EPiC Series in Computing

Volume 75, 2021, Pages 69–79

CAINE 2020. The 33rd International Conference on Computer Applications in Industry and Engineering

Image Processing for High-Throughput Phenotyping of Seeds

Venkat Margapuri, Chaney Courtney and Mitchell Neilsen
Department of Computer Science
Kansas State University
Manhattan, KS, USA
marven@ksu.edu, chaneylc@ksu.edu, and
neilsen@ksu.edu

Abstract

High-throughput phenotyping of seeds is the assessment of seed morphometry to aid in the prediction of yield, tolerance, resistance, and development of seeds in various environmental conditions. The paper focuses on the application of 3D graphics to image processing as a means to conduct seed phenotyping better. The paper proposes two algorithms - similar in the outcome, but different in implementation. The algorithms perform image processing on a variety of seeds such as wheat, soy, sorghum, rough rice, white rice, and canola to arrive at their morphometric estimations. In the area of static image processing, addressed are at least three common yet significant problems of seed clusters on images, skewed images, and poor image quality. As a means to address the problems, we propose the use of low-cost physical components. The algorithms provide the estimated count, area, perimeter, length, and width of seeds within an image.

Keywords: High-throughput phenotyping, 3D graphics, Watershed algorithm, image processing, seed morphometry

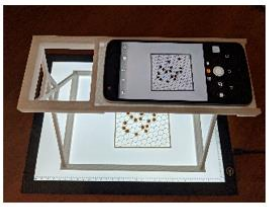
1 Introduction

At the current growth rate of 1.3%, the population of the world is estimated to be at 9.3 billion by 2050. However, the world cereal yield and agricultural production have decreased since 1961 [1]. A suitable means to tackle this challenge is the application of phenotyping techniques to seeds. Seed Phenotyping is the comprehensive assessment of complex seed traits such as growth, development, tolerance, resistance, ecology, yield, and the measurement of parameters that form more complex traits [2]. High-throughput phenotyping increases the accuracy of measurements while reducing costs. Automation, remote sensing, data integration, and experimental design help reduce the costs of measurements. Several studies [3][4][5][6] emphasize the importance of morphometry in predicting the behavior of

performance in different environmental settings.

One of the foundational ideas for the work is the application of 3D graphics to high-throughput phenotyping. Common problems faced in the area of image processing include clusters on images, skewness of images, and poor image quality. The work proposes a technique that uses low-cost physical components, which improves the accuracy of morphometric estimations. The proposed components are as follows:

1. 3D printed stand to hold the image capture device. For this experiment, the image capture device is a phone; hence, the 3D printed stand has a groove fit to hold a phone.
2. 3D printed mesh with the idea that seeds lay within the hexagons of the mesh.
3. A lightbox to ensure that the images lay on a common background at the time of image capture leading to good quality images.



3D Stand holding a Phone

3D Mesh

Light Box

Figure 1: Proposed Components

Furthermore, the work provides an algorithm that does not use the components described above. Instead, the alternate approach leverages the Watershed algorithm for image segmentation.

In the remainder of the paper, section 2 specifies related work, section 3 discusses the proposed algorithms for static image processing, and section 4 concludes the paper.

2 Related Work

One of the building blocks for the algorithms is the Watershed algorithm, as discussed by F. Meyer and S. Beucher in the work, 'The Morphological Approach to Segmentation: The Watershed Transformation' [7]. The work discusses in detail the intricacies of the algorithm, including the tools, transformations, uses, and the application of Watershed to images.

Some of the Android applications which were similar and acted as useful validation/ comparison metrics were Smart Grain [8], Grain Scan [9], Leaf IT [10], and Seed Counter [11]. While Leaf IT was built for the morphometric estimation of leaves, all of the other applications above were developed for morphometric estimations of seeds.

3 Image Processing Algorithms

The following sub-sections present the developed image processing algorithms. The algorithms are developed using OpenCV-Python and applied to six different types of seeds: wheat, soy, rough rice, white rice, sorghum, and canola.

3.1 Mesh Algorithm

The Mesh algorithm refers to the static image processing algorithm involving the 3D stand, 3D mesh, and lightbox, as shown in Figure 1. Each of the components has a specific use case. The algorithm solves multiple problems in the following ways:

1. **3D Stand:** The 3D stand solves the problem of skew that most image processing algorithms encounter. The stand ensures that the image capture device such as a phone is always orthogonal to the surface. This angle ensures the image is not skewed. The stand used for the experiment has a height of 110 mm (11 cm) and printed using a white filament.
2. **3D Mesh:** The 3D mesh addresses the problem of the object touching on images. The hexagonal boundaries act as barriers ensuring that the seeds do not touch each other at all times. Each of the hexagons within the mesh has a side length of 5 mm and built using a white filament.
3. **Lightbox:** The lightbox ensures that the images which are captured have a bright background. A bright background makes it easier to identify the objects on the image and eliminate any noise such as tiny dirt particles that might be present on the image.

The algorithm can be generalized as a two-step process, described as follows:

1. Detect the mesh on the image and estimate the morphometry. The hexagons within the mesh are saved as ground truth values.
2. Detect the seeds on the image and use the estimated morphometry of the mesh to infer on the seeds.

3.1.1 Mesh Detection and Estimation

A key point worth reiterating here is that the filament used to build the mesh and the light emitted by the lightbox are white. From Figure 2, the mesh is merely a shade darker to the light emitted by the lightbox.

1. For reproducibility, the mesh should have pixel values between 170 and 255 for each channel.

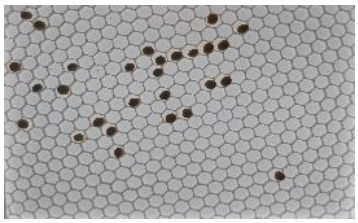


Figure 2: Image of Soy Seeds within a Mesh

2. Perform a median blur and apply canny edge detection to detect the edges on the image.
3. Detect the contours of the edges identified by canny edge detection.
4. (Observation/ Assumption) Find all contours which occupy an area greater than an individual hexagon of the mesh in pixels.
5. Compute the median area occupied by the mesh in pixels and consider it as the area occupied by each hexagon in the mesh. Likewise, compute the median perimeter of the mesh.

A relationship between the number of pixels to the area in metric units is now established. The area of a hexagon in mm^2 is given by $(3 * 1.732 * a * a)/2$ where a is the side of the hexagon in mm, and the perimeter is given by $6 * a$ where a is the side of the hexagon in mm.

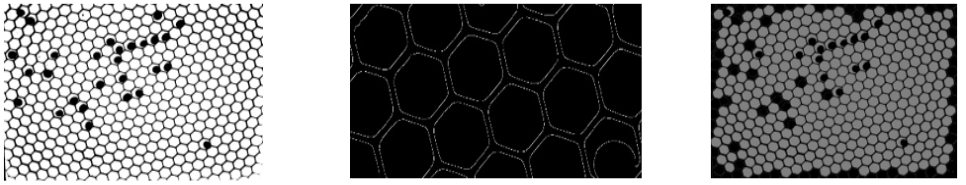


Figure 3: Mesh Detection and Estimation

3.1.2 Seed Detection and Estimation

1. Apply binary thresholding on the source image to filter out all of the white pixels and retain only the black pixels, i.e., pixels that represent seeds on the image.
2. Median blur the image and perform canny edge detection on the image.
3. Detect seeds on the image and compute the area occupied by each of the seed contours.
4. (**Seed Count**) The number of contours deemed seeds is the estimated number of seeds on the image.
5. (**Area and Perimeter Estimation**) The area of seeds in metric units is estimated by taking the median area occupied by the mesh's hexagons in pixels and metric units as ground truth values, and cross-multiplying, i.e., $seed\ area\ in\ mm^2 = (seed\ area\ in\ pixels * hexagon\ area\ in\ mm)/(median\ hexagon\ area\ in\ pixels)$. Likewise, a simple cross-multiplication is be applied to obtain the perimeter of the seed in mm.
6. (**Length and Width Estimation**) The lengths and widths of each contour are estimated using minimum area rotated rectangles. Rectangles are fit to the contour samples. The longest axis returned is considered the length, while the shorter axis is the width. Besides, connecting the left-most to the right-most and top-most to the bottom-most points, and computing the length of the connecting lines is a means to estimate the length and width of the seeds.

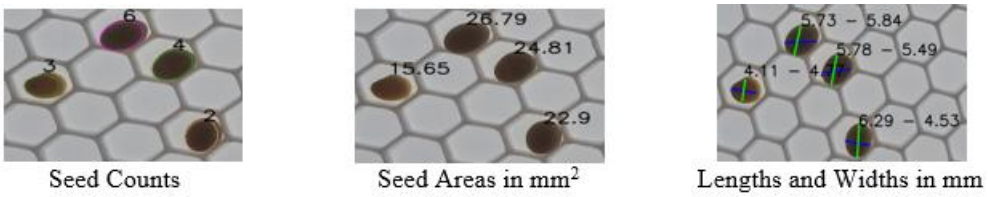


Figure 4: Estimated Seed Morphometry (Image Zoomed In)

As mentioned, the experiment is performed on six different types of seeds. The algorithm performs well consistently on five of the six types of seeds in question failing on white rice. The reason is that the light emitted by the lightbox is the same color as the seed of white rice. As a result, the algorithm fails to distinguish between the mesh and seeds. While the proposed algorithm still holds, a different experimental setup involving a contrasting background is essential for white rice. An implementation of the algorithm in Python-OpenCV is available at MeshAlgorithm.

3.2 No-Mesh Algorithm

The No-Mesh algorithm is a static image processing algorithm that does not involve the 3D printed mesh, unlike the algorithm stated in section 3.1. Instead, the algorithm leverages the popular Watershed algorithm to segment seed clusters on images. The other components, 3D stand, and the lightbox are optional but recommended to achieve the proper performance of the algorithm. The algorithm is explained as a three-step process, described as follows:

1. Detect segmented seeds on the image i.e. seeds not in contact with other seeds on the image.
2. Apply the Watershed algorithm on the image and segment the seeds to perform morphometric estimations.
3. Estimate the morphometry of seeds using a ground truth object.

3.2.1 Detection of Segmented Seeds

The detection of segmented seeds makes use of the interquartile range, convexity defects, and the convex hull. The **Interquartile Range (IQR)** is a measure of the middle 50% of the values in a dataset. It is a proven mathematical technique that aids in the detection of outliers.

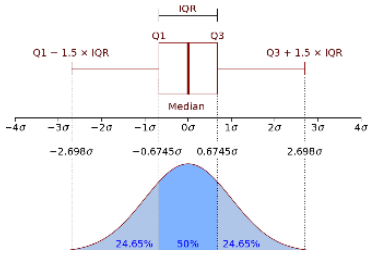


Figure 5: IQR Calculation

In Figure 5, the IQR is $Q3 - Q1$. Q1 represents a quarter of the way through the list, and Q3 represents three-quarters of the way through the list. The IQR rule is that all values that are not between $Q1 - 1.5\ x\ IQR$ and $Q3 + 1.5\ x\ IQR$ are outliers.

Convex hull and convexity defects: Firstly, a convex object is one where none of the interior angles is greater than 180 degrees. The convex hull is a tight-fitting boundary around the object. As shown in Figure 6, the image on the left is convex. It has a convex hull that wraps entirely around the boundary, whereas the image on the right is not convex. The convex hull does not wrap around correctly. The imperfections in the convex hull are known as convexity defects.

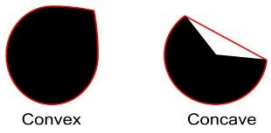


Figure 6: Convex hull and convex defects

1. Convert the image to grayscale and gaussian blur the image using a $3\ x\ 3$ kernel.
2. To threshold the seeds from the background, apply inverted binary Otsu thresholding on the image.
3. Detect the contours of the seeds along with the area occupied by each of the contours. Also, find the convex hull of each of the seed contours.
4. Detect a segmented seed by evaluating the convexity of a contour, i.e., $if\ length(contour)/\ length(convexhull)\ <=\ 3$, the contour is identified as a potential segmented seed.
5. Find the area occupied by each of the contours of the potential segmented seeds.
6. Remove the outliers from the potential segmented seeds by applying the IQR rule for outliers, stated above.

7. **(Case for Touching Seeds)** An edge case is that a few false positives might satisfy the step-four check. To eliminate them, filter any contours where the center is relatively white, i.e., with an intensity of about 170 or higher.
8. After segmentation, compute the morphometries, including the farthest point from the convex hull to the seed. Results are utilized further in section 3.2.2.3.

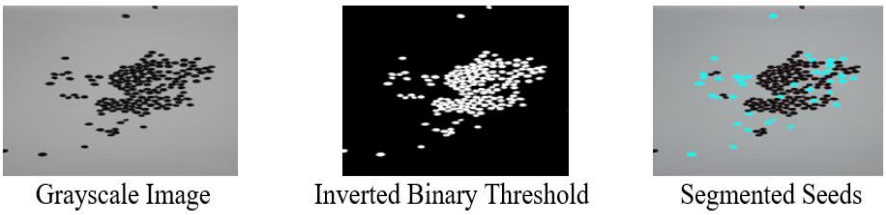


Figure 7: Detection of Segmented Seeds

3.2.2 Application of Watershed Algorithm

The Watershed algorithm's application relies on the concepts of morphological operations, erosion, dilation, closing, and opening, briefly explained as follows:

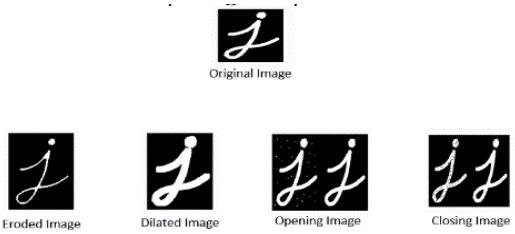


Figure 8: Morphological Operations

The principle behind morphological operations is the idea that 'on a grayscale image, black (0) and white (255) pixels can be identified with good precision'.

Erosion: Erosion is a means to pare the image. A kernel of any size is convolved across the image. If one of the pixels along the kernel is black, all pixels on the image with respect to the kernel, are set to black (0).

Dilation: Dilation is a means to enhance the size of the image. A kernel is convolved across the image. If one of the pixels on the kernel is white, all pixels on the kernel are set to white (255).

Opening: Opening is a means to eliminate noise from an image. It is an erosion followed by a dilation.

Closing: Closing is a means to fill out the gaps in the image. It is a dilation followed by an erosion.

3.2.2.1 Preprocessing and Application of Watershed Algorithm

1. Convert the image to grayscale and apply binary thresholding to the image.
2. Perform a morphological opening to remove noise and a morphological closing to remove any holes in the image.
3. **(Background Identification)** Sure background area on the image is identified by performing a dilation.
4. **(Foreground Identification)** Compute the distance transform on the image and apply a threshold on the distance transform. The threshold values depend on the image.

5. **(Unknown/ Border)** The regions which are neither foreground nor background are the unknown/ border areas. They are computed by subtracting the foreground area from the background area.
6. **(Label Assignment)** Assign labels to the identified foreground, background, and unknown areas. If using OpenCV, the cv.ConnectedComponents() function can be used for this purpose. It assigns 0 to the background and random numbers starting from 1 to the objects in the foreground. However, the Watershed algorithm assumes all regions with a value of 0 as unknown areas. To avoid this scenario, increment all values by one, so the background has a value of 1 and assign 0 to all pixels belonging to the unknown area.
7. Apply a gaussian blur to the image to reduce the inner contour noise and avoid over-segmentation.
8. Apply the Watershed algorithm to the blurred image and capture the output markers.

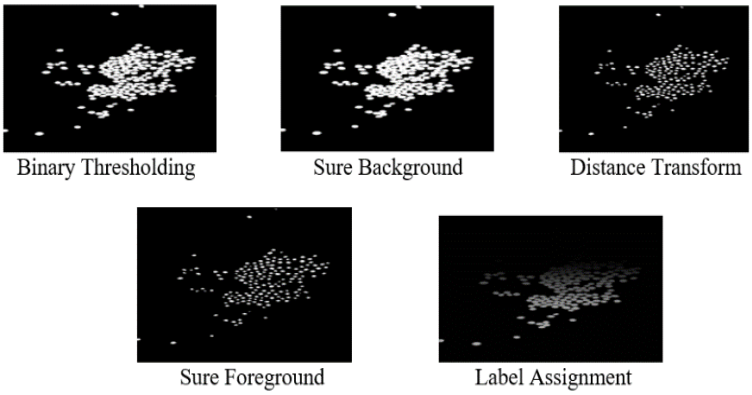


Figure 9: Preprocessing and Application of Watershed

3.2.2.2 Image Segmentation Post-Watershed Application

1. Draw the boundaries of the watershed segments on the markers produced by applying the Watershed algorithm (Remember: The Watershed algorithm indicates boundaries by -1).
2. Change the boundary markings from -1 to background, i.e., 0.
3. Invert the background and the foreground, i.e., set all 0's to 255's and the others to 0's, to identify boundaries in white since markers is a grayscale image.
4. Find contours on markers that also return the hierarchy of the contours.
   Note: Contour hierarchy is an OpenCV concept. The hierarchy is a data structure that allows contours to access four different relative values, next contour, previous contour, first child, and parent contour.
5. Use the hierarchy to find all contours which are not considered noise. Identify child contours larger than a quarter of the average segmented seed contour area and mark them as ‘parent’ contours.

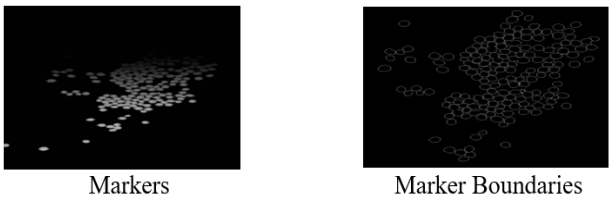


Figure 10: Post-Watershed Image Segmentation

3.2.2.3 Counting Seeds

Seeds in images are often overlapping or clustered; therefore, counting the number of seeds on the image is a tricky task. The algorithm, however, provides two types of counting techniques, namely, area-based count and contour-based count.

Area-based Count: The area-based counting technique uses the average segmented seed areas to estimate the number of seeds in a given contour containing a cluster of seeds. For example, if the average pixel area of segmented seed contours is 10000 and the contour with a cluster of seeds has an area of 50000, the area based counting technique estimates the number of seeds in the cluster to be five.

Contour-based Count: The contour-based counting technique uses the results computed in section 3.2.1, i.e., the mean (SS_Mean), median (SS_Median), and standard deviation (SS_SD) of the segmented seeds. The count also makes use of the contour area (PC_CA), and the fixed-point depths (PC_FPD) of the contours identified as 'parent' contours in section 3.2.2.2. The estimation of the seed count is defined as the following:

$$If\ Mean\ (PC\_FPD) > SS\_Median + 3 * SS\_SD:$$
$$count = rounded\ value\ of\ (PC\_CA/SS\_Mean)$$
$$Else: count = 1$$

The algorithm is used to count the number of seeds on images of different seeds. The results are shown below.

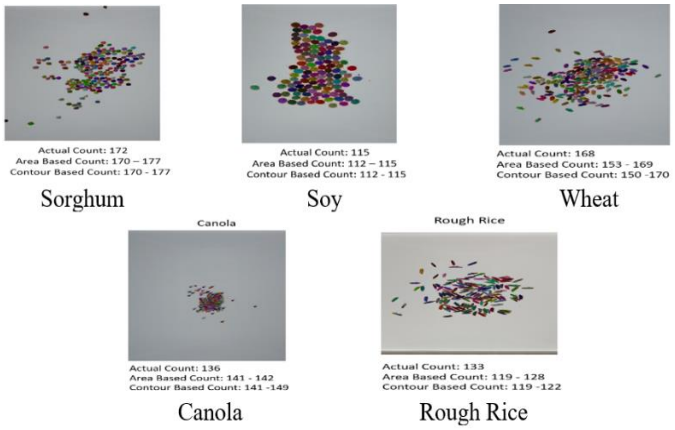


Figure 11: Seed Counts using No-Mesh Algorithm

As observed from the experimental results in Figure 11, the algorithm performs well on the images of sorghum, soy, and wheat, where the range of estimated count is within the actual number of seeds on the image. However, on the seeds of canola and rough rice, the algorithm overcounts and undercounts, respectively. The results for white rice are not presented because the algorithm fails to detect the seeds precisely. Note that a change in the arrangement, be it the position or the orientation of seeds on the image, leads to the detection of different contours and might influence the seed count since it is entirely dependent upon the detected contours.

3.3 Seed Morphometry Estimations

Unlike the Mesh algorithm, the No-Mesh algorithm has no ground truth value. Hence, the idea is to use to capture the image with an object of known morphometry and base the estimations on the ground truth. For this experiment, the Penny, a US coin, is used.

For the experiment, a total of four pennies are placed in each of the four corners of the lightbox when the image is captured. The pennies on the image are identified based on the size of the contour, i.e., the four most massive contours found on the image are assumed to be those of coins and circularity computed as $4 * pi * (area/perimeter^2)$ where area and perimeter are pertinent to the contour in question.

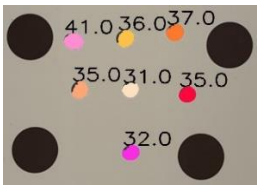


Figure 12: Estimated Seed Areas in mm^2 using Pennies

3.4 Comparison to Other Applications

The areas estimated by the proposed algorithms are compared against the estimates of other applications. Seed Counter, LeafIT, SmartGrain, and a manual estimation performed using a vernier caliper are compared. A sample of seven soy seeds is used for the comparison. The reason for the soy seed sample is that some of the applications in consideration are built to estimate specific types of seeds and do not necessarily pick up seeds smaller or larger than a specific size. Soy is one of the seeds which is estimated by all of the applications in question.

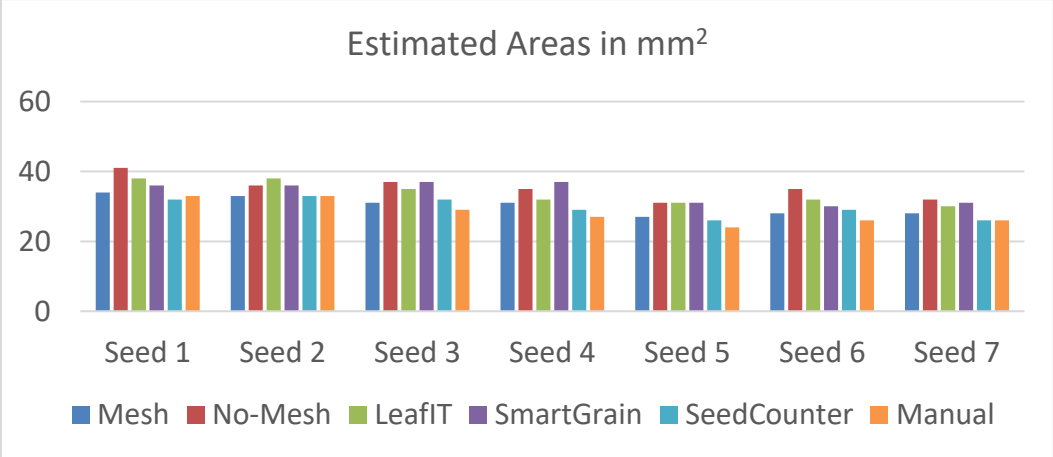


Figure 12: Estimated Seed Areas in mm^2 using Pennies

From the results:

1. Manual estimations using a vernier caliper are consistently lower than those provided by any of the applications and proposed algorithms.
2. The results provided by the Mesh algorithm are most similar to those of the results provided by the android app, Seed Counter with the most considerable difference in estimates being two mm^2 for seeds one, four, and seven.
3. The results provided by the No-Mesh Algorithm are most similar to that of LeafIT's, with the most considerable difference in estimations being three mm2. Not surprisingly, both algorithms use ground truth reference objects. The difference, though, is that LeafIT estimates morphometry of leaves,

one at a time. In contrast, the No-Mesh algorithm estimates morphometry of seeds, multiple at a time.

4 Future Work and Conclusion

The Mesh algorithm is an excellent addition to the current image processing techniques using 3D graphics. With encouraging results in the initial phase, the focus in the next phase is to solidify the algorithm by testing the feasibility and accuracy of the algorithm on various kinds of meshes and further study the impact of factors such as the height of image capture, image size, camera distortion and seed orientation on the algorithm. Current plans to implement the No-Mesh algorithm as part of an Android app are in progress, and an initial version of the implementation is currently being tested.

References

[1] Maisonet-Guzman, O. E. (2011). Food security and population growth in the 21st century. Retr. from http://www.e-ir.info/2011/07/18/food-security-and-population-growth-in-the-21st-century

[2] Li, L., Zhang, Q., & Huang, D. (2014). A review of imaging techniques for plant phenotyping. *Sensors*, *14*(11), 20078-20111.

[3] Neilsen, M. L., Gangadhara, S. D., & Rife, T. (2016, September). Extending watershed segmentation algorithms for high throughput phenotyping. In *Proceedings of the 29th International Conf. on Computer Applications in Industry and Engineering, Denver, CO*.

[4] Mir, R. R., Reynolds, M., Pinto, F., Khan, M. A., & Bhat, M. A. (2019). High-throughput phenotyping for crop improvement in the genomics era. *Plant Science*, *282*, 60-72.

[5] Wiley, V., & Lucas, T. (2018). Computer vision and image processing: a paper review. *International Journal of Artificial Intelligence Research*, *2*(1), 29-36.

[6] Kehel, Z., Sanchez-Garcia, M., El Baouchi, A., Aberkane, H., Tsivelikas, A., Chen, C., & Amri, A. (2020). Predictive characterization for seed morphometric traits for genebank accessions using genomic selection. *Frontiers in Ecology and Evolution*, *8*, 32.

[7] Beucher, S., & Meyer, F. (1993). The morphological approach to segmentation: the watershed transformation. *Mathematical morphology in image processing*, *34*, 433-481.

[8] Tanabata, T., Shibaya, T., Hori, K., Ebana, K., & Yano, M. (2012). SmartGrain: high-throughput phenotyping software for measuring seed shape through image analysis. *Plant Physiology*, *160*(4), 1871-1880.

[9] Whan, A. P., Smith, A. B., Cavanagh, C. R., Ral, J. P. F., Shaw, L. M., Howitt, C. A., & Bischof, L. (2014). GrainScan: a low cost, fast method for grain size and colour measurements. *Plant Methods*, *10*(1), 23.

[10] Schrader, J., Pillar, G., & Kreft, H. (2017). Leaf-IT: An Android application for measuring leaf area. *Ecology and Evolution*, *7*(22), 9731-9738.

[11] Komyshev, E., Genaev, M., & Afonnikov, D. (2017). Evaluation of the SeedCounter, a mobile application for grain phenotyping. *Frontiers in plant science*, *7*, 1990.

[12] Amaravadi, S. (2018). *Mobile applications for high-throughput seed characterization* (Master's thesis, Kansas State University).

[13] Neilsen, M. L., Courtney, C., Amaravadi, S., Xiong, Z., Poland, J., & Rife, T. (2017, October). A dynamic, real-time algorithm for seed counting. In *Proc. Of the 26th International Conference on Software Engineering and Data Engineering*.

[14] Suzuki, S. (1985). Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing*, *30*(1), 32-46.

[15] Kornilov, A. S., & Safonov, I. V. (2018). An overview of watershed algorithm implementations

in open source libraries. *Journal of Imaging*, *4*(10), 123.

[16] Yuheng, S., & Hao, Y. (2017). Image segmentation algorithms overview. *arXiv preprint arXiv:1707.02051*.