Phenotyping and Skeletonization for Agricultural Robotics

Eric Schneider

CMU-RI-TR-23-23

June 20, 2023

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
George Kantor, *chair*
Sebastian Scherer
Jason Zhang

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Keep your fire burning

Abstract

Scientific phenotyping of plants is a crucial aspect of experimental plant breeding [1]. By accurately measuring plant characteristics, phenotyping plays a vital role in the development of new plant varieties that are better adapted to specific environments and have improved yield, quality, and resistance to stress and disease.

In addition to observing plants, robotic plant manipulation is becoming an increasingly important area of research in agriculture [3, 49, 61], as it has the potential to revolutionize farming practices. By using robots to interact with plants, farmers could eventually achieve greater precision and efficiency in tasks such as pruning, pollinating, and harvesting, leading to improved yields and reduced labor costs.

However, obtaining labeled data for the assessment of phenotype estimates or plant models can be an extremely challenging and time-consuming process in agriculture [15, 16, 32, 39]. We tackle this common problem in agricultural robotics along several avenues. First, we propose an unsupervised assessment method for reconstructed 3D sorghum clouds, which are used to count sorghum seeds for non-destructive phenotyping. Second, we use highly consistent outdoor imagery to simplify vine segmentation with low amounts of training data. Finally, we build 3D skeletal vine models intended for vine pruning, and assess these vine models using an unsupervised approach from previous work.

These skeletal vine models are then used in a case study in which we predict pruning weight in grapevines, one of the factors in optimizing grape quality and yield. Our results show that our approach outperforms previous methods in predicting pruning weight, demonstrating the potential for our method to improve agricultural practice.

Overall, our work highlights the benefits of 3D plant models for phenotyping and manipulation in agriculture, and presents a new approach to assessing reconstructed point clouds. Our findings have implications for the development of more efficient and effective agricultural practices, with the potential to play a role in simplifying sorghum breeding and grapevine pruning efforts through automation.

Acknowledgments

Firstly, I would like to extend my sincere appreciation to Harry Freeman for being a great teammate on the sorghum project. I would also like to thank John Kim and Mark Lee, who provided invaluable assistance with data gathering and writing.

For the skeletonization project I would like to thank Abhi Silwal, Francisco Yandun, Terry Bates, and the Cornell team for their help data gathering, as well as their previous efforts in this space which provided an essential foundation. Furthermore, I am grateful to Adarsh Krishnamurthy for the discussions we had about building and scoring skeletons. I would also like to thank Sushanth Jayanth for his contributions to the skeleton experiments and the writing process.

I also extend my gratitude to my advisor George Kantor for the many insightful discussions and accurate scoping predictions, which were extremely helpful in making my academic journey smoother.

Finally, I would like to thank all my lab mates for being a wonderful and supportive group. I hope you all find projects in the future that you love working on.

The support of ARPA-E TERRA, USDA NIFA, and NSF Robust Intelligence were crucial in supporting this work, and are much appreciated.

Contents

List of Figures

List of Tables

Chapter 1

Background

1.1 Motivation

With recent advancements in data-driven computer vision, agriculture is widely adopting image-based techniques to efficiently inspect vast quantities of crops. Automated crop inspections, which were not easily done before, enable farmers and breeders to make real-time decisions to manage pests, disease, and drought, and to automate laborious tasks such as phenotyping and yield prediction.

An agricultural phenotype is the set of observable characteristics of an individual plant resulting from the interaction of its genotype with the environment, which encompasses a broad set of plant features. Phenotypes of interest can include widely varying characteristics from average kernel size, to the grams of protein per kilogram of crop, to the number of grains per stalk in a cereal crop. In scientific plant breeding, the collection of phenotype data for a given crop is important for decision making. The earlier this phenotypic data can be conclusively collected for a crop variant being tested, the better. Breeding experiments often stretch for multiple growing cycles to conclusively test new plant breeds, so the ability to know a phenotype earlier allows expedited decision making, saving breeders time and money. In Section 3 of this thesis we explore a method of non-destructive sorghum seed counting, which could be used to measure seed count before the end of the growing season.

Although accurate phenotype measurement is valuable for scientific plant breeding, collection of phenotypic data is often very labor-intensive to gather. In practice, these values are often gathered by graduate students at agricultural institutions using manual tools such as protractors and measuring tapes, or by inspection of visual cues such as color and disease markers. For some phenotypes, such as number of seeds per stalk of sorghum, it is infeasible to gather these values without automation and so these measurements are not currently available to breeders. Automated phenotyping of various characteristics is a common project in agricultural robotics, and in this work we assess robotic pipelines to calculate two phenotypes:

1. Number of sorghum seeds per panicle
2. Pruning weight of a grapevine (a measure vine vigor/health)

In the common robotic paradigm of "Sense → Think → Act", automated phenotyping generally falls mainly into the categories of "Sense → Think". However, as agricultural robotics has progressed, robots in the field have gained a stronger understanding of their environment, allowing increased research into actions on that environment. In this work we develop a portion of a perception system for vine pruning, a type of plant manipulation.

Vine pruning during the dormant season is an important annual operation for grape growers. It is a costly and labor-intensive process, one that growers may struggle to staff due to shortages in skilled labor in agriculture. In some areas, mechanized systems have taken over as the most cost-effective solution, but lack the ability to selectively prune vines in a balanced manner to maximize grape yield and quality in the way that a skilled pruner can [20]. Robotic pruning has the potential to achieve superior outcomes compared to mechanized approaches by handling each vine according to its needs, and has been an active area of research for multiple groups [3, 14, 43]. However, robotic pruning efforts have focused on relatively simple vines that are mostly planar, with vertically aligned growth. Vine segmentation and skeletonization are discussed in Sections (4, 5).

Throughout this work we wrestle with the issue of scarce training data. Many of the problems we tackle, such as sorghum seed counting and grapevine skeletonization, would be amenable to machine learning techniques such as convolutional neural net regression if there was a robust associated dataset. A number of factors play into this issue. First, to tackle a given agricultural task you may need very specific labels, such as pixel-wise labels of a certain plant variety. Depending on the requirements of the modelling project, you may even require labels of specific portions of each plant. Commonly used non-agricultural datasets that feature greenery, however, may lump plants together into large generic categories. Cityscapes, a commonly used outdoor dataset for self-driving, labels only “terrain” and “vegetation” because those are what are relevant for their use case. Compared to the well-funded commercial efforts that have gone into building self-driving datasets, there has been much less focus on building up robust agricultural datasets. In addition, it is difficult to create accurate synthetic outdoor scenes because of the organic nature of plants. Many synthetic datasets use widely available models of human-designed objects (furniture, cars, toys, etc.) to construct artificial scenes. However, while plants are difficult to model, a number of groups such as video game companies are concerned with making useful plant models and this is an active area of development that should be leveraged by researchers. That said, many plant models are only concerned with getting the general look and feel right, without being accurate in the small details, which can limit their usefulness in creating synthetic training data.

1.2 Scope and contributions

We propose a computer vision-based method for non-destructive counting of sorghum seeds for early forecasting of yield. The non-destructive nature is important, as typical methods of seed counting require the harvesting and stripping of the seeds from the stalk. Our method, in contrast, could potentially be run on plants mid-way through the season without interrupting growth. In order to create a 3D model of each sorghum panicle from multiple stereo views, seeds are used as landmarks in reconstruction. Next, to count seeds a density maxima-finding method is presented. Accurate phenotype forecasting is valuable for sorghum breeding programs, as it would allow faster decision-making on variant suitability, which could expedite the current five-year breeding process [19]. Seed count would be a valuable phenotypic trait, but it is currently not possible to sample in a non-destructive way.

In contrast to the large and well separated fruits typically inspected, we investigate seed modelling on a sorghum panicle, which is more challenging from a computer vision perspective. The seeds are much smaller than typically studied crops (average diameter 3.3mm), making them difficult to detect and track. In addition, there is significantly more occlusion due to dense packing and clutter from husks. Although there has been work on 2D image based instance counts for other crops [13, 21, 26], it is still difficult to obtain an accurate count with sorghum.

We also propose a pipeline to create 3D skeletal models of dense dormant grapevines using a graph-and-refine skeletonization strategy on semantically segmented point clouds. These skeletal models are useful low-dimensional representations of vines, and can be used for cut location identification and motion planning in the grapevine pruning process. Previous skeletonization approaches are generally designed for simpler, sparse, and planar grapevines, so we improve current plant skeletonization methods in order to produce accurate skeletons of complex, heavy growth vines. In particular, we add the ability to model cycles in the skeletal model to better capture dense overlapping growth.

In addition to producing vine skeletons, we also predict grapevine pruning weight, a measure of each vine's health and vigor. Knowing the pruning weight of vines is an important step in balance pruning [44], as it is used to determine how much growth to keep or remove in the pruning process. Collecting pruning weight manually is time-intensive and disruptive to the pruning process, requiring the cut-offs from every vine to be carefully segregated and weighed [46]. Improving the ability of a robot to assess pruning weight has two benefits, it gives the grower plant by plant health data and also allows the robot to choose better automated pruning locations. We use 3D and skeletal data to predict pruning weight on dense and occluded vines more accurately than previous works.

We make the following contributions towards agricultural modeling and analysis:

- An end-to-end pipeline for non-destructive sorghum seed counting, along with a related public dataset.
- A new unsupervised quality metric for reconstructed point clouds, developed in order to improve 3D reconstruction quality and refine seed count.
- An end-to-end pipeline for skeletonization of dense and complex dormant grapevines, along with a related public dataset.
- A modification to graph-and-refine skeletonization strategies that handles cycles in the structure graph, allowing more accurate skeletal models.
- State of the art automated pruning weight prediction results.

In conclusion, both sorghum seed counting and grapevine skeletonization benefit from a shared set of methods, including stereo imaging, 3D modeling, and AI image processing. In both cases we also assess crop phenotypes, estimating respectively the seed number and pruning weight. The general approach presented here could be leveraged by future researchers in precision agriculture, hopefully smoothing the path for future insights and development.

Chapter 2

Related Work

Our work covers both 3D reconstruction of sorghum stalks for the purpose of counting seeds, and the segmentation of grapevine point clouds for the purposes of skeletonization and pruning weight estimation. Each of the subsections below cover the literature related to a portion of the covered work.

2.1 3D Plant Reconstruction

With regards to reconstruction in agriculture, most works are focused on larger maps and fields rather than single plants. For example, large field maps are reconstructed in [7, 37], as well as many others. Orchard rows are reconstructed in [41] by merging views using cylinders fit to trunks. This does not adapt well to merging sorghum views, as the stems are too small to effectively fit in our data. Although localized views of flowers and vines are captured in [36, 43], they do not get a complete 360° scan as we do when capturing data on sorghum stalks as described in Section 3.2. Some works like [6] use 360° point clouds of plants, but they are collected using high-quality scanning tables and do not need reconstruction algorithms to combine successive views.

2.2 Counting of Fruits and Seeds

Phenotyping during the breeding process is laborious if done manually, and as a result several works address automated phenotyping using robots. In [47], UAV images taken early in the growing season are used to predict end of season above-ground biomass of growing sorghum plants. [2, 55] show that images collected from mobile robots can be used to assess plant height and stalk size more easily than manual collection. Component traits such as these are used in genetic research to improve biomass yield. Our work explores seed counting as a form of phenotype assessment, which was not possible at the resolution of these previous systems.

There has been a significant amount of recent work dedicated towards reconstruction and counting in agricultural settings. Mapping and estimating the yield of mangos in occluded environments using a FasterRCNN segmenter is presented in [45] and [29]. Mapping and counting grapes in 3D by fitting spheres to point clouds along with a Mask-RCNN network and TSVM classifier is presented in [34]. While these methods work in their respective domains, they do not extend well to sorghum where the seeds are smaller and the level of density and occlusions are higher, making the seeds hard to consistently segment and fit shapes to.

There has also been relevant work in estimating seed counts for smaller crops from single 2D images. Counting rice and soybeans with density maps using convolutional neural networks (CNNs) is addressed in [13] and [26] respectively. These counting networks overcome challenges such as variable object shapes and semi-translucent, alabaster rice grains. However, the rice and beans have been stripped from the plant and laid out such that there are few occlusions, which we do not do. Density maps have also been used to count corn kernels on the cob, where the final count is proportional to the density map count as a result of corn's symmetric shape [21]. Similarly, [35] uses a KD-Forest based approach to detect grapes in clusters using keypoint-based features, and estimates yield using a scale factor. These methods employing scale factors from single images do not adapt well to sorghum due to asymmetry in sorghum panicles. The projected sorghum seed count from a single view of a panicle does not scale reliably to a full seed count.

2.3 Segmentation for Agriculture

Semantic segmentation is the process of calculating pixel-wise boundaries of objects, and it is a well-studied problem in computer vision. The first deep CNN approach to segmentation was [31], and in the following decade there have been a large number of works that expand the expressiveness, effectiveness, and flexibility of semantic segmentation. Many of the fruit detection works discussed in Section 2.2, for example, use some variety of popular open-source segmentation approaches. Progressive versions of Mask-RCNN [17] and YOLO [38] are common choices with available code.

However, when it comes to segmentation of outdoor images, getting labelled data is a challenging issue. This is even more true if you want to detect particular parts of plants instead of just labelling the entire area as something generic like “plant”, “foliage”, “brush”, etc. Training segmentation networks on low amounts of real-world labelled data is an open problem, with many current approaches being explored. Works like [15] explore automated approaches to propose labels to a human, speeding up the labelling process by only requiring human corrections. Some segmentation approaches such as [40] are designed to require less training data by relying on smaller model size and using significant data augmentation. Pre-training networks on related or unrelated images is a common approach to reduce training data, works like [39] explore that for agricultural robotics. Style transfer between images is also being explored to take a model trained in one outdoor setting and apply it to another [16], because outdoor environments experience wide visual variation. We explore the effects of model and augmentation choices on our task, grapevine segmentation with low numbers of training images.

2.4 Skeletonization

A common general approach for skeletonization of plants from point clouds is the graph-and-refine strategy. As presented in [5, 11, 30, 51], the first step in this strategy is to turn points into a dense graph, choose an initial single path, and then refine that path using a variety of methods. Finding a Minimum Spanning Tree (MST) path through the dense graph is a common way to construct an initial path, which we adopt. At the heart of graph-and-refine processes are the ideas that physically proximate points represent connected paths in the final skeleton, and the true skeleton will be well represented by an directed acyclic (tree) graph. However, when processing dense and overlapping vines, as is found in our dataset, connecting nearby points causes loops that are poorly represented by a tree graph. In addition, it is common in graph-and-refine methods to make allometric assumptions, where the radius shrinks from the trunk through the branches in a known fashion, which does not hold for grapevines. We choose AdTree [11] as a baseline representing this family of methods as it has open-source code to compare against.

[30] represents a good point of discussion reference for the graph-and-refine strategy, as it is a relatively early work that has multiple influences on the methods of this thesis. The purpose of [30] is to take in real-world point clouds of trees and produce both skeletal graphs and the predicted radius at that part of the skeleton. The initial assumptions are that

1. The branch chains (subgraph) are smooth, as reflected by small bending angles between adjacent edges.
2. The branch chains are longer and thicker near the root of the tree and shorter and thinner near the crown.
3. The density of the branch chains is inversely proportional to their corresponding thickness.

The presented procedure is to turn the raw points into a minimum weight spanning tree that connects all points using Djikstra's Algorithm. Then vertex importance is calculated using the length of each vertex's sub-tree, and an orientation field is used to smooth the tree graph while taking vertex importance into account. Edges are then collapsed so that long straight sections are represented minimally. Finally, the above process is repeated iteratively to get better reconstruction results. The skeletal output is used to update point weights in the Djikstra's Algorithm step, leading to refinement on multiple runs. As they claim, while it is able to connect skeletal segments that are disconnected due to small-scale occlusions, it is not designed to reconstruct the skeleton over large regions of missing data. In addition, it is not designed to function with the level of dense vine growth that we attempt to model.

Skeletons are often generated for plants because cylindrical segments capture most plant growth, but skeletonization is also studied in other contexts. Laplacian

Contraction (LC) [4] is a method of general skeletonization, based on point contraction. By design LC on points returns a cloud without connections, where points have been compressed to the predicted skeletal axis, and does not calculate skeletal radii. Learning based skeletonization methods exist as well [27, 52], but thus far they are based on synthetic data and have not been trained or evaluated on field data, and thus are not compared against in this work.

A variety of works deal with robotic pruning, for both trees and vines. In [53] a human chooses tree branch pruning points in an image, and the robot makes cuts using image-guided control. In [54] skeletons are built for pruning based on strong geometric assumptions about Upright Fruiting Offset (UFO) tree structure, which are not suitable for vines. [3, 14, 43] represent fully integrated pruning efforts that build vine models using 2D image edge tracing, region growing, and proximity-based 2D node connections, respectively. These works are evaluated on simple vines, and the modelling approaches do not generalize to dense growth. In this paper we push perception capabilities that could enable integrated approaches on denser vines.

[3] was the first fully integrated vine pruning robotic system, and represents a relatively heavy-duty and capable approach. In this work a mobile platform completely covers the vines and is moved along the row with a winch. Inside the mobile platform active lighting, three cameras for trinocular stereo, and a robotic arm with an endmill-based cutting implement are placed to interact with the vines. For computer vision, cane edges are detected and combined into 2D vine structures using stochastic image grammar. These 2D vine structures are then combined between multiple views in an bundle adjustment approach, and path planning for pruning cuts are done using this 3D model. As discussed earlier, pixel-wise labelled data for ground-truth assessment is difficult to obtain for agricultural images. This work used an unsupervised reconstruction metric for vineyard-scale assessment that we adopt, intersection-over-union (IoU) between the reconstructed vine models and the foreground/background segmented images. This is computed by back-projecting the skeletal model into every image and comparing it with the segmentation masks. Although [3] represents a full integrated grape pruning pipeline, the methods are designed for and evaluated on relatively simple and planar vines. We propose a pipeline to perform a single step in the pruning process, skeletonization, but develop it to handle much more dense and complex vine structures.

2.5 Pruning Weight Estimation

Because pruning weight is a valuable metric for grape growers that is labor-intensive to collect by hand, there is prior research into automated pruning weight prediction from images. In [33] pruning weight is estimated from 2D cane segmentation using a monocular camera and active lighting at night, mounted on an all-terrain vehicle. Linear regression is used to map the number of pixels segmented as “vine” to the pruning weight. In [22], pruning weight is estimated from foreground segmentation using depth data. Similarly to the above method, linear regression is used to map the surface pixel area of the foreground (obtained using the depth data) to the pruning weight. In both cases the vines being assessed are relatively simple and planar, and the methods do not transfer well to the dense vines in this dataset.

Chapter 3

Sorghum Seed Counting

3.1 Task

Counting sorghum seeds in a non-destructive manner is a useful task within the field of experimental plant breeding. Given seed count, researchers could rank variants earlier in the growth cycle, without having to wait for the end of season harvest. We explore the use of a 3D reconstructed model of a sorghum panicle, captured from multiple viewpoints around the stalk, to accurately count sorghum seeds. In order to count seeds in the 3D model it is necessary for the reconstructed model to achieve a high level of accuracy, specifically at the scale of seed size (typically a few millimeters across). An overview of our counting pipeline is shown in Fig. 3.1. All work in this chapter on sorghum seed counting was done in collaboration with Harry Freeman.

3.2 Dataset

In order to generate a high-quality 3D model of sorghum panicles, we set up an automatic data collection process by attaching a flash stereo camera [42] to the wrist of a UR5 arm. The robot follows a circular trajectory around the panicle as shown in Fig. 3.1, which results in 360° images of each panicle as illustrated in Fig. 3.2.

These stereo images were collected for 100 sorghum panicles. There were 10 panicles from 10 different species as seen in Fig. 3.3(a). To evaluate our proposed

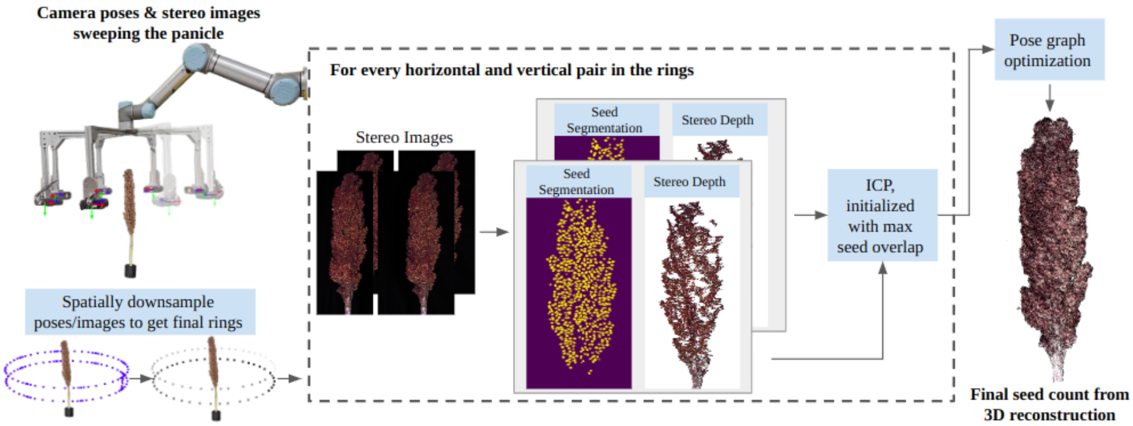

Camera poses & stereo images sweeping the panicle
Spatially downsample poses/images to get final rings
For every horizontal and vertical pair in the rings
Stereo Images
Seed Segmentation
Stereo Depth
Seed Segmentation
Stereo Depth
ICP, initialized with max seed overlap
Pose graph optimization
Final seed count from 3D reconstruction

Figure 3.1: 3D Reconstruction pipeline for the sorghum stalk

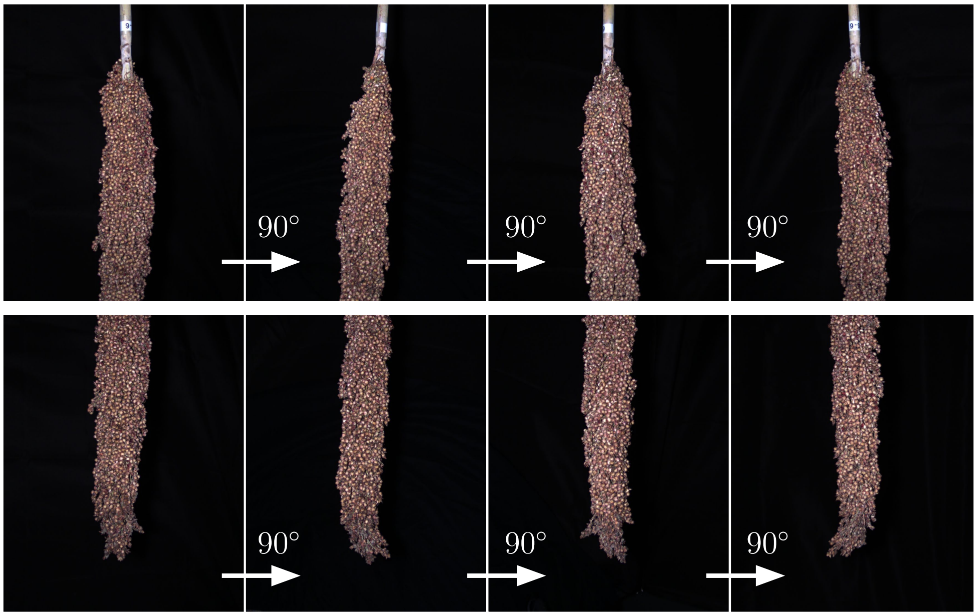
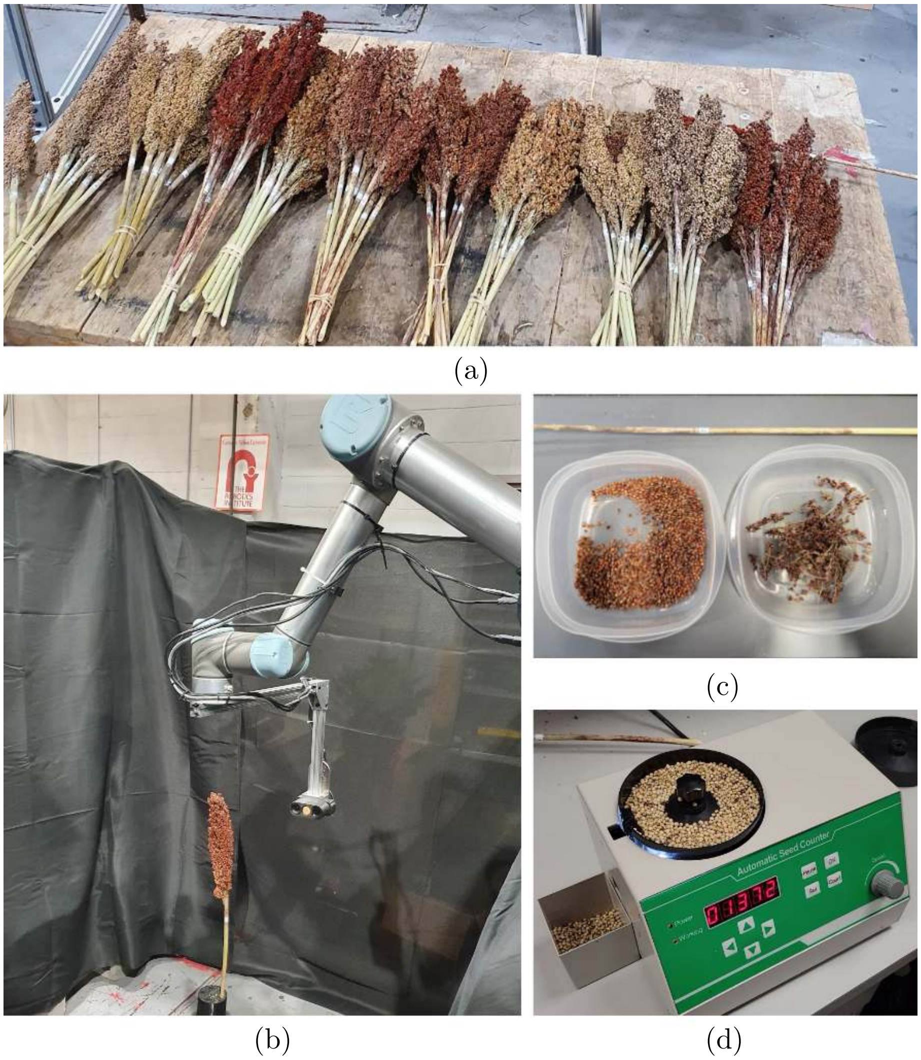


Figure 3.2: Illustration of 90° separated images for the top and bottom rings.

method, we manually stripped panicles (Fig. 3.3(c)) and counted all seeds using an automatic seed counting machine[1] (Fig. 3.3(d)), which serves as ground truth. The process of stripping seeds, removing husks, and counting took significant effort, on average 40 minutes per panicle, which reinforces the usefulness of an automated method for yield estimation.

[1] Wadoy Automatic Seeds Counter, Sly-C



(a)

(b)

(c)

(d)

Figure 3.3: (a) 100 sorghum panicles from 10 different sorghum species. (b) Our data collection system, a stereo camera attached to the UR5 robot arm. (c) Seeds were manually stripped and (d) counted using a seed counting machine.

Random errors in the seed count include some lost seeds that fell off panicles between image collection and hand-counting. Affecting the count in the opposite direction, some unremoved husks were counted as seeds by the counting machine despite manual efforts to separate seeds from husks. We expect the effect on the ground truth to be small. The stereo images, camera poses, human-labeled seed segmentations, panicle weights, and human-counted seed counts can be found in our dataset[2]. An example of the images, depth data, and segmentation results are seen in Fig. 3.4.

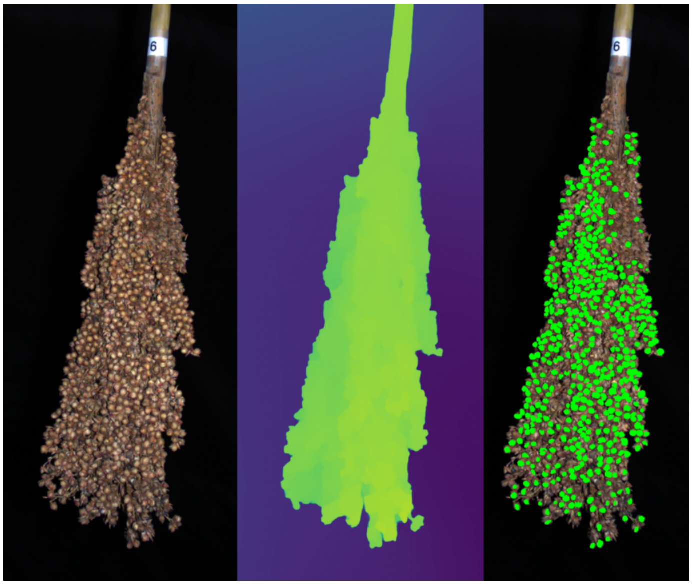


Figure 3.4: Visualized example of the images, depth data, and segmentation results in our sorghum dataset.

3.3 Reconstruction and Counting Approach

We go through a multi-step process to create a 3D model of the sorghum panicle, from which a seed count is calculated. To build the 3D model we spatially downsample the images taken to only consider images $\mathbf{I}_i \in \mathbb{I}$ and poses $\mathbf{T}_i \in \mathbb{T}$ in the shape of a double ring, spaced 5cm apart, as seen in Fig. 3.1. Roughly 85 images per panicle are left. A double ring was used because the camera field of view could not capture the entire panicle height. We use a trained instance segmentation model to identify seed masks, and use RAFT-Stereo [28] to construct point clouds for each frame. Using Iterative Closest Point (ICP) on the segmented seeds, we construct a pose graph that aligns all point clouds to create the final high quality point cloud $\mathbf{C}$. Lastly seed masks are combined between all images $\mathbf{I}_i$ to obtain a final seed count.

[2] High-Resolution Stereo Scans of 100 Sorghum Panicles at https://labs.ri.cmu.edu/aiira/resources/

3.3.1 Instance Segmentation

Instance segmentation is a form of image segmentation that detects single instances of objects (in this case seeds) and delineates their boundaries. We need instance segmented seeds for two purposes:

1. We found reconstruction is more reliable when using seeds as landmarks instead using full point clouds (Method: Section 3.3.2, Results: Section 3.5)
2. We used seed masks as the the basis of our counting approach (Method: Section 3.3.3, Results: Section 3.6)

Given a stereo image pair, we acquire a 3D point cloud semantically labeled with individual sorghum seeds by first calculating instance segmentation on 2D images, then projecting those masks onto the stereo points. In order to train an image segmentation model, hand segmented seeds from ten 1440 × 1080 sorghum images, spread evenly across different species, were used to fine-tune an ImageNet-1K pretrained CenterMask [24] instance segmentation network. Inference is performed on 120 × 90 tiles, then merged. Conflicting masks are merged if the IoU is high enough, or the higher confidence mask is chosen. After training, seed masks are projected onto the point cloud.

In order to choose high-quality seed points, the segmented seeds go through a few filters. First, segmented seeds where more than 15% of the segmented pixels have invalid disparity are removed. Then seeds where more than 15% of the pixels are dropped by a radius outlier filter are also removed. The remaining points are used as 3D seed points, and the median of each seed cloud is treated as the seed center.

3.3.2 Global Registration

Given a series of stereo images from different viewpoints, we need to determine the placement of the cameras in each viewpoint to create a combined point cloud of the panicle. We do this by registering point clouds from different viewpoints via pose graph optimization [8]. One challenge is that the clouds are dense, and ICP on the full cloud performed poorly due to bad correspondences, an example of ICP falling into local minima. Instead we choose a limited set of high-quality points in the cloud and run ICP only on those points, somewhat analogous to performing optical flow on higher quality landmarks like SIFT features. Instance segmented seeds with high confidence are identified based on their inference scores. The set of good seeds from image $\mathbf{I}_i$ are then used as node $\mathbf{P}_i$ in the pose graph, and pose graph optimization is performed using the Levenberg-Marquardt algorithm [25]. An example of a reconstructed panicle is shown in Fig. 3.5(b).

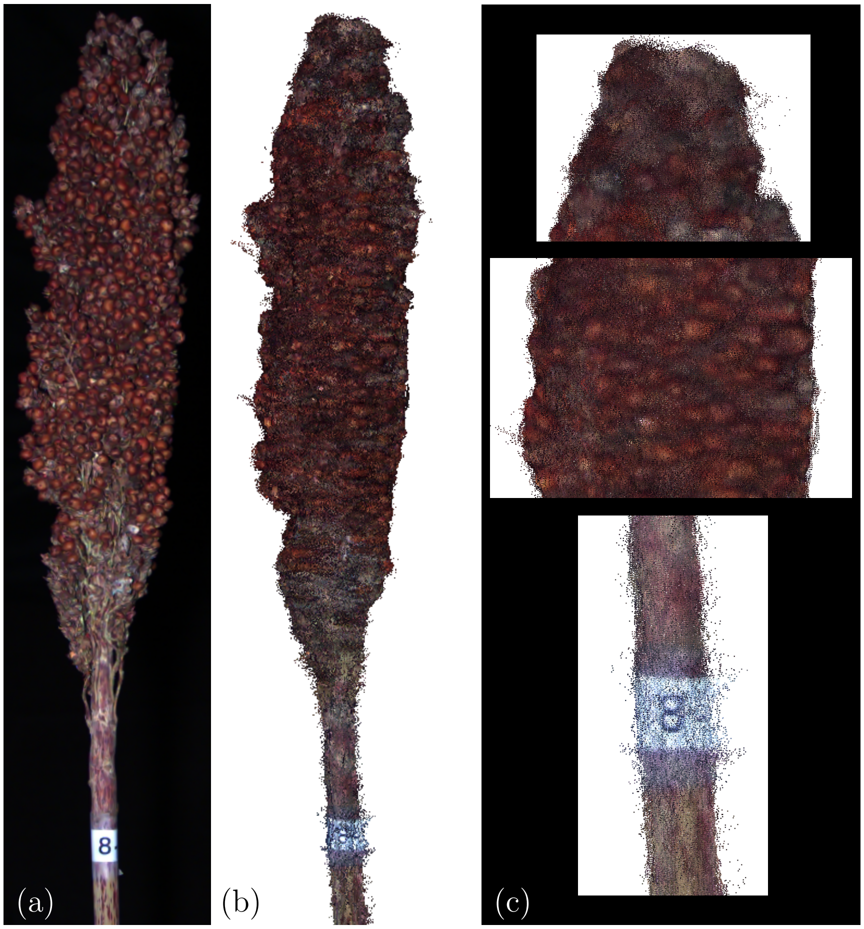


Figure 3.5: Example reconstruction results. (a) one of the original RGB images, (b) the colorized point cloud, (c) zoomed view of the colorized point cloud at the stem, mid-body, and tip. Some points of interest include the "8" on the stem label, and the body outline which matches the RGB outline well.

We observe that using camera poses from arm kinematics to initialize ICP yields poor results on the scale of seeds. This is due to error in extrinsic camera parameters, despite using a standard hand-eye calibration process. Hence, we refine the camera pose priors by maximizing seed mask overlap. The seed masks of two neighboring nodes $\mathbf{P}_i$ and $\mathbf{P}_j$ are projected into a common image frame, at the average pose between $\mathbf{T}_i$ and $\mathbf{T}_j$. We search for the pixel shifts that yield maximum intersection over union (IOU) of seed masks as shown in Fig. 3.6. The *No Shift Maximize* ablation test in Fig. 3.14 shows that this IOU maximization improves reconstruction.

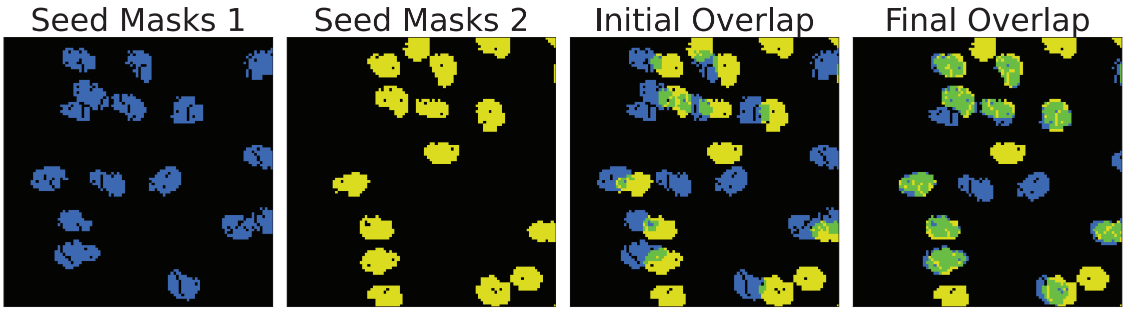


Figure 3.6: Matching mask structure with maximum IOU. Seed masks 1, seed masks 2, and their intersection are colored blue, yellow, and green in respective order.

3.3.3 Counting

In order to obtain a final seed count, we use the 3D model to ensure that a single true seed segmented in multiple images will be counted only once. The following 3D counting method performs this combination of 2D counts while handling the close proximity of neighboring seeds, spurious detections, and noise in the point cloud.

First, 3D seed centers are clustered using density-based spatial clustering (DBSCAN) [12], as shown in Fig. 3.7(d). Seeds are then counted in each cluster. Next, we adapt the concept of 2D image smoothing and apply it to 3D point clouds. In image processing, a 2D Gaussian filter smooths an image by calculating a weighted average around each pixel's neighborhood. We take this idea and extend it to 3D. In our method, each seed center in the cluster is treated as a unit-impulse, and each impulse is smoothed around a volume of space using a 3D Gaussian sphere. Areas of space near multiple centers will have a higher density that those that are further away or near fewer centers. An example of this density map can be seen in Fig. 3.8(c).

Once the density values are calculated for the cloud points, the final step to calculate the number of seeds in each cluster is to find the local maxima within a defined radius. This is a type of non-maximal suppression (NMS) on the density

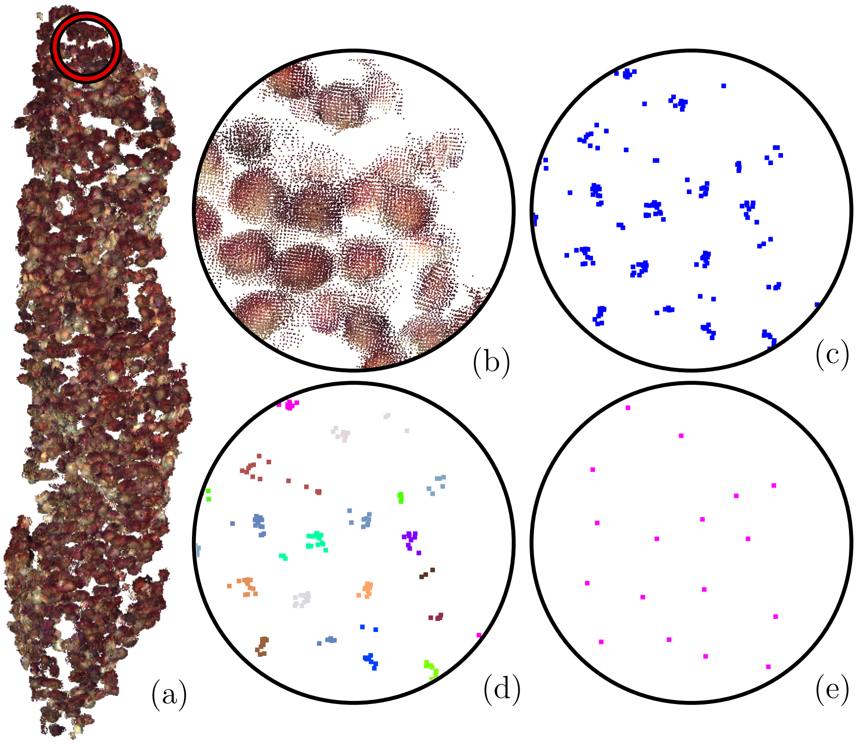


Figure 3.7: (a) An example of a final point cloud seed mask, (b) zoomed seeds, (c) seed centers, (d) seed centers clustered with DBSCAN, and (e) final seed sites.

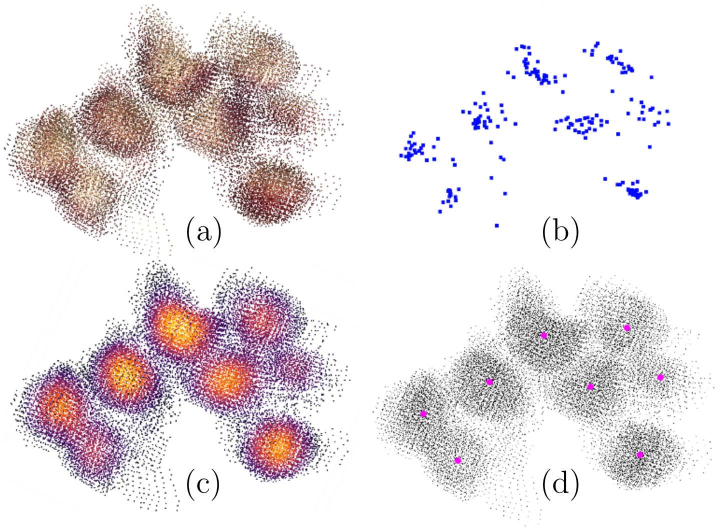


Figure 3.8: (a) Seed point cloud that has been put in a single cluster by DBSCAN, (b) seed centers from individual images, (c) seed points weighted by seed-center density, and (d) local maxima (pink) that have been chosen as seeds.

values. Each local maximum corresponds to a unique seed and is treated as the location of the seed's center as shown in Fig. 3.8(d). Once all local maxima are found for each cluster, the total number of maxima becomes the final seed count. Fig. 3.9 gives an example of this seed-detection process on an entire panicle.

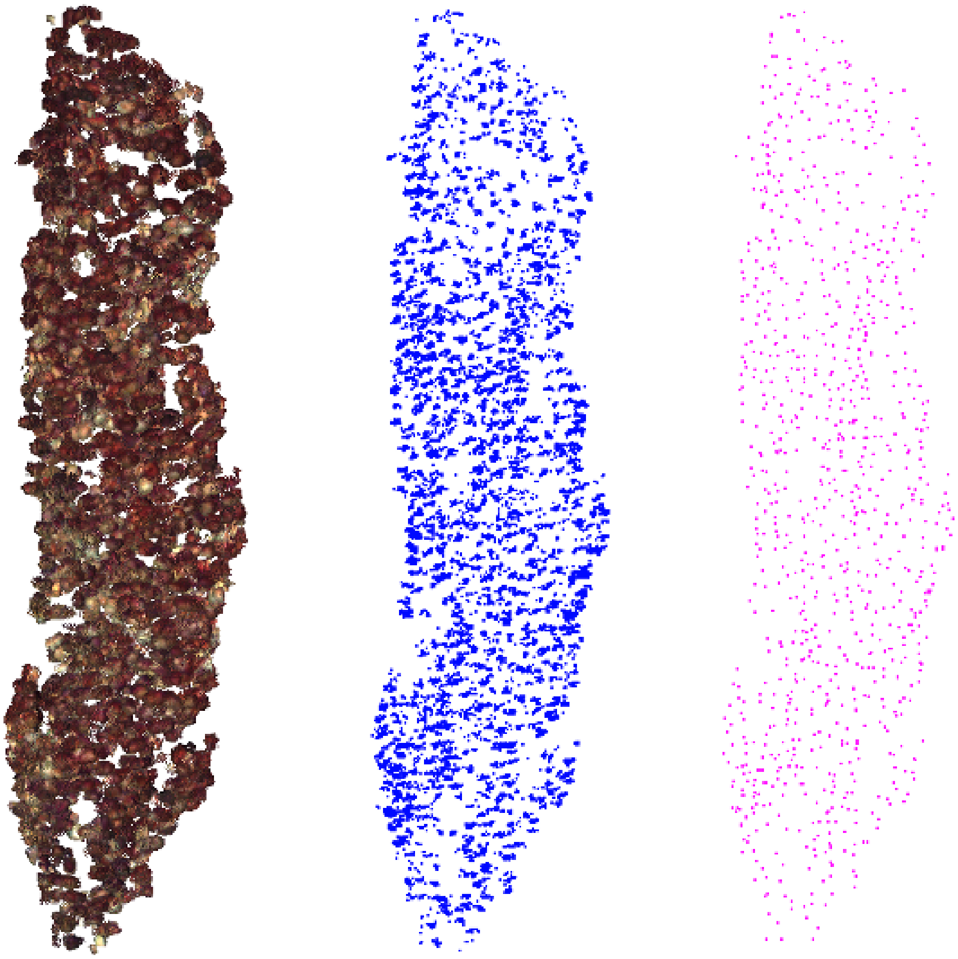


Figure 3.9: From the cloud of masked seed instances (left), we find detected seed centers from all views (middle). After identifying maxima in the density cloud the final, filtered seed positions are given (right).

3.4 Unsupervised Evaluation Metric

Several prior works [57, 59] discuss quantitative reconstruction evaluation in the absence of ground truth, but they require that the final output to evaluate against is a mesh. Our reconstruction method produces a dense point cloud, so we developed and validated a novel cloud-only rendering based method for assessing reconstruction quality in the absence of ground truth. We compare a small circle of pixels sampled from an RGB image $\mathbf{I}_i \in \mathbb{I}$, centered on a sampled seed, against a projected render of the same seed made using the full reconstructed cloud. The point sampling process is visualized in Fig. 3.10, and example renders are shown in Fig. 3.11. A sampling function λ is defined so that K seeds are sampled per image along the center of the vertical axis where the projections are cleanest. This method experimentally indicates relative levels of noise in the reconstructed point clouds by comparing rendered sections to the original RGB images.

To validate this framework, noise was purposefully introduced in the camera poses $\mathbf{T}_i$. In this way we created multiple reconstructed clouds, with varying levels of introduced noise. The strongest response to introduced noise came from normalized grayscale image patches. Both the mean-squared error (MSE) on image gradients, and the Structural Similarity [48] (SSIM) on image Laplacians responded well to the introduced noise, shown in Fig. 3.12.

A variety of comparisons were run on pairs of RGB image patches and the corresponding rendered patches in order to settle on these operators. To find the strongest response to noise we checked all combinations of RGB/grayscale, normalized/unnormalized, and comparing the intensity/gradient/Laplacian of the two patches. These operations are visualized in Fig. 3.11. Two examples of our image-to-render comparison with their corresponding MSE and SSIM scores are shown in Fig. 3.13.

Our reconstruction quality metrics "$\alpha\beta$-MSE" and "$\alpha\beta$-SSIM" are defined as follows. For each image, the sampling function λ samples K seeds from $\mathbb{S}_i$, where $\mathbb{S}_i$ are the seeds in image $\mathbf{I}_i$. For a sampled seed $s_{ik} \in \mathbb{S}_i$, the image patch α_{ik} and rendering of the point cloud β_{ik} are generated, both of which are grayscaled and normalized. The MSE and SSIM of α_{ik} and β_{ik} are calculated, then averaged over all seeds and panicles.

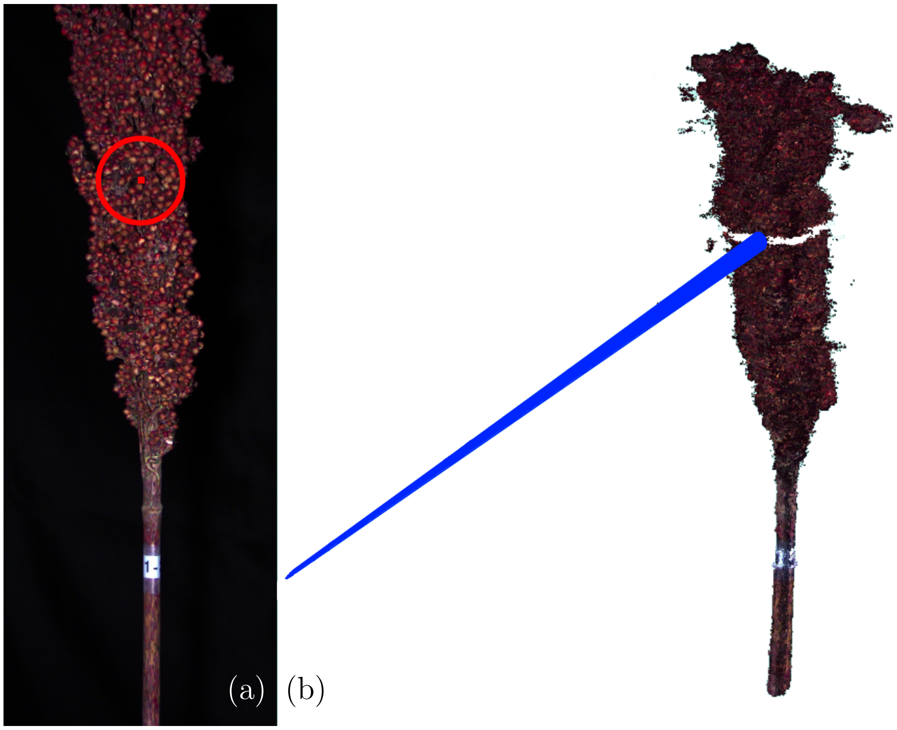


Figure 3.10: (a) RGB image of a sorghum panicle, where a single seed (highlighted in red) has been selected by the sampling function λ. (b) Visualization of the render projection, where a cone (blue) reaching out from the render origin selects only the points around the chosen seed.

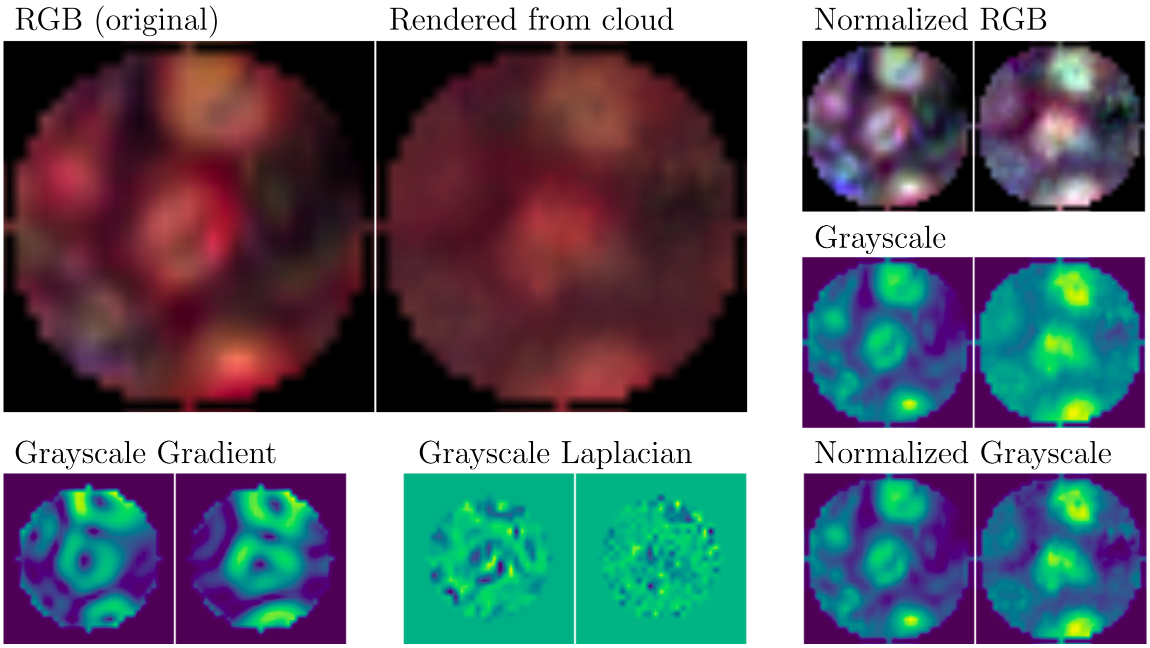


Figure 3.11: Examples of the image operations that were explored when finding patch comparisons most sensitive to reconstruction noise.

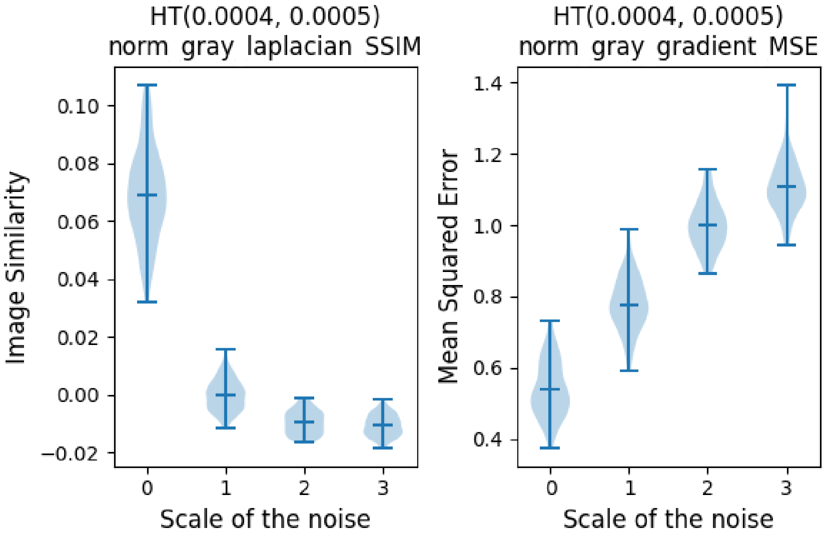


Figure 3.12: Response of chosen metrics to introduced noise. Noise took the form of homogeneous transforms, with translational noise drawn from a Gaussian $\mathcal{N}(0, \sigma = \text{scale} * 0.4\text{mm})$ and rotational angle noise drawn from a Gaussian $\mathcal{N}(0, \sigma = \text{scale} * 0.5\text{mrad})$. After the random transforms the cloud was recalculated and rendered.

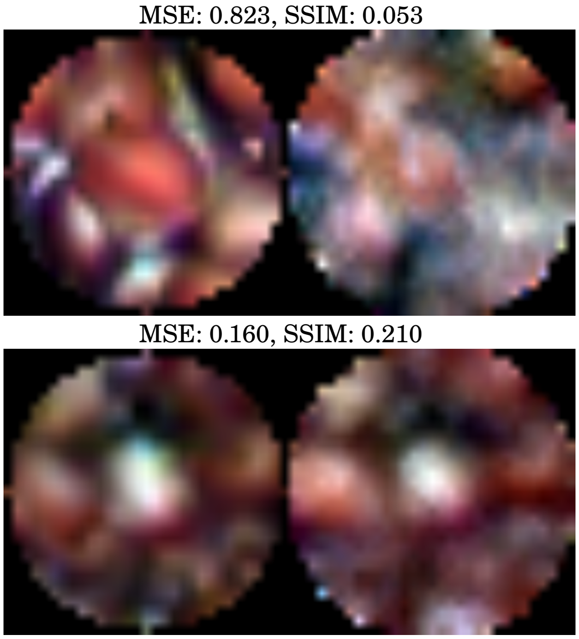


Figure 3.13: Qualitative examples of the reconstruction metrics. On the left are image patches, on the right are patches rendered from the reconstructed point cloud. Patches are normalized so each channel has min/max values of 0/255.

$$\begin{aligned}
\mathrm{MSE}_{ik} &= \frac{1}{N}\sum_{pixels}\big[\nabla\alpha_{ik} - \nabla\beta_{ik}\big]^2 && (3.1)\\
\alpha\beta\text{-MSE} &= \frac{1}{P}\sum_{p}\frac{1}{IK}\sum_{i}\sum_{k\in\lambda(\mathbb{S}_i)}\mathrm{MSE}_{ik} && (3.2)\\
\mathrm{SSIM}_{ik} &= \mathrm{SSIM}\big(\mathcal{L}\big(\alpha_{ik}\big),\mathcal{L}\big(\beta_{ik}\big)\big) && (3.3)\\
\alpha\beta\text{-SSIM} &= \frac{1}{P}\sum_{p}\frac{1}{IK}\sum_{i}\sum_{k\in\lambda(\mathbb{S}_i)}\mathrm{SSIM}_{ik} && (3.4)
\end{aligned}$$

Here ∇ is the image gradient, $\mathcal{L}$ is the image Laplacian, $\frac{1}{IK}\sum_i\sum_{k\in\lambda(\mathbb{S}_i)}$ indicates an average over sampled seeds in all images, and $\frac{1}{P}\sum_p$ indicates an average over all panicles.

3.5 Reconstruction Results

We assess the effectiveness of our reconstruction approach with ablation tests using the reconstruction metrics described in 3.4. Below references to “$\alpha\beta$-MSE” and “$\alpha\beta$-SSIM” are referring to these specific operations on image and rendered patches. Note that growing $\alpha\beta$-MSE (error) and dropping $\alpha\beta$-SSIM (similarity) both indicate a worse match. Fig. 3.14 shows results of ablation and comparison tests on reconstruction quality.

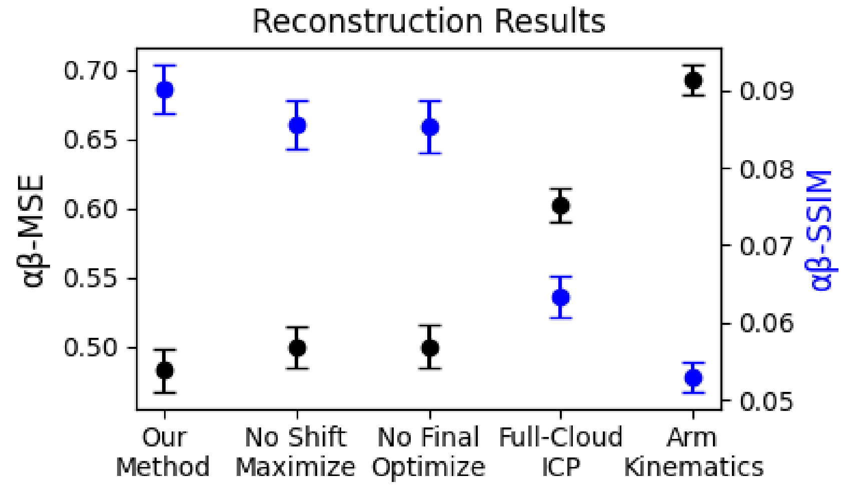


Figure 3.14: Noise metric results showing growing error and dropping similarity for reconstruction experiments. The vertical bars are the 95% confidence intervals for the mean of the per-panicle scores.

1. *Our Method*: Our final method. All experiments below are tweaks to this approach. This had the best average $\alpha\beta$-MSE and $\alpha\beta$-SSIM scores.
2. *No Shift Maximize*: The mask overlap maximization discussed in Section 3.3.2 is not used. This resulted in a slight decrease in reconstruction quality.
3. *No Final Optimize*: The pair-wise ICP transformations discussed in Section 3.3.2 are still used to adjust cameras relative to the first frame, but the final optimization is not applied.
4. *Full-Cloud ICP*: Instead of running pair-wise ICP on masked seed points, ICP was run on the full point clouds. This test showed a significant drop in

reconstruction quality.

5. *Arm Kinematics*: Views were combined using the arm kinematics, with no pose optimization. Although kinematically reconstructed panicles could be used for applications like collision avoidance, they had the worst reconstruction scores and could not be used for counting. Single seeds were clearly represented in multiple 3D locations, "smeared" cylindrically around the panicle.

The best reconstruction results came from pose adjustment using ICP on points determined to be high-quality seeds, and did notably better than ICP naively done using the full cloud from each image. Our hypothesis on why full-cloud ICP is worse is that sorghum is very organic and complex, and picking out meaningful, high-quality areas for ICP to operate on reduces the likelihood of ICP falling into a local minimum. As was discussed in *Arm Kinematics*, the required quality of reconstruction depends on your application. When using 3D structure to identify overlaps in 2D segmentation, decreasing reconstruction quality will lead to counting errors as identifications of the same seed drift apart in space.

3.6 Counting Results

As shown in Fig. 3.15, the seed count produced by our method has a strong linear fit to the ground truth seed count, with an R^2 of 0.875. The 10-fold RMSE using a 75/25 train/test split calculates an average prediction error of 295 seeds. We are pleased with the quality of this fit, since sorghum panicles have internal, hidden seeds that cannot be seen from an outside view. The only way to expose all seeds is to strip them off the panicles, a time-consuming process.

Ultimately, the primary characteristic for sorghum is its yield weight, which represents a sellable quantity of the crop. The fit between count and seed weight is still reasonably representative, with an R^2 linear fit of 0.819 as shown in Fig. 3.16, but it fits slightly less well than the seed count. This may be due to variations in seed density. The 10-fold RMSE using a 75/25 train/test split calculates an average prediction error of 8.5 grams per panicle.

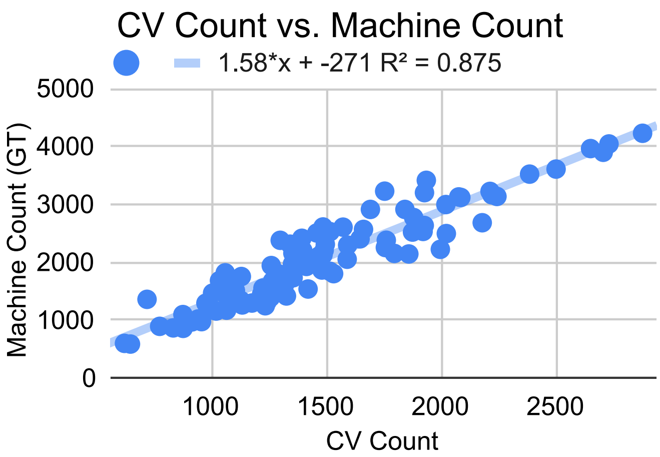


Figure 3.15: Fit between our method's count (Computer Vision/CV Count) and the ground truth count as described in Section 3.2.

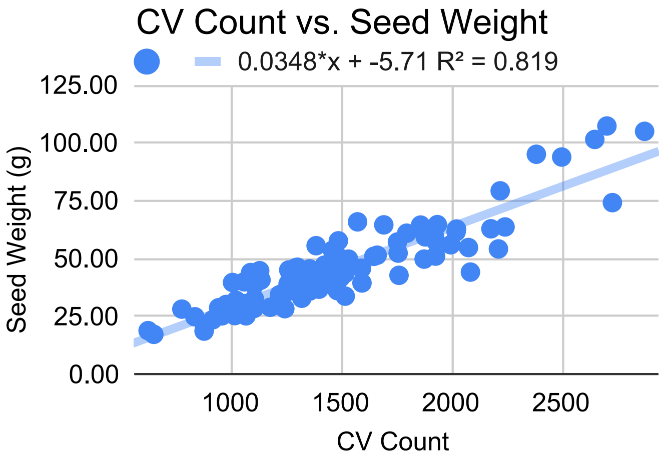


Figure 3.16: Fit between counted seeds and seed weight, which is the weight of seeds after they have been stripped off a panicle and cleaned of husks.

3.6.1 Benefits of 3D Data over 2D

In [21] it was shown that it is sufficient to take a 2D count of one side of an ear of corn and scale that to a full kernel count. To test this, ears were rotated around their long axis in 90° increments and imaged, and it was found that the single-image kernel counts at each 90° increment had low variation because kernels were generally evenly distributed. In contrast, sorghum is more complex in shape, and therefore has more variation when a full count is extrapolated from a single image. In Fig. 3.17 and Fig. 3.18 we compare the predictiveness of 2D and 3D counts.

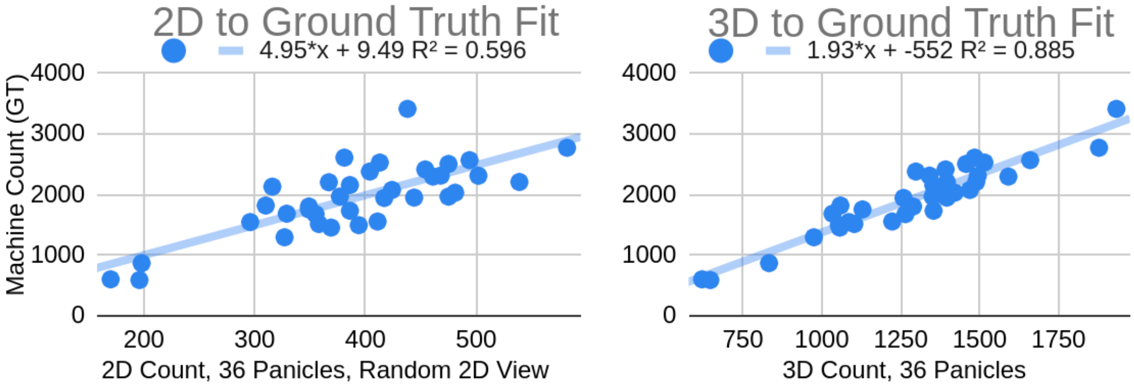


Figure 3.17: Comparison of 2D and 3D counts fit to ground truth. 2D count comes from a single image per available panicle and has a lower R^2 score, indicating worse predictive performance for linear regression. The 10-fold RMSE for these 2D and 3D counts are 353 and 204 respectively.

Comparing 2D and 3D extrapolation is a somewhat unfair comparison because 3D methods have more data available (dozens of images vs. a single image), but it is important to evaluate for hardware considerations. Getting images surrounding a plant for 3D reconstruction is more costly in terms of system complexity, requiring the camera to be actuated rather than fixed to a mobile base such as a tractor, so it is important to assess what relative benefit the 3D method brings.

In order to test the extrapolation principle, we obtained 2D segment counts from images spaced 90° apart. This was complicated by the fact that some panicles were too tall to be captured in a single frame. To avoid trying to combine segmentation counts from multiple images, we only use counts where the full panicle is visible in four 2D views. 36 out of the 100 panicles met this criteria, enough to get a reasonable representation.

As seen in Fig. 3.17, 3D counts have a significantly better linear fit to the ground truth counts, with an R^2 of 0.885 compared to 0.596 for 2D counts (sampled randomly from the 90° separated views), demonstrating that 3D count is a better predictor of the desired feature. The variation in 2D count within each panicle can be seen in Fig. 3.18. There are significant variations in extrapolated counts within each panicle, often stretching to 20-40% of the ground truth value.

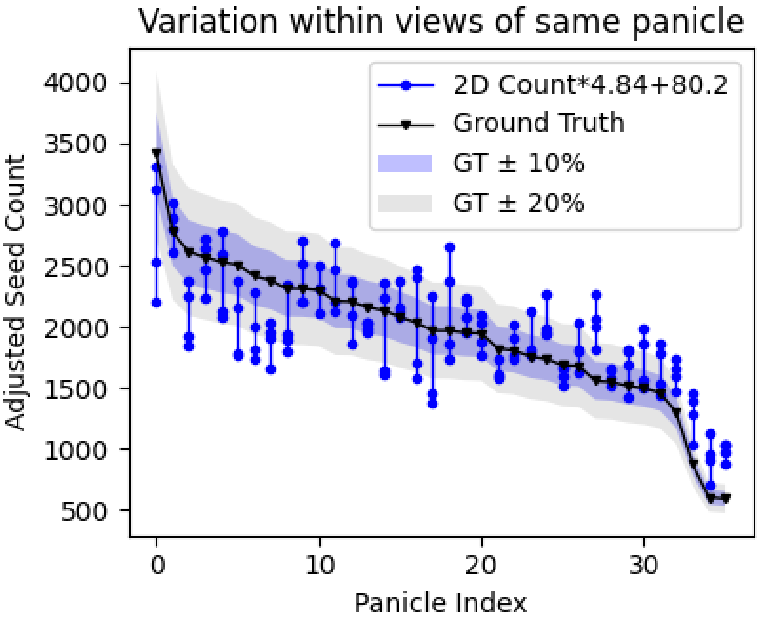


Figure 3.18: Variation across viewpoints among the 36 panicles, using a linear fit to extrapolate from 2D count to an estimated full count. Linear fit parameters have been recalculated to use all four 90° separated images per panicle instead of a random one as in Fig. 3.17. R^2 on the increased views was 0.634.

3.7 Extensibility

Although the images in this section were captured in the lab, in-field image capture from an arm mounted on a mobile base could be achieved with further research. The primary challenges that would have to be overcome are:

- Sorghum grows close together, and in the field a single stalk would likely not be naturally available for use to encircle with the robot arm. For imaging purposes it would be necessary to isolate a stalk with the required amount of surrounding free space. Our lab is investigating the use of robotic arms for manipulating branches, which could naturally extend to pushing neighboring stalks away.
- In the lab, we took images on a black background for simple color-based foreground segmentation. Images taken in the field would require a more nuanced foreground segmentation approach, but this should be achievable because we have depth data from stereo images and additionally image segmentation is a well-researched area of study.
- Our currently trained instance segmentation network would not work on field images due to domain shift, so new data from the field would need to be labeled. The use of an illumination invariant camera for imaging [42] will ease that process by requiring less data for a good model.

Chapter 4

Low Data Cane Segmentation

4.1 Task

In addition to counting sorghum seeds, a separate challenge explored in this work is the skeletonization of dormant grapevines for pruning purposes. The skeletonization process is designed to model vines of higher complexity and growth than previous robotic pruning works. As part of processing data for skeletonization, it was necessary to separate out points in the point cloud according to their class to determine which points formed the vine body. We solved this using semantic segmentation, a well-studied image processing problem. In our images from the vineyard, we classify pixels into the classes of background, cane, cordon, post, leaf, and sign, where posts were the metal stake supporting each vine and the signs were black and white fiducial boards. The full skeletonization process, which uses the semantically segmented images produced by the models discussed in this section, is presentated in Section 5.

One particular challenge in training semantic segmentation models for this use case is the extreme lack of labelled data. For context, two popular semantic segmentation datasets today are ADE20k [60] and Cityscapes [10]. ADE20k contains 25,574 training images, while Cityscapes has 20,000 coarsely labelled images and 5,000 finely labelled images. As discussed in Section 4.2.1, we work with 91 labelled images total, 64 of which are in used for training. Thus our task was to train a vine-segmentation network to operate well with low amounts of training data, leveraging illumination invariant imaging [42] and geometric augmentations.

Key Terminology

Vine: one full plant, including the cordon and all individual canes

Cordon: oldest part of vine, similar to a tree trunk

Cane: one branch of the vine, growing from the cordon and potentially splitting into further canes

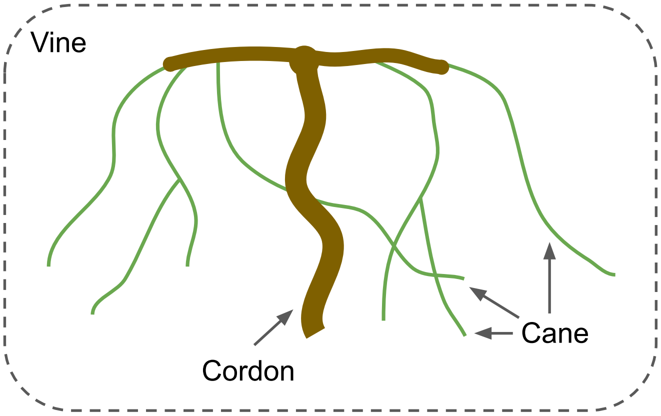


Figure 4.1: Simple diagram for the Key Terminology.

4.2 Dataset

When capturing grapevine data for skeletonization, the primary sensor data captured consisted of stereo images from side and down-facing camera pairs along a linear slider. Images were taken at seven points along a linear slide for each grapevine, then the mobile base moved to the next vine. Data capture was done using the platform from [43] (Fig. 4.2), using the flash camera from [42] which collects consistent images in varied outdoor lighting conditions using active lighting.

The cross-image consistency greatly benefits image processing algorithms like segmentation, because vines at different times appear substantially similar as shown in Fig. 4.3. In total 144 Concord vines were scanned. The grapevines are ownrooted vines planted in 2012 with a 8.5 foot row spacing and 8 foot vine spacing. They are single-wire trained to a six-foot high bi-lateral cordon and cane pruned with a sprawling growth habit.

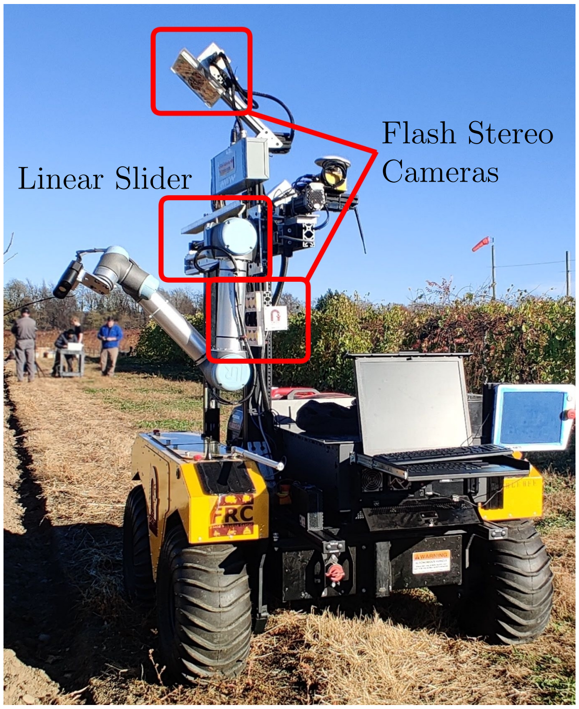


Figure 4.2: Robotic data capture platform with two stereo pairs on a linear slider, introduced in [43]. In this dataset the arm-mounted camera images were not used.

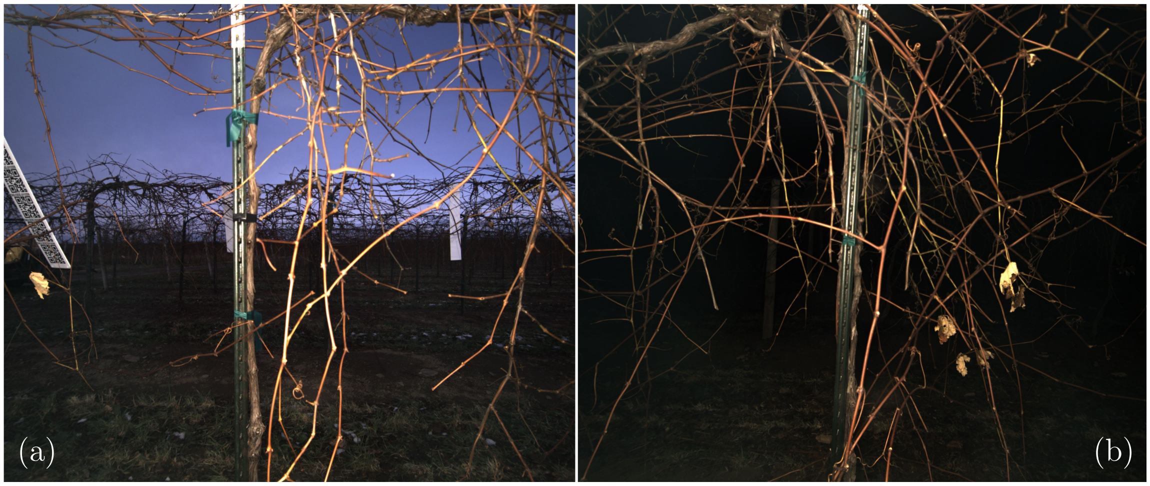


Figure 4.3: (a) Image taken close to noon, in full daylight. (b) Image taken after sunset. Although the background differs significantly, the appearance and texture of the vines is very similar. As investigated in [42] the consistency aids in learning with smaller datasets.

4.2.1 Annotation

Annotating pixel-wise classification of these images took a significant amount of time, around 1-3 hours per image. Compared to simple shapes where a pixel-accurate boundary can be captured with a few lines, hundreds of dense and extremely thin canes per image took significant effort to carefully delineate. Due to the high effort of annotating segmentation for high resolution images with many thin features, only 91 images were labeled pixel-wise using polygons, broken into the classes (background, cane, cordon, post, leaf, sign). The stereo images, class annotations, and pruning statistics, along with more information about the robot platform and vine variety, are available as a public dataset[1] as shown in Fig. 4.4.

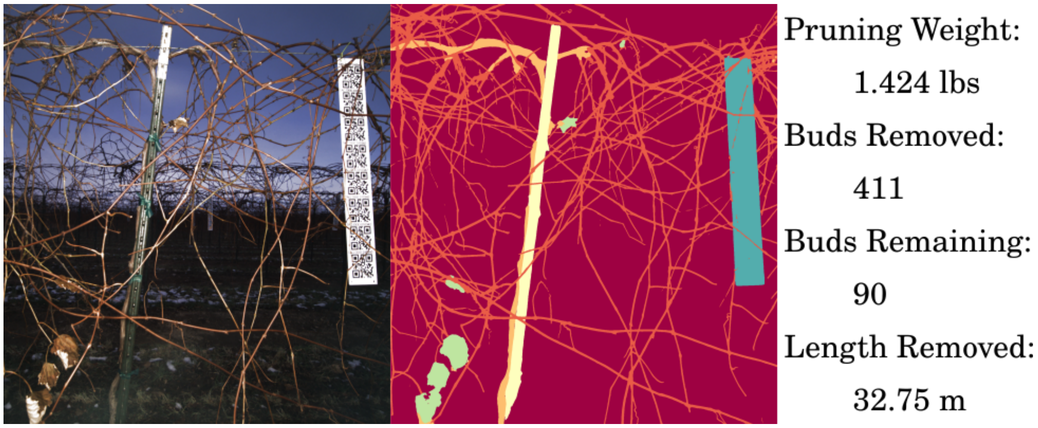


Figure 4.4: Example of a captured stereo image (left), class annotations (middle), and expert-assessed pruning values (right).

4.3 Results

In order to separate cane points from other classes in the point cloud, we use learning-based 2D segmentation to classify pixels in the stereo images, then apply class masks onto the stereo disparity. We assessed various image-based segmentation models and picked the most performant.

[1] Stereo Data for 144 Winter Grapevines at https://labs.ri.cmu.edu/aiira/resources/

4.3.1 Model

Using the MMLab segmentation toolkit [9], we tested a series of models. The goal was not to fully explore the space of models, but instead span the conceptual space to try and find a functional model. The models tried were:

- FCN [31]: fully connected autoencoder-like model
- UNet [40]: designed for simplicity and low amounts of training data
- BiSeNet [56]: split network designed to pass semantic and spatial information along separate paths
- Segformer [50]: transformer-based architecture

The 91 labeled images were split randomly (70/20/10) into train/validate/test sets, leaving 64 images for the training set. One weakness when setting up the training approach was that all images were treated equally, and the train/test split did not purposefully keep all labelled images of a given vine in the same split. However, because there are 91 labelled images and 144 vines, the cases where images of a single vine are randomly double-sampled and then found in both train and test should be extremely rare and have minimal impact on the results. The models were trained from scratch[2] for 125 epochs on a NVIDIA GeForce GTX 1080 Ti. As seen in Fig. 4.5, UNet has the highest performance among the tested models. This is interesting since it is a smaller and less expressive architecture, but in this case the relatively small amount of training data available likely made the lower capacity model more viable.

We assess the models using F1 score, which combines precision and recall into a single classification metric.

$$\begin{aligned}
\text{Precision} &= \frac{TP}{FP + TP} && \text{Proportion of correct positives} \\
\text{Recall} &= \frac{TP}{FN + TP} && \text{Proportion of real positives caught} \\
\text{F1} &= \frac{TP}{TP + \frac{1}{2}(FP + FN)} && \text{Common combined score}
\end{aligned}$$

We use the F1 Score Macro Average value to assess performance on all classes,

[2] Except for Segformer, comes pre-trained on ImageNet-1K

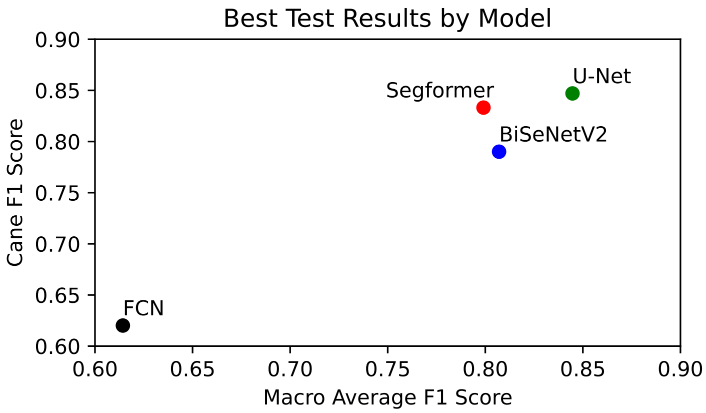


Figure 4.5: Best results by model We can see that UNet does slightly better than Segformer at predicting canes, but significantly better when predicting all classes.

defined as the per-class F1 mean. Macro-averaging is encouraged when working with imbalanced datasets where the importance of all classes is equal, which is a good description of our label distribution.

Table 4.1 contains the per-class F1 results for the best UNet model, showing a good balance of performance across all classes, with the exception of the sign class which is of particularly low incidence.

Class 1 background	Class 2 cane	Class 3 cordon	Class 4 post	Class 5 leaf	Class 6 sign
0.973	0.847	0.843	0.896	0.884	0.625

Table 4.1: Per-class F1 scores for the final model

A variety of hyperparameter and augmentation values (Section 4.3.2) were tried during model testing. In Fig. 4.6 we can see the full spread of resulting performance on the test dataset. Although UNet performed worse than Segformer in a few experimental conditions, in general it held strong as the most reliable architecture.

4.3.2 Augmentations

When training on low amounts of labeled data, data augmentation is important to improve performance and reduce overfitting. Speaking generally, data augmentation

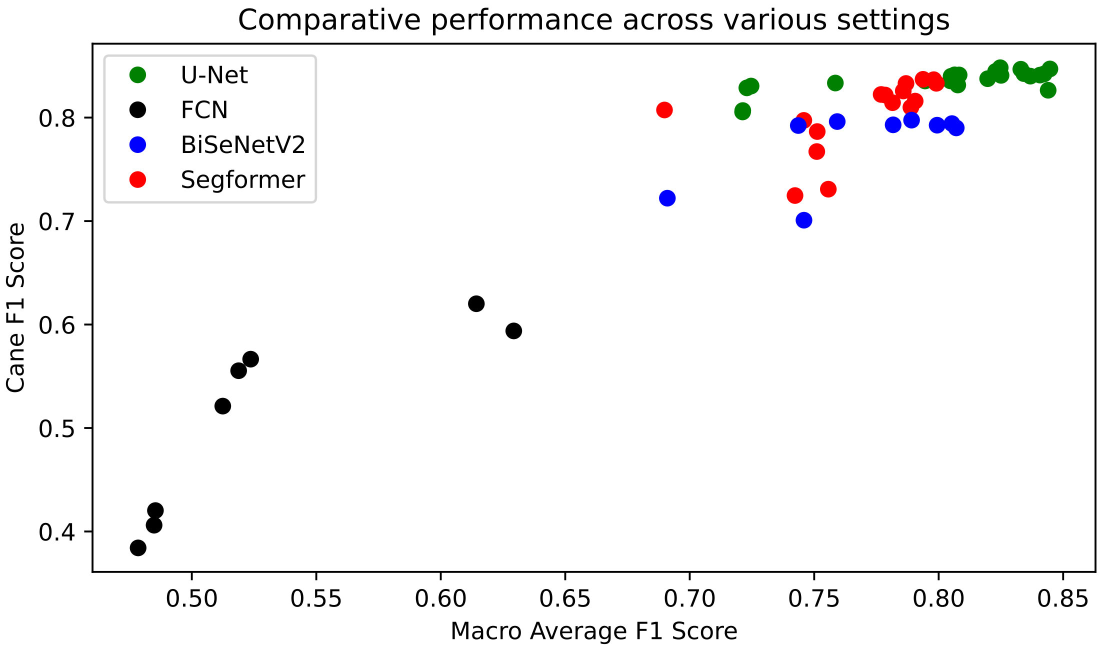


Figure 4.6: Segmentation architecture comparison under a variety of settings.

for machine learning is the process of artificially increasing the size of a training dataset by applying various transformations or modifications to existing data. We tested the common geometric augmentations of image rotation, image resizing, and random cutouts. In addition, we tested a photometric augmentation function where the brightness, contrast, saturation, and hue were modified randomly. Results of these tests are shown in Fig. 4.7.

We found that geometric augmentations improved performance across the board. In particular, randomly resizing the input image was strongly beneficial to accuracy, presumably by helping the model learn features that are more scale-independent. Meanwhile, photometric augmentations slightly degraded performance. This is likely because the flash images have such a tight domain that photometric augmentation pushed the training images outside of the domain seen in validation/test images. For segmentation we therefore use UNet with geometric augmentations on the training images.

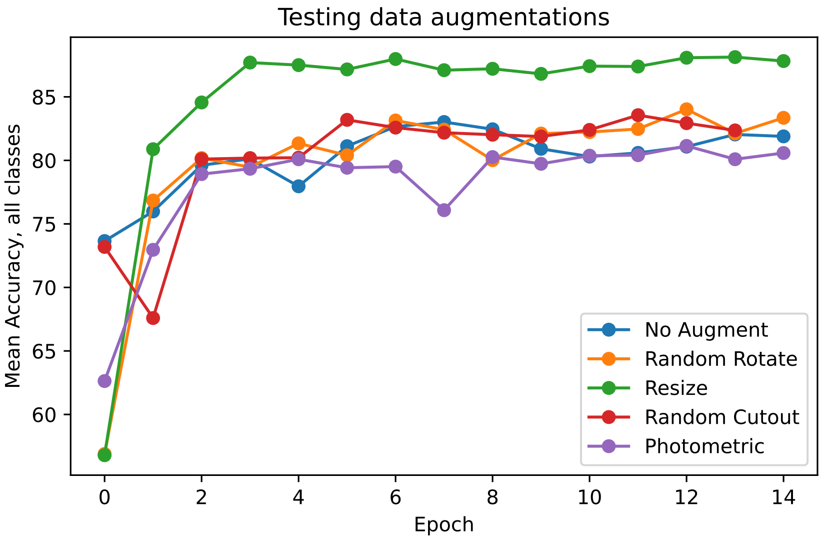


Figure 4.7: Tested a variety of image augmentations, and found that at convergence “Resize” had the largest beneficial effect and “Photometric” was the only augmentation that hurt performance.

4.4 Dilation for Precision

Using the final UNet model, we noticed a pattern in cane segmentation errors where the centers were largely correct, with errors at the cane edges. This is understandable, since getting sharp edges is a consistent issue with current segmentation architectures, but is also problematic since thin structures like canes have a significant proportion of edge per instance. Vine skeletonization is the end goal of this segmentation effort (Section 5), and in skeletoniation we only care about tracing cane centers, so we improved performance in the regions of interest by discarding cane edges.

This is done by getting the 2D skeleton of the segmentation using [58], growing the 2D skeleton with dilation, then taking the intersection of the dilated shape with the original so we do not expand outside the original mask. Note that we are not eroding the cane mask by a set number of pixels, which would potentially erase small canes entirely, but are instead growing outwards from the 2D skeleton so that small lines are preserved. Precision improvements were assessed on test set images in Fig. 4.8. By using a 3-pixel dilation, we improve precision by 4.6% while keeping a functionally reasonable number of points available for processing, as shown in Fig. 4.9.

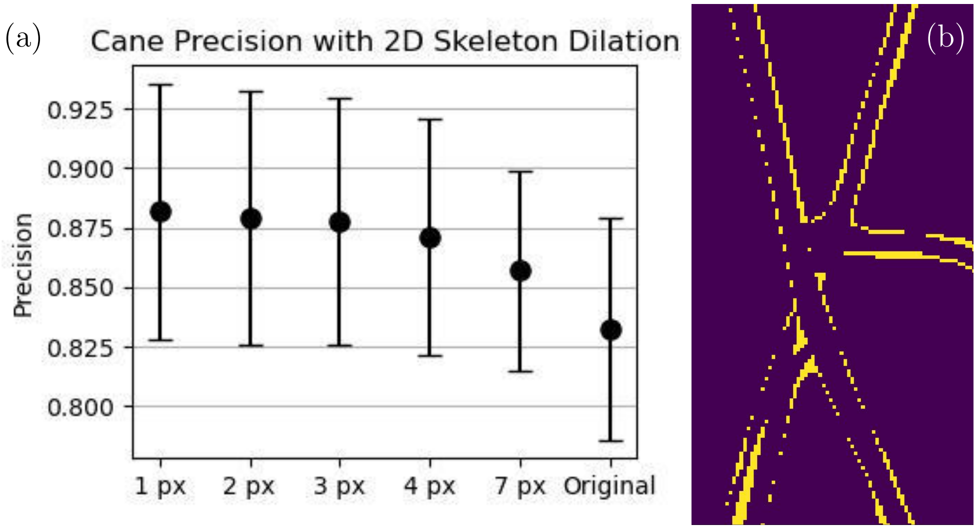
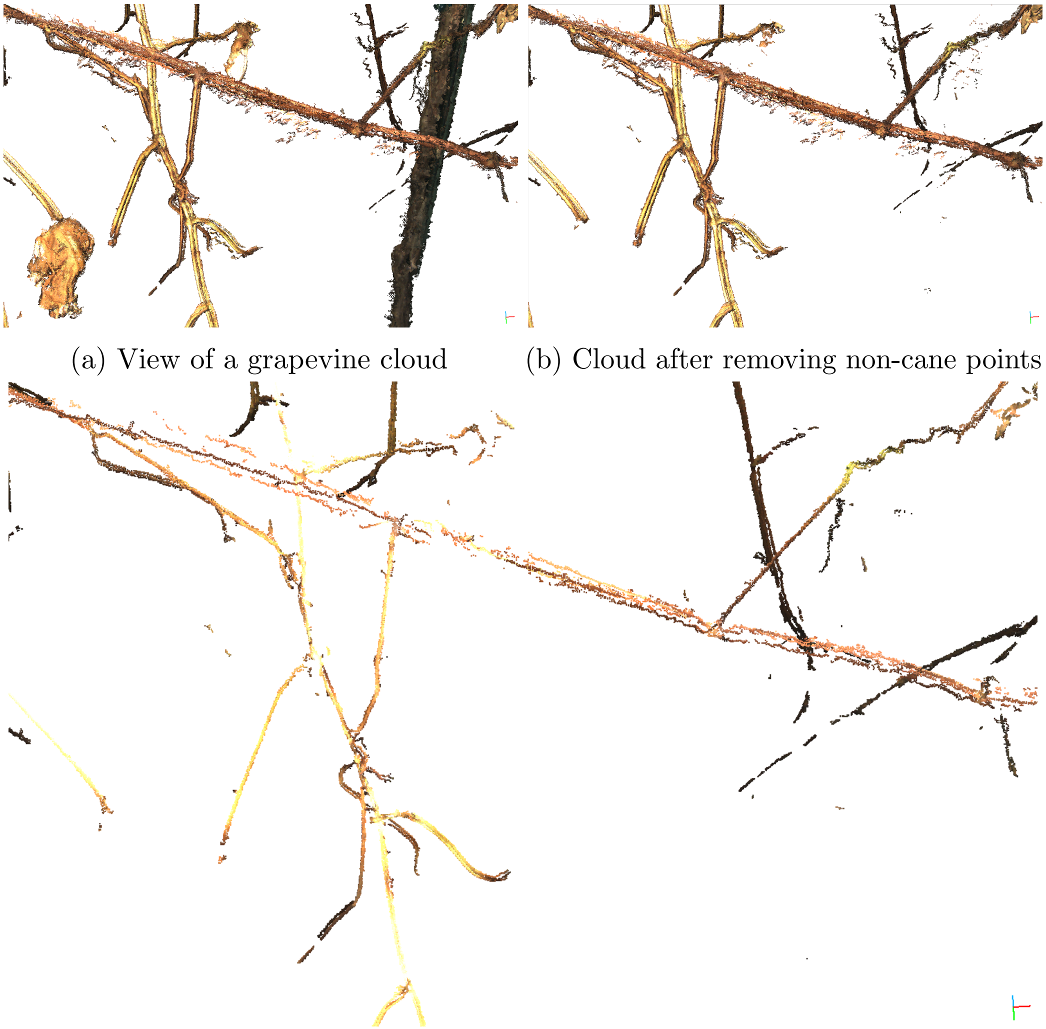


Figure 4.8: (a) Improving cane precision by discarding edge predictions, showing the mean and std. dev. From the original to the 3px cutoff the precision improved from 0.832 to 0.878, +4.6%. 1px on the x-axis means that the most edge points were removed, whereas Original does not discard any points. For this dataset 3px appears to be good compromise between getting more points to work with while keeping precision high. (b) Qualitative example of segmentation error (yellow indicates classification error), most errors are at the cane edge.

4.5 Conclusion

In the end, we use UNet trained with geometric augmentations as the cane segmentation model. The precision of the cane segmentation is then improved before use by discarding mask edges, allowing us to just trace points along the cane centers. This segmentation model is used as an early building block for Section 5. Once model inference is run to create segmentation masks on one image in the stereo pair, we project these masks onto the stereo point clouds and use clouds consisting only of cane points for skeletonization.



(a) View of a grapevine cloud

(b) Cloud after removing non-cane points

(c) Point cloud after dilation for precision improvements have been applied. As can be seen there are still some reconstruction issues, such as the separated cane portions running parallel, but a great deal of edge noise is eliminated.

Figure 4.9: Illustration of the filtering effects of (b) class segmentation and (c) dilation on the reconstructed vine point cloud.

Chapter 5

Grapevine Skeletonization

5.1 Task

In order to robotically prune vines, an accurate and connected model of the vine is necessary. When pruning, this model has multiple uses - the robot must first choose cut locations based on pruning rules, then plan a motion path to those cut locations while avoiding potentially damaging collisions. In this section, our main objective is to generate high-quality skeletons of vines that are more dense and complex than in previous works. The skeletal models consist of line segments in 3D space representing the center of the cane, along with associated radii. The baselines and the presented method are designed to process a point cloud that has been semantically cropped to retain only vine points, and from that generate a skeletal model. After skeletonization we discuss one potential use case, which combines 3D and skeletal data to predict pruning weight (a measure of vine vigor) for a given vine.

Key Terminology

Balanced pruning: method of pruning more or less of a vine based on vine vigor, aimed at balancing yield, quality, and growth
Allometric growth: in plants, the consistent patterns of growth, especially for proportions and sizes
Pruning weight: mass of all canes *less than one year old* cut off a single vine

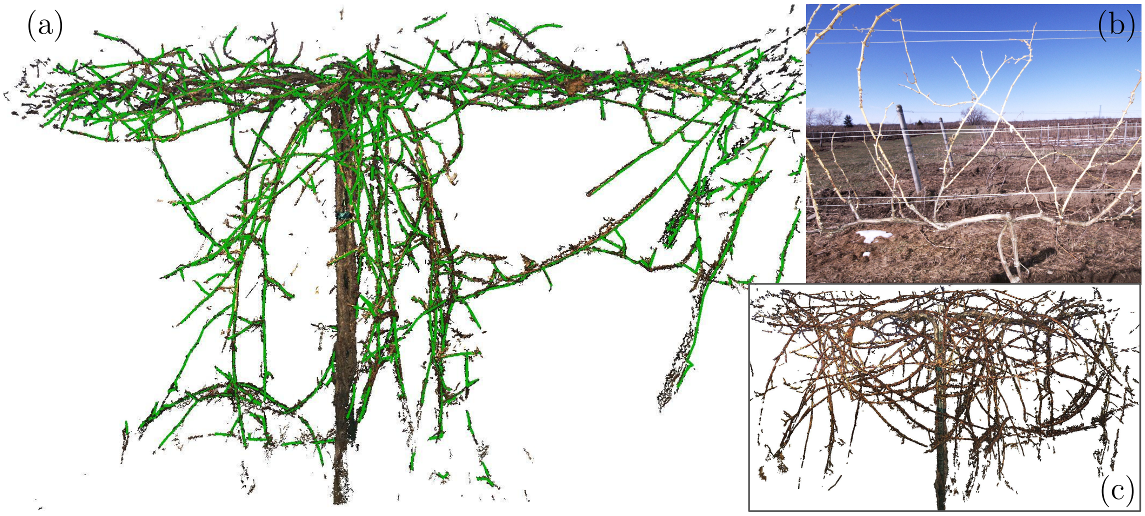


Figure 5.1: (a) Point cloud data and overlaid skeleton for a typical vine in this dataset. (b) Typical simple vine used in prior robotic grapevine research. Note that vines are trained in wires to roughly grow vertically in-plane. (c) Vigorous example of vines in this dataset, scene size is {width: 3.3 m, height: 1.8 m, depth: 1.2 m}.

5.2 Dataset

For skeletonization, we use the stereo images described in Section 4.2. In addition, we use our skeletons to assess pruning weight, a measure of vine health. In order to get ground-truth pruning weight data, collaborators at the Cornell Lake Erie Research and Extension Laboratory captured pruning weight measurements for all 144 imaged vines. For a subset of 30 vines they also captured number of buds removed, length removed, and number of buds remaining during the pruning process.

5.3 Skeletonization Approach

We present a pipeline (Fig. 5.2) which takes a set of stereo images of a vine, registers the images to create a unified point cloud, and generates segmentation masks using the trained model from Section 4. After the segmentation masks are projected onto the point cloud, the resulting vine-only cloud is used to build a skeletal model. We propose a modification to typical graph-and-refine skeletonization strategies that handles cycles in the structure graph, allowing more accurate skeletal models.

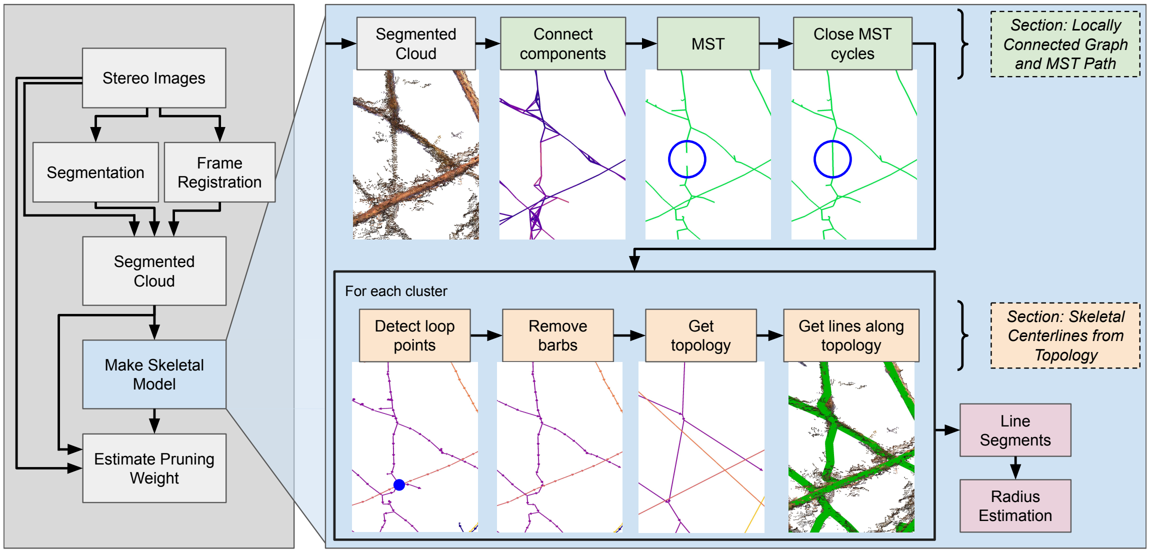


Figure 5.2: System diagram, showing the whole pipeline from stereo images to skeletonization and finally pruning weight estimation. Skeletal models can then be used for path planning for pruning operations.

5.3.1 Frame to Frame Point Cloud Registration

In order to get accurate 3D data of the full scene it is important to get good registration between cameras. After turning stereo pairs into point clouds using SGBM [18] and placing the cameras initially using robot extrinsics, we fine-tune camera positions similar to [43] by running Iterative Closest Point (ICP) between frames first horizontally and then vertically to get a final combined cloud.

We found that one particular type of noise was easily removable at this point. Since stereo depth error goes up quadratically with distance, we check regions seen by multiple cameras and discard points that come from significantly more distant (and therefore less trustworthy) viewpoints.

5.3.2 Make Skeletal Model

Our skeletonization approach draws on other graph-and-refine methods, and is essentially a two-part process. First a dense starter graph is created and a starting path is found through each connected cluster, then pathways are traced and turned into line segments. An overview of these steps with visualizations is shown in Fig. 5.2.

Locally Connected Graph and MST Path

Given a segmented point cloud, we create a locally connected graph along with an initial MST path as the starting point for skeleton generation. This approach of starting from a locally connected lattice is inspired by [11, 30, 51].

Connect components: We consider all points sufficiently close as candidates for skeletal connectivity. This is accomplished by sweeping a sphere of radius r_s across each point in a downsampled cloud, and building a graph where all points within the radius of point p_i are connected to p_i as shown in Fig. 5.3. For this dataset we chose a radius r_s=2.5 cm. Neighbor querying is accomplished efficiently using a k-d tree.

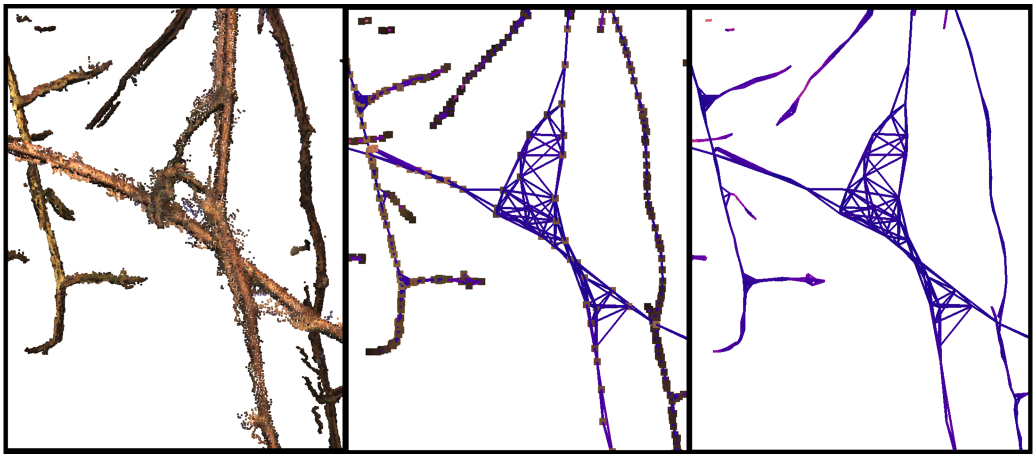


Figure 5.3: From the full point cloud (left) a locally densely connected graph (right) is constructed by sweeping a sphere of radius r_s over downsampled points (middle) and connecting to all points within its radius.

MST: For each cluster in the locally connected graph, we find an MST using the Kruskal algorithm [23], where Euclidean distance is the edge cost. This selects a path out of the locally connected graph which visits every node, shown in Fig. 5.4.

Close MST cycles: By its construction, the locally connected graph will have connected edges wherever canes pass closely, which leads to many graph loops when canes drape over each other. The MST, by its nature as a tree graph, breaks these loops while minimizing path length. However, we found that broken loops led to poor skeletons because the broken loop halves would either get pruned (**Remove barbs**) or fit as separate branches with a disconnect. In Fig. 5.5 we can see an obvious cane that is broken by the MST process, then fixed by closing MST cycles.

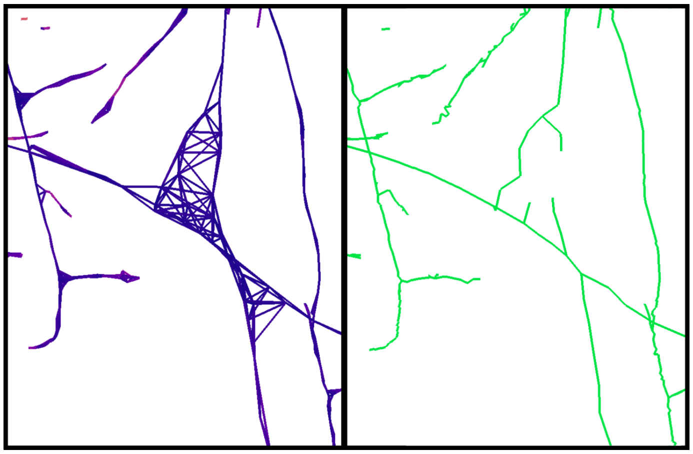


Figure 5.4: Minimum spanning tree path (right) through the locally connected graph (left), which forms a starting point for skeleton consideration.

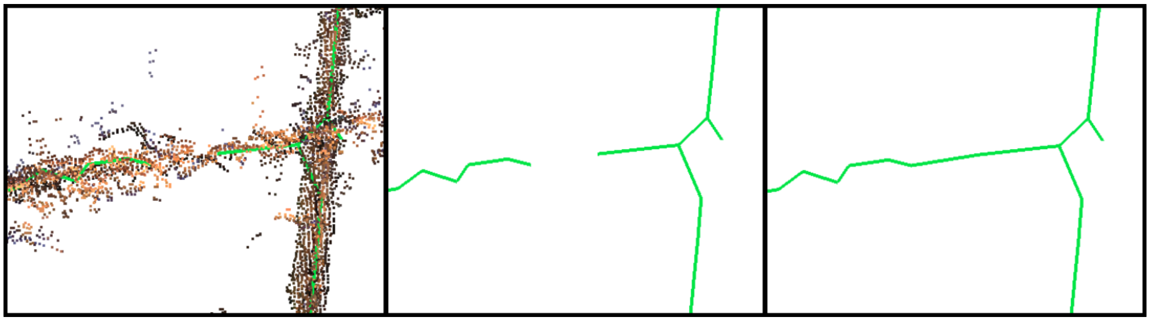


Figure 5.5: A cane, visible in the colorized point cloud (left) is broken by the Minimum Spanning Tree path (middle), since loops are not allows in a tree graph. Closing MST cycles (right) heals that break, allowing a more faithful skeleton.

We close the loops broken by the MST by finding leaves of the MST graph where a single step in the locally connected graph connects to another leaf, then adding that edge back. Because of the way the locally connected graph is constructed, we know the re-added edge will be shorter than r_s. In order to prevent nearby barbs from connecting in tiny loops, a pre-closure graph distance of δ_l is required between the two leaves.

Skeletal Centerlines from Topology

Now that we have an initial path, a series of steps are performed to generate skeletal line segments. Because the starter graph is formed by sweeping a sphere of size r_s

across the cloud, any gap larger than r_s will cause separated clusters to form. The following steps are done independently on each cluster.

Detect loop points: In order to ensure the directed graph maintains loops, loop points are detected. Any nodes that are the common endpoint of two or more directed edges are saved as a loop point, as demonstrated in Fig. 5.7(a).

Remove barbs: MSTs of the locally connected graph form long paths with small offshoots to span every node in the original graph, much like barbed wire. Inspired by [11], we remove small barbs by removing nodes where the downstream edge length is below a threshold δ_b as shown in Fig. 5.6 (δ_b = 3cm in this work).

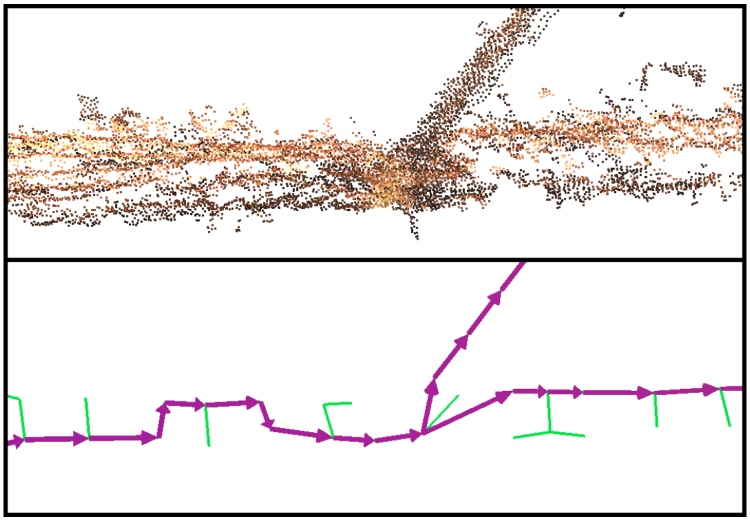


Figure 5.6: A wide/messy point cloud (top) produces an initial MST path (green line segments) with offshoots. The barbs with low downstream edge length (bottom) are removed to simplify the graph structure. The smoothed path (arrows) remains.

In order to keep nodes on either side of loop points from being eroded, edges ending at loop points have their weight temporarily increased over δ_b.

Get topology: from the smooth graph we identify isolated cane sections by eliminating all but junction/leaf nodes as illustrated in Fig. 5.7(b). All mid-nodes with only one parent and one child are discarded, leaving a topological graph where each edge represents a single cane of variable length.

In order to handle loops, loop points with two incoming edges are treated as mid-nodes to eliminate, where the shorter side of the loop is reversed so that when the loop is collapsed the topological edge reaches from one junction to the other.

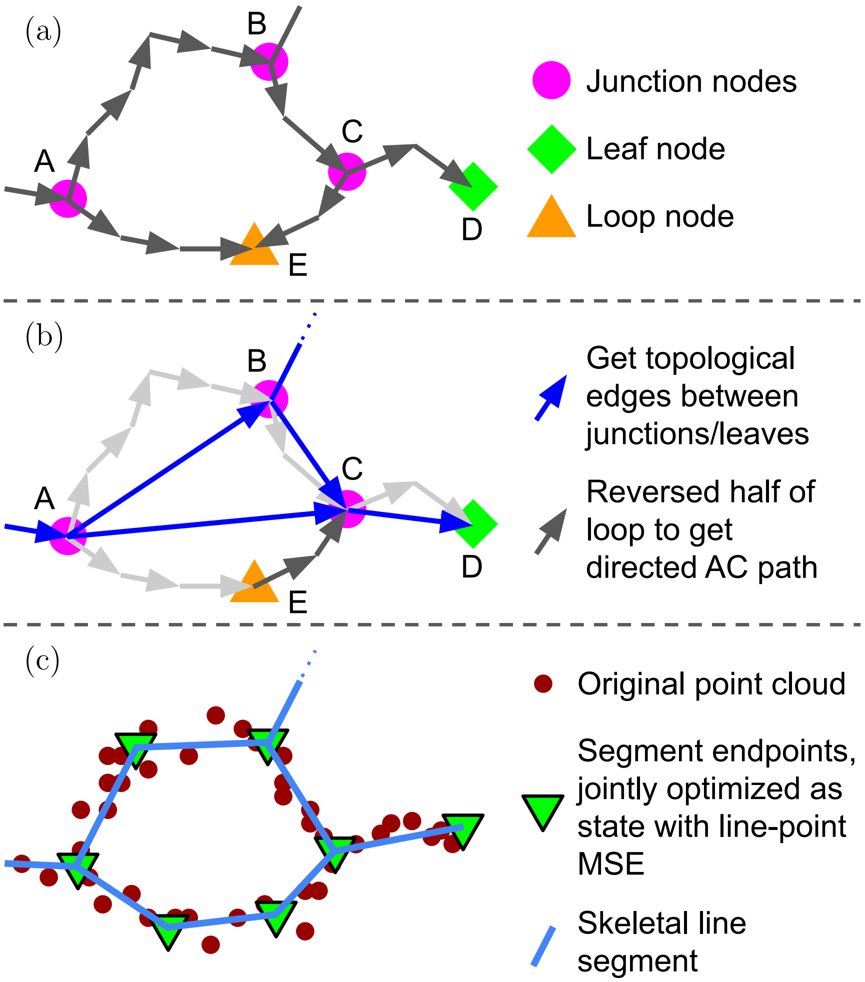

(a)
A
B
C
D
E
Junction nodes
Leaf node
Loop node
(b)
A
B
C
D
E
Get topological edges between junctions/leaves
Reversed half of loop to get directed AC path
(c)
Original point cloud
Segment endpoints, jointly optimized as state with line-point MSE
Skeletal line segment

Figure 5.7: (a) Smooth graph from **Remove barbs**, with junctions and a leaf. (b) In **Get topology**, mid-nodes are eliminated. The shorter loop half, CE, is reversed to get a continuous AC path. (c) In **Get lines along topology**, when fitting lines to the points, the state is the positions of all green triangles. As one endpoint is moved, it effects the fit of all lines connected to it, fitting the best combination.

Get lines along topology: given topological edges have been identified where each topological edge represents a non-branching stretch of cane, skeletal line segments that represent the center of the skeleton are generated. This is done by associating the original points to a single topological edge, then fitting line segments to minimize the point-to-line Mean Squared Error (MSE) as illustrated in Fig. 5.8.

The (x, y, z) values of all line endpoints form the state when concatenated, so

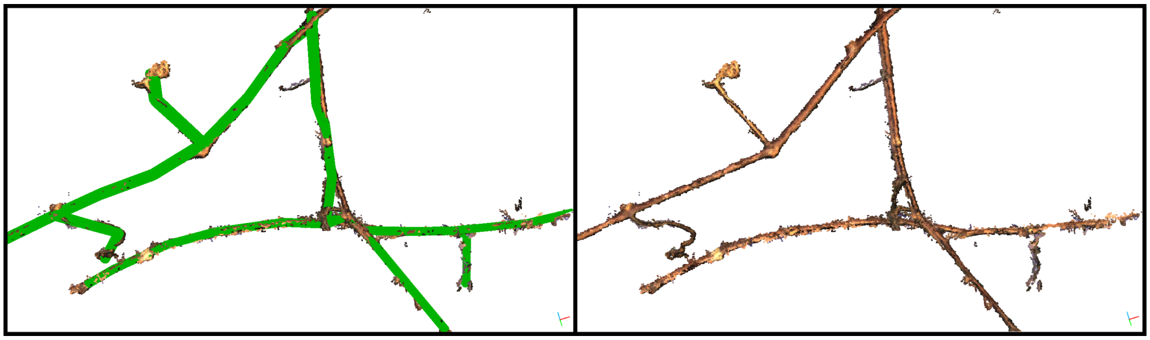


Figure 5.8: Lines are jointly optimized along the topological edges by minimizing point-to-line MSE between point cloud and skeleton as the line endpoints move. Left we have the estimated line segments with radii, right shows the underlying colorized point cloud.

batches of line ends are optimized jointly as shown in Fig. 5.7(c). It is important to optimize over batches of lines with shared endpoints to preserve connectivity. Fitting is done using the L-BFGS-B non-linear optimization method [62] in `scipy.optimize.minimize`. When the connected cluster is too large, driving the number of states past a computational threshold, this process is performed on segments of the cluster.

Radius Estimation

Finally, after finding skeletal center lines, the radii of all canes in a cluster ($r_i \in R$) are estimated jointly using a novel linear regression formulation. Three aspects are balanced to determine the radii: a prior value, a smoothing term, and point-fitting. For the prior, we set the radius for a given line r_i equal to the prior radius, r_{prior}. For smoothing, for every pair of line segments (l_i, l_j) that share a junction the radii are set equal: $r_i = r_j$. Finally, each point p_k associated with a given line segment l_i is set so that the distance δ_k from p_k to l_i is equal to r_i. Here is the linear system in matrix form:

γ_p and γ_s are weights for the prior value and smoothing terms, scaled by $k = \frac{|P|}{|R|}$, the average points per radius. The best results were with $\gamma_p = k, \gamma_s = 0.1k$, with $r_{\text{prior}} = 5$ mm.

$$
\begin{array}{l}
\text{Radius} \\
\text{Prior} \\
\\
\text{Smoothing} \\
\text{Terms} \\
\\
\\
\text{Point} \\
\text{Fitting} \\
\\
\\
\\
\end{array}
\begin{bmatrix}
\gamma_p & 0 & 0 & \cdots \\
0 & \gamma_p & 0 & \cdots \\
 & \cdots & & \\
-\gamma_s & \gamma_s & 0 & \cdots \\
-\gamma_s & 0 & \gamma_s & \cdots \\
0 & -\gamma_s & \gamma_s & \cdots \\
 & \cdots & & \\
1 & 0 & 0 & \cdots \\
1 & 0 & 0 & \cdots \\
 & \cdots & & \\
0 & 1 & 0 & \cdots \\
 & \cdots & &
\end{bmatrix}
\begin{bmatrix} r_1 \\ r_2 \\ \cdots \\ r_N \end{bmatrix}
=
\begin{bmatrix}
\gamma_p r_{\text{prior}} \\
\gamma_p r_{\text{prior}} \\
\cdots \\
0 \\
0 \\
0 \\
\cdots \\
\delta_1 \\
\delta_2 \\
\cdots \\
\delta_k \\
\cdots
\end{bmatrix}
$$

5.4 Skeletonization Results

5.4.1 Skeleton Quality Metrics

It is difficult to define a metric for measuring the correctness of a skeletonization method, especially with dense and intertwined objects. Hand-labeling ground-truth skeletal paths in 3D is infeasible, both because the true branching structure is not always clear to an observer given a sparse set of images, and because labeling accurate paths in 3D data is extremely time intensive when there are hundreds of individual canes per scene. We therefore adopt the unsupervised skeletal reconstruction metric from [3], which uses Intersection over Union (IoU) of the model projected onto a segmentation mask, defined as:

$$\frac{I}{U} = \frac{\Sigma \text{ projected model pixels} \cap \text{cane pixels}}{\Sigma \text{ projected model pixels} \cup \text{cane pixels}}$$

As shown in Fig. 5.9, IoU checks whether the skeletal model covers areas classified as cane. It is penalized both for modelling skeletal links where no cane was seen, and also for failing to model links where canes are identified. We assess only against the cane/background, discarding pixels segmented as other classes. During model projection the camera position is allowed to adjust within 5mm to find the position with greatest intersection, keeping over-generous radius predictions from dominating due to slight inaccuracies in registration.

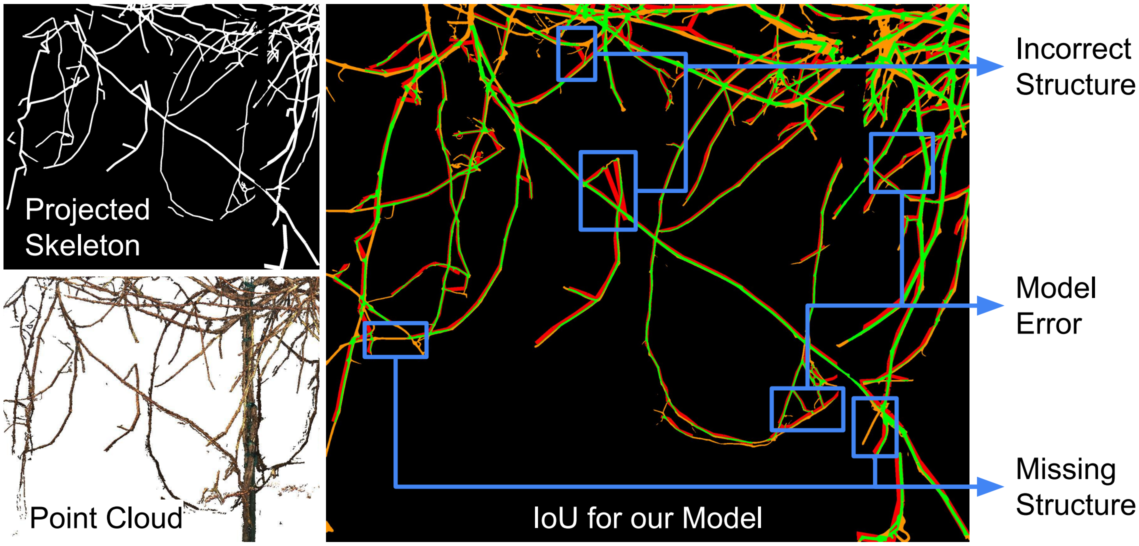


Figure 5.9: Skeleton quality is assessed by projecting the skeleton (position, radius) onto cane segmentation. **Green**: match between model, segmentation. **Orange**: cane segmentation with no projected model. **Red**: projected model with no segmentation.

The IoU-based reconstruction based metric does have some weaknesses compared to a true assessment of skeleton quality. If structure is obscured in the images (perhaps on the backside of a vine), then a skeletal model will not be penalized for failing to capture that structure. The obstruction weakness is somewhat mitigated by using multiple images that capture different views through the vine. Additionally, objects closer to the camera appear larger in the segmentation mask and are therefore over-represented in the reconstruction score. However, IoU is the simplest and most effective unsupervised quality metric available, so we use it despite the shortcomings.

In addition to IoU, we also assess the number of connected clusters. In general more clusters means a more fragmented skeleton, which provides less connectivity information.

5.4.2 Results and Comparison

Fig. 5.10 shows the results of our skeletonization method against two baselines with open source code, Laplacian Contraction [4] and AdTree [11]. We produce skeletons that recreate the visible vine structure more accurately, while providing higher vine connectivity than Laplacian Contraction. Although AdTree is more connected than our approach, in this context AdTree's assumption that all points connect is too strong, leading to forced connections to the central cordon that do not truly exist.

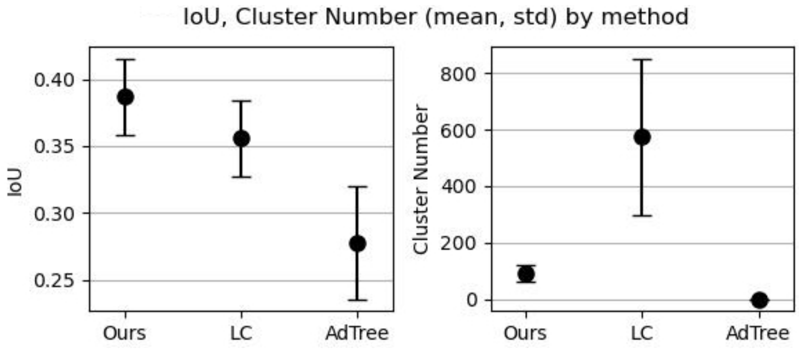


Figure 5.10: Reprojection scores and cluster count across methods.

Since Laplacian Contraction only returns contracted points, to evaluate it as a skeleton we assume each contracted point is connected to its two nearest neighbors and has a fixed radius that maximized IoU. In addition, Laplacian Contraction parameters were swept to find the set that resulted in maximum IoU. Skeletons formed by this method are relatively fragmented, with no method of joining likely paths, which makes them less useful for robotic pruning use cases. Qualitative views found in Fig. 5.11 (a-c).

AdTree represents the opposite extreme, where each vine model is assumed to consists of a single cluster, as shown in Fig. 5.12. In a complex vineyard setting, neighboring vines grow into the space, and assuming all canes connect to the central cordon degrades the IoU. In addition, AdTree uses allometric tree growth assumptions to calculate radii, and grapevines do not follow the same patterns as trees. Radii are therefore over-estimated near the cordon and unrealistic hair-like tendrils are formed at the tip. Qualitative views of AdTree skeletonization can be seen in Fig. 5.11 (d-f).

5.4.3 Timing

All of these methods are currently designed to be run offline, with our method falling between Laplacian Contraction and AdTree after minimal time optimizations. All

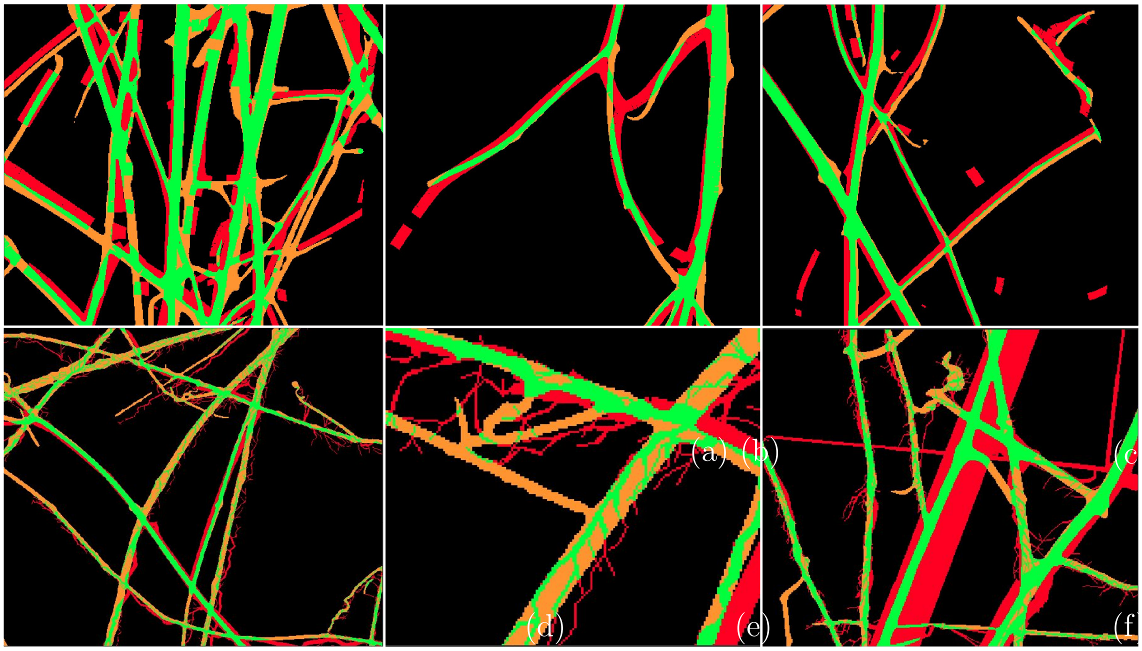


Figure 5.11: Qualitative plots, of IoU diagrams against the cane segmentation. **Green**: match between model, segmentation. **Orange**: cane segmentation with no projected model. **Red**: projected model with no segmentation. In open spaces Laplacian Contraction (a-c) is drawn correctly onto the cane structure, but junctions pull neighboring points awry (b), and it leads to fragmentation (c). AdTree (d-f) has two primary issues, it produces filaments (e) and over-estimates radii (f) due to allometric assumptions about tree growth that do not transfer well to vines.

methods are very dependent on the total number of points being processed, so in some sense the timing is somewhat arbitrary based on downsampling and filtering decisions made within each system. Evaluated over 36 vines, the average per-vine computation time for each method is:

- **Our method**: 3.7 minutes per vine
  - 1.3 minutes (Locally Connected Graph and MST Path)
  - 2.4 minutes (Skeletal Centerlines from Topology)
- **Laplacian Contraction**: 6.9 minutes per vine
- **AdTree**: 0.8 minutes per vine

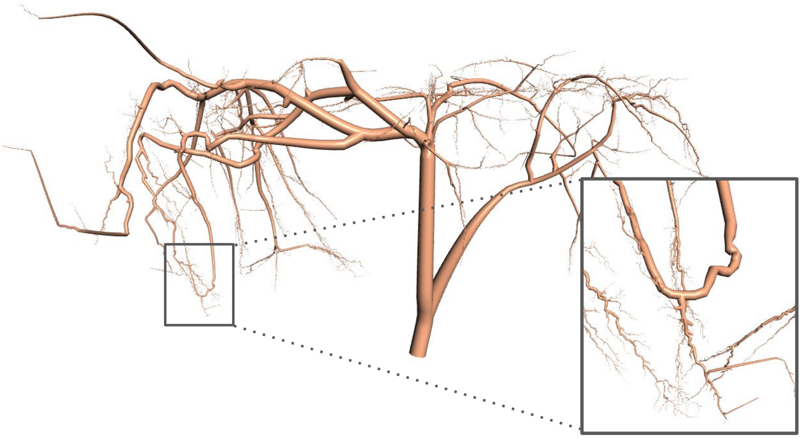


Figure 5.12: Full output of AdTree, showing allometric radius estimation, the growth of all points out from the central cordon, and the generation of hair-like structures (zoomed box). Although the model looks good, AdTree had the lowest model reconstruction scores (IoU) because it added false structure to connect all growth to the central cordon.

5.5 Use Case: Pruning Weight Estimation

One important measurement of a vine's health and vigor is pruning weight, the mass of canes less than a year old cut off during the pruning process. In order to do balanced pruning for higher quality grapes, a grower will adjust the amount of cane to remove based on the vigor of a given vine. We use 3D and skeletal data to predict pruning weight on these dense and occluded vines, comparing against values collected by human pruners.

5.5.1 Pruning Weight Estimation Method

After skeletonization, we calculate pruning weight using a simple linear regression approach on five variables available from each vine that were determined to add value to the fit. Z-score normalization is used so variable magnitudes are balanced. The variables are:

- Cane voxels: number of filled voxels after voxel downsampling the cane cloud at voxel size of 2cm.
- Cordon voxels: number of filled voxels after voxel downsampling the cordon

cloud at a voxel size of 2cm.

- Pole distance: average distance from the robot to the central pole the vine grows on.
- Skeleton length: sum of line segments in the skeleton.
- Cane pixels: total number of pixels segmented as cane.

Other variables investigated included vine brightness and number of pixels on a 2D skeletonization of the cane mask, but those were found to be unhelpful using Lasso regression, a supervised regularization method used to select useful subsets of variables. We chose a linear model because of the low number of data points. With more data a 2D or 3D learning model with higher capacity than linear regression could produce better results, but for small datasets a low capacity model is simple to implement and prevents overfitting.

5.5.2 Pruning Weight Estimation Results

Table 5.1 contains the results of our best linear model for predicting pruning weight, compared against two prior works. We assess model quality using the coefficient of determination R^2, as well as the root mean squared error (RMSE) of our predicted weight vs. ground-truth. To assess stability, we do a 100-fold assessment with a (70/30) train/test split, and report standard deviation over folds.

	$\mathbf{R^2}$		**RMSE** (kg)	
Method	Avg.	Std. Dev.	Avg.	Std. Dev.
Ours	0.51	0.10	0.33	0.03
Cane pixel count	0.33	0.10	0.39	0.04
Cane surface area	0.38	0.10	0.38	0.04

Table 5.1: We show results from our method, the correlation of cane pixel count to PW as in [33], and the correlation of cane surface area to PW as in [22].

As shown in Table 5.1, our pruning weight predictions are more accurate than prior works when run on dense vines. One grower we spoke to said RMSE of less than (0.5lbs/0.23kg) would be when they would consider a system that predicts pruning weight operationally viable. Our system approaches that functional level but still needs improvement.

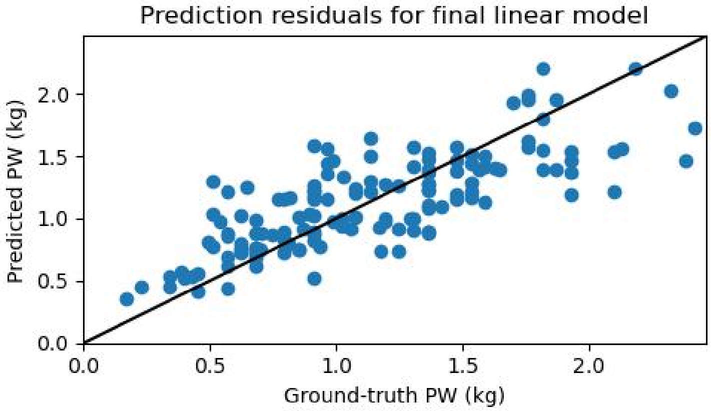


Figure 5.13: Pruning weight residual plot showing distribution of prediction vs. ground-truth. The linear model weights are {Cane voxels: 0.674, Cordon voxels: -0.061, Pole distance: -0.086, Skeleton length: -0.354, Cane pixels: 0.140}. We can see that there isn't any significant pattern to the residuals, which would be a red flag.

We explore the effects of dropping any one variable from the model in Fig. 5.14. Our initial assumption was that skeletal length would be a strong predictor of pruning weight, however in practice it appears to be beneficial but play a smaller predictive role than cane voxels, cordon voxels, and pole distance.

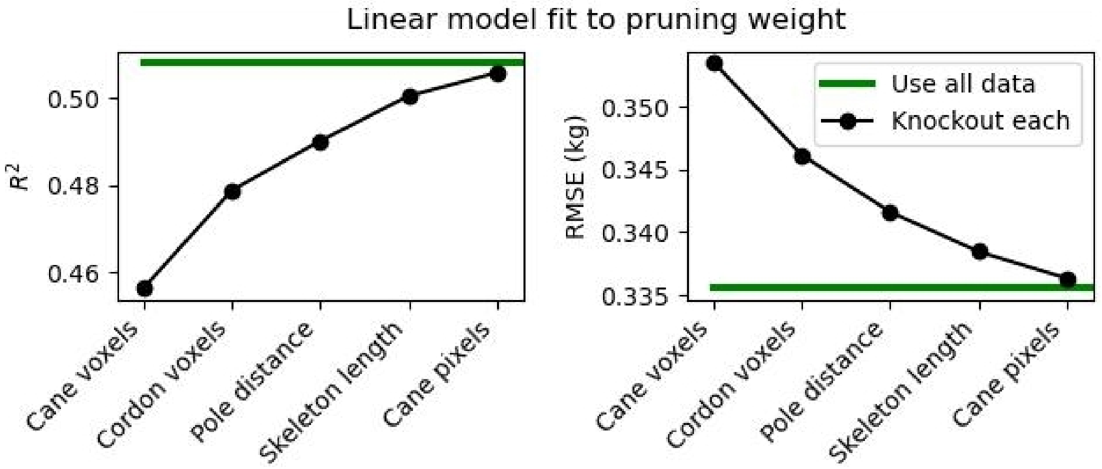


Figure 5.14: Effects on test data of dropping each variable from the linear model.

One factor that may degrade the predictive capacity of skeleton length in pruning weight estimation is neighboring vines. With this type of vine architecture and robustness of growth, it is common for canes to grow into the spaces of neighboring vines. These neighbor canes are captured in images, but when measuring pruning weight are disentangled and assigned to the vine they originated from. Thus part of the skeletal structure detected for a given vine should theoretically be assigned to its neighbors. Pruning weight prediction could be improved if a reliable method of attributing visible canes to the correct source were developed.

Chapter 6

Conclusions

In this work we propose two agricultural analysis pipelines. The first pipeline counts sorghum seeds in a 3D reconstruction process, using seeds as landmarks in reconstruction and a density maxima-finding method for counting. The second pipeline creates 3D skeletal models of dormant grapevines using a graph-and-refine skeletonization strategy on semantically segmented point clouds. We make the following contributions towards agricultural modeling and analysis:

- An end-to-end pipeline for non-destructive sorghum seed counting, along with a related public dataset.
- A new unsupervised quality metric for reconstructed point clouds, developed in order to improve 3D reconstruction quality and refine seed count.
- An end-to-end pipeline for skeletonization of dense and complex dormant grapevines, along with a related public dataset.
- A modification to graph-and-refine skeletonization strategies that handles cycles in the structure graph, allowing more accurate skeletal models.
- State of the art automated pruning weight prediction results.

In the end we were able to able to create higher quality sorghum models than the baseline methods using high-confidence seed points as landmarks, and are able to calculate seed counts highly correlated to the ground-truth count (R^2 of 0.875). For grapevine skeletonization we achieve higher quality reconstructions than the baselines, measured using an unsupervised IoU metric. In addition, we achieve an R^2 value of

0.51 for pruning weight prediction, compared to 0.33/0.38 for previous methods.

Throughout this entire work, the overarching theme is a lack of labelled training data. Machine learning models had to be trained with minimal data, and unsupervised quality metrics were employed both in the sorghum and grapevine pipelines. Agricultural data is generally scarce because outdoor scenes are complicated and organic, the plants of interest are not human-made and therefore lack source design models, and large commercial interests have not built up datasets as has happened in self-driving. Given these challenges, it is evident this field must continue to adapt and innovate to overcome the scarcity, and we hope that this work contributes to that goal.

Chapter 7

Future Work

For the sorghum pipeline, the most obvious deficiency is the large amount of superfluous data required by the pipeline - the use of many views (typically ≈ 80) is an intensive process. It would be worthwhile to create methods to find the minimal image set that could reliably create a high-quality model, reducing runtime and resource requirements. Dense panicle models could also be put to other uses. In addition to extracting counts, other phenotyping or health characteristics could be evaluated, perhaps based on crop volume, color, or texture.

One fact that has become clear during the development of the skeletonization pipeline is that while it is useful to represent the location of canes as we do with skeletal links, in reality all canes must grow from some source, and knowledge of plant growth patterns gives insight into which source a given cane comes from. In future work, developing a new type of plant skeletal model based on growth sources and likely growth pathways, along with methods to accurately construct those models, would be beneficial not just for pruning but for a variety of robotics challenges that deal with the dynamics and manipulability of plants, such as harvesting, grafting, and pollinating.

One complication with tracing vine pathways based on growth in a crowded vineyard arises from the fact that neighboring vines greatly increase the possible growth-source locations. Instead of assuming that canes must come from a central source such as the visible cordon, any cane touching the edge of the imaged scene may have grown horizontally from a neighbor. A growth-tracing approach that could evaluate both the central cordon and neighboring vines as possible sources would add great deal of value to a robot's understanding of each vine.

Finally, for pruning weight prediction, it would be promising for future work to use higher capacity 2D or 3D learning methods, which would necessitate larger amounts of training data. As robotic pruning gets more mature, it is not preposterous to assume that robots could eventually predict pruning weight, prune the vine, collect the ground-truth pruning weight, and finally close the loop to improve the predictive performance in a beneficial data collection cycle.

Bibliography

[1] Abbas Atefi, Yufeng Ge, Santosh Pitla, and James Schnable. Robotic technologies for high-throughput plant phenotyping: Contemporary reviews and future perspectives. *Frontiers in Plant Science*, 12, 2021. ISSN 1664-462X. doi: 10.3389/fpls.2021.611940. URL https://www.frontiersin.org/articles/10.3389/fpls.2021.611940. (document)

[2] Yin Bao, Lie Tang, Matthew W. Breitzman, Maria G. Salas Fernandez, and Patrick S. Schnable. Field-based robotic phenotyping of sorghum plant architecture using stereo vision. *Journal of Field Robotics*, 36(2):397–415, 2019. doi: https://doi.org/10.1002/rob.21830. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21830. 2.2

[3] Tom Botterill, Scott Paulin, Richard Green, Samuel Williams, Jessica Lin, Valerie Saxton, Steven Mills, XiaoQi Chen, and Sam Corbett-Davies. A robot system for pruning grape vines. *Journal of Field Robotics*, 34(6):1100–1122, 2017. doi: https://doi.org/10.1002/rob.21680. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21680. (document), 1.1, 2.4, 5.4.1

[4] Junjie Cao, Andrea Tagliasacchi, Matt Olson, Hao Zhang, and Zhinxun Su. Point cloud skeletons via laplacian based contraction. In *2010 Shape Modeling International Conference*, pages 187–197, 2010. doi: 10.1109/SMI.2010.25. 2.4, 5.4.2

[5] Ayan Chaudhury and Christophe Godin. Skeletonization of plant point cloud data using stochastic optimization framework. *Frontiers in Plant Science*, 11, 2020. ISSN 1664-462X. doi: 10.3389/fpls.2020.00773. URL https://www.frontiersin.org/articles/10.3389/fpls.2020.00773. 2.4

[6] Ayan Chaudhury and Christophe Godin. Skeletonization of plant point cloud data using stochastic optimization framework. *Frontiers in Plant Science*, 11, 2020. ISSN 1664-462X. doi: 10.3389/fpls.2020.00773. URL https://www.frontiersin.org/articles/10.3389/fpls.2020.00773. 2.1

[7] Nived Chebrolu, Philipp Lottes, Alexander Schaefer, Wera Winterhalter, Wolfram Burgard, and Cyrill Stachniss. Agricultural robot dataset for plant classification,

localization and mapping on sugar beet fields. *The International Journal of Robotics Research*, 36:027836491772051, 07 2017. doi: 10.1177/0278364917720510. 2.1

[8] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5556–5565, 2015. doi: 10.1109/CVPR.2015.7299195. 3.3.2

[9] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. https://github.com/open-mmlab/mmsegmentation, 2020. 4.3.1

[10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4.1

[11] Shenglan Du, Roderik Lindenbergh, Hugo Ledoux, Jantien Stoter, and Liangliang Nan. Adtree: Accurate, detailed, and automatic modelling of laser-scanned trees. *Remote Sensing*, 11(18), 2019. ISSN 2072-4292. doi: 10.3390/rs11182074. URL https://www.mdpi.com/2072-4292/11/18/2074. 2.4, 5.3.2, 5.3.2, 5.4.2

[12] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of 2nd International Conference on Knowledge Discovery and*, pages 226–231, 1996. 3.3.3

[13] A. Feng, H. Li, Z. Liu, Y. Luo, H. Pu, B. Lin, and T. Liu. Research on a Rice Counting Algorithm Based on an Improved MCNN and a Density Map. *Entropy (Basel)*, 23(6), Jun 2021. 1.2, 2.2

[14] Miguel Fernandes, Antonello Scaldaferri, Giuseppe Fiameni, Tao Teng, Matteo Gatti, Stefano Poni, Claudio Semini, Darwin G. Caldwell, and Fei Chen. Grapevine winter pruning automation: On potential pruning points detection through 2d plant modeling using grapevine segmentation. *CoRR*, abs/2106.04208, 2021. URL https://arxiv.org/abs/2106.04208. 1.1, 2.4

[15] Sambuddha Ghosal, Bangyou Zheng, Scott C. Chapman, Andries B. Potgieter, David R. Jordan, Xuemin Wang, Asheesh K. Singh, Arti Singh, Masayuki Hirafuji, Seishi Ninomiya, Baskar Ganapathysubramanian, Soumik Sarkar, and Wei Guo. A weakly supervised deep learning framework for sorghum head detection and counting. *Plant Phenomics*, 2019, 2019. doi: 10.34133/2019/1525874. URL https://spj.science.org/doi/abs/10.34133/2019/1525874. (document), 2.3

[16] Dario Gogoll, Philipp Lottes, Jan Weyler, Nik Petrinic, and C. Stachniss. Unsu-

pervised domain adaptation for transferring plant classification systems to new field environments, crops, and robots. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2636–2642, 2020. (document), 2.3

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. URL `http://arxiv.org/abs/1703.06870`. 2.3

[18] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008. doi: 10.1109/TPAMI.2007.1166. 5.3.1

[19] Leland Ralph House. A guide to sorghum breeding. *ICRISAT*, 1985. 1.2

[20] G. Stanley Howell. Sustainable grape productivity and the growth-yield relationship: A review. *American Journal of Enology and Viticulture*, 52(3): 165–174, 2001. ISSN 0002-9254. doi: 10.5344/ajev.2001.52.3.165. URL `https://www.ajevonline.org/content/52/3/165`. 1.1

[21] Saeed Khaki, Hieu Pham, Ye Han, Andy Kuhl, Wade Kent, and Lizhi Wang. Deepcorn: A semi-supervised deep learning method for high-throughput image-based corn kernel counting and yield estimation. *Knowledge-Based Systems*, 218: 106874, 2021. 1.2, 2.2, 3.6.1

[22] A. Kicherer, M. Klodt, S. Sharifzadeh, D. Cremers, R. Töpfer, and K. Herzog. Automatic image-based determination of pruning mass as a determinant for yield potential in grapevine management and breeding. *Australian Journal of Grape and Wine Research*, 23(1):120–124, 2017. doi: https://doi.org/10.1111/ajgw.12243. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/ajgw.12243`. (document), 2.5, 5.1

[23] Joseph B Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50, 1956. 5.3.2

[24] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *CVPR*, 2020. 3.3.1

[25] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944. 3.3.2

[26] Yue Li, Jingdun Jia, Li Zhang, Abdul Mateen Khattak, Shi Sun, Wanlin Gao, and Minjuan Wang. Soybean seed counting based on pod image using two-column convolution neural network. *IEEE Access*, 7:64177–64185, 2019. doi: 10.1109/ACCESS.2019.2916931. 1.2, 2.2

[27] Cheng Lin, Changjian Li, Yuan Liu, Nenglun Chen, Yi-King Choi, and Wenping Wang. Point2skeleton: Learning skeletal representations from point clouds.

CoRR, abs/2012.00230, 2020. URL https://arxiv.org/abs/2012.00230. 2.4

[28] Lahav Lipson, Zachary Teed, and Jia Deng. RAFT-Stereo: Multilevel Recurrent Field Transforms for Stereo Matching. *arXiv preprint arXiv:2109.07547*, 2021. 3.3

[29] Xu Liu, Steven W Chen, Chenhao Liu, Shreyas S Shivakumar, Jnaneshwar Das, Camillo J Taylor, James Underwood, and Vijay Kumar. Monocular camera based fruit counting and mapping with semantic data association. *IEEE Robotics and Automation Letters*, 4(3):2296–2303, 2019. 2.2

[30] Yotam Livny, Feilong Yan, Matt Olson, Baoquan Chen, Hao Zhang, and Jihad El-Sana. Automatic reconstruction of tree skeletal structures from point clouds. In *ACM SIGGRAPH Asia 2010 Papers*, SIGGRAPH ASIA '10, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450304399. doi: 10.1145/1866158.1866177. URL https://doi.org/10.1145/1866158.1866177. 2.4, 5.3.2

[31] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038, 2014. URL http://arxiv.org/abs/1411.4038. 2.3, 4.3.1

[32] Andres Milioto, Philipp Lottes, and Cyrill Stachniss. Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in cnns. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2229–2235, 2018. doi: 10.1109/ICRA.2018.8460962. (document)

[33] Borja Millan, Maria Paz Diago, Arturo Aquino, Fernando Palacios, and Javier Tardaguila. Vineyard pruning weight assessment by machine vision: towards an on-the-go measurement system: This article is published in cooperation with the 21th giesco international meeting, june 23-28 2019, thessaloniki, greece. guests editors : Stefanos koundouras and laurent torregrosa. *OENO One*, 53, May 2019. doi: 10.20870/oeno-one.2019.53.2.2416. URL https://oeno-one.eu/article/view/2416. (document), 2.5, 5.1

[34] Anjana K Nellithimaru and George A Kantor. Rols: Robust object-level slam for grape counting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 2.2

[35] Stephen T. Nuske, Kyle Wilshusen, Supreeth Achar, Luke Yoder, Srinivasa G. Narasimhan, and Sanjiv Singh. Automated visual yield estimation in vineyards. *J. Field Robotics*, 31:837–860, 2014. 2.2

[36] Nicholas Ohi, Kyle Lassak, Ryan Watson, Jared Strader, Yixin Du, Chizhao Yang, Gabrielle Hedrick, Jennifer Nguyen, Scott Harper, Dylan Reynolds, Cagri Kilic, Jacob Hikes, Sarah Mills, Conner Castle, Benjamin Buzzo, Nicole Waterland,

Jason Gross, Yong-Lak Park, Xin Li, and Yu Gu. Design of an autonomous precision pollination robot. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7711–7718, 2018. doi: 10.1109/IROS.2018.8594444. 2.1

[37] Ciro Potena, Raghav Khanna, Juan Nieto, Roland Siegwart, Daniele Nardi, and Alberto Pretto. Agricolmap: Aerial-ground collaborative 3d mapping for precision farming. *IEEE Robotics and Automation Letters*, 4(2):1085–1092, 2019. doi: 10.1109/LRA.2019.2894468. 2.1

[38] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015. URL http://arxiv.org/abs/1506.02640. 2.3

[39] Gianmarco Roggiolani, Federico Magistri, Tiziano Guadagnino, Jan Weyler, Giorgio Grisetti, Cyrill Stachniss, and Jens Behley. On domain-specific pre-training for effective semantic perception in agricultural robotics, 2023. (document), 2.3

[40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. URL http://arxiv.org/abs/1505.04597. 2.3, 4.3.1

[41] Pravakar Roy, Wenbo Dong, and Volkan Isler. Registering reconstructions of the two sides of fruit tree rows. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9. IEEE, 2018. 2.1

[42] Abhisesh Silwal, Tanvir Parhar, Francisco Yandun, Harjatin Baweja, and George Kantor. A robust illumination-invariant camera system for agricultural applications. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3292–3298, 2021. doi: 10.1109/IROS51168.2021.9636542. (document), 3.2, 3.7, 4.1, 4.2, 4.3

[43] Abhisesh Silwal, Francisco Yandún, Anjana K. Nellithimaru, Terry Bates, and George Kantor. Bumblebee: A path towards fully autonomous robotic vine pruning. *CoRR*, abs/2112.00291, 2021. URL https://arxiv.org/abs/2112.00291. (document), 1.1, 2.1, 2.4, 4.2, 4.2, 5.3.1

[44] Richard Smart, Mike Robinson, et al. *Sunlight into wine: a handbook for winegrape canopy management.* Winetitles, 1991. 1.2

[45] Madeleine Stein, Suchet Bargoti, and James Underwood. Image based mango fruit detection, localisation and yield estimation using multiple view geometry. *Sensors*, 16(11):1915, 2016. 2.2

[46] James A Taylor and Terence R Bates. Sampling and estimating average pruning weights in concord grapes. *American Journal of Enology and Viticulture*, 63(4): 559–563, 2012. 1.2

[47] Sebastian Varela, Taylor Pederson, Carl J. Bernacchi, and Andrew D. B. Leakey. Understanding growth dynamics and yield prediction of sorghum using high temporal resolution uav imagery time series and machine learning. *Remote Sensing*, 13(9), 2021. ISSN 2072-4292. doi: 10.3390/rs13091763. URL https://www.mdpi.com/2072-4292/13/9/1763. 2.2

[48] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861. 3.4

[49] Henry Williams, Mahla Nejati, Salome Hussein, Nicky Penhall, Jong Yoon Lim, Mark Hedley Jones, Jamie Bell, Ho Seok Ahn, Stuart Bradley, Peter Schaare, Paul Martinsen, Mohammad Alomar, Purak Patel, Matthew Seabright, Mike Duke, Alistair Scarfe, and Bruce MacDonald. Autonomous pollination of individual kiwifruit flowers: Toward a robotic kiwifruit pollinator. *Journal of Field Robotics*, 37(2):246–262, 2020. doi: https://doi.org/10.1002/rob.21861. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21861. (document)

[50] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *arXiv preprint arXiv:2105.15203*, 2021. 4.3.1

[51] Hui Xu, Nathan Gossett, and Baoquan Chen. Knowledge and heuristic-based modeling of laser-scanned trees. *ACM Trans. Graph.*, 26(4):19–es, oct 2007. ISSN 0730-0301. doi: 10.1145/1289603.1289610. URL https://doi.org/10.1145/1289603.1289610. 2.4, 5.3.2

[52] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, and Karan Singh. Predicting animation skeletons for 3d articulated models via volumetric nets. *CoRR*, abs/1908.08506, 2019. URL http://arxiv.org/abs/1908.08506. 2.4

[53] Alexander You, Fouad Sukkar, Robert Fitch, Manoj Karkee, and Joseph R. Davidson. An efficient planning and control framework for pruning fruit trees. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3930–3936, 2020. doi: 10.1109/ICRA40945.2020.9197551. 2.4

[54] Alexander You, Cindy Grimm, Abhisesh Silwal, and Joseph R. Davidson. Semantics-guided skeletonization of upright fruiting offshoot trees for robotic pruning. *Computers and Electronics in Agriculture*, 192:106622, 2022. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2021.106622. URL https://www.sciencedirect.com/science/article/pii/S0168169921006396. 2.4

[55] Sierra N. Young, Erkan Kayacan, and Joshua M. Peschel. Design and field evaluation of a ground robot for high-throughput phenotyping of energy sorghum. *Precision Agriculture*, 20(4):697–722, 2019. doi: https://doi.org/10.1007/s11119-018-9601-6. URL https://link.springer.com/article/10.

1007/s11119-018-9601-6. 2.2

[56] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 325–341, 2018. 4.3.1

[57] Guoxian Zhang and Yangquan Chen. A metric for evaluating 3d reconstruction and mapping performance with no ground truthing. In *ICIP*, 2021. 3.4

[58] T. Y. Zhang and C. Y. Suen. A fast parallel algorithm for thinning digital patterns. *Commun. ACM*, 27(3):236–239, mar 1984. ISSN 0001-0782. doi: 10.1145/357994.358023. URL https://doi.org/10.1145/357994.358023. 4.4

[59] Xu Zhao, Rui Wu, Zhong Zhou, and Wei Wu. A new metric for measuring image-based 3d reconstruction. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 1030–1033, 2012. 3.4

[60] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5122–5130, 2017. doi: 10.1109/CVPR.2017.544. 4.1

[61] Hongyu Zhou, Xing Wang, Wesley Au, Hanwen Kang, and Chao Chen. Intelligent robots for fruit harvesting: Recent developments and future challenges. *Precision Agriculture*, 23(5):1856–1907, 2022. (document)

[62] Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw.*, 23(4):550–560, dec 1997. ISSN 0098-3500. doi: 10.1145/279232.279236. URL https://doi.org/10.1145/279232.279236. 5.3.2