Journal of Applied Logic 13 (2015) 643–653

Contents lists available at ScienceDirect

Journal of Applied Logic

www.elsevier.com/locate/jal

Automatic grape bunch detection in vineyards with an SVM classifier

Scarlett Liu [*], Mark Whitty

School of Mechanical and Manufacturing, University of New South Wales, Sydney 2052, Australia

A R T I C L E   I N F O

Article history:
Received 11 December 2014
Received in revised form 11 May 2015
Accepted 29 May 2015
Available online 9 June 2015

Keywords:
Bunch detection
Image processing
Low cost computation
Precision viticulture
Yield estimation
Field application

A B S T R A C T

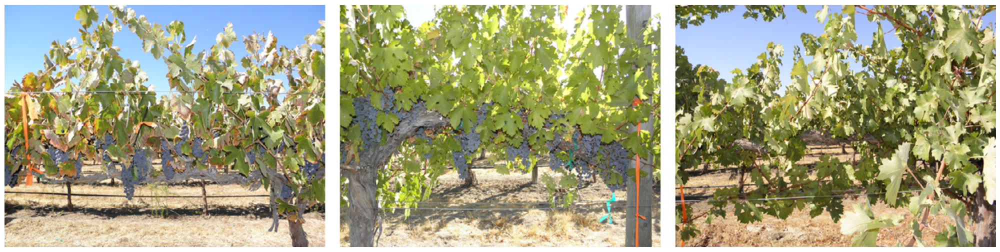
Precise yield estimation in vineyards using image processing techniques has only been demonstrated conceptually on a small scale. Expanding this scale requires significant computational power where, by necessity, only small parts of the images of vines contain useful features. This paper introduces an image processing algorithm combining colour and texture information and the use of a support vector machine, to accelerate fruit detection by isolating and counting bunches in images. Experiments carried out on two varieties of red grapes (Shiraz and Cabernet Sauvignon) demonstrate an accuracy of 88.0% and recall of 91.6%. This method is also shown to remove the restriction on the field of view and background which plagued existing methods and is a first step towards precise and reliable yield estimation on a large scale.

1. Introduction

Precise yield prediction and forecasting in vineyards are predicted to help the Wine Industry save 100 million dollars per year since it provides the base for wine marketing and management. The supply chain from grape growing to wine marketing relies heavily on yield estimation at times well before harvest. Poor yield estimation or forecasting could damage relations between grape growers and wineries, considering both sides need to endure the cost of inaccurate harvest planning and intake purchasing. As a result, accurate yield estimation and forecasting are in high demand in the viticulture industry both for practical and academic reasons. Nowadays, yield estimation is still undertaken by human hands worldwide, and its precision depends on the scale of hand sampling. Large scale sampling is not a guarantee of successful yield forecasting since usually some vineyards are not uniform [4] and many variables may affect the accuracy as harvest approaches, such as temperature and water availability. At the same time, this hand sampling work is tedious, expensive, inaccurate and has human bias (humans tend to choose healthier and bigger bunches when they are doing sampling in the field) [13]. So an automatic yield estimation system based on

[*] Corresponding author.
E-mail addresses: sisi.liu@unsw.edu.au (S. Liu), m.whitty@unsw.edu.au (M. Whitty).

http://dx.doi.org/10.1016/j.jal.2015.06.001
1570-8683/© 2015 Elsevier B.V. All rights reserved.



(a) An image from Block 11 (b) An image from Block 19 (c) An image without bunches from Block 19

Fig. 1. Three original images with different uninformative regions of rocks, grass, posts and trunks, captured by consumer-grade camera in vineyard during daytime. The three images were taken at different times from different blocks at the same distance from the cordon.

image processing in vineyards has become a mainstream research topic in recent years. For tackling this problem, Nuske et al. [14] developed a prediction method by detecting the berry number from images taken in a vineyard. This paper applied a Radial Symmetry Transform (RST) [12] to find keypoints, which are potential berry locations, from every single image as the initial step for later feature analysis. This novel work was the first one to process a large collection of ground truth vine images to generate a yield estimate for viticulture. However, before the keypoints are extracted, there is no image size reduction. The problem is that the computation of high level image processing methods becomes slower with increased image size. Take RST for instance, the theoretical order of computation is O(KN), where K is the total pixel number of the image and N is the size of the neighbourhood [12]. That is, the computational cost doubles if 50% of an image was not useful for yield estimation.

In reality, for accurate yield estimation in vineyards, images are usually captured by a camera which is mounted on a viticultural vehicle. With the aim of not missing bunches, the field of view is set large enough to photograph the grape bunches in every single shot. Other interrelated objects, such as rocks, posts, grass, and other grapevines behind the current row are visible in the images. Every single image taken along a row in each location in a vineyard needs to cover all bunches in vertical direction, which means that majority of the vine canopy needs to be captured in each image, as shown in Fig. 1. Since the berry size is small compared with the canopy, and extracting the details of berries, such as berry size, is vital to late yield calibration [11], large field of view images need to be processed. High resolution images are slow to process when some high level image processing methods, such as RST and Zernike Moments Extraction [9], are applied. For the required field of view, images captured in vineyards contain around 70% meaningless information for yield estimation, as Fig. 1 illustrates. This means 70% of the processing time is redundant. Therefore, this paper aims to reduce the image size or 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.

Feature extraction and grapevine structure classification has been researched in recent years. In 2011, Correa et al. performed a comparison [2] 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 [2,7] 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. [5]. 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. Mahalanobis measures were applied in papers [6,18] 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. [5], 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, as shown in Fig. 1, the procedure of classification becomes time consuming and complicated. At least 50% computation time is wasted on clustering useless information so it is not practical for yield estimation, considering the size of the vineyard.

For the purpose of efficient yield estimation, the visible fruit, namely grapes, is the only segmented object. In prior papers on bunch detection, Chamelat [1] and Reis [16] calculated the detection accuracy based on the number of isolated bunches. The later work combined a segmentation of RGB colour thresholds with further morphological dilation to detect the grape bunches at night. This approach is fast, since it only computed the low level features and applied low level image operations. But this work is limited to images that only contain a major bunch and the bunch must be distinguishable from the uniform background. Additionally, the central value (RGB value) defined in this paper is not easy to identify for practical application and it needs to be identified manually (clicking grapes in images and obtaining the RGB value). Their four central values were determined by trial and error in this paper and the authors claimed that four was sufficient for achieving good results. Even though this method is suitable for small scale data, four central values are not sufficient when it is applied in large data set for yield estimation. Note that the light condition in this paper was controlled (images were captured with internal flash during night time), the central value still changes from image to image (even in one image) as maturity of grapes is not consistent. So this method is not robust for bunch detection in real field conditions. Chamelat [1] explored a new method that combined the Zernike moments value and colour information as features and implemented an SVM for binary classification. Apart from the high level feature extraction being computationally costly, there are two disadvantages with this paper: firstly the image needs to be divided into processing blocks at the first step, and the size of each processing block affects the accuracy of classification rate and accuracy of detected bunch shape; the second is that feature extraction needs to be applied to every processing block, which is computationally expensive.

Decomposing a large image into potential bunch areas to save on the computational cost of feature extraction and later berry detail analysis [11] for yield estimation by low cost image processing methods is the core of this paper. In the proposed vision algorithm, colour thresholding, morphological operation and bunch shape information are implemented to get the initial bunch areas. The ReliefF [17] sequential feature selection method [10] is adopted for feature selection. Then an SVM [3] is applied for training data and classifying the bunches. All grapevine images without controlled conditions are divided into several potential bunch areas by low cost image processing steps before analysing berries in detail. This increases the processing speed in two aspects for yield estimation: no feature extraction on irrelevant areas at this step, so at least 70% are filtered for the dataset; and instead of analysing the whole image for tracking the berry location or berry size by high level image processing methods, only potential bunch areas are needed for later image analysis, which eliminates unrelated areas and saves on computation time in the later stages.

The rest of this paper is organised as follows: Section 2 presents the image processing algorithm in detail, including image pre-processing, feature selection and determining training size of data. In Section 3, the experimental materials and data scope is displayed. Section 4 is devoted to experimental results and Section 5 discusses the performance of the proposed algorithm by comparing it to existing techniques. Finally Section 6 presents the conclusions.

2. Definition of the 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 colour space for extracting the initial bunch hypotheses, and then a shape filter is utilised to exclude incorrect bunches. Next, a group of true bunch areas are picked by

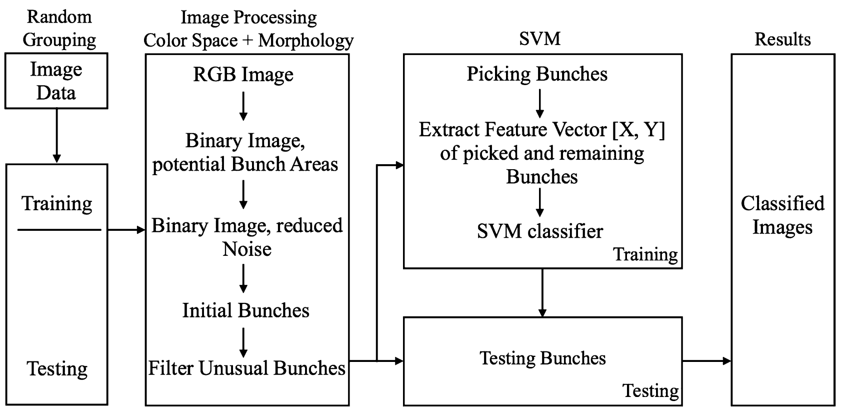


Fig. 2. Bunch detection flow chart proposed in this paper.

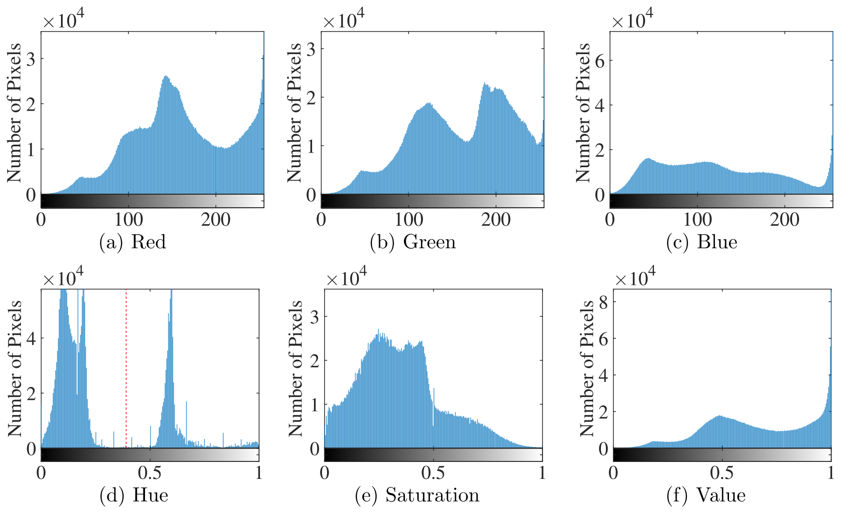


Fig. 3. Six histograms in RGB and HSV colour space of original figure (as shown in Fig. 1a). (For interpretation of the colours in this figure, the reader is referred to the web version of this article.)

hand as the training set for extracting the features. An SVM [3] is applied to segment bunches in the test set (the remaining images). Fig. 2 demonstrates the process flow for bunch segmentation and data analysis. The images taken in Block 11 were used to develop the algorithm.

2.1. Image pre-processing

In the image pre-processing stage, the first step is to eliminate irrelevant regions in the image by automatically thresholding the image in the H and V channels. In this first step, around 80% of irrelevant area is removed. The threshold is automatically adjusted according to the illumination condition in each image. Otsu's method [15] is implemented in the H and V channels to locate a threshold value automatically based on each image's condition, achieving two binary images. The reason for choosing HSV colour space is that local minimum is obvious in colour distribution in the H channel so grape and sky are easily automatically separated from other objects, as Fig. 3d (red dashed line) shows. As to sky area, this can be detected consistently from V channel (Fig. 3f). Compared with the B channel, the V channel is more robust for eliminating sky area from image since the pixel intensity distribution in the B channel has more local maxima and it is not consistent between images, as illustrated in Fig. 3c. After colour segmentation, potential bunch areas are obtained by calculating the intersection of these two binary images (as shown in Fig. 4a). Next by applying several morphological operations on this intersection, noise is reduced on a large

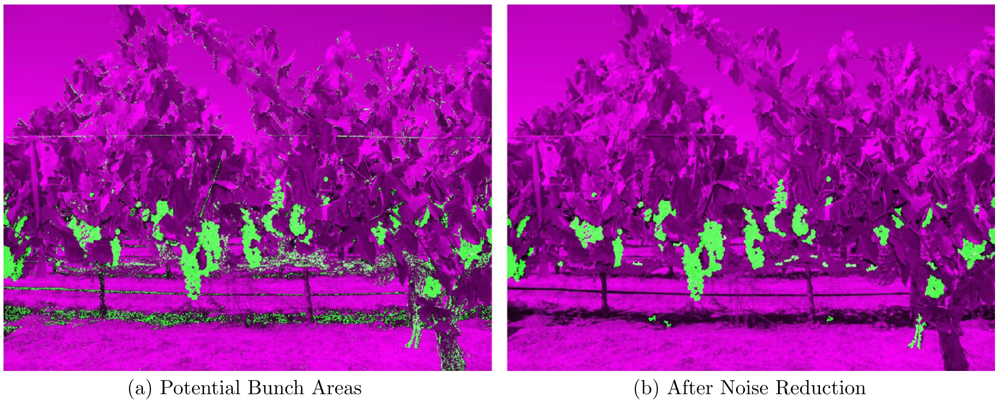

(a) Potential Bunch Areas (b) After Noise Reduction

Fig. 4. Colour segmentation and morphological operation in image pre-processing.

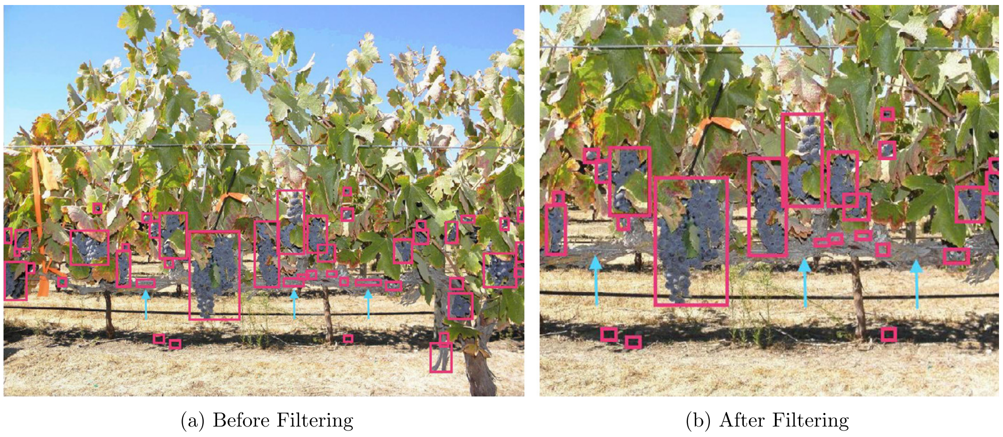

(a) Before Filtering (b) After Filtering

Fig. 5. Filtering, blue arrows point out the bunches removed by filtering, but prior to SVM classification. Image (b) is zoomed after filtering image (a). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

scale, as displayed in Fig. 4b. A new shape filter is defined in this paper by calculating the ratio between the width and height of the bounding box of these potential bunch areas. This filter is applied to eliminate the bunches with unusual shapes, as illustrated in Fig. 5. The ratio range of usual bunches is set as from 0.25 to 2.

2.2. Feature selection

80 images taken from Block 11 were pre-processed and there were still some false bunches, as illustrated in Fig. 5b (empty red boxes in shadow and on cordon). For feature selection, all real bunches in these 80 images were labelled by hand. The labelled bunches were set as true responses while the remaining unlabelled bunches were set as false responses.

For all initial bunches detected by image pre-processing, a feature vector with 58 dimensions is extracted for each potential bunch. This feature vector contains the closeness value, solidity, extent, and compactness of bunch regions (4 dimensions), and the texture information in three channels in RGB, HSV and L*a*b colour spaces, giving 54 dimensions in total. In order to simplify the classifier model, reduce memory usage and improve the computational speed, the ReliefF algorithm [17] and sequential feature selection [10] were applied for decreasing the feature dimensions. The ReliefF algorithm uses k-nearest neighbours to calculate the attribute importance and attribute weights for each feature, as demonstrated in Fig. 6. It demonstrates that it is reasonable to select predictors whose weights are above the threshold value 0.003. As for sequential

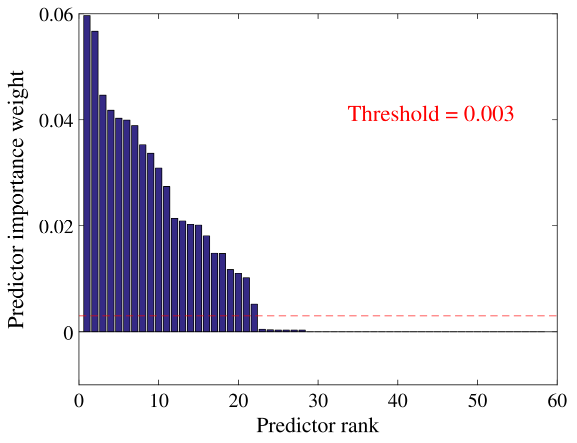


Fig. 6. Importance of predictors by ReliefF.

Table 1
Description of features selected for use in the final classifier, and the closeness is a new feature that is designed in this paper for bunch classification.

Feature	Description	Dimension
Closeness	Scalar that specifies the ratio between the y-position of each initial bunch and average y-position for all bunches in one image	1
Extent [8]	Scalar that specifies the ratio of pixels in the region to pixels in the total bounding box	1
Compactness [8]	Scalar that specifies the ratio of perimeter to the area of the region	1
Texture [8]	T1 — Average gray value	1 — h
	T2 — Average contrast	2 — h, s
	T3 — Measure of smoothness	3 — h, s, v
	T4 — Third moment	2 — h, s
	T5 — Measure of uniformity	1 — s
	T6 — Entropy	2 — h, s
In total		14

feature selection, misclassification rate was used as the criterion for minimising the feature subsets. To be more specific, sequential forward selection (SFS) is utilised by sequentially adding features to a feature candidate set. The criterion is continually calculated until the criterion stops increasing and 14 features are selected by SFS. The final feature candidates (14 dimensions) are decided by the intersection between two groups of features and they are demonstrated in Table 1.

A closeness value is also a newly defined feature in this paper. As Fig. 1 shows, most bunches are located in a certain position in the vertical direction. For each image, Otsu's method [15] is applied again to divide the y-positions (horizontal) of all initial 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, an SVM is implemented for analysing the size of the training set as part of the learning step. 80 images sampled in Block 11 were pre-processed automatically by the developed image processing algorithm. For verifying the effectiveness of proposed algorithm, these images were manually labelled in the final step as ground truth. The training range for number of images is set to [4, 76] and the interval is 4 images. For every potential bunch in each training set, a feature vector combining 14 features selected in the previous section and its response (true or false) are used as a classifier in a binary classification system (grape bunch or non-grape bunch). The confusion matrix for each training set is calculated for classification performance analysis:

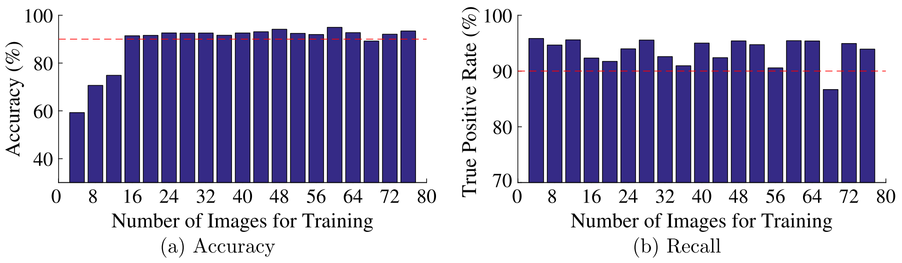


Fig. 7. Evaluation results for training size selection, based on the data set of Block 11.

Table 2
Image data information, properties of vineyards where experiments conducted.

Block	Images	Sampled vine	Photography time	Cultivar
11	80	20	28 Sep. 2013	Shiraz
19	80	20	09 Oct. 2013	Cabernet Sauvignon

- *True positive* (TP) refers a true grape bunch automatically classified by developed algorithm and also labelled as a real grape bunch by hand
- *True negative* (TN) annotates a grape bunch not detected by the algorithm and also manually labelled as non-grape bunch
- *False positive* (FP) represents a true grape bunch detected by the algorithm but which is actually not a real bunch
- *False negative* (FN) refers a bunch not classified by the algorithm but it is a real bunch labelled by hand

In order to getting the right number of images for training step, *accuracy*, *recall* and *precision* are used as justification by applying the SVM classifier obtained from training step:

- *Accuracy* (ACC) refers the percentage of true and false grape bunch automatically classified correctly in total population

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

- *Recall* (True Positive Rate, TPR) annotates the percentage of true grape bunches automatically classified in all manually labelled grape bunches

$$TPR = \frac{TP}{TP + FN} \tag{2}$$

All results for each training set are illustrated in Fig. 7. It is clear in Fig. 7a that 16 images is sufficient for achieving 90% accuracy for bunch classification.

3. Experimental materials and data scope

Field experiments were conducted in Camatta Hills, California, in Sep. and Oct. 2013. In total 160 images were shot under natural illumination field conditions. All images were photographed using an OLYMPUS Camera (Model: SP600UZ) with focal length 5 mm, internal flash and an automatic mode. The resolution for all images was 3968 × 2976. This vineyard is divided into several blocks, where each block contains a different variety of grapes. A total of 80 images were captured in Block 11 (Shiraz) while another 80 images were photographed in Block 19 (Cabernet Sauvignon), for which details are displayed in Table 2. Four

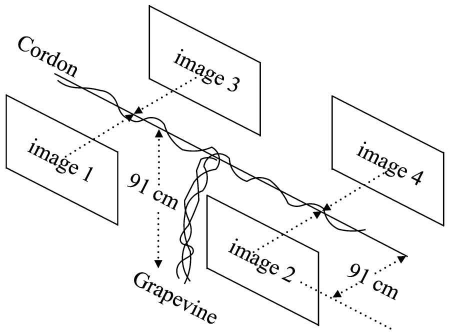


Fig. 8. Photography arrangement. Four images were taken by one camera for each vine. The camera was perpendicular to cordon. The height between ground and camera and the distance between camera and cordon were approximately 91 cm (inches).

Algorithm 1 Individual bunch area detection.

```
Input: Captured Image
Output: Detection Accuracy
 1: Randomly divide 80 images into Training (16 images) and Test groups
 2: for all images in Training group do
 3:     Image processing to get initial bunch areas
 4:     Feature extraction for all initial bunch areas
 5: end for
 6: Organise the data of features and response of Training group
 7: Apply SVM on Training data get the Classifier
 8: for all images in Testing group do
 9:     Image processing to get initial bunch areas
10:     Feature extraction for all initial bunch areas
11:     Classify initial bunch areas by the Classifier
12: end for
13: Organise the classification data of Test group
14: Calculate the detection accuracy  return Accuracy
```

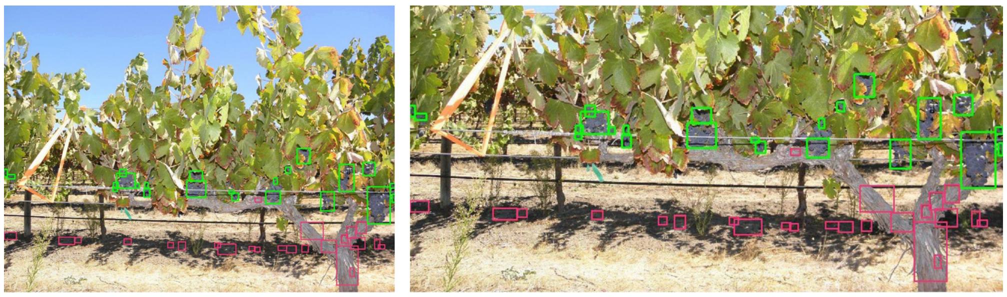


Fig. 9. Classification: green boxes are classified as true bunches while pink boxes are classified as false bunches. The picture in right column is zoomed from the picture in left column. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

images were taken for each sampled vine, including two arms on each side (East and West), as shown in Fig. 8. The distance between camera and cordon and the cordon height both were approximately 91 cm (36 inches).

4. Experimental results

The optimised features and the training size (16 images) were applied on another group of images captured in Block 19 at a different time of day. The image processing procedure followed the flowchart in Fig. 2 and the pseudo-code of the algorithm is illustrated in Algorithm 1. The detected results (green boxes) are demonstrated in Fig. 9.

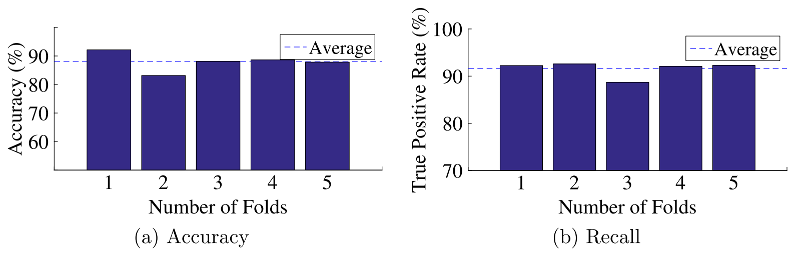


Fig. 10. Testing results of proposed image processing algorithm by five folds cross-validation, based on the data set of Block 19.

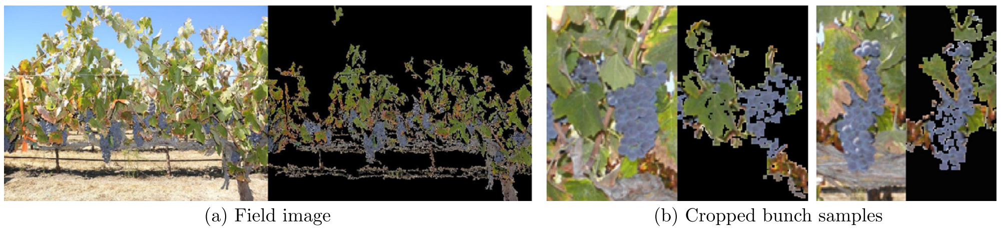


Fig. 11. Testing results on data set of Block 11 by the method presented in paper [16].

In order to avoid overfitting, for all 80 images of Block 19, again bunches were hand labelled as true bunches while the remaining unlabelled bunches were catalogued as false bunches. Then K-fold cross-validation was implemented for 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. So some images in these 80 images do not contain any bunches or just contain several small bunches, as shown in Fig. 1c. This means that training size is smaller than 16 images (as verified in the previous section). However, it still obtains the average accuracy and recall of 88.0% and 91.6% respectively by using 5 folds validation. The accuracy and recall are displayed in Fig. 10a and Fig. 10b.

5. Performance of the algorithm

The performance of proposed algorithm is compared with other two bunch detection methods from aforementioned literature. The image data set from Block 11 was tested using all three approaches.

Fig. 11 demonstrates two major results by the method presented by Reis [16]. There, the three central RGB values were obtained by trail and error. In this paper, central values are extracted by the following steps: firstly, all bunches in the first 16 images in data set are cropped out; secondly, the average RGB value of each bunch area is calculated; thirdly, K-nearest neighbour classification is applied for clustering these values into three groups; at last, for each group, the average RGB value is calculated, which consists of three central values. Then these three central RGB values are applied on the remaining 64 images. However, this method cannot segment bunches successfully, as Fig. 11a illustrates. Also, for meeting the requirements of controlled lighting condition in paper [16], several small images that only contain one major bunch are cropped out from one original image, and still, the major bunch cannot be segmented by this method (Fig. 11b).

The reason for this is that, even though under controlled lighting conditions, the central values are not enough for bunch segmentation, given the truth that grapes are not in the same maturing stage (the colours are different) even at the same time. Further, for the same type of grape, for instance, red grapes, the colour varies from light red to deep purple. As author described, there are more than 120 cultivars (including red and white), three central values for red and four for white are not sufficient for achieving good detection results. In addition, the authors criticised paper [1] for having an inevitable disadvantage in that this method has a slow training phase (since it needs humans to train data). However, to define these central values also

Table 3
The comparison of the proposed method and the approach in paper [1].

In total: 80 images	Paper [1]	Proposed method
Processed object	Cells (window size = 100 pixels)	Initial bunches
Feature Optimisation	No	Yes
Attributes Dimension	24	14
Time for training 16 images	113.4 sec	47.9 sec
Time for testing 64 images	473.6 sec	159.1 sec
Accuracy	54.83%	91.77%

is a training phase. It needs human hand to crop out grapes and calculate central values on average. Based on the fact of vineyards, to obtain the central value for 120 cases, you need to find enough central values for each cultivar. As to the perspective of saving time comparing with any supervised image segmentation techniques, the method proposed by paper [16] has no superiority, not to mention the accuracy.

Table 3 compares the performance of the proposed algorithm against the approach described in paper [1]. Both methods are implemented in Matlab on an Intel i5-3470 with a processor 3.2 GHz and the first 16 images are utilised for training. From the Table 3, it is clear that for the purpose of decreasing computational complexity, the proposed method saves the time for calculating feature vectors of 80% area (which is not insignificant area) in an images. The approach in paper [1] processes each cell in each image while the proposed method pre-processes images to get initial bunch areas and processes these initial bunches instead of cells for feature extraction. In addition, since the proposed algorithm can reduce the instance space in learning step, the training time for generating a proper classier is much quicker. Last, the accuracy of bunch detection achieved by the proposed algorithm is higher.

6. Conclusions and outlook

In this paper, a bunch segmentation algorithm, which is the procedure about processing images with grape bunches, is proposed. Secondly, one new filter and one new feature are designed for eliminating false detected bunches. Thirdly, optimisation of feature selection is performed for decreasing the dimension of attribute space. In short, 160 images were processed for bunch area detection algorithm development and verification. For each image, all bunches were identified manually to properly evaluate the results. Simple features were extracted for each image to save on computational load, and the feature dimension was optimised to 14 elements, thus showing which features are important for robust bunch detection. The training size was optimised to 16 images for testing the remaining images captured in the vineyards, and this size achieved a detection accuracy of 88.0% and recall of 91.6%. The algorithm is developed based on the data of Shiraz and tested on the data of Cabernet Sauvignon. These two data sets were took at different time of day, which means the illumination conditions for both are not identical, as shown in Fig. 1. Therefore the algorithm could be applied on other datasets taken from different blocks with different cultivars. However, it is limited to purple grapes and the detected bunch area may contain more than one bunch. Improving on these limitations is a key focus in the direction of our future work. The proposed image processing algorithm can be used for automatic yield estimation in viticulture. It can decrease the noise and is more computationally efficient at a large scale compared with existing approaches for automatic grape detail evaluation in later stages.

Acknowledgements

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. One author is partly supported by the Chinese Scholarship Council.

References

[1] R. Chamelat, E. Rosso, A. Choksuriwong, C. Rosenberger, H. Laurent, P. Bro, Grape detection by image processing, in: IECON 2006 – 32nd Annual Conference on IEEE Industrial Electronics, IEEE, 2006, pp. 3697–3702.
[2] C. Correa, C. Valero, P. Barreiro, J. Tardaguila, M. Diago, A comparison of fuzzy clustering algorithms applied to feature extraction on vineyard, in: J. Lozano, J. Gomez, J. Moreno (Eds.), Avances en Inteligencia Artificial, vol. 1(1), 2011, pp. 1–10.
[3] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge University Press, 2000.
[4] I. Dami, P. Sabbatini, Crop estimation of grapes, Tech. rep. HYG-1434-11, The Ohio State University, 2011.
[5] D. Dey, L. Mummert, R. Sukthankar, Classification of plant structures from uncalibrated image sequences, in: 2012 IEEE Workshop on Applications of Computer Vision, WACV, IEEE, 2012, pp. 329–336.
[6] M. Diago, C. Correa, B. Millán, P. Barreiro, Grapevine yield and leaf area estimation using supervised classification methodology on RGB images taken under field conditions, Sensors 12 (12) (2012) 16988–17006.
[7] C.C. Farias, C.V. Ubierna, P.B. Elorza, Characterization of vineyard's canopy through fuzzy clustering and SVM over color images, in: International Conference of Agricultural Engineering, Valencia, Spain, 2012.
[8] R.C. Gonzales, R.E. Woods, Digital Image Processing, 2nd edition, 2002.
[9] A. Khotanzad, Y.Y.H. Hong, Invariant image recognition by Zernike moments, IEEE Trans. Pattern Anal. Mach. Intell. 12 (5) (1990) 489–497.
[10] J. Kittler, Feature set search algorithms, in: Pattern Recognition and Signal Processing, 1978, pp. 41–60.
[11] S. Liu, S. Marden, M. Whitty, Towards automated yield estimation in viticulture, in: Proceedings of Australasian Conference on Robotics and Automation, Sydney, Australia, 2013, pp. 2–4, araa.asn.au.
[12] G. Loy, A. Zelinsky, Fast radial symmetry for detecting points of interest, IEEE Trans. Pattern Anal. Mach. Intell. 25 (8) (2003) 959–973.
[13] R.D. Stephen Martin, G. Dunn, How to forecast wine grape deliveries, Technique report, Department of Primary Industries, October 2003.
[14] S. Nuske, K. Gupta, S. Narasimhan, S. Singh, Modeling and calibrating visual yield estimates in vineyards, in: Field and Service Robotics, 2014, pp. 343–356.
[15] N. Otsu, A threshold selection method from gray-level histograms, Automatica 11 (285–296) (1975) 23–27.
[16] M.J.C.S. Reis, R. Morais, E. Peres, C. Pereira, O. Contente, S. Soares, A. Valente, J. Baptista, P.J.S.G. Ferreira, J. Bulas Cruz, Automatic detection of bunches of grapes in natural environment from color images, J. Appl. Log. 10 (4) (2012) 285–290.
[17] M. Robnik-Šikonja, I. Kononenko, Theoretical and empirical analysis of ReliefF and RReliefF, Mach. Learn. 53 (1–2) (2003) 23–69.
[18] J. Tardaguila, M. Diago, B. Millán, 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, vol. 978, 2012.