FEATURE EXTRACTION FROM IMAGES

A presentation by **Hans Emmanuel Hernandez** (2020-11387)

OBJECTIVES

In this activity, we use the image processing software ImageJ to analyze various images. We have the following objectives:

- Extract various features from selected images
- Separate overlapping elements by using watershed segmentation
- Count the number of elements and determine the area of each in a given image
- Perform selective feature extraction
- Perform logical operations in ImageJ

01 RESULTS AND ANALYSIS

PRELIMINARY CELL PROFILING

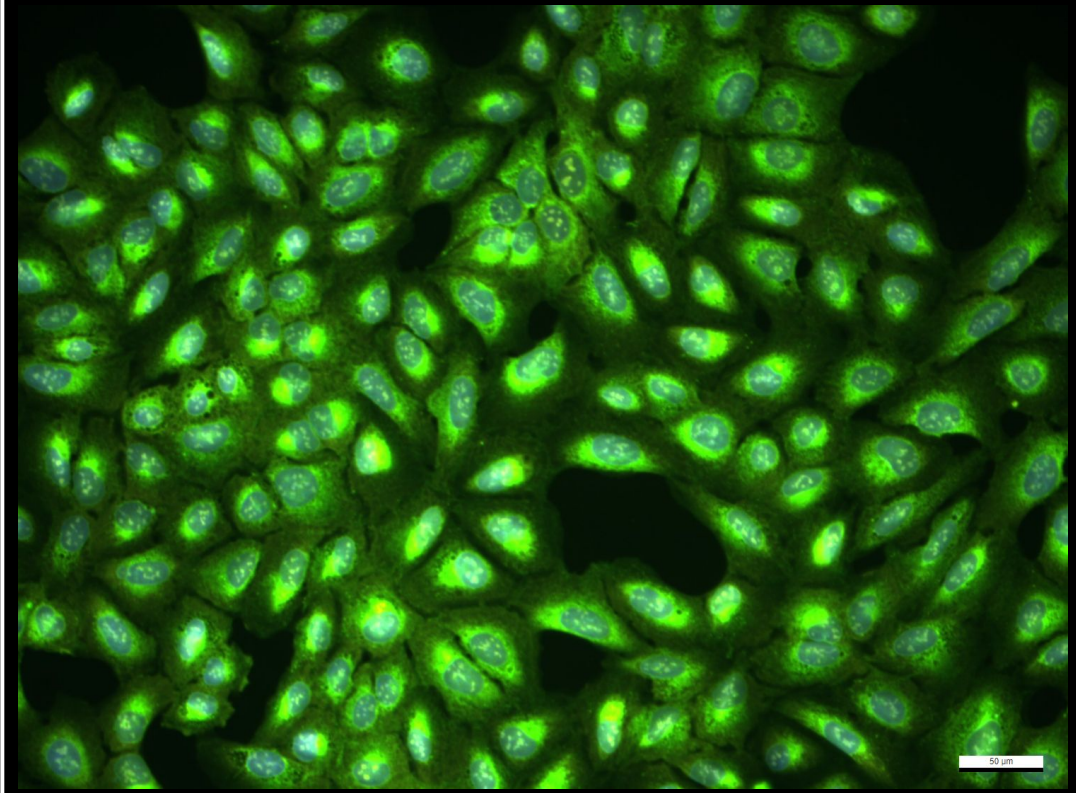

Figure 1. Fluorescent stained U2-OS cells from the Open Microscopy Environment under the Image Data Resource website. Scale: 50 μm per 116 pixels.

We select a sample image from the **Open Microscopy Environment under the Image Data Resource website.** The screening description states that these cells originate from the U2-OS cell line which derives from a moderately differentiated sarcoma of the tibia of a 15-year-old patient with osteosarcoma (bone cancer) in 1964. To morphologically profile the cells, **a cell painting assay was performed using green multiplexed fluorescent dyes to highlight features such as the nuclei, cytoplasm, and mitochondria.** As we can see in the figure, the cells are suspended in a viscous material which appear to be stained in green dye as well. Much of the background is black while the foreground consisting of the elements falls within a range of greens. **Upon preliminary inspection, we can observe that there are clusters of cells that could appear to be somewhat difficult in differentiating due to significant clumping.**

PREPARING THE IMAGE

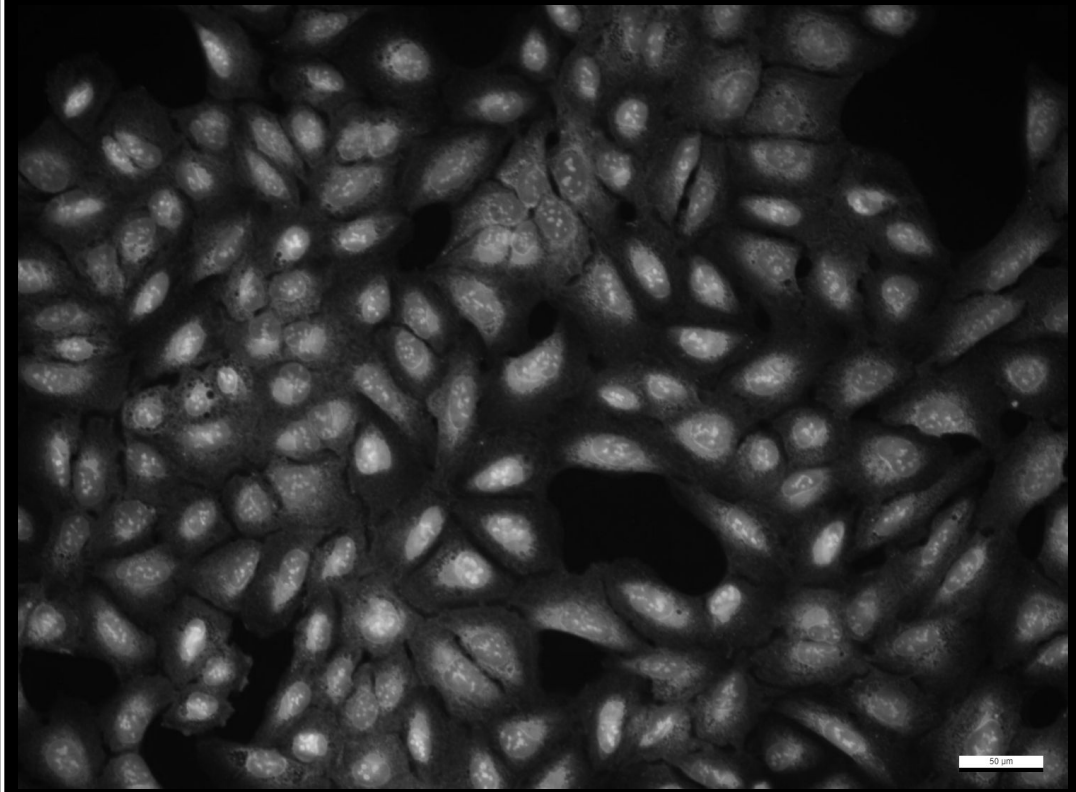

Figure 2. 8-bit image with global scale of 2.32 pixels/micron.

Before we can analyze the image, **we first convert it to grayscale by specifying the image type to 8-bit.** This ensures that we can perform thresholding for image segmentation. We then set the scale under the Analyze tab for more accurate calculations of the cell area. We use the line tool to set the pixel distance for the specified 50 μm scale. **This gives a global scale of 2.32 pixels/micron.**

CLEANING THE IMAGE

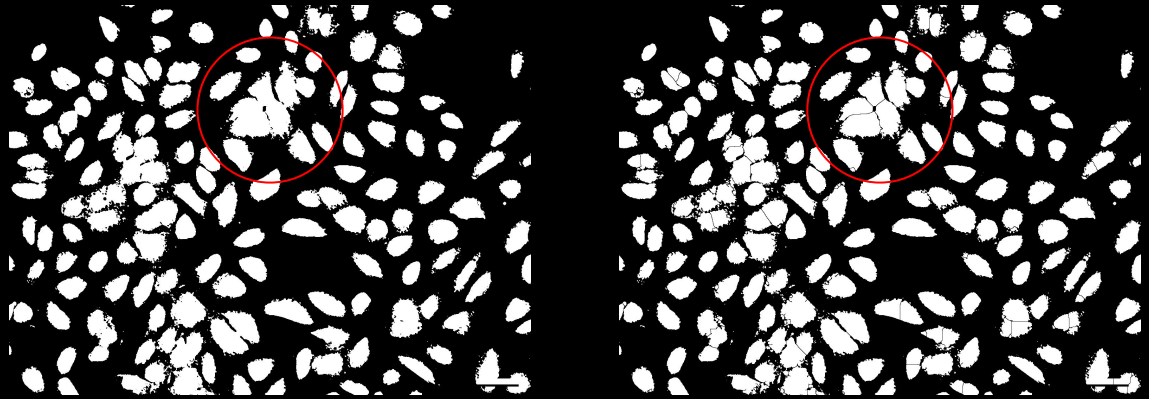

Figure 3. Before and after watershed segmentation of the thresholded image.

We clean the image by thresholding which adjusts the bounds of its grayscale histogram and then by binarizing to further eliminate noise. This ensures that there is clear separation between the elements and the background for accurate analysis. The left image clearly shows imperfect thresholding since the background and foreground of some cells had no significant distinction, however, we can try a workaround to this later. Since we are also dealing with overlapping cells, **we apply watershed segmentation to cut apart connected components as seen in the right image.** This ensures a more accurate cell count for later.

COUNTING THE CELLS

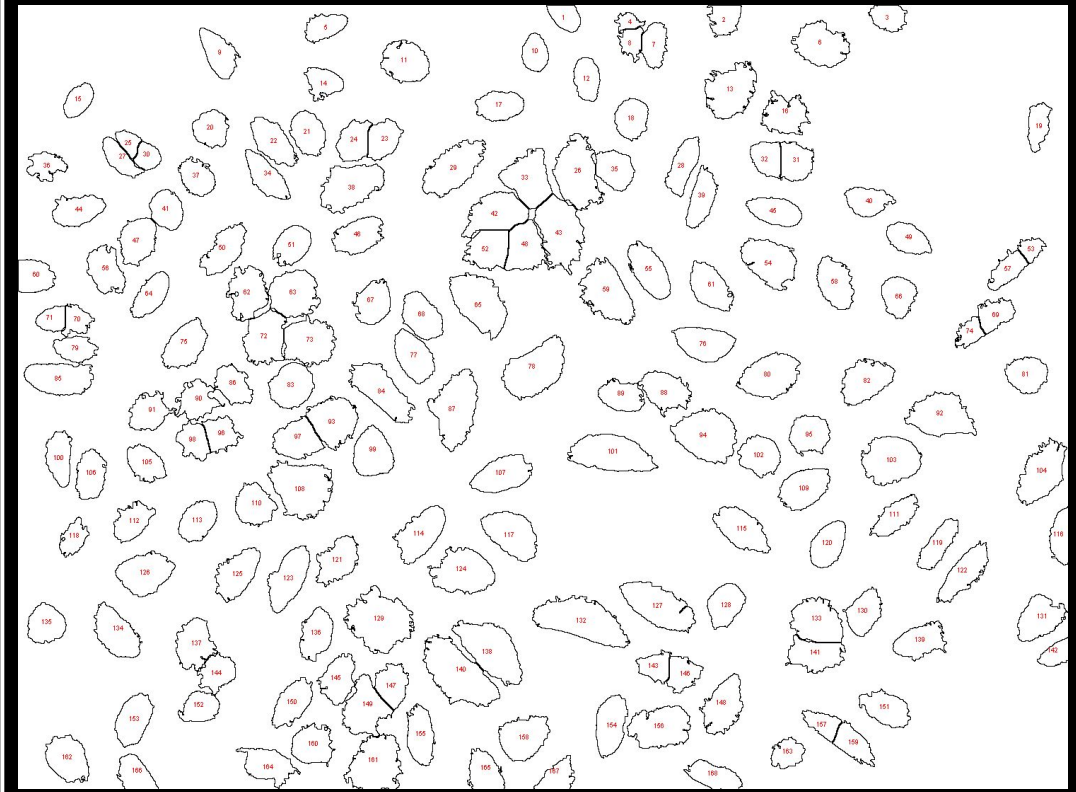

Figure 4. Resulting image with approximated cell count.

We perform particle analysis and set an appropriate range of pixel size to eliminate very small artefacts in the image. We retain the lower bound of our circularity as 0.0 since we are dealing with very irregularly-shaped cells. **In Fig. 4, we can see that we have successfully counted the cells, albeit only approximately, and calculated the area of each in Table 1 under the appendix.** Since the original image contained a bar scale on the lower right, **we invoke another function of ImageJ called the freehand selection tool to select appropriate areas for analysis.** We choose all areas except that region since we are only trying to show the capabilities of ImageJ, however, this is obviously not accepted in real scientific studies.

02 EXTRA ACTIVITIES

COUNTING SAND ELEMENTS

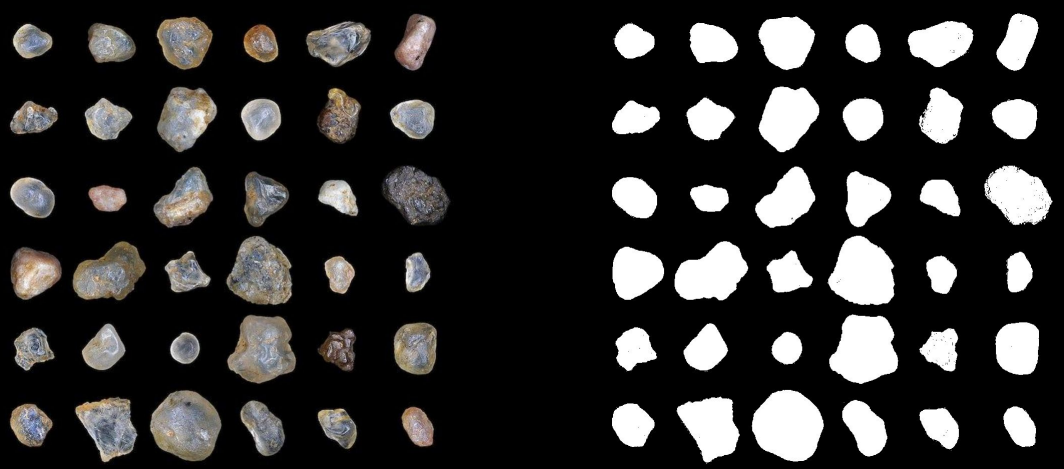

Figure 5. Particle counting using the previous algorithm. From left to right: (1) original sand image, (2) thresholded and binarized 8-bit image, (3) particle count of the image.

Using the previous algorithm with cell counting, we now count and calculate the area of sand elements. As we can see in Fig. 5, **we are now dealing with solid elements that are significantly distinct from their background unlike in our previous cell image.** Applying the same methods without using watershed segmentation, since the elements are vastly separate, **we see in the right image that we have successfully counted each sand element from the original image.** We also present additional features such as area, centroid location, and more in Table 2 under the Appendix.

SHAPE-BASED OBJECT EXTRACTION

Figure 6. Feature extraction on human retina. Right image shows the extracted blood vessels with apparent white spots all over.

In this part, **we implement shape-based object extraction on an image of a human retina.** As we can see in Fig. 6, the left image shows a highly vascular retina sample with bright, white objects called cotton wool spots. These spots are primarily caused by retinal arteriole obstruction (blood blockage to the eye) which is known to arise from several factors like diabetes and hypertension. **To extract these objects, we introduce another feature of ImageJ called the brightness/contrast under the Adjust tab.** By manually setting the brightness and contrast of the image, **we retain the brightest features such as the blood vessels and the spots.**

SHAPE-BASED OBJECT EXTRACTION

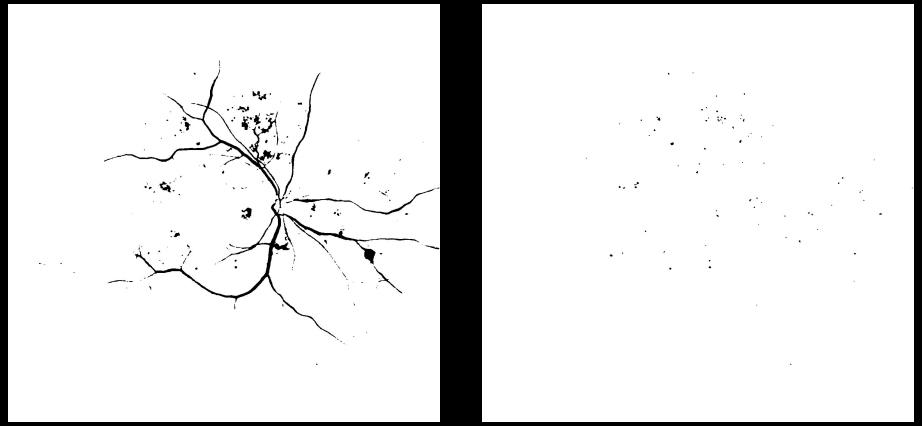

Figure 7. Before and after selective object extraction of the thresholded image. Right image shows the extracted spots.

Similar to the previous algorithms, we convert the image to 8-bit then threshold by manipulating the lower and upper bounds of its histogram to account for the objects that we want to extract. After this, **we binarize then invert the image.** This now serves as our mask for the upcoming image manipulation. **We apply particle analysis** and choose the following parameters: **1) Lower bound of 4 pixel units for the size since we do not want extremely small artefacts** and **2) Lower bound of 0.70 for circularity since we do not want to extract the blood vessels which probably have a circularity closer to 0.0 since they are more linear.** The right image in Fig. 7 shows the extracted cotton wool spots of the retina image.

LOGICAL OPERATORS IN IMAGEJ

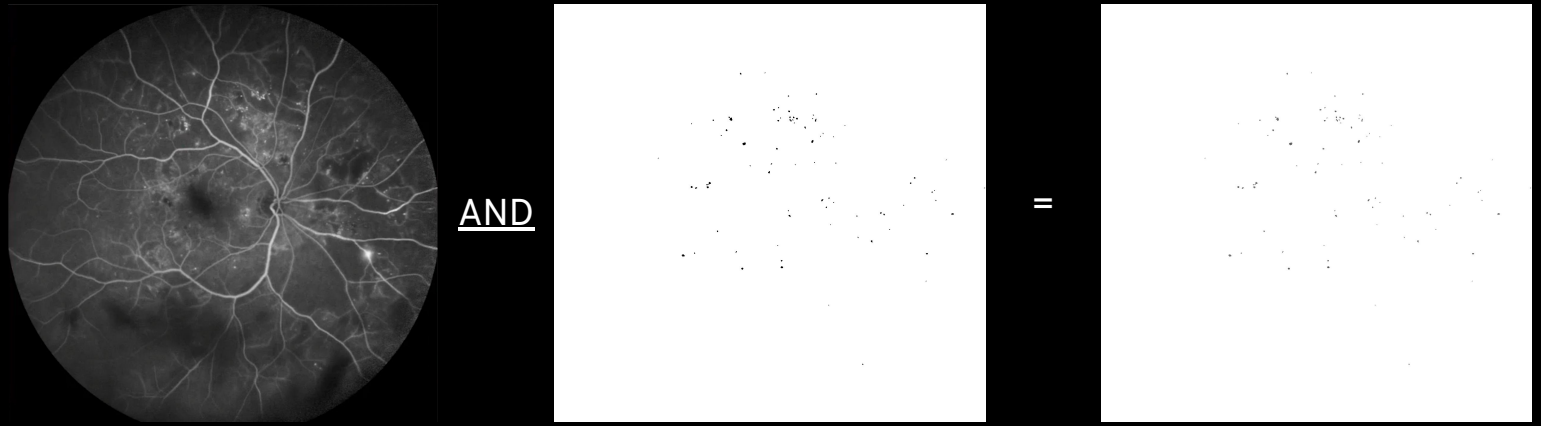

Figure 8. Original image of the retina is morphed with the binarized mask using AND operator. Resulting image on the right consists of spots with intensity.

Finally, we introduce the last feature of ImageJ called the Image Calculator. This function performs **arithmetic and logical operations between two selected images such as Add, Subtract, Multiply, AND, OR, etc.** Since we already have a binarized mask from the previous slide, **we use the AND operator to morph it with the original retina image.** As we can see in the rightmost of Fig. 8, **the resulting image is simply the intersection between the original image and the mask. This creates the same spots of interest but now consisting of intensity.** We analyze this further in the next slide.

INTENSITY IMAGE OF EXTRACTED FEATURES

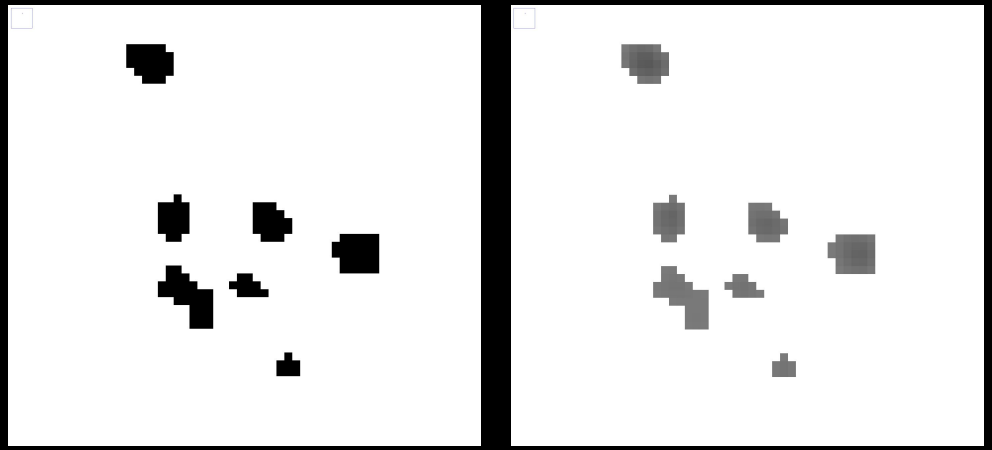

Figure 9. Comparing the extracted spots of the binarized mask and the resulting morphed image. Right image shows intensity.

We have successfully and selectively extracted the cotton wool spots from the original retina image. As we can see in Fig. 9, **we have retained the intensity of these spots by performing logical operation using ImageJ's Image Calculator.** Again, we take note that this is not a perfect implementation of selective feature extraction because certain parameters were arbitrarily selected by human choice. However, **this technique is especially useful in evaluating biological masses such as irregular growths in the human flesh since we still get a preliminary idea of their shapes, structures, and sizes.**

REFLECTION

Overall, this was a great activity!

ImageJ is such a powerful tool despite feeling so compact and looking so dull with its UI. It really delivers very useful image processing techniques. I can even see myself using this software instead of opening Jupyter and setting up my environments to process simple tasks especially when I am on the go. I have learned very relevant techniques that I would say are useful in fields like biomedical imaging and general feature extraction. I especially enjoyed discovering the Image Calculator and messing with all of the operations.

I believe that my results are accurate and stayed true to the topic and my objectives. After cross-validation with resources on the web, my instructors, and my peers, I believe that my presentation more than suffices the requirement for this activity. However, if I had more time to allot, I would like to use the counterpart of ImageJ in analyzing astronomical data called AstroImageJ. I would like to visualize the data I am working with on my thesis but I have no idea yet on how to visualize a FITS file using imaging softwares like these.

On the issue of yet another late submission, I would like to argue that my ADHD and my mental health in general has been worsening. I have depleted my stimulant medication and I won't be getting any more until I go back home to the province in June. I do not know what else to say except I swear that I try so hard and I feel like I have to try even harder than my neurotypical peers. I feel so bad that I have to make excuses like this but what can I really do if this is my reality?

SELF-GRADE

Technical Correctness: 35/35

I believe that my results are correct through math, research, and through validation with my peers and with my instructors.

Quality of Presentation: 35/35

I believe that the quality of my powerpoint is up to par with the course expectations. I constructed the figures as instructed, and exported my data accordingly.

Self-Reflection: 30/30

I believe that I have acknowledged and reflected upon the activity well enough. I also have complete citation on the next slide.

Initiative: 10/10

I went above and beyond with my data presentation, and included extra analyses for the activities.

REFERENCES

Bray, M., Singh, S., Han, H., Davis, C. T., Phanse, S., Hartland, C. L., Kost-Alimova, M., Gustafsdottir, S. M., Gibson, C., & Carpenter, A. E. (2016). Cell Painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. *Nature Protocols*, *11*(9), 1757–1774. https://doi.org/10.1038/nprot.2016.105

Dahlin, J. L., Hua, B. K., Zucconi, B. E., Nelson, S. D., Singh, S., Carpenter, A. E., Shrimp, J. H., Lima-Fernandes, E., Wawer, M. J., Chung, L. P. W., Agrawal, A., O'Reilly, M., Barsyte-Lovejoy, D., Szewczyk, M., Li, F., Lak, P., Cuellar, M. E., Cole, P. A., Meier, J. L., . . . Wagner, B. K. (2023). Reference compounds for characterizing cellular injury in high-content cellular morphology assays. *Nature Communications*, *14*(1). https://doi.org/10.1038/s41467-023-36829-x

Ferreira, T. (n.d.). *ImageJ User Guide - IJ 1.46r | Process Menu*. https://imagej.nih.gov/ij/docs/guide/146-29.html

Measuring cell fluorescence using ImageJ — The Open Lab Book v1.0. (n.d.). https://theolb.readthedocs.io/en/latest/imaging/measuring-cell-fluorescence-using-imagej.html

APPENDIX

	Label	Area	X	Y	Circ.	AR	Round	Solidity
1	cells_wate	319.234	185.148	13.053	0.616	1.855	0.539	0.91
2	cells_wate	700.197	485.66	23.597	0.537	1.29	0.775	0.897
3	cells_wate	328.914	384.726	23.938	0.509	2.032	0.492	0.879
4	cells_wate	443.422	121.51	29.175	0.524	1.865	0.536	0.904
5	cells_wate	289.649	312.379	28.971	0.825	1.444	0.692	0.954
6	cells_wate	574.184	233.843	34.192	0.558	1.227	0.815	0.923
7	cells_wate	346.994	343.6	44.601	0.756	1.698	0.589	0.946
8	cells_wate	792.972	430.178	51.841	0.407	1.231	0.813	0.88
9	cells_wate	326.722	184.267	46.922	0.446	1.155	0.866	0.866
10	cells_wate	264.081	36.415	57.442	0.805	1.669	0.599	0.951
11	cells_wate	425.707	291.08	60.771	0.717	1.635	0.611	0.935
12	cells_wate	399.226	370.691	68.471	0.777	1.243	0.805	0.935
13	cells_wate	329.827	616.854	72.502	0.489	2.167	0.462	0.868
14	cells_wate	401.782	116.589	74.904	0.686	1.07	0.934	0.93
15	cells_wate	449.814	175.192	77.521	0.7	1.445	0.692	0.932
16	cells_wate	510.629	154.132	82.633	0.486	2.081	0.481	0.908
17	cells_wate	788.041	211.598	81.545	0.563	1.673	0.598	0.879
18	cells_wate	161.809	66.768	82.487	0.654	1.44	0.695	0.884
19	cells_wate	292.936	61.414	92.665	0.568	2.568	0.389	0.875
20	cells_wate	412.923	401.868	96.059	0.581	2.519	0.397	0.91
21	cells_wate	790.598	262.967	98.158	0.419	2.061	0.485	0.893
22	cells_wate	202.352	78.268	91.386	0.772	1.068	0.937	0.914
23	cells_wate	748.229	462.003	93.43	0.651	1.703	0.587	0.926
24	cells_wate	713.164	308.399	104.449	0.57	1.191	0.84	0.896
25	cells_wate	445.796	150.061	103.426	0.582	2.387	0.419	0.913

Table 1. First 25 counted cells using watershed segmentation.

	Label	Area	X	Y	Circ.	AR	Round	Solidity
1	Screensho	6153	842.926	73.548	0.725	1.872	0.534	0.944
2	Screensho	8933	380.858	72.949	0.81	1.047	0.955	0.968
3	Screensho	8370	689.665	75.5	0.711	1.586	0.631	0.951
4	Screensho	5985	232.012	77.511	0.799	1.331	0.751	0.962
5	Screensho	4050	75.349	73.115	0.753	1.188	0.842	0.958
6	Screensho	4093	534.086	77.337	0.835	1.199	0.834	0.966
7	Screensho	10835	383.215	224.136	0.804	1.296	0.772	0.963
8	Screensho	6608	690.829	223.346	0.614	1.37	0.73	0.915
9	Screensho	5429	226.055	228.997	0.764	1.144	0.874	0.939
10	Screensho	5280	534.058	226.686	0.802	1.052	0.951	0.968
11	Screensho	5514	838.953	226.303	0.816	1.175	0.851	0.966
12	Screensho	4535	78.134	227.332	0.742	1.569	0.637	0.935
13	Screensho	10757	843.827	380.134	0.487	1.399	0.715	0.92
14	Screensho	8852	381.341	389.745	0.74	1.581	0.633	0.942
15	Screensho	6769	537.662	385.662	0.752	1.174	0.852	0.941
16	Screensho	5528	78.022	384.306	0.757	1.35	0.741	0.966
17	Screensho	4042	689.27	383.373	0.736	1.529	0.654	0.944
18	Screensho	3217	227.192	385.836	0.822	1.453	0.688	0.959
19	Screensho	13555	533.581	536.136	0.759	1.132	0.883	0.957
20	Screensho	11751	231.197	537.05	0.731	1.57	0.637	0.925
21	Screensho	8087	79.502	534.36	0.841	1.071	0.933	0.974
22	Screensho	4849	384.524	537.219	0.663	1.116	0.896	0.885
23	Screensho	3192	844.551	534.52	0.728	1.592	0.628	0.949
24	Screensho	3357	689.993	537.743	0.713	1.346	0.743	0.934
25	Screensho	13947	537.387	688.503	0.763	1.121	0.892	0.937
26	Screensho	7775	843.13	685.255	0.861	1.202	0.832	0.98
27	Screensho	5391	220.09	687.47	0.783	1.257	0.795	0.964
28	Screensho	4720	76.709	686.54	0.699	1.087	0.92	0.899
29	Screensho	4353	690.786	687.866	0.657	1.127	0.887	0.884
30	Screensho	2961	382.255	686.171	0.872	1.084	0.922	0.963
31	Screensho	15288	384.186	840.824	0.869	1.066	0.938	0.979
32	Screensho	10647	230.198	841.576	0.63	1.291	0.774	0.916
33	Screensho	6468	538.131	844.497	0.756	1.783	0.561	0.944
34	Screensho	5365	72.633	838.753	0.831	1.15	0.87	0.963
35	Screensho	3816	847.449	842.688	0.767	1.606	0.623	0.961
36	Screensho	4394	688.411	842.462	0.751	1.548	0.646	0.95

Table 2. All 36 counted elements of the sand image after particle analysis.