Automatic Grape Bunch Detection in Vineyards for Precise Yield Estimation

Scarlett Liu Mark Whitty
Stephen Cossell
School of Mechanical and Manufacturing Engineering,
University of New South Wales, Sydney 2052, Australia
sisi.liu@unsw.edu.au, m.whitty@unsw.edu.au, macgyver@unsw.edu.au

Abstract

Precise yield estimation using image processing techniques has been demonstrated conceptually on a small scale. Expanding these solutions to larger scale applications requires significant computational power, which need to analyze the entirety of all captured image data. However, many images captured for yield estimation in these processes only contain small areas of useful features for analysis. This paper introduces an image processing algorithm combining color and texture information, and the use of a support vector machine, to accelerate fruit detection by isolating useful features in images. Experiments carried out on two varieties of red grapes (Shiraz and Cabernet Sauvignon) demonstrate an accuracy of 87% and recall of 90%. This method is also shown to remove the restriction on the field of view and background, which limited existing methods and is a first step towards precise and reliable yield estimation on a large scale.

Keywords: Bunch Detection, Image Processing, Low Cost Computation, Precision Viticulture, Yield Estimation, Field Application

1 Introduction

Precise yield forecasting is predicted to help the Australian Wine Industry save $100 million per year as this forms the basis of the wine making process through to wine marketing and sales. As a result, accurate yield estimation and forecasting is in high demand in the viticulture industry both for practical and academic reasons. Nowadays, yield estimation is still performed manually by human hands worldwide, and its precision depends on the scalability of hand sampling. Large scale sampling is not a guarantee of successful yield forecasting as some vineyards are not uniform [3] and many variables can affect approaches such as temperature and water availability. At the same time, thand sampling is tedious, expensive, inaccurate and has human bias (humans tend to choose healthier and bigger bunches when randomly sampling in the field [15]). So an automatic yield estimation system based on image processing has become a mainstream research topic in recent years. For tackling this problem, Nuske *et al.* [12] developed a prediction method by detecting the berry number from images taken in a vineyard. This paper applied a Radial Symmetry Transform (RST) [11] to find keypoints, which are potential berry locations, as the initial step for later feature analysis. This novel work was the first to process a large collection of ground truth vine images to generate a yield estimate. However, before the keypoints are extracted, there is no image size reduction. The problem is that the computational cost of high level image processing methods becomes slower with increased image size.

In reality, for accurate yield estimation, images are usually captured by a vehicle mounted camera. The entire height of a vine needs to be captured and hence a wide field of view is chosen to ensure nothing is missed. As a result, other interrelated objects such as rocks, posts, grass, and other grapevines behind the current row are visible in images. Since the berry size is small relative to the field of view, a berry detail is vital to late yield calibration [10], high resolution images need to be captured. High resolution images are slow to process when certain high level image processing methods, such as RST and Zernike Moments Extraction [8], are applied. For the required field of view, images captured in vineyards contain around 70% meaningless information for yield estimation. This means 70% of the processing time is redundant. Therefore, this paper aims to reduce the relevant image size by extract relevant windows before extracting details of berries in the images for accelerating the image processing tasks. Extracting relevant windows for further processing also allows an estimate of the number of bunches to be made.

In 2011, Correa *et al.* performed a comparison [1] of different Fuzzy C-Means (FCM) clustering algorithms to extract features from vineyard images. The 20 segmented images achieved an accuracy of 85%, 87% and 88% by Robust Fuzzy Possibilistic C-Means, FCM and FCM-GK (FCM with Gustafson-Kessel) methods respectively. In the next year, another two extended papers [1, 6] by the same research team combined Support Vector Machines (SVM), K-Means and the Scale-Invariant Feature Transform (SIFT) to cluster different objects in vineyard images. In the same year, a classification of grapevine structures from uncalibrated image sequences was presented by Dey *et al.* [4]. In this paper, Structure-From-Motion (SFM) was implemented to build a 3D reconstruction of grapevines as a first step, then a saliency feature was applied for traversability analysis of point cloud data. An SVM and a conditional random field (CRF) were adopted for spatially smoothing the 3D model in the final step, improving the accuracy of classification. But Dey's algorithm requires 3D point cloud data, and classifying each pixel into bunch or non-bunch instead of counting bunch number. Mahalanobis measures were applied in papers [5, 16] for grape, branch, leaf and background classification. The accuracy of classification in the aforementioned papers were high, but the computational cost was expensive, especially for high resolution images, and the accuracy was based on pixel level.

In addition, except for the work presented by Dey *et al.* [4], all images tested in these papers had a white background with perfect view proportion and the scale of tested images was small. For bunch detection under field conditions the procedure of classification becomes time consuming and complicated. At least 50% of the computation time is wasted on clustering useless information so it is not practical for yield estimation, considering the size of the vineyard.

2 Image Processing Algorithm

In this paper, the proposed bunch segmentation algorithm contains 3 main steps: image pre-processing, training on a subset of images and segmentation on the test set. For both groups (training and testing), morphological operations are applied in HSV color space for extracting the initial bunch hypotheses, and then a shape filter is utilized to exclude incorrect bunches. Next, a group of true bunch areas are picked by hand as the training set for extracting the features. An SVM [2] is applied to segment bunches in the test set (the remaining images). **Figure 1** demonstrates the process flow for bunch segmentation and data analysis. The images taken in Block 11 were used to develop the algorithm.

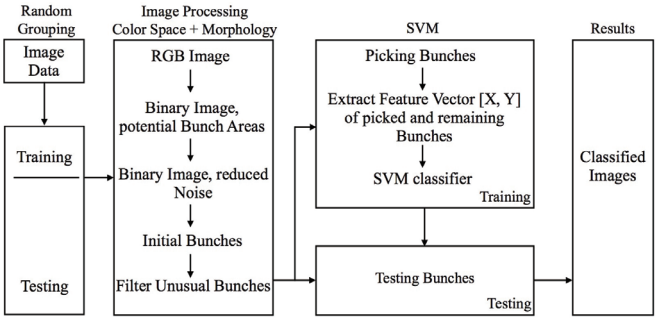

Figure 1: Bunch detection flow chart proposed in this paper.

2.1 Image Pre-processing

Image pre-processing consists of first thresolding an image on the H and V channels, which removes around 80% of irrelevant pixels. Otsu's method [13] is applied to the H and V channels to detect the thresolding value, which is dependent on illumiation conditions. Using the HSV color space allows easy detection of the difference between vine and sky regions, as seen with the distinguishable local minima and the dashed line in **Figure 2d**. The sky area can also be consistently detected from the V channel, as seen in **Figure 2f**. In contrast, the B channel is not a good indicator of irrelevant regions as there are more local maxima and there is large inconsistency between images, as seen in **Figure 2c**. Potential bunch areas are obtained by calculating the intersection between the H and V binary images. Next several morphological operations are performed on the resulting image to reduce noise. A new filter introduced in this paper is then applied to remove unusual detected bunch shapes and is based on the aspect ratio of the detected bounding box of a bunch area, as seen in **Figure 3**. Acceptable aspect ratios are usually between 0.25 and 2.

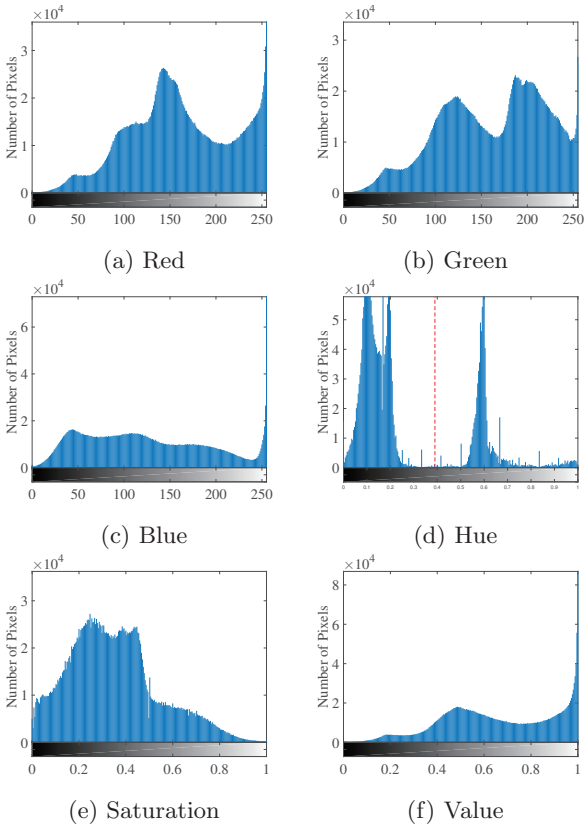

Figure 2: Six histograms in RGB and HSV color space of a sample image.

Figure 3: Filtering: blue arrows point out where filtered out bunches were.

2.2 Feature Selection

A total of 80 images take from Block 11 were pre-processed with some false positive areas identified in some images, as seen in **Figure 3**. Each identified region was then manually identified as being correct or incorrect. For each bunch detected via pre-processing, a 58 dimensional feature vector is extracted and contains features such as closeness, solidity, extent and compactness, the texture information in each channel in RGB, HSV and L*a*b color spaces.

In order to simplify the classifier model, reduce memory usage and improve the computational speed, the ReliefF algorithm [14] and sequential feature selection [9] were applied for decreasing the feature dimensions. The ReliefF algorithm uses k-nearest neighbors to calculate the attribute importance and attribute weights for each feature. It demonstrates that first 20 predictors have positive weights so 20 significant features were selected. As for sequential feature selection, the misclassification rate was used as the criterion for min-

imising the feature subsets. To be more specific, sequential forward selection (SFS) is utilized by sequentially adding features to a feature candidate set. The criterion is continually calculated until the criterion stops increasing and 21 features are selected by SFS. The final feature candidates (17 dimensions) are decided by the intersection between two groups of features and are described as follows:

- **Closeness**: Scalar specifying the ratio between the y-position (y coordinates in an image) of each initial bunch and average y-position for all bunches in one image.
- **Solidity** [7]: Scalar specifying the proportion of pixels in the convex hull that are also in the region.
- **Extent** [7]: Scalar specifying the ratio of pixels in the region to the total bounding box.
- **Compactness** [7]: Scalar specifying the ratio of perimeter to area of the region.
- **Texture** [7]: Average gray value of HSV, average contrast of HSV, measure of smoothness of HSV, third moment of HS, measure of uniformity of H, entropy of HV

A *closeness* value is also a newly defined feature in this paper, and figures in this paper show that most bunches are located in a certain position in the vertical direction. For each image, Otsu's method [13] is applied again to divide the y-positions of all initially detected bunches into two groups. The reference y value is the average y value that is computed from the larger group. The closeness is the ratio between the y value of the potential bunch and the base y value.

2.3 Training Size Selection

After decreasing the dimension of the feature vector, a SVM was implemented to analyse the size of the training set as part of the learning stage. The 80 images pre-processed using the method outlined in **Section 2.1** were use to train the SVM using training group sizes in intervals of 4 from 4 to 76 images. For every potential bunch in the training set, a feature vector of 17 features selected in **Section 2.2** and its response of being a valid bunch or not (true or false) are used as a classifier. The confusion matrix for each training set is calculated for classification performance analysis based on classifications of *true positive* (TP), *true negative* (TN), *false positive* (FP) and *false negative* (FN).
In order to calculate the correct number of images for the training step, *accuracy* (ACC), *recall* (True Positive Rate: TPR) and *precision* (Positive Predictive Value: PPV) are used as justification by applying the SVM classifier obtained from the training step:

$$ACC = \frac{TP + FP}{TP + TF + FP + FN} \quad (1)$$

$$TPR = \frac{TP}{TP + FN} \quad (2)$$

$$PPV = \frac{TP}{TP + FP} \quad (3)$$

All results for each training set are illustrated in **Figure 4** and **Figure 5**. It is clear that 16 images is sufficient for achieving 90% accuracy for bunch classification.

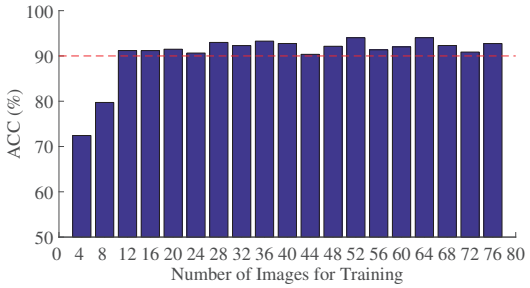

Figure 4: Accuracy for training size selection.

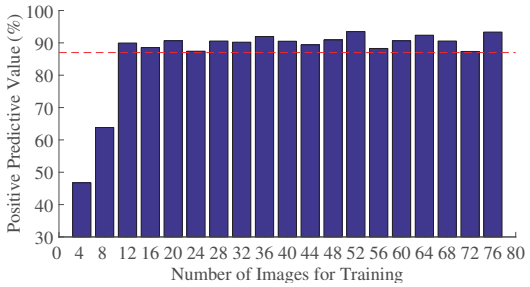

Figure 5: Precision for training size selection.

3 Experimental Materials and Data Scope

Field experiments were conducted in Camatta Hills, California, with a total of 160 images taken under natural illumiation conditions, the details of which are given in **Table 1**. All images were photographed using an OLYMPUS Camera (Model: SP600UZ) with focal length 5 mm, internal flash, automatic mode and a resolution of 3968×2976 pixels. Four images were taken for each sampled vine, including two arms on each side (East and West), as shown in **Figure 6**.

Table 1: Image data information, properties of vineyards where experiments conducted.

Block	Images	Sampled Vine	Photo Date	Cultivar
11	80	20	28 Sep. 2013	Shiraz
19	80	20	09 Oct. 2013	Cabernet Sauvignon

4 Experimental Results

The selected image features and a training set of 16 images were applied to another group of images captured in Block 19 at a different time of day, following the same procedure shown in **Figure 1**. The detected bunch regions are shown in **Figure 7**.
To avoid overfitting, for each image of Block 19, bunches were manually labelled as true or false bunches. Then K-fold cross-validation was applied for

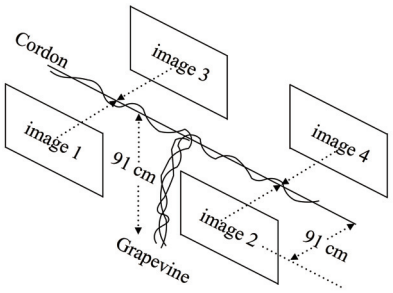

Figure 6: Photography arrangement.

Figure 7: Classification: green boxes are classified as true bunches while pink boxes are classified as false bunches.

validating the detection algorithm. The fold number was set to 5, as there were 16 images for training. All raw data was tested by cross-validation for achieving the real truth in the vineyard for yield estimation. As such, some images in these 80 images do not contain any bunches or just contain several small bunches. This meant that the training size was smaller than 16 images in some cases. However, it still obtains the average accuracy and recall of 87% and 90% respectively by using 5 fold validation.

5 Conclusions and Outlook

This paper presented an method for efficiently detecting bunch areas in photographs. Simple features were extracted to reduce computational load and verified against manual labelling to properly evaluate classification results. For 16 images used for training, the algorithm was able to classify bunch areas with 87% accuracy and 90% recall. As the algorithm was trained on photographs of Shiraz and tested on Cabernet Sauvignon, both of which take at different times of the day, this proves that the algorithm works across different cultivar and different lighting conditions, a key limitation of existing methods. The algorithm, however is curretly limited to varieties of purple grapes and an area may contain more than one bunch. Improving on these limitations is planned as future work, but the proposed algorithm already allows bunch area detection more accurately and on a larger scale, due to lower computation cost, than existing methods in later stages of the season.

6 Acknowledgments

The authors would like to thank Will Drayton, Franci Dewyer, Joseph Geller and Angus Davidson from Treasury Wine Estates for collecting the raw images used in this paper. First author is partly supported by the Chinese Scholarship Council.

References

[1] Christian Correa, Constantino Valero, Pilar Barreiro, J Tardaguila, and MP Diago. A Comparison of Fuzzy Clustering Algorithms Applied to Feature Extraction on Vineyard. *Avances en inteligencia artificial, J. Lozano, J. Gomez, and J. Moreno, Eds*, 1(1), 2011.

[2] Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[3] I Dami and P Sabbatini. Crop Estimation of Grapes. Technical Report HYG-1434-11, The Ohio State University, 2011.

[4] Debadeepta Dey, Lily Mummert, and Rahul Sukthankar. Classification of plant structures from uncalibrated image sequences. In *2012 IEEE Workshop on Applications of Computer Vision (WACV)*, pages 329–336. IEEE, 2012.

[5] MP Diago, C Correa, B Millán, and P Barreiro. Grapevine yield and leaf area estimation using supervised classification methodology on RGB images taken under field conditions. *Sensors*, 12(12):16988–17006., 2012.

[6] C Correa Farias, C Valero Ubierna, and P Barreiro Elorza. Characterization of vineyard's canopy through fuzzy clustering and SVM over color images. In *International Conference of Agricultural Engineering*, Valencia, Spain, 2012.

[7] Rafael C Gonzales and Richard E Woods. Digital image processing, 2-nd edition, 2002.

[8] Alireza Khotanzad and Y.H. Yaw Hua Hong. Invariant image recognition by Zernike moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):489–497, May 1990.

[9] Josef Kittler. Feature set search algorithms. *Pattern recognition and signal processing*, pages 41–60, 1978.

[10] Scarlett Liu, Samuel Marden, and Mark Whitty. Towards Automated Yield Estimation in Viticulture. In *Proceedings of Australasion Conference on Robotics and Automation*, pages 2–4, Sydney, Australia, 2013. araa.asn.au.

[11] Gareth Loy and Alexander Zelinsky. Fast radial symmetry for detecting points of interest. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(8):959–973, August 2003.

[12] Stephen Nuske, Kamal Gupta, Srinivasa Narasimhan, and Sanjiv Singh. Modeling and Calibrating Visual Yield Estimates in Vineyards. *Field and Service Robotics*, pages 343–356, 2014.

[13] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27, 1975.

[14] Marko Robnik-Šikonja and Igor Kononenko. Theoretical and empirical analysis of ReliefF and RReliefF. *Machine learning*, 53(1-2):23–69, 2003.

[15] Rebecca Dunstone Stephen Martin and Gregory Dunn. How to forecast wine grape deliveries. Technique report, Department of Primary Industries, October 2003.

[16] J Tardaguila, MP Diago, and B Millan. Applications of Computer Vision Techniques in Viticulture to Assess Canopy Features, Cluster Morphology and Berry Size. In *International Workshop on Vineyard Mechanization and Grape and Wine Quality*, volume 978, 2012.