ROLS : Robust Object-level SLAM for grape counting

Anjana K Nellithimaru, George A Kantor
Carnegie Mellon University
anjanak@andrew.cmu.edu , kantor@ri.cmu.edu

Abstract

Camera based Simultaneous Localization and Mapping (SLAM) in an agricultural field can be used by crop growers to count fruits and estimate yield. It is challenging due to dynamics, illumination conditions and limited texture inherent in an outdoor environment. We propose a pipeline that combines the recent advances in deep learning with traditional 3D processing techniques to achieve fast and accurate SLAM in vineyards. We use images captured by a stereo camera and their 3D reconstruction to detect objects of interest and divide them into classes: grapes, leaves and branches. The accuracy of these detections is improved by leveraging information about objects' local neighborhood in 3D.

We achieve a F1 score of 0.977 with ground truth grape counts from images. Our method builds a dense 3D model of the scene with a localization accuracy in centimeters without any assumption of constant illumination conditions or scene dynamics. This method can be easily generalized to other crops such as oranges and apples with minor modifications in the pipeline.

1. Introduction

Modelling and mapping outdoor agricultural environments has applications ranging from crop monitoring to yield estimation, especially in a high value crop like grapes. Characterization at plant level is very important to track the dynamics of crop development. Producing 30,000 dollars worth of produce per acre, grapes are often the beachhead industry for modernization. Because of the large scale extent of a typical vineyard as seen in Figure 1, a periodic and exhaustive manual measurement of plant characteristics is inaccurate and time consuming. Over the past years, it is largely done by interpolating the measurements from coarsely sub sampled fractions of the vineyard. This results in a poor estimation as growth patterns may vary across the vineyard due to factors such as soil characteristics, incident radiation angle, light interception and irrigation practises.

Recently, several researchers have focused on developing both airborne and ground based robotic solutions to high throughput phenotyping. Among them, cameras prove to be a low cost solution as opposed to its counterparts such as LIDAR, ultrasonic sensors and hyperspectral cameras. In addition, cameras provide additional information such as color and access to finer structures.

Figure 1. Aerial view of a vineyard depicts the large scale nature of plant modeling and grape counting in vineyards

Figure 2. *Left*: Closeup view of the vineyard. *Right* Imaging equipment with camera and air blower

To accurately model plant structures using a camera, high-quality 3D reconstruction/mapping of the environment is necessary. However, it is challenging because of the dynamics, like the movements of leaves due to wind, and changing illumination conditions based on cloud cover and self shadow often present in an outdoor environment. Classical 3D reconstruction and mapping approaches such as Structure from motion (SfM), Multi-View Stereo (MVS), ORB-SLAM often fail when used individually due to lack of texture in the scene or presence of dynamic objects and repetitive structures. Interestingly, recent advances in deep learning frameworks are able to detect and track objects even with dynamics and changing illumination. Although these frameworks are presumed to require a large amount of training data, by carefully designing the cost function, a framework such as Mask R-CNN [7] can be made to detect instances of new objects, say grapes, given a small number of labeled images.

We propose a camera based SLAM solution to accurately count grapes by leveraging the information from 2D as well as 3D. We demonstrate our approach on stereo images collected from a camera and air blower setup mounted on a ground vehicle shown in Figure 2. The air blower is attached below the camera to move leaves and increase fruit visibility in dense canopy regions. We adapt Mask R-CNN [7] to perform instance segmentation of grapes in images and develop a module that computes instance segmentation masks in parallel to generating disparity based point cloud. Instance masks are propagated to 3D to generate several regions of interest (ROIs) in 3D. At each of these ROIs, Singular Value Decomposition (SVD) is performed at three different scales. Based on the SVD outputs, ROIs are regrouped to include any point belonging to grape class that was missed earlier.

Merging SVD results with masks not only improves the accuracy of classification of points into grapes and non-grapes but also demarcates leaves from branches. We iteratively fit shapes to these classes and use it to estimate frame-to-frame transformation. The benefit of using fitted shapes is twofold. Firstly, it is robust to the dynamics introduced in the scene by leaf blower where other feature/intensity based SLAM techniques will perform poorly. Secondly, it limits the number of landmarks to the number of parameterized shapes thereby decreasing the optimization time for SLAM. By tuning the segmentation module and reparameterizing the shape fitting, this solution can be extended to other varieties of fruits/vegetables without any hassle.

2. Related work

As our work spans across different domains such as fruit detection and classification of plant parts, SLAM, instance segmentation, and 3D reconstruction, we divide this section accordingly.

Several camera-based systems are present in the literature which perform phenotyping tasks such as fruit detection, leaf area measurement and yield estimation. Classification of grapevines into branches, leaves and grapes with an area under the curve (AUC) of 0.94 has been achieved using color and shape features along with conditional random field by [5]. They use feature based SfM for 3D reconstruction which consumes significant time and is prone to accumulate drift and feature correspondence errors. They also require a large dataset with labels in 3D and cannot identify instances of grapes separately for the purpose of counting. A set of custom image filters and transforms have been developed in [3] to detect shoot and leaves in vineyard, but they do not focus on 3D modeling or fruit detection.

There are different image-based algorithms to detect fruits and estimate yield which assume the scene to be static. A fruit counting pipeline which combines image segmentation, frame to frame tracking, and 3D localization to count visible fruits is presented in [11]. They achieve a count error of around 3% mean and 4% standard deviation for apples. [16] detect and count grapes from images using shape and visual texture features. Although they can predict yield within 9.8% of actual crop weight, it involves calibrating their counts based on ground truth count over a small sample of vineyard. [17] present a keypoint based algorithm to detect round fruits like apples and grapes from images which achieves a F1 score of 0.82.

Current state-of-the-art SLAM systems like ORB-SLAM2 [13], LSD-SLAM [6] do not leverage semantic information. DeepSLAM [4] aims at finding a small set of geometrically stable point features from images that can be directly used in SLAM. However, recent advances in object-detectors and semantic mapping networks have made it possible for researchers to use classification, detection and segmentation in SLAM [2, 12, 15, 19]. Semantic SfM [2] uses points, object detections and semantic regions to impose geometrical constraints on estimation of camera and object pose.

Fusion++ [12] presents an object-level SLAM for indoor scene understanding. It fuses instance segmentation with Truncated Signed Distance Functions (TSDF) presented in Kinect Fusion [14]. Although they use Mask-RCNN based object detections as landmarks, they do not consider the detection accuracy. Their method is applicable to static indoor scenes and requires computationally expensive optimization and re-localization steps when the TSDF is reset.

SLAM++ [19] is another object-level SLAM framework where objects are detected, tracked and mapped in real-time in a dense RGB-D framework which can identify and ignore dynamics in the scene. However, it requires a database of well defined 3D models of the objects to refer to, and it calls for creation of such a database offline.

QuadricSLAM [15] is another object-oriented SLAM system that estimates camera pose and object pose simultaneously from noisy odometry and object detection bounding boxes. It incorporates state-of-the art deep learning object detectors as sensors to build surfaces in 3D space called “quadrics” by combining object detections from multiple views. Quadrics act as 3D landmarks and provide a compact representation that approximates an objects position, orientation and shape. Performance of this algorithm will be poor in a cluttered scene such as ours where the bounding boxes for grapes will significantly overlap on each other.

For instance segmentation of grapes, we considered both

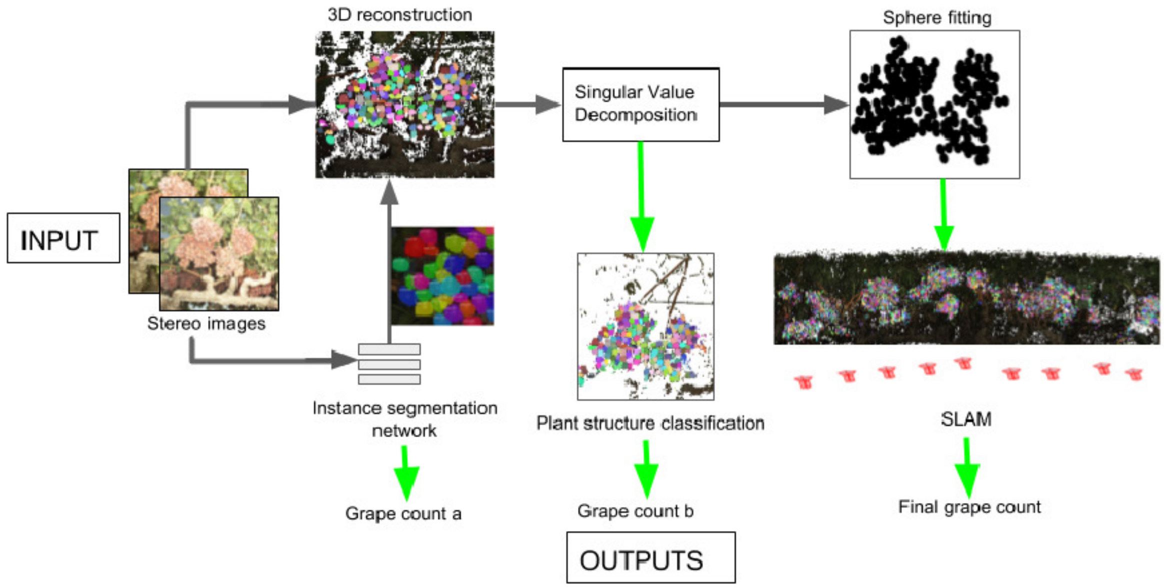


Figure 3. Visualization of our pipeline. Inputs are stereo images shown on the left. Data flows from left to right and is represented by gray arrows. Green arrows indicate the outputs at each stage. Final outputs are Dense 3D map, camera trajectory (in red) and final grape count which are at the right end of the pipeline.

image and 3D classifiers. Fully Convolutional Network [9] and Mask R-CNN [7] identify objects of interest in an image and provide the location of all the pixels belonging to that object. Capsule network [27] is a family of neural networks which explicitly models the spatial relationship between objects in an image. Replacing the feature points with these instance masks to find correspondence not only brings down the dimensionality of state-space but improves the quality of dense reconstruction. In addition, they generate a multi-resolution map, where regions have resolutions based on object size. SplatNET [22] is another network architecture which performs spatially-aware feature learning, as well as joint 2D-3D reasoning. While Mask R-CNN is currently the state-of-the art in 2D instance segmentation, 3D networks such as PointNet [18] perform object classification and part segmentation from unstructured point clouds. PointFusion [25] is another 3D object detection method that intelligently fuses both image based and point cloud based object detectors to predict 3D bounding boxes.

All of these 3D segmentation techniques require extensive training data. No such training data readily available for vineyards, and labeling point clouds is not feasible at such a scale. Moreover, grapes cannot be assigned a predefined 3D shape as most of the grapes are partially visible either due to self occlusion or due to lack of different camera views resulting in 2.5 dimensional shape. Folding Net [26] on the other hand does unsupervised learning with end-to-end deep auto-encoder. Although this seems promising to use in a low training data setup, it will fail to perform in our case where the canopy does not have a predefined 3D shape.

3. Method

We present a method that combines features and ideas from the literature to create an end-to-end vineyard mapping pipeline. Specific contributions of this paper are:

- Combination of image features and local shape information from 3D to improve accuracy of plant structure classification.
- Simplifying SLAM by using shapes fitted in 3D as object-level features.
- End-to-end pipeline for accurately counting grapes.

Figure 3 gives a visualization of our approach where gray arrows represent data flow and green arrows represent the outputs. 3D reconstruction module and instance segmentation module run in parallel by taking stereo images and left image as inputs respectively. Masks generated by the instance segmentation module are projected onto the 3D model constructed using disparity. This information flows to the SVD module which further improves classification and generates labels for plant structures. Spheres are fit to the SVD output and passed to SLAM module. SLAM module stitches consecutive point clouds based on correspondence between fitted shapes to build a dense map and estimate camera trajectory. Grape counts are obtained from

3D model given out by SLAM and are reprojected to image space to compare with image-wise ground truth count. Grape count after every step is also given out to perform ablation experiments detailed in section 4.3 .

3.1. 3D reconstruction

We use point clouds to represent the 3D profile of the scene. As the goal here is to model the plant and classify its structures, a dense point cloud is necessary. Therefore, a disparity-based method is needed as opposed to modern SIFT and SURF based correspondence matchers for 3D reconstruction. Looking at the taxonomy of stereo matching algorithms in [20, 23], we found that Semi-Global Matching best serves our purpose. The algorithms calculates the disparity for each pixel in the left image where matching cost calculation is based on Mutual Information (MI) which is insensitive to recording and illumination changes. We refer readers to [8] for further details.

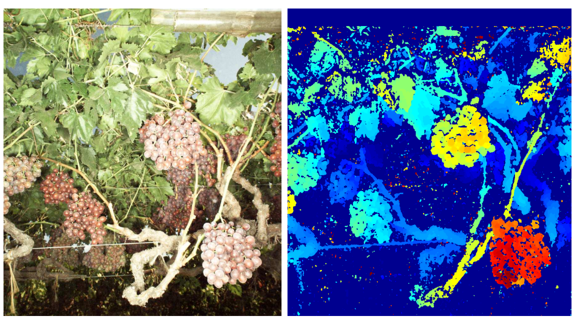


Figure 4. Left : Input image (histogram equalized for visualization) Right: Disparity map colored based on depth from the camera where red indicates points are closer to camera and blue indicates the points are away from camera.

3.2. Instance segmentation

We chose Mask R-CNN [7] for instance segmentation after experimenting with various network architectures. The network configuration was selected empirically by training and analyzing the test results and are listed in Table 1. We created a training and validation dataset to fine-tune the pretrained Mask R-CNN. We observed that the segmentation network performs well when the object of interest occupies at least 5% of the image. Hence, 9MP RGB images were divided into tiles of 256X338 size which contained 0-20 grapes each. Up-down and left-right image augmentations were used on these tiles to increase the size of dataset. Grapes closer to the camera were labeled and tiny grapes captured from background rows were ignored so that the network learns to detect grapes present on the nearer side of the row. Retraining the network head that was pre-trained on MS-COCO dataset [10] using this grape dataset gave the expected results as shown in Figure 5. We refer readers to [1] for implementation details.

Parameters	Values
Backbone layer	ResNet 50
Head layer	Faster R-CNN
Input size	256X338
Anchor ratio	[0.5,1,2]
Learning rate	0.007
Epochs	200
Steps per epoch	100
RPN anchor scale	[16,32,64,128]
Avg grape size in pixels	50X50
Augmentation	LR, UD flip
Training images	62 image segments
Validation set	7 image segments

Table 1. Network parameters for instance segmentation of grapes using Mask R-CNN

Figure 5. *Top: Left:* Input image equalized for visualization *Right:* Instance masks overlayed on input. *Bottom: Left:* Input image showing grapes of varying pixel size equalized for visualization. *Right:* Midpoints of masks marked in red on a histogram equalized image for better visualization

3.3. Singular Value Decomposition

We use SVD to characterize the local neighborhood around each 3D point in terms of its 'pointness', 'curveness' and 'surfaceness'. If the local neighborhood is compact and spatially isotropic then it corresponds to grape berry. A distribution with a single dominant axis of spatial variation corresponds to branch and a distribution with two axes of variation corresponds to leaves. We store the point cloud in a k-d tree, which enables fast lookup of the local neighborhood around each point. We perform the following operations for each point p obtained by picking the center of instance segmentation masks projected to 3D:

- Retrieve points within a distance of d and store it in N, a k x 3 matrix where k is the number of points in the neighborhood.:

$$N = \{p_i : \|p - p_i\| < d\}$$

- Compute the co-variance matrix C_N for every N:

$$C_N = \frac{1}{k}\sum_{i=1}^{k}(p_i - \overline{p})(p_i - \overline{p})^T$$

where $\overline{p}$ denotes the mean of the 3D points in N.

- Perform SVD on all C_N.

- Analyze the top three singular values ($\lambda_3 \leq \lambda_2 \leq \lambda_1$) to identify the number of dominant directions:

$$\begin{aligned} \lambda_3 = \lambda_2 = \lambda_1 &\rightarrow \text{ isotropic spatial distribution} \\ \lambda_3 = \lambda_2 << \lambda_1 &\rightarrow \text{ linear distribution} \\ \lambda_3 << \lambda_2 = \lambda_1 &\rightarrow \text{ planar distribution} \end{aligned}$$

- Generate saliency feature for every point

$$\text{saliency}_p = \begin{bmatrix} \text{grape} \\ \text{branch} \\ \text{leaf} \end{bmatrix} = \begin{bmatrix} \lambda_3 \\ \lambda_1 - \lambda_2 \\ \lambda_2 - \lambda_3 \end{bmatrix}$$

We perform the above steps at three different neighborhood sizes to obtain a 9X1 feature vector. These features are used to train a Transductive Support Vector Machine (TSVM) that can classify point cloud into plant structures.

TSVM is a semi-supervised learning algorithm that makes use of unlabeled data for training. We chose this particular classifier as it is able to achieve a high accuracy with a dataset that has only 20% labelled and 80% unlabelled data. Let w be the weight vector and $x_i \in \mathbb{R}^9$ be the input feature vector and $y_i \in \{-1, 1\}$ be the output. Let r be the fraction of labeled examples that belong to the positive class. TSVM minimizes the objective function of linear Support Vector Machines (SVM) by adding an additional term to drive the classifier away from high-density regions:

$$\min_{w, {y'_j}_{j=1}^{u}} \frac{\lambda}{2}||w||^2 + \frac{1}{2l}\sum_{i=1}^{l} l(y_i w^T x_i) + \frac{\lambda'}{2u}\sum_{j=1}^{u} l(y' w^T x'_j) \tag{1}$$

$$\text{subject to } \frac{1}{u}\sum_{j=1}^{u} max[0, sign(w^T x'_j)] = r$$

where $l(z) = max(0, 1 - z)$ is the hinge loss function, λ is the regularization parameter and λ' is a tunable parameter that controls the influence of unlabeled data. We build on the SVM toolbox implemented in [21] to achieve multi-class TSVM classification. We refer the user to [21] for further details on TSVM and its implementation. TSVM estimates a label $y_i \in \{1, -1\}$ where {1,-1} corresponds to {grape, non-grape} classes respectively for every point. Points belonging to non-grape class are again passed through TSVM to get leaf vs branch classification.

3.4. Sphere fitting

We fit spheres to grapes and use them as object features for SLAM. A sphere is defined by its centre (x_0, y_0, z_0) and radius r. If a point (x_i, y_i, z_i) lies at a distance r from (x_0, y_0, z_0) , then it is on the surface of the sphere. Here, we fit spheres to the points which were classified as grape by estimating r, x_0, y_0, z_0 using M-estimator SAmple Consensus (MSAC) algorithm [24]. MSAC is a variant of RANdom SAmple Consensus (RANSAC) which iteratively estimates the parameters of a sphere by choosing a set of points that minimize the cost function :

$$\sum_{i=1}^{n}[(x_i - x_0)^2 + (y_i - y_0)^2 + (z_i - z_0)^2 - r^2)] \tag{2}$$

subject to $(x_i - x_0)^2 + (y_i - y_0)^2 + (z_i - z_0)^2 - r^2) \leq d$

d, the maximum allowable error per point, and the number of iterations are set empirically. Range of the radius, r, and the regions for search are provided from the ROIs that were obtained by projecting instance segmentation masks to 3D and refined by SVD module. For every predicted grape, MSAC estimates the corresponding parameters and fits a sphere. After fitting every sphere, the points belonging to it are removed from the point cloud so that the MSAC does not incorrectly use same subset of points to fit two different spheres.

Fitting a parameterized model such as a sphere has multiple benefits. Firstly, it provides a compact high level representation that naturally aids map compression in large-scale mapping. Secondly, it provides higher-level descriptions of the object geometry which provides a closed form solution to data association problem.

3.5. Simultaneous Localization and Mapping

Gps data is accurate only upto sub meters and is often not reliable in an agricultural field due to occlusions. Hence, registering images and grapes to each other calls for a localization method with an accuracy in sub-centimeter range. Our SLAM module finds the 6 DOF camera position and incrementally stitches point cloud frames into a 3D map with sub-centimeter accuracy. A world frame with origin at the starting point of the camera, and a current frame with the origin at the current camera location are maintained. We use a 4X3 transformation matrix to represent the camera pose and also to transform the current frame to world frame.

The state-of-the-art methods to estimate the transformation between two frames either involve probabilistic estimation or error minimization techniques such as Iterative Closest Point (ICP). ICP alignment has the problem of low efficiency in large data sets. Using fitted shapes instead of raw points reduces the amount of data and overcomes this problem.

Input to the SLAM module is the point cloud of fitted spheres. Given two point clouds fixed, **F**= $\{f_1, f_2, ..., f_n\}, f_i \in R^3$ and moving, **M**= $\{m_1, m_2...., m_n\} m_i \in R^3$, ICP finds a rotation matrix **R** and a translation vector **t** such that error between transformed moving cloud and fixed cloud is minimum. We take advantage of the fact that if the correct correspondences between **F** and **M** are known, **R** and **t** can be calculated in closed form. We experiment with two variants of ICP point-to-point and point-to-plane. In both the cases, rotation matrix is represented in quaternion form and is converted to linear approximation problem. Although point-to-plane ICP takes more time per iteration due to added cost of normal computation, it converges in fewer iterations. Hence, we chose point-to-plane ICP for our pipeline.

3.5.1 Point-to-plane ICP

In this variant of ICP, **R** and **t** are iteratively estimated to minimize the distance between every point m_i and the tangent plane at its corresponding point f_i. Tangent plane is represented by its unit normal n_i computed around a small 3D neighborhood with 50 points around the point f_i.

$$\min_{R,t} \sum_{i=1}^{k} (|(Rm_i + t - f_i)^T n|^2) \tag{3}$$

This provides a more accurate estimation in cases where one point cloud is denser than the other and exact correspondences are sparse.

The transformation parameters given out by ICP is converted to global frame to get camera trajectory and dense 3D model. Grape counts are obtained from this 3D model and are reprojected to image space using the estimated camera poses in world frame to get final grape count per image.

4. Experiments

4.1. Dataset

Data was collected from a vineyard in Delano, California in July 2018. An air blower was attached below the camera to blow leaves occluding the grapes to increase visibility. This blowing causes movement in the leaves and introduces dynamics to the scene. Images were taken using two 9MP PointGrey cameras with a 0.22m baseline which was mounted on a field vehicle. Consecutive images in a given sequence overlap by at least 50% in the dataset at any time and overlap more than 85% when the leaf blower is on. Images were collected from more than 20 rows with each row being 187m long, comprised of 70-80 vines captured in about 750 images. Experiments were conducted with and without the blower whose results are separately reported.

4.2. Main results

Grape counts were re-projected onto images and compared with the grapes manually counted from images. We obtained an average precision of 0.9662, recall of 0.9881 and F1 score of 0.977. For a randomly selected group of 20 consecutive images, an R^2 correlation of 0.9989 was achieved. Figure 6 shows the plot of final grape count estimated from 20 consecutive images re-projected to images against the ground truth grape count from those images.

With the assumption that the plants were planted in a straight line, we interpolated the gps location of starting and ending points of the row to test the localization accuracy. Dense 3D SLAM output obtained from 20 consecutive images is shown in Figure 9 (a) along with the straight line in green and camera trajectory in red. Mean square error was calculated using the deviation of stitched model from the straight line at every frame and was found to be 20cm for a run of 187m.

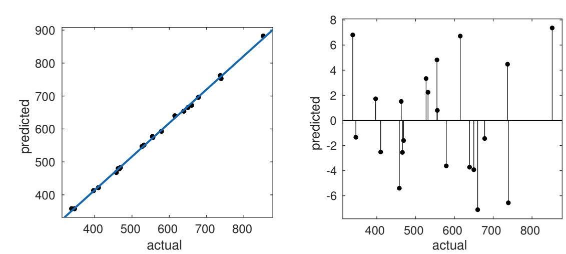


Figure 6. *Left:* Plot of grape count estimated by our pipeline against ground truth grapes for 20 consecutive images *Right:* Residual error plot for the same.

4.3. Ablation experiments

We ran ablation experiments to demonstrate the importance of each module in the pipeline.

4.3.1 Mask R-CNN

If we remove the instance segmentation part from our pipeline, the SVD module will have to choose random ROIs and classify them into grape, branch and leaf. We experimented with different strategies to pick ROI: random selections, voxel grid, uniform distribution and color based. All of these strategies result in only a fraction of grapes being selected as ROI. This leads to noisy transformation estimations which keep compounding and results in a poor map.

4.3.2 SVD

Roles of the SVD module are to reduce the false positive grape detections and to classify non-grape points into leaf and branch. We examined the spheres fitted onto ROIs projected from instance segmentation directly by skipping the SVD module. Since ROIs from Mask R-CNN often overlap or cover lesser number of pixels than expected, this either results in multiple spheres being fitted to one grape or one sphere fitted to multiple grapes. This behaviour is shown in Figure 7 where multiple spheres intersect on the grape.

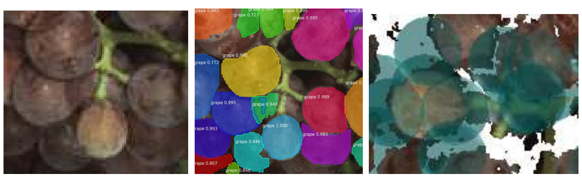


Figure 7. *Left:* Raw image. *Center:* Predicted masks. *Right:* Snapshot showing sphere overlapping as a result of overlapping and inaccurate instance mask predictions. Fitted spheres in blue are overlapped on original point cloud with grapes.

Removing the SVD module reduces the pipeline to an end-to-end network which estimates grape centres and transformations from stereo image inputs. Although this sounds promising because of the proven performance of 3D segmentation networks in the literature, we expect it to require a large amount of labelled 3D data to reach our pipeline's level of accuracy and is left for future work.

4.3.3 Sphere fitting

Sphere fitting acts as a landmark extractor for SLAM. It reduces the complexity of SLAM module to iterative fitting of point clouds. In ideal conditions, when the segmentation module produces non-overlapping masks with perfect boundaries and completely static scene, this module can be replaced by a tracking estimator such as optical flow. We observed that decreasing the number of iterations of MSAC reduces the accuracy of fitted spheres and leads to poor registration of point clouds in SLAM module. When we removed this module and used ROI along with a mean radius guess to obtain a sphere, we saw that the false positives remained and occluded grapes were missed as shown in Figure 8.

5. Timing

We summarize module wise time requirements in Table 3. All the modules were tested on a 6 core i7 processor. Instance segmentation network and 3D point classifier were trained on a GeForce GTX 1080 Ti GPU. Implementations are done in C++ using OpenCV and PCL libraries.

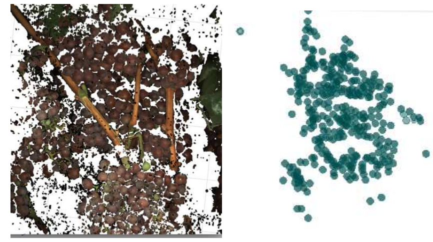


Figure 8. 3D snapshots showing *Left*: Raw point cloud with grapes *Right*: Spheres drawn with guessed radius and ROI centres instead of iterative MSAC of sphere fitting module

Module	Time
Nearest neighbour	0.02 ms per ROI
SVD	0.15 ms per ROI
TSVM classification	1 ms per frame
Sphere fitting	0.1 ms per sphere
SLAM	1 ms per frame
Mask R-CNN training	10 hours
TSVM training	2 minutes

Table 2. Module wise time requirements for testing and training

6. Comparisons

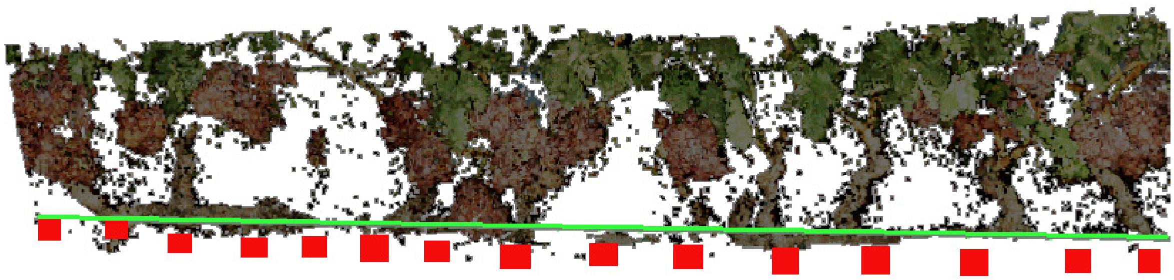
We compare our fitted shape based SLAM method against ICP on raw point cloud and feature-point based SfM and show them in Figure 9. In the sub-figures a and b, green line represents the expected straight line in which the plants must lie. It is obtained by interpolating start and end gps points of the row. Red box represents the camera positions.

Map built by applying Point-to-plane ICP on raw point cloud is shown in Figure 9 (b) where grape vines are seen drifting from the expected line shown in green.

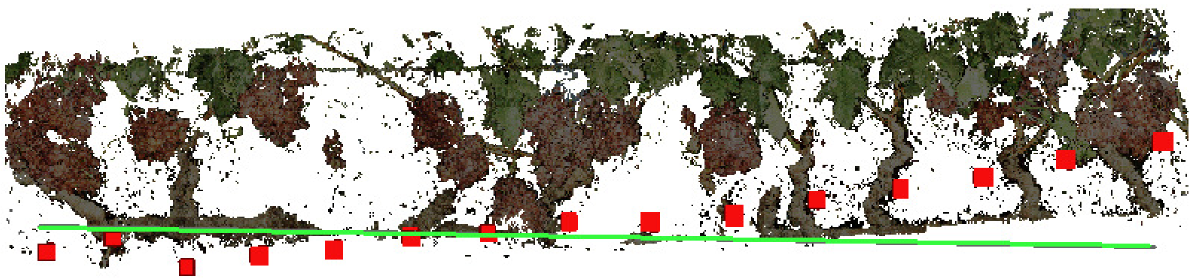
Feature based SfM was ran using SIFT, SURF and corner features and the best result is shown in Figure 9 (c). We observed that in every frame, strongest of these features always lied on leaf corners shown in Figure 9 (c) *Left* and gave wrong feature correspondences as shown in Figure 9 (c) *Right*. We ran ORB-SLAM2 on our dataset and found the map to be too sparse to extract any information regarding grapes.

7. Discussion

We presented a camera based end-to-end pipeline to build a dense 3D model of a vineyard and count grapes. Our pipeline makes use of 3D characteristics of a point in addition to semantic information from images to improve the accuracy. The process of estimating transformation is



(a) SLAM output from our pipeline



(b) SLAM model obtained using ICP on raw point cloud

(c) *Left:* Detected SURF features marked in red. *Right:* Feature matching between images where yellow dotted line represents matches between red feature points in left image and green points in right image.

Figure 9. Comparisons of SLAM model built from 15 consecutive images using different techniques. Green line indicates the ground truth position of the base of point cloud which was obtained by connecting the gps coordinates of starting and ending point of the plant row. Red points are camera position outputs. All point clouds are down-sampled and filtered for visualization purpose.

made robust and efficient by using object-level representation instead of feature points. As every part of the pipeline except instance segmentation is independent of the sensor used to acquire data and the type of fruit, this method can be extended to other fruit types such as apples, oranges, and other sensor types by just retraining the instance segmentation module which is left to future work. Another direction of future work is to generate maps based on gps location of start and end point of rows and overlay it on latest satellite images to get an overview of grape distribution in the field.

8. Acknowledgements

Authors would like to thank Stephen Nuske for technical assistance, Abhisesh Silwal, Zania Pothen, Omeed Mirbod, Tanvir Parhar, Harjatin Baweja and Tevon Walker for assistance in data collection.

References

[1] W. Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. https://github.com/matterport/Mask_RCNN, 2017. 4

[2] S. Y.-Z. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2703–2710, 2012. 2

[3] H. S. Baweja, T. Parhar, and S. Nuske. Early-season vineyard shoot and leaf estimation using computer vision. In *2017 ASABE Annual International Meeting*, St. Joseph, Michigan www.asabe.org, November 2017. American Society of Agricultural and Biological Engineers. 2

[4] D. DeTone, T. Malisiewicz, and A. Rabinovich. Toward Geometric Deep SLAM. *CoRR*, abs/1707.07410, 2017. 2

[5] D. Dey, L. Mummert, and R. Sukthankar. Classification of plant structures from uncalibrated image sequences. *IEEE Workshop on the Applications of Computer Vision (WACV)*, pages 329–336, Jan 2012. 2

[6] J. Engel, J. Stckler, and D. Cremers. Large-scale direct slam with stereo cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1935–1942, Sep. 2015. 2

[7] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. 2, 3, 4

[8] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 807–814 vol. 2, June 2005. 4

[9] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. *CoRR*, abs/1611.07709, 2016. 3

[10] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 4

[11] X. Liu, S. W. Chen, S. Aditya, N. Sivakumar, S. Dcunha, C. Qu, C. J. Taylor, J. Das, and V. Kumar. Robust fruit counting: Combining deep learning, tracking, and structure from motion. *CoRR*, abs/1804.00307, 2018. 2

[12] J. McCormac, R. Clark, M. Bloesch, A. J. Davison, and S. Leutenegger. Fusion++: Volumetric Object-Level SLAM. *CoRR*, abs/1808.08378, 2018. 2

[13] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robotics*, 33(5):1255–1262, 2017. 2

[14] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, Oct 2011. 2

[15] L. Nicholson, M. Milford, and N. Sünderhauf. QuadricSLAM: Constrained Dual Quadrics from Object Detections as Landmarks in Semantic SLAM. *CoRR*, abs/1804.04011, 2018. 2

[16] S. T. Nuske, S. Achar, T. Bates, S. G. Narasimhan, and S. Singh. Yield estimation in vineyards by visual grape detection. In *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '11)*, September 2011. 2

[17] Z. Pothen and S. T. Nuske. Texture-based fruit detection via images using the smooth patterns on the fruit. In *IEEE conference on Robotics and Automation*, May 2016. 2

[18] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *CoRR*, abs/1612.00593, 2016. 3

[19] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. In *CVPR*, pages 1352–1359. IEEE Computer Society, 2013. 2

[20] D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, pages 131–140, Dec 2001. 4

[21] V. Sindhwani and S. S. Keerthi. Large scale semi-supervised linear svms. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 477–484, New York, NY, USA, 2006. ACM. 5

[22] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M. Yang, and J. Kautz. SPLATNet: Sparse Lattice Networks for Point Cloud Processing. *CoRR*, abs/1802.08275, 2018. 3

[23] F. Tombari, F. Gori, and L. D. Stefano. Evaluation of stereo algorithms for 3d object recognition. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 990–997, Nov 2011. 4

[24] P. H. S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:138–156, 2000. 5

[25] D. Xu, D. Anguelov, and A. Jain. PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation. *CoRR*, abs/1711.10871, 2017. 3

[26] Y. Yang, C. Feng, Y. Shen, and D. Tian. FoldingNet: Interpretable Unsupervised Learning on 3D Point Clouds. *CoRR*, abs/1712.07262, 2017. 3

[27] D. Zou and P. Tan. Coslam: Collaborative visual slam in dynamic environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):354–366, Feb 2013. 3