Automated Image Processing Workflow for Morphological Analysis of Fluorescence Microscopy Cell Images

SVEN P. VOIGT,¹ K. RAVIKUMAR,² BIKRAMJIT BASU,^{2,3}
and SURYA R. KALIDINDI ^{1,4,5,6}

1.—School of Materials Science and Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0245, USA. 2.—Laboratory for Biomaterials, Materials Research Centre, Indian Institute of Science, Bangalore 560012, India. 3.—Centre for Bio Systems Science and Engineering, Indian Institute of Science, Bangalore 560012, India. 4.—George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0405, USA. 5.—School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA. 6.—e-mail: surya.kalidindi@me.gatech.edu

Computerized image analysis of biological cells and tissues is a necessary complement to high-throughput microscopy, allowing researchers to effectively analyze large volumes of cellular data. It has the potential to dramatically improve the throughput and accuracy of measurements and related downstream analyses that may be obtained from images. This study presents a novel workflow for automated analysis of fluorescence microscopy images, which benefits from running multiple segmentation workflows and combining them to produce the best final segmentation. It is tested using a dataset of 42 fluorescence microscopy cells, evaluated against a hand segmented dataset using the F_1 score, and critically compared to a single segmentation workflow, which served as a control. The accuracy and reliability of the novel workflow are demonstrated to be superior to the control workflow, which achieved F_1 scores of 0.845 and 0.608, respectively. The workflow and example code are available through an open-source software platform.

INTRODUCTION

The last few decades have witnessed significant efforts directed into research programs to develop new biomaterials and bioengineering approaches to enhance biocompatibility in terms of the ability of a material to support cell proliferation (growth) and differentiation (changes in cell functionality).¹ Biocompatibility of a material is experimentally assessed using standard cell biology protocols, which are quite expensive. Fluorescence microscopy is frequently used in conjunction with biological tests to provide reliable quantitative cellular information.² Such information can then be used to predict the cell response and thereby reduce the number of expensive, ethically rigorous pre-clinical biocompatibility testing. Various researchers are interested in quantitative information in images such as cell area

coverage,³ cell count,^{4–6} and cell morphology.^{7–12} The generation of many cell features can additionally be used for informatics-style analysis to develop further correlations between cell shape and function.^{13–15} For example, muscle progenitor cells, when grown on a suitable cytocompatible material, show a gradual increase in aspect ratio and exhibit similar orientations to adjacent muscle cells.^{16,17} However, despite being highly beneficial, most of the current digital image analysis methods are customized to specific applications¹⁵ and/or not available to the broader research community in open-source repositories.

For the analyses of cell images, a primary challenge is that of distinctly identifying all of the living cells in an image,¹⁸ referred to as cell segmentation. Some of the primitives, building blocks or algorithms used in segmentation workflows that have been useful in cell segmentation include local thresholding,¹⁰ watershed,¹⁰ Voronoi evolution,⁸

level sets,^{19,20} morphological snakes,²¹ wavelets,⁹ graph cuts,²² contour edge detection,²³ peak detection,⁶ and neural networks.²⁴

However, despite the wide availability of primitives in applications such as ImageJ and Fiji,^{25,26} the design of segmentation workflows is often hampered by the lack of expertise and/or the demands placed on the development of highly customized workflows sometimes even for separate images in the same dataset.^{4,5} Therefore, some of the most commonly employed segmentation methods are the ones that are simpler to understand for non-specialists.¹⁵ This includes workflows incorporating the very popular local thresholding technique.¹⁰ However, the typical workflows incorporating local thresholding have several drawbacks that make them inaccurate, as described later and shown in Figs. 2 and 6.

This study develops a new image analysis workflow, incorporating local thresholds, that addresses the previous deficiencies and improves the segmentation accuracy. The segmentation task can be viewed as a classification task where each pixel is classified as either foreground or background. In the proposed workflow, multiple segmentations, using a single local thresholding operation, are combined using a majority vote to decide if each pixel should be foreground or background.

The developed workflow performs well in segmentation of multiple organelles, including the cell nuclei and cytoskeleton, and different cell types, including mouse muscle cells, human bone cells, and human mesenchymal stem cells. A traditional workflow using one local thresholding operation is critically compared to the proposed automated workflow using the F_1 score. The critical comparison shows an improvement from an F_1 score of 0.608 to an F_1 score of 0.845 when using the new workflow. The new workflow developed in this work is shared via ImageMKS (Python Package and Computer Program),²⁷ which provides broad access for researchers. ImageMKS also provides additional segmentation tools and automatically generates morphology measurements per cell (e.g., cell morphology, cell count, cell spacing) to ease incorporation into analysis and research.

METHODS

The workflow developed in this work is implemented in the python programming language using the open-source NumPy, SciPy²⁸ and scikit-image packages.²⁹ It first performs image segmentation to generate labels for each pixel in the image, where each label is uniquely associated with a cell in the image. The labeled images are then used to extract the statistics of various cell morphology measures of interest.

Segmentation

The segmentation algorithm developed in this work was designed to accept two coordinated monochrome images: a blue image showing the cell nuclei and a green image showing the cell cytoskeleton. The coordination between the two images dictates that each pixel in both images corresponds to the same physical location, implying that camera position and magnification with respect to the sample are kept constant. Figure 1 outlines the major steps in the proposed segmentation workflow, which are described next in more detail.

Image pre-processing in the proposed workflow starts with the conversion of both the nucleus and the cytoskeleton color images into grayscale images using a linear weighted approximation of the exact grayscale transformation, which is designed to produce approximately the same luminescence as observed in the color image.³⁰ The grayscale images are then smoothed using a Gaussian kernel; this process is usually referred to as a Gaussian blur operation.³¹ The pixel values of both images are nonlinearly scaled to the range [0,1] using the power law shown in Eq. (1), where (i,j) indexes all of the pixels in the image, I , the subscripts max and min indicate the maximum and minimum values, respectively, and p and P denote the transformed and original pixel values, respectively. This transformation is designed to amplify the contrast between pixel values close to one (i.e., the bright pixels in the image) by selecting a value of b larger than one. For instance, with $b = 2$, two pixels with normalized values of 0.90 and 0.80 would acquire transformed values of 0.81 and 0.64, respectively.

$$p(i,j) = \left(\frac{P(i,j) - P_{\min}}{P_{\max} - P_{\min}} \right)^b \quad (1)$$

After the common pre-processing steps described above, the feature detection is accomplished using multiple local thresholding algorithms.³² In the proposed approaches, thresholds are calculated for each pixel from a consideration of the neighborhood, $K_r(i,j)$, identified as the set of pixels within a scalar distance r to the pixel of interest (i,j) (see Eq. (2)). The mean pixel value (used as threshold) over this neighborhood is denoted as $t_r(i,j)$, where the subscript r reminds us of its dependence on the value of r .

$$K_r(i,j) = \{(u,v) \in I | \text{distance}((i,j), (u,v)) < r\} \quad (2)$$

The segmented image, S_r , are then identified as

$$S_r^{ij} = \begin{cases} 1, & p(i,j) > t_r(i,j) \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

Once again, the subscript r in S_r denotes the dependence of this variable on the selection of r . Choosing a small radius results in accurate

Fig. 1. The proposed segmentation workflow in this work incorporates pre-processing, multiple segmentations, and post-processing algorithms to transform the raw experimental image to a labeled image that distinctly identifies all of the constituents along with their associations (i.e., nuclei and their cytoskeleton).

Fig. 2.. Illustration of the labeling of the nuclei pixels and the background pixels using local threshold algorithms. These algorithms employ neighborhoods of different sizes in identifying the foreground pixels. **a** Original image with axes indicating pixel rows and columns. **b** Local thresholded image using a smaller neighborhood size. **c** Local thresholded image using a larger neighborhood size. **d** Merged foreground image. The nuclei detected by the watershed algorithm are marked with red markers. The images show that the use of a smaller neighborhood generates artifacts, while the use of a larger neighborhood misses some of the individual cells. Merging of the different local threshold images provides the best segmentation result.

detection of edges, but introduces artifacts in the foreground (see the top left of the local threshold 1 image in Fig. 2b). A larger radius results in less accurate detection of the edges, but does not produce artifacts (see the highlighted boxes in the local threshold 2 image in Fig. 2c and compare with the original image in Fig. 2a and the local threshold 1 image in Fig. 2b). Merging multiple local threshold

images was found to consistently provide the best overall segmented image (see Fig. 2d). Mathematically, the merging of local threshold images is expressed as

$$S_*^{ij} = \begin{cases} 1, & \left(\sum_{r \in R} S_r^{ij} \right) > \left(\frac{|R|}{2} \right) \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

where it is implied that pixels labeled in the merged image were labeled in more than half of the segmented images, where $|R|$ is the number of different r values in the set. For the present work, a set of two radius values, $R = \{r_1, r_2\}$, was chosen for the nuclei and the cytoskeleton, which was found to produce the best results, while not adding significantly to the computational cost. The length scales of nuclei and brightness fluctuations were used to select the values of r_1 and r_2 . The value of r_1 was chosen to correspond to the average diameter of the cell nucleus, while the value of r_2 was chosen to be one-fourth of the largest image dimension. The cytoskeleton was segmented using a single local threshold corresponding to r_2 , because these images do not have a clear feature-length scale, but exhibit similar brightness fluctuations in the image.

Post-processing is then applied to smoothen the images by considering the neighborhood of each pixel in the image. In this step, any background pixel whose neighborhood is dominated by foreground pixels (i.e., $> 50\%$ of the nearest neighbors in a pixelated circle of radius 6) is switched into a foreground pixel. Similarly, any foreground pixel whose neighborhood is dominated by background pixels is switched into a background pixel. The post-processing step described above has separated the foreground pixels from the background pixels, but did not distinctly label all of the individual cells or organelles (nuclei and cytoskeleton). This is accomplished through further post-processing steps that mainly employ binary watershed transform,³³ marker extraction, and flood-filling algorithms.^{30,33}

The marker extraction algorithm identifies the peaks in the distance transform of a binary image.^{30,33} The red dots in Fig. 2 show an example of the set of marker pixels, $m \in M$. The flood filling algorithm finds all foreground pixels which are closest to each marker m while being connected continuously (i.e., without being separated by background pixels or pixels which are associated with a different marker). The result of flood filling is a set of pixel regions l_m for each marker $m \in M$. The labeled images may be further post-processed as needed. For example, one might consider computing the number of pixels in each marker region l_m and adding it as an attribute to each marker. This will allow subsequent computation of relevant statistics (e.g., size distributions of cell nuclei) or further refinement of the images (e.g., removal of marker regions below a specified size as these are most likely artifacts from the image analyses). Figure 3 shows an example of the labeled nuclei and cytoskeleton images produced using the workflow described above. Note that the nuclei and their cytoskeleton are fully associated with each other in these images (reflected by the same color assignments in both images). Also, inverted versions of the raw images are shown to help the readers inspect the raw images better.

Image Measurements

Morphological measurements are readily extracted from a labeled cell image using established methods,³⁰ which are available in a variety of programming languages as the `regionprops` function.²⁹ Various exact attributes may be extracted for each labeled entity, including the area and perimeter. Other attributes such as major axis length, minor axis length, eccentricity, and orientation are calculated from an equivalent ellipse with the same image moments as the labeled object.³⁰

Automation

The workflow described above is provided as a sequence of python functions and classes in the ImageMKS library.²⁷ This library can be installed using the package manager pip with the command `pip install imagemks`. This installs both the python package and a command line program on the user's computer. The main segmentation function `segment_fluor_cells` accepts the images of the nuclei, cytoskeleton, and analysis parameters (discussed earlier) as inputs, performs the workflow outlined in Section 2.1, and returns the labeled images. The function `measure_fluor_cells` accepts the labeled images as inputs and returns a panda dataframe³⁴ containing the cell measurements. The command line program called `cellanalysis` calls the function `measure_fluor_cells` and saves the measurements to a csv file.

The subroutine `cellanalysis getparams` saves the default segmentation parameters to a json file such that they can be modified using a text editor. The command `cellanalysis segment` produces segmented images, and the command `cellanalysis measure` generates the measurements csv file. The command line options are `-fn <directory>` (folder of nucleus images), `-fc <directory>` (folder of cytoskeleton images), `-sn <directory>` (folder for saving labeled nuclei), `-sc <directory>` (folder for saving labeled cytoskeleton), `-fp <path>` (path to parameters file), `-sp <path>` (path to save default parameters file), `-sm <path>` (path to save measurements file), and `-z <float>` (μm per pixel). Further documentation is provided in the ImageMKS documentation.²⁷

Validation Strategies

The segmentation results obtained from the proposed workflow need to be critically validated. For this purpose, it was decided to compare them against results obtained by two other approaches: (1) manual segmentation and (2) a typically employed workflow that utilized a single local Otsu threshold followed by a binary watershed.¹⁸ The comparisons were performed on randomly selected

Fig. 3.. Illustration of the segmentation workflow developed in this work on an example fluorescence cell image.

nuclei from the set of images utilized in this study. Each element of this random selection set was identified by first choosing a random image from the complete dataset of 42 images segmented using the workflow proposed in this article and then randomly selecting one of the cell nuclei present in the image. The same image was then segmented using the typical approach described above, and the same cell nucleus was identified from the segmented image for comparison against the result from the proposed workflow. As another comparison, the same cell nucleus was identified in the original raw image file and manually segmented. To avoid bias, only the cell labels and locations were used to identify the cells for both the manual segmentation as well as the typical segmentation, and the researcher was not allowed to see the segmentation result from the proposed workflow.

The manual segmentations were treated as ground truth, and F_1 scores³⁵ (also referred to as Dice coefficients) were obtained for both the typical segmentation as well as the proposed workflow. The F_1 score provides accuracy measures based on precision (P) and recall (R), which are defined in terms of the true positives (tp), false positives (fp), and false negatives (fn). True positives count the successes (i.e., pixels identified as belonging to the selected cell nucleus in the manual segmentation). On the other hand, false positives count wrongly labeled cell nucleus pixels and false negatives count the wrongly labeled pixels outside the cell nucleus (both measures are based on ground truth established in the manual segmentation of the same nucleus). The F_1 score is computed as

$$P = \frac{tp}{tp + fp}, R = \frac{tp}{tp + fn}$$

$$F_1 = \frac{2PR}{P + R} = \frac{2 * tp}{2 * tp + fp + fn} \quad (5)$$

It should be noted that the F_1 score³⁵ is in the range [0,1], where the value of one indicates a perfect segmentation and the zero value indicates the worst segmentation (Fig. 4).

RESULTS AND DISCUSSION

Accuracy of the Proposed Segmentation

The proposed segmentation workflow was applied to a dataset of 42 fluorescence microscopy images of biological cells. This dataset contains a diverse set of cells, with image variations arising from differences in cell phenotype (morphological features), pixel scale ($\mu\text{m}/\text{pixel}$), and cell spreading (area density). Figure 5 presents selected images representing the variety in the dataset, where the image segmentation is presented alongside the original raw image (presented here using inverted colors as they are brighter and clearer to inspect compared to the darker raw images). The workflow developed in this work is seen to perform well despite the wide variety of images included in this study. A close inspection reveals the association between the raw images and the segmented images. To inspect the images, a reader may want to select objects (e.g.,

```

File Edit View Search Terminal Help
(seg) sven:~/cell_examples$ cellanalysis getparams -sp params.json
Saved parameters to `params.json`.
(seg) sven:~/cell_examples$ cellanalysis measure -fn nucleus_imgs -fc cytoskeleton_imgs
-sm measurements.csv -fp params.json -z 0.48
Loading nuclei from `nucleus_imgs`.
Loading cytoskeletons from `cytoskeleton_imgs`.
Loading parameters from `params.json`.
Saving measurements to `measurements.csv`.

Analyzing Nucleus and Cytoskeleton with id: `1`.

Analyzing Nucleus and Cytoskeleton with id: `2`.

Analyzing Nucleus and Cytoskeleton with id: `3`.
Saved measurements to `measurements.csv`.
(seg) sven:~/cell_examples$ █

```

Fig. 4.. This output shows the two steps needed to run the segmentation framework from the command line to produce measurements. The program is installed as `cellanalysis` and has several subroutines including `getparams` and `measure`. The subroutine `getparams` is used to generate a parameter file. The subroutine `measure` runs the segmentation and measurement algorithms for a folder of nucleus and cytoskeleton images. After the subroutine completes, results are saved to a csv file.

Fig. 5.. A selection of images from the dataset that represent variety that may be found in the data. **a, b** Muscle progenitor cells at high magnification. **e, f** Densely packed mouse muscle progenitor cells at low magnification. **i, j** Bone marrow derived human mesenchymal stem cells at high magnification. **c, d, g, h, k, l** Labeled organelles, where labels in a row match by association of the organelle with a cell. The labels are assigned colors to help the reader compare labels in the third column (nuclei) to the fourth column (cytoskeletons). The matching colors and spatial location of the organelles (in a row) indicate their association with the same cell.

nucleus or cytoskeleton) in the raw images and look for them in the segmented images. This inspection method can help make sure that all objects were identified, and it can also help qualitatively establish the accuracy of the segmentation.

Quantitative Approach for Segmentation Validation

A randomly chosen set of ten nuclei was segmented by hand and served as the ground truth data. Although ground truth implies a perfect

Fig. 6.. Comparison of segmentation methods. The same nucleus is segmented three different ways (manually segmented, traditional workflow, proposed workflow) in each row. The manual segmentation is assumed to be the ground truth and used to evaluate the effectiveness of the other methods using the F_1 score. The axes on the figures indicate rows and columns of pixels. The random selection is representative of the dataset, which contained a larger proportion of muscle cells than stem cells.

Table I. Comparison of F_1 scores for ten randomly selected nuclei.

Image	Nucleus	Cell type	Traditional F_1 score	Workflow F_1 score
10	257	Muscle cell	0.354	0.807
10	55	Muscle cell	0.000	0.672
16	101	Muscle cell	0.968	0.924
1	240	Muscle cell	0.000	0.803
26	55	Muscle cell	0.943	0.918
29	140	Muscle cell	0.855	0.897
30	34	Muscle cell	0.574	0.767
30	63	Muscle cell	0.903	0.830
37	17	Stem cell	0.618	0.972
9	101	Muscle cell	0.860	0.858
Averages			0.608	0.845

segmentation, a human segmentation inherently contains bias and errors. However, a definition for the perfect segmentation is not available in this case, and the human segmentation is the closest to ground truth that can be achieved. Figure 6 shows a subset of six nuclei along with the hand segmentations. The segmentations generated by the traditional Otsu workflow and the proposed workflow are compared pixel by pixel to the ground truth data and assessed using the F_1 score. The F_1 score achieved by each method along with the segmented image is also shown in Fig. 6. Inspection of Fig. 6 shows the traditional Otsu workflow successfully segments some nuclei, but fails for some other nuclei. In contrast, the proposed workflow consistently produces reasonably accurate segmentations of all nuclei.

Additionally, the results in the table show that the proposed workflow never performs much worse than the traditional workflow. It also shows that the proposed workflow scores much higher than the traditional workflow for other nuclei, demonstrating a consistent performance and improvement over the Otsu workflow (Table I).

Although the validation strategy chosen here captures some of the improvements of the proposed workflow, it does not capture all improvements. For example, one of the benefits demonstrated in Fig. 2 is the ability to remove artifacts while preserving precise segmentation of edges. The traditional workflow used here does not generate as many artifacts, but trades off precision in the segmentation of edges. In contrast, a traditional workflow incorporating a smaller neighborhood without the multiple segmentations used in the novel workflow would have generated several artifacts in the segmented image. This type of trade-off is eliminated using the multiple segmentations.

Cell Morphology and Probability Distributions

The ImageMKS workflow also automatically extracts statistics from the segmented images. The ability to automate the extraction of the statistics allows increased sampling or acquisition of data points from the image data. This constitutes an important advantage, especially when the determination of cell shape/morphology descriptors from large datasets becomes challenging. For example, one can consider extracting the morphological features, such as cell shape index, cytoskeletal orientation, and cell aspect ratio. In particular, the cell shape index is derived from the cell area and perimeter,

$$\text{CSI} = \frac{4\pi A}{P^2} \quad (6)$$

For cells in adherent culture, the cell orientation provides critical information about the biophysical state of the cells to indicate the cell phenotype and cellular processes, such as proliferation, differentiation, and cell apoptosis or necrosis, as these cell fate processes have their morphological signatures.

In the ImageMKS framework, the focus has been on extracting the morphological features corresponding to cell differentiation (cell shape attributes), where it is known that fully developed muscle cells are elongated ($\text{CSI} \approx 1$) and have a preferred orientation. The results are presented in Fig. 7, where each row corresponds to the measurements obtained from the corresponding cell image in Fig. 5.

CONCLUSIONS

The present study demonstrates that ‘automated image processing workflows’ have the potential to increase the accuracy of traditional segmentation

Fig. 7.. Each row shows four measurements for the segmented cells in the corresponding rows of Fig. 5. The columns represent the nucleus area, cell shape index, cytoskeleton orientation, and nucleus to cytoskeleton length ratio as histograms and probability densities. The histogram shows the total number of cells in an image having the property on the x-axis. From these histograms, it can be seen that the second image (row two) has the greatest number of cells compared to the first and third images. The probability is normalized by the total number of cells and therefore is not impacted by the total number of cells.

workflows by combining multiple segmentation workflows into one using a majority vote. The increase in accuracy comes with only a small increase in the computational effort. This study further established the advantages of the developed method visually and through comparison with traditional methods. The workflow is accurate at segmenting cell nuclei from a dataset of 42 images, with very different features, such as cell density and brightness. It also performs significantly better than traditional methods and only requires a small number of intuitive parameters to be defined by the user. The framework is provided through the open-source ImageMKS platform, which is accessible via python or through command line tools and gives users the ability to quickly and reliably analyze the cellular images.

ACKNOWLEDGEMENTS

SPV and SRK acknowledge support from NSF-IGERT Award 1258425. SRK acknowledges the Vajra fellowship. Several of the references were curated in a NIST survey.[36]

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest. On behalf of all authors, the corresponding author states that there is no conflict of interest.

REFERENCES

1. B. Basu, *Biomaterials science and tissue engineering: principles and methods* (Cambridge University Press, New York, 2017).
2. M.M. Frigault, J. Lacoste, J.L. Swift, and C.M. Brown, *J. Cell Sci.* 122, 753. (2009).
3. K. Ravikumar, S.K. Boda, and B. Basu, *Bioelectrochemistry* 116, 52. (2017).
4. J. Lozano-Gerona, and Á.-L. García-Otín, *Anal. Biochem.* 543, 30. (2018).
5. I.V. Grishagin, *Anal. Biochem.* 473, 63. (2015).
6. M.A. Alyassin, S. Moon, H.O. Keles, F. Manzur, R.L. Lin, E. Hæggestrom, D.R. Kuritzkes, and U. Demirci, *Lab Chip* 9, 3364. (2009).
7. A. Merouane, N. Rey-Villamizar, Y. Lu, I. Liadi, G. Romain, J. Lu, H. Singh, L.J.N. Cooper, N. Varadarajan, and B. Roysam, *Bioinformatics* 31, 3189. (2015).
8. W. Yu, H.K. Lee, S. Hariharan, W. Bu, and S. Ahmed, *Cytometry* 77A, 379. (2010).
9. D. Padfield, J. Rittscher, and B. Roysam, in *Proceedings of the 2008 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, 376, (2008).
10. J. Shu, H. Fu, G. Qiu, P. Kaye, and M. Ilyas, in *Proceedings of the 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 5445, (2013).
11. M. Schwendy, R.E. Unger, M. Bonn, and S.H. Parekh, *BMC Bioinform.* 20, 39. (2019).
12. S.J. Florczyk, M. Simon, D. Juba, P.S. Pine, S. Sarkar, D. Chen, P.J. Baker, S. Bodhak, A. Cardone, M.C. Brady, P. Bajcsy, C.G. Simon, and A.C.S. Biomater, *Sci. Eng.* 3, 2302. (2017).

13. L. P. Coelho, A. Shariff, and R. F. Murphy, in *Proceedings of the 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, 518, (2009).
14. J. Hua, C. Sima, M. Cypert, G. C. Gooden, S. Shack, L. Alla, E. A. Smith, J. M. Trent, E. R. Dougherty, and M. L. Bittner, *J. Biomed. Opt.*, 17, 046008, (2012).
15. E. Meijering, and I.E.E.E. Sig, *Process. Mag.* 29, 140. (2012).
16. K. Ravikumar, G.P. Kar, S. Bose, and B. Basu, *RSC Adv.* 6, 10837. (2016).
17. G. Thiruvikraman, P.K. Mallik, and B. Basu, *Biomaterials* 34, 7073. (2013).
18. V. Ljosa and A. E. Carpenter, *PLoS Comput. Biol.*, 5, e1000603, (2009).
19. P. Márquez-Neila, L. Baumela, and L. Alvarez, *IEEE Trans. Pattern Anal. Mach. Intell.* 36, 2. (2014).
20. O. Dzyubachyk, W.A. van Cappellen, J. Essers, W.J. Niesen, and E. Meijering, *IEEE Trans. Med. Imag.* 29, 852. (2010).
21. G. Srinivasa, M.C. Fickus, Y. Guo, A.D. Linstedt, and J. Kovacevic, *IEEE Trans. Image Process.* 18, 1817. (2009).
22. S. Dimopoulos, C.E. Mayer, F. Rudolf, and J. Stelling, *Bioinformatics* 30, 2644. (2014).
23. R. Bise, K. Li, S. Eom, and T. Kanade, 12, (2009).
24. Seferbekov, Selim, “2018 Data Science Bowl [ods.ai] top-coders 1st place solution” (Kaggle, 2019). <https://kaggle.com/c/data-science-bowl-2018/discussion/54741>.
25. T.J. Collins, *Biotechniques* 43, S25. (2007).
26. J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, B. Schmid, J.-Y. Tinevez, D.J. White, V. Hartenstein, K. Elieiri, P. Tomancak, and A. Cardona, *Nat. Methods* 9, 676. (2012).
27. S. Voigt, “ImageMKS” (ImageMKS, 2021). <https://svenpvoigt.github.io/ImageMKS>.
28. T.E. Oliphant, *Comput. Sci. Eng.* 9, 10. (2007).
29. S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, *PeerJ*, 2, e453, (2014).
30. W. Burger, and M.J. Burge, *Principles of Digital Image Processing* (Springer, London, 2009).
31. L.G. Shapiro, and G.C. Stockman, *Computer vision* (Prentice Hall, Upper Saddle River, NJ, 2001).
32. M. B. Ahmad and Tae-Sun Choi, *IEEE Trans. Consum. Elect.*, 45, 674, (1999).
33. N. Malpica, I. Vallcorba, and J.M. Garcia-Sagredo, *Cytometry* 28, 289. (1997).
34. W. McKinney, in *Proceeding of the 9th Python in Science Conference*, 56, (2010).
35. C. Goutte and E. Gaussier, *Adv. Inform. Retrieval.* (Springer, Berlin Heidelberg, 2005).
36. P. Bajcsy, A. Cardone, J. Chalfoun, M. Halter, D. Juba, M. Kocielek, M. Majurski, A. Peskin, C. Simon, M. Simon, A. Vandecreme, and M. Brady, *BMC Bioinform.* 16, 330. (2015).

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.