A Vision-Based Robust Grape Berry Counting Algorithm for Fast Calibration-free Bunch Weight Estimation in the Field

Scarlett Liu[a,b], Xiangdong Zeng[a], Mark Whitty[b,*]

[a]*School of Traffic and Transportation Engineering, Central South University, Changsha, Hunan 410075, China*
[b]*School of Mechanical and Manufacturing Engineering, UNSW Sydney, NSW 2052, Australia*

Abstract

Counting the number of berries per bunch is a key component of many yield estimation processes but is exceptionally tedious for farmers to complete. Recent work into image processing in viticulture has produced methods for berry counting, however these require some degree of manual intervention or need calibration to manual counts for different bunch architectures.

Therefore, this paper introduces a fast and robust calibration-free algorithm for berry counting for winegrapes to aid yield estimation. The algorithm was tested on 529 images collected in the field at multiple vineyards at different maturity stages and achieved an accuracy of approximately 89% per bunch. As it would mostly likely be used to obtain an average value across a block, the low bias of this method resulted in an average accuracy of 99% and was shown to be robust from pea-sized to harvest and between both red and green bunches.

Taking only 0.1 to 1 second per image to process and requiring only a smartphone and small backing board to capture, the algorithm can readily be applied to images which are captured in the field by farmers. This allowed bunch weights to be estimated to within 92% accuracy and assisted larger scale yield estimation processes to achieve accuracies of between 3% and 16%. The robustness of the method lays the foundation for fast fruit-set ratio determination and more detailed bunch architecture studies in-vivo on a large scale.

Keywords: grape bunch, bunch reconstruction, image processing, berry counting, bunch weight, yield estimation

1. Introduction

Automating yield component analysis is vital for improving yield estimation in viticulture since the current manual approaches can not meet the requirements of fast measurement and large sampling scale to secure the accuracy of grape production forecasting. The small sample size and lack of objectivity in interpreting the state of vine development also leads to poor accuracy in yield estimation in the wine industry. State-of-the-art manual sampling immediately prior to harvest result in errors from 3 to 30% [1] (Table 6.9), which anecdotally matches industry experience. Subsequently, wineries are forced to bear the cost of suboptimal tank space allocation, oak barrel purchases and contract adjustments as well as undertake the challenging task of managing harvest logistics within a decreasing harvest window. Hence, researchers in viticulture have been seeking solutions from image processing and computer vision to accelerate crop yield forecasting. Nuske *et al.* [2] presented image processing methods which were able to

*Corresponding author
Email addresses: scarlett.liu.academic@gmail.com (Scarlett Liu), eastward219@gmail.com (Xiangdong Zeng), m.whitty@unsw.edu.au (Mark Whitty)

generate unbiased estimates notably smaller than manual estimates.

Image processing has also been applied for grape bunch phenotyping in the context of breeding programs, and such phenotyping includes three quantitative methods of analysis [3]; the number of components, overall morphology of components and overall length of components, with components being berries, rachis internodes and other internodes *etc.* The number of berries remains stable after fruit set and it has vital impact on final yield for a bunch (Martin *et al.* 2003 [4]). Besides that, the ratio between bunch size and the number of berries per bunch is one of many factors governing the quality of the fruit at harvest. Given the number of berries per bunch and bunch weight are critical parameters for early forecasts of production we mainly focus on the yield component of berry number and its contribution to bunch yield estimation in this paper.

Currently, berry counting and bunch weighing across the grape growing season are accomplished by tedious and labour intensive manual measurements. To expedite this, two main approaches have been used: image based (2D, RGB images) berry counting and 3D point cloud sensor-based berry counting (by laser or RGB-D camera).

Under the category of 3D point cloud sensor-based methods, initial work was conducted by Florian and Volker [3] who presented a fully-automated sensor-based 3D reconstruction approach to phenotyping grape bunches. The proposed approach is able to generate a comprehensive bunch structure based on the 3D point cloud by iteratively optimizing parameters which define the bunch structure. As to berry number estimation, their approach was shown to achieve 12.35% error (see Table 1 in that work [3]). Later on, the same research group extended this work by developing software called "3D-Bunch-Tool" based on new lightweight 3D scanner [5] which can be utilized in the field. That software achieved 78.83% accuracy with $R^2 = 0.95$ ((see Table 2 in that work[5]) on lab-based berry counting and the process of scanning in the lab took approximately 1 minute. Field based scanning, observing only one side of a bunch meant approximately 50% of berries were observed, and these were correlated to the total number of berries from a 360deg scan with an R^2 value of 0.83; the actual error in terms of berry count was not presented in the paper.

The intricacy of the 3D scanning approach and cost of the sensors has meant considerable focus has been given to the imagery based solution in the field. Liu *et al.* [6], Diago *et al.* [7] and Ivorra *et al.* [8] showed how yield component analysis could lead to more efficient forecasts using image processing. Kicherer *et al.* [9] presented the Berry Analysis Tool (BAT) for counting berry number, diameter and volume, which is reliant on destemming a bunch and arranging berries on a performated metal plate in laboratory conditions. Grossetete *et al.* [10] and later Diago *et al.* [7] processed RGB images to count berries using a photo of one side of a bunch, obtaining R^2 values of 0.92 and 0.82 respectively between the real and detected number of berries. The work presented by Diago *et al.* [7] was tested with a dataset of ten images for each of seven varieties, with R^2 values varying from 0.62 to 0.95 with an average of 0.82 across the seven cultivars. Aquino *et al.* [11] developed an algorithm to detect visible berries from a single bunch photo in the field, achieving $F1$ score $= 0.89$ based on their best parameter settings. As for actual berry estimation, that paper showed results of $R^2 = 0.75$ and an accuracy of 84.35% between estimated berries and actual berries. This work was extended into an app [12] which was not available online at the time of writing. However, the image processing algorithms proposed by Grossetete *et al.* [10] and Aquino *et al.* [11] rely on a specular reflection at a single point on each berry and are not robust following veraison since the surface of the berries may become matte and in some cases shrivelled.

Besides estimating the number of berries by processing single 2D images, 3D bunch reconstruction has also been achieved using stereo imagery [13]. There, two 3D bunch models were built with substantial manual input and the method achieved $R^2 = 0.797$ using a point model and $R^2 = 0.778$ against a CAD model. This customized stereo camera arrangement also has a natural minimum range and limits applicability to *ex-vivo*[1] analysis and maneuvering such a setup within a sprawling canopy is impractical.

Commercial mobile solutions have became the main objective which is challenging the robustness of existing image processing algorithms [11]. A common approach relies on a backing board with contrasting color to the bunch. As for actual berry counting, the most common approach is to estimate the occluded berries based on the detected number of visible berries [11]. This needs calibration and varies between cultivars and lighting con-

[1] *in-vivo* (also referred to as in-field) is used in this paper to refer to experimentation done without removing bunches from the shoot while *ex-vivo* means the condition that bunches are detached from the shoot

ditions since most existing algorithms are sensitive to illumination changes. Cultivar dependency of this calibration has not been investigated to date given the tedious data collection procedure.

In order to relieve the burden of building calibrations for each cultivar, Liu *et al.* [14] proposed a novel approach to count berries from a single image. Their method is limited to red grapes and can only deal with conical or cylindrically shaped bunches because the reconstruction procedure only follows the main branch of the bunch. A range of berry radii needs to be manual defined using their approach. The tested images were collected under laboratory conditions.

Hence with the assistance of a backing board and under the condition that the bunch can be segmented out from a high contrast backing board by mature bunch segmentation solutions [15, 16, 17] this paper focuses in particular on the robust berry counting solutions for both red and green bunches and its contribution to bunch weight estimation.

For direct comparison with other approaches, we provide the benchmark of collected bunch images and related metrics. The datasets are published[2] on the Smart Robotic Viticulture group's website[3].

In the remainder of this paper, Section 2 presents a field-robust algorithm for *in-field* berry counting based on a single RGB image, catering

[2]http://www.robotics.unsw.edu.au/srv/datasets.html
[3]http://www.robotics.unsw.edu.au/srv

for red and green bunches with a range of berry diameters. Section 3 describes the experimental data and procedures used to validate the algorithm. Section 4 shows the accuracy of the results, as well as evaluating the robustness of the algorithm to differences in development stage, the contribution of berry counting to bunch weight estimation, and the possibility of yield estimation by image-based berry counting. Section 5 then draws conclusions and makes recommendations for future work.

2. Methodology

In general, berry counting is divided into three steps, Region of Interest (ROI) extraction, visible berry detection and actual berry count estimation, which are demonstrated in Figure 1. We propose a novel algorithm for 3D bunch reconstruction based on a single image for fast berry counting in vineyards. According to the flowchart in Figure 1, the proposed approach starts with sub-bunch detection then processes each sub-bunch [4] before calculating the bunch sparsity factor (mentioned in Section 2.4) and reconstructing the 3D bunch model (explained in Section 2.3).

The proposed berry counting procedure is divided into two parallel steps once the sub-bunches are defined by color segmentation: on one hand, obtaining the initial 3D bunch model to get a berry number (see Figure 2); on the other hand, calculating the sparsity factor based on the difference of two color channels. The final berry number is obtained by combining this

[4] A sub-bunch refers to the separate sections of the bunches which are visually disconnected in the image other than by rachis structure

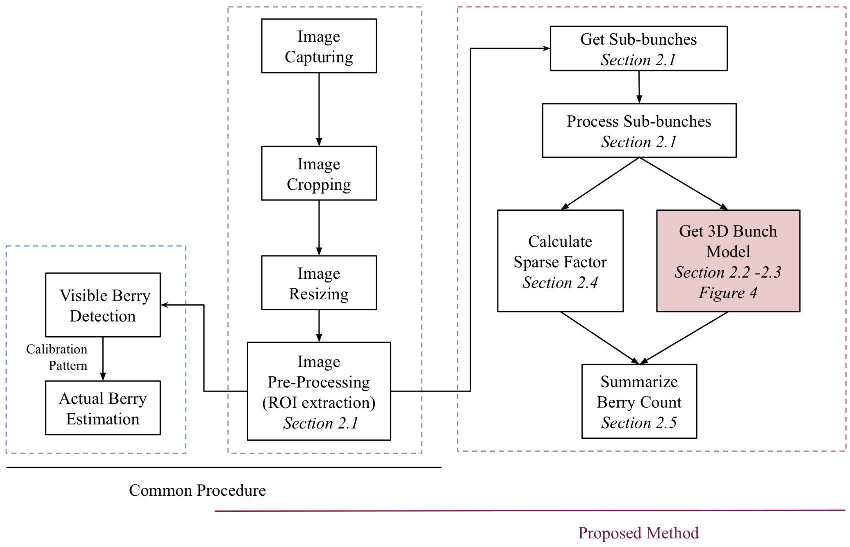


Figure 1: The proposed fast berry counting approach by 3D bunch reconstruction

initial berry number and the sparsity factor. Figure 2 gives an overview and Section 2.3 and Section 2.4 give detailed explanations of these steps. For convenience, the list below provides the explanation of major variables referred to in this paper:

Nomenclature

σ	w/l, Bunch width / Bunch length
a_r	Sum of areas in pixels calculated by Red channel from RGB color
a_s	Sum of areas in pixels calculated by Saturation channel from HSV color space
a_v	Sum of areas in pixels calculated by Value channel from HSV color space
b_e; b_{g_i}	Estimated berries at the bunch edge; Estimated berries in each group
B_x	Estimated Sub-bunch
bw_e	Binary image of B_x's edge
C	Histogram bin counts
E	The value of bin edge
e_p, e_a	The percentage error and absolute error, see Equations 3 and 4
f	Sparsity factor of a bunch or sub-bunch
g_i	Groups of berries at the bunch edge
I_g	Grayscale image of B_x
id	Index
k	Curvature for each pixel
L	The set of edge line segments

n_e, n_a Estimated and actual berry number per bunch/sub-bunch

n_i Initial berry number per bunch or sub-bunch

P Points extract from L

Po Fitted polygon

R; R^2 Radius of berries; R-square value, the goodness of fit

r, c Row and column numbers of an image

s Step size for 3D position adjusting of a berry along a track

t Tolerance of berry overlapping

tmp Temporary count

v The metric value of detected berries by Hough Transform

w_b Weight of berry

w_{B_e}, w_{B_a} Estimated bunch weight and actual bunch weight

x, y, z Coordinates of a berry

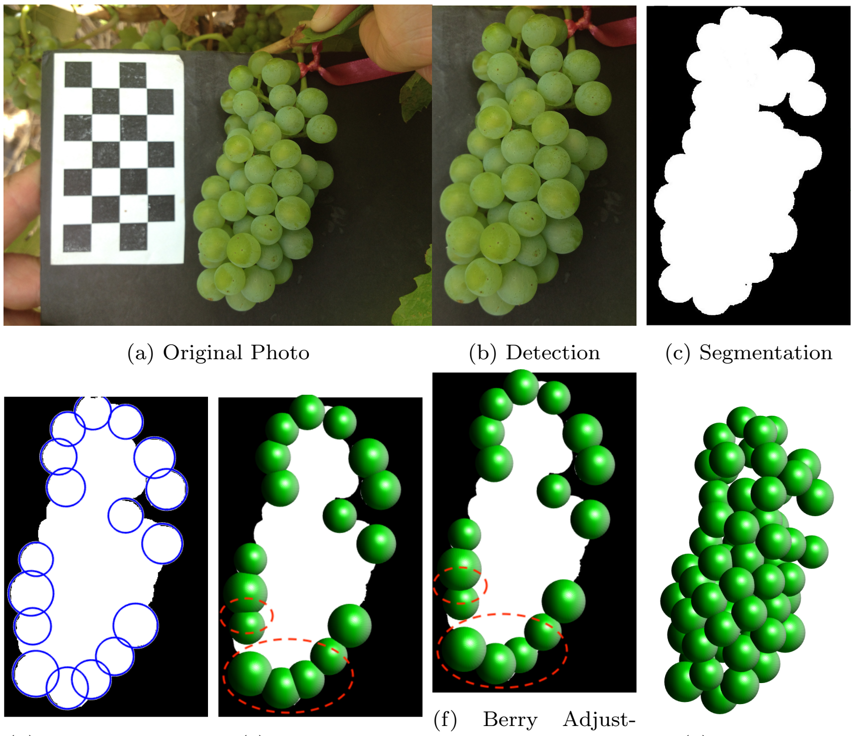
2.1. Sub-bunch Segmentation

A backing board is used during image collection to aid segmentation and is recommended to be of contrasting color with the berries. In this work, a black backing board was utilized for green bunches while a white backing board was used for red bunches. The backing board provides good contrast which leads to better bunch identification.

Alternatively, bunch detection (Figure 2b) can be done by the methods described by Luo *et al.* [15, 16] or Perez-Zavala *et al.* [17] amongst many existing approaches. In this paper ROI extraction (Figure 2c) is conducted by Otsu's method [18], which is commonly adopted for bunch segmentation [14, 11]. The non-connected ROIs are treated as sub-bunches and labelled as B_x, and for each n_i and f are calculated individually.

2.2. Robust Berry Detection

A vital step in berry detection is that of initializing the range of radius of berries for applying the Hough Transform. Approaches presented by Mirbod *et al.* [19] and Dahal *et al.* [20] could be applied on bunches with a high-reflection spot (from artificial lighting or under lab conditions), but this was not possible for the field-based datasets in this paper. For a more practical usage, we applied Sobel edge detection followed by several basic morphological operations [21], which includes removing small objects from binary image (in this case those less than 100 pixels in area) and connection checking between segmented lines (in this case those that have less than a 5 pixel gap), to extract the edges of visible berries (Figure 3b).



(a) Original Photo (b) Detection (c) Segmentation

(d) Circle Detection (e) Berry Filling (f) Berry Adjustment (g) 3D Bunch

Figure 2: Steps in the detailed bunch reconstruction process, illustrated with an *in-vivo* image of a Chardonnay bunch. The original photo (a) has been cropped (b) and segmented from the background (c). From this segmentation outline, berries on the exterior are fitted (d) where the outline has appropriate curvature. Where these berries overlap (e), they are adjusted forwarded or backwards until there are no collisions (f). Then the remaining berries are placed inside the hull formed from the segmented outline as per Figure 4.

We then propose a new algorithm (Algorithm 1) to estimate the initial range of berry radii for edge berry and internal berry detection. Part curvature calculation is demonstrated in Figure 3c; showing that a large proportion of the edge points (k) have similar radii. This guides the choice of the initial berry radius range as described in Algorithm 1.

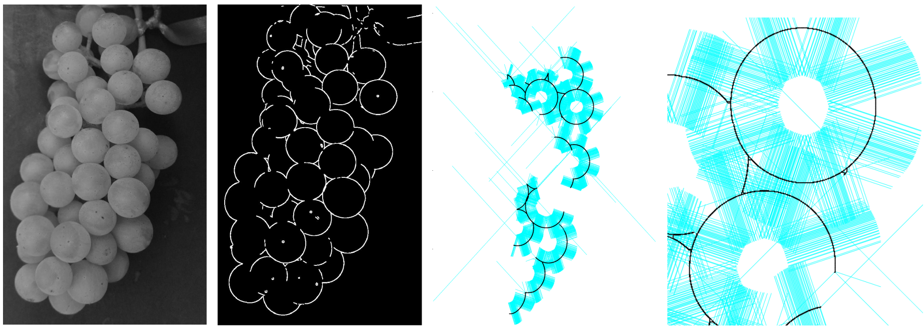
Algorithm 1: Estimation of initial berry radius range

Data: I_g of B_x
Result: Radius range for berry detection, R

$bw_e \leftarrow$ *edge detection by Sobel [21]* ; `// see Figure 3b`
$L_i \leftarrow bw_e$; `// The set of edge line segments`
$tmp \leftarrow 0$;
for $i \leftarrow 1$ **to** `findNum(`L_i`)` **do**
    $P_j \leftarrow$ `discrete(`L_i`);`
    for $j \leftarrow 2$ **to** `findNum(`P_j`)`*-1* **do**
        $tmp \leftarrow tmp + 1$;
        $Po_j \leftarrow$ `fit(`P_{j-1}, P_j, P_{j+1}`);`
        $k_{tmp} \leftarrow$ `cur(`Po_j`)` ; `// Curvature from the fitted polygon`
    end
end
$k \leftarrow k($`abs(`k`)`> 0 AND `abs(`k`)` $< Inf)$; `// Filter curvature`
$R \leftarrow 1./k$; `// Calculate radii for each edge pixel`
$[C, E] \leftarrow$ `histcounts(`R`);` `// Histogram of radii`
$id \leftarrow$ `FindMax(`C`);` `// Find the bin with maximum counts`
$R \leftarrow$ `Round(`$[E(id-1), E(id+1)]$`);` `// Specify a range of radii from bin widths`

2.3. Sub-bunch Reconstruction

Figure 4 shows the process of how each sub-bunch, B_x, is reconstructed to give an initial berry number.



(a) Gray Bunch (b) Edge Detection (c) Curvature (d) Detail of curvature

Figure 3: Estimation of initial berry radius range. Cyan lines indicate the magnitude and direction of the normal of each edge pixel.

The berries at the edges are detected by the Hough transform [21](Figure 2d), aided by the algorithm in Section 2.2. Those berries are defined as b_e and the diameter and location of each one are used to place corresponding spheres in a plane parallel to the image sensor as shown in Figure 2e. Neighbourhood searching [22] using a threshold of the largest estimated berry radius is applied for grouping the overlapping berries in our proposed edge berry adjusting algorithm 2. For each group, berries are sorted based on the likelihood of being a circle, as calculated by the Hough transform. The berry with largest likelihood is moved closer the camera (along the z-axis) while the berry with smallest metric is moved further from the camera until there is no overlap (Figure 2f). The overlap has an additional tolerance, t, added as a proportion of the berry radius.

Once the edge berries have been positioned, the B_x is divided into two parts if the ratio between the width and length of a bunch is larger than $3:4$. A watershed point of the divided bunch is formed 3/4 of the way

Algorithm 2: Berry adjustment at bunch edges

input : A binary image bw_e of bunch with size $r \times c$, step size s, tolerance t

output: 3D location (X_e, Y_e, Z_e) of detected berries at a bunch edge

$bw_e \leftarrow$ *bunch segmentation*;
$b_e[x, y, R, v] \leftarrow$ `hough`(bw_e);
$g_i \leftarrow b_e$; `// Nearest Neighbor Search[22]`
for $i \leftarrow 1$ **to** `findNum`(g_i) **do**
    $b_{g_i} \leftarrow$ `sort`$(g_i(v))$;
    $v_m \leftarrow$ `findMedian`$(g_i(v))$;
    $b_{g_{im}} \leftarrow$ `Find`$(g_i(v) = v_m)$;
    for $j \leftarrow 1$ **to** `findNum`(b_{g_i}) **do**
        down $\leftarrow$ moving berry backward along the z-axis by s;
        up $\leftarrow$ moving berry forward along the z-axis by s;
        stay $\leftarrow$ don't move;
        keep $\leftarrow$ save current $b_{g_{i_j}}(x, y, z)$ in a stack (X_e, Y_e, Z_e);
        if $b_{g_{i_j}}(v) < b_{g_{im}}(v)$ **then**
            while $b_{g_{i_j}}(x, y, z)$ *collides with* (X_e, Y_e, Z_e) *by* t **do**
                down $(b_{g_{i_j}}(x, y, z))$
            keep $b_{g_{i_j}}(x, y, z)$
        else if $b_{g_{i_j}}(v) > b_{g_{im}}(v)$ **then**
            while $b_{g_{i_j}}(x, y, z)$ *collides with* (X_e, Y_e, Z_e) *by* t **do**
                up $(b_{g_{i_j}}(x, y, z))$
            keep $b_{g_{i_j}}(x, y, z)$
        else
            stay $(b_{g_{i_j}}(x, y, z))$

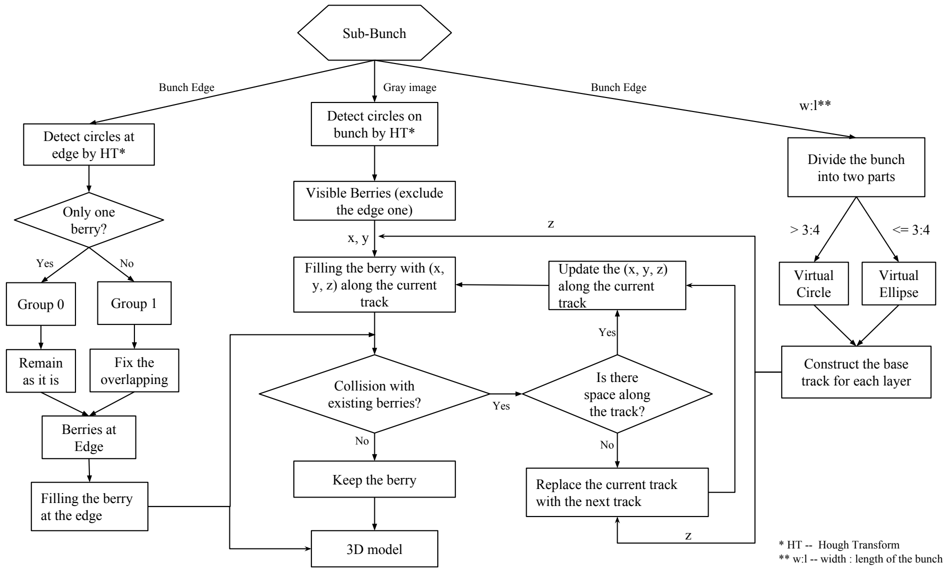


Figure 4: Flowchart of the proposed 3D bunch reconstruction algorithm based on a single sub-bunch

down the length of the longitudinal axis. This value has been defined empirically, based on the available data. Then, virtual tracks are formed to place berries by finding the first and last pixel of a section through the width of the B_x. This section is revolved about the longitudinal axis through its middle, forming the virtual track. If the B_x has been divided into two parts, the upper part is revolved as an ellipse whereas the lower part is revolved as a circle. This provides the z value for each berry candidate. The video attached to this article simulates this process.

The next step is detection of all visible berries from the gray image of the B_x, again using the Hough transform. Berries at the edges are subtracted and the remaining berries are placed where they are detected. Their z position is provided by virtual track generated in the proposed algorithm to form the three dimensional position for each berry.

The next step is to fill non-visible or interior berries in a shell of tracks that is generated based on image processing. The radius for each populated berry is chosen according to the distribution of radii detected from all visible berries and assumes the berries are all spherical. Starting at the top detected berry and moving around the circumference of each virtual track, placement of a new berry is attempted at regular intervals (1 degree is used in this work) and is considered successful if no collision with any existing sphere is detected.

The berry placement moves down the bunch by a step size defined as s (chosen to be two pixels in this work) and repeats the placement attempts on the track below until the bottom of the bunch is reached and the model is complete. The number of berries placed is tallied to form the Initial Berry Number (n_i).

2.4. Sparsity Factor Calculation

The initial 3D model is built based on the assumption that the processed bunch is healthy and compact. When berries fill the convex hull estimated from a single image there are no concerns about the 'empty space'. This means that berries are tightly packed in the initial 3D bunch model and does not allow for any of the rachis structure, resulting in an overestimate of berry number. An extended sparsity factor (f) is proposed to process both red and green grapes according to the visible proportion of berries within the bunch area — a proxy for bunch compactness:

$$f = \begin{cases} (a_r - a_s)/a_r & \text{red} \\ (a_s - a_v)/a_s & \text{green} \end{cases} \tag{1}$$

2.5. Berry Count Estimation

The estimation of the number of berries is improved through the application of the extended sparsity factor according to the following formula:

$$n_e = (1 - f) \times n_i \tag{2}$$

where n_e is the final estimate of the berry number per sub-bunch. This is then tallied across the sub-bunches to give the final number of berries in the entire bunch. Both of the equations above were determined empirically, and found to be appropriate for all the datasets tested.

3. Data Scope and Experimental Design

In total, 529 bunch images from two cultivars were tested and the details of each dataset are illustrated in Table 1 including whether *in-vivo* or *ex-vivo*[1] and which model of smartphone was used. All bunches were photographed in the field (whether *in-vivo* or *ex-vivo*) without artificial lighting, replicating end-user usage, the only requirement being the use of a backing board to aid segmentation. The proposed 3D reconstruction algorithm and sparsity factor calculation was implemented in Matlab (R2018b, Mathworks, MA, USA) and a PC with Intel Core i5-6500 and 16 GB RAM was used to process all the images to obtain the final estimates of the number of berries (n_e).

The proposed method was validated in two aspects regarding berry number and one aspect regarding bunch weight as well as in application to yield estimation:

- A1, the accuracy of the berry counting algorithm for bunches of different colours
- A2, the robustness of the algorithm to different development stages of bunches
- A3, the accuracy of bunch weight estimation, derived from fast berry counting
- A4, the accuracy of yield estimation from bunch weight estimates

Table 1: Details of datasets D1 to D8 used in this paper

Dataset	**Num. of images**	***in/ex-vivo*** [1]	**Cultivar**	**Color**	**Development (E-L) Stage [23]**	**Location**	**Date Imaged**	**Smartphone**	**Resolution [pixels]**	**Aims**
D1	94	*ex-vivo*	Chardonnay	green / light yellow	38 Harvest	Clare Valley, SA, Australia	28/02/2017	LG G3	4160*2340	A1, A3, A4
D2	73		Shiraz	red / light yellow			28/03/2017			
D3	86		Chardonnay	green / light yellow		Orange, NSW Australia	02/03/2017	Google Nexus 5X	4032*3024	
D4	44	*in-vivo*	Shiraz	red		Clare Valley, SA, Australia	10/02/2015	iPhone 5	2448*3264	A1, A2
D5	56					Orange, NSW Australia	23/02/2015		1536*2048	
D6	63			green	31 Berries pea-sized	Clare Valley, SA, Australia	30/12/2014	iPhone 4S	2448*3264	
D7	56		Chardonnay		33 Berries still hard and green		29/12/2014			
D8	57			green	34 Berries begin to soften		05/01/2015			

Each photographed bunch was deconstructed and the number of berries counted manually with shrivelled or substantially smaller berries being excluded from the count.

For comparison of the bunch weight, this was calculated using the estimated berry number and average berry weight: $w_{B_e} = n_e * w_b$ and then compared with the measured bunch weight. In this dataset, the average berry weight was calculated by weighing five berries individually from each bunch; in practice a number of alternate methods are available to determine average berry weight.

With reference to the nomenclature section above, the evaluation indicators used in this paper are:

- The R^2 value, based on a linear correlation between the actual and estimated number of berries, which can be used to reflect the goodness of fit between these two groups of numbers.

- The percentage error $e_p(\%)$, which is defined as:

$$e_p(\%) = \begin{cases} (n_e - n_a)/n_a * 100 & \text{berry counting} \\ (w_{B_e} - w_{B_a})/w_{B_a} * 100 & \text{bunch weight estimation} \end{cases} \quad (3)$$

  In many cases, the user would desire an average berry count per bunch over a number of samples in the block, so this metric is used as an estimate of the bias of the average value and thus the robustness of the approach in practice.

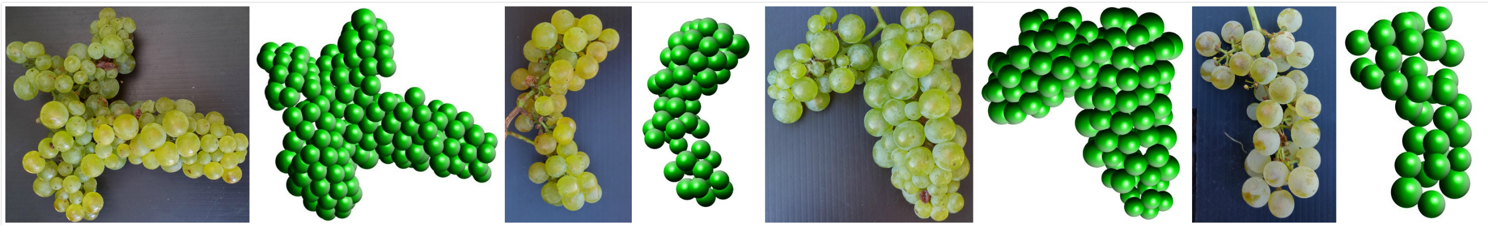
- The percentage of the absolute error $e_a(\%)$, which is defined as:

$$e_a(\%) = \begin{cases} |n_e - n_a|/n_a * 100 & \text{berry counting} \\ |w_{B_e} - w_{B_a}|/w_{B_a} * 100 & \text{bunch weight estimation} \end{cases} \quad (4)$$

  This measure is best suited for determining the accuracy of the method for a single image as errors are not balanced out when averaged over a large number of bunches.

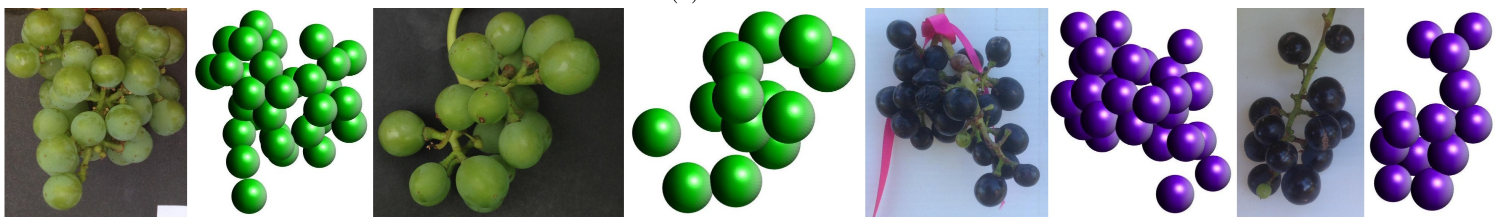
4. Experimental Results and Discussion

Using the computer described above, each image took 0.1 seconds to be processed, without any code optimisation. Qualitatively, manual observations of the real and reconstructed bunches matched quite well, as the proposed method fits berries around the outer profile of the bunch, similar to common bunch architectures. Figure 5 demonstrates some 3D bunch models reconstructed by the proposed method. Note the variation in the illumination conditions and bunch structure that exists among the original photos and thereby the robustness of this method to handling both symmetric and non-symmetric bunches. It should be emphasised that the reconstruction is an estimate and the berries may appear smaller due to the shading used. The results clearly show (Figure 6) the number of berries calculated is reasonably accurate. The exact positioning of berries not detected from one view is only an estimate.



(a) Green bunches

(b) Red bunches



(c) Looser bunches

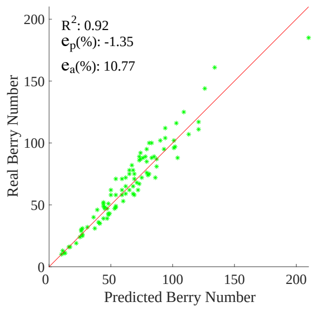
Figure 5: Reconstructed 3D models of bunches for both green and red bunches. Images were collected in the field by smartphones and the variation in lighting conditions and bunch architecture can be seen.

4.1. Accuracy of Berry Counting (A1)

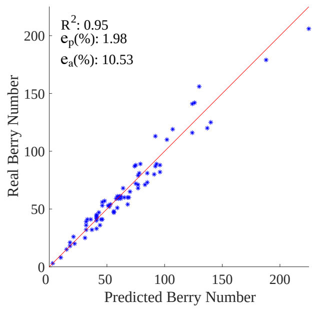
Figure 6 represents the quantitative relationship between estimated and actual berry numbers found from the processed images of both red and green cultivars. Eight datasets (Table 1) were tested and the detailed results are illustrated in Table 2. R^2 values varied from 0.82 to 0.95 (average 0.91), along with the percentage of absolute error 10.2% $\sim$ 15.26% (average 11.85%) and the percentage of error -6.0% $\sim$ 8.16% (average 3.1%), indicating the good performance of the proposed fast berry counting algorithm. A slight decrease in the performance of the algorithm was seen in-vivo (datasets D4-D5) as opposed to ex-vivo (datasets D1-D3), most likely due to more variable illumination and a smaller datasets more prone to outliers.

Table 2 compares the proposed method with five state of the art methods presented in the literature. The methods proposed by Diago *et al.* [7]and Aquino *et al.* [11] require calibration of the relationship between visible and non-visible berries, which varies between cultivars and development stage. The approaches presented by Herrero-Huerta *et al.* [13], Schőler *et al.* [3] and Rist *et al.* [5] need human interaction with software or at least the tuning of parameters. Alternatively, the method presented in this paper is not-cultivar or development stage specific and requires no human interaction once the image has been acquired. Furthermore, the proposed method works accurately with both red (Shiraz) and green (Chardonnay and preveraison Shiraz) cultivars and is able to achieve R^2 values on par with the existing work without the drawbacks mentioned above. In addition, direct comparison of the accuracy ($e_p(\%)$ and $e_a(\%)$) shows that the accuracy is at least as high as the existing work as shown in Table 2. Results presented in the original papers are directly provided for comparison, as the corresponding datasets have not been made available[5].

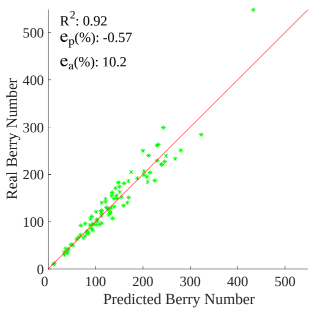
[5]Our data: `http://www.robotics.unsw.edu.au/srv/datasets.html`



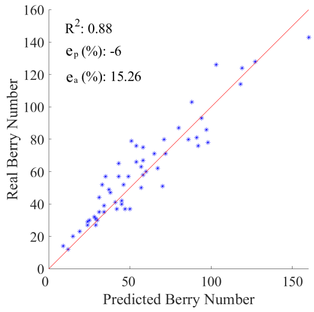
(a) D1, Chardonnay



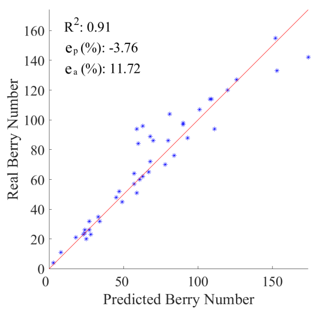
(b) D2, Shiraz



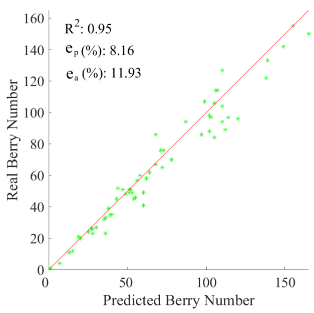
(c) D3, Chardonnay



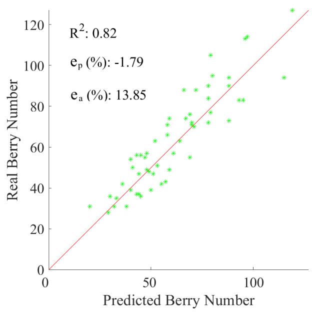
(d) D4, Shiraz



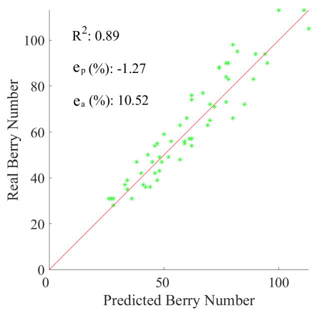
(e) D5, Shiraz



(f) D6, Shiraz (green)



(g) D7, Chardonnay



(h) D8, Chardonnay

Figure 6: The accuracy of actual berry counting both for green and red grapes (A1). The red line indicates a 1:1 relationship.

Table 2: Comparing existing algorithms with the presented algorithm for berry counting

Method	**Cultivar**	**Color**	**Number of Bunches**	**Process Time per Bunch**	R^2	$e_a(\%)$	e_p(%
Schőler *et al.* [3]	Riesling	Green	4	$> 10m$	NA	12.35	3.30
Rist *et al.* [5]	Calardis Blanc, Dornfelder, Pinot Noir and Riesling	Red and Green	12 per cultivar	$\approx 1m$	0.94	NA	NA
	Calardis Blanc	Green	222		0.95		-21.4
Diago *et al.* [7]	7 common European cultivars	Red	10 per cultivar	$\approx 2s$	0.82	NA	NA
Aquino *et al.* [11]		Green	11 per cultivar	$\approx 0.0407s$	0.75	15.65	
Herrero-Huerta *et al.* [13]	Tempranillo	Red	20	slow	0.78	NA	
Proposed Method	Chardonnay, D1	Green / Light Yellow	94	$\approx 0.1s$	0.92	10.77	-1.35
	Shiraz, D2	Red	73		0.95	10.53	1.98
	Chardonnay, D3	Green / Light Yellow	86		0.92	10.20	-0.57
	Shiraz, D4	Red	44		0.88	15.26	-6.00
	Shiraz, D5		56		0.91	11.72	-3.76
	Shiraz, D6	Green	63		0.95	11.93	8.16
	Chardonnay, D7		56		0.82	13.85	-1.79
	Chardonnay, D8		57		0.89	10.52	-1.27

In addition, the code was converted into an iOS app (3DBunch) tested on an iPad mini 4 with an Apple A8 processor, PowerVR GX6450 GPU, 2G LPDDR3 RAM using iOS version 12.3.1. 100 photos were tested on that device and the average processing time for each image was approximately 1 second (excluding human interaction). The average absolute error of berry counting for those 100 images is around 8% and the average percentage error is 2.6%. Discrepancies between the desktop and mobile versions of these results are only due to limited image processing library functions on mobile operating systems. Detailed experimental results from the mobile platform are expected to be provided in future work.

4.2. Robustness to Development Stage (A2)

Since the compactness or sparsity of the bunches varies as they grow, the accuracy of the 3D reconstruction method was examined in the context of these different growth stages and the results are shown in Table 3. Datasets D4 to D8 were of images captured *in-field* and their development stage varied from E-L stage 31 (Pea-size stage) to E-L stage 38 (Harvest) following the Modified E-L naming convention [23]. The absolute average error from lag-stage to harvest stage for the green bunches was in the range of 10.20% — 15.26% with an average of 12.66%, very similar to the harvest stage results. Hence, this method is robust to different development stages, back as far as pea-sized bunches.

The ability of the proposed method to accurately estimate the berry number in the early stages of development allows the user to rapidly estimate current bunch weights non-destructively.

Table 3: Performance of the algorithm for bunches at different development stages (A2)

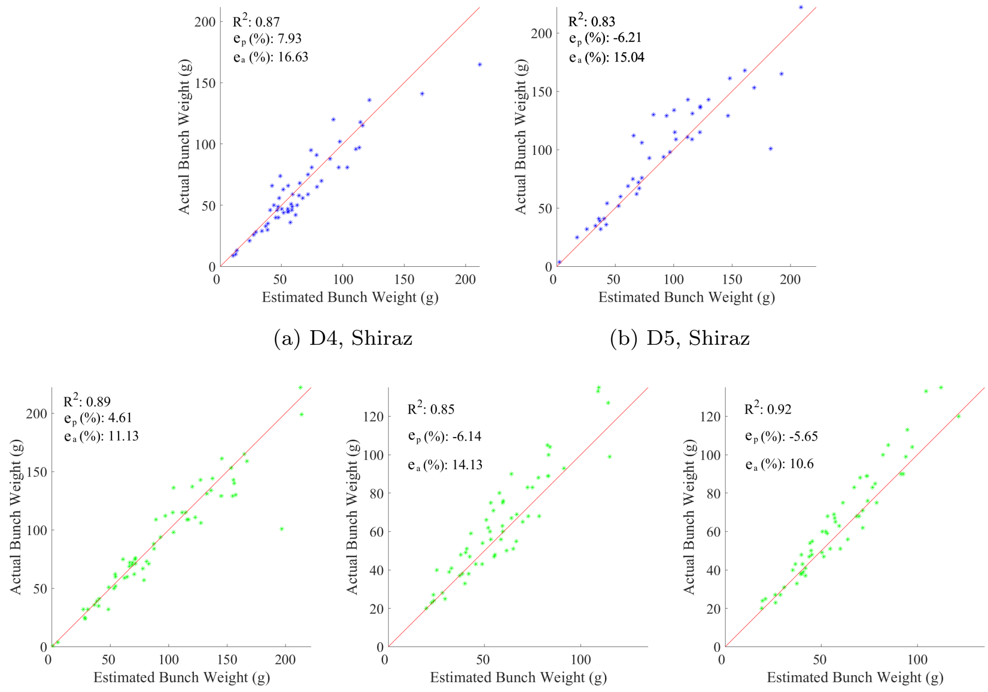
Dataset	**Cultivar**	**Color**	**E-L Stage**	R^2	$e_a(\%)$	$e_p(\%)$
D4	Shiraz	Red	38 Harvest	0.88	15.26	-6.00
D5				0.91	11.72	-3.76
D6		Green	31 Berries pea-size	0.95	11.93	8.16
D7	Chardonnay		33 Berries still hard and green	0.82	13.85	-1.79
D8			34 Berries begin to soften	0.89	10.52	-1.27

4.3. Accuracy of Bunch Weight Estimation (A3)

Figure 7 and Table 4 show the results of the algorithm being applied to estimate bunch weights. The results demonstrate high correlation (R^2 ranges from 0.83 — 0.92) between estimated bunch weight and measured bunch weight. For an individual bunch, the errors were larger than by direct comparison with berry number, but this is to be expected given the uncertainty in berry weight. Averaged over several dozen bunches, the error reduced to less than 8%, suggesting this is a feasible method for farmers to rapidly obtain a measure of average bunch weight non-destructively.

4.4. Yield Estimates Utilising Berry Counts (A4)

We then applied the algorithm along with a shoot counting method [24] to assist grape yield estimation in 2017 at two different vineyards, using the yield estimation method and data specified in Whitty et al. [1]. In summary, shoots were counted from a mobile camera across the whole block, then in



(a) D4, Shiraz

(b) D5, Shiraz

(c) D6, Shiraz (green)

(d) D7, Chardonnay

(e) D8, Chardonnay

Figure 7: A3, the accuracy of bunch weight estimation based on the proposed berry counting algorithm. The red lines indicate 1:1 relationships.

Table 4: Performance of the reconstruction algorithm when comparing estimated and actual bunch weights

Dataset	**Cultivar**	**Color**	**E-L Stage**	R^2	$e_a(\%)$	$e_p(\%)$
D4	Shiraz	Red	38 Harvest	0.87	16.63	7.93
D5				0.83	15.04	-6.21
D6		Green	31 Berries pea-size	0.89	11.13	4.61
D7	Chardonnay		33 Berries still hard and green	0.85	14.13	-6.14
D8			34 Berries begin to soften	0.92	10.60	-5.65

Table 5: Yield estimation assisted by the proposed berry counting method in 2017 for three blocks. Two methods of determining berry weight at harvest are compared, along with a standard industry approach for yield estimation.

Block		**40A**	**47A**	**B12**
Actual yield (t)		33.50	106.09	45.27
Estimated yield (t)	Using historical berry weight	34.12	77.99	50.43
	Measured berry weight	35.56	88.99	46.77
Manual yield Estimate (t)		30.04	80.17	43.97
Yield estimation error (%)	Using historical berry weight	1.84	-26.49	11.39
	Measured berry weight	6.16	-16.11	3.31
Manual yield estimate error (%)		-10.34	-24.43	-2.88

30 sample locations the number of bunches per shoot was measured and the bunches were photographed using a smartphone. The average berry weight at harvest was calculated by two different methods, either using historical averages for each block or by direct measurements of berry weight. The combination of shoot counts, bunches per shoot, berries per bunch (from the method in this paper) and harvest berry weight, along with correction factors such as harvester efficiency were used to predict the yield in each block, as shown in Table 5. Manual yield estimation results using industry standard approaches are also provided for comparison, see Whitty et al. [1] for further details.

The data demonstrated in Table 5 presents an encouraging application for the proposed berry counting algorithm. Given automated image-based shoot counts from over 65km of vine rows in three blocks, combined with 253 smartphone images of bunches and average berry weights from a subset of bunches, the yield close to harvest was estimated within 6%, 16% and 3% of the actual yield respectively. This was generally better than industry standard manual yield estimates of the same blocks, despite shoot counts being taken five months prior to harvest and a substantially reduced requirement for manual labour. The result for block 47A was noticeably poorer due to an unexpectedly large drop in shoot number between the shoot and harvest stage.

When the historical berry weights were used, the error increased substantially, as there was a notable variation in average berry weight in this season for the blocks tested. Hence a reliable measure of berry weight is required to improve the yield estimation.

5. Conclusions

This paper has presented a novel and fast algorithm which is able to count berries and estimate the 3D structure of both red and green grapes *in-field* from pea-sized to harvest development stages from a range of bunch architectures. Using only a single image from a smartphone and no calibration or prior information, the accuracy of the method was 89% when directly compared to the number of berries on a bunch. When averaged across 50-80 images, the accuracy was over 99%, showing the limited bias present in this uncalibrated approach.

The algorithm was found to be robust to different bunch architectures qualitatively as well as give consistent results from pea-sized to harvest development stages. The rapid processing time of 0.1 seconds per image is dramatically faster than manual counting and faster than existing approaches in the literature as well as requiring no human interaction once the image has been captured. When implemented on a mobile operating system, the processing time was within one second per image, allowing it to be used in the field.

When the proposed method was used to estimate bunch weights, an accuracy of more than 92% was found on average over several dozen bunches in each dataset. Furthermore, the algorithm was applied to yield estimation and found to have an error of between 3% and 16% when compared directly to measured tonnes at harvest, using automated shoot counting [24] and a berries per shoot method [1]. Hence, the algorithm has applicability in field scenarios and the potential to speed up and improve the accuracy of yield estimates for farmers using smartphones.

Given the location of each image from a smartphone, this could be directly applied to map bunch weight and yield variation across a block. In addition, by combining with existing work on flower counting [25, 26, 27], the possibility of efficient determination of fruit set ratios on a large scale is envisaged.

Some varieties of grape berries elongate noticeably following véraison, and this method could be extended to fitting ellipses to each berry [28] and reconstruction using corresponding ellipsoids. Improvements could be made to the determination of the extended sparsity factor however this has been left to future work and to aid the generalisability of the algorithm. The reconstruction also provides opportunities for estimating more detailed bunch parameters, which are left for future work. Converting this algorithm into an app for farmers would allow rapid and non-destructive estimates of berry counts and bunch weights with limited bias. Further research is currently conducted on the bunch segmentation step with the aim of removing the need for the high-contrast back board. This would make it easier for farmers to adopt the method in the field.

Acknowledgements

The datasets used in this paper were obtained with the support of Wine Australia project DPI1401 Improved Yield Estimation for the Australian Wine Industry. All these datasets are freely available for download and comparison from the Smart Robotic Viticulture group's website: `http://www.robotics.unsw.edu.au/srv/datasets.html`.

References

[1] M. Whitty, S. Liu, S. Cossell, H. Jayakody, M. Woods, J. Tang, S. Singh, *etc.*, Improved yield estimation for the australian wine industry, Tech. rep., Wine Australia (11 2017).

[2] S. Nuske, K. Gupta, S. Narasimhan, S. Singh, Modeling and calibrating visual yield estimates in vineyards, in: Proceedings of Field and Service Robotics, Springer, 2014, pp. 343–356.

[3] F. Schöler, V. Steinhage, Automated 3d reconstruction of grape cluster architecture from sensor data for efficient phenotyping, Computers and Electronics in Agriculture 114 (2015) 163–177. doi:10.1016/j.compag.2015.04.001.

[4] S. Martin, R. Dunstone, G. Dunn, How to forecast wine grape deliveries using grape forecaster excel workbook version 7, GWRDC: Adelaide, Australia (2003) 100.

[5] F. Rist, K. Herzog, J. Mack, R. Richter, V. Steinhage, R. Töpfer, High-precision phenotyping of grape bunch architecture using fast 3d sensor and automation, Sensors 18 (3) (2018) 763. doi:10.3390/s18030763.

[6] S. Liu, S. Marden, M. Whitty, Towards automated yield estimation in viticulture, in: Proceedings of the Australasian Conference on Robotics and Automation, Sydney, Australia, Vol. 24, 2013.

[7] M. P. Diago, J. Tardaguila, N. Aleixos, B. Millan, J. M. Prats-Montalban, S. Cubero, J. Blasco, Assessment of cluster yield components by image analysis, Journal of the Science of Food and Agriculture 95 (6) (2015) 1274–1282. doi:10.1002/jsfa.6819.

[8] E. Ivorra, A. Sánchez, J. Camarasa, M. Diago, J. Tardaguila, Assessment of grape cluster yield components based on 3D descriptors using stereo vision, Food Control 50 (2015) 273–282. doi:10.1016/j.foodcont.2014.09.004.

[9] A. Kicherer, R. Roscher, K. Herzog, S. Šimon, W. Förstner, R. Töpfer, BAT (Berry Analysis Tool): A high-throughput image interpretation tool to acquire the number, diameter, and volume of grapevine berries, Vitis 52 (3) (2013) 129–135.

[10] M. Grossetete, Y. Berthoumieu, J.-P. Da Costa, C. Germain, O. Lavialle, G. Grenier, et al., Early estimation of vineyard yield: site specific counting of berries by using a smartphone., in: Infomation Technology, Automation and Precision Farming. International Conference of Agricultural Engineering-CIGR-AgEng 2012: Agriculture and Engineering for a Healthier Life, Valencia, Spain, 8-12 July 2012, CIGR-EurAgEng, 2012, pp. C–1915.

[11] A. Aquino, M. P. Diago, B. Millán, J. Tardáguila, A new methodology for estimating the grapevine-berry number per cluster using image analysis, Biosystems engineering 156 (2017) 80–95. doi:10.1016/j.biosystemseng.2016.12.011.

[12] A. Aquino, I. Barrio, M.-P. Diago, B. Millan, J. Tardaguila, vitisBerry: An Android-smartphone application to early evaluate the number of grapevine berries by means of image analysis, Computers and Electronics in Agriculture 148 (2018) 19–28. doi:10.1016/j.compag.2018.02.021.

[13] M. Herrero-Huerta, D. González-Aguilera, P. Rodriguez-Gonzalvez, D. Hernández-López, Vineyard yield estimation by automatic 3D bunch modelling in field conditions, Computers and Electronics in Agriculture 110 (2015) 17–26. doi:10.1016/

j.compag.2014.10.003.

[14] S. Liu, M. Whitty, S. Cossell, A lightweight method for grape berry counting based on automated 3d bunch reconstruction from a single image, ACRA, Seattle, WA, USA, 2015.

[15] L. Luo, Y. Tang, X. Zou, C. Wang, P. Zhang, W. Feng, Robust grape cluster detection in a vineyard by combining the adaboost framework and multiple color components, Sensors 16 (12) (2016) 2098. doi:10.3390/s16122098.

[16] L. Luo, Y. Tang, X. Zou, M. Ye, W. Feng, G. Li, Vision-based extraction of spatial information in grape clusters for harvesting robots, Biosystems Engineering 151 (2016) 90–104. doi:10.1016/j.biosystemseng.2016.08.026.

[17] R. Perez-Zavala, M. Torres-Torriti, F. A. Cheein, G. Troni, A pattern recognition strategy for visual grape bunch detection in vineyards, Computers and Electronics in Agriculture 151 (2018) 136–149. doi:10.1016/j.compag.2018.05.019.

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

[19] O. Mirbod, L. Yoder, S. Nuske, Automated measurement of berry size in images, IFAC-PapersOnLine 49 (16) (2016) 79–84. doi:10.1016/j.ifacol.2016.10.015.

[20] K. Dahal, S. Bhattarai, A. Kicherer, D. Oag, K. Walsh, Assessment of hen and chickendisorder for marketable yield estimates of table grape using the berry analysis tool, Vitis: Journal of Grapevine Research 57 (1) (2018) 27–34.

[21] R. C. Gonzalez, R. E. Woods, S. L. Eddins, Digital image processing using MATLAB, Pearson Education India, 2004.

[22] J. H. Friedman, J. L. Bentley, R. A. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Transactions on Mathematical Software (TOMS) 3 (3) (1977) 209–226. doi:10.1145/355744.355745.

[23] P. Dry, B. Coombe, Grapevine growth stages-the modified e-l system, Viticulture 1 (2004) 150–154.

[24] S. Liu, S. Cossell, J. Tang, G. Dunn, M. Whitty, A computer vision system for early stage grape yield estimation based on shoot detection, Computers and Electronics in Agriculture 137 (2017) 88–101. doi:10.1016/j.compag.2017.03.013.

[25] S. Liu, X. Li, H. Wu, B. Xin, J. Tang, P. R. Petrie, M. Whitty, A robust automated

flower estimation system for grape vines, Biosystems Engineering 172 (2018) 110 – 123. doi:10.1016/j.biosystemseng.2018.05.009.

[26] A. Aquino, B. Millan, D. Gaston, M.-P. Diago, J. Tardaguila, vitisFlower: Development and Testing of a Novel Android-Smartphone Application for Assessing the Number of Grapevine Flowers per Inflorescence Using Artificial Vision Techniques, Sensors 15 (9) (2015) 21204–21218. doi:10.3390/s150921204.

[27] J. Grimm, K. Herzog, F. Rist, A. Kicherer, R. Töpfer, V. Steinhage, An adaptable approach to automated visual detection of plant organs with applications in grapevine breeding, Biosystems Engineering 183 (2019) 170–183.

[28] Q. Wang, Y. Tang, Z. Xiao, Grape size detection and online gradation based on machine vision, International Journal of Agricultural and Biological Engineering 10 (1) (2017) 226–233. doi:10.25165/ijabe.v10i1.2517.