# Instance segmentation for leaf counting

Ronan Fraser
Dr. Richard Green
University of canterbury
Rff17@uclive.ac.nz

***Abstract* — This paper proposes a method of leaf instance segmentation, focusing directly on accurate leaf counting in rosette plants. The difficulty of accurate leaf counting through instance segmentation arises due to the partial or full occlusion of leaves by neighboring leaves in the image. The neighboring leaf occlusion causes difficulties as the occlusions are interpreted as a single larger leaf when performing segmentation based on the difference between the leaf and the background.**

**The method proposed in this paper utilizes the Canny edge detection algorithm to detect borders between occluded leaves. The detected border is added to a border of the foreground segmentation which is found by subtracting an eroded version of the segmentation from a dilated version of the same segmentation. A distance transform and binary threshold is applied to the foreground-background segmentation of the original image to further define the leaves and to remove the petiole (leaf stalk). The total border is subtracted from the binary threshold of the distance transform to segment leaf instances. A labelling algorithm is applied to the segmented image to provide a count of the leaves.**

**The method was developed and tested by using a dataset provided by the Computer Vision Problems in Plant Phenotyping organization [7]. The method is evaluated by finding the absolute value of the difference in count between the result of the method and the absolute truth provided by the used dataset. When evaluated on the first 50 images of the A1 dataset, the method had an average absolute difference in count of 2.7 with a standard deviation of 2. This result is comparable to other methods and an improvement upon the Nottingham method [5], which only achieves an absolute difference in count of 3.8 with a standard deviation of 2.**

## I. Introduction

The lack of robust automated, image-based phenotyping methods is widely recognized as the major obstacle to ensuring global food security [1]. In an effort to contribute to the solution of this problem, this paper aims to provide a possible simple solution to a specific phenotyping problem. The phenotype focused on by this paper is leaf number. This paper proposes a method of leaf instance segmentation, focusing directly on leaf counting in rosette plants. Instance segmentation is the problem of detecting and delineating each distinct object of interest appearing in an image [2]. The difficulty of accurate leaf counting through instance segmentation arises through partial to full occlusion of leaves by neighboring leaves in an image. The neighboring leaf occlusion causes difficulties as the occlusions are interpreted as a single larger leaf when performing segmentation based on the difference between the leaf and the background. One such method for leaf instance segmentation (figure 1) [3] uses a variation on chamfer matching [4], where a template is matched to an instance using different shape templates, scale and rotations. A limitation of this work is the use of labelled images in template generation, meaning that the method could not be extended to different species or picture format without having a set of labelled images for said species or format. The Nottingham method of leaf instance segmentation [5] utilizes simple linear iterative clustering (SLIC) based super pixels. Individual leaf seed matching between a distance map [10] maxima and a super pixel centroid was used to overcome occlusion (Figure 4). The watershed segmentation algorithm [9] was used to apply the instance segmentation. A method of instance segmentation from the Leibniz Institute of plant genetics and crop plant research (IPK) (figure 3) [6], involves finding local maximum values from a distance map [10] calculation on a foreground-background segmented image. The IPK method resolves leaf occlusions by generating a skeleton image from which split points at the thinnest connection points are detected. A list of edge attributes is used to detect the exact positions of the leaf split points for the skeleton graph. A border line is calculated from the nearest background pixels to the split points and is used to separate leaf instances for segmentation using a region growing algorithm. This method is complex and requires multiple steps to determine occluded leaf borders, although can still provide results even when boarders have no visually detectable boundaries. Another method that is used for instance segmentation is the Wageningen method (figure 2) which consists of using the watershed algorithm on an Euclidian distance map [10] generated from a plant mask image which is created using a neural network. Utilizing a neural network requires that there is training data [13] consisting of ground truth segmentations, which can limit this method in some applications.

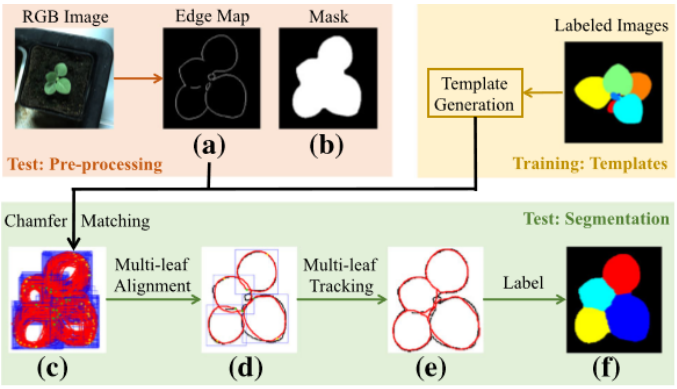


*Figure 1. Diagram of Chamfer matching method (MSU).*

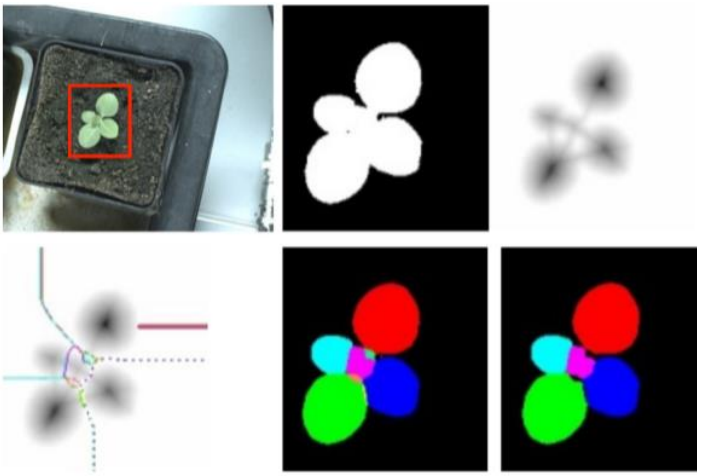

*Figure 2. Diagram displaying the Wageningen method.*

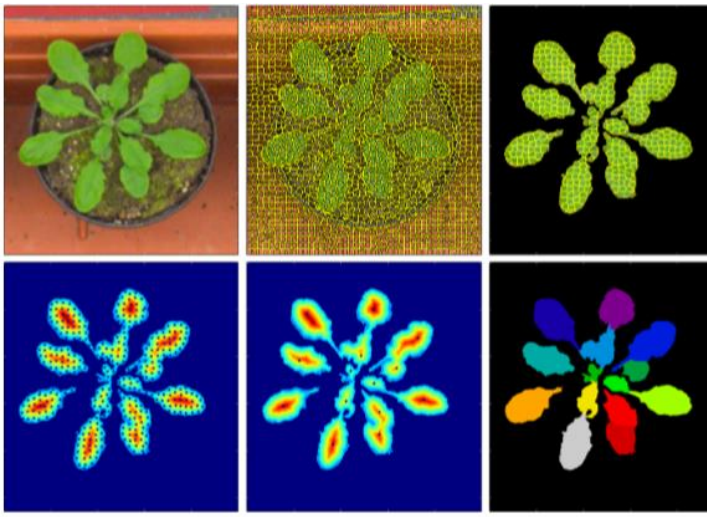

*Figure 4. Diagram showing the steps in the Nottingham method.*

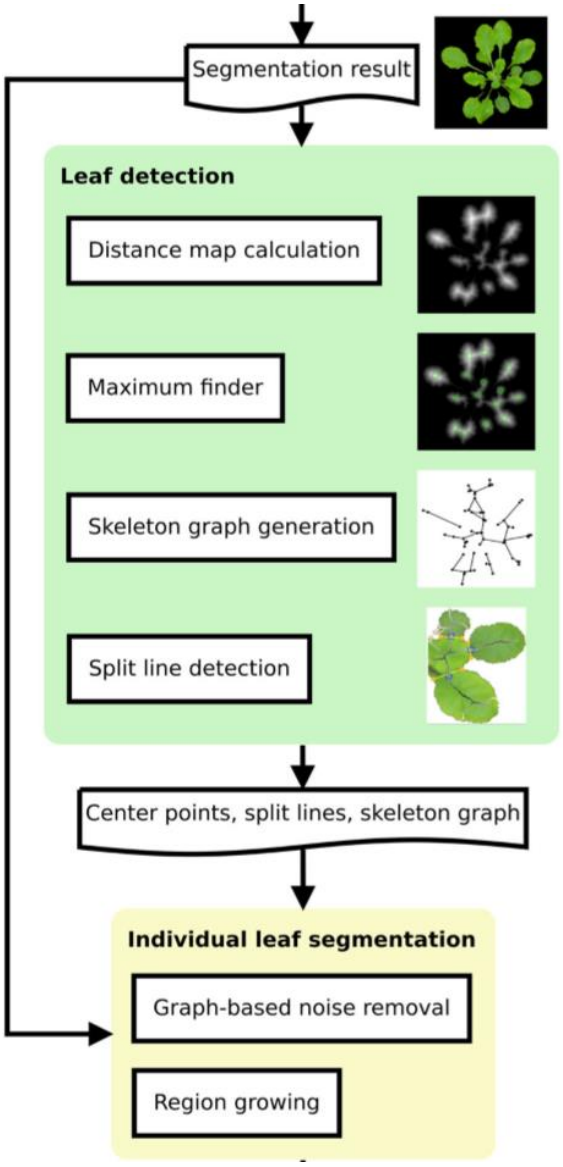


*Figure 3. Diagram showing the IPK method.*

## II. Method

This paper proposes a method of leaf instance segmentation, focusing directly on leaf counting. A requirement of this method is an image of the plant in the RGB color-space and a foreground-background binary segmentation of the image. The images that were used for the design and testing of this algorithm were sourced from the leaf segmentation challenge 2017 dataset published by the computer vision problems in plant phenotyping organization (CVPPP) [7]. The images in the dataset provides RGB color space images (figure 5) and a ground truth foreground-background binary image segmentation (figure 6).

The method deals with leaf occlusion by detecting the edges of the leaf in the foreground of the occlusion.

*Figure 5. Raw unprocessed RGB image from A1 dataset.*

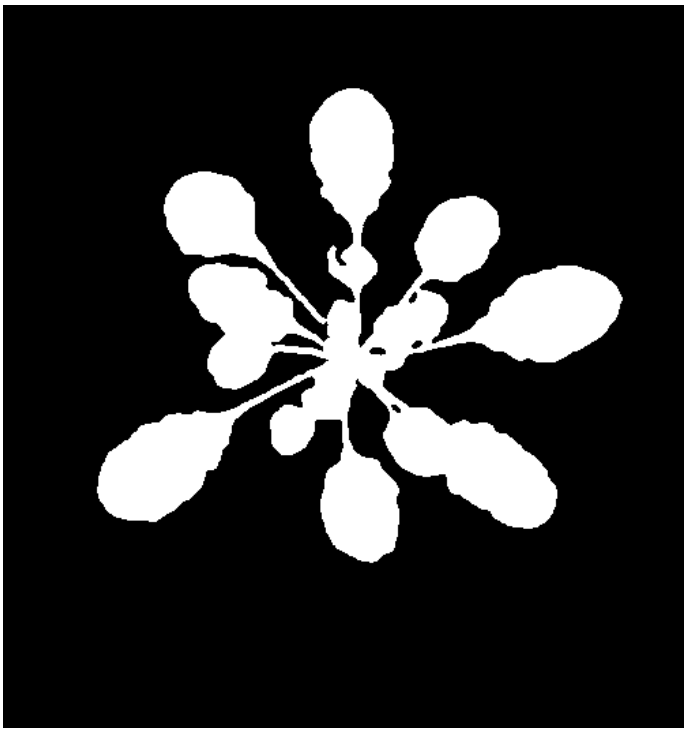

*Figure 6. Binary Foreground-Background segmentation from A1 dataset.*

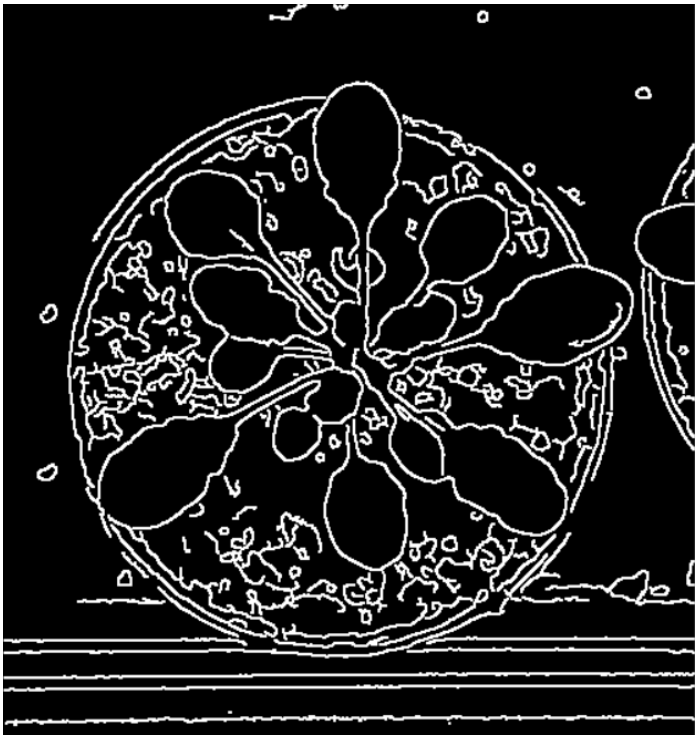

*Figure 8. Canny edge detection used on the original RGB image. Sigma = 2, Low threshold = 15, High threshold = 40.*

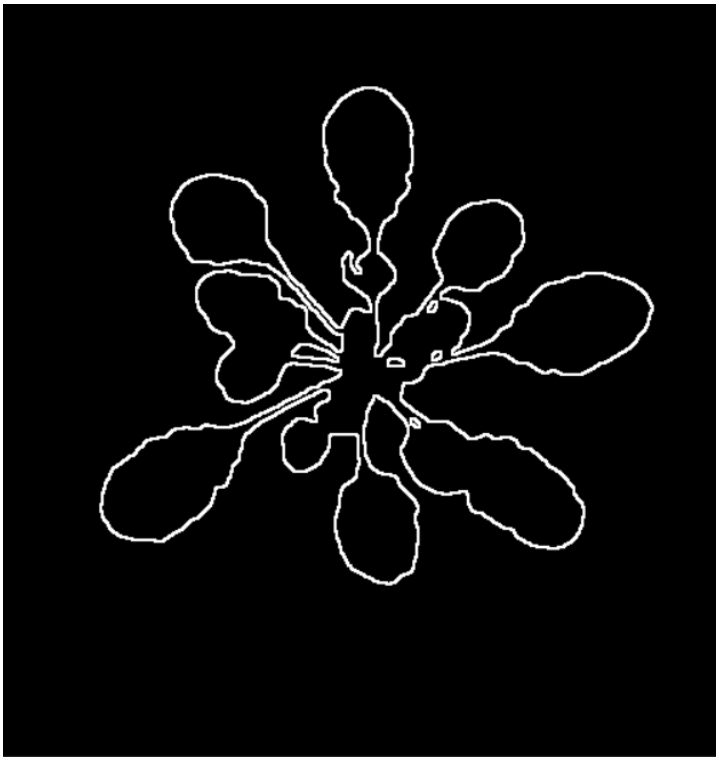

*Figure 7. Border of foreground-Background segmentation.*

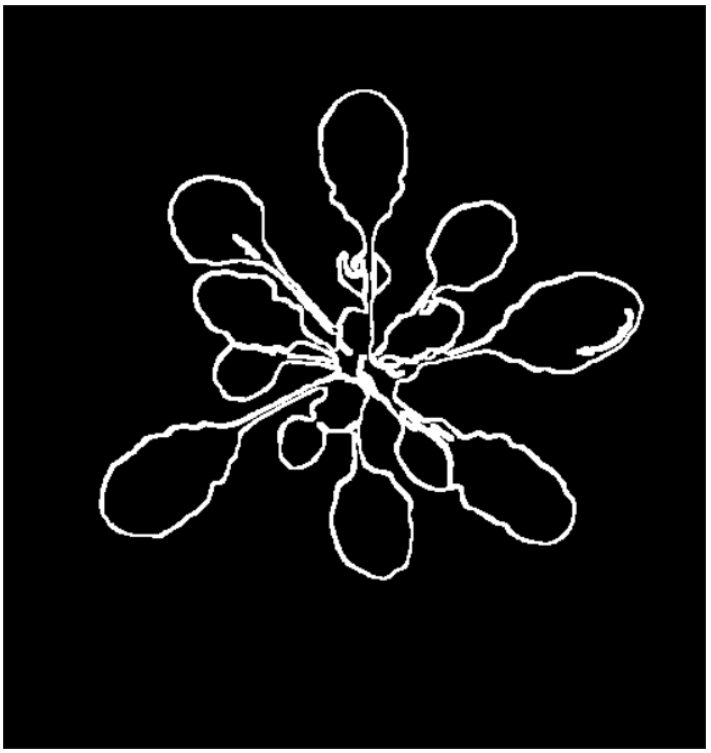

*Figure 9. Combined border generated through Canny and foreground segmentation. Background removed with binary mask.*

The first step of the method required the conversion of the binary foreground segmentation into a grayscale image so morphological [11] operations can be applied to it.

The first core operation of the method uses the Canny edge detection algorithm [8]. This algorithm is run on the original RGB image and is used to detect the edges of individual leaves (figure 8) and is fundamental to find borders between overlapping leaves. The foreground segmentation is used as a mask to discard edges that are found outside of the foreground segmentation (figure 9)

A border of the foreground segmentation is found by subtracting an eroded version of the segmentation from a dilated version of the same segmentation [11]. (figure 7) This border is of the foreground segmentation and does not contain any information about the borders of individual leaves.

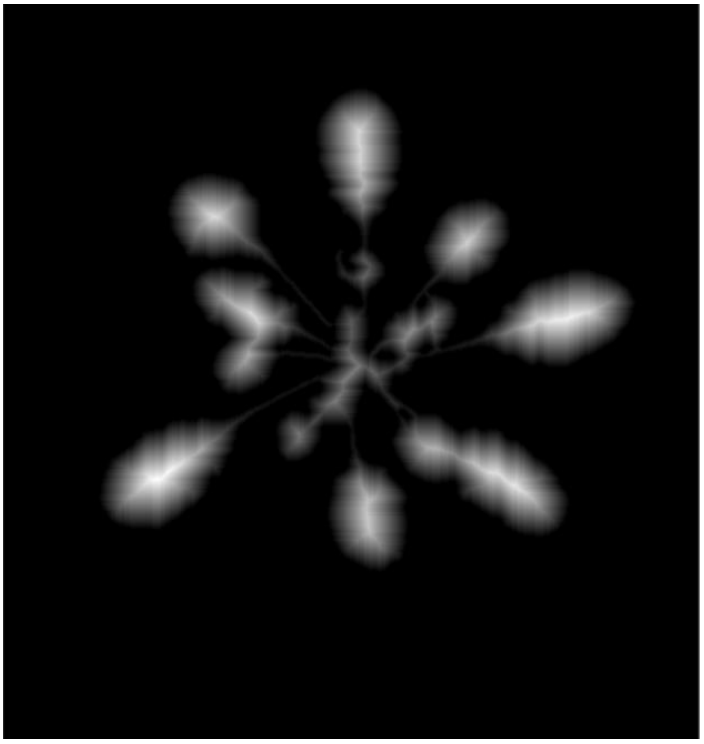

*Figure 10. Distance transform of foreground binary mask.*

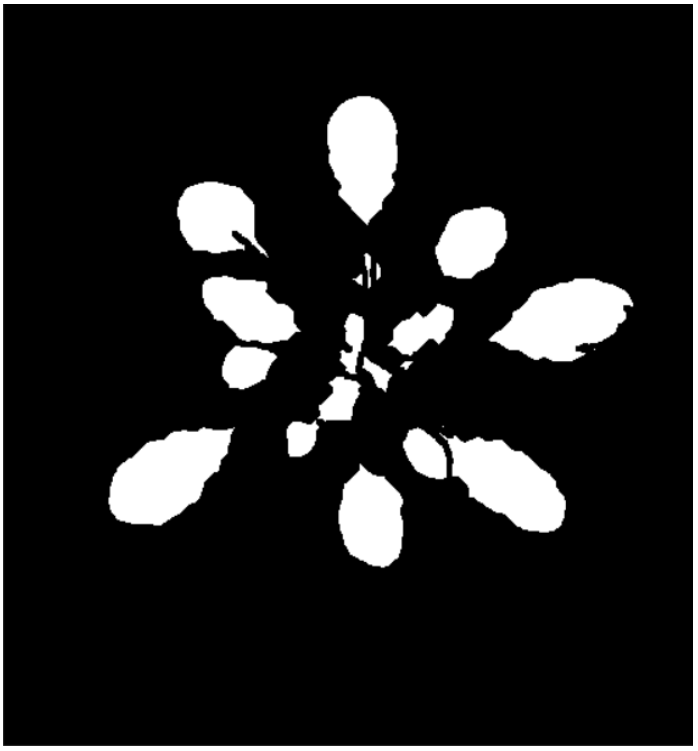

*Figure 12. The total boarder subtracted from the threshold of the distance transform.*

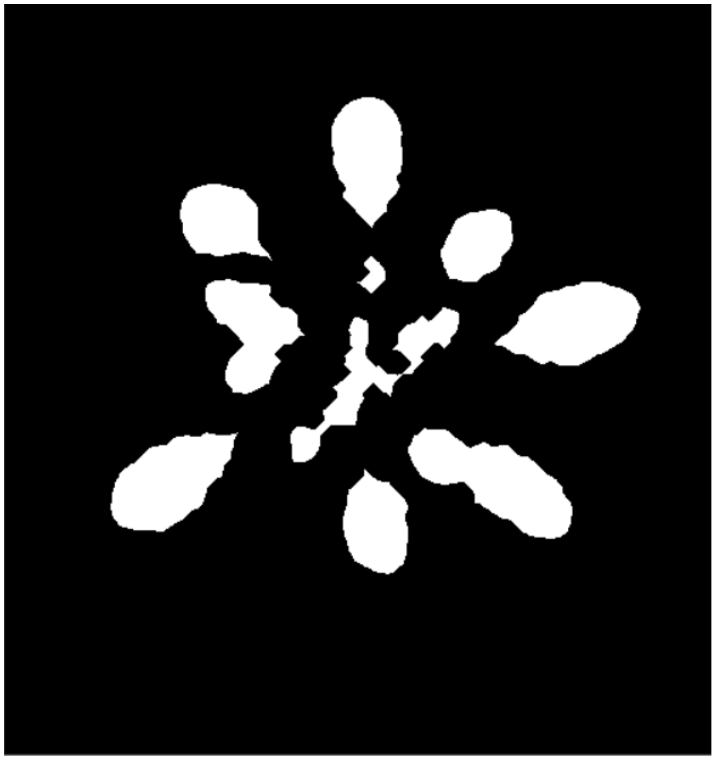

*Figure 11. Binary threshold of the distance transform using 40 as the threshold value.*

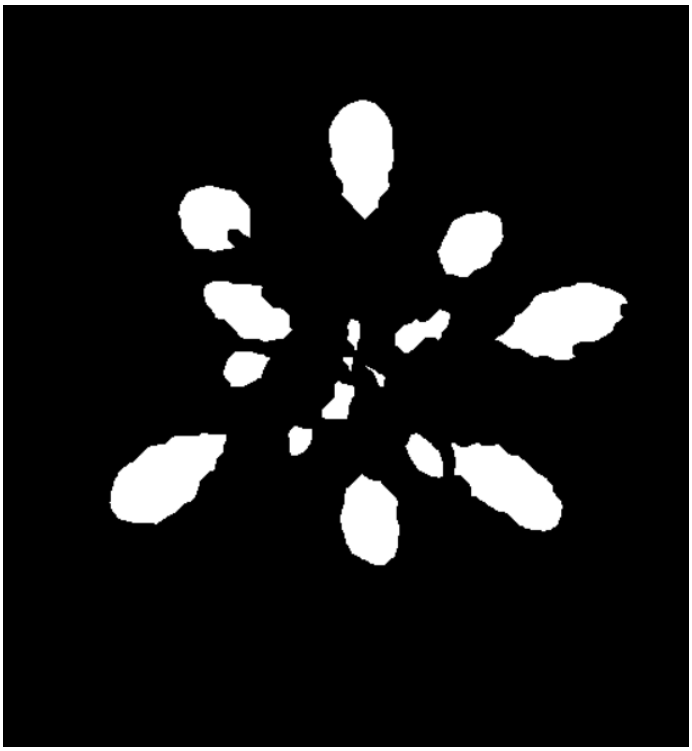

*Figure 13. Erosion and closing applied to image in figure 7.*

The foreground segmentation-based border was then combined with the Canny edge border using a bitwise OR operation [12] (figure 9). This operation created a border which contains lines between individual leaves which are overlapping.

A distance transform [10] is applied to the foreground segmentation (figure 10). The distance transform was used to find local maxima and allowed for a binary threshold to be applied to further define the individual leaf locations and remove the petiole (leaf stalk) (Figure 11).

To segment the leaf instances, the combined border is subtracted from the binary threshold generated from the distance transform (figure 12).

Two iterations of erosion [11] are then applied to the resulting image to further separate the instances. A closing operation [11] is also applied to fill/reduce any minor noise in the resulting image (Figure 13). A labelling algorithm is then applied to the segmented image to count the unique features (leaves) for the final leaf count. The final segmentation overlaid on to the border can be seen in figure 14.

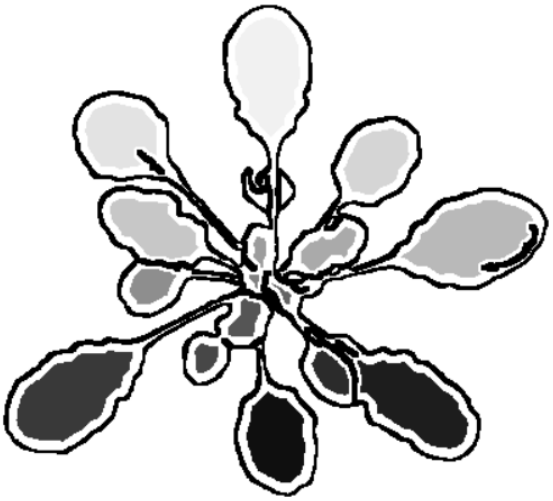

*Figure 14. The final segmentation overlaid on to the foreground segmentation border.*

*Table 1.Performance of different methods [5].*

| | SBD (%) | FBD (%) | \|DiC\| | DiC |
|---|---|---|---|---|
| IPK | | | | |
| A1 | 74.2 (7.7) | 97.4 (1.8) | 2.6 (1.8) | −1.9 (2.5) |
| A2 | 80.6 (8.7) | 99.7 (0.3) | 0.9 (1.0) | −0.3 (1.3) |
| A3 | 61.8 (19.1) | 98.2 (1.1) | 2.1 (1.7) | −2.1 (1.7) |
| ALL | 73.5 (11.5) | **98.0** (**1.9**) | 2.2 (1.7) | −1.7 (2.3) |
| Nottingham | | | | |
| A1 | 68.0 (7.4) | 94.6 (1.6) | 3.8 (2.0) | −3.6 (2.4) |
| A2 | 60.9 (18.5) | 87.5 (19.7) | 2.5 (1.5) | −2.5 (1.5) |
| A3 | 47.1 (25.0) | 79.4 (34.5) | 2.3 (1.8) | −2.3 (1.9) |
| ALL | 63.8 (15.3) | 91.2 (16.2) | 3.4 (2.0) | −3.2 (2.2) |
| MSU | | | | |
| A1 | 78.0 (6.4) | 95.8 (1.9) | 2.3 (1.5) | −2.3 (1.6) |
| A2 | 72.3 (9.5) | 94.1 (4.1) | 1.6 (1.4) | −1.3 (1.7) |
| A3 | 69.6 (16.5) | 95.0 (6.5) | 1.4 (1.5) | −1.3 (1.5) |
| ALL | **75.8** (**9.6**) | 95.4 (3.4) | 2.1 (**1.5**) | −2.0 (**1.7**) |
| Wageningen | | | | |
| A1 | 72.8 (7.8) | 95.0 (2.4) | 2.2 (2.0) | 0.4 (3.0) |
| A2 | 71.7 (8.0) | 95.2 (2.4) | 1.3 (1.1) | −0.6 (1.6) |
| A3 | 69.6 (19.9) | 96.1 (5.1) | 1.7 (2.4) | 0.6 (2.9) |
| ALL | 72.2 (10.5) | 95.2 (3.0) | **2.0** (2.0) | **0.3** (2.8) |

Average values are shown for metrics described in Sect. 3 and in parenthesis standard deviation. 'ALL' denotes the average (and standard deviation) among the three datasets for each method. Other shorthands and abbreviations as defined in text (Sects. 3, 6)
The best results for each metric are denoted in bold

## III. Results

This algorithm was developed using an Intel Core i5-4300U CPU with a clock speed of 1.9 GHz, running Windows 10 OS. The Program was written in PyCharm IDE using Python 3.7. The method uses functions from PlantCV ver. 3.3.0 [15] and from OpenCV-python [14] ver. 3.4.5.

Table 1 shows the results of the different algorithms discussed. The result that is the most important for consideration when comparing this papers algorithm is the average absolute value of the difference in count, |DiC|. Testing this papers method on the first 50 images in the A1 dataset produced an average |DiC| of 2.7 with a standard deviation of 2. Although the results would differ when tested on the full A1 data set and the algorithm used a perfect foreground segmentation, the results show that this algorithm is comparable to the described methods. This preliminary result shows that this method is viable as it has a 1.1 |DiC| improvement over the Nottingham method, with both methods having equivalent standard deviations.

### Limitations.

The algorithm described in this paper has limitations. The core limitation of the algorithm stems from the use of the Canny edge detection algorithm. The issue with the use of Canny is that it relies on there being a gradient change between occluded leaves. This gradient change is not always present between occluded leaves and therefore the individual leaves cannot be properly segmented, resulting in a count lower than the absolute truth count. Another issue with using Canny is that it can detect edges that are caused by natural contours within a leaf itself and other noise. This can cause a whole leaf to be split and counted as multiple leaves resulting in a higher leaf count. The final major limitation is that the algorithm was optimized for the lighting conditions, image format, and the species of plant. This optimization was in the form of an empirically found Sigma, low, and high threshold for the Canny algorithm. The optimization was also in the form of an empirically found threshold value used after the distance transform. These specific optimizations limit the method from working on new images or datasets without optimizing the parameters first.

## IV. Conclusion

The method showed results which are comparable to other methods discussed in this paper. With a |DiC| of 2.7 and standard deviation of 2 for the first 50 values in the A1 dataset. These values are within the range of performance values presented by the other methods.

The strength of this method is its simplicity, using only edge detection, distance transform, and morphology algorithms. This contrasts with the other discussed methods of which require training [4] or a more complex set of operations [6] which are harder to interpret.

There are limitations to this method which stem from the edge detection algorithm and the optimization of the algorithms used.

Future research into this method could improve both the functionality and the accuracy. To improve functionality, the optimization process could be streamlined using intuitive controls (i.e. sliders). This would allow changes in algorithm parameters to display in real time, allowing for faster optimization of the method for specific image formats. Accuracy improvements could be made by making changes to the image before applying the method covered in this paper. Such changes could include sharpening to increase the definition of borders between occluded leaves. Another change could be to identify, count, and remove leaves which aren't involved in occlusions before applying the method. This would reduce the likelihood of leaf noise changing the leaf count as described in limitations. Improvements in accuracy could also be achieved by the testing and implementation of different edge detection methods or contour finding methods.

## References


[1] D. Houle, D. Govindaraju and S. Omholt, "Phenomics: the next challenge", Nature Reviews Genetics, vol. 11, no. 12, pp. 855-866, 2010. Available: 10.1038/nrg2897.

[2] B. Romera-Paredes and P. Hilaire Sean Torr, Recurrent Instance Segmentation. Department of Engineering Science, University of Oxford, 2016.

[3] X. Yin, X. Liu, J. Chen and D. Kramer, "Joint Multi-Leaf Segmentation, Alignment, and Tracking for Fluorescence Plant Videos", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 6, pp. 1411-1423, 2018. Available: 10.1109/tpami.2017.2728065.

[4] H. Barrow, J. Tenenbaum, R. Bolles and H. Wolf, "Parametric Correspondence and Chamfer Matching: Two New Techniques for Image Matching", Apps.dtic.mil, 2019. [Online]. Available: https://apps.dtic.mil/docs/citations/ADA458355. [Accessed: 23- Jun- 2019].

[5] H. Scharr et al., "Leaf segmentation in plant phenotyping: a collation study", Machine Vision and Applications, vol. 27, no. 4, pp. 585-606, 2015. Available: 10.1007/s00138-015-0737-3.

[6] J.M. Pape and C. Klukas, "3-D histogram-based segmentation and leaf detection for rosette plants", Computer Vision—ECCV 2014 Workshops, vol. 8928, pp. 61-74, 2015. [Accessed 23 June 2019].

[7] "CVPPP", Plant-phenotyping.org, 2019. [Online]. Available: https://www.plant-phenotyping.org/CVPPP2019. [Accessed: 22- Jun- 2019].

[8] J. Canny, "A Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. -8, no. 6, pp. 679-698, 1986. Available: 10.1109/tpami.1986.4767851.

[9] L. Belaid and W. Mourou, "IMAGE SEGMENTATION: A WATERSHED TRANSFORMATION ALGORITHM", Image Analysis & Stereology, vol. 28, no. 2, p. 93, 2011. Available: 10.5566/ias.v28.p93-102.

[10] R. Kimmel, N. Kiryati and A. Bruckstein, "Sub-pixel distance maps and weighted distance transforms", Journal of Mathematical Imaging and Vision, vol. 6, no. 2-3, pp. 223-233, 1996. Available: 10.1007/bf00119840.

[11] J. Serra, "Introduction to mathematical morphology", Computer Vision, Graphics, and Image Processing, vol. 35, no. 3, pp. 283-305, 1986. Available: 10.1016/0734-189x(86)90002-2.

[12] C. Strachey, "Bitwise operations", Communications of the ACM, vol. 4, no. 3, p. 146, 1961. Available: 10.1145/366199.366254.

[13] L. Abbott, "Learning in neural network memories", Network: Computation in Neural Systems, vol. 1, no. 1, pp. 105-122, 1990. Available: 10.1088/0954-898x/1/1/008.

[14] "OpenCV", Opencv.org, 2019. [Online]. Available: https://opencv.org/. [Accessed: 24- Jun- 2019].

[15] P. Team, "Home - PlantCV", Plantcv.readthedocs.io, 2019. [Online]. Available: https://plantcv.readthedocs.io/en/stable/. [Accessed: 24- Jun- 2019].