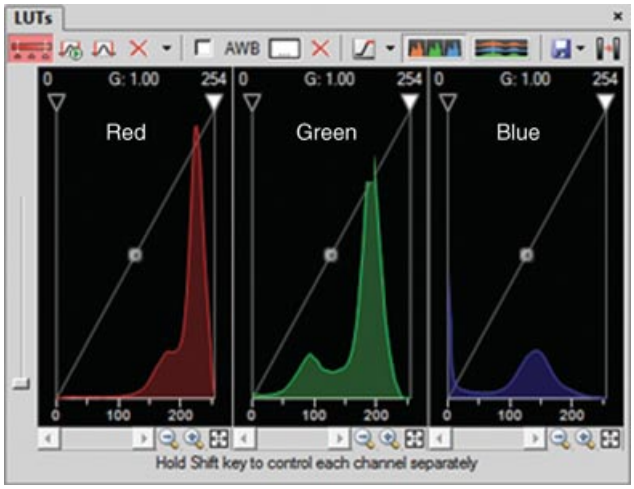


Figure 12.21.3 RGB histogram of the original image shown in Figure 12.21.5. This figure shows that it is more difficult to delineate the larger "brown" nuclei for the red, green, and blue histogram components of the image. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

by channel, color, hue, intensity, or saturation. Case 1 below shows a binary layer created from the thresholding.

Establishment of a threshold can be aided by visualizing the image's histogram, which is a two-dimensional area plot of the intensity values of an image on the *x* axis and the number of pixels found at each intensity on the *y* axis. The histogram displays the number of pixels in the image at each gray level (Inoué and Spring, 1997). In thresholding an image, the user chooses a portion of the histogram (which with simple thresholds is continuous and without gaps) that relates to parts of the image that are of interest (see Fig. 12.21.2).

Generally, thresholding is easiest if performed on monochrome (one channel) data, because it is easier to segment data from one channel at a time than to attempt to pick aspects of multiple channels (e.g., picking "brown" from an RGB image histogram since is not an easy method to define brown by its red, green, and blue components; see Fig. 12.21.3). A good place to start with thresholding is to convert color images into monochrome images. As shown in Figure 12.21.4, it is easier to view the dark and light objects from a histogram of a monochrome image.

Different strategies can be employed when converting to monochrome, if not already using monochrome images. One strategy is to look at each color channel individually on a color image to determine if one channel creates better contrast than another. The channel containing the highest contrast is the best one to choose for use for thresholding later on.

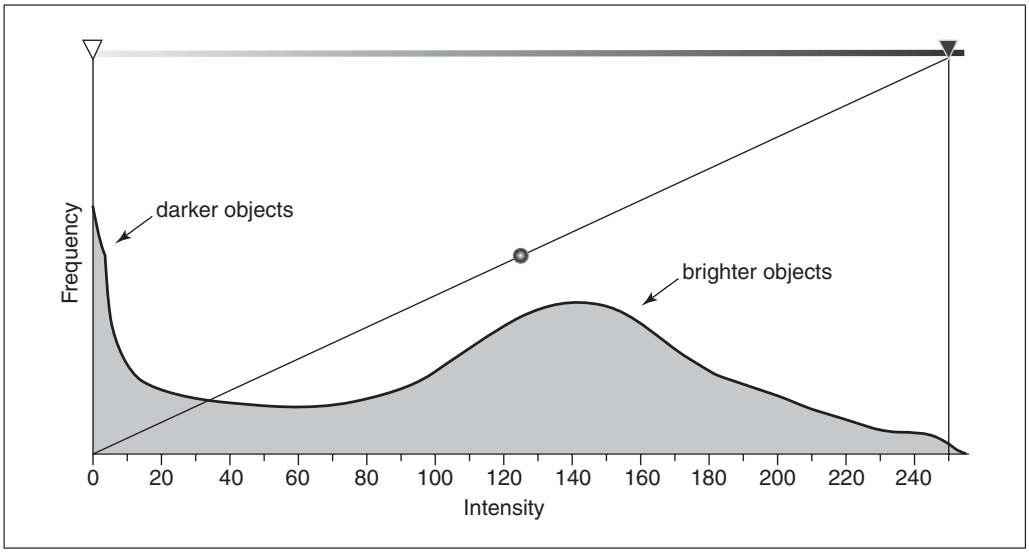


Figure 12.21.4 Monochrome histogram of the original image shown in Figure 12.21.5. The *x* axis represents intensity, and the *y* axis represents frequency of pixels. Dark objects are easier to visualize than bright objects by locating the two peaks (dark objects at 0, bright objects around 140).

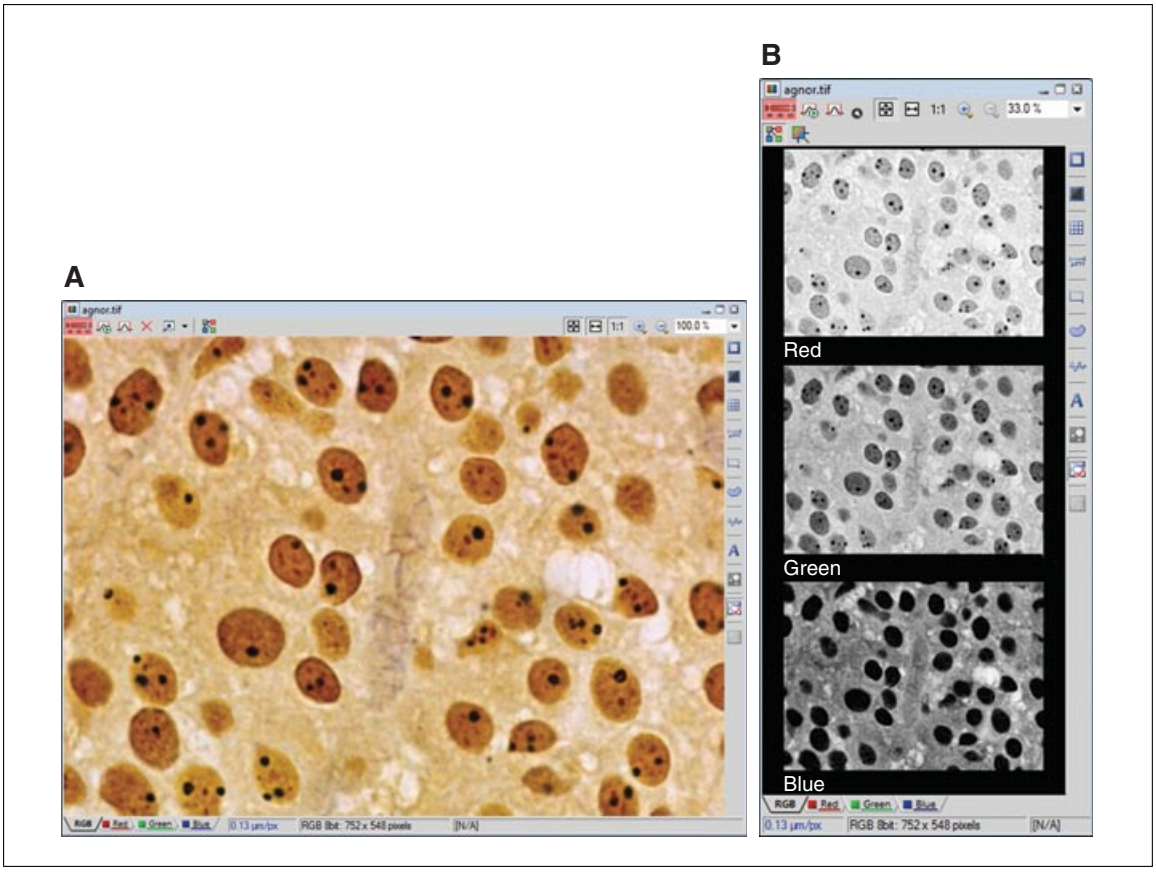


Figure 12.21.5 (**A**) Original RGB Image of nuclei labeled with diaminobenzidine (DAB). (**B**) Red, Green, and Blue channels shown individually. Separating the channels can help improve contrast for further image processing steps. Note that the contrast is highest in the blue channel between the nuclei and the surrounding tissue. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

Take for example the nuclei staining from Figure 12.21.5A; if the image is split into its individual RGB channels (Fig. 12.21.5B), one can observe how great the contrast is in the blue channel between the nuclei and the surrounding tissue. If the goal of the analysis were to count the nuclei, choosing the blue channel for thresholding would be a good first step. The goal of thresholding is to highlight areas of interest and ignore background items. Again, to best prepare for thresholding, seek methods to increase the contrast.

Contrast Enhancement

Successful thresholding requires adequate contrast. At times, the subject acquired or the imaging technique used does not produce adequate contrast for thresholding. Usually this is due to one of two reasons:

1. The image intensity variation is very small (low contrast).

2. The areas of interest in the image are "touching" and the threshold does not separate them into discrete individual objects.

Brightness and contrast adjustment

Low contrast can be dealt with by a variety of techniques. The first is to adjust the brightness and contrast of the image; however, it is important to note that these adjustments will alter the image intensities for use with analysis. Thus, caution should be exerted if using this technique when performing intensity analysis. Most software for image analysis allows methodology to modify the image in order to segment it and then copy that segmentation "mask" back to the original unaltered image for intensity analysis. An example of how to create a mask is provided in Case 3.

Brightness and contrast adjustment simply shifts the relationship between actual image intensity and the display intensity (normally a 1:1 relationship). By adjusting either brightness or contrast (typically: contrast), segmentation between areas of interest and background may be possible.

Kernel-Based Image Processing

When these simple changes are not enough, there is a host of tools for processing images in an effort to improve their contrast. Most of these tools modify the intensity of the image, so always be aware of what image processing is doing to image data before making measurements, especially intensity measurements.

Image processing techniques generally involve mathematical processes that can be applied to pixels or groups of pixels or "neighborhoods" (group by group) across an image (Russ, 1995). These processes are generally referred to as kernels. Kernels have been developed to sharpen, smooth, find edges, repair uneven illumination problems, and many other techniques.

In general, it is important to know the result of a kernel-based processing technique more than the details of how the mathematical processes are performed. However, it is critical to note that kernels can vary in size (often referred to in their names as $n \times n$) and shape, as shown in Figure 12.21.6.

To simplify these parameters, the larger a kernel is, generally the stronger the processing result. In addition, the shape of the kernel, as shown in Figure 12.21.7, should generally be based on the shape of the areas or objects of interest (e.g., linear versus round).

Usually more than one kernel technique may be applied on an image to improve the contrast adequately enough for thresholding and ultimately, data extraction. However, oftentimes the same types of kernels are generally used to achieve better image contrast on most images, thus this unit outlines some of the most commonly used kernels and their results. Kernels can be applied to raw image data or to binary image data (binary data obtained as a result from the threshold).

There are two general kernel techniques for image segmentation that can be applied depending on the input image type and challenge:

1. *Edge detection technique:* Tool designed for accentuating the edges of objects rather

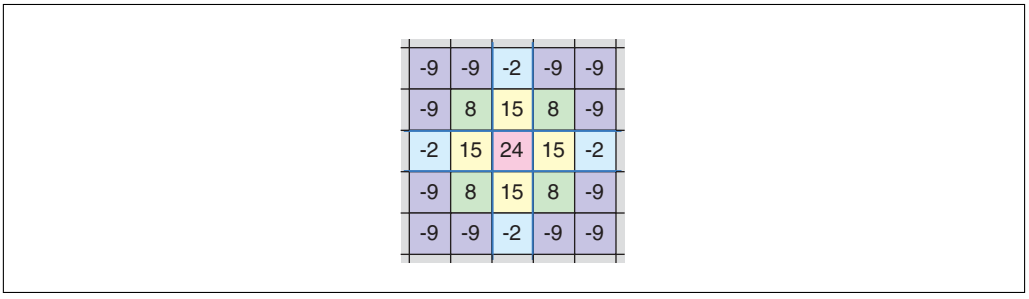


Figure 12.21.6 The Mexican hat 5 × 5 kernel is a mathematical array that is 5 rows by 5 columns of pixels wide, and is applied to each pixel in the image and its two neighbors in all directions, pixel by pixel.

than the objects themselves. As a last step, make sure to invert the selection (the threshold) from edges back to objects so that the segmentation reflects the objects and not the space between the objects. This approach is useful for a field of touching objects (e.g., tissues).

2. *Contrast enhancement:* Tool employed to allow simple thresholding when not all items of interest in an image field may be the same brightness, or there is an unevenness of illumination causing some areas of the image to be brighter than others.

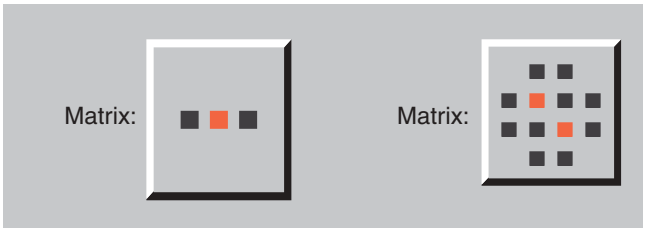


Figure 12.21.7 Matrices can have variable shapes. Linear shapes process linear structures best; round shapes process round structures best.

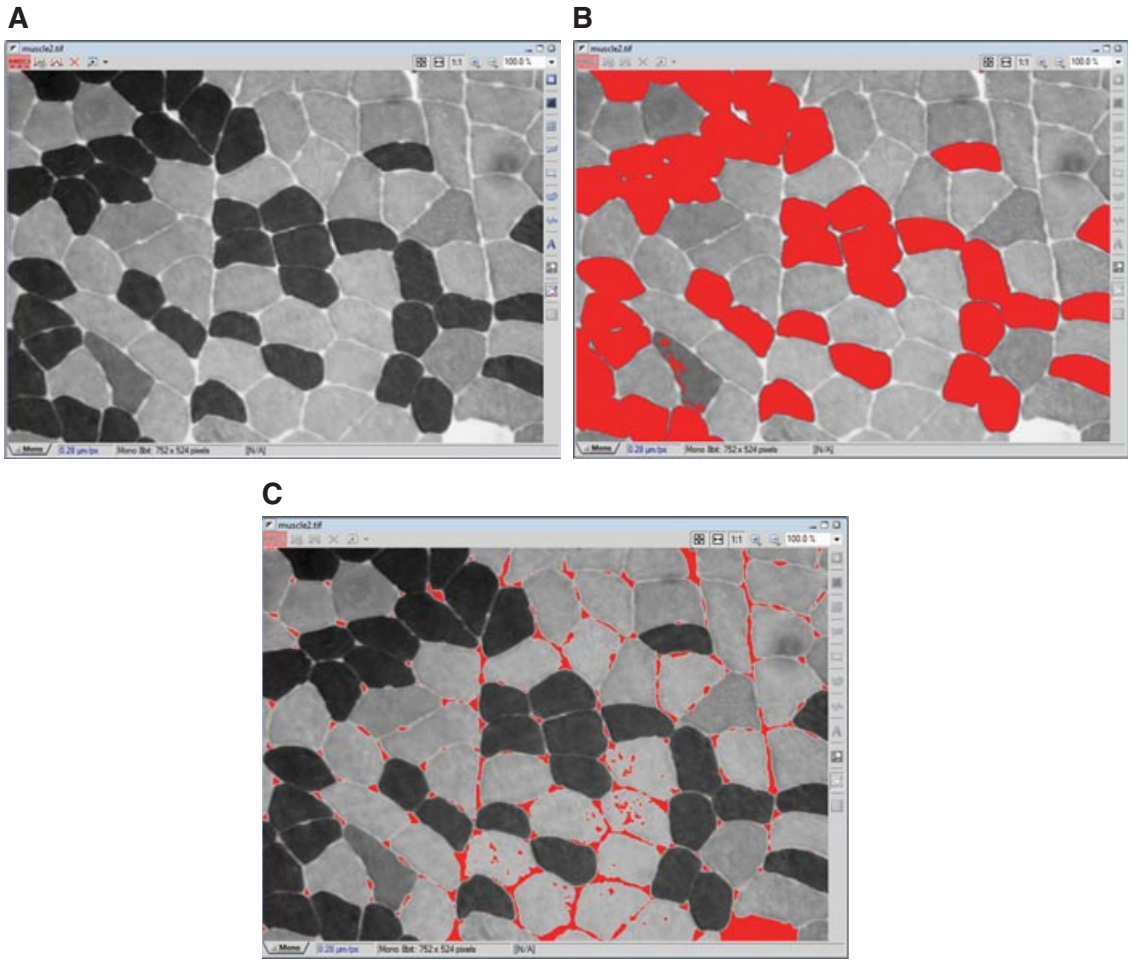


Figure 12.21.8 Using an original image for thresholding. (**A**) Original Image. (**B**) First attempt is to threshold the darker segments; because the image has "touching" areas of interest, the threshold will not result in separate discrete objects. (**C**) Second attempt is to threshold the image based on the white spaces between the objects, the threshold will select unwanted objects. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

The following are example use cases for both types of images.

Case 1: Field of touching objects

This section uses an example of a cross-section of muscle to illustrate image processing for edge detection. In the case of the muscle fiber image, standard thresholding to select the muscle fibers does not accomplish the goal of identifying each individual muscle fiber because the objects are "touching" (they lack enough contrast to be adequately thresholded). In addition, the fibers are of different intensities, so not all of them can be thresholded at the same time (Figure 12.21.8A,B).

Alternatively, attempting to threshold on the white spaces between the muscle fibers yields similar results, as shown in Figure 12.21.8C. This strategy is also not producing the desired results.

The following steps will highlight the workflow for thresholding the area around the objects using a kernel-based method. Using image kernels is one method to try to improve the image contrast for thresholding either the fibers themselves or the spaces between them. Since the fibers are of two different types and are of different intensities, it is better to attempt to threshold on the area surrounding them instead.

Gradient contrast enhancement is a commonly used edge detection kernel. Essentially, this tool finds areas where pixel intensities change from high to low or low to high intensity. The goal is to outline the spaces between the muscle fibers.

1. Add a preprocessing smoothing operation, if needed, as shown in Figure 12.21.9A,C. Because edge detection looks for changes in intensities, oftentimes it is recommended to smooth the image before attempting edge detection techniques, because any "noise" in the image will result in being detected as an "edge." Commonly, users abandon edge detection because the results will show the edges that were desired, but in addition, false edges due to noise (as shown in Figure 12.21.9B) are also shown. Most

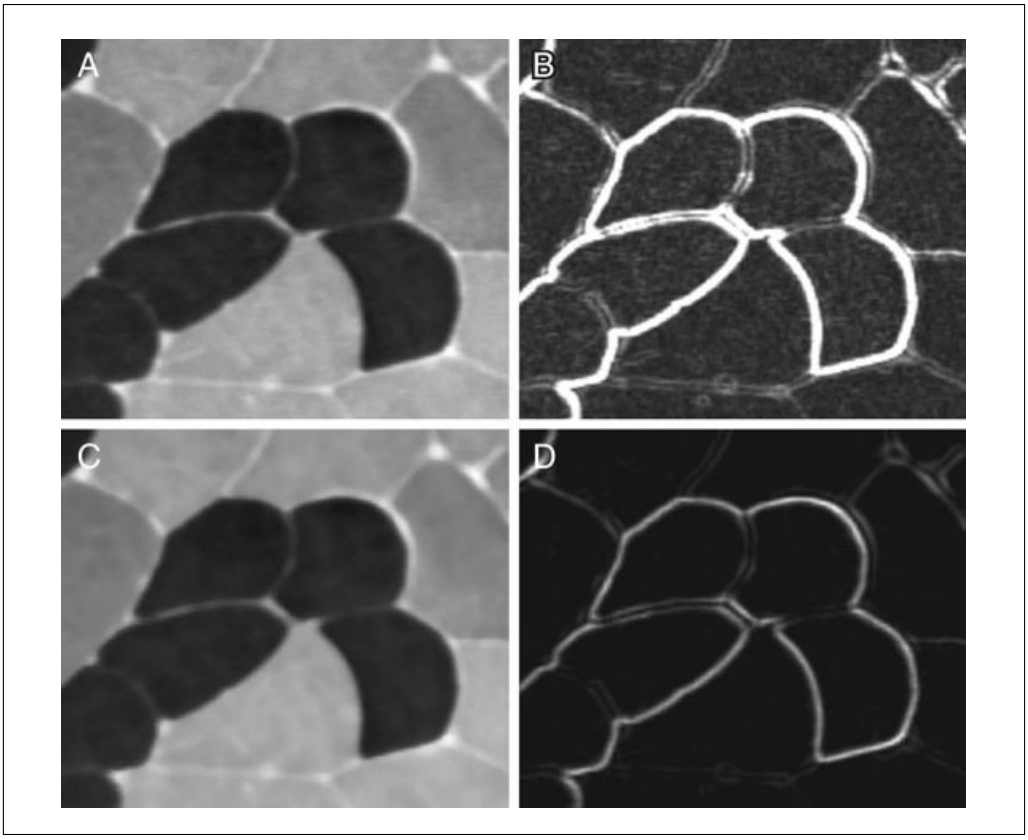


Figure 12.21.9 Image processing steps such as smoothing kernel and edge detection to improve contrast for successful thresholding. It is important to perform the smoothing prior to the edge detection. (**A**) Original image. (**B**) Original image after edge detection alone produces false edges. (**C**) Original image after smoothing kernel was applied to reduce "noise." (**D**) Smoothing kernel and then a subsequent edge detection/gradient contrast minimizes the false edges because of the noise reduction.

image-processing software has a smoothing kernel, or a median filter kernel that you can apply to the data; either can produce the desired effects.

2. Apply an edge-detection kernel-based operation to the image, as shown in Figure 12.21.9C,D. As discussed in step 1, it is recommended to perform the smoothing prior to the edge detection step in order to avoid false edges.

3. Inspect the edge detection results, as shown in Figure 12.21.10. The result of applying edge detection/gradient contrast to the muscle image is that the edges between the fibers become accentuated because each of these edges was formed by a gradation in intensity in the image.

4. Create a threshold of the image, as shown in Figure 12.21.11. As a result of the edge detection, now thresholding on light areas has a much more successful result.

5. Refine and edit the threshold results. The threshold result from step 4 is not yet optimal. The gradient kernel has missed some borders, overall it has formed double lines, and has included some of the "intracellular spaces," not the "muscle." The next step is to attempt to further improve our threshold. The goal is to improve the threshold by attempting to connect broken borders, remove double lines, and ultimately, invert the threshold to highlight muscle and not borders.

a. Use Binary Processing Kernels. Now that the image has been thresholded, a "binary"

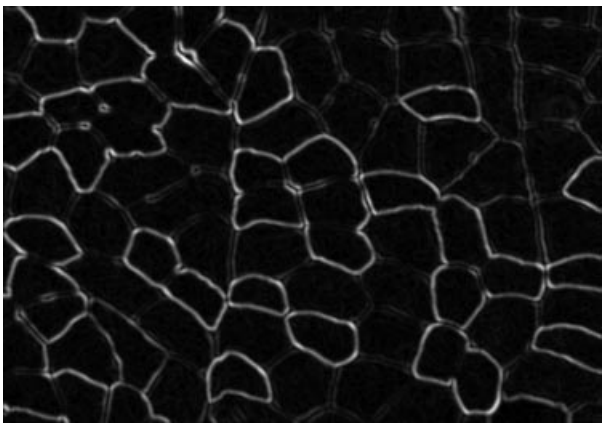


Figure 12.21.10 Result of gradient contrast edge detector kernel—the fibers are outlined in bright pixels.

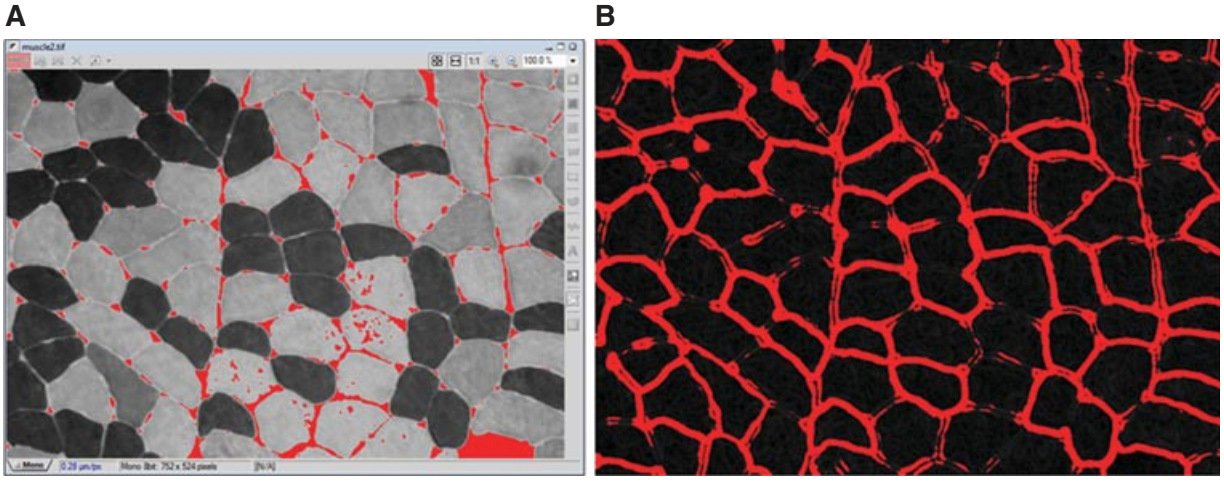


Figure 12.21.11 Example of successful threshold after edge detection/gradient contrast operation. Panel **A** shows an unsuccessful attempt to threshold between the fibers on the raw image. Panel **B** shows thresholding between the fibers after gradient contrast edge detection. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

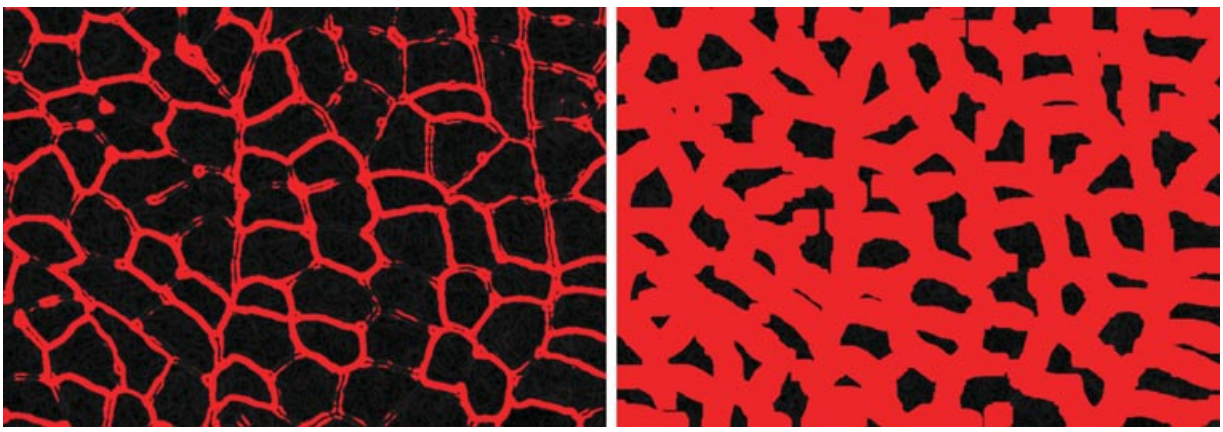


Figure 12.21.12 Resulting binary layer after dilation step. Panel **A** shows the original threshold and panel **B** shows the result of dilating the original threshold 20 times. All of the double-lines have been filled in, and most of the broken gaps have been re-connected. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

layer is available. Within this binary layer, the pixels have two optional values: value = 225, which is the area of interest and labeled with the color overlay; and value = 0, which is the rest of the image defined as "black" (Inoué and Spring, 1997). Binary processing kernels can be applied at this point to the binary layer. The binary kernel processing is similar to those performed earlier on the nonbinary layer, but of course much simpler to perform than standard image kernels that apply to multiple gray shades.

b. Use Dilation Kernel Operation. To close the broken borders and remove double lines, the binary kernel "dilate" can be used (see Fig. 12.21.12). Dilation is simply a repetitive process by which, each time the kernel is applied an extra layer of pixels is added to the outside of the existing threshold (Inoué and Spring, 1997). It also has a nice side effect of closing holes in the threshold. It is important to note that over-dilation should be avoided as it causes the holes of interest to close.

6. Inspect the dilation results, as shown in Figure 12.21.12. Dilation works well but the obvious side effect is apparent: the borders are too thick to be useful.

7. Use the Skeletonize Kernel Operation, as shown in Figure 12.21.13A. Another binary processing kernel, aptly named "skeletonize," can be used to reverse this effect. Skeletonize will reduce the binary to a 1-pixel width result (Inoué and Spring, 1997). Note that it is important to avoid choosing "erode" because generally this would destroy the progress we had made in reconnecting the broken segments achieved through the dilation in step 5b.

8. Inspect the skeletonized results, as shown in Figure 12.21.13A. Skeletonize works best on very smooth binaries. The borders (or spaces between objects) from steps 2 to 3 were very smooth so the result of skeletonize is sufficient. There are some noticeable extra segments present from places where the binary was not smooth (branch-like segments in Fig. 12.21.13A); this can usually be corrected. Alternatively, is also possible to smooth the dilated binary image generated from step 5b first to anticipate these segments.

9. Use the Prune Tool if needed, as shown in Figure 12.21.13B. To remove unwanted segments from a skeletonized image, as shown in Figure 12.21.13A, select the binary process named "pruning" (Russ, 1995). Just like pruning hedges, choose how many pixels to "prune" from the skeletonized image. It will only prune away end-pieces (the segments that are unwanted), and not contiguous structures (the borders or space between the objects). The pruning result image is a nicely segmented view of the borders around the muscle fibers.

10. Inspect the prune results, as shown in Figure 12.21.13B. After pruning, the binary processing is almost complete. It is important to note that sometimes, no matter the quality of the image or the processing, it is not possible to segment with 100% accuracy. It is apparent in the pruned result (Fig. 12.21.13B) that some borders are not drawn; this error should be noted. It should also be noticed that once a

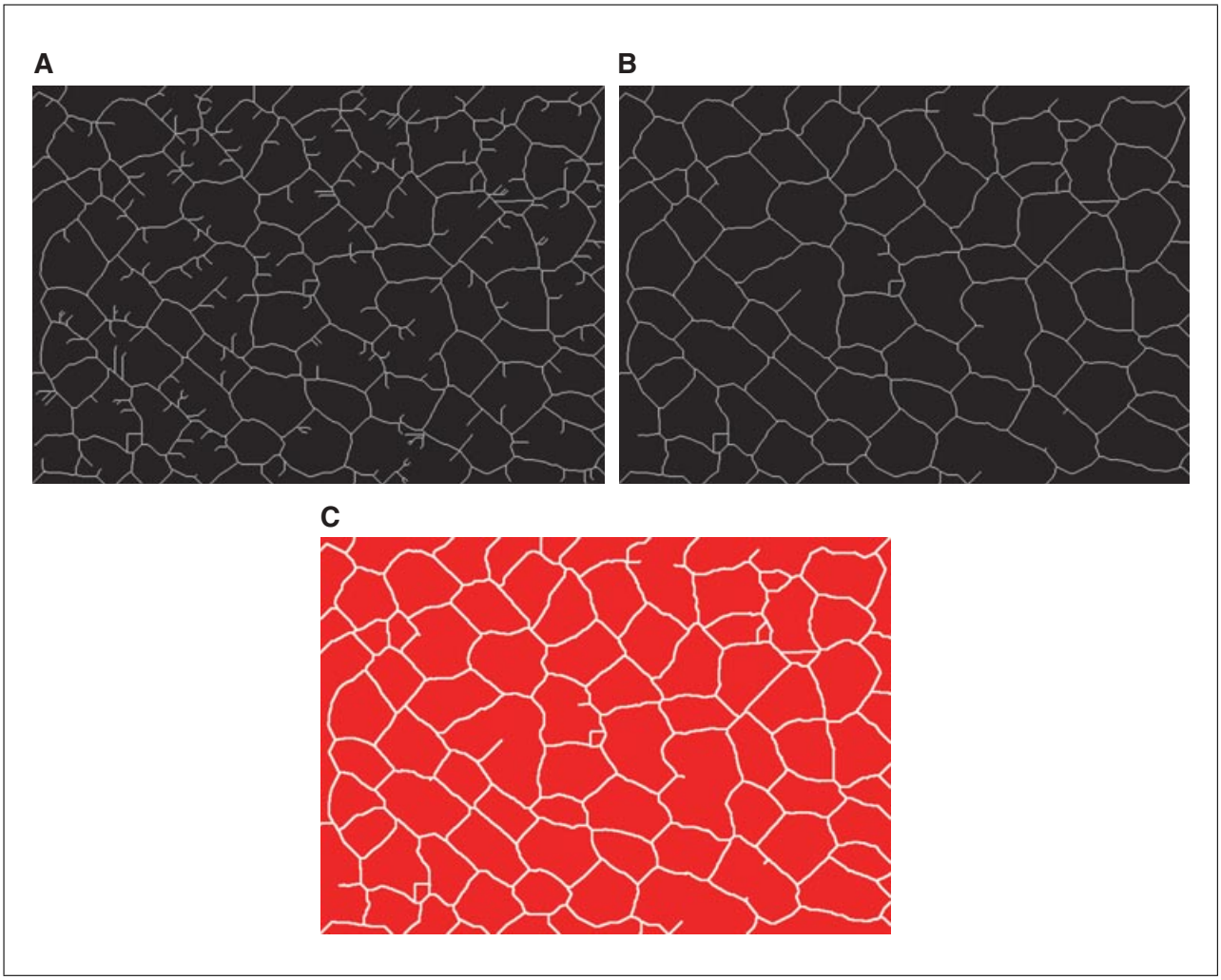


Figure 12.21.13 Resulting skeletonized binary image, where the skeletonize function traces the center line of the binary objects. (**A)** The result of "skeletonize" is close to the goal of defining the spaces between the objects. (**B**) Result of the prune tool on skeletonized image to remove unwanted segments. (**C**) A simple invert binary swaps the borders and muscle to identify areas of interest. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

technique has been established for image processing it can be subsequently applied to several images in a quick and efficient manner, and the user's ability to increase the sample size can tremendously improve the statistical significance of data analysis despite these occasional systematic errors.

11. Invert the binary image, as shown in Figure 12.21.13C. Typically, there is a command in the software's binary menu named "Invert" or "Invert Binary." At this step it is now possible to threshold the muscle rather than the border between the muscle.

12. Convert the binary image into Regions of Interest (ROIs), as shown in Figure 12.21.14A. The final step is to convert the binary layer (adequately separated now by white borders) into individual ROIs for each muscle fiber. ROIs are created by automatically tracing an outline around each binary section, which is outlined by the cell borders in Figure 12.21.13C. After the conversion, it is possible to use the ROIs to count each type of fiber, calculate areas, intensities, or a number of other morphological details, which were not possible to perform automatically with the original image. It is also possible to ignore any objects that touch the edge of the image as shown in Figure 12.21.14B (likely, these are incomplete muscle fibers not intended for counting).

13. Perform measurements based on the ROIs (Fig. 12.21.15). Most current software packages will display the results of the measurements of the multiple ROIs.

Case 2: Counting objects within objects

This section uses an example of an image with objects within objects. In addition, this example demonstrates the use of more than one binary layer. The goal of analysis for this

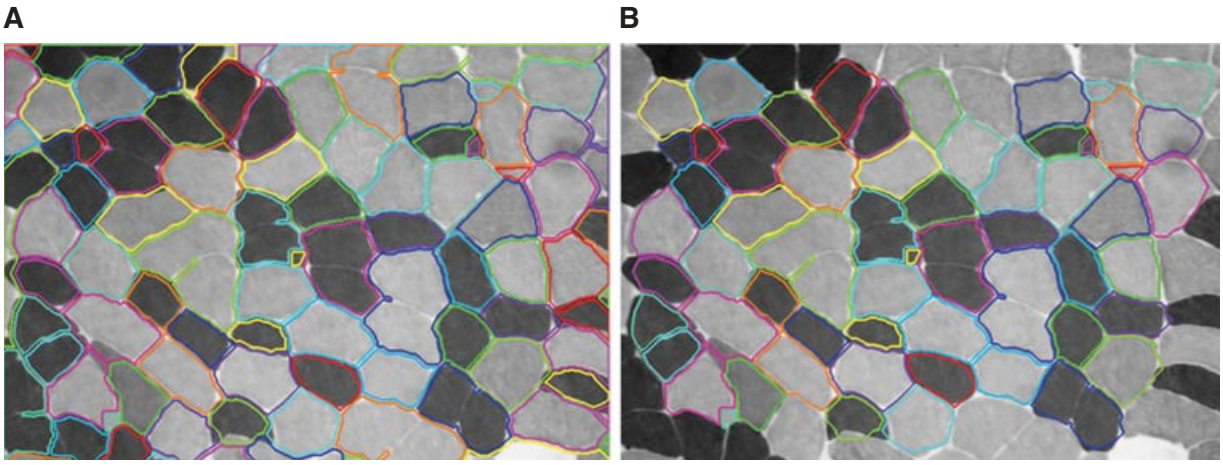
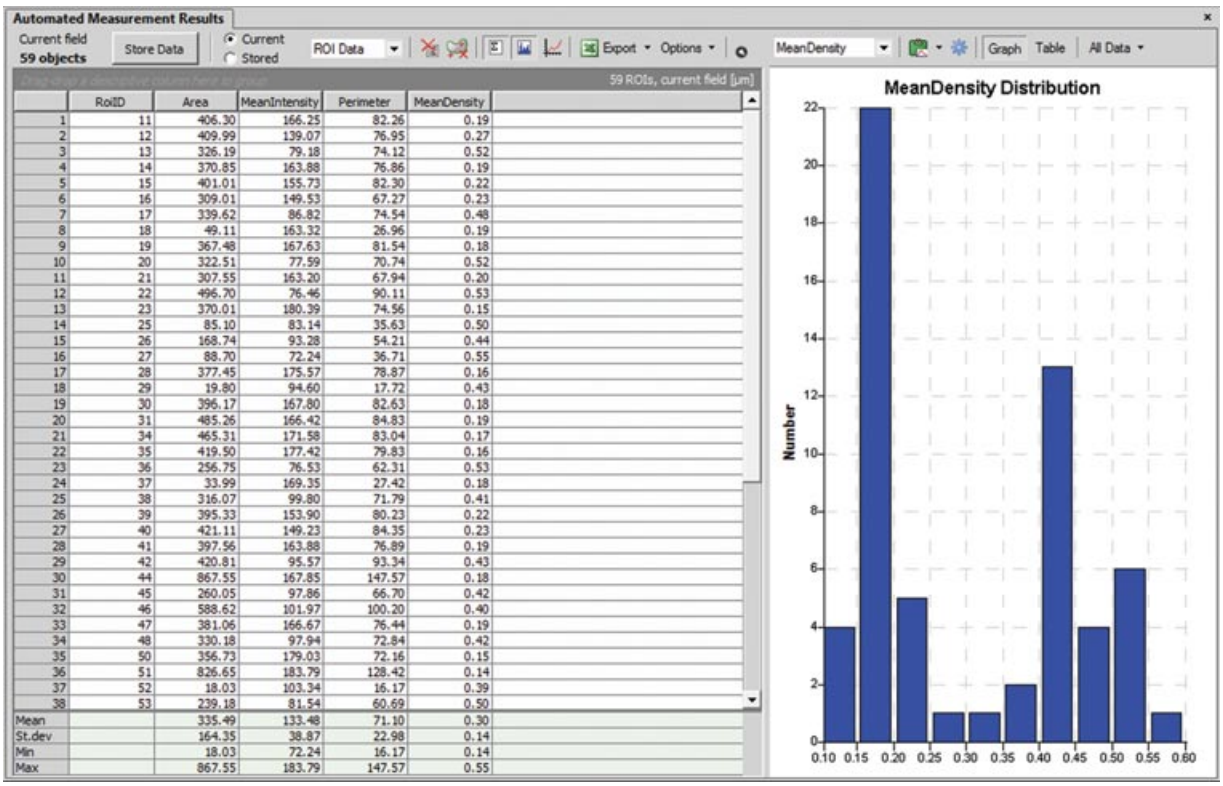


Figure 12.21.14 Creation of ROIs (regions of interest) around the borders of the binary objects. (**A**) Shows the image after converting the binary to ROIs, and overlaying it on the original image. (**B**) Optionally, exclude ROIs that are touching the border, which may be incomplete and not desired in the count or measurement. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.



	RoiID	Area	MeanIntensity	Perimeter	MeanDensity
1	11	406.30	166.25	82.26	0.19
2	12	409.99	139.07	76.95	0.27
3	13	326.19	79.18	74.12	0.52
4	14	370.85	163.88	76.86	0.19
5	15	401.01	155.73	82.30	0.22
6	16	309.01	149.53	67.27	0.23
7	17	339.62	86.82	74.54	0.48
8	18	49.11	163.32	26.96	0.19
9	19	367.48	167.63	81.54	0.18
10	20	322.51	77.59	70.74	0.52
11	21	307.55	163.20	67.94	0.20
12	22	496.70	76.46	90.11	0.53
13	23	370.01	180.39	74.56	0.15
14	25	85.10	83.14	35.63	0.50
15	26	168.74	93.28	54.21	0.44
16	27	88.70	72.24	36.71	0.55
17	28	377.45	175.57	78.87	0.16
18	29	19.80	94.60	17.72	0.43
19	30	396.17	167.80	82.63	0.18
20	31	485.26	166.42	84.83	0.19
21	34	465.31	171.58	83.04	0.17
22	35	419.50	177.42	79.83	0.16
23	36	256.75	76.53	62.31	0.53
24	37	33.99	169.35	27.42	0.18
25	38	316.07	99.80	71.79	0.41
26	39	395.33	153.90	80.23	0.22
27	40	421.11	149.23	84.35	0.23
28	41	397.56	163.88	76.89	0.19
29	42	420.81	95.57	93.34	0.43
30	44	867.55	167.85	147.57	0.18
31	45	260.05	97.86	66.70	0.42
32	46	588.62	101.97	100.20	0.40
33	47	381.06	166.67	76.44	0.19
34	48	330.18	97.94	72.84	0.42
35	50	356.73	179.03	72.16	0.15
36	51	826.65	183.79	128.42	0.14
37	52	18.03	103.34	16.17	0.39
38	53	239.18	81.54	60.69	0.50
Mean		335.49	133.48	71.10	0.30
St.dev		164.35	38.87	22.98	0.14
Min		18.03	72.24	16.17	0.14
Max		867.55	183.79	147.57	0.55

Figure 12.21.15 Example result data from ROIs (see Fig. 12.21.14) of the muscle image of Case 1, including measurement and statistical information on the number of fibers, area, intensity, and density.

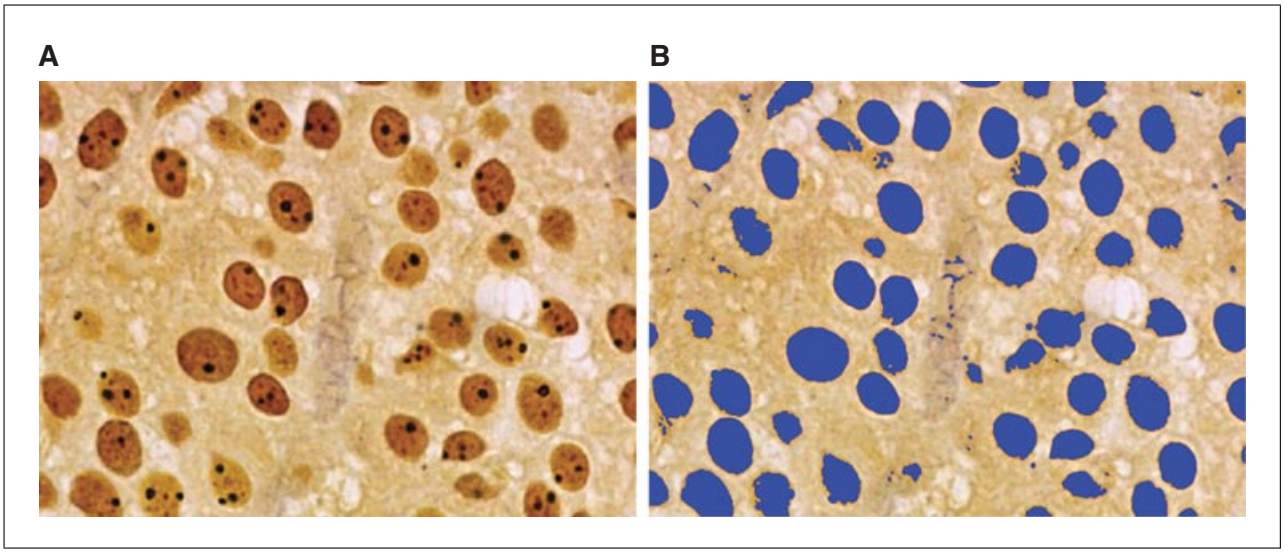


Figure 12.21.16 Creation of threshold of larger objects, which will be used to define the boundaries of the larger objects in subsequent steps in Case 2. Panel **A** shows the original image and panel **B** shows thresholding on larger objects. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

section is to generate the number of larger objects and the number of objects within the larger object.

1. Create threshold for the first binary layer of larger objects (see step 3 of Case 1), which encompass the smaller objects.

2. Inspect the threshold, as shown in Figure 12.21.16. The initial threshold may include unwanted objects such as debris or artifacts.

3. If needed, modify the threshold through filters such as "clean" or a manual removal of objects. In this example, a "clean" binary operation is used to remove small unwanted objects in the middle of Figure 12.21.16 that are not of interest (Fig. 12.21.17A). Some images may also have touching objects that can be separated via a watershed operation (sometimes referred to as a "morpho-separate" operation).

4. As an alternative to step 4, use the resulting threshold from step 1 and perform a size and shape restriction to exclude objects that are below an area size or circularity shape factor (Fig. 12.21.17B).

5. The purpose of this step is to create a mask that will be used later in the Case to clearly define each larger object's boundaries/borders. Convert this first binary layer to ROIs (Fig. 12.21.18A). If an option, create ROIs from this binary layer (see step 11 from Case 1).

6. Create a new threshold on the smaller objects found within the larger objects thresholded in step 1 (Fig. 12.21.18B).

NOTE: *The previous threshold was converted to ROIs in step 5. In the process of the conversion, the first binary layer was intentionally removed as well. The purpose of this step is to create a mask that will be used later in the Case to clearly define the objects intended for counting within each larger object's boundaries, which are now ROIs.*

7. Enable the ROIs that were created in step 5 to view the boundaries of each large object and the thresholded objects within the boundaries from step 6. This will display two categories of objects: the larger objects defined by the ROI and the smaller objects defined by the threshold performed in step 6 (Fig. 12.21.18C).

8. Automatically count the number of thresholded objects. Because ROIs were created in step 5, the measurement results shown in Figure 12.21.19 reflect and are organized into the number of objects within each ROI, rather than the total number of thresholded objects in the image field. In addition, ROI statistical data such as intensity and area are available as well.

Case 3: Object identification with uneven illumination/varying brightness

A common barrier to successful thresholding and segmentation is uneven illumination or varying brightness due to light source, alignment, and probe concentration (Fig. 12.21.20). The following example details a simple strategy to help address this issue.

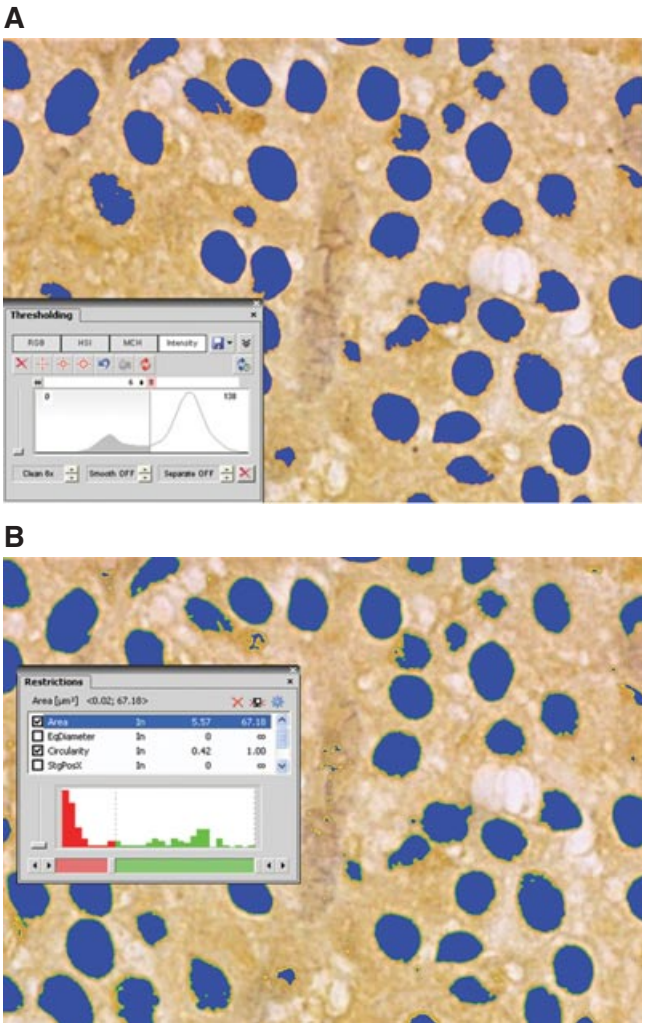


Figure 12.21.17 Modification of the binary layer to remove unwanted objects using filtering and restrictions. (**A**) Resulting binary layer after "Clean" operation. (**B**) Resulting binary layer using area and/ or shape restrictions to exclude unwanted objects. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

1. Apply a "Regional Maxima" kernel to the image, as shown in Figure 12.21.21. "Regional Maxima" is sometimes referred to as "Rank Leveling" (Russ, 1995). In general, this function uses neighborhood-based processing to find the brightest pixels in discrete areas and separate these values from the dark pixels. Applying this function area by area results in intensity equalization of the "brightest" parts of all areas of the image and suppression of dark areas. It is important to note that this process modifies the gray-scale values, which will affect intensity values. This step is used for creating the mask, which is the thresholding defining each object. Make sure to save the original data before saving the modified image.

2. Review the results from the Regional Maxima operation, as shown in Figure 12.21.22A. The resulting image has even illumination across the image field and is ready for thresholding.

3. Generate a threshold of the image along with additional operations, as described in step 3 of Case 2, to clean and separate touching objects, as shown in Figure 12.21.22B.

4. The mask generated in step 3 can be used for image analysis such as ROI or field-based measurements.

CONCLUDING REMARKS

Computer-assisted image analysis is performed primarily based on the process of segmentation, where a user defines areas of interest by a threshold or region of interest using image-processing software. A variety of common image analysis solutions utilize

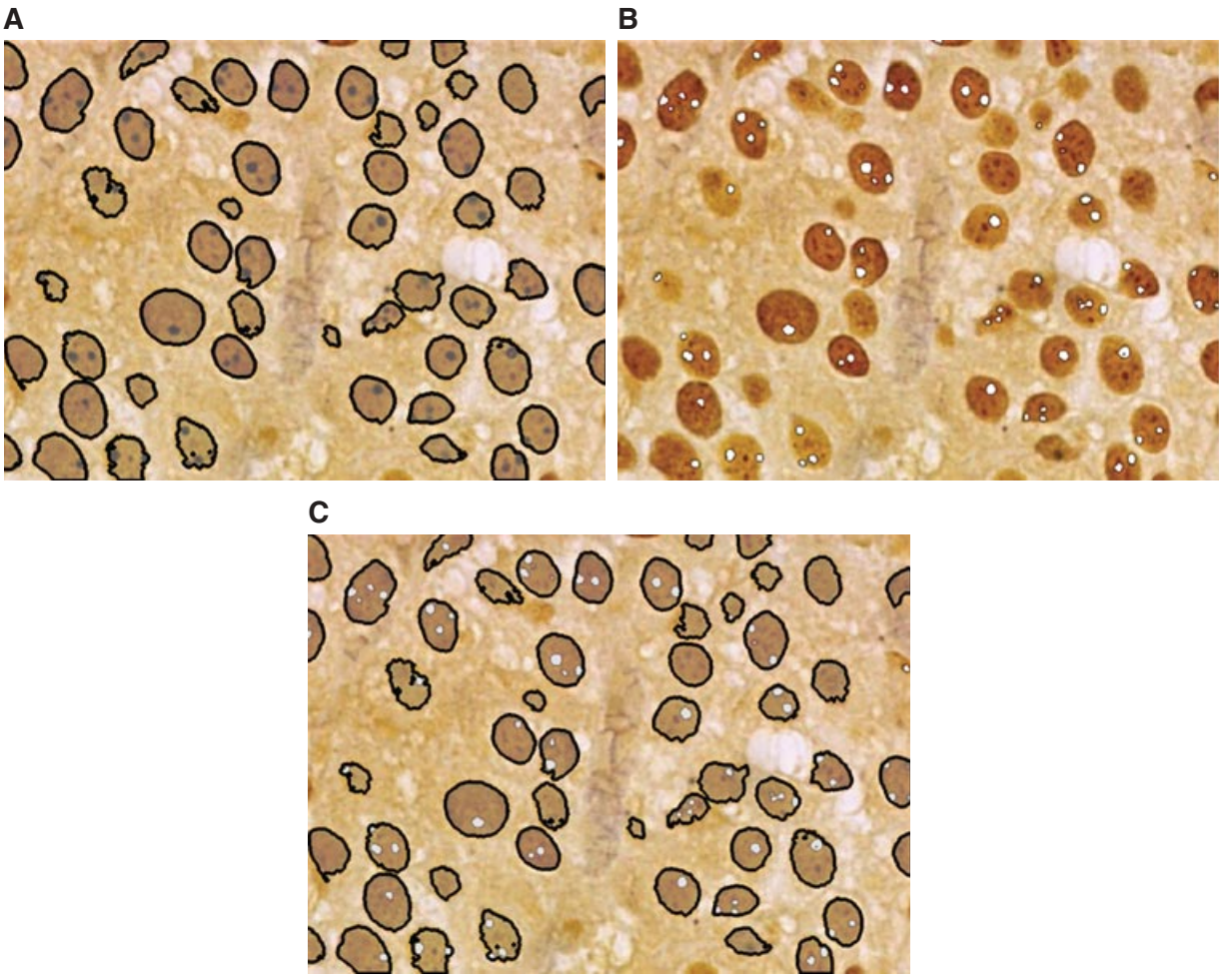


Figure 12.21.18 Creation of ROIs around the boundaries of the larger binary objects. (**A**) Creating ROIs around larger objects from the first binary layer. (**B**) Resulting binary layer after thresholding on the smaller objects within the larger objects' borders. (**C**) Resulting ROI around larger objects and binary layer defining the smaller objects. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

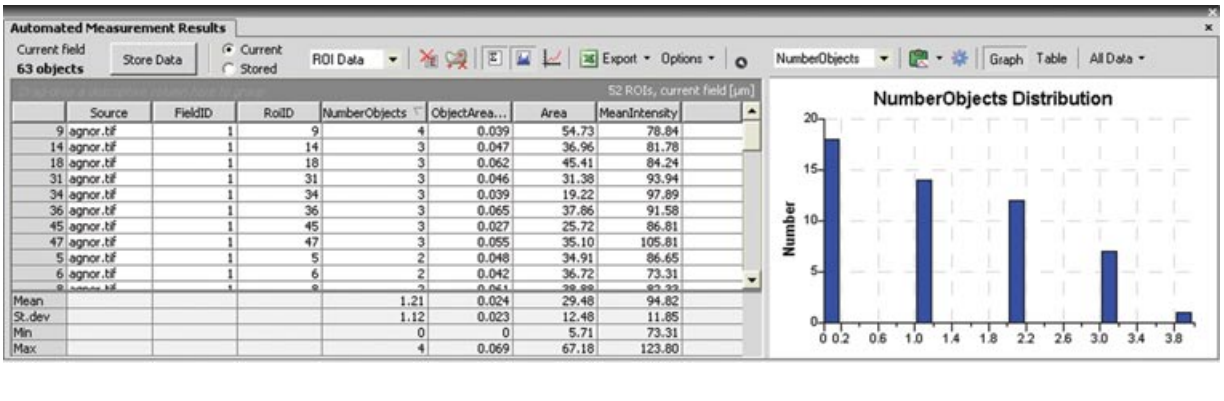


Figure 12.21.19 Measurement results and statistic summarizing both the number of objects per ROI and measurements for the thresholded objects.

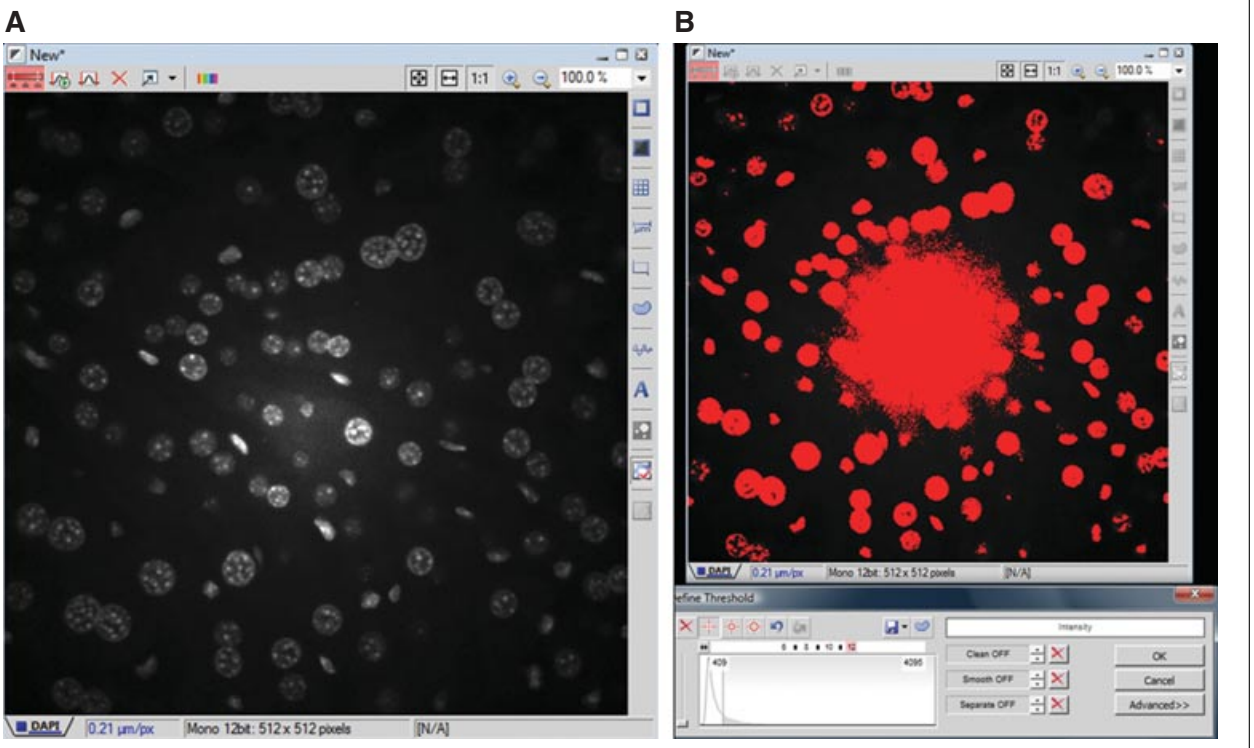


Figure 12.21.20 Unsuccessful thresholding on an image with uneven illumination. (**A**) Original image with uneven illumination (note that there is a brighter area in the middle of panel A.) (**B**) Thresholding cannot be successfully performed on this image due to the underlining uneven illumination shown in panel A. Note that the objects in the middle of the image are not detected as separate objects due to brighter illumination of the center of the image. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

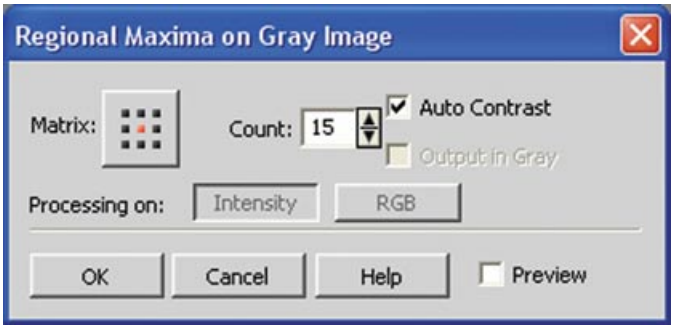


Figure 12.21.21 NIS-Elements Regional Maxima function to apply regional maximum and create intensity transformation to even the illumination.

preprocessing in order to improve contrast in raw images, segmentation to identify areas of interest in the image, and ROIs to allow for quantitative measurements. Using these core tools of an imaging software package allows users to convert raw pixel information to useful statistical data. More complex software operations are also available for multi-dimensional data such as time, Z, and multi-channel to address objects that change shape and intensity across the dataset. However, the basis of object detection, simple or complex, has its roots in improving contrast at the pixel and neighborhood level.

Disclaimer

Laura R. Sysko and Michael A. Davis are employed by Nikon Instruments, the

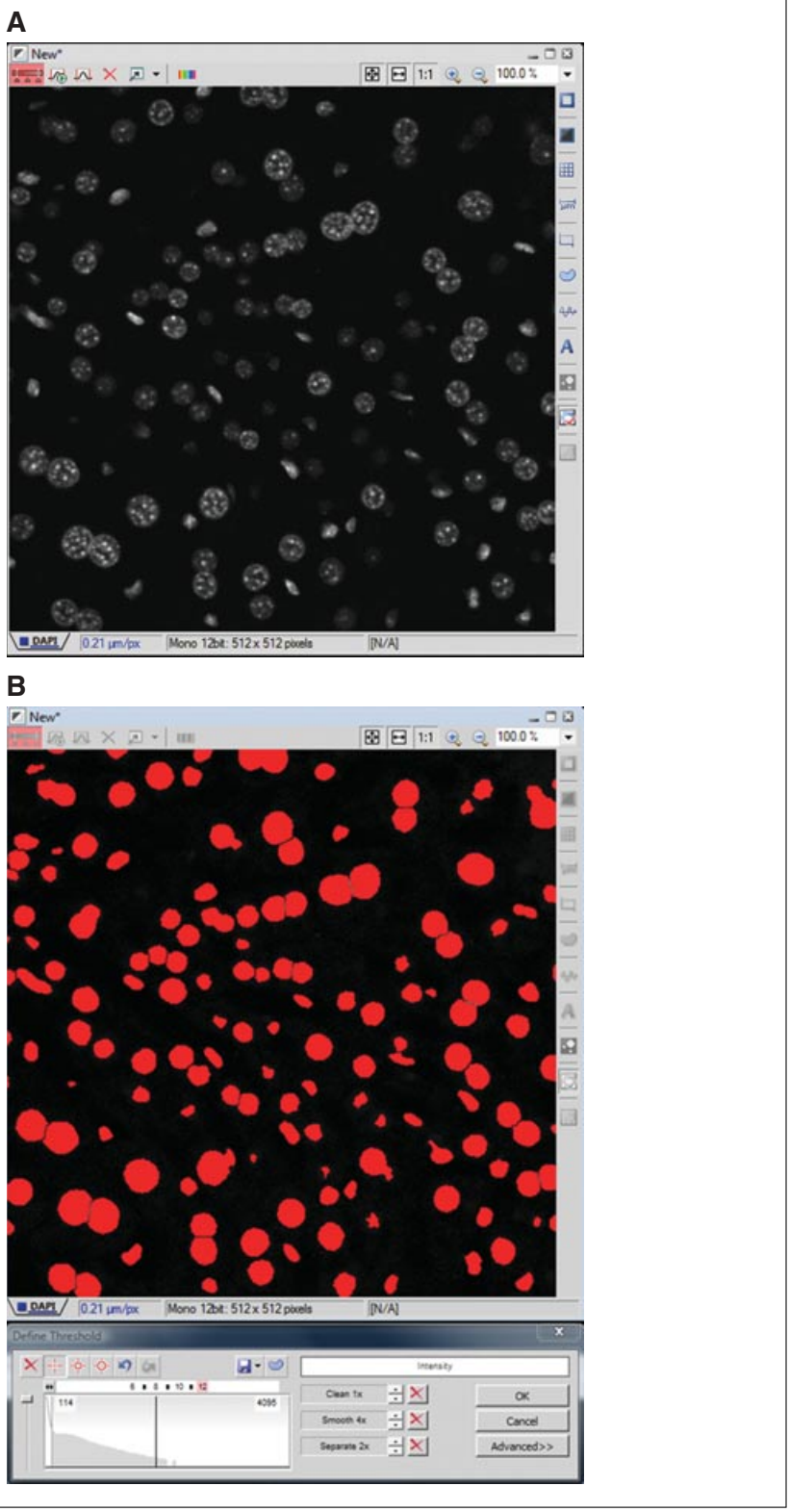


Figure 12.21.22 Application of Regional Maxima and subsequent steps for successful thresholding of objects across the image field. (**A**) Resulting Image after Regional Maxima operation. (**B**) This panel shows thresholding on the image with Regional Maxima, along with smoothing edges, cleaning to remove single pixels, and watershed segmentation/separation to split objects, which results in distinct binary objects across the image. For the color version of this figure go to *http://www.currentprotocols.com/protocol/cy1221*.

manufacturer of NIS-Elements, the software that was used for the step-by-step instructions included in the three cases in this manuscript.

LITERATURE CITED

Cox, P.W., Paul, G.C., and Thomas, C.R. 1998. Image analysis of the morphology of filamentous micro-organisms. *Microbiology* 144:817-827.

Hinchcliffe, E.H. 2003. The use and manipulation of digital image files in light microscopy. *In* Methods in Cell Biology, Vol. 72. Digital Microscopy: A 2nd ed. of Video Microscopy (G. Sluder and D.E. Wolf, eds.) pp. 271-288. Elsevier Academic Press, San Diego.

Inoué, T. and Gliksman, N. 2003. Optimizing microscopy and analysis. *In* Methods in Cell Biology, Vol. 72. Digital Microscopy: A 2nd ed. of Video Microscopy (G. Sluder and D.E. Wolf, eds.) pp. 243-270. Elsevier Academic Press, San Diego.

Inoué, S. and Spring, K.R. 1997. Digital image processing. *In* Video Microscopy: The Fundamentals, 2nd ed. pp. 509-558. Plenum Press, New York.

Pawley, J.B. 2006a. Points, pixels, and gray levels: Digitizing image data. *In* Handbook of Biological Confocal Microscopy (J.B. Pawley, ed.) pp. 59-79. Springer-Science+Business Media, New York.

Pawley, J.B. 2006b. More than you ever really wanted to know about charged-coupled devices. *In* Handbook of Biological Confocal Microscopy (J.B. Pawley, ed.) pp. 918-931. Springer-Science+Business Media, New York.

Russ, J.C. 1995. The Image Processing Handbook, 2nd ed. CRC Press, Boca Raton, Fla.

Waters, J.C. 2009. Accuracy and precision in quantitative fluorescence microscopy. *J. Cell Biol.* 185:1135-1148.

Zwier, J.M., Van Rooij, G.J., Hofstraat, J.W., and Brajenhoff, G.J. 2004. Image calibration in fluorescence microscopy. *J. Microsc.* 216:15-24.