University of Salento

Faculty of Engineering

Degree Course in Computer Engineering

Master Thesis

in

Image Processing

Titolo tesi

Supervisor:
Ch.mo Prof. Nome COGNOME
Co-Supervisor:
Ch.mo Prof. Nome COGNOME

Candidate:
Davide Basile
Matricola 20034689

Academic Year 2020/2021

Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis.
Curabitur dictum gravida mauris.

— *Donald Ervin Knuthyu*

Acknowledgments

A conclusione di questo lavoro di tesi ringrazio...

G. M.

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Contents

1 Introduction **1**

2 Related Works **2**
2.1 Plant Phenotyping 2
2.2 Convolutional Neural Networks 3
2.2.1 Image Classification 3
2.2.2 Object Detection 4
2.2.3 Image Segmentation 4
2.3 CNN Applied to Plant Phenotyping 5
2.3.1 Plant Development 5
2.3.2 Plant Stress Phenotyping 5
2.3.3 Plant Counting 6
2.4 Leaf Instance Segmentation 8
2.5 Multi-Task Plant Phenotyping 10
2.6 Fine-Grained Recognition in HTP 11
2.7 Deep Leaf Counting 12
2.8 Leaf Counting with DNN 13
2.9 Weed Growth Estimator with CNN 14
2.10 Drought Stress Classification 15
2.11 CNN Encoder Decoder for Semantic Segmentation 17
2.12 Leaf Counting Without Annotation 19
2.13 Fusing Network Component for Improved Accuracy 19
2.14 Available Dataset 19
2.14.1 3D Phenotyping for Komatsuna dataset 19
2.14.2 Multi-Modality Plant Imagery Database 21

3 Algorithm Design **24**

4 Implementation **25**

5 Experimental Results **26**

6 Conclusion **27**

List of Tables

List of Figures

Chapter 1

Introduction

Chapter 2

Related Works

2.1 Plant Phenotyping

Plant phenotyping has been recognized as a bottleneck for improving the efficiency of breeding programs. In recent years, image-based approaches have shown great potential for high-throughput plant phenotyping, resulting in an increased focus on image-based plant phenotyping. To meet the world's growing demand, current crop productivity must be roughly doubled by 2050, due to global population growth that may exceed 9 billion by 2050 and 11 billion by 2100. Over the past years, various HTP solutions have been developed to improve phenotyping capability and throughput, including tower systems, gantry systems, mobile ground systems, and low and high-altitude aerial systems, one obvious trend has been noted in recent HTP systems, computer vision. Cameras have been used more frequently because of their extensive ability to extract complex features. 2D images can provide spatial information of a scene plus an additional dimension of data such as spectral information. 3D images can provide a 3D structure of a scene that can be used to calculate morphological features of objects. Finally, 2.5D images retain information about the image plane structure, which is similar to that of 2D images, and capture depth information about a scene, which can be used to reconstruct the 3D structure of that scene. one of the first operations to be performed for phenotyping is to identify and define the phenotypic traits to be measured, which largely determine the use of appropriate imaging modalities for plant detection. Measuring phenotypic traits usually requires one or more computer vision tasks that can be solved by developing new algorithms or improving existing ones through conventional image/signal processing, machine learning, or a combination of them. Conventional machine-learning based approaches have generally improved generalizability, but most still cannot meet the requirements for current phenotypic purposes. Deep learning is a subset of machine learning and enables hierarchical learning of data. The key benefit is that features will be learned automatically from the input data, thus

breaking down barriers to developing intelligent solutions for different of applications. A commonly used DL architecture is deep convolutional neural networks(CNNs), that were developed for image classification and eventually demonstrated better performance than humans on the same dataset.

2.2 Convolutional Neural Networks

Most imaging-based phenotyping applications essentially require solutions for one or more tasks related to image classification, object detection, and segmentation.

2.2.1 Image Classification

Image classification is one of the main tasks of computer vision and aims to assign images with predefined class labels. The development of modern CNNs for image classification can be grouped into three phases:

- emergence of modern CNNs;
- intensive development and improvement of the CNN architecture;
- reinforcement learning for the design of the CNN architecture.

After these improvements, representative CNNs can now usually outperform humans in image classification for various datasets. It should be noted that the performance improvement following the change in CNN architecture was highly dependent on human expertise and tuning efforts, which means that improving the CNN architecture could be as laborious as feature engineering in traditional ML. A reinforcement learning framework was introduced to search for optimal convolutional cells on a small annotated dataset, and the resulting cells were stacked in different ways and transferred to an unknown large dataset. In addition to performance improvement, studies have been conducted to understand the mechanism of CNNs. This leads to the development of techniques towards explainable artificial intelligence that helps in developing interpretable and inclusive machine learning models. The study also showed that learned features could be generalized to various classifiers, suggesting that CNNs could learn general representations of images rather than specific features for classification. Subsequent studies have continued in this direction and developed various gradient-based methods that can visualize the importance of features for classification results. Commonly used methods include guided backpropagation, gradient-weighted class activation mapping, and layered relevance propagation.

2.2.2 Object Detection

Object detection attempts to detect and classify all potential objects in a given image. The use of CNNs for object detection can be categorized into two groups: one-stage and two-stage CNN architecture. Two-stage models first detect candidate object regions and then classify the candidate regions into different object categories. The OverFeat framework was developed to use a single CNN to extract features for training classifiers and regressors separately. The trained classifiers and regressors were used to predict class labels and bounding box coordinates, respectively, for candidate region of interest generated using a sliding window method.

Three key techniques in the CNN architecture of the RCNN family were identified, including the region proposal network, the region-of-interest pooling operation, and the multitask loss function. A RPN was developed to generate ROIs of candidate objects using features extracted from CNNs, which simultaneously saved processing time and increased the accuracy of the region proposal. A ROI pooling operation was developed to extract a fixed number of features from ROIs of various sizes, thus avoiding repeated computation of features for different ROIs. A multitask loss function was used to unify the training process, which allowed for a training process for object detection.

Representative one-stage models include the you-onlylook-once family and the single-shot detector framework. A critical problem, however, has been discovered for these one-stage models: an extreme imbalance in the number of object and background regions. Most image regions contain only background information, providing little contribution to the model training process. However, if detection accuracy is the most important factor to consider, two-stage models would be the option; otherwise, single-stage models provide better computational efficiency for embedded systems and real-time applications.

2.2.3 Image Segmentation

Semantic segmentation seeks to provide masks for objects with the same semantic meaning, while instance segmentation seeks to provide individual objects in a given image. These in general can be divided into two categories: those encoder-decoder based, and those based on detection. The encoder-decoder-based is divided into two phases. The encoding phase uses CNNs to extract semantically meaningful feature maps from the input images, and the decoding phase uses transposed convolution for upsampling the extracted feature maps to labels. A detection-based framework is based on the CNN architecture for object detection. Several studies have explored the use of object detection models for instance segmentation, including concurrent detection and segmentation based on RCNN and DeepMask based on Faster RCNN.

2.3 CNN Applied to Plant Phenotyping

2.3.1 Plant Development

Morphological changes in plant shoots are critical in describing plant development. Canopy cover and leaf area are two commonly used parameters to quantify plant growth and development. In order to segment plants many studies used color-based features, but usually had imperfect segmentation because plant color could have large variations due to illumination, shade, occlusion, and so forth. Most studies conducted on CNN in plant phenotyping consider it a semantic segmentation task and use an encoder-decoder-based CNN architecture for processing. Although these studies have shown improved segmentation accuracy, annotation of training data can be extremely difficult. To solve this problem, one study attempted to generate synthetic images with semantic annotations automatically for training the CNN model. The combination of synthetic and real images would improve the generalizability of CNNs for plant segmentation and thus the accuracy of growth analysis. The researchers also combined CNNs with other DL methods for characterizing plant development. CNNs were used as a feature extractor to encode the spatial state of plants at individual growth stages, and RNNs were used to incorporate all spatial encodings to learn the temporal changes of plants. In this way, plant growth patterns could be fully encoded by neural networks to reveal differences between cultivars and treatment groups. This indirect phenotyping scheme could be particularly useful for selection-oriented programs, but explaining selection would be a significant challenge and barrier to many research studies that aim to understand the mechanism of many plant responses. In addition to morphological measurements, CNNs could be used to monitor some plant development events such as plant lodging. A new CNN architecture (LodgeNet) was developed by integrating a custom 7-layer CNN model with handcrafted features. Compared with 10 well-established CNN architectures, LodgeNet provided comparable or better performance on differentiation between allurement and regular plots, but with a significant improvement in processing speed.

2.3.2 Plant Stress Phenotyping

Plant stress phenotyping aims to identify and evaluate plant responses to abiotic and biotic stresses, providing information for selection of accession lines with high stress resistance and tolerance in breeding programs and understanding of intrinsic mechanisms in genetics/genomics studies. Plant stress phenotyping can be categorized into four stages:

- identification (presence of stress);
- classification (type of stress);
- quantification (severity of stress);

- prediction (possibility of stress occurrence).

The development of image classification-based approaches can be divided into two phases. In the first phase, studies have intensively investigated known and customized CNN architectures because of the availability of annotated datasets and the simplicity of implementing and training CNN for image classification. Data annotation for image classification is also relatively easy, so a large number of images in a newly collected dataset can be annotated in reasonable time and cost, especially when a proper data collection procedure is used. As a result, studies related to plant stress detection typically have a sufficient number of images annotated for model training. In the second phase, pioneering studies sought to understand the reasons leading to the high performance of CNNs for stress identification and classification, because understanding would not only help to improve CNNs but also ensure the biological correctness of the results obtained. Although some studies have adopted deconvolution layers to visualize activated pixels in different convolution layers, the visualization results have not been used to compare with human evaluation or correlate with biological knowledge. Compared with the first phase studies, the two pioneering studies demonstrated the importance of understanding the mechanism of CNNs for stress phenotyping, as well as the potential for quantifying stress severity. Image annotation is still recognized as a limiting factor for the use of many DL algorithms, so the researchers investigated the use of generative adversarial networks to generate synthetic images for training CNN models for plant stress detection and classification. A very recent study explored the use of a custom CNN architecture to detect plant diseases in hyperspectral images. The novelty of the custom architecture is the use of a 3D convolution operation that can directly convolve spatial and spectral information into hypercubes. This would not only inspire future studies related to plant stresses, but also allow the reanalysis of many previous hyperspectral images collected for plant stress analysis. With improved detection accuracy, subtle differences in stress between cultivars/treatments can be revealed to improve our understanding of plant responses to stresses.

2.3.3 Plant Counting

Counting plants and plant organs is critical to characterizing plant development. Regression or image classification is the simplest and most straightforward way for fruit/organ counting from the perspective of technical development. For regression-based methods, a major modification was made that replaced the softmax layer of a CNN with a single neuron for regression of numerical values. This simple end-to-end counting solution provided high accuracy for counting fruit and leaves from plants. A particular challenge of regression-based solutions is the limited availability of annotated images, which leads to many potential problems such as poor generalizability of the model. To address this issue, one study attempted to generate synthetic data of tomatoes to increase data availability and diversity. Although the trained

CNNs achieved 91% counting accuracy on real images, the study tested only red tomatoes, which have distinctive color characteristics from the background. The generalizability of this approach should be further validated for challenging situations such as detecting green tomatoes from leaves. GANs were also used to generate synthetic data for model training.

An alternative approach was to use patch-based training. TasselNet was developed to count corn tassels in two phases. In the first phase, a local CNN regression model was established to predict the number of tassels in each patch of an image. In the second step, the estimated count in each image patch was averaged over the individual pixels in that patch to create a count map with the same spatial dimension as the original image. The sum of all pixel intensities in the count map represents the final count of tassels in that image. Experimental results showed that TasselNet achieved counting accuracies from 74.8% to 97.1%, which were 2 to 5 times higher than those of conventional methods. TasselNet uses the patch-based training method, which substantially increases the number of training images. For classification-based methods, plant/organ counts were treated as a discrete counting problem and, therefore, a predefined score or grade was assigned to a given image rather than an exact count. An example of a classification-based method is WheatNet, which was developed to predict the percentage of flowering in wheat images. Multiple images were acquired for each plot. A total of 11 classes were annotated for each plot, corresponding to 11 visual scores with a percentage header from 0 to 100% with a 10% interval. The average prediction of all images in a plot was the final percent header for that plot, which reduced counting errors due to inaccurate classification.

Object detection is an intuitive approach to counting plants and plant organs in still images: accurate object detection ensures accurate object counts. DeepFruits was the first study to explore the use of modern CNN architecture for fruit detection. Several key contributions were recognized in this study. First, transfer learning was used to train a Faster RCNN model with 100 labeled images, demonstrating the potential of using limited labeled images to train the CNN architecture. Second, when using RGB images, the trained Faster RCNN model provided a 1% improvement in F1 score over that of the CRF model. Third, data fusion was conducted at the level of the raw data and at the decision level for the Faster RCNN models. A custom two-step framework was proposed using the superpixels generated by the simple iterative linear clustering algorithm as proposed regions. A CNN model was used to classify each superpixel as a flower or a non-flower object. While this approach showed higher performance than conventional ML methods, it has a potential limitation in region proposal. The advantage of the end-to-end CNN architecture is that they are able to use richer features for accurate localization, especially when the images vary greatly. However, superpixels are subject to image variation and may not provide optimal region proposals. The generalizability of this approach, therefore, is most likely lower than that of end-to-end methods. Many studies have also investigated semantic segmentation-based approaches for plant/plant organ counting. CNN architectures for semantic

segmentation were first used to obtain plant/plant-organ masks. Then, the obtained masks were post-processed using conventional computer vision methods to isolate individual plants/plant organs so that the objects could be counted. An obvious concern is that although CNNs could provide accurate semantic masks, counting accuracy may still suffer from inaccurate post-processing. To address this concern, studies have explored the use of instance segmentation CNNs that can directly segment individual objects in images. These studies faced the same challenge in the lack of training data. To overcome this limitation, most of these studies developed algorithms to generate synthetic images for model training. Two types of image synthesizing methods have been proposed: rule-based and GAN. Rule-based methods use a predefined leaf model to generate a plant based on predefined growth rules. GAN-based approaches,however, could generate synthetic images without sacrificing leaf structure. Thus, a method combining rule-based and GAN-based methods was developed for image synthesizing.

2.4 Leaf Instance Segmentation

In this paper, the authors aim to implement a novel strategy to exploit the global feature of invariant leaf shapes across plant species, and create an automatic generation of leaf patterns for incorporating leaf shape knowledge. Thus, the proposed framework enables the automatic analysis of different plant species with rich leaf shape variation, thus alleviating the current bottleneck of annotated data.

They present a novel segmentation of the leaf instance with a top-down view of plant images in high-throughput phenotyping platforms. The acquired images suffer from uneven illumination, moss, and shadows. To address the problem of scene variability, a statistical image segmentation algorithm Meanshift Bandwidths Searching Latent Dirichlet Allocation is employed. In order to find the optimal MeanShift bandwidth for stable segmentation, the authors used LUV distances between topics. However, the color of moss is very similar to plant leaves, so using LUV distance is not suitable for the studied images. To solve this problem, the authors employed the distance between texture descriptors. Plant leaves spread out from the center showing different orientations, so finding candidate regions to leaves will require examining different combinations of orientations and widths in the RGB image. To constrain the search for possible leaf candidate regions, first the segmented image is transformed to polar coordinates. The center and radius required for polar transformations are calculated based on its skeletal image. The center is extracted based on the betweenness centrality of the nodes in the skeleton image. The goal of nominating leaf candidates is to extract leaf representatives that can be processed by the model generation step. Their algorithm represents the image as a graph with a fixed number of nodes and neighboring nodes connected by weighted edges. The seed extraction strategy based on the hierarchical representation of slices and its use in the random walk results in an

under-segmentation of the plant image, this is due to the connection property of vertical slices. However, this property also allows for the optimal segmentation of non-occluded leaves that exist as disconnected regions in the plant image. Given the output of the random path instance, they automatically isolate the non-occluded leaves by exploiting the symmetry property of the leaves around the medial axis. Two classes of leaf and non-leaf candidates are generated, then, multiple leaf candidates are removed from the segmented image to obtain a modified mask. If the modified mask belongs to a small area of the input segmented mask, then these leaf candidates are the final instances of the leaves extracted from the proposed picture. Otherwise, these candidates form a part of the output instance and the modified mask is used as input for the multi-leaf matching model. Leaf templates have a great influence on matching multi-leaf templates. Thus, to generate templates that accommodate large variations in the shape and size of the leaves of a plant species in its growth cycle. A post-processing pipeline is used to remove stems based on the Hough transform. The Hough transform generates a parameter space, where each pixel in an x-y image space is mapped to a line of an m-c parameter space and the peaks represent the lines. Since the canonical leaf shape is used, peaks with m centered around zero degrees correspond to stems. Since the multi-leaf template matching algorithm is sensitive to rotation and scale, the representative shapes are varied using geometric transformations such as scaling and rotation. This database of templates with different shapes, sizes, and orientations, generated through real-time image sampling, is employed in multi-leaf template matching. For each pattern, the algorithm first computes the position of the minimum matching Chamfer, then, an optimal set of leaf candidates based on an objective function with three terms:

- average CM of the selected leaves that promotes the selection of leaf patterns with the minimum CM distance;
- the number of estimated leaves selects an optimal number of leaf patterns;
- the distance of the synthesized mask from the selected leaves and the target image mask is selected and that when disposed generates the target mask accurately.

Overall, they observed that the output of the Gaussian mixture model appears noisy, while K-means, Otsu's threshold based on excess green features, and Minervini's algorithm excessively segment the foreground resulting in a disconnected vegetation region. On the other hand, significant agreement with ground truth can be observed for the modified MSBS-LDA algorithm using texture-based distance metrics. One approach was based on the orientation field map and leaf tips of the polar transformed image. For leaf tip detection, they chose the Harris-Stephens angle detector. The second approach was based on the skeletal image of the polar transformed image. Here, the path from the end nodes of the image is traversed and a minimum deviated direction is employed at the junctions. Based on the quantitative result, it was observed that

due to the oversegmentation of the non-overlapping leaves, the instance output failed the symmetry threshold, however, the instances of the used approaches that labeled as leaf candidates showed good agreement with the corresponding non-overlapping leaves of the ground truth. It should be noted that the number of leaf candidates extracted plays an important role in the extraction of representative shapes, especially at later growth stages. Based on the database of plant phenotyping images, for early and middle growth stage images, multi-leaf pattern matching was not frequently used because the generated leaf candidates comprised 90% of the input segmented mask. This was due to the non-overlapping scenarios observed in these images. However, it also led to the neglect of leaves that were highly. The comparative study demonstrates the efficiency of our proposed structure in relation to different plant species. The strategy to extract non-occluded leaves involves the automatic generation of shape priors, which can be further used to segment individual leaves into overlapping regions in the image.

2.5 Multi-Task Plant Phenotyping

The main discussion of this paper is the ACID dataset consisting of a doubled haploid population of spring wheat plants was obtained from the Nottingham/BBSRC Wheat Research Centre and grown in 2l pots in a greenhouse. Imaging was conducted using a 12MP consumer grade camera attached to a custom built imaging system that provides a consistent black background containing multiple high resolution peaks. Instead of predicting location directly, the authors performs a pixel-wise regression, identifying the high probability areas of each target. Heatmap regression has been successfully used in, among other things human pose estimation. The network architecture is based on an encoding/decoding framework, in which a series of convolutional and spatial downsampling operations begin by computing a fixed-dimensional feature representation of the image. This feature space is then resampled to the original resolution. During training, random augmentation was applied with random rotation and scaling extracted from normal distributions with standard deviations. The network was trained end-to-end, from scratch, using RMSProp. The authors used a mean square error loss function with an initial learning rate of $2.5x10^{-4}$, and reduced by a factor of 10 every 200 epochs. Spike and spikelet positions are computed from each output heatmap using non-maximum suppression. For all output pixels, any pixel with an intensity greater than its four neighbors is classified as a feature. The number of additional false positives generated by the non-maximum suppression component of their approach is negligible. Accuracy represents the fraction of detections that are true positives. They applied a distance threshold for successful detection, then varied this threshold to explore the effectiveness of the approach at different tolerances. A true positive for a spike or spikelet is any predicted location that is within this normalized distance of a ground truth point. Similarly, a false negative is any ground truth point that is not within the normalized distance

of a predicted feature. To measure counting accuracy, they compared only the number of predicted features against the number of ground truth points and calculated a percentage error, to simulate a counting-based phenotyping task. Given the lower recall performance on the tips for the 256-pixel input size, one should expect a higher error. they also found a negative value for the error of the tips indicates that the network tends to underestimate, rather than overestimate the number of spikes.

2.6 Fine-Grained Recognition in HTP

In this paper, the authors pre-process the images with object localization directly based on prior knowledge such as sensor fusion and experiment setup, object segmentation using green-based brightness thresholds or Otsu threshold, and finally contour extraction as an array of contour pixel coordinates. Then we unfold the 2-dimensional contour into a 1-dimensional radial object descriptor, which indicates the shortest distance from each pixel to the seed. Then the authors post-process these results, where they normalize the values in ROD in a range of [0,1] and scale them from different lengths into a fixed by sampling or interpolation. To complete the ROD feature they merge them with another low-level local feature descriptor, the histogram of oriented gradients, which have local gradients at different orientations, which is powerful for differentiating objects with meaningful texture patterns. The histograms of neighboring cells are further grouped into block groups with an overlap ratio to enhance local contrast. Another way the authors used to integrate ROD is ResNet-18 which is a type of CNN with residual parts to skip layers so that higher accuracy can be efficiently achieved by using deeper layers. It is the representative of deep feature extraction methods because its performance has been demonstrated in plant research. To classify and calculate the loss function, the authors used Softmax regression, and learned the set of weight parameters by minimizing the loss function. The first experiment involves soybean plots grown in the field using UAS imagery, where we highlight the robustness of our approach under changing camera angle of view. The second experiment involves rosette plants in the greenhouse using a fixed camera, where we highlight the robustness of our approach under sus change in growth phase. They evaluated the robustness of ROD and its fusion with HOG and ResNet in Softmax using seven different approaches. The first two approaches, CNN and ResNet-18, are based on CNN, Each convolution layer is followed by a batch normalization layer, a ReLu layer, and a max pooling layer. For both CNN and ResNet-18, they manually tuned the mini-batch size and learning rate based on each validation dataset before finalizing its classification result. By combining all these layers, they obtained five models, which will be trained using the Scaled Conjugate Gradient method. Using the HTP-Soy dataset, the authors found that ROD-ResNet-Softmax achieves the highest validation mA of 0.947 while ROD-HOG-Softmax achieves the highest test mA of 0.866, which slightly exceeds ROD-ResNet-Softmax. ROD-Softmax significantly out-

performs Fourier-Softmax, which indicates the advantage of ROD over Fourier as a contour-based mid-level descriptor. It is even surprising to see that ROD-Softmax achieves mA comparable to HOG-Softmax even though HOG contains much richer information. ROD-HOG-Softmax outperforms ResNet-18 in inference of soybean plots for most pairs of growth stages. The advantage of ROD-HOG-Softmax becomes more apparent during the later growth stages, where the soybean canopy grows faster and its contour becomes more discriminating. In contrast, in Arabidopsis, Bean, and Komatsuna, the authors found that ROD-ResNet-Softmax outperforms almost all other approaches in terms of both validation mA and test mA. ROD-Softmax significantly outperforms Fourier-Softmax and achieves classification accuracy comparable to HOG-Softmax.

2.7 Deep Leaf Counting

In this paper, the authors modified the ResNet50 residual neural network to learn a leaf counter by taking an RGB top-view image of a rosette plant as input. They used this network for its ability to generalize, which was crucial for this challenge due to its "in the wild" setting, as well as its speed of training and convergence. The ResNet architecture is easier to optimize than other deep networks and addresses the degradation problem present in very deep networks, which states that as deep networks converge, the accuracy becomes saturated. The network is modified by removing the last layer intended for classification, flattening the network, and adding two fully connected layers FC1 and FC2 of 1024 and 512 nodes respectively followed by ReLU activations. We apply an L2 activation regularization on layer FC2 to penalize the layer activity during training and prevent overfitting. For preprocessing, each image was resized to be 320x320x3 pixels and a histogram stretch was applied on all images to improve contrast as some images were darker than others. To evaluate our architecture, we first trained the network on each of the CVPPP datasets individually. We then added more data by combining the datasets together and finally a combination of all four datasets. Cross-validation was performed four times on training images sampled differently during training. Data augmentation was performed during training of all models. We used a generator that assigns training images a random affine transformation from a pool of random rotations from 0-170 degrees, zooming between 0-10% of the total image size, and vertical or horizontal flipping. The authors found that tuning the Resnet50 network, pretrained on the ImageNet dataset, yielded better and more consistent results than providing stronger annotations and using random initialization. First, the network can work with images of different sizes and scales present in each dataset. Second, our model was able to learn to count leaves of different shapes, sizes, and orientations provided with only minimal annotations. Data labeling is increasingly time and resource consuming when moving from weak to strong annotations, which is one reason for the relative lack of publicly available plant phenotype datasets. Finally,

the datasets provided contained a limited amount of images compared to the amounts of data traditionally used for deep learning models. The results show that combining datasets from different sources and even different species is beneficial, as it improves test accuracy for all datasets and more generally for all evaluation metrics. The worst performance is seen in networks that are trained on only a single dataset. Combining any two datasets produces similar results.

2.8 Leaf Counting with DNN

For this paper, the authors studied an architecture used for segmentation and counting are trained separately, but not independently, since the binary mask generated by the segmentation model is used to train the counting model along with the RGB channels. The idea of deconvolutionary networks is to construct a compact and informative set of feature maps or vectors from a set of input images. Usually, the design of a deconvolutionary network contains fully connected layers in between to generate the feature vector from the clustered feature maps. However, we propose that the features in the semi-global context should be sufficient to segment the leaf regions from the background in the color images, and thus the FC layers could be omitted for our application. One advantage of eliminating FC layers is that it greatly reduces the number of trainable parameters without sacrificing performance. For these reasons, we adopt the SegNet architecture. Instead, the convolutional front-end substructure of the network is the VGG architecture with batch normalization followed by each convolutional layer. In the convolutional front-end of SegNet, there are five 2×2 pooling operations with zero overlap following multiple convolutional layers and rectification each time. Then, the convolutional feature maps are compressed 32 times before starting decompression through the deconvolutional back-end. The logic behind providing the counting module with the segmentation mask and the original RGB image instead of providing the segmented region in the RGB image or the binary mask alone. Although the segmentation results generated by SegNet are sufficiently accurate for the counting step for many images in the dataset, our network generates spurious segmentations for some of them. The design of our regression leaf counting network takes inspiration from the VGG architecture, which reinforces the idea of deeper architectures with a long list of convolutional and rectification layers stacked one after the other with several pooling layers in between and then the classification layer follows a pair of fully connected layers. The authors trained their model from scratch without using any pre-trained weights for initialization. Also in SegNet, the authors used different learning rates for different modules, while a fixed learning rate was used for all layers in their training. They augmented the data and trained the network in 3 steps. First, for each image, we extracted the union of the first 20 object proposals, flipped from top to bottom and left to right, rotated with an angular step of 4 degrees, cropped the largest square from the central position to avoid dark regions due

to rotation, and created a pair of blurred versions with Gaussian and the corresponding sharp images. Second, they took the proposed images and their flipped versions and generated nearly 0.3M subsamples of size 224×224 deterministically with a fixed step and train the network for 8 more epochs. Finally, we generated another 0.19M samples in a similar manner to the second step, but this time from the original images instead of the proposals. Spatial cross-entropy was used as an error criterion. The ratio of foreground to background weights in the cross-entropy calculation for the first step was 2.0 and 1.2 for the later steps. In the test phase, we took dense 224 × 224 samples deterministically with fixed step from each of the test images and classified each pixel according to the aggregate probability on the samples. During the design of the network architecture, we experimented with adaptive operations to handle images of varying sizes, but they did not seem to work any better than resizing the images to a fixed size. In addition, we had to be cautious in choosing the size for the resize operation so that for larger images with resolution such as 2000 × 2500, the properties of the small leaf regions would not deteriorate much. Considering this fact, we chose the size of the modified image to be 448×448 while preserving the aspect ratio. After performing the resizing operation, each of the images was augmented 8 times using intensity saturation, Gaussian blur and sharpening, and additive Gaussian noise. Each image was also flipped from top to bottom and left to right and rotated. After data generation, the counting or regression network was trained for 40 epochs using Adam with a fixed learning rate. The authors found that their predictions can be accurate for easier samples with no leaf overlap or moderate leaf size or both, but deteriorate for more difficult cases with smaller or overlapping leaves. In this sense, our generalized framework is able to model and infer leaf shapes under deformation and partial occlusion better than their main competitor given a few hundred images for a particular species.

2.9 Weed Growth Estimator with CNN

In this research, a CNN was used to count the number of leaves on 18 different weed species or families. However, deep CNNs need to be trained on a large number of images to learn how to automatically extract general features from the input data. Overfitting makes it difficult for the network to generalize to new examples that were not in the training set. In addition, the training dataset was augmented by using horizontal tilt, rotation, zoom, width shift, height shift, and Gaussian smoothing filters. The authors selected the Inception-v3 architecture, which is a refinement of the GoogLeNet architecture. The Inception-v3 architecture was chosen because of its good performance, ease of implementation, and relatively low computational cost, which yielded excellent results. This network with their dataset was trained for 20 epochs and the predictions from the 20 models were combined to increase the confidence in the predictions. Because the 20 models were not identical, they could predict different growth stages for the same plants. Predictions of

the growth stage of the same plant by different models were more different when more plants were present in an image, resulting in a higher standard deviation for those images. In order to improve overall accuracy, the softmax outputs from all 20 models were aggregated, and the maximum value index was used as a sample class prediction. The accuracy results were 83%. Count-Diff is the average forecast bias, which is 0.07 leaves. This indicates that the model has a small tendency to overestimate the number of leaves. The Abs. CountDiff shows that on average, the model is off by 0.51 leaves, and for 70% of the samples, it is off by zeroThus, according to the obtained results, it can be concluded that the developed model can be implemented with 70% accuracy on variable field machines, including weed control machine; because in these systems, the amount of toxic material applied to the weed in neighboring field classes does not differ much. Confidence in the estimated accuracy was calculated using Wilson confidence intervals over 10,000 iterations. The best network performance was obtained for some weed species, probably due to their large number of training samples and shared physical properties, for the others the average accuracy was less than 60% due to fewer training images. Therefore, this study presents a convolutional neural network-based method to estimate the growth stage in terms of leaf number of various weed species. Images from various camera models were collected in fields with different soil types and light conditions. Because the images were collected under field conditions, plants often overlapped each other, which this network was typically able to overcome.

2.10 Drought Stress Classification

The authors modify the standard SfM pipeline as proposed in to use learned keypoints and descriptors based on deep networks. They employed Learned Invariant Feature Transform (LIFT) for keypoint detector and descriptor learning instead of Scale-Invariant Feature Transform. The key points in LIFT correspond to distinctive regions, where the conditions that define distinctiveness are learned with a Siamese deep network on the dataset for the target domain. Thus, the authors selected LIFT for its ability to learn key points and the corresponding descriptor based on the specific characteristics of the dataset. Learning descriptors and patch detectors based on deep networks requires training the detector-describer with specific examples. This allows the network to adapt to the complexities of plant structure, especially curvature and color variation. Therefore they perform LIFT detector-orientation estimator-descriptor pipeline training with a leaf matching dataset. For segmentation, they used and Voxel Cloud Connectivity Segmentation to segment the plant canopy from the reconstructed point cloud. It works directly on 3D point clouds. While leaf segmentation in 2D images is an active area of research, due to the availability of depth information the problem in 3D can be looked at from a pure computer vision perspective. The method consists of converting the input point cloud into a voxelized point cloud and construct an

adjacency graph. To construct the adjacency graph, a grid of voxels is formed and seeded voxels are selected and initialized. Isolated voxels are filtered considering a small search volume around the seeded voxels. The voxels are then clustered, conditioned by the smallest gradient within the search volume. The clusters are further aggregated into supervoxels by comparing a feature vector 39 a dimensional feature vector derived using XYZ, RGB etc. The authors considered two types of descriptors Local Descriptors and Deep Descriptors. For the former, they evaluated and Signature of Histograms of Oriented Gradients, Rotational Projection Statistics, and Fast Point Feature Histograms. The applicability and comparison of these descriptors in characterizing various plant-related tasks, and in particular the identification of drought stress, have not been studied before. The main advantage of local descriptors lies in their ability to encode geometric properties of the model. This feature makes them suitable for quantifying various phenotypic traits involving structural changes. For the second, there are two types of deep architectures to process 3 data:

- 3D Convolutional Neural Networks (3D-CNN);
- PointNet.

Due to the inherent nature of the convolution operation, 3D-CNNs work on structured data, i.e., voxelised clouds. However, PointNet is a recent architecture that works directly on unstructured 3D PoCloud data. The voxelization of the point cloud introduces an approximation to the model as it is essentially a quantization process. PointNet generates a global feature on the point cloud point cloud. This is done by learning a permutation invariant representation of points from the input point cloud that is encoded into a vector using a symmetric function. However, by design, the global feature produced by PointNet does not capture local geometric information. We observed that this is the reason for the relatively poor performance of PointNet global features trained on traditional objects. To overcome this limitation, we aggregate the local and global information in a manner similar to PointNet's segmentation network, i.e., the global descriptor is fed back into the network along with the keypoint descriptor to generate a more robust keypoint descriptor. Next, we quantize the resulting local descriptors for the detected keypoints on the point cloud and concatenate them with the global descriptor. The pipeline starts by extracting 3D features from the segmented 3D point cloud. This is followed by a training and testing phase. It should be noted that the number of key points is different for each point cloud. Therefore, during the training phase, they learned a quantized representation of the 3D features. Quantization is necessary to obtain a single uniform length descriptor for each point cloud. In this work, we directly quantize the features using Fisher Vector. As will be evident in the experimental section, Fisher Vectors are capable of discriminatively encoding features in point clouds. Since the size of the codebook from Fisher Vector is small, we use a linear classifier, Support Vector Machine for further classification. This approach allows us to reduce the overall computational complexity while maintaining robustness. For PointNet, we report results on

both the pre-trained model and after tuning PointNet with 3D Point Cloud models from the training dataset. The fine-tuning is performed by initializing the weights from the pre-trained PointNet for object classification and then continuing the training process with the 3D point cloud of the wheat plants. The reconstruction of such fine details can be attributed to the quality of matches on the leaf surfaces from the key points and descriptors learned for 3D reconstruction. . Accuracy is calculated as the percentage of the number of correct classifications over the total number of test inputs for the respective drought and health classes. However, it is interesting to note that PointNet pre-trained on rigid objects performs poorly against all compared descriptors. This could be due to two reasons: the default architecture of PointNet is not easily generalized, and plants have a smooth, textureless surface and are usually heavily occluded, which is not usually found in rigid bodies. In addition, RoPS outperforms all other local descriptors followed by SHOT (FV). It can be seen that descriptors quantified with Fisher Vector consistently perform better than the corresponding coding with Bag of Visual Words with an average gap of 1.6%. This shows that as with 2D descriptors, Fisher Vector is able to encode a more discriminative representation of local descriptors than compared to compared to BoVW. Therefore, in aggregating features in fine-tuned PointNet, Fisher Vector was used as a feature quantization technique.

2.11 CNN Encoder Decoder for Semantic Segmentation

The authors in this paper analyzed, SegNet models based on CNN, U-Net and residual U-Net are used for semantic segmentation of plant images. For data images are involved with scaling, normalization and data augmentation. image pixel values are normalized between 0 and 1 by dividing each pixel with 255. Normalization ensures that each pixel has a similar distribution which helps fast convergence in the process of training deep neural networks. SegNet is an encoder-decoder type model with 13 convolutional layers in the encoder/-contraction and decoder/expansion part. Two fully connected layers are used between the encoding and decoding networks, as shown in. Each encoder in the encoding network performs 3×3 convolution with a filter bank to produce a set of feature maps. Batch normalization is performed after each convolution layer followed by a rectified linear unit. There is a decoding layer corresponding to each encoding layer. At each decoding layer, the number of feature channels is reduced by half. The decoder samples its input feature map by a factor of 2. These feature maps are applied by repeated 3×3 convolution with multiple filter banks to produce dense feature maps. After each convolution layer, a batch normalization is used along with the ReLU activation function. The output with the multidimensional feature map is finally applied to the sigmoid function since we want to separate the plant pixels from the background, which

is a two-class classification problem. The U-Net architecture includes an encoding network and a decoding network. The encoding network resembles the convolutional neural network. The encoding network includes repeated implementations of two 3×3 convolutions, rectified linear unit activation function, and a 2×2 max pooling with step 2 for down sampling. Each down sampling step doubles the number of feature channels. Max pooling helps capture a large context of input images with a compact feature map. Decoding helps to obtain more complex features at the loss of location information. At the final layer each 64-dimensional feature vector is mapped to the desired number of classes using 1×1 convolutions. Finally, the sigmoid function is used to classify image pixels as plant and non-plant pixels. In total, the network has 23 convolution layers. The modified residual U-Net is constructed using residual blocks and has a similar structure to the U-Net. The residual blocks make training easy, while the skipped connections help the information flow without degradation. The combination of batch normalization layer, ReLU activation, and convolution layer is repeated twice in each residual block. At each coding layer, instead of max-pooling, a stride of 2 is used for the first convolution operation to down-sample the feature map by 2. In the decoder, before each residual block, the lower level feature maps are up-sampled and combined with the feature maps from the corresponding coding blocks. After the last decoding layer, a 1×1 convolution and a sigmoid activation function is used for the segmentation task. Data augmentation is used to increase the variations in the dataset. The ReLU activation function is used in both networks in all layers except the output layer, where the sigmoid activation function is used. In this paper, a total of 8 evaluation metrics are used to analyze and compare the performance of residual U-Net with SegNet, U-Net and existing algorithms available in the literature. The metrics used for the analysis are dice coefficient, precision, recall, f1-measure, specificity, negative predictive value, mean absolute error, and accuracy. In these equations, the true positives are the correctly predicted plant pixels, the false positives are the predicted plant pixels that actually belong to the background, the false negatives are the plant pixels present in the ground truth but not predicted by the algorithm, and the true negatives are the correctly predicted background pixels. Although the LSC and fig datasets are completely different, residual U-Net and simple U-Net work well on both datasets. Dice coefficient is one of the important metrics for image segmentation evaluation as it provides the spatial overlap between the ground truth and the output segmented image. All three networks have better segmentation performance for the LSC dataset than the Fig dataset also in terms of other metrics such as precision, recall, F1-score. The output masks predicted by U-Net and UNet residuals closely match the truth images, while the masks predicted by SegNet have subtle changes that can be observed through visual inspection. These results show that the UNet and U-Net residuals are able to draw accurate boundaries for plant leaves and perform better than SegNet in overlapping leaf segmentation. From these results, we can conclude that, the residues of U-Net and U-Net have comparable segmentation results segmentation results while both networks (U-Net

residual and U-Net) perform significantly better than SegNet in predicting leaf shape and boundary areas of fig plants. U-Net and U-Net residual have better segmentation results than the state-of-the-art model for fig plant segmentation, while SegNet achieves comparable performance. Although there is a less significant difference in segmentation accuracy values, other metrics such as specificity, precision, recall, NPV, and F1 score are significantly higher for SegNet, U-Net, and residual U-Net than the state-of-the-art. Since U-Net residual and U-Net have better performance, both these networks are also tested on 10 high-resolution images of the whole fig field provided in the fig dataset. From the above results and discussion, it is clear that U-Net residual achieves better performance than other methods for the LSC dataset and comparable results with U-Net for the fig dataset. U-Net residual achieves this segmentation performance with about half of the trainable parameters compared to SegNet or U-Net. As we can see in Table 6, U-Net residual has the lowest, about 15.3 million trainable parameters followed by U-Net with 31.03 million and SegNet with 33.3 million parameters. This indicates that residual U-Net, although computationally very efficient, achieves better results than SegNet and comparable results to U-Net.

2.12 Leaf Counting Without Annotation

2.13 Fusing Network Component for Improved Accuracy

2.14 Available Dataset

2.14.1 3D Phenotyping for Komatsuna dataset

In this paper, they built a platform to capture overhead views of a plant and measure the environmental conditions around it. They first selected a plant species, Komatsuna, which is a Japanese mustard spinach and a leafy vegetable. It can be grown indoors, and is known to be resistant to pests and grows very fast. Hydroponic culture is an alternative method.It is clean, irrigation and fertilization are automated so that plant growth is accelerated, fewer agricultural chemicals are needed than soil culture, and replanting failures do not normally occur. Another advantage is that plant root systems can also be measured because roots growing in water can be captured through cameras. For lighting, they used LED lights, with lighting durations and colors programmable using the software provided. To measure ambient temperature, humidity, and light intensity, they used a Sony MESH, where light intensity is measured, which can be measured in lux. However, lux may not be appropri-

ate in a precise sense because it is based on the property of human eyes. To measure the pure energy of lights, photosynthetic photon flux density (PPFD) is more appropriate in biology. In their platform, lux is used as a simplified index in environmental information. An RGB-D camera consists of an RGB camera and a structured light or time-of-flight depth camera based on infrared lights. Since the leaf size of plants in the early stages of growth is small, depth images usually need to be captured at a closer distance from the plant to magnify the size in the captured images. However, some RGB-D cameras are not designed for such short depth ranges. In their platform, they used Intel RealSense SR300 which is specifically optimized for short-range acquisition. They created two types of datasets using an RGB-D camera and a multiple RGB camera at approximately 2400 lux, 28 °C, and 30% humidity.

RGB-D Dataset

Since one RGB-D camera was attached to capture the entire part of the toolkit, each plant region was manually segmented as a plant image and labeled as a label image. In order to use the dataset for temporal leaf tracking, the same leaf label was assigned to the same leaf in images captured at different times. The original camera resolution was 640 × 480 and the resolution of the plant images was, for example, 166 × 190 pixels. Because the viewpoints of the RGB and depth images were aligned by the camera library, the labels were valid for both images. Due to the mismatch between RGB and depth camera, the resulting images were not aligned, so the depth image was aligned into an RGB image in our platform. For this reason, the authors provided the original depth image and a transformation matrix from the depth image viewpoint to the RGB image viewpoint so that the point cloud could also be transformed as needed without any interpolation.

Multi-View Dataset

For the multiview dataset, they captured five plants from three different viewpoints and manually segmented them into an individual plant like images. To use the dataset for spatiotemporal tracking of leaves, the same leaf label was assigned to the same leaf in images captured with different cameras at different times. The ground truth of the 3D shape was not measured because it was not easy to capture more accurate 3D shapes than when using multiple high-resolution images. This dataset will be useful for evaluating the segmentation of spatiotemporal instances for multiple views. In the relevant literature, instance segmentation has typically been performed using a single view image, and may be more difficult than using multiple views.

2.14.2 Multi-Modality Plant Imagery Database

This dataset consist of 2 main species the first one is Arabidopsis thalian plants were grown at 20°C, in a 16 h/8 h day-night cycle with daylight intensity set at $100\frac{\mu molphotons}{ms}$. The second is black bean plants of the cultivar Jaguar, were grown in a day-night cycle of 14 h/10 h with day and night temperatures of 24 and 18°C, respectively, and a daylight intensity set at $200\frac{\mu molphotons}{ms}$. In all cases, seeds were planted in soil covered with a black foam mask to minimize the fluorescence background of algal growth. Two-week-old plants (Arabidopsis and bean) were transferred to imaging chambers and allowed to acclimate for 24 hours to LED illumination before beginning data collection. Different types of images were collected; fluorescence images were captured once every hour during the daylight period in a growth chamber. A series of five images were captured using a Hitachi KP-F145GV CCD camera, during a short period of bright light saturating photosynthesis provided by an array of white Cree LEDs. Fluorescence was excited using monochromatic red LEDs, collimated using Ledil reflector optics, and pulsed for 50 μ s during a short window when the white LEDs were electronically turned off. Infrared images were collected once every hour with the same camera and filter used for chlorophyll a fluorescence. RGB color and depth images were collected using a Creative Senz3D sensor. The sensor contains both a 1280×720 color camera oriented parallel to, and separated approximately 25 mm from, a depth camera that has a resolution of 320×240 pixels. The depth sensor uses a near IR flash illuminator and measures the time-of-flight of the beam in each pixel to obtain depth estimates in each pixel. Low-reflectivity surfaces, such as foam under plant leaves, are accurate only at close range. Image data, including fluorescence, IR reflectance, RGB color, and 3D depth images, were collected once every hour. Five minutes before the end of each hour, the 3D depth images and color image were captured using the Creative Senz3D sensor, followed by fluorescence and IR reflectance images collected sequentially at 2-minute intervals from the CCD camera with IR filter. A planar checkerboard model was used to calibrate all three cameras to obtain both intrinsic and extrinsic parameters. While the grid pattern is not visible in the depth image, it is still observable in the reflected IR image whose pixels correspond to the depth pixels. The intrinsic and extrinsic parameters are stored as text files, and a Matlab function is provided that reads the parameters and can plot the camera positions. Time-of-flight depth measurements can have significant noise, and it is useful to both model and quantify it. Depth noise ϵ is modeled as the sum of an image-dependent term ϵ_I and a sensor-dependent term $\epsilon_S : \epsilon = \epsilon_I + \epsilon_S$. The term ϵ_I is a random variable for each pixel with a value that varies between successive images taken from a fixed pose of a static scene. On the other hand, ϵ_S is a random variable for each pixel that models its depth offset, and its value changes only when the scene changes. The variance of ϵ_I is estimated for each pixel of a fixed scene observed over multiple images. In our experiments, we observed flat albedo and uniform albedo. MSU-PID includes two subsets, one for each plant type: Arabidopsis and bean. Since there is no light at night, the

plants cannot be captured by the RGB fluorescence and color sensors, while the IR and depth cameras can still perform capture at night. To make sure that all four modes are present at the same time, we release the portion of images captured only during daylight hours, which is 15 images per day for Arabidopsis and 13 for bean for all four modes. The two subsets differ in image resolutions. For Arabidopsis images, we labeled four frames per day, while for bean images, we labeled seven frames per day due to the fast and spontaneous leaf movements of the plant. Leaf labeling can be propagated from fluorescence images to each of the other modes for Arabidopsis sequences. To quantify the accuracy of label propagation, we manually labeled three images for each of the three Arabidopsis plants on the fluorescence and RGB modes. The label in each mode is propagated on other modes. label propagation to provide labels for all four modes for Arabidopsis sequences. There are two benefits:

- decreases the need for manual labels,
- the labeling is consistent for all four modes.

In the case of bean images, pixel association between modalities is more difficult because the depth variations within the plant are large. We found that a homograph-based mapping worked poorly, and thus the manual annotations we provide apply only to infrared and fluorescence images using the same camera. Leaf segmentation, leaf counting, leaf alignment, and leaf tracking. To facilitate future research, we separate the database into training sets and test sets. Forty percent of the data is used for training and 60% for testing. Specifically, six Arabidopsis plants and two bean plants are selected for training. The user can decide to use one or more modes of the plant images for training and testing, respectively. Using RGB and depth modes for training and RGB for testing can take advantage of additional information during learning without incurring additional sensor costs during testing, which can be implemented either by learning with side information or transferring learning with the missing mode. To evaluate the performance of leaf segmentation, alignment, tracking, and counting, we use four performance metrics, whose Matlab implementations will be provided along with the data. Three of them are based on tip-based error, which is defined as the average distance of a pair of estimated leaf tips with a pair of labeled leaf tips, normalized by the length of the labeled leaves.

The authors defined a threshold τ to operate on the corresponding tip-based errors, from this τ three metrics are calculated:

- Unmatched leaf rate, the percentage of unmatched leaves relative to the total number of labeled leaves. This can be attributed to two sources. The first is missed detections and false alarms. The second is the matched leaves with tip-based errors greater than τ. When τ is large enough, this value is equal to the leaf count error,
- Landmark error, the average of the tip-based errors smaller than τ of all matched leaves frame-to-frame. It is used to measure the alignment error of the leaf tips,

- Tracking consistency, the percentage of matching leaves from video to video whose tip-based errors are smaller than τ. This is used to measure leaf tracking.

In order to evaluate the accuracy of leaf segmentation, an additional metric based on the Dice score of estimated segmentation results and ground truth labels was adopted: Symmetric best Dice, the average of the best symmetric Dice among all labeled leaves. The authors treated all images over nine days as a video from the first image on the first day to the last image on the last day. To generate the template set, they first selected 12 templates with different aspect ratios from the labeled images in the training set along with the corresponding tip positions. For each template, the images are scaled to ten different sizes to cover the full range of leaf sizes in the database. For plant segmentation, we use a simple threshold and edge detection process to generate an image mask and an edge map. The best threshold is learned from the training set, which is done by tuning the threshold in a certain range and we find the best one by evaluating the overlap of the segmented masks with the ground truth label masks. Firstly, they find the best position of each model in the edge map which has the minimum CM distance, secondly, the authors applied multi-leaf alignment, thirdly, they applied multi-leaf tracking. In the tracing process, we delete a leaf when it becomes too small. A new leaf is generated when there is a relatively large portion of the mask that has not been covered by the current leaf candidates.

Chapter 3

Algorithm Design

Chapter 4

Implementation

Chapter 5

Experimental Results

Chapter 6

Conclusion

Bibliography