Precision Agriculture: Exploration of machine learning approaches for assessing mango crop quantity

by

Anand Koirala

MEngSci (Electrical and Electronic Engineering)

Thesis Submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

DOCTOR OF PHILOSOPHY (SCIENCES, ENGINEERING AND HEALTH) - CD62

School of Health, Medical and Applied Sciences Central Queensland University

18th November 2019

Declaration of Authorship

By submitting this thesis for formal examination at CQUniversity Australia, I declare that it meets all requirements as outlined in the Research Higher Degree Theses Policy and Procedure.

By submitting this thesis for formal examination at CQUniversity Australia, I declare that all of the research and discussion presented in this thesis is original work performed by the author. No content of this thesis has been submitted or considered either in whole or in part, at any tertiary institute or university for a degree or any other category of award. I also declare that any material presented in this thesis performed by another person or institute has been referenced and listed in the reference section.

Copyright Statement

By submitting this thesis for formal examination at CQUniversity Australia, I acknowledge that thesis may be freely copied and distributed for private use and study; however, no part of this thesis or the information contained therein may be included in or referred to in any publication without prior written permission of the author and/or any reference fully acknowledged.

Anand Koirala

18 November 2019

Abstract

A machine vision based system is proposed to replace the current in-orchard manual estimates of mango fruit yield, to inform harvest resourcing and marketing. The state-of-the-art in fruit detection was reviewed, highlighting the recent move from traditional image segmentation methods to convolution neural network (CNN) based deep learning methods. An experimental comparison of several deep learning based object detection frameworks (single shot detectors versus two-staged detectors) and several standard CNN architectures was undertaken for detection of mango panicles and fruit in tree images. The machine vision system used images of individual trees captured during night time from a moving platform mounted with a Global Navigation Satellite System (GNSS) receiver and a LED panel floodlight. YOLO, a single shot object detection framework, was redesigned and named as MangoYOLO. MangoYOLO outperformed existing state-of-the-art deep learning object detection frameworks in terms of fruit detection time and accuracy and was robust in use across different cultivars and cameras. MangoYOLO achieved F1 score of 0.968 and average precision of 0.983 and required just 70 ms per image (2048 × 2048 pixel) and 4417 MB memory. The annotated image dataset was made publicly available. Approaches were trialled to relate the fruit counts from tree images to the actual harvest count at an individual tree level. Machine vision based estimates of fruit load ranged between -11% to +14% of packhouse fruit counts. However, estimation of fruit yield (t/ha) requires estimation of fruit size as well as fruit number. A fruit sizing app for smart phones was developed as an affordable in-field solution. The solution was based on segmentation of the fruit in image using colour features and estimation of the camera to fruit perimeter distance based on use of fruit allometrics. For mango fruit, RMSEs of 5.3 and 3.7 mm were achieved on length and width measurements under controlled lighting, and RMSEs of 5.5 and 4.6 mm were obtained in-field under ambient lighting. Further, estimation of harvest timing can be informed by assessment of the spread of flowering. Deep learning object detection methods were deployed for assessment of the number and development stage of mango panicles, on tree. Methods to deal with different orientations of flower panicles in tree images were implemented. An R^2 >0.8 was achieved between machine vision count of panicles on images and in-field human count per tree. Similarly, mean average precision of 69.1% was achieved for classification of panicle stages. These machine vision systems form a foundation for estimation of crop load and harvest timing, and for automated harvesting.

Acknowledgements

I would like to thank my principal supervisor Professor Kerry B. Walsh for his continual support and practical advices which has always been a source of motivation for me to develop machine vision solutions for precision agriculture. I would also thank my associate supervisors, Dr Cheryl McCarthy, Dr William Guo, and Dr Zhenglin Wang for their supervision and guidance throughout my candidature.

I would like to dedicate this thesis to my loving parents (Mr. Chiranjibi Raj Koirala and Mrs. Uma Devi Koirala) without whom I wouldn't have existed, and this piece of work would have never come up. I would like to remember my family members (Santosh Koirala, Ashmita Pathak and Mitasha Koirala) and my loving wife (Binita Aryal) for their love, care and support. I would like to thank Dr Phul Subedi and Mr. Nicholas Anderson for their assistance during data collection and analysis. Thanks goes to Dr Surya Bhattarai and Bhima Bhattarai for their love and support.

This RHD candidature was supported under the Commonwealth Government's Research Training Program/Research Training Scheme. I gratefully acknowledge the financial support provided by the Australian Government. I am grateful to be awarded Regional University Network (RUN) PhD Scholarship provided by Central Queensland University. I gratefully acknowledge the funding received from the Federal Department of Agriculture and Water and Horticulture Innovation Australia through the project Multiscale monitoring of tropical fruit production (ST15005) which has supported this research.

Finally, I would like to thank all the industry and intuitional partners supporting this project and especially the farm owners who provide access to their orchards for data collection.

Table of Contents

Declarati	ion of Authorship	ii
Abstract		iii
Acknowl	edgements	iv
List of Fig	gures	vii
List of Ta	bles	x
List of Ak	bbreviations	xii
Chapter	1. Introduction	1
1.1	Introduction to the mango industry	1
1.2	Motivation for application of machine vision to the mango crop	3
1.3	Thesis Objectives	3
1.4	Publications associated with this thesis	4
1.5	Thesis Structure	6
Chapter estimatio		nd yield
2.1	Introduction	9
2.2	CNN and deep learning – a background	11
2.3	Deep learning object detection framework	16
2.4	Network and model training	20
2.5	Performance assessment	24
2.6	Architecture and model optimization	27
2.7	Fruit detection using deep learning	31
2.8	Tree fruit yield estimation	33
2.9	Conclusion and recommendations	35
Chapter benchma	3. Deep learning for real-time fruit detection and orchard fruit load estimation arking of ' <i>MangoYOLO</i> '	
3.1	Introduction	
3.2	Materials and Methods	44
3.3	Results	51
3.4	Discussion	58
3.5	Conclusion	61
Chapter	4. Deep learning for mango panicle stage classification	63
4.1	Introduction	64
4.2	Materials and methods	66
4.3	Results	70
4.4	Discussion	76

4.5	Conclusion	83
Chapter ! machine	 Estimating the unseen – correction for occluded fruit in tree fruit load es vision 	
5.1	Introduction	
5.2	Materials and Methods	
5.3	Results and Discussion	97
5.4	Conclusions	
Chapter	6. In Field Fruit Sizing Using A Smart Phone Application	
6.1	Introduction	
6.2	Materials and Methods	
6.3	Results and Discussion	
6.4	Conclusions	
Chapter	7. Conclusion	115
7.1	Summary	115
7.2	Future directions	117
List of Re	ferences	
Appendix	<	
Other	Published Research	
Appen	dix A. Published version of Chapter 3	
Appen	dix B. Published version of Chapter 4	
Appen	dix C. Published version of Chapter 6	
Appen	dix D. Method comparison	
Appen	dix E. Automated Mango Flowering Assessment via Refinement Segmentation	
Addendu	ım	

List of Figures

panel)
Figure 3-1. Lighting and imaging camera rig mounted on a farm utility vehicle, operated at 6 km/h (left panel), with components of LED floodlights, RGB camera and Time of Flight camera (right
some FN; NMS 0.3, one detection for each fruit with no FN; NMS 0.5, but not all detections merged.
Figure 2-16. Effect of NMS setting: left to right panels: NMS = 0.1, one detection for each fruit but
fruit (lower FP), but failure to detect some fruit (higher FN)
threshold values (0.1, 0.8, 0.95, for left to right panels) resulting in fewer multiple detections per
Figure 2-15. Object detection with no suppression (NMS=1.0) and an increasing level of confidence
(right panel)29
an image of mango fruit on tree, produced by Faster-RCNN using ZFNet (left panel) and VGGNet
Figure 2-14. An example display of class label 'm' (mango fruit) and associated confidence scores for
in whole tree images, obtained for varying IoU thresholds (0.1-0.7) at a NMS threshold of 0.5 26
Figure 2-13. Precision-Recall curves for a Faster R-CNN with ZFNet model used to detect mango fruit
boxes25
Recall is the proportion of detected boxes that matched the ground truth boxes on all ground truth
annotation. <i>Precision</i> is the proportion of detected boxes that matched the ground truth boxes.
Figure 2-12. Interpretation of <i>Precision</i> and <i>Recall</i> for object detection task using bounding box
Figure 2-11. Definition of IoU in the object detection task using bounding box annotations
with varying levels of occlusions by other fruits or leaves21
Figure 2-10. Examples of ground truth labelling of individual fruit using LabelImg software in scenes
mango) and probability score20
'concatenation layer'. Output image displays bounding box on ROI, classification result ('m' for
Figure 2-9. Block diagram of YOLOv3 architecture. 'Conv' refers to 'convolution layer' and'concat' to
on ROI, classification result ('m' for mango) and probability score
'concatenation layer' and 'reorg' refers to 'reorganize/route'. Output image displays bounding box
Figure 2-8. YOLOv2 object detection framework. 'Conv' refers to 'convolution layer', 'concat' to
displays bounding box on ROI, classification result ('m' for mango) and probability score19
Figure 2-7. SSD object detection framework. 'Conv' refers to 'convolution layer'. Output image
probability of correct classification18
shows bounding boxes (Bboxes) on detected objects of class 'm' (mango), with associated
Figure 2-6. The Faster R-CNN object detection framework (RPN plus Fast R-CNN). Output image
Figure 2-5. The fast R-CNN object detection framework17
heat-map (left) and a heat map superimposed on input image (right)16
learning 'Xception' (Chollet 2017) model (trained to directly predict fruit count from input images) as
Figure 2-4. Grad-CAM visualization of activation map of the final convolutional layer of a deep
after training is provided in the top right15
left of the graphic, and visualization of the output of example convolution filters (feature extraction)
subsample data from the convolution layers. Example input training images are provided in the top
through five layers. Multiple feature maps are created in each convolution layer. Pooling layers
Figure 2-3. The LeNet (LeCun et al. 1998) CNN classifier accepts 28x28 pixel images and processes
by a cascade classifier for mango fruit detection14
Figure 2-2. Visualization of some Haar features (left panel) and some LBP features (right panel) used
Figure 2-1. HOG features, using different block and cell sizes, overlaid on the fruit image
Figure 1-1. 2019/2020 mango crop forecast - dispatch to the markets

Figure 3-3. Block diagram of architecture YOLOv3	48
Figure 3-4. Block diagram of architecture of <i>MangoYOLO</i>	
Figure 3-5. Average Precision for MangoYOLO(s)-512 on validation set 1 (Table 2) plotted against the	
number of training images from Training set 1. Error bar represents standard deviation on three	
repeated assessments	52
Figure 3-6. Example of fruit detection on images of same tree (Test set 2A) for different cameras	
(Basler, Canon and Kinect), using a <i>MangoYOLO(s)</i> model trained on Train set 1	55
Figure 3-7. Example of fruit detection on images of cultivars HoneyGold and R2E2, using a	
MangoYOLO(s)-512 model trained on cultivar Calypso images (orchard A)	57
Figure 4-1. Left to right: original image, upright bounding box and rotated bounding box	66
Figure 4-2. Training examples for stages X to Z, by rows	67
Figure 4-3. Display of ground truth bounding box original (rotated) red colour for R ² CNN training ar	۱d
transformed (upright) blue colour for MangoYOLO training	68
Figure 4-4. Pixel segmentation (left panels) and deep learning R ² CNN (right panels) results for the	
same images. Flowers in the dark background did not segment properly (top left). Branches and	
leaves were erroneously segmented as flower pixels (bottom left).	71
Figure 4-5. Example images processed with three methods. Top panel: Panicle stage	
detection using YOLO method. Orange, green and blue coloured boxes represent panicle stages X,	Y
and Z respectively. Middle panel and bottom panel: Panicle stage detection using R ² CNN and	
R ² CNN-upright methods, respectively. Green, pink and red coloured boxes represent panicle classe	
of X, Y and Z, respectively.	73
Figure 4-6. Example images of panicle stage detection by YOLO and R ² CNN methods on Canon	
images of Wang et al. (2018b). Top panel: Panicle stage detection using YOLO method. Orange,	
green and blue coloured boxes represent panicle stages X, Y and Z respectively. Middle panel and	
bottom panel: Panicle stage detection using R ² CNN and R2CNN-upright methods, respectively.	
Green, pink and red coloured boxes represent panicle classes of X, Y and Z, respectively.	
Figure 4-7. Time course (weeks 1, 3, 5 and 7) of panicle number by developmental stage per tree for	
a row of trees	
Figure 4-8. Flowering intensity level (green, orange and red colour corresponds to low (<30 panicle	
or 10% pixels), medium (30 to 70 panicles or 10 to 25% pixels) and high (>70 panicles or 25% pixels	
(top panel) or panicle count (using R ² CNN display) (bottom panel) of an orchard. In this software, a	n
individual tree can be selected to display the flowering intensity level, tree image, image capture	70
date and tree id	
Figure 4-9. Flower stages trend analysis across weeks for an orchard	
Figure 4-10. Peak flowering event detection on stage-X panicle counts for two different trees acros	
9 weeks of imaging. Single peak (left) and double peak (right) marked with a coloured dot Figure 4-11. Plot displaying week in which a peak flowering event was noted (top panel) and plot	80
displaying the week of the largest flowering event (bottom panel) for 168 trees in which two	
flowering peaks were recorded.	Q1
Figure 4-12. Flower stages X to Z (top to bottom panel) selected for one imaging run of week-1 and	
corresponding flowering intensity level (yellow, orange and red colour corresponds to low (<30	1
panicles), medium (30 to 70 panicles) and high (>70 panicles) display of an orchard	82
Figure 6-1. Application scenario (left) and main user interface (right), illustrating fruit real	02
dimensions of length (L), width (W), and thickness (T).	06
Figure 6-2. Image processing: image acquisition (left); circle identification (middle); and fruit	
segmentation (right)	06
Figure 6-3. Panels from left to right display (a) RGB image of fruit against a blue board and canopy	
background, in conditions of partial direct sunlight; (b) gray scale in the b* channel for the same	

image, (c) the segmented image; and (d) a histogram of b* channel values from the image, with t Otsu's method optimum threshold for separation of background from Region of Interest pixels	he
indicated by the grey arrow	. 107
Figure 6-4. RMSE of camera to object distance estimation (for 10 replicate measurements) as	
influenced by reference circle size	110
Figure 6-5. Absolute residual of camera to object distance estimation as influenced by angle	
between camera (phone major axis) and object planes, for tilt forward (down) and back (up).	
Camera lens held at a set distance to object plane, while phone body was tilted	111
Figure 6-6. Correlation between mobile application and caliper measurement of fruit length (left	
panel) and width (right panel)	.112
Figure 6-7. Fruit length and width estimated using the mobile application on two phone types (H	ГC,
Samsung), relative to caliper measurement, for a set of mandarin, orange and apple fruit (n = 21	
fruit)	.113
Figure 6-8. Time course of average mango fruit lineal dimensions (n = 17 fruit) for length (top line	2
pair) and width (bottom line pair), as assessed using calipers (dashed line) or machine vision	
(FruitSize application; solid line). Error bars represent the standard error of the mean	.113

List of Tables

Table 2-1. Scientific reports on use of deep learning models in estimation of tree fruit number per image. The best result of each paper is shown. When available, the F1 score is recorded, otherwise the validation metric used by the authors is included	/ 4
variants5 Table 3-4. Model performance (F1-score and AP) for fruit count on Validation set 1 (512×512) images and average detection/inference times for different models. Values assessed on HPC hardware and using Darknet repository (https://github.com/AlexeyAB/darknet, accessed on 15/03/2018) in the YOLO variants	
Table 3-5. Detection speed and memory usage for three models used with full canopy images (2048×2048). Values assessed on HPC hardware and using Darknet repository (https://github.com/AlexeyAB/darknet: accessed on 21/11/2018) in the YOLO variants. Within the GPU memory constraint the maximum network resolutions for YOLOv3 were 1888×1888 and 1184×1184 for HPC and Nuvo computing platforms respectively (note: YOLO requires the network input resolution to be multiple of 32)	
Table 3-6. Model performance (R ² , Root Mean Square Error (RMSE) of prediction and Bias) for fruit counts using the overall test set (Test set 1 All) and subsets of low, medium and high frequency of fruit occlusion (n=100 tiles in each set). Units for RMSE and bias are fruit number per tile. Best result within a column is indicated in bold. Input resolution was 416×416 pixels for YOLO original variants. Images were rescaled for the shorter side to be 600 pixels for Faster R-CNN original variants. 5 Table 3-7. Regression statistics for fruit detection by <i>MangoYOLO(s)</i> and <i>MangoYOLO(pt)</i> on sets Test set 2 A, Test set 2 A-can and Test set 2 A-kin captured using Basler, Canon and Kinect cameras	4
respectively	
Table 3-9. Regression statistics of machine vision count against human count of fruit on images of sample trees (Table 3-1) for five orchards. Best result within a row is indicated in bold. Network resolution for Faster R-CNN and <i>MangoYOLO</i> models was set to 2048×2048 but for YOLOv3 it was 1888×1888 (maximum possible within available GPU memory of 16 Gb)	
Table 3-10. Pack-house fruit count and number of fruits detected for each orchard using dual view imaging six weeks before harvest, for each of five orchards. Best result within a row is indicated in bold. Netwo rk resolution was set to 2048×2048 for Faster R-CNN and <i>MangoYOLO</i> models, and to 1888×1888 (maximum possible within available GPU memory of 16 Gb) for YOLOv3	6
Table 4-2. Number of panicles in training and validation data sets	ð

Table 4-3. Correlation R ² between flowering intensity level per tree from pixel segmentation method
and Y stage or all stages panicle counts, respectively, from the R ² CNN method71
Table 4-4. Panicle stage detection results on the validation set using three methods. RMSE refers to a
comparison with ground truth assessments of panicles per image. All values refer to # panicles/tree
image. Lowest RMSE values are shown in bold72
Table 4-5. Prediction statistics for the validation set using YOLO and R ² CNN methods. Highest results
are shown in bold72
Table 4-6. Comparison of several flower assessment methods on the test image set (as used
by Wang et al. (2018b) in terms of the R ² and RMSE between machine vision panicle (sum of two
sides of a tree) count on images (two per tree) versus in-field human counts of panicles per tree74
Table 5-11. Correlation between hidden and exposed fruits for 2 seasons obtained from MLP_yield
model. Units are fruit number per tree
Table 6-1. Allometric relationships based on linear regression with intercept of zero for fruit real

List of Abbreviations

ANN	Artificial Neural Network
CNN	Convolutional Neural Network
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
LED	Light Emitting Diode
loU	Intersection over Union
mAP	mean Average Precision
MLP	Multi Layered Perceptron
NMS	Non-Maximum Suppression
NN	Neural Network
ReLU	Rectified Linear Unit
RGB	Red, Green and Blue
RMSE	Root Mean Square Error
Rol	Region of Interest
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
YOLO	You Only Look Once

This page is intentionally left blank.

1 Chapter 1. Introduction

2 1.1 Introduction to the mango industry

- 3 Mangoes are popular summer fruit in Australia. Northern Territory, northern and central
- 4 Queensland and Western Australia are the major mango production areas in Australia, with
- 5 Queensland and NT accounting for 95% of the crop (Margetts 2014). The seasonal harvest starts
- 6 from September to March for different regions. Approximately 92% of the Australian mango
- 7 produce is consumed domestically, while about 8% (4,500-5,000 tonnes) is exported annually to
- 8 other countries (Margetts 2014).
- 9 Many orchards are in remote areas far from markets and sources of labour supply. Further, because
- 10 of the seasonal variations and irregular fruit bearing properties of the crop, mango fruit timing and
- 11 yield is highly variable between seasons. This creates problems in organising labour and transport.
- 12 Predictions of harvest timing and volume can guide agronomic treatments, labour resource
- 13 management, and support market planning.
- 14 On farm yield estimation is typically performed based on weather data, previous yield history, and
- 15 manual estimation of flowering and fruit numbers on the trees. The timing of flowering events is
- 16 recorded to inform estimates of harvest timing, and of the volume spread fruit numbers through the
- 17 harvest period. The fruit count is made soon after 'stone-hardening' stage, some six weeks before
- 18 harvest. At this time fruit drop effectively halts and the fruit will persist to harvest.
- 19 The Australian Mango Industry Association attempts to collate industry wide harvest data on a
- 20 weekly basis, to inform member marketing decisions. The latest 2019/20 mango crop forecast from
- 21 the Australian mango industry is available at <u>https://www.industry.mangoes.net.au/resource-</u>
- 22 <u>collection/2019/7/30-crop-forecast</u> (also Fig. 1-1). Information is presented for all major mango
- 23 production regions, with data on actual weekly volume of trays dispatched from each region and
- 24 varieties presented to the current week, and forecast presented for future weeks.

TOTAL RETIMATED FOR ALL RELOW REGIONS AND VARIETIES (R.1 million trave)	105 (\$1 m	New Long																									
2 2 7 X X X	「日本	-2		88	-8		-8		-\$	-}	-		**		#¥	÷ž	-4	靖	= 1	28	-1			22 22	-1	-1	27
				-					1		- -				I	5 . 1	20	-	8 .							-	- 1
								1		1 +	-			1 7	1						.,						•
		3	3	-	*		2	-	5	-	-			8	8	1	1	1	8		:			÷		-	-
DARWIN (2.8 million trays)																											
1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 -	4	-5	=	-		-	-1	4	-	•1	-	-		=	2	=}	-1	1	-	-		-	1		-	-	8
0.40 0.10 0.74 0.45	5			5	1 41	- 2	13		-																		
an eas are an	-								-			5															
							an an	11.54	N.																		
		3	8	3	4 12			-	1	=	1		-														
and teact																											
with the rest of the rest of the rest of the rest	1.41		-		-			=	-	•		-	-	=	2	-	-	-		2	4		1	1			B
				-0			1			-																	
					-			1				1															
									Ę					-	4110	1010											
						•		2	*							-								F			
A THE BRACK (1.8 molitum trend)																											
「「「」」」「「」」」」			-	2		-	=	-	-1	-	-	-	1	=}	23	2	-	2	•	23	-	-	2	-	1	2	*
						- 21	12	1.5	-		12			-										_			
								-	Martin.	and a	-	- 141	In All														
									-	-	and in	1. mm	sin														
									-	5	-	-	MAN NAM	M Let													
						-		1		5	1	-	-	=	8												
DOWEX/BUBDEXIN (1.6 million trays)																											
二日 二		-1	=	23	- 8	001 001		=3	-}	-1		-		-3	21	28	-1	0	-	28	-1	-	2		-	=1	2
									\$					-	-												
							-	-	110	1114	-	-	STATE AND	NI III	-	1417	-										
													1	-	and a	1	5	E.									
							•		Ŧ	-	÷.		-	-	2	#	Ŧ	R									
HARCEBA, DENBUCAH (LB million trans																											
二十二十二十二十二十二十二十二十二十二十二十二十二十二十二十二十二十二十二十		-1	-	2	-	and and	- 3	-	-}	-1				=}	23	=	-	1		2	-	-1	2			-	4
											10	- 0	10			- 14	N.M.	5	-							-	_
											1	5	-	A 2.5	in a	N H IN	-	NAM	-		1						
															-	-	-	1	1	-							
																N.	4.1	1	-	-	-	5					
																1		5	1444	K2	4	111		-		-	1
										i						5	1	5	5	1	5		New In	-			1

26 Figure 1-1. 2019/2020 mango crop forecast - dispatch to the markets

1.2 Motivation for application of machine vision to the mango crop

29 The current manual estimates of flowering level and fruit load are recognised to be error prone, and 30 highly operator dependent (pers. comm. Ian Groves, mango producer). A machine vision based 31 system that offers better accuracy and precision in an easy to operate format should have good 32 adoption 'pull' for this application. Australian mango growers in partnership with Department of 33 Agriculture and Fisheries (DAF) have semi commercial trials underway on high density planting 34 in parts of Queensland and Northern Territory. These high-density small tree plantation systems 35 have promise to significantly improve productivity and profitability of the mango industry and suit 36 the adoption of in-field machine vision and the implementation of automation in harvesting. These 37 trends provides motivation to research the use of machine vision to in-field applications in the 38 mango orchard.

39 1.3 Thesis Objectives

The common challenges for machine vision to detect and recognize objects in a real-scene arevarying illumination, occlusions, object-background colour overlap and changing object orientation. To overcome these problems a well-generalised model that is illuminance and position invariant is sought. The proposed research is inspired by the discriminative and low-level feature extraction capabilities of deep neural networks and its success in many machine vision challenges, as seen in recent years.

- 46 This thesis reviews several machine learning and deep learning methods of object detection for
- 47 application to mango flower and fruit detection, to support estimation of crop timing and volume.
- 48 The primary application area is an assessment of mango flowering stages and green fruit (at the
- 49 stone-hardening stage prior to harvesting) number and size. The overall aim of this thesis is to
- 50 provide management tools to the grower to assist in timing and resourcing of harvest and reduction
- of labour, contributing to the development of a decision support system based on machine vision
- 52 technologies that is effective in terms of cost and complexity.
- 53 The thesis is focussed to the development of machine vision algorithms specific to the application of 54 interest while reducing hardware complexities as much as possible. For example, some researchers 55 have used high intensity Xenon strobe lights to allow day-time imaging. This requires additional cost
- 56 in hardware, and the microsecond exposure time requires more sophisticated interfacing between
- 57 strobe and camera triggering. Night time imaging with LED flood lighting allows for lower cost
- hardware, with the added benefit that the machine vision model can be less complex, i.e.,
- 59 background noise is greatly reduced with night images There is also potential to couple a night
- 60 imaging task to other farm tasks that can occur at night, such as spraying.
- 61 In summary, as set at the start of activity in 2016, the objectives of this thesis were to:
- 62 Define applications for machine vision in mango culture for precise estimation of crop yield
- 63 Contribute to the design of a low-cost imaging system for use on mango orchards
- Establish a database of images with different mango varieties and associated actual
 flowering/fruit count collected at different times of season and from different orchards.
- 66 Compare and benchmark the state-of-the-art machine learning methods for estimation
 67 mango flower and fruit detection and yield estimation
- 68 Contribute to the building of a decision support tool using output of the machine vision
 69 analysis

70 **1.4 Publications associated with this thesis**

71 Journal articles

72 The following papers were published in refereed journals.

73 Appearing as chapter 2 -

Koirala A, Walsh KB, Wang Z, McCarthy C (2019a) Deep learning – Method overview and review
 of use for fruit detection and yield estimation. *Computers and Electronics in Agriculture* 162:219 234 doi:https://doi.org/10.1016/j.compag.2019.04.017

I was the main author for this journal paper (Koirala et al. 2019a). The manuscript was written
by me with guidance from my supervisors, Zhenglin Wang, Kerry Walsh, and Cheryl McCarthy.
Kerry Walsh and Zhenglin Wang reviewed and edited the manuscript. This paper forms
Chapter 2 of this thesis.

81 Appearing as chapter 3 -

Koirala A, Wang Z, Walsh K, McCarthy C (2019b) Deep learning for real-time fruit detection and
 orchard fruit load estimation: benchmarking of *'MangoYOLO'*. *Precision Agriculture* 20:1107-1135
 doi:<u>https://doi.org/10.1007/s11119-019-09642-0</u>

85 I was the main author for this journal paper (Koirala et al. 2019b), driving the image data
86 acquisition, experimental design, framework development, data analysis, manuscript writing
87 and editing. I received guidance from my supervisors, Zhenglin Wang, Kerry Walsh, and Cheryl
88 McCarthy. Kerry Walsh and Zhenglin Wang reviewed and edited the manuscript. This paper
89 forms Chapter 3 of this thesis.

90 Appearing as Appendix A -

Wang Z, Walsh K, Koirala A (2019) Mango Fruit Load Estimation Using a Video Based
 MangoYOLO—Kalman Filter—Hungarian Algorithm Method. Sensors 19:2742
 doi:https://doi.org/10.3390/s19122742

- 94I was the secondary author of this journal paper (Wang et al. 2019), developing the95MangoYOLO fruit detection algorithm, model training and testing, conducting preliminary96tracking experiments, data acquisition, analysis and manuscript editing. The majority of the97framework implementation and experimental analysis and software development was98performed by the primary author, Zhenglin Wang. Kerry Walsh who also wrote the
- 99 manuscript and provided supervision. This paper forms Appendix A of this thesis.

100 Appearing as Appendix B -

- Koirala, A.; Walsh, K.B.; Wang, Z.; Anderson, N. Deep Learning for Mango (*Mangifera indica*)
 Panicle Stage Classification. *Agronomy* 2020, *10*, 143. Doi:
 https://doi.org/10.2200/organomy10010142
- 103 <u>https://doi.org/10.3390/agronomy10010143</u>
- 104I was the main author for this journal paper (Koirala et al. 2020), driving the image data105acquisition, experimental design, framework development, data analysis, manuscript writing106and editing. I received guidance from my supervisors, Zhenglin Wang, Kerry Walsh, and Cheryl107McCarthy. Nicholas Anderson helped in image data acquisition and annotation. Kerry Walsh,108Nicholas Anderson and Zhenglin Wang reviewed and edited the manuscript. This paper forms
- 109 Appendix B of this thesis.

110 Appearing as chapter 6 -

Wang Z, Koirala A, Walsh K, Anderson N, Verma B (2018) In Field Fruit Sizing Using A Smart Phone
 Application. Sensors 18:3331 doi:<u>https://doi.org/10.3390/s18103331</u>

I was the secondary author of this journal paper (Wang et al. 2018a), driving the image data 113 114 acquisition, part of method development and implementation, experimental design, data analysis and manuscript editing and reviewing. Most of my work was on image processing 115 116 for fruit segmentation in images- trailing several color spaces, thresholding methods and morphological operations. Zhenglin Wang was mostly involved in mobile app software 117 programming and developing method to relate object size in images to the actual fruit size. 118 Kerry Walsh provided supervision and guidance during all phases of the experiments- from 119 120 conceptualization to final edits of the paper. Zhenglin Wang and Kerry Walsh wrote the 121 manuscript, proof-read and edit. Nicholas Anderson was involved in recording the fruit 122 measurements in-field and in lab and establishing the lineal relationships for several mango 123 cultivars. Nicholas was of assistance in data analysis and reviewing draft papers. Brijesh

124 Verma reviewed and edited the manuscript. This paper forms Chapter 6 of this thesis.

125 **Conference publications**

126 The following publication was associated with a conference:

127 Appearing as appendix C -

Koirala A, Walsh K, Wang Z, McCarthy C (2017) Mobile device machine vision estimation of mano
 crop load. In: International Tri-Conference for Precision Agriculture, New Zealand, 2017.
 doi:https://doi.org/10.5281/zenodo.895382

131I was the primary author for this conference paper (Koirala et al. 2017), driving the image132data acquisition, experimental design, framework development, data analysis, manuscript133writing and presentation of the paper at the conference. The mobile software was written by134Zhenglin Wang. I received guidance from my supervisors, Zhenglin Wang, Kerry Walsh, and135Cheryl McCarthy. Kerry Walsh and Zhenglin reviewed and edited the manuscript. This paper136forms Appendix C of this thesis.

137 Appearing as appendix D -

Underwood JP, Rahman MM, Robson A, Walsh KB, Koirala A, Wang Z (2018) Fruit load estimation
in mango orchards - a method comparison. Paper presented at the ICRA 2018 Workshop on
Robotic Vision and Action in Agriculture, Brisbane, Australia.

- 141I was involved in the data acquisition, method implementation, and data analysis and poster142presentation at the workshop for the MangoYOLO method included in this paper143(Underwood et al. 2018). Andrew Robson and Moshiur Rahman contributed to the satellite144imagery method in the paper. James Underwood contributed to the Faster-RCNN based145method in the paper. Kerry Walsh, Zhenglin Wang, and James Underwood prepared the146manuscript and all authors contributed to editing. This paper forms Appendix D of this
- 147 thesis.

148 Appearing as appendix E -

Wang Z, Verma B, Walsh KB, Subedi P, Koirala A (2016) Automated mango flowering assessment
 via refinement segmentation. In: International Conference on Image and Vision Computing New
 Zealand (IVCNZ) 2016 IEEE, pp 1-6. doi: <u>https://doi.org/10.1109/IVCNZ.2016.7804426</u>

152I was the secondary author of this publication (Wang et al. 2016) conducting image data153acquisition, part of data analysis and manuscript editing. Most of the work on image154processing framework development experimental analysis and manuscript writing was155performed by the primary author Zhenglin Wang. Kerry Walsh, and Brijesh Verma, helped in156manuscript preparation providing appropriate supervision and guidance. Phul Subedi,157provided field arrangements for image data acquisition and helped in manuscript reviewing.158This paper forms Appendix E of this thesis.

159

160 **1.5 Thesis Structure**

161 The thesis is presented as seven chapters and four appendices. The first chapter is a general

162 introduction to the topic of the thesis, while the second is a detailed review of the use of deep

learning in object recognition (published as Koirala et al. 2019a). The next four chapters present

164 experimental work. The third chapter presents the development of 'MangoYOLO' for fruit detection

165 (published as Koirala et al. 2019b), the fourth, the use of deep learning frameworks in panicle

detection, the fifth a consideration of estimation of the proportion of totally occluded fruit from a

tree image, and the sixth presents on the development of a machine vision based system for in-field

168 fruit sizing. The final chapter presents a summary and future directions for the use of machine vision 169 in estimation of timing and volume of the mango crop. The following text expands on this

in estimation of timing and volume of the mango crop. The fordescription.

171 Chapter 1 provides the overview on the contents of the thesis and details on the background and172 motivation for the research. The aims and objectives of the thesis are outlined.

173 **Chapter 2** presents a review of developments in the rapidly developing field of machine learning

174 with emphasis placed on practical aspects of deep learning for the task of fruit detection and

localization, in support of tree crop load estimation. This chapter provides the background

176 on various standard convolutional neural network (CNN) classifiers and several one-stage detection

and two-stage detection frameworks. Several examples and figures are presented in support of the

178 use of CNN for fruit detection and localization tasks and comparison is made with traditional

179 methods of image processing and object segmentation.

180 Chapter 3 forms the core of this thesis. This chapter compares the performance of several single and
 181 double staged object detection frameworks and various CNN architectures on a common image

182 dataset created during this research for the task of fruit detection. This technology can be applied in

183 real time for orchard fruit detection and automated harvesting.

184 Chapter 4 extends the deep learning methods of object detection to the application of flower
 185 assessment. Two different deep learning object detection frameworks are compared on a common
 186 assessment and a set of the set of the

186 dataset produced during this research for the task of mango panicle detection and panicle

187 development stages classification. Methods to deal with different orientations of flower panicles in

188 tree images are implemented. Some example applications of this system are also presented in the

189 chapter. This technology can provide information on the number of panicles and spread of flowering

190 which forms a foundation for estimation of crop load and harvest timing, and for automated

191 harvesting.

Introduction

- 192 **Chapter 5** deals with the issue of hidden/occluded fruits (fruits that are not captured in images of
- 193 trees because of occlusions from the camera view point). This chapter explores several machine
- 194 learning algorithms and deep learning CNNs, separately and in combination, to automatically
- accommodate the occlusion factor within the model itself. Image processing methods to segment
- canopies and fruits in images and shape fitting techniques to extract partially occluded fruits are alsoconsidered.
- 198 **Chapter 6** reports on the development of an on-tree mango fruit sizing mobile phone app as a
- 199 component of the yield estimation framework. A simple method of image processing object
- segmentation and morphological operation is presented for fruit segmentation. Similarly,
- 201 fruit allometrics and thin lens formula is applied for accurate measurement of fruit lineal dimension
- and relating to the fruit weight. This technology provides an affordable solution that allows
- estimation of fruit size distribution which can provide information on fruit maturity and yieldestimates.
- 205 **Chapter 7** concludes this thesis by summarizing on the achievements and contributions from this
- 206 work. Key limitations on practical implementation of current methods and technologies are
- 207 identified, future possibilities on adoption of related technologies are outlined, and suggestions for
- 208 future research are made.
- 209

210

Chapter 2. Deep learning – method overview and review of use for fruit detection and yield estimation

213	
214 215 216	This chapter was published as a journal paper on April 2019 in Computer and Electronics in Agriculture as:
210 217 218 219 220	Koirala A, Walsh KB, Wang Z, McCarthy C (2019a) Deep learning – Method overview and review of use for fruit detection and yield estimation Computers and Electronics in Agriculture 162:219-234 doi: <u>https://doi.org/10.1016/j.compag.2019.04.017</u>
221 222	Responses to minor revisions as requested by the thesis examiners can be found in the Errata section.
223	
224	
225	
226	
227	
228	
229	
230	
231	
232	
233	
234	
235	
236	
237	
238	
239	
240	
241	
242	

Abstract 243

244 A review of developments in the rapidly developing field of deep learning is presented.

245 Recommendations are made for original contributions to the literature, as opposed to formulaic

246 applications of established methods to new application areas (e.g., to new crops), including the use

247 of standard metrics (e.g., F1 score, the harmonic mean between Precision and Recall) for model

248 comparison involving binary classification. A recommendation for the provision and use of publicly

249 available fruit-in-orchard image sets is made, to allow method comparisons and for implementation 250 of transfer learning for deep learning models trained on the large public generic datasets. Emphasis

251 is placed on practical aspects for application of deep learning models for the task of fruit detection

252 and localisation, in support of tree crop load estimation. Approaches to the extrapolation of tree

253 image counts to orchard yield estimation are also reviewed, dealing with the issue of occluded fruit

254 in imaging. The review is intended to assist new users of deep learning image processing techniques,

255 and to influence the direction of the coming body of application work on fruit detection.

256 2.1 Introduction

257 In any given discipline there are periods of rapid advance and periods of incremental progress.

258 Machine vision and machine learning is in a period of rapid advance. Typically, step advances are

259 catalysed by some combination of expertise, resources and application need, and then diffuse into

260 other application areas. Advances can originate in the agricultural area, e.g., the discipline of near 261 infrared spectroscopy was born from the need to assess forage and grain quality (Norris 1996), but

262 more often advances occur in better resourced sectors (e.g., medical and security) and diffuse into

263 the agricultural sector. The latter is true of deep learning (multiple layer neural networks) in

264 machine vision, which represents a step advance from algorithms based on hand crafted features 265 such as colour, shape and texture.

266 Application of deep learning techniques to agricultural applications is nascent, with the 2018 review 267

of Kamilaris and Prenafeta-Boldú (2018) reporting just 40 published studies (taking a broad

268 definition of deep learning), with four reports on the topic of in-field fruit counting. Subsequent

269 progress has been rapid, both in the field of deep learning, and in application to the task of fruit 270 counting. In the current review we seek to extend the review of Kamilaris and Prenafeta-Boldú

271 (2018) in context of developments in deep learning, broadening the topic focus to provide

272 background on the techniques, and narrowing the application topic to that of fruit detection and

273 localisation.

274 Previous reviews of the task of in-field fruit detection, e.g., Gongal et al. (2015), Kamilaris and 275 Prenafeta-Boldú (2018), Naik and Patel (2017) and Syal et al. (2013), have reinforced the choice of 276 the RGB camera as the detector of choice (based on the practicality of cost and ease of 277 implementation) and discussed use of handcrafted features such as colour, texture and shape in fruit 278 detection. However, these techniques require re-design when used outside of a set of conditions 279 particular to the calibration conditions, e.g., variation in fruit or foliage colour, illumination, camera 280 viewing angle and camera to fruit distance. Gongal et al. (2015) noted that supervised methods 281 based on machine learning yield better results than simple image processing techniques but require 282 greater computational resources and a greater resource of labelled data for training was required. In 283 recent years, however, high performance GPU has become available, and the task of labelling object 284 in images has become easier with the advent of freely available graphical annotation tools (e.g., 285 LabelImg https://github.com/tzutalin/labelImg). Moreover, many state-of-art deep learning 286 frameworks such as Faster Regional Convolutional Neural Network (Faster R-CNN) (Ren et al. 2015), 287 Single Shot multibox Detector (SSD) (Liu et al. 2016), and You Only Look Once (YOLO) (Redmon and

288 Farhadi 2018) have scripts to parse the commonly used PASCAL-VOC (Everingham et al. 2010)

annotation format for training the network. Thus, constraints to practical adoption of the deep
learning methods have recently fallen away.

291 Naik and Patel (2017) briefly reviewed some popular feature extraction methods e.g., Speeded Up 292 Robust Features (SURF) (Bay et al. 2006), Histogram of Oriented Gradients (HOG) (Dalal and Triggs 293 2005) and Linear Binary Patterns (LBP) (Ojala et al. 1996) and machine learning algorithms e.g., K-294 Nearest Neighbour (KNN), Artificial Neural Network (ANN) and Convolutional Neural Network (CNN) 295 for use in fruit classification and grading. Liakos et al. (2018) presented an overview of machine 296 learning (including ANN and deep learning) and its application in the agriculture domain including its 297 use in yield prediction and detection/classification of weed, crop quality and disease. It was noted 298 that the majority of published papers target applications of machine learning in crop management 299 with the most popular models being ANNs. In a more extensive review, Kamilaris and Prenafeta-300 Boldú (2018) reviewed the use of deep learning methods in agricultural applications in general, 301 concluding that the deep learning methods provide improved detection accuracy than previous 302 image processing techniques.

- However, other deep learning techniques have become available since the 2018 reviews were undertaken, particularly the so called 'single shot detectors' that offer improvements in speed (and thus potential for real time application). Further reports on the use of deep learning approaches for fruit detection have also appeared, noting the techniques to generalize very well in real orchard scenes and to be robust to issues such as fruit occlusion and variable lighting conditions for the object detection task (Koirala et al. 2019b).
- 309 Given the success of the deep learning technique in other application areas and the increasing ease 310 of use, there will be a flood of application reports in the agricultural domain. In an attempt to guide 311 such work, this review has three areas of recommendation that shape its outline:
- 312 Applying deep learning: This paper is intended to provide a background on machine vision concepts 313 and terminology, an insight of object detection framework and a review of deep learning approaches 314 for fruit detection based on deep learning. An evolution of both frameworks and detectors can be 315 traced in which detection speed and accuracy has markedly improved over a few years. Object 316 detection frameworks include Faster R-CNN, SSD and YOLO, while detectors include Oxford Visual 317 Geometry Group Network (VGGNet) (Simonyan and Zisserman 2014), Residual Network (ResNet) (He 318 et al. 2016), Zeiler and Fergus Network (ZFNet) (Zeiler and Fergus 2014). Emphasis is placed on 319 practical aspects that require consideration when adopting standard deep learning models for the 320 fruit detection task. Recommendations are made on what is required to make original contributions 321 to the literature, as opposed to formulaic applications of established methods to new application 322 areas (e.g., to a new commodity), including the use of standard metrics for model comparison 323 involving binary classification.
- 324 *Common image sets:* The deep learning community has benefited from publicly available annotated 325 image datasets such as PASCAL Visual Object Classes (PASCAL VOC); Microsoft Common Objects in 326 COntext (COCO) (Lin et al. 2014), and ImageNet (Deng et al. 2009) which contain thousands of 327 common object classes and millions of images, and are available to developers for model training or 328 for benchmarking object recognition algorithms. Unfortunately, these datasets do not contain 329 orchard images. Kamilaris and Prenafeta-Boldú (2018) also commented on the paucity of publicly 330 available data sets for agricultural applications. The deep learning models trained on the large 331 public generic datasets can be fine-tuned for fruit detection with addition of training data through 332 transfer learning. To facilitate such development, and for benchmark comparison, it is

- recommended that fruit-in-orchard image sets be made publicly available for all major fruit treecommodities.
- 335 *Orchard yield estimation:* Much published work has focused on improving the accuracy of algorithms
- to accurately predict the number of fruits within images of tree canopies. Less work has been
- reported that relates image fruit counts to actual yield of an orchard block. Therefore, approaches to
- the issue of occluded fruit are also reviewed.
- 339 The following detail is therefore intended to assist new users of deep learning image processing
- techniques, and hopefully, if in some small way, influence the direction of the coming body of
- 341 application work, particularly in context of fruit detection.

342 **2.2** CNN and deep learning – a background

- 343 Deep learning with convolutional networks (convNets) are widely used for image processing tasks as
- 344 convNets can learn translational invariant patterns, allowing detection of objects wherever
- 345 positioned in an image, and can extract complex visual concepts through detection of a hierarchy of
- 346 increasingly complex patterns (early layers learn simple local patterns, e.g., edges, while later layers
- 347 capture more semantic representations of the object, e.g., shape).
- 348 A deep learning revolution started when AlexNet (Krizhevsky et al. 2012) won the 2012 ImageNet
- 349 Large Scale Visual Recognition Challenge (ILSVRC) (<u>http://image-net.org/challenges/LSVRC/</u>) by a
- 350 large margin (85% accuracy compared to 74% for the runner up model which was based on
- 351 traditional Support Vector Machine classifiers (SVM) (Cortes and Vapnik 1995). The winning entries
- 352 in subsequent years (ZFNet, VGGNet, GoogleLeNet, GBD-Net, SENet; 2013-17) have all involved deep
- learning. The top-5 classification error rate decreased for 2015, 2016 and 2017 ILSVRC challenges, to
- 354 3.6, 3.1 and 2.3%, respectively, while the average human error rate was 5% (He et al. 2015). The
- improved accuracy is in general associated with increased model depth, tempered by use of
- 356 connections between layers. However, training and testing error increase with depth, making
- deeper models more difficult to train (e.g., 'vanishing gradient' problem; He et al. 2016). This issue
- 358 has been addressed in newer architectures that use skip connections and residual networks (He et
- 359 al. 2016).
- 360 ZFNet used an architecture similar to AlexNet but with more convolution filters of smaller sizes. A
- 361 smaller filter size allows capture of information that is more locally distributed in the images, which
- 362 can lead to more accurate classification/detection results. VGGNet, used even smaller filter sizes and
- 363 more convolution layers (16-19 layers) but a huge memory requirement rendered it computationally
- 364 expensive. The use of more layers (deeper model) sacrificed computation speed for accuracy.
- 365 GoogLeNet (Szegedy et al. 2015) introduced inception modules and was deeper (22 layers), but
- 366 computationally efficient. A basic inception module consists of filters of multiple sizes operating on
- the same level. ResNet used a residual learning framework to achieve efficient training of even
- deeper (up to 152 layers) networks. ResNet used residual blocks that featured 'identity shortcut
 connection' which allowed the information to flow without being lost ('vanishing gradient' problem)
- in the deeper networks. Gated Bi-Directional Network (GBD-Net) (Zeng et al. 2018) is a CNN
- 371 architecture which utilizes the relationship among the features of different resolution and candidate
- 372 support regions to detect objects in image. GBD-Net is an attempt to integrate local and contextual
- 373 visual information for more accurate object classification. In the Squeeze-and-Excitation Network
- 374 (SENet) (Hu et al. 2017) a squeeze-and-excitation (SE) block was added to convolution layers to
- boost representational (classification) power. The SE block dynamically models the

interdependencies between convolutional feature maps by exciting relevant features whilesuppressing irrelevant features.

378 Object detection frameworks, which combine both classification and localization into a single system 379 to detect and draw boxes around objects in images, have also evolved markedly in recent years. 380 Region-CNN (R-CNN) (Girshick et al. 2014), released in 2012, combined heuristic region proposal -381 selective search (Uijlings et al. 2013) with Convolutional Network (ConvNet) feature extractors for 382 object detection. OverFeat (Sermanet et al. 2013) implemented feature extraction from multiple 383 square grid cells over multi-scale input image, without the need of separate region proposal. Thus, 384 OverFeat was faster than R-CNN but was less accurate in object localization. Spatial Pyramid Pooling 385 net (SPPNet) (He et al. 2014) introduced adaptively-sized pooling to extract features from a 386 common global feature volume, reporting SPPNet to operate faster than R-CNN. MultiBox (Szegedy 387 et al. 2014) proved that ConvNets are more efficient for region proposals. In Fast R-CNN (Girshick 388 2015), the SPP layer was replaced by a fixed size region of interest pooling (ROIPooling) layer, 389 enabling a speed increase over R-CNN. In 2015, Faster R-CNN replaced the selective search (heuristic 390 region proposal) in Fast R-CNN with a region proposal network (RPN) and was end-to-end trainable 391 (i.e., all model parameters were simultaneously trained using a multi-task loss function). In 2017, He 392 et al. (2017) introduced Mask-RCNN as an extension to Faster R-CNN for instance segmentation (i.e., 393 location of exact pixels followed by masks for each object inside the bounding box).

394 While previous object detection framework were 2-stage methods (region proposal stage followed 395 by classification stage), YOLO was developed as a one stage (single shot) unified object detection 396 model. In YOLO, a single CNN is able to simultaneously predict multiple bounding boxes and their 397 class probabilities. The 'single shot detector' SSD implemented prior boxes (subsets of fixed sized 398 anchor boxes) at different resolutions of feature maps at different levels inside the network for 399 multiscale training, making it a very fast (faster than Faster R-CNN) yet accurate framework for 400 object detection. YOLOv2 (Redmon and Farhadi 2017) gave a speed and accuracy improvement on 401 YOLOv1 (Redmon et al. 2016) through introduction of a pass-through layer, higher resolution 402 classifier and anchor boxes, and achieved nearly the same mean Average Precision (mAP) as Faster 403 R-CNN and SSD on the PASCAL VOC dataset. In 2017, RetinaNet (Lin et al. 2017b), a one-stage 404 detector, outperformed all the previous one-stage and two-stage detectors available at that time in 405 terms of both speed and accuracy. YOLOv3 (Redmon and Farhadi 2018), released in 2018, is deeper 406 than the previous YOLO variants. This architecture achieves an accuracy similar to SSD and 407 RetinaNet, at three and four times the speed, respectively (Redmon and Farhadi 2018). The various 408 YOLO architectures offer a trade-off between speed and accuracy. Pre-trained YOLO models are 409 available from the github repository https://github.com/AlexeyAB/darknet.

410 **2.2.1 Object (fruit) detection in images**

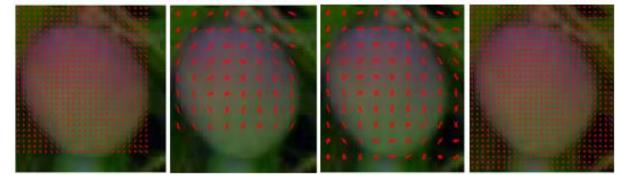
Tree fruit yield estimation by machine vision typically involves the steps of classification of possible regions as fruit objects (using, e.g., colour thresholding, key-point extraction or convolutional filtering) followed by location of individual object (e.g., blob segmentation, shape fitting and bounding box regression). The segmented blob or bounding boxes can then be counted to provide the fruit number in images. Object detection represents one of the most important steps towards yield estimation of fruits. The following section provides the information on what constitutes an object detection framework and details the feature extraction methods.

418 **2.2.2** Framework steps

- An object detection framework (object classification plus localization) generally follows the followingsteps.
- 421 Image pre-processing (image resizing, change of colour space, image data normalization ٠ 422 etc.) 423 Generation of hypotheses (generate possible regions containing objects e.g. test patch at • 424 each location of sliding window, voting from patches or key-points, and region-based proposals using selective search algorithms) 425 Score hypotheses (assign level of probability/confidence for an object to belong to a 426 • 427 particular class/category using classifiers)
- Resolve detection (remove low scoring hypotheses, e.g., using class confidence threshold, and suppress multiple redundant detection, e.g., using Non-Maximal Suppression (NMS), with a goal to assign one box per object).

431 2.2.3 Feature extraction

- 432 In traditional feature extraction, 'handcrafted' features (e.g., colour, shape, texture, intensity) are
- used to compute class membership. As colour of fruit can vary, segmentation should be based on a
 wide colour window, and additional features such as shape and texture should be used (Gongal et al.
 2015).
- The modelling of object class is made difficult by variation in illumination, viewing angle, pose of the
 object (orientation) and position of object (tight clustering, occlusions). Traditional image
- 438 segmentation techniques that rely on morphological operations in binary images are affected by
- 439 strong shadows and occlusion from stem and leaves which split the fruit image into smaller
- segments (e.g., Payne et al. (2013) and Annamalai et al. (2004)). These techniques also tend to count
- 441 fruit clusters as a single blob (fruit) because of pixel connectivity (Annamalai et al. 2004). Algorithms
- such as Circular Hough Transform (CHT) for apple (e.g., (Bargoti and Underwood 2017b; Sengupta
- and Lee 2014; Stajnko et al. 2009)) and citrus (e.g., Choi et al. (2015)) and Random Hough Transform
- 444 (RHT) for mango (e.g., (Kadir et al. 2015; Nanaa et al. 2014)), have been widely used to fit circular
- and oval shapes around possible regions in an image to segment fruits.
- 446 Features such as HOG (Dalal and Triggs 2005), LBP (Ojala et al. 1996), Scale-Invariant Feature
- 447 Transform (SIFT) (Lowe 1999), SURF (Bay et al. 2006) and Haar-like features (Viola and Jones 2001)
- have been widely used in traditional object detection/classification methods in computer vision.
- HOG features (Fig. 2-1) have been used in fruit detection by a number of authors. For example,
- 450 Pothen and Nuske (2016) used a modified HOG feature extractor for detecting apple and grape fruit
- 451 in images. Sa et al. (2015) compared HOG and LBP texture features for capsicum fruit detection.



- 453 Figure 2-1. HOG features, using different block and cell sizes, overlaid on the fruit image.
- 454 Cascade classifiers with Haar-like features (Fig. 2-2) have also been adopted for fruit detection tasks.
- 455 For example, Zhao et al. (2016) attempted detection of ripe tomatoes in a greenhouse setting with a
- 456 cascade classifier using Haar-like features followed by average pixel value (APV) based colour
- 457 analysis. Similarly, Wachs et al. (2009) implemented the Viola Jones cascade classifier with Haar-like
- 458 features for real time detection of green apples within a tree canopy using RGB-IR images.

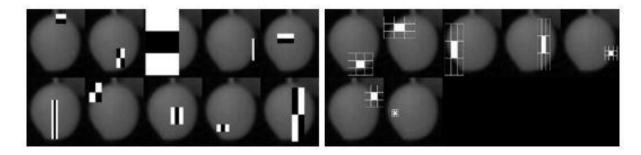


Figure 2-2. Visualization of some Haar features (left panel) and some LBP features (right panel) used by a cascade classifier for mango fruit detection.

462 Other feature extractors are also available. For example, for green citrus fruit detection, Kurtulmus

463 et al. (2011) used the eigenface (Turk and Pentland 1991) ('eigenfruit') approach in combination with464 color and a circular Gabor filter.

465 **2.2.4** CNN as a feature extractor

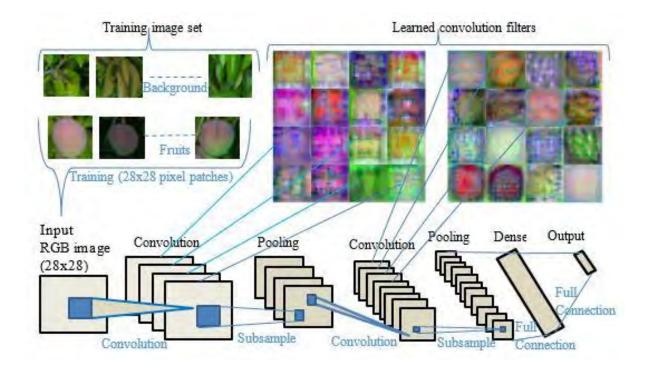
With deep learning methods there is no requirement for manual handcrafting of features, althoughsuch features can be used as a pre-processing input. During training, the CNN automatically

468 undertakes the task of feature selection and classification, and models can be developed to detect

469 many object classes in images. For example, Redmon and Farhadi (2017) reported a YOLOv2 model

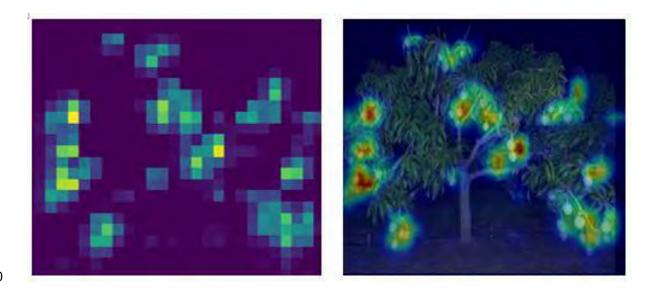
470 recognizing 9,000 common object categories. The tasks involved in use of the traditional

- 471 segmentation (handcrafting) method to detect such a large number of classes would be daunting.
- A basic CNN consists of an input and an output layer, with intervening convolution and pooling/subsampling layers (Fig. 2-3). Image data fed to the input layer passes through the intermediate layers
 to produce a vector of distinct features at the output layer. During training, a CNN develops filters in
 the convolution layers that extract useful information for different object classes. Feature extraction
- 476 occurs in the convolution layers, the output of which can be visualized in an attempt to understand
- 477 the features utilised in the model. Each layer down-samples the image data. Thus, in general, more
- 478 complex and semantic features such as shape and patterns are learned in deeper layer while basic
- 479 features such as colour, edges and lines are learned in the early layers.



- 481 Figure 2-3. The LeNet (LeCun et al. 1998) CNN classifier accepts 28x28 pixel images and processes through
- 482 five layers. Multiple feature maps are created in each convolution layer. Pooling layers subsample data from
- the convolution layers. Example input training images are provided in the top left of the graphic, and
 visualization of the output of example convolution filters (feature extraction) after training is provided in
- 485 the top right.
- 486 The class activation map from the final convolutional layer can be visualized (Fig. 2-4) using methods
- 487 such as Gradient-weighted Class Activation Mapping (Grad-CAM, Selvaraju et al. (2017)).
- 488 Visualisation allows for some interpretation of the model function. In the case of Figure 2-4, the
- 489 model visualisation indicates weighting of the fruit regions of the images.

Deep learning- review



490

491 Figure 2-4. Grad-CAM visualization of activation map of the final convolutional layer of a deep learning 492 'Xception' (Chollet 2017) model (trained to directly predict fruit count from input images) as heat-map (left) 493 and a heat map superimposed on input image (right).

494 The CNN feature extractor is used by object detection frameworks for classification tasks, using box

495 regression for object localization. For example, AlexNet has been implemented in RCNN and

496 Overfeat and ZFNet is used in SPPNet, VGG-16 is used in Fast R-CNN, Faster R-CNN and SSD, Darknet497 19 is used in YOLOv2, and Darknet-53 is used in YOLOv3. However, depending upon the application,
498 feature extractors can be used interchangeably. For example, ZFNet could be used in a Faster R-CNN

499 framework.

2.3 Deep learning object detection framework

The short developmental history of deep learning frameworks for image object detection are
reviewed in the following sections, finishing with consideration of the 'state of art' one stage
detectors.

504 2.3.1 CNN Sliding window detectors

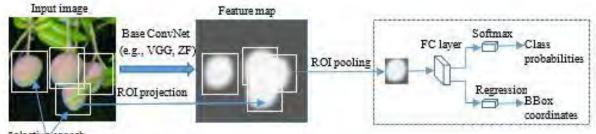
505 Early CNN based object detection frameworks such as Overfeat used a sliding window approach 506 where the classifier is run at evenly spaced locations over the image. A large number of patches are 507 generated at each position of the sliding window, with each patch classified as containing an object 508 or not. For multi-scale detection, patches are generated at each scale. Feeding all available patches 509 to a CNN slowed down the object detection framework.

- 510 R-CNN replaced the sliding window method with relatively faster heuristic Selective Search
- algorithm (Uijlings et al. 2013) to filter out some regions, feeding only the potential region proposals
- 512 into the CNN. Fine-tuned CNN was used to extract features from each region proposal for a SVM
- 513 classifier. However, the feeding of nearly 2,000 region proposals per image through a CNN rendered
- this method slow.

515 2.3.2 Two stage detection

- 516 To decrease detection time, it is desirable to feed the whole image to the CNN model in one pass to
- 517 generate a final feature map. A two-stage approach (region proposal stage followed by
- 518 classification/detection stage) can deliver this goal, as achieved in Fast R-CNN and Faster R-CNN.

- 519 **Fast R-CNN** was developed to improve on the detection speed of R-CNN. Feature extraction was
- 520 performed on the whole image, and region proposals were generated on the final feature map (Fig.
- 521 2-5). Region of Interest (RoI) pooling was implemented to obtain a fixed sized feature vector from all
- 522 cropped feature maps for classification. Detection speed was increased by use of a single Softmax
- 523 layer, which was sufficient for prediction instead of training several SVMs. However, the generation
- of region proposals using Selective Search remained as a bottleneck to further speed improvement.



Selective search

526 Figure 2-5. The fast R-CNN object detection framework.

- 527 Faster R-CNN replaced the heuristic selective search method of Fast R-CNN with a Region Proposal
- 528 Network (RPN), which uses CNN and anchor boxes for speed improvement. RPN slides a 3x3
- 529 convolution window on the final feature map and at each window location considers 9 different
- anchor boxes (composed of 3 scales and 3 aspect ratios) to generate region proposals (Fig. 2-6).
- 531 These proposed regions are filtered based on an objectness score (probability the proposed region
- 532 contains object) and passed to next stage (essentially a Fast R-CNN) for object detection. However,
- 533 despite the improved detection speed achieved in Faster R-CNN compared to Fast R-CNN, the
- 534 pipeline was still too slow to apply on real-time videos/streaming.

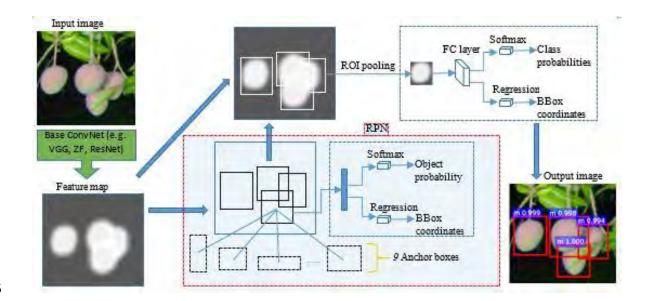
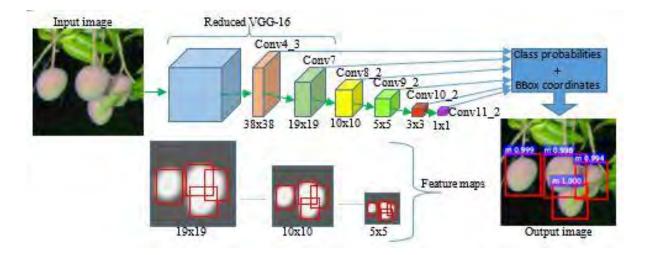


Figure 2-6. The Faster R-CNN object detection framework (RPN plus Fast R-CNN). Output image shows
 bounding boxes (Bboxes) on detected objects of class 'm' (mango), with associated probability of correct
 classification.

539 2.3.3 One stage detection (Single shot detectors)

- 540 To achieve further improvement in speed, SSD and YOLO removed the region proposal stage and the 541 CNN was designed to consider dense sampling of possible object locations for object detection.
- 542 Single Shot MultiBox Detector (SSD) simultaneously predicts the object class and bounding box on
- 543 the image making it faster than Faster R-CNN. SSD feeds the input image through series of
- 544 convolution and pooling layers to generate feature maps at different scales (Fig. 2-7). A 3x3
- 545 convolutional window at each location of feature maps evaluates a small set of default anchor boxes
- 546 (separate set of boxes of different aspect ratios at different resolution of feature maps) for which
- 547 SSD simultaneously predicts the class probabilities and bounding box offsets for different scales.
- 548 There can be different number of detections at each scale. Due to the multiscale detection
- approach, SSD is very effective in detecting objects of different sizes in the image.

Deep learning- review



550

551 Figure 2-7. SSD object detection framework. 'Conv' refers to 'convolution layer'. Output image displays 552 bounding box on ROI, classification result ('m' for mango) and probability score.

YOLO (You Only Look Once) or YOLOv1 is a single shot detector in which a fully convolution neural
 network converts the input image to a tensor of scores for object detection. The prediction of class

probabilities and bounding box coordinates from the final feature map in a single forward pass
 through the CNN makes YOLO one of the fastest object detection methods. YOLO divides the input

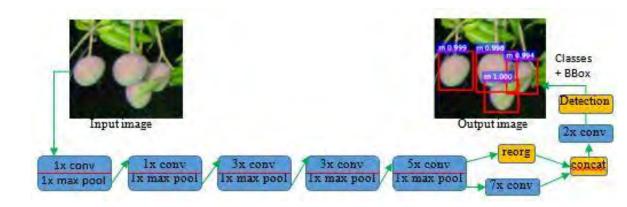
image into grid cells (each grid cell is 1/32 times the network input resolution), and each grid cell is

responsible for detecting object.

559 YOLOv1 does not use anchor boxes like Faster R-CNN and SSD, but instead directly predicts two 560 bounding boxes and one class per grid cell. Since YOLOv1 learns to predict boxes directly from the

561 image data it produced many localization errors (for small objects and objects in groups). YOLOv2

- 562 provided an accuracy and speed improvement on YOLOv1. YOLOv2 achieved nearly the same mean
- 563 Average Precision (mAP) as Faster R-CNN and SSD on PASCAL VOC dataset and was still the fastest
- 564 detector. Redmon and Farhadi (2017) reported the object detection speed of Faster R-CNN (VGG-
- 16), SSD-300 and YOLOv2-288 at 7 fps, 46 fps and 91 fps respectively.YOLOv2 introduced anchor
- boxes, with prediction of more than 1000 boxes per image, compared to just 98 for YOLOv1. YOLOv2
- also implemented a pass-through layer that concatenates the features from a higher resolution layer
- to lower resolution layer, providing the advantage of small object detection from the finer grained
 features (Redmon and Farhadi 2017) (Fig. 2-8). The backbone feature extractor for YOLOv1/v2 was
- 570 the Darknet-19 framework which has 19 convolution layers, mostly 3x3 filters similar to VGG net.



572 Figure 2-8. YOLOv2 object detection framework. 'Conv' refers to 'convolution layer', 'concat' to

573 'concatenation layer' and 'reorg' refers to 'reorganize/route'. Output image displays bounding box on ROI, 574 classification result ('m' for mango) and probability score.

575 **YOLOv3** was an improvement over YOLOv2 in terms of detection accuracy. According to Redmon

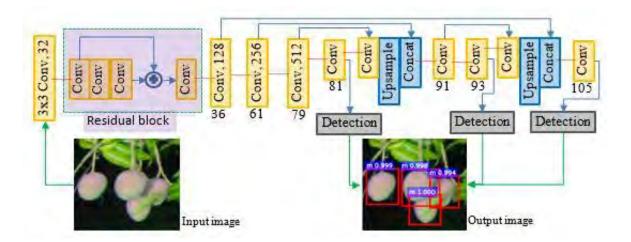
576 and Farhadi (2018), YOLOv3 is as accurate as SSD and RetinaNet, but 3.0 and 3.8 times faster,

577 respectively. YOLOv3 implements similar concept to feature pyramids (Lin et al. 2017a) for

578 multiscale box prediction. The backbone feature extractor for YOLOv3 is Darknet-53, which has 53

579 convolution layers (successive 3x3 and 1x1 convolutional layers with skip connections similar to

580 ResNet) (Fig. 2-9).



581

Figure 2-9. Block diagram of YOLOv3 architecture. 'Conv' refers to 'convolution layer' and'concat' to
 'concatenation layer'. Output image displays bounding box on ROI, classification result ('m' for mango) and
 probability score.

585 2.4 Network and model training

The following sections cover the typical steps involved in developing and testing a deep learning
model. Emphasis is placed on appropriate data set structure, determination of the appropriate size
of the training set, the use of pre-existing models and the image size.

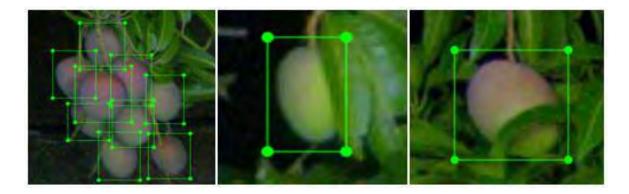
589 **2.4.1 Training and testing requirements**

590 The creation and validation of a model requires (i) training, (ii) validation (tuning) and (iii) test sets of 591 images and associated annotation files containing bounding box co-ordinates and the class of the

- 592 object. Manual bounding box annotation is tedious and prone to user bias in ground truth labelling,
- 593 especially for images having a large number of objects and clusters with a high degree of occlusion
- 594 (Bargoti and Underwood 2017a). The quality of annotated training image sets can thus be an issue, a
- 595 factor which should drive the use of shared image sets.

596 The validation set is used for tuning of model parameters such as confidence and overlap thresholds 597 before the model is applied on the test set, e.g., Bargoti and Underwood (2017a) and Koirala et al.

- 598 (2019b). The test set should be "independent" of the training and validation sets, where
- 599 independence refers to different orchards or seasons for the machine vision task of detection of fruit
- 600 on canopy. Unfortunately, many published papers used an image set collected in one event for these
- 601 three functions. It is a common practice to reserve a randomly selected 10 to 20% of an initial set
- 602 for use as a validation set. The reservation of images from one row of trees as the test set is better
- practice, but ideally the test set should involve images of another orchard or season, to emulate thereal-world condition for model use.
- 605 Various object detection methods require different data formats for processing image data during
- training. Many object detection systems include the scripts to parse the annotation files in PASCAL
- 607 VOC format to their own formats. For example, the online code repository of Faster R-CNN, YOLO
- and SSD have scripts to parse PASCAL VOC annotation files. Therefore, structuring dataset and
- 609 directories similar to PASCAL VOC dataset format can avoid the need to change the scripts that are
- 610 ready to parse annotations to framework specific format. The following recommendations are
- 611 offered as practical advice in the creation of models:
- 612 Object detection frameworks train object classes against background while trying to match the
- 613 detected box to the ground truth box for maximum overlap. Therefore, ground truth boxes should
- be tight enough to cover the object and some background around the object perimeter (Fig. 2-10).
- For overlapping (clustered) fruit, emphasis should be put on reducing the box overlaps among
- neighbouring fruits as much as possible to avoid merging into single detection from NMS (Koirala et
- 617 al. 2019b).



618

Figure 2-10. Examples of ground truth labelling of individual fruit using LabelImg software in scenes with varying levels of occlusions by other fruits or leaves.

- To enable another user to replicate the development of a deep learning model, it is necessary to
- detail the process followed in organizing data and in pre-processing of data (e.g., colour space
- 623 conversion, histogram normalization, image resizing/cropping etc.). This includes consideration of
- 624 the number of training/test/validation images used.

625 2.4.2 Number of training images

626 The minimum number of images required for training a deep learning model depends on the visual 627 complexity of the image for object detection and the deep learning models used for learning. An 628 assessment of minimum number of training images sufficient to capture the variation in the test set 629 can be done as described by Bargoti and Underwood (2017a) and Koirala et al. (2019b) for mango detection in day-time and night-time images, respectively. In this approach, AP is calculated for 630 631 models developed using an increasing number of images sampled from the training set. Bargoti and 632 Underwood (2017a) report that for the image set used, AP asymptoted with use of ~729 apple fruit 633 images, while AP for models for detection of almond and mango did not stabilise with full available 634 set of 385 and 1154 training images, respectively. Koirala et al. (2019b) imaged the same mango 635 orchard as Bargoti and Underwood (2017a), but used different imaging hardware, with images 636 collected at night under artificial light. An AP plateau was achieved with fewer (approximately 400) 637 training images, a result attributed to a lower level of variation and background noise in the images. 638 Data augmentation techniques (e.g., random crop, flips, zoom, change in hue, saturation etc.) can be 639 used to expand the variation in training images. This augmentation helps to increase generalizability 640 of the model and to reduce the chance of data overfitting. Bargoti and Underwood (2017a) reported 641 that an AP score of 0.86 was achieved for apple detection in images with just 100 training images 642 when data augmentation was used, compared to 300 images without data augmentation. Large 643 numbers of artificial synthetic images can be generated for training neural networks, as 644 demonstrated for a fruit detection application by Rahnemoonfar and Sheppard (2017). All deep 645 learning object detection frameworks provide some level of data augmentation which can be 646 selectively turned on or off.

- In summary, different numbers of training images are required for different crops and different lighting conditions, in consequence of the amount of image variation in each case. Therefore, training data should be carefully chosen such that it is relevant to the problem space and has enough variation in relation to the production scope or context of deployment. The availability of labelled training data can limit the choice of network architecture to be used. Larger networks can be more accurate but will require more training data. However, lack of data should not be a defence for the lower performance of a deeper model.
- Publicly available annotated fruit image datasets would facilitate comparison of models, with use of
- the same training and validation data. Some annotated fruit datasets are publicly available. For
- example, the 575 images at various resolutions of avocado, orange, apple, mango, strawberry,
- rockmelon and capsicum fruits acquired in glasshouse and through Google image search is available
- at <u>http://enddl22.net/wordpress/datasets/deepcrops-datasets-and-annotation-tool</u> (accessed on
- 659 3/01/2019), as reported by Sa et al. (2016)). The image data used by Bargoti and Underwood (2017a)
- 660 (1120 apple fruit images at 308x202 pixels, 1964 mango fruit images at 500x500 pixels and 620
- almond fruit images at 308x202 pixels) is available at
- 662 <u>https://data.acfr.usyd.edu.au/ag/treecrops/2016-multifruit</u> (accessed on 3/01/2019). Koirala et al.
- 663 (2019b) (http://hdl.cqu.edu.au/10018/1261224) provides 1,730 mango fruit on canopy images (at
- 664 612x512 pixels). These images were collected at night. The Fruit-360 dataset by Mureşan and Oltean
- 665 (2018) (https://github.com/Horea94/Fruit-Images-Dataset: accessed on 1/03/2019) contains 6,5429
- 666 images (100x100 pixels) of popular fruits (95 fruit classes) and is intended for fruit recognition
- 667 applications. The MangoNet semantic dataset by Kestur et al. (2019)
- 668 (https://github.com/avadesh02/MangoNet-Semantic-Dataset: accessed on 28/02/2019) contains 49
- 669 daytime mango fruit on canopy images (at 4,000x3,000 pixels).

670 **2.4.3 Transfer learning**

671 For efficient and more stable training of deep learning models, it is common practice to employ 672 transfer learning (also referred to as or fine-tuning) on a network pre-trained on a large dataset (e.g., 673 ImageNet). Transfer learning allows the re-use of existing parameters (convolution weights) from a 674 model trained on large datasets for training new models from relatively less number of training 675 images. Publically available datasets such as ImageNet (450k images, 200 classes), PASCAL VOC (12k 676 images, 20 classes) and COCO (120k images, 80 classes) provide labelled data of common objects in 677 images, free for training and benchmarking object detection models. During transfer learning, the 678 weights from the earlier layers of pre-trained model are copied to a new model. These weights 679 contain information about the basic features found in common objects, such as colour, shape, 680 edges, and lines. The final classification layer, which is responsible for high level classification of the object into class categories, is not transferred. The model is then trained further on new classes. 681

If there is a sufficient number of training images available for a particular application, the transferlearning process holds no advantage. Bargoti and Underwood (2017a) reported that there was no significant performance difference between weights initialized from ImageNet and from other orchards (almond and mango) for training a Faster R-CNN model to detect apple on images. Koirala et al. (2019b) also reported no significant performance gain for a mango fruit detection model (MangoYOLO) trained with over 1,000 image tiles when initialized with the COCO pre-trained

688 weights.

689 2.4.4 Image and object sizes

The optimum image/tile size should be established for the imaging conditions (camera resolution,
lens, camera to canopy distance, fruit size, etc.) and the machine vision model architecture used. A
larger object size (pixel number) in images is desirable for proper object detection and localization,
but higher resolution incurs greater computational time and memory. Zhao et al. (2016), Cheng et al.
(2017b), Hung et al. (2015) and Sengupta and Lee (2014) down-sized images for computational
efficiency in fruit detection applications.

- 696 Image resizing (subsampling) is inherent in most deep learning architectures. The CNNs have a 697 predefined input size (network input resolution). All input images will be resized to the network 698 input resolution before feeding through the network. For example, SSD-300 resizes all input images 699 to 300 x 300 pixels before further processing inside the network. This step reduces the size of the 700 objects in the image, which can impact the performance of object detection models. Moreover, 701 there are several subsampling layers inside a CNN which cause further resizing of the input image. 702 For example, CNN feature extractors such as ZF and VGG, as used in the object detection 703 frameworks of Faster-RCNN and SSD, use a subsampling factor of 16. Thus, object pixel size is further 704 decreased in the final feature map, providing little pixel information for detection. Network
- architecture can be changed to accept images of higher resolution, but at the cost of highercomputation and training memory requirement.
- 707 Close-up views of fruit from Google Images or from imaging of fruit in glasshouses, e.g., as used by
- 708Sa et al. (2016), provide high resolution detail of the fruit. Such images can tolerate a subsampling
- factor of 16 and still have objects (fruit) of a reasonable pixel dimension. In comparison, canopy
- orchard imagery provides relatively less resolution for fruit in the image, unless a very high-
- 711 resolution camera is used. Use of tiles from the image rather than the whole image is useful in
- decreasing the effect of downsizing. Splitting of large images into sub-images also decreases the fruit
 count per image, helping to reduce human labelling errors. Moreover, object detection frameworks
 - Page | 23

- trained on tiles (e.g., 500 x 500 pixels) can be used for inference on whole images (e.g., 2048 x 2048
- pixels) without need for further training (Koirala et al. 2019). Alternatively, to reduce memory
- requirement, a tiling approach can also be followed during detection, as used by Bargoti and
- 717 Underwood (2017a). In this process, detections are performed using a sliding window on sub-
- sections of the image, with thresholding and NMS applied over fused output on large images.

719 **2.5 Performance assessment**

The algorithm of a new deep learning based machine vision system should be appropriately

documented to enable a skilled practitioner to replicate the work, and the performance of the

- 722 model must be evaluated relative to existing best practice. There are a set of parameters and
- metrics that are commonly used in machine vision evaluation that should be reported in anypublished study.

725 **2.5.1 Parameters**

726 **2.5.1.1** IoU

For the object detection task, the Intersection over Union (IoU, also known as the 'Jacard index' is
the ratio of the area of overlap to the area of union between detected and the ground-truth

bounding boxes (Fig. 2-11). IoU can vary from 0 (no overlap) to 1 (full overlap). This metric is set for

730 use in model training, with use during testing by the NMS to suppress multiple redundant

detections. The object detection challenges of PASCAL-VOC and ImageNet both set an IoU of 0.5 of

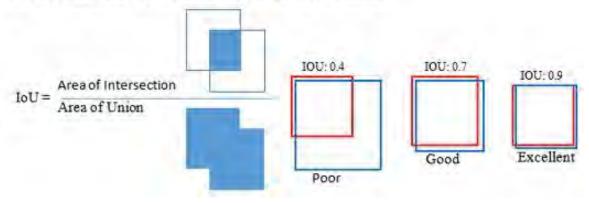
732 detection on ground-truth bounding as valid detections. Any detections with an IoU between

detected box and ground truth annotation greater than the set IoU threshold is considered as a

positive detection. Detections with an IoU less than threshold value are false detections.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} ; 0 \le J(A,B) \le 1$$

Where A and B are two finite sets and J is the Jaccard index.



735

736 Figure 2-11. Definition of IoU in the object detection task using bounding box annotations.

737 Research reports should report the IoU used in their work and may seek to optimise the value used.

738 For example, for pepper fruit detection, Song et al. (2014) considered a detection to be a true

positive when IoU > 0.5 (between predicted and ground truth boxes). In contrast, Bargoti and

740 Underwood (2017a) used an IOU > 0.2 to better detect small fruit. Sa et al. (2016) also justified use

of an IOU>0.4 as the threshold for detection of relatively small fruit size in their image dataset.

Deep learning- review

Similarly, the non-maximal suppression (NMS) threshold value (the IoU used to remove the

redundant overlapping detections in output images) should be reported for replication of the work.

744 **2.5.2** Evaluation metrics

745 **2.5.2.1** Useful metrics

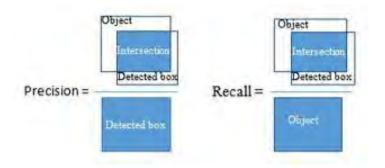
There are a range of metrics useful for the characterisation of model performance in the application of tree fruit detection in canopy images. For example, the review by Kamilaris and Prenafeta-Boldú (2018) tabulates values for classification accuracy, Precision, Recall, F1 score, L2 error, IoU, ratio of estimated to actual fruit count per image and root mean square of error (RMSE) and mean residual error (MRE) on estimated compared to actual fruit count per image.

751 2.5.2.2 Precision and Recall

- 752 The fruit detection task is a binary classification problem with the following results:
- True positive (TP): Fruit detected as fruit
- True negative (TN): Background detected as background. This is not applicable for deep
 learning object detection frameworks like (Faster R-CNN, SSD and YOLO) which do not
 require labelling of the background class.
- False positive (FP): Background detected as fruit

False negative (FN): Fruit not detected
Precision(P) =
$$\frac{TP}{TP+FP}$$
 (1)
Recall(R) = $\frac{TP}{TP+FN}$ (2)

The term *Precision* gives the number of true detections out of total detections, while *Recall* gives the number of true detections out of total ground truth annotations (Fig. 2-12).



763

Figure 2-12. Interpretation of *Precision* and *Recall* for object detection task using bounding box annotation.
 Precision is the proportion of detected boxes that matched the ground truth boxes. *Recall* is the proportion
 of detected boxes that matched the ground truth boxes.

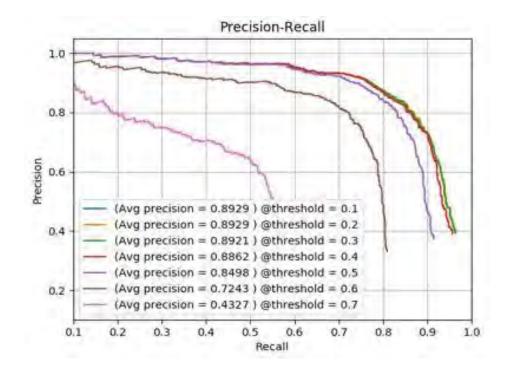
767 Both *Precision* and *Recall* measures are essential to characterisation of the performance of an

algorithm on an image set. A high *Precision* is sought when there is high cost associated with false

positives. A high *Recall* is sought when there is high cost associated with false negatives. *Precision*

- and *Recall* values are affected by the IoU threshold between the predicted and ground truth box
- (Fig. 2-13). *Precision* and *Recall* values can be optimized by changing the IoU threshold or limiting

- number of detections (changing confidence threshold and NMS threshold). For a good model,
- 773 Precision remains high as Recall increases, i.e., the model is able to detect a high proportion of true
- positives before it starts collecting false positives.



775

Figure 2-13. Precision-Recall curves for a Faster R-CNN with ZFNet model used to detect mango fruit in whole tree images, obtained for varying IoU thresholds (0.1-0.7) at a NMS threshold of 0.5.

778 2.5.2.3 Average Precision

The area under the P-R curve (known as Average Precision, AP) can be used as a single metric to
summarize the performance of the object detection model (Fig. 2-13). For example, Bargoti and
Underwood (2017a) and Koirala et al. (2019b) used AP in assessment of performance of deep
learning models in context of the number of training images. A model with high Precision at all levels
of Recall will have a high AP score, while methods that return a high Precision with a subset of
detections will not.

785 Multi class object detection challenges (like PASCAL VOC) commonly use mean Average Precision
 786 (mAP) as metric to evaluate model performance. mAP is the mean of AP computed over all classes.

787 2.5.2.4 F1 Score

For any classifier (object detection model), there is a trade-off between Precision and Recall, and a
model can be optimised (e.g., through change in class confidence threshold and NMS threshold) for
either higher Precision or Recall, depending on the detection task.

F1 is the weighted average (harmonic mean) of Precision and Recall (eqn. 3), calculated from the
point on the P-R curve where P and R have higher and identical values. The F1 score varies between
0 (worst) and 1 (best). If the model is tuned in favour of either Precision or Recall, the F1 score will

794 decrease.

795
$$F_1 = 2 * \frac{precision*recall}{precision+recall}$$

796 2.5.2.5 Fruit detection evaluation

The similarity of fruit and foliage features can cause many false positives detections, while false
negatives can result from partial occlusion and poor illumination. The success of a model for fruit
detection in orchard scenes can thus be evaluated in terms of Precision, Recall, AP and F1 score.

A recommendation is made for use of F1 score as an overall performance measure, allowing fruit

detection models/methods to be compared and benchmarked with a single metric. The metric is
 already in common use to compare fruit detection models, e.g., Bargoti and Underwood (2017a), Sa

803 et al. (2016), Pothen and Nuske (2016) and Koirala et al. (2019b).

804 2.6 Architecture and model optimization

A 'network architecture' is defined by its structure in terms of number of layers, filters and
connections. The network also contains several hyper-parameters (e.g., learning rate, momentum,
weight initialization and activation function) which determines the performance of a trained model.
A model is created through the process of training a network using a training image set. Both the
network and the model can be optimised for a given application.

810 **2.6.1** Architecture optimization

811 Many published works in precision agriculture compare established network architectures for a

812 particular application. In general, a deeper network architecture with more layers in CNN will

813 improve prediction accuracy and achieve higher accuracy compared to a shallower network. For

814 example, VGG is a deeper model with higher accuracy but requires greater memory and

815 computation time than ZF-Net, as demonstrated for a fruit counting application by Koirala et al.

- 816 (2019b) and Bargoti and Underwood (2017a). Such comparisons have practical merit, but to advance
- the discipline some optimisation of the model architecture should be attempted. Of course, new
- 818 architectures should be benchmarked to an existing, well reported one, with Faster R-CNN(VGG)
- 819 acting as the current 'standard'.
- 820 Existing CNN architectures can be modified depending upon the application task to optimise
- 821 performance in context of speed, accuracy and memory requirement. For example, Sa et al. (2016)
- altered the VGG structure to accept four channels (RGB and NIR). Rahnemoonfar and Sheppard
- 823 (2017) modified Inception-ResNet model, achieving an accuracy of 91% compared to 76% with the
- original inception-ResNet. Koirala et al. (2019b) modified the YOLO architecture to create an
- architecture deeper than YOLOv1 but shallower than YOLOv3, for memory and speed optimization in
- the task of mango fruit detection.

To advance the common scientific base, a published report on performance of a new architecture should be accompanied by a clear description of the algorithm, and a performance comparison to an existing standard architecture. If an existing architecture is being used, the report should detail the source of the open-source code used for compiling the deep learning models. Ideally, the officially

- released algorithm codes from published papers should be implemented, to allow for replication of
- 832 work.

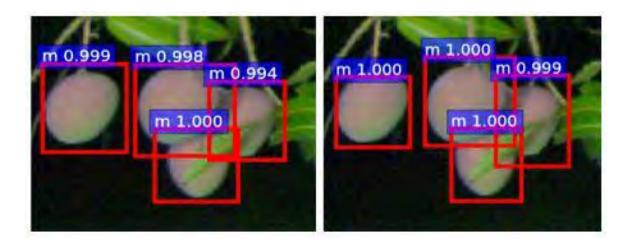
833 **2.6.2 Model optimization**

In the process of training a given architecture, a model is created with weightings unique to the
training set used. Parameters such as learning rate and momentum of the network and the number
of filters in each layer can be varied, depending on the visual complexity of the object class to be
modelled, while NMS and class confidence thresholds can be varied to obtain the desired detection
output (Koirala et al. 2019b).

839 2.6.2.1 Class confidence scores

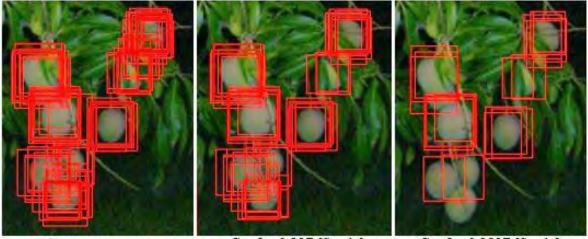
- 840 The class confidence (or probability) score is a numeric value (0-1) assigned to each detection
- 841 describing the confidence or probability of a detected object belonging to a particular class (Fig. 2-
- 14). If the confidence score threshold is relaxed (set low) many detections will be accepted
- 843 (increasing TP and FP) (Fig. 2-15). The confidence cut-off (threshold) must be selected for the
- 844 application.

Deep learning- review



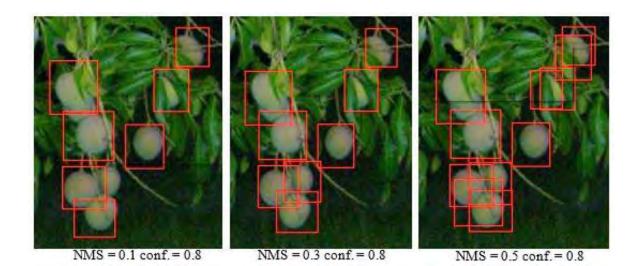
845

- Figure 2-14. An example display of class label 'm' (mango fruit) and associated confidence scores for an
- image of mango fruit on tree, produced by Faster-RCNN using ZFNet (left panel) and VGGNet (right panel).



848

- Conf. = 0.1 NMS = 1.0
- Conf. = 0.8 NMS = 1.0
- Conf. = 0.95 NMS = 1.0
- Figure 2-15. Object detection with no suppression (NMS=1.0) and an increasing level of confidence
- threshold values (0.1, 0.8, 0.95, for left to right panels) resulting in fewer multiple detections per fruit (lower
 FP), but failure to detect some fruit (higher FN).
- 852 2.6.2.2 Non-maximum suppression
- 853 NMS is a common technique used by various object detection frameworks to suppress multiple
- redundant (low scoring) detections with the goal of one detection per object in the final image (Fig.
- 2-16). All detected boxes with an overlap greater than the NMS threshold are merged to the box
- 856 with the highest confidence score. NMS accepts IoU values between 0 (no overlap) to 1 (complete
- 857 overlap).



858

Figure 2-16. Effect of NMS setting: left to right panels: NMS = 0.1, one detection for each fruit but some FN; NMS 0.3, one detection for each fruit with no FN; NMS 0.5, but not all detections merged.

Fruit detection models should therefore be tuned for both confidence threshold and NMS values toachieve the highest F1-score on the validation set as illustrated in Koirala et al. (2019b).

863 **2.6.3 Training and test time**

Few papers report on training and detection time. Even when such information is reported,
comparison is difficult given use of different CPUs and image resolution. Model training time is
generally not critical as it represents a one off exercise. Training time is a function of batch size
(number of images processed in parallel), which depends on available memory and, thus, on the
computing resource employed. Institutional researchers often access High Performance Computing
(HPC) clusters. Deep learning models can also be trained on cloud based high performance
computers (e.g., Amazon's AWS, Microsoft's Azure and Google's GCP), alleviating the need for a local
computing resource

- 871 computing resource.
- 872 In contrast, test or detection time is of importance to the use of a detection system if real or near
- 873 real time detection is required, e.g., at a processing time of 1 second per image, an orchard yield
- estimation for a large orchard of 100,000 trees, using two images per tree, requires 55 h. Future fruit
- harvesting operations will require near real time detection. In general, a processing rate of >30
- 876 frames (images) per second (<30 ms per image) is regarded as a real-time speed for video
- 877 processing, but slower rates may be acceptable for particular applications.
- 878 Detection speed will be a function of algorithm complexity and computing capacity. For example,
- 879 good accuracy on sweet pepper detection was noted for a Conditional Random Field (CRF)
- 880 framework relative to a deep learning framework (Faster R-CNN), but at the cost of processing time
- 881 (842 times greater) and a more tedious ground truthing for the pixel-level approach, compared to
- bounding box annotation (Sa et al. 2016). System design will therefore depend on the importance of
- accuracy relative to speed.
- 884 Sa et al. (2016) employed a GeForce GTX980M 8GB GPU, reporting processing of 1297x964 images
- within ~341 ms, and 1920x1080 images within ~393 ms for a multimodal Faster R-CNN model used
- in detection of sweet peeper fruit. Rahnemoonfar and Sheppard (2017) reported processing time
- using a NVidia 980Ti 6GB GPU for fruit count on tomato plant (128x128) images at 6 and 50 ms,
- using deep learning CNN and area-based methods, respectively. On a NVDIA 980 Ti GPU, Bargoti and

- Underwood (2017a) reported average detection times of 130 ms per (500x500) image using a Faster
- 890 R-CNN framework VGG-16 model, 40 ms per image using a ZFNet model and 2,300 ms per image
- using a pixel-wise CNN model. Koirala et al. (2019b) reported on performance, memory requirement
- and inference time for several deep learning models (VGG, ZFNet and DarkNet) and object detection
- frameworks (Faster R-CNN, SSD and YOLO) in context of a mango fruit image dataset. For an image
- size of 512x512 pixels and using a HPC platform (NVIDIA Tesla P100 GPU), SSD and YOLO variants
 YOLOv3, YOLOv2, YOLOv2(tiny) and MangoYOLO took 70, 25, 20, 10 and 15 ms per image,
- respectively, for fruit detection, while Faster R-CNN(ZF) and Faster R-CNN(VGG) took 37 and 67 ms,
- respectively, for full detection, while faster R-Criticity and faster R-Criticity (VGG) took 57 and 07 ms respectively. For 2048x2048 input images, the detection times for MangoYOLO and YOLOv2 (tiny)
- 898 were 20 and 10 ms and 70 and 51 ms on the HPC and on a NVIDIA GeForce GTX 1070 Ti GPU (local
- 899 computer) platforms, respectively.

900 2.7 Fruit detection using deep learning

- Prior to 2016, published work to discriminate fruits from background in images of tree fruit canopies
 was based on hand-engineered features to encode visual attributes (Gongal et al. 2015).
- 903 Subsequently there have been 12 papers published on use of deep learning models in estimation of
- 904 tree fruit number per image (Scopus data base, <u>www.scopus.com</u>,
- keywords:'deep'+'learning'+'fruit'+'detection', accessed 19/01/2019, revealing 9 publications, with
 an additional four publications located 'by hand') (Table 2-1).

907

Table 2-1. Scientific reports on use of deep learning models in estimation of tree fruit number per image.
 The best result of each paper is shown. When available, the F1 score is recorded, otherwise the validation
 metric used by the authors is included.

Author	Commodity	Method	Validation statistics
Sa et al (2016)	sweet pepper	Faster R-CNN	F1 0.84
Bargoti and Underwood	apple, mango, almond	Faster R-CNN ZF	F1 0.89, 0.88, 0.73
(2017a)	apple, mango, almond	Faster R-CNN VGG	F1 0.90, 0.91, 0.76
Bargoti and Underwood (2017b)	apple, mango	Pixel wise CNN	F1 0.86, 0.84
Chen et al. (2017)	apple	FCN + CNN regression	Ratio counted 0.91
	orange	FCN + CNN regression	Ratio counted 0.97
Choi et al. (2017)	Citrus (classification)	AlexNet	TP rate 96%
Habaragamuwa et al. (2018)	strawberry	CNN	AP 88.0%
			Bounding Box Overlap 0.7
Liang et al. (2018)	mango	SSD VGG	F1 0.91
Peng et al. (2018)	apple, litchi, navel orange, Huangdi gan	SSD ResNet	F1 0.96
Tao et al. (2018)	peach, apple, orange	Faster R-CNN	AP 91.5, 94.2, 90.2 %
Xiong et al. (2018)	green citrus	Faster R-CNN	F value 77.5%
			mAP 85.5
Koirala et al. (2019b)	mango	Faster-RCNN ZF	F1 0.94
		Faster-RCNN VGG	F1 0.95
		SSD VGG	F1 0.96
		MangoYOLO	F1 0.97
Kestur et al. (2019)	mango	CNN	F1 0.84

911

- 912 The first report on use of deep learning with CNN for fruit detection was by Sa et al. (2016). These
- authors implemented a multimodal faster R-CNN (using both RGB + NIR), reporting an improved F1
- score (increased from 0.81 to 0.84) for sweet pepper detection compared to their prior work

- 915 (McCool et al. 2016), which was based on a pixel-wise segmentation technique. A F1 score >0.8 was 916 obtained for all 6 fruit categories (apple, orange, avocado, mango, strawberry) considered.
- 917 Subsequent studies have implemented the latest architectures from the deep learning CNN
- 918 community, with some studies attempting to modify architecture for the fruit
- 919 classification/detection application. F1 scores of >0.9 are being routinely achieved. A summary of
- 920 representative studies follows:
- 921 Chen et al. (2017) proposed a deep learning method to directly estimate total fruit count from input
- 922 images, with integration of fruit segmentation and a count regression network into a single pipeline.
- 923 A Fully Convolutional Network (FCN) (Long et al. 2015) was used to extract candidate regions in the
- 924 images as blobs, while another CNN was used to count the number of fruits (oranges and apples)
- inside each blob of fruit region. A regression model related CNN derived fruit count to human fruit-in-image count.
- 927 Other studies have used deep learning with CNN to detect fruit, then counted the detected fruit. For
- 928 example, Bargoti and Underwood (2017a) used the original Faster R-CNN framework, achieving
- higher F1 scores using VGG-16 than ZFNet (Table 2-1). The results were superior to work using pixel-
- 930 wise CNN (a multiscale Multi Layered Perceptron with three hidden layers and a CNN of two
- 931 convolution, pooling and fully connected layers respectively) (Bargoti and Underwood 2017b). The
- 932 F1 score for fruit detection was thus improved with use of deeper networks (pixel-wise CNN vs ZFNet
- vs VGG-16) (Table 2-1). Similarly, (Kestur et al. 2019), used a full convolutional deep CNN-
- 934 'MangoNet' to segment mango fruit in the images followed by connected object detection for fruit
- 935 counting in images.
- Rahnemoonfar and Sheppard (2017) took the innovative step of deep simulated learning within the
- task of tomato fruit counting. Inception-ResNet architecture was modified, with the number of fruit
- 938 objects estimated without detection and localization of objects, acting on a view of the entire image.
- 939 The model was modified and trained entirely on synthetic (tomato) fruit images and tested on
- natural images, with a 91% average test accuracy, R² of 0.90 and RMSE of 2.52 fruit/image on real
- images (downloaded from Google images; only 100 images used) and 93% on synthetic images. The
- algorithm was robust to varying degree of lighting conditions, occlusions and fruit overlaps. The
- 943 modified Inception-ResNet architecture achieved a higher fruit detection accuracy than the standard
 944 Inception-ResNet architecture, while a shallow CNN of four layers (2 convolution and 2 fully
- 945 connected layers) performed poorly (91.0, 70.0 and 11.6% accuracy, respectively).
- 946 Mureşan and Oltean (2018) used deep learning (a CNN with four convolutional layers) for fruit
- 947 classification (recognition) of 60 fruit categories, reporting a classification accuracy of 96.3% on a
- 948 test set. In this application fruit localisation is not required. The datasets ('Fruit-360') were released
- 949 by the authors, allowing comparative work to be undertaken by other researchers.
- 950 Koirala et al. (2019b) compared the detection results for a number of detection frameworks (Faster-
- 951 RCNN, SSD and YOLO) for mango fruit detection, and re-designed the YOLO architecture (to
- 952 'MangoYOLO') for speed and memory optimization for the mango fruit application. All models
- achieved F1 > 0.90 on an independent validation set of 512×512 pixels images with the highest score
- 954 (F1 of 0.97) from the MangoYOLO model. The superior result for MangoYOLO compared to the
- 955 deeper architecture of YOLOv3 demonstrates that it is not number of layers alone that defines the
- 956 performance, but also the design of the architecture. In MangoYOLO, Koirala et al. (2019b) merged
- 957 the information from early detection layers to that of the later detection layers.

958 2.8 Tree fruit yield estimation

959 **2.8.1** Can't see the forest for the trees?

960 Uses for machine vision detection of fruit in images of tree canopies include estimation of fruit
961 number per tree ('load'), in-field fruit sizing, and automated harvest. Estimation of fruit size together
962 with fruit number allows estimation of fruit weight ('yield') per orchard.

Fruit weight can be correlated to fruit lineal dimensions in many fruit. Such a relationship allows fruit weight to be estimated using machine vision, given a measure of camera to fruit distance, e.g. using a time of flight camera. For whole canopy images, as used in fruit load estimation, only a fraction of fruit in the image have entire outlines (i.e., not partially occluded), but that is sufficient to provide a size class distribution, assuming visible, outer canopy fruit are representative of all fruit on a tree (as achieved by (Wang et al. 2017b)).

- 969 Of course, a tree fruit load estimate relies on assessment of the total number of fruit per tree, not
- 970 the number of fruit visible in an image. With high accuracies reported for fruit detection in an image
- 971 using deep learning methods, as discussed earlier in section 8, research attention should now shift to
- 972 approaches to estimate total fruit per tree and per orchard.

973 2.8.2 Seeing all fruit

- 974 One approach to the estimation of fruit number per tree and orchard involves increasing the
- 975 number of viewpoints of images of the tree, in an attempt to visualise all fruit on the tree (akin to a
- 976 human estimate of fruit on tree, which involves walking around the tree). Payne et al. (2013)
- 977 reported a higher (R²) between image counts and human counts of total fruit on tree based on a
- 978 summation of counts from images taken on four sides of each canopy, compared to counts from
- 979 images of two or one side.
- 980 Multiple imaging of one canopy can result in over-estimation of fruit load by multiple counting of the
- 981 same fruit. This issue has been addressed by background removal or masking of fruits based on
- 982 depth or localizing fruit in 3D space (Wang et al. 2013), or use of stereo imaging with fruit
- 983 registration between frames from multiple viewpoints (Moonrinta et al. 2010). Wang et al. (2013)
- and Song et al. (2014) implemented an object (fruit) tracking algorithm with multiple viewpointimages of apple and pepper plants, respectively.
- 986 Stein et al. (2016) described a 'multiview' method in which 37 images of each side of mango tree 987 canopies were captured from a passing platform. Fruit detection was done using deep learning 988 (Faster R-CNN), inferring the instances of detected bounding-box as fruit counts as in (Bargoti and 989 Underwood 2017a). Fruit number per tree was estimated based on an epipolar projection approach 990 with fruit tracking using trajectory data (camera pose) provided by the navigation system and 991 association of fruit to individual trees achieved using a LiDAR mask of orchard trees. A R²=0.90 for 992 total fruit numbers for 16 trees (harvest count) with an error rate of only 1.36% for individual tree. A 993 slope ~1 was reported on actual fruit number per tree (for 18 'calibration' trees, as determined by 994 harvest). In this case, any double counting of fruit was balanced by non-detection (hidden fruit).
- 995 Similarly, Liu et al. (2018) used a FCN to segment fruit pixels followed by fruit tracking across
- 996 sequence of video frames. Fruit were then localized in 3D (to minimize errors of double counting
- 997 fruit on same tree and form other rows) using a Structure from Motion (SfM) algorithm. For fruit
- 998 counts against human count on images, L1 error of 203 and 322 and error mean of -0.2% and 3.3%

- and error standard deviation of 7.8% and 4.1% were reported for orange (daytime) and apple (night-time) images respectively
- 1001 However, these multiple view approaches are computationally complex and require precise object
- 1002 tracking algorithms. Tracking can be challenging in scenes with large number of similar objects
- 1003 (fruit), as present in a canopy scene. This can be an issue particularly for fruit clusters. Also, double
- 1004 counting of fruit will occur if the tracking of a fruit is lost due to fruit occlusion or sudden
- 1005 movements in the camera platform. Extra hardware (e.g., an inertial navigation system) is used to
- 1006 minimise such issues.

1007 2.8.3 Occluded fruit correction factor

- 1008 Another approach to provide a tree fruit load estimate from a count of fruit in a tree image involves 1009 calculation of a correction factor for occluded fruit. The simplest approach is to use a single factor
- 1010 per orchard. For example, Payne et al. (2013), Wang et al. (2013), Stein et al. (2016), Chen et al.
- 1011 (2017) and Koirala et al. (2019b) report use of the slope of the linear regression between the
- 1012 machine vision image count and harvest count for a set of calibration trees for fruit load estimation.
- 1013 This method relies on consistent tree architecture across the orchard (i.e., a single correction factor
- 1014 is applied to all trees). The factor must be established empirically for each orchard and may have to
- 1015 be repeated each season if canopy structure / fruit position in canopy changes. For example, for the
- 1016 five mango orchards considered, Koirala et al. (2019b) report that an orchard wide occluded fruit
- 1017 correction factor used in conjunction with MangoYOLO based estimates of fruit number per canopy
- 1018 image achieved fruit yield estimates between 4.6 to 15.2% of packhouse fruit count. The correction
- 1019 factor varied from 1.05 to 2.43 across the five orchards.
- 1020 Ideally, a correction factor could be established on an individual tree basis rather than for the whole
 1021 orchard block. For example, the number of totally occluded fruit may be proportional to the number
 1022 of partly occluded fruit.
- 1023 A per tree correction factor can be built into the model used to estimate fruit count per image. For 1024 example, Crtomir et al. (2012) trained an ANN using several input parameters (fruit area, canopy 1025 area and number of visible fruits on images) against actual harvest count for apple trees. The ANN model provided an improved yield estimation, at R^2 =0.83 (Golden Delicious) and R^2 =0.78 (Braeburn), 1026 1027 compared to a simple regression model (fruit count on images as the only input), at $R^2=0.73$ (Golden 1028 delicious) and R²=0.51 (Braeburn). Cheng et al. (2017b) used fruit and canopy features (number and 1029 area of fruit, area of fruit clusters, and leaf area per tree image) to train a back-propagation neural 1030 network (BPNN) model for prediction of apple yield (fruit weight per tree), reporting a R^2 of 0.75, 1031 RMSE of 2.5 kg /tree, for a crop near harvest maturity.

1032 2.8.4 Estimating fruit weight

As noted earlier, allometric relationships exist between fruit lineal dimensions and weight. Stajnko et al. (2009) employed traditional machine vision segmentation techniques in estimation of fruit number per canopy image using two images per tree and fruit size, using a size marker placed in the tree. Apple yield per tree (Y_t) was calculated based on Mitchell's (Mitchell 1986) equation, involving number and diameter of fruits.

- 1038 $Y_t = N \times a \times D^b$,
- where: Yt = yield per tree, N=number of fruits per tree, D=average diameter of fruit, a,b = constants
 depending on apple variety. For average yield per tree (kg) an R² of 0.96 between manual
- 1041 measurement and machine vision estimation was reported.

1042Črtomir et al. (2012) used an ANN (4-6-1 and 5-14-1 network architectures for varieties Golden1043Delicious and Braeburn respectively) to estimate apple fruit yield (kg per tree) from a single image1044per canopy. The ANN regressed the inputs of fruit segmentation, as described in Stajnko and Čmelik1045(2005) against weighted yield per tree. An R² of 0.69 and 0.61 and standard deviation of 2.83 and10462.55 kg/tree between forecasted and actual (harvest) yield per tree was achieved for Golden and1047Braeburn varieties respectively. This result was superior to that obtained using the method of1048Stajnko et al. (2009).

1049 Cheng et al. (2017b) used fruit and canopy features from images (fruit number estimated using a 1050 traditional colour segmentation method as described by Zhou et al. (2012), fruit area, fruit cluster 1051 area, and foliage leaf area) as input parameters to an ANN (Backpropagation Neural Network, BPNN) 1052 for yield (fruit weight per tree) prediction. Two separate ANN architectures were applied: a 4-12-1 1053 architecture for the early period of fruiting and a 4-11-1 architecture for the ripening period. Both 1054 models performed well for apple prediction with the later period model slightly better than the early 1055 period model. The later period model achieved an R² of 0.82 and RMSE of 2.31 kg/tree on a Gala 1056 apple test set from the season of model development, and an R² of 0.75 and RMSE of 2.54 kg/tree 1057 for a next season test set. A BPNN model (4-10-1 architecture) on Pinova variety achieved an R² of 1058 0.88 and RMSE of 2.53 kg/tree on the test set.

- 1059 Other workers have reported relationships between yield (fruit number or fruit weight per tree and 1060 canopy characters based on remote (satellite) imagery such as spectral indices and canopy area. For 1061 example, Rahman et al. (2018) report use of an ANN regression model (10-5-1 architecture) between 1062 spectral indices and crown area against mango fruit yield. However, while canopy area and plant 1063 health will be linked to the potential to flower and for fruit set (i.e., the number of vegetative 1064 terminals on a mango canopy sets the upper limit for the number of panicles that can form), there 1065 can be high levels of flower and fruit drop. Therefore, such models will require annual 'calibration' 1066 to actual yield, to accommodation changes in flowering and fruit set conditions.
- 1067 Deep learning has yet to be applied to such data sets for estimation of fruit load per tree and per 1068 orchard. There are examples with other crops where deep learning models have been applied with 1069 more complex spatio-temporal data and remotely sensed multi-spectral images for yield (t/ha) 1070 prediction. Kuwata and Shibasaki (2015) used a Caffe based deep learning regression model 1071 (Gaussian Radial Basis Function) trained with remotely sensed data (satellite data, climate data and 1072 environmental metadata) to model corn crop yield estimation at a county level. Jiang et al. (2018) 1073 used weather and soil data as inputs to a Long Short-Term Memory (LSTM) model for corn yield at a 1074 county level. Similarly, You et al. (2017) used deep learning (CNN and LSTM) models trained on 1075 yearly yield and multi-spectral remote sensing data for real-time forecasting of soybean yield at a 1076 county level. The advantage of deep learning is the automatic extraction of meaningful information 1077 from the real-world raw data. There is a need to explore different deep learning architectures for

1078 their possible implementation in the precision agriculture field.

1079 **2.9 Conclusion and recommendations**

1080 The need for handcrafting of features and designing of feature descriptors is removed in an ANN 1081 through automated learning. In general, ANN model accuracy improves with number of model 1082 layers, but at the cost of increased computational complexity. Object detection frameworks have 1083 evolved under the selective pressure of a requirement for higher speed, moving from two-staged 1084 region-based detectors to single shot dense object detectors. Deep learning models are reported to 1085 outperform pixel-wise segmentation techniques involving traditional machine learning and 1086 shallower CNN and Neural networks in the task of fruit-on-plant detection. The availability of publicly available detection frameworks with pre-trained weights and large public datasets of
annotated images for training have made adopting deep learning relatively easy. Typically, with use
of some hundreds of training images, the models can be fine-tuned for use in a particular image
object detection task. Public datasets of annotated images of fruit on tree (and other agricultural
applications) should be created to facilitate algorithm comparison using the same image training,
validation and test sets.

1093 There will be many reports generated on the adoption of a deep learning framework for specific 1094 precision agricultural applications. Published research studies must present information on number 1095 of training samples and values of network parameters used for model optimization. For replication 1096 of a given study and for comparison/benchmarking, the source of the open-source codes/algorithms 1097 and the model parameters (IoU and NMS thresholds) must be reported. A recommendation is made 1098 for use of the F1 score as a common evaluation metric for performance measure of the prediction 1099 models. Studies should also include several test sets independent to training set for the evaluation 1100 of model robustness.

- 1101 However, while reports of performance of a given model have technical merit to achieve an
- application end, the research frontier in this field has now moved to creation of existing deep
- 1103 learning architecture to suit a particular application, in terms of either increased speed or accuracy.
- 1104 Recent application of deep learning methods to directly predict fruit numbers on images without the
- 1105 need for labour intensive data labelling (pixel-wise or bounding box method) is of practical
- 1106 importance. Given the accuracy of current frameworks in fruit detection, research focus should also
- 1107 shift towards accurate estimation of orchard fruit number, i.e., to a consideration of systems to
- avoid or adjust for occluded fruit. Reports of fruit load estimation should be validated using test sets
 from more than one season, one cultivar and one orchard (i.e., one canopy architecture). It is
- 1110 recommended that fruit load estimation be attempted using several deep learning approaches (e.g.,
- 1111 CNN detectors, deep regression and LSTM) involving several data sources (e.g., tree images, orchard
- 1112 metadata, weather and yield history).
- 1113 Precise fruit detection allows generation of yield maps, providing information on spatial variation on
- 1114 which to base agronomic decisions. Fruit load estimation is also useful to inform harvest resourcing
- and management, and marketing. Light weight deep learning models will also support robotic
- 1116 harvesting.
- 1117 Deep learning has been discussed in context of machine vision in this review, but the technique has
- relevance with other data types. The ability of deep learning methods to automatically extract
- 1119 useful information from multi-dimensional raw data (e.g., digital images, LIDAR data, multi spectral,
- 1120 remote sensing and satellite data, weather, climate and environmental data from several sensors)
- and its scalability with big data holds great promise for applications such as real-time yield
- estimation and forecasting, crop growth modelling and plant disease and pest modelling. Application
- development will depend on quality data encompassing a range of environmental conditions,
- 1124 cultivars and crop types and modalities (sensors and data types).

1125 Acknowledgement

- 1126 This work received funding support from the Federal Department of Agriculture and Water and
- 1127 Horticulture Innovation Australia (project ST15005, Multiscale monitoring of tropical fruit
- 1128 production). Anand Koirala and Zhenglin Wang acknowledge receipt of an Australian Regional
- 1129 Universities Network scholarship and a CQUniversity Early Career Fellowship, respectively.
- 1130

- 1131
- 1132
- 1133

1134 Chapter 3. Deep learning for real-time fruit detection and 1135 orchard fruit load estimation – benchmarking of 1136 'MangoYOLO'

- 1137 This chapter was published as a journal paper in then February 2019 edition of Precision Agriculture 1138 as:
- Koirala A, Wang Z, Walsh K, McCarthy C (2019b) Deep learning for real-time fruit detection and
 orchard fruit load estimation: benchmarking of '*MangoYOLO*'. Precision Agriculture
 doi:<u>https://doi.org/10.1007/s1119-019-09642-0</u>
- 1143 The dataset used in this publication was made publicly available at 1144 http://hdl.cqu.edu.au/10018/1261224.
- 1145 Responses to minor revisions as requested by the thesis examiners can be found in the Addendum 1146 section.
- 1147

1142

- 1148
- 1149
- 1150
- 1151
- 1151
- 1152
- 1153
- 1154
- 1155
- 1156
- 1157
- 1157
- 1158
- 1159
- 1160

1161 Abstract

1162 The performance of six existing deep learning architectures were compared for the task of detection

of mango fruit in images of tree canopies. Images of trees (n = 1515) from across five orchards were

acquired at night using a 5 Mega-pixel RGB digital camera and 720 W of LED flood lighting in a rig

1165 mounted on a farm utility vehicle operating at 6 km/h. The two stage deep learning architectures of

1166 Faster R-CNN(VGG) and Faster R-CNN(ZF), and the single stage techniques YOLOv3, YOLOv2,

1167 YOLOv2(tiny) and SSD were trained both with original resolution and 512×512 pixel versions of 1300

training tiles, while YOLOv3 was run only with 512×512 pixel images, giving a total of eleven models.

1169 A new architecture was also developed, based on features of YOLOv3 and YOLOv2(tiny), on the 1170 design criteria of accuracy and speed for the current application. This architecture, termed 1171 'MangoYOLO', was trained using: (i) the 1300 tile training set, (ii) the COCO dataset before training 1172 on the mango training set, and (iii) a daytime image training set of a previous publication, to create 1173 the MangoYOLO models 's', 'pt' and 'bu', respectively. Average Precision plateaued with use of 1174 around 400 training tiles. MangoYOLO(pt) achieved a F1 score of 0.968 and Average Precision of 1175 0.983 on a test set independent of the training set, outperforming other algorithms, with a detection 1176 speed of 8 ms per 512×512 pixel image tile while using just 833 Mb GPU memory per image (on a 1177 NVIDIA GeForce GTX 1070 Ti GPU) used for in-field application. The MangoYOLO model also 1178 outperformed other models in processing of full images, requiring just 70 ms per image (2048×2048 pixels) (i.e., capable of processing ~ 14 fps) with use of 4417 Mb of GPU memory. The model was 1179 1180 robust in use with images of other orchards, cultivars and lighting conditions. MangoYOLO(bu) 1181 achieved a F1 score of 0.89 on a day-time mango image dataset. With use of a correction factor 1182 estimated from the ratio of human count of fruit in images of the two sides of sample trees per 1183 orchard and a hand harvest count of all fruit on those trees, MangoYOLO(pt) achieved orchard fruit 1184 load estimates of between 4.6 and 15.2 % of packhouse fruit counts for the five orchards 1185 considered. The labelled images (1300 training, 130 validation and 300 test) of this study are 1186 available for comparative studies¹.

1187 Contribution:

1188 1189		• The first use of deep learning single stage detectors (YOLO and SSD) for fruit load estimation is reported.
1190 1191		• Comparison of single stage detectors to two stage detectors (such as Faster R-CNN) using common datasets.
1192 1193 1194 1195		• The YOLO architecture was optimised (as ' <i>MangoYOLO</i> ') for improved speed without sacrificing accuracy in mango fruit detection and benchmarked to other popular models on a night image set and a day image set collected using another camera and illumination system.
1196 1197		 Model robustness was demonstrated across orchards (growing conditions, cultivars), illumination condition and camera hardware.
1198		• An annotated image dataset is made available for benchmarking of new algorithms.
1199 1200		 The technology can be applied in real time for orchard fruit load estimation and automated harvesting.
1201	31	Introduction

1201 **3.1 Introduction**

1202 **3.1.1 The horticultural issue**

1203 Knowledge of tree fruit crop load and timing of maturation can guide agronomic treatments, labour 1204 resource management and support market planning. In Australia, mango harvest timing is based on 1205 heat sums from flowering and fruit dry matter content, assessed using handheld near infrared 1206 spectroscopy (Walsh and Wang 2018). For mango crop load estimation, the appropriate orchard 1207 yield metrics are fruit size distribution and fruit number, rather than tonnes per hectare, as fruit are 1208 typically sold at wholesale level as trays of uniform sized fruit and at retail level per piece of fruit, 1209 rather than on weight of fruit.

1210 In current practice, harvest forecasts of crop load for the Australian mango industry are based on1211 previous yield history and manual counting of fruit on trees. Harvest forecasts are required six weeks

1212 before actual harvest to be of practical use in guiding harvest resourcing decisions, i.e., as early as

- 1213 possible to allow for management decision making, but after stone (endocarp) hardening and the
- 1214 period of fruit drop. However, the current manual yield estimation technique is time consuming,
- labour intensive and inaccurate (Anderson et al. 2018). Indeed, given the level of variability in fruitload per tree even in a well-managed orchard, Anderson et al. (2018) has calculated that typically
- 1217 approximately 200 trees should be assessed per orchard management unit, for a 0.95 probability
- 1218 and a percentage error of 10% on fruit load estimation. This level of human based sampling effort is
- 1219 impractical.

1220 **3.1.2** Machine vision in the orchard

- Machine vision can be applied in-field for fruit load estimation, extending the application of this technology from its current use in the fruit pack-house. RGB camera images of tree canopies have been coupled to depth camera images to provide information on fruit size distribution within an orchard. For example, Wang et al. (2017b) employed a time-of-flight depth camera on the (6 km/h) moving platform used in the current study to estimate mango fruit lineal dimensions to a RMSE of 5 mm, and applied an allometric relationship to relate lineal dimensions to fruit weight and tray size category. Given knowledge of fruit size distribution, the remaining element for a practical yield
- 1228 estimation technique for mango is orchard fruit load estimation.
- Mango fruit hang on long stalks derived from the panicle, making imaging from the orchard interrow more practical than imaging from above (e.g., by drone). Mounting of imaging gear to a farm
 vehicle allows for easy deployment on farm, however, imaging of a whole farm is not a trivial task.
 For example, driving a medium sized farm with 30,000 trees at 4 m spacing involves 120 km, or 240
 km if the imaging rig faces one direction only. Imaging in two directions from a farm vehicle that is
 traversing the farm for other tasks, such as a spray rig, is a possibility. Further, night imaging is not a
 constraint in the Australian mango industry, with several operations currently conducted at night,
- 1236 from spraying to harvesting, to avoid day time heat.
- 1237 In orchard scenes, common challenges for machine vision detection of fruit include variation in 1238 illumination, fruit occlusion, fruit orientation and similarities in colour and texture of fruit and foliage 1239 (Gongal et al. 2015; Jimenez et al. 2000; Kamilaris and Prenafeta-Boldú 2018; Payne and Walsh 2014; 1240 Syal et al. 2013). To minimize illumination issues inherent in daylight photography, a number of 1241 researchers have advocated night imaging with artificial illumination (e.g., Qureshi et al. (2017)) or 1242 imaging under a shade structure (Gongal et al. 2015). Bargoti and Underwood (2017a) reported 1243 imaging of mango orchards in day light hours using high intensity strobe lighting and very short 1244 exposure times to avoid issues in direct sun lighting, but this required a camera capable of 1245 microsecond exposures and use of high intensity Xenon strobe lights, increasing deployment cost 1246 and complexity.
- 1247 During the past decade there have been many reports of object classification in orchard scenes 1248 (Gongal et al. 2015) involving manually defined ('handcrafted') parameters for features such as 1249 intensity, colour space, shape and texture, or Histogram of Oriented Gradients (HOG), Local Binary 1250 Patterns (LBP) or Haar-like features (Gongal et al. 2015). For in-field detection of mango fruit, Payne 1251 et al. (2013) applied a colour based segmentation technique followed by blob detection, while Payne 1252 et al. (2014) and Qureshi et al. (2017) placed emphasis on texture analysis to achieve an improved fruit detection rate. Nanaa et al. (2014) detected mango fruit based on elliptical shape fitting and 1253 1254 Kadir et al. (2015) applied texture analysis to define the fruit edge, followed by morphological 1255 operations on binary image and ellipse fitting using Randomized Hough Transform (RHT) technique 1256 for detection of mango fruit in clusters. Detection errors were associated with leaves of similar

shape to fruit and occluded fruit. However, these 'handcrafted' approaches often fail to generalize
to other conditions (cultivars, growing conditions, lighting conditions), as reported for the mango
application by Payne et al. (2014) and Qureshi et al. (2017).

1260 A generalised multi-scale feature (unsupervised) learning approach based on a sparse autoencorder 1261 and a backpropagation neural network was applied by Hung et al. (2015) to estimate almond and 1262 apple fruit load (with F1 scores of 84.8 and 87.3, respectively). More recently, deep neural networks 1263 have been successfully used in a range of machine vision applications, including medical, 1264 automotive, aerospace, defence, consumer electronics and natural language processing. Sa et al. 1265 (2016) utilised the deep learning framework of Faster Regional-Convolutional Neural Network 1266 (Faster R-CNN) (Ren et al. 2015) with Oxford Visual Geometry Group network (VGGNet) (Simonyan 1267 and Zisserman 2014) for detection of various fruits through transfer learning. In the transfer 1268 learning technique used, a pre-trained model was further trained using small numbers of images 1269 (ranging from 43 to 136), primarily sourced from the internet, with F1 scores of 0.848, 0.948, 0.938, 1270 0.932, 0.942, 0.915 and 0.828 achieved for rockmelon, strawberry, apple, avocado, mango, orange 1271 and sweet pepper, respectively. In the following year, Bargoti and Underwood (2017a) used Faster 1272 R-CNN to detect fruit in day-time images of real orchard scenes illuminated with high intensity 1273 strobe lights, reporting higher F1 scores with the deeper (i.e., more layers) VGGNet than when using 1274 the Zeiler and Fergus network (ZFNet) (Zeiler and Fergus 2014) (0.904, 0.908 and 0.775 compared to 1275 0.892, 0.8876 and 0.726, for apple, mango and almond fruit detection, respectively). These F1 scores 1276 were higher than that achieved using pixel-wise fruit segmentation with a Convolution Neural 1277 Network (CNN) followed by Watershed Segmentation (WS) and blob detection (F1 scores 0.861 and 1278 0.836 for apple and mango, respectively) (Bargoti and Underwood 2017b) but lower than those 1279 reported by (Sa et al. 2016) for apple and mango detection, presumably as images of real orchard 1280 scene present more challenges (such as varying illumination, large number of fruits per image, low 1281 pixel count per fruit) for training and testing fruit detection models than for close-up images 1282 collected from a greenhouse and Google search. More recently, Wang et al. (2018b) reported on the 1283 use of the Faster R-CNN(VGG) deep learning architecture for task of segmenting and counting mango 1284 panicles in canopy images, and compared the result to a panicle associated pixel count (but not 1285 panicle count) achieved using a traditional method based on handcrafted colour based features.

1286 **3.1.3** Aim

Deep learning architectures appear to hold value for fruit load estimation from orchard canopy images. However, in reports to date (e.g., Bargoti and Underwood (2017a), Sa et al. (2016) and Wang et al. (2018b)), image processing has utilised high performance computing (HPC) resources over a period of days following the actual imaging event. For practical application, an architecture was sought that was sufficiently light (i.e., fewer convolutional and detection layers) to allow deployment on readily available PCs, to allow for real time application in orchard, while maintaining accuracy.

1293 The 'need for speed' is illustrated in the following calculations. For a medium sized farm of 30,000 1294 trees there are 60,000 (dual-view) 5 Mega pixels images that are currently uploaded by satellite link 1295 by farm management for image processing on a HPC resource. Local processing would avoid the 1296 need for data transfer off-farm. If processing took 1 second per image, the 60,000 images would 1297 consume 17 h of computing resources, either in the farm office or during imaging. At the typical 1298 speed of 6 km/h of a spray rig (a potential carrier for an in-field tree imaging system) and a tree 1299 spacing of 4 m, there is 2.4 sec between trees, but only a fraction of this time is available for image 1300 processing, especially for systems capturing views in two directions (both rows), or employing 1301 multiple images per tree or video for fruit tracking between images (e.g., Stein et al. (2016). Further, 1302 real-time detection of fruits is a precursor to use in an autonomous harvesting system.

1303 It is difficult to assess the performance of deep learning architectures across published studies

- 1304 because of difference in datasets, computing hardware, network parameters and performance
- 1305 metrics. The current study undertook a comparison using common datasets of the deep learning
- architectures employed to date in fruit load estimation (i.e., Faster R-CNN with VGG and ZF, twostage detectors), and added consideration of current 'state of art' single stage detector
- 1308 architectures, which have not yet been reported in context of fruit detection to our knowledge. The
- 1309 one-stage detectors offer increased speed of operation. Further, reported work to date has been
- 1310 based on use of 'off the shelf' architectures that were trained for fruit detection, but were created
- 1311 for a purpose other than the current application (mango fruit load estimation). Therefore, a new
- 1312 deep learning architecture was proposed, based on the single stage architectures, to optimize
- 1313 memory and speed of detection for the task of mango fruit detection, and performance
- 1314 benchmarked to the existing 'off-the-shelf' algorithms.
- 1315 Compared with previous published work, the current work contributes to the development of a 1316 solution for fruit detection and yield estimation by examining the hypothesis that a single stage deep 1317 learning detector is faster than a two-stage detector with similar accuracy, and that a deep learning 1318 architecture can be optimized for speed and accuracy for a given application (mango fruit detection 1319 in this study). A target processing speed of 500 ms on a field deployable computer was set to allow for in-field image processing. The ultimate goal is provision of a yield estimate to growers to within 1320 1321 15% of fruit harvest count (i.e., information useful to farm management; Walsh and Wand, 2018). 1322 This aim requires accurate assessment of both fruit seen in images (i.e., a high F1 score) and the 1323 proportion of fully occluded fruit. Fruit load estimation of five orchards is estimated to guage utility of the estimation technique. 1324

1325 **3.1.4 Deep Learning – object detection frameworks**

- A brief review of deep learning methods is presented to place the current work in context. Deep learning methods have been reported to outperform traditional approaches in the majority of object segmentation and classification tasks in agriculture, with the automatic feature extraction capabilities of deep learning architectures reported to solve complex problems more easily, with greater accuracy and better generalization than traditional approaches (Kamilaris and Prenafeta-
- Boldú 2018). While a greater number of training image is required for the deep learning methods
- 1332 than those based on handcrafted features, the need for labour intensive and expert knowledge for
- 1333 feature extraction is avoided (Bargoti and Underwood 2017a; Kamilaris and Prenafeta-Boldú 2018) .
- 1334 The training of the deep learning frameworks involves optimization of a cost/loss/objective function 1335 (combination of classification, confidence and box regression losses) for minimum error in object 1336 classification and localization. As deep learning architectures have millions of parameters to be 1337 optimized, these models are commonly trained on large datasets, such as ImageNet (450k images in 1338 200 classes, (Deng et al. 2009), PASCAL VOC (12k images in 20 classes, (Everingham et al. 2010) and 1339 COCO (120k images in 80 classes, (Lin et al. 2014). These online datasets provide labelled data 1340 (images and annotation files for different object classes) free for training and benchmarking object 1341 detection models. While these sets do not contain mango orchard images, deep-learning models can 1342 be reused in new applications, with the learned features transferred through transfer learning. 1343 Transfer learning is a method involving use of a relatively small number of images for 're-training' or 1344 'fine tuning' the knowledge (convolutional weights) of a model trained on a very large dataset, to 1345 create a new model.
- 1346 Current state of the art object detection frameworks include Faster R-CNN (Ren et al. 2015), You
 1347 Only Look Once (YOLO) (Redmon and Farhadi 2018), Single Shot MultiBox Detector (SSD) (Liu et al.

2016), RetinaNet (Lin et al. 2017b) and Mask R-CNN (He et al. 2017). There is a fast pace of
developments in this field. For example, the recent review and survey of deep learning in agriculture
by Kamilaris and Prenafeta-Boldú (2018) did not report on use of single stage detectors (SSD and
YOLO).

1352 Two stage detectors like Faster R-CNN have a region proposal network as a first stage which 1353 proposes possible areas likely to contain objects (regions of interest, ROI), followed by a feature 1354 extractor (a base CNN such as VGGNet or ZFNet) that extracts visual features from the ROIs to 1355 determine the presence of object. A second stage undertakes classification and bounding box 1356 regression. This pipeline is generally considered to be too slow for tasks requiring real-time 1357 processing, e.g., on a GPU (GeForce GTX Titan X) hardware, Redmon and Farhadi (2017) reported 1358 processing at 7, 46 and 91 frames per second using Faster R-CNN with VGG-16, SSD-300 and 1359 YOLOv2-288, respectively. In a single stage detector, there is no region proposal stage as the 1360 prediction is made directly on the input image in single forward pass through the CNN. A single stage 1361 detector like YOLO converts the input image to a vector of scores and produces coordinates for the 1362 predicted boxes with a single CNN, making a very fast detector.

- 1363 A general object detection framework involves: (i) input image pre-processing (resizing, normalizing,
- 1364 colorspace change etc.); (ii) detection of objects; (iii) placement of a bounding box (usually
- 1365 rectangle) around the detected objects for the task of object localization; (iv) a calculation of class
- 1366 specific confidence score for each detection; and (v) post processing filtering of detections on the
- 1367 basis of a confidence score criterion, with merging of multiple detections of one object using the
- 1368 non-maximum suppression (NMS) technique to supress detections based on an overlap threshold.
- 1369 In the current study, three different state-of-the-art deep learning architectures for object detection1370 (Faster R-CNN, SSD, and YOLO) were trained and tested for mango fruit detection.

1371 Faster R-CNN: Faster R-CNN (Ren et al. 2015) is a two stage complete object detection framework 1372 which replaced the selective search method of R-CNN (Girshick et al. 2014) with a Region Proposal 1373 Network (RPN), with the advantage of higher detection speed. RPN slides a 3x3 window across the 1374 final feature map and at each window location considers k different anchor boxes centred on the 1375 location to generate possible region proposals. Each region proposal consists of an objectness score 1376 for that region as well as the co-ordinates of the boxes. The regional proposals are filtered based on 1377 objectness threshold and passed to what is essentially a Fast R-CNN (Girshick 2015) for object 1378 detection. In general, Faster R-CNN can be regarded as an RPN on the top of Fast R-CNN. Faster R-1379 CNN predicts bounding boxes using handpicked anchors (k=9) composed of three scales and three 1380 aspect ratios and employs one detection layer. The base CNNs ZFNet and VGGNet (VGG-16) have 7 1381 and 16 layers respectively.

1382 SSD: SSD (Liu et al. 2016) is a one stage complete object detection framework. Unlike Faster R-CNN 1383 which performs region proposal and classification separately, SSD simultaneously predicts the object 1384 class and places a bounding box on the image. An input image is fed through a series of layers in the 1385 base CNN, generating different sets of feature maps at different scales. For each feature map 1386 position, a 3x3 convolutional filter is used to assess a set of default anchor boxes referred to as 1387 'priors'. Boxes are then drawn and classified at every single position in the image and at several 1388 scales, generating a very large number of negative examples and few positives. To address the class 1389 imbalance problem, SSD uses a technique called 'hard negative mining' to balance the positive and 1390 negative classes during training, keeping the negatives to positives ratio to 3:1 by using only the 1391 subset of negatives with highest training loss.

1392 YOLO: YOLOv2 (Redmon and Farhadi 2017), a single stage complete object detection framework, is 1393 an improvement on YOLO (Redmon et al. 2016) making it better, faster and stronger. Class 1394 probabilities, objectness scores, and bounding box coordinates are predicted from the final feature 1395 map in one evaluation (forward pass) making it one of the fastest object detection methods. A pass-1396 through layer is used that concatenates features from the higher resolution 26×26 layer to a 13×13 1397 layer. This process may have an advantage in small object detection involving finer grained features 1398 (Redmon and Farhadi 2017). The backbone feature extractor for YOLOv2 is the Darknet-19 1399 framework which has 19 convolution layers, mostly using 3×3 filters similar to VGGNet. YOLOv2 has 1400 22 convolutional layers and 1 detection layer. YOLOv2(tiny) has a small architecture, with just 9 1401 convolutional layers, 6 pooling layers, and 1 detection layer, sacrificing accuracy for speed (Redmon 1402 2018).

1403 YOLOv3 (Redmon and Farhadi 2018) is based on Darknet-53 and is an improvement over YOLOv2 1404 (Redmon and Farhadi 2017). Darknet (Redmon 2018) is an open source deep learning framework 1405 written in C and CUDA. Darknet-53 is a new feature extractor (53 convolution layers), improving on 1406 Darknet-19 by addition of successive 3×3 and 1×1 convolutional layers with skip connections similar 1407 to ResNet (He et al. 2016). Like SSD, YOLOv3 uses the concept of feature pyramids (Lin et al. 2017a) 1408 to extract features at three different scales for box predictions. In order to capture more meaningful 1409 and fine-grained information, the feature maps from lower layers are merged with up-sampled 1410 feature map from higher layers and processed further. Like Faster R-CNN, YOLOv3 uses logistic 1411 regression to predict objectness score of bounding boxes but only one bounding box anchor is 1412 assigned per ground truth object. YOLOv3 employs 75 convolution layers and 3 detection layers.

1413 **3.2 Materials and Methods**

1414 **3.2.1 Orchard sites**

1415 Images were acquired of all trees in each of five mango (*Mangifera indica*) orchards (farm 1416 management units) located in Queensland, Australia, on December 7 and 8, 2017 (Table 3-1). The 1417 orchard E set represented only a part of an orchard. Imaging occurred at least six weeks before 1418 harvest, but after the 'stone hardening' stage (after which stage fruit drop is typically small). The 1419 orchards varied in in cultivar and canopy structure.

1420	Table 3-1. Description of orchards used for imaging. Sample trees refers to trees that were manually
1421	harvested for a ground truth fruit load estimate (# Harvested fruit).

Location	Orchard	Cultivar	Row spacing	Tree spacing	Tree height	Canopy width	# Trees	# Sample	# Harvested
lat, long	Orenard	Cultival	(m)	(m)	(m)	(m)	imaged	trees	fruit
-25.144, 152.377	А	Calypso	9.5	4	3	4	494	17	3519
-25.144, 152.377	В	Calypso	10.5	4	4.5	4	121	6	1676
-25.144, 152.377	С	Calypso	12	4	4	4	256	12	1784
-25.162 152.351	D	Honey Gold	8	4	3	2.8	566	18	1302
-25.205, 152.284	E	R2E2	9	5	3.3	2.8	78	18	729

1422

1423 **3.2.2 Field imaging hardware**

1424 The field imaging rig (Fig. 3-1) consisted of an aluminium frame (1×1 m) with combination of LED

bars (SCA, 126W, 5400 Lumens) and LED flood lamps (Calibre, 35W, 2500 Lumens) totalling 720 W,

1426 powered with a 24 V DC lead acid battery. Reflectors were removed from the LED bars to improve

1427 light spread. A Basler acA2440-75um machine vision camera (75 fps, 5 Mp, 2448×2048 pixels) with a

1428 Goyo (GM10HR30518MCN) lens (5 mm focal length) was operated at FStop 3, gain 5 and exposure 1429 2.5 ms for RGB imaging. The rig also carried a Canon EOS750D with 10-22 mm lens and a Kinect RGB-1430 D camera, as described in (Wang et al. 2018b). Camera colour calibration was not performed. Images of tree canopies were captured at night with a typical camera to canopy distance of 2 m. The rig was 1431 1432 mounted to the tray of a farm utility vehicle and operated after sunset, with the vehicle driven at a 1433 speed of about 6 km/hr. The use of night imaging allows for consistent illumination conditions 1434 compared to the variation experienced in daylight hours due to sun angle and weather conditions 1435 (Payne et al. 2013). The camera was triggered automatically, with reference to the pre-acquired 1436 GNSS position of trees. As a one-off task, every tree was geo-located to an accuracy of within 2 cm 1437 using a Leica GS-14 unit operating on a Continuously Operating Reference Stations (CORS) network. 1438 The receiver was mounted to a mast above average canopy height. Imaging of an orchard consisted 1439 of collection of two images per tree, imaged from opposite sides of each tree (termed 'dual-view' 1440 images). The system has been previously reported in (Wang et al. 2018b) and evolved from the 1441 systems used by Payne et al. (2013), Qureshi et al. (2017) and Underwood et al. (2018).



1442

Figure 3-1. Lighting and imaging camera rig mounted on a farm utility vehicle, operated at 6 km/h (left panel), with components of LED floodlights, RGB camera and Time of Flight camera (right panel).

1445 **3.2.3** Computing hardware

1446Model training and testing was implemented on the CQUniversity High Performance Computing1447(HPC) facility graphics node with following specifications: Intel® Xeon® Gold 6126 (12 cores,

1448 2600MHz base clock) CPU, NVIDIA[®] Tesla[®] P100 (16 GB Memory, 1328 MHz base clock, 3584 CUDA

1449 cores) GPU. Red Hat Enterprise Linux Server 7.4 (Maipo) and 384GB RAM. CUDA v9.0, cuDNN v7.1.1,
1450 OpenCV v3.4.0, Python v2.7.14, GCC v4.8.5.

For in-field use, a Nuvo-6108GC industrial-grade computer with following specifications was used:
Intel[®] Core[™] i7-6700TE CPU @ 2.40 GHz, 32 GB RAM, NVIDIA GeForce GTX 1070 Ti GPU (1607 MHz
GPU clock) with 8 GB dedicated memory (2002 MHz memory clock), 64-bit Windows 10 Pro, CUDA
v9.1, cuDNN v7.0.5, OpenCV v3.4.0.

1455 **3.2.4 Image pre-processing**

All object detection methods re-size the input image to a specific resolution ('network resolution')for training. The feature extractors (base CNN) used by deep learning frameworks usually require a

- square input resolution. In its original configuration, Faster R-CNN will resize the input image such
- 1459 that the shorter dimension of height or width is resized to 600 pixels, while keeping the image
- aspect ratio unchanged. SSD-300 and YOLO resize input images to 300x300 pixels and 416x416
- 1461 pixels, respectively. Network resolution can be increased to accept larger input images (e.g.,
- 1462 2048x2048 pixels), but at the cost of increased memory and computation requirement. The resizing
- 1463 of Basler 2448x2048 pixel images to 717x600 pixels for use in Faster R-CNN training resulted in a
- 1464 decrease in typical fruit image size from ~ 16x16 pixels to ~ 7x7 pixels.
- Network stride, the factor by which network scales down the input image to its final feature map, is 1466 16 and 32 in Faster R-CNN (for ZF and VGG) and YOLO, respectively. With a stride of 16, a 16x16 pixel 1467 image corresponds to one pixel in the final feature map. To have object size greater than the stride 1468 size on the final feature map, 2448x2048 pixel images were split into tiles (sub-images) of 612x512 1469 pixels with a 4x4 grid, maintaining the original image aspect ratio. This allowed for training of 1470 different detection frameworks with similar network resolution (around 512x512 pixels) and object 1471 size, while reducing memory requirement in training.
- 1472 Training object detection models requires labelled data, i.e., the class-label and the position (co-
- 1473 ordinates) of all ground truth bounding boxes in training images. While labelling is a manual and
- 1474 labour-intensive process, annotation (drawing of ground truth bounding boxes) was easier on tiles
- 1475 than on the full image, as the lower number of fruits on a tile compared to a full image reduces
- 1476 chances of human error. The graphical image annotation tool *labelImg*
- 1477 (<u>https://github.com/tzutalin/labelImg</u>: accessed 15/08/2017) was used to hand label all the ground
 1478 truth bounding boxes, with annotation files saved in PASCAL VOC format.

1479 3.2.5 Image sets

The different object detection frameworks, i.e., Faster R-CNN, SSD and YOLO, require different data formats for processing image data during training and testing. Datasets and directories were structured similar to the PASCAL VOC dataset (Everingham et al. 2010), avoiding the need to change scripts, with the detection frameworks parsing PASCAL VOC annotations into their format. For each processed image, an XML annotation file (filename = image name) was created containing the image attributes (name, width, height), the object attributes (class name, object bounding box coordinates.

1400 Orumates.

1487 Tiles of 612×512 pixels were randomly drawn from images from three rows of orchard A. Tiles with 1488 no fruit were excluded. Training was undertaken using a set of 1300 tiles, with validation undertaken 1489 on a further 130 tiles from the same set of trees (Table 3-2). Test set 1 consisted of 300 tiles from 1490 three different rows of the same orchard and was split into three subsets containing 100 tiles each. 1491 Subset L (Low complexity) contained tiles with well separated fruits. Subset M (Medium complexity) 1492 contained tiles with some clustered (partially occluded) fruits. Subset H (High complexity) contained 1493 tiles with many fruits in clusters. Test set 2 consisted of images 6 -18 trees in each of five orchards (A 1494 to E, Table 3-2). The fruit load of these trees was manually assessed at harvest. There was no 1495 overlap of the orchard A prediction set with images used in the training and validation sets. The 1496 trees of test set 2-A were imaged with Canon and Kinect cameras, creating two further image test 1497 sets (2A-can and 2A-kin, respectively). Finally, the training and validation set images used by Bargoti 1498 and Underwood (2017a), accessed from https://data.acfr.usyd.edu.au/ag/treecrops/2016-1499 multifruit/, were employed (Train set 2-bu and Test set 3-bu, respectively).

1500Table 3-2. Description of image tile sets used in training, validation (model tuning) and testing. The training,1501validation and test set 1 were from images of trees of two rows from within orchard A (cultivar Calypso).

Suffixes 'can' and 'kin' refers to Canon and Kinect camera images, while 'bu' refers to an image set fromBargoti and Underwood (2017a).

	Number		Total #	
Dataset names	of images or tiles	Image size (pixels)	fruits in images	Cultivar
Train set 1	1300	612x512	11820	Calypso
Validation set 1	130	612x512	861	Calypso
Test set 1 All	300	612x512	2600	Calypso
Test set 1 Low	100	612x512	341	Calypso
Test set 1 Medium	100	612x512	636	Calypso
Test set 1 High	100	612x512	1623	Calypso
Test set 2A	34	2448×2048	2151	Calypso
Test set 2B	12	2448×2048	964	Calypso
Test set 2C	24	2448×2048	842	Calypso
Test set 2D	36	2448×2048	1229	Honey Gold
Test set 2E	36	2448×2048	552	R2E2
Test set 2A-can	34	6000×4000	2137	Calypso
Test set 2A-kin	34	1920×1080	1746	Calypso
Train set 2-bu	1154	500×500	7065	Calypso
Test set 3-bu	270	500×500	947	Calypso

1504

Trained models were also compared in terms of prediction results for sets of 6 -18 trees in each of five orchards (Test set 2, Table 3-2), for which fruit load was manually assessed at harvest. Models trained using images from Orchard A (Train set 1) were compared in terms of prediction results for the image test sets from different orchards (differing in growing conditions and cultivars, Test set 2 A-E) and from different cameras (Test set 2A, 2A-can and 2A-kin).

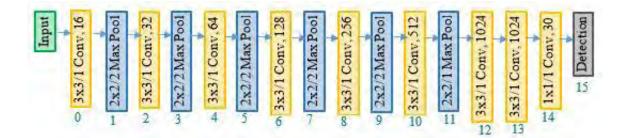
1510 Further benchmarking of the model developed in the current study was undertaken by comparison 1511 to the Faster R-CNN(VGG) and Faster R-CNN(ZF) models used by Bargoti and Underwood (2017a), 1512 employing their training and test sets (Train set 2-bu and Test set 3-bu; Table 3-2). These images 1513 were acquired of trees in orchard A in the season preceding imaging undertaken for the current 1514 study, and were captured in daytime using a 20 Mp Prosilica GT3300c camera and Xenon strobe 1515 lamps with a very short exposure time to reduce the effect of sunlight. The images are visually more 1516 complex and challenging compared to the night acquired images of the current study, because the 1517 images are relatively under-exposed, and the background is illuminated.

1518 **3.2.6 Deep learning architectures**

1519 3.2.6.1 Architectures used

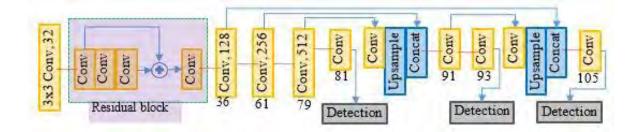
1520 Open source codes of the deep learning algorithms were downloaded from their official repositories through the links provided by Redmon et al. (2016) for YOLO, by Liu et al. (2016) for SSD, and by Ren 1521 1522 et al. (2015) for Faster R-CNN. As a general naming convention, deep learning model names are 1523 suffixed with a number representing the input image resolution of the network. In the current paper, 1524 any model names with suffix -original refers to a model used with the default network resolution, as 1525 available in the official repositories. The original arhitectures require input of 416×416 and 300×300 1526 pixels for YOLO and SSD respectively. Faster R-CNN on other hand resizes and rescales the input 1527 image to make the shorter side equal to 600 pixels (e.g., 612×512 pixel input image gets resized to

- 1528 600×717 pixels). The suffix -512 on a model name refers to the models whose network resolution1529 was changed to 512×512.
- 1530 The algorithms implemented in this study are sourced from open source code available from online
- repositories. The YOLO repository is constantly maintained, with more features added (e.g., object
- 1532 tracking) and frequent code optimizations for better speed and accuracy. Performance (speed and
- 1533 memory consumption) results for YOLO variants in HPC and Nuvo hardware are presented for the
- 1534 code implementation of 12/2018.
- 1535 Redmon and Farhadi (2018) note that the YOLO object detection framework allows for a trade-off
- between speed and accuracy. Network resolution can be changed to detect objects on different
- 1537 input image resolutions without the need for further training, and the heavy data augmentation
- 1538 (modification of training image hue, saturation, rotation, jitter, multiscale etc.) available with YOLO
- allows for the training of a robust model with a lower number of training images. YOLOv2(tiny) (Fig.
- 1540 3-2) is reported to have the highest speed and lowest accuracy of the YOLO variants. According to
- 1541 Redmon and Farhadi (2018), YOLOv3 is deeper (i.e., has more convolutional layers; Fig. 3-3) than the
- 1542 previous YOLO versions, but is as accurate and 3.0 and 3.8 times faster than SSD and RetinaNet,
- 1543 respectively.



1544

1545 Figure 3-2. Block diagram of architecture YOLOv2(tiny)



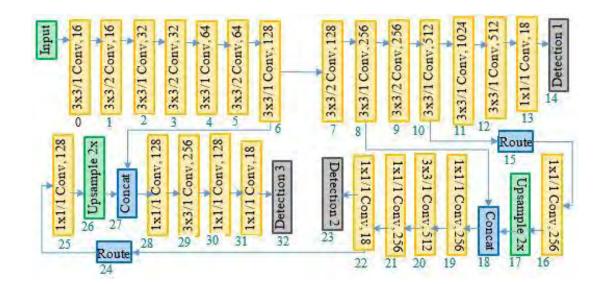
1546

1547 Figure 3-3. Block diagram of architecture YOLOv3

1548

1549 3.2.6.2 YOLO re-design

An attempt was made to optimize a model for speed, memory requirement and accuracy through a re-design of the YOLO framework, to take advantages of both YOLOv2(tiny) (fewer layers and higher speed) (Fig. 3-2) and YOLOv3 (multiple detection layers and high accuracy) (Fig. 3-3). The design of *MangoYOLO-512* incorporated 33 layers (Fig. 3-4), in comparison to 106 layers in YOLOv3 (Fig. 3-3) and 16 in YOLOv2(tiny) (Fig. 3-2).



1555

1556

1557 Figure 3-4. Block diagram of architecture of *MangoYOLO*

1558 The main rationale behind the modifications was to allow detection on multiple feature maps from 1559 different layers of the network on the premise that this would allow for accurate detection of mango 1560 fruit even with a reduced number of layers in the network. Similarly, it was reasoned that features 1561 from the early stages (layers) would assist detection of smaller and darker fruit whose pixel 1562 information could be lost when passed through a large number of layers in a deeper network. 1563 Moreover, the decrease in the number of layers was expected to result in decreased 1564 computation/detection time. MangoYOLO (Fig. 3-4) (33 layers) was created by modification of 1565 YOLOv2(tiny) as follows:

1566 1567 1568 1569	•	All 6 max-pooling layers in YOLOv2(tiny) were replaced with convolution layers as implemented by YOLOv3. Convolution layers are considered to have the same effect as pooling layers but are considered computationally more efficient (Springenberg et al. 2014).
1570 1571	•	In Tiny YOLOv2, layer 13 (with 1024 filters) is a replica of layer 12 (Fig. 3-2). This layer was removed, reducing computation time with no noticeable change in performance.
1572 1573 1574	•	To capture fine-grained information, feature maps from intermediate layers (6, 8 and 10) were further processed by up-sampling to merge with feature maps of different resolutions followed by four convolution layers, before detection (Fig. 3-4).
1575 1576	•	The first detection was implemented at layer 42 in <i>MangoYOLO</i> , intermediate between the positions implemented in YOLOv3, YOLOv2(tiny) (at layer 82 and 15, respectively).
1577 1578	•	Total depth was 33 layers, less than one-third of YOLOv3 (107 layers), and so involves less computation resulting in an expected improved detection speed.
1579 1580 1581 1582 1583	•	Unlike YOLOv3 that uses residual block (successive 3×3 and 1×1 convolutional layer with skip connections similar to ResNet) (Fig. 3-3), the <i>MangoYOLO</i> architecture involves convolution layers as in YOLOv2(tiny) (Fig. 3-4). A residual block is useful for resolving the 'vanishing gradient' problem in training of deeper networks (He et al. 2016). This implementation was not necessary for shallower networks like <i>MangoYOLO</i> .
1584 1585	•	As in YOLOv3 implementation, detections in <i>MangoYOLO</i> were made at three scales on feature maps of 16×16, 32×32, 64×64 pixels for input images size 512×512 pixels, which

1586 were appropriate for the mango image dataset (average fruit size 43×48 pixels and 1587 minimum fruit size 16×15 pixels in the Train set 1) under current study (Table 3-2).

1588 **3.2.7 Model training**

1589 *3.2.7.1 Number of training images*

1590 The number of training images required by deep learning models largely depends on the visual 1591 complexity of the images in training and test datasets, the network architecture, image 1592 augmentation techniques and machine learning parameters of the network. Similar to (Sa et al. 1593 2016) and (Bargoti and Underwood 2017a), an experiment was performed to provide a guide to the 1594 number of training images required for training of a deep learning architectures, with default 1595 network parameters. The training set of 1300 tiles was sampled randomly for subsets of 10, 50, 100, 1596 200, 400, 600, 800, and 1300 tiles, with selection of 100 and 600 subsets repeated three times for an 1597 estimate of the impact of sampling variation. MangoYOLO(s) models were created based on training 1598 with each subset, with optimisation on the number of iterations in each case based on average 1599 training loss following https://github.com/AlexeyAB/darknet#when-should-i-stop-training.

1600 3.2.7.2 Pretraining

- 1601 All models but *MangoYOLO(s)* and *(sbu)* were initialized with pre-trained models/weights (referred
- to as transfer learning) for training. *MangoYOLO(s)* was trained from scratch (random weight
- 1603 initialization of convolution filters) as a custom designed CNN architecture with no existing pre-
- trained weights. Pre-trained weights for initializing *MangoYOLO(pt)* and (*ptbu*) were obtained by
- 1605 training *MangoYOLO(s)* and (*sbu*), respectively, on the COCO dataset (following the instruction from
- 1606 https://pjreddie.com/darknet/yolo) for 120K iterations. VOC 2007 pre-trained Caffe models (ZF and
- 1607 VGG16) were downloaded for Faster R-CNN by running the fetch_fsater_rcnn_models.sh script
- 1608 provided on the repository (<u>https://github.com/rbgirshick/py-faster-rcnn</u>, accessed 21/02/2018).
- 1609 The ImageNet Large Scale Visual Recognition Competition (ILSVRC) pre-trained VGG16 model was 1610 downloaded for SSD from the link provided in the official repository
- 1611 (https://github.com/weiliu89/caffe/tree/ssd, accessed 16/03/2018). ImageNet pre-trained
- 1612 convolutional weights for YOLOv2 (Darknet-19_448.conv.23) and YOLOv3 (darknet53.conv.74) were
- 1613 downloaded from the links provided in the YOLO official website
- 1614 (<u>https://pjreddie.com/darknet/yolo</u>, accessed 15/03/2018) (Redmon 2018).
- 1615 3.2.7.3 Training with fruit images
- 1616 Models were trained on the training set (Train set 1; Table 3-2) using default parameters of the
- 1617 networks (YOLO, SSD and Faster R-CNN). The trained models were then tuned for the highest F1-
- score using the validation set. F1 score is the weighted average (harmonic mean) of precision and
- 1619 recall, and varies between 0 and 1. To tune the model on the validation set, non-maximal
- suppression (NMS) was varied across the range 0.1 to 0.6, in 0.1 steps, but including 0.35 and 0.45,
- 1621 while the confidence threshold was kept to the default minimum values for each detection
- 1622 framework. The NMS threshold and class confidence threshold associated with the highest F1 score
- 1623 were used as the 'tuned' parameters for the model for fruit detection on test sets.
- 1624 Detection files were generated for the test set using the scripts: *test_net.py* for Faster R-CNN and
- score_ssd_pascal.py for SSD from their official repositories. Generation of detection files in
- 1626 YOLOv2/v3 required passing 'valid' as a command line argument to Darknet's 'detector' function. To
- 1627 generate precision and recall scores as well as the average precision, the script 'voc_eval_py3.py'

- included in the Faster R-CNN official repository was used. This script was also modified to output thehighest F1 score and associated confidence cut-off thresholds for model tuning.
- 1630 Two models were created using the *MangoYOLO* architecture, one involving training from scratch
- 1631 (no transfer learning used) on the mango fruit training image set (*MangoYOLO(s)-512*), while the
- 1632 second model involved training on the COCO dataset (following https://pjreddie.com/darknet/yolo/)
- 1633 for 120K iterations before training on the mango fruit image training dataset (*MangoYOLO(pt)-512*).

1634 **3.2.8 Model comparisons**

- 1635 Seven models were compared, relative to human labelling of images, using: test sets 1 (orchard A 1636 test set tiles) and 2 (images from trees from each of five orchards) (Table 3-2). In the latter case,
- 1637 models trained on 512×512 pixel tiles were applied directly on the full (2448×2048 pixels) images by
- 1638 setting the model input resolution to 2048×2048 pixels in configuration files. The increase in
- 1639 network resolution allowed processing of large images without need for tiling, i.e., the model
- 1640 weights remained the same but the network used larger feature maps for fruit detection. Fruit pixel
- size remained the same whether it was from either tiled image or a full image.
- 1642 MangoYOLO was also compared to the Faster R-CNN(VGG) and Faster R-CNN(ZF) results of Bargoti
- and Underwood (2017a). For the latter case, a *MangoYOLO* model (*MangoYOLO(sbu)-512*) was
- 1644 trained from scratch with the training set of Bargoti and Underwood (2017a) (Table 3-2) for
- 1645 comparison to a model trained with the night imaging training set used in other exercises in the
- 1646 current study (*MangoYOLO(s)-512*). These exercises were undertaken to gauge model robustness to
- 1647 plant phenotype and to lighting conditions.
- 1648 The *MangoYOLO(s)-512* model, trained on orchard A training tiles extracted from images acquired
- 1649 with a Basler camera, was also used in detection of fruit in images from the Canon and Kinect
- 1650 cameras for the 17 test trees of orchard A. This exercise was undertaken to demonstrate model
- 1651 robustness across camera platforms.

1652 **3.2.9 Orchard fruit load estimation**

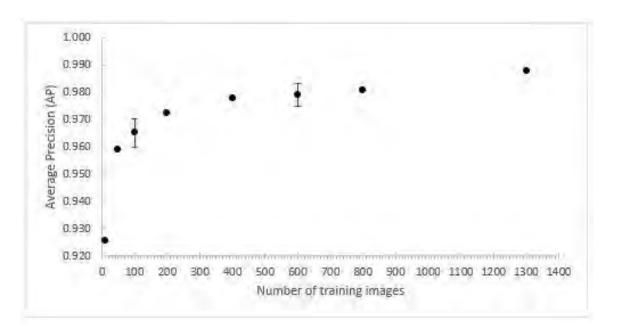
- 1653 To estimate the fruit load of an orchard from a machine vision count of two sides of all trees, a 1654 correction factor was calculated for 'hidden' fruit (i.e., fruit not seen in either side view of the 'dual-1655 view' per tree). The correction factor per orchard was calculated as a ratio of the sum of the number 1656 of fruit (human count) in images of each side of the canopy to the hand harvest count per sample 1657 tree (approximately 18 trees per orchard, Table 3-1). The total number of fruit detected on the block by machine vision was multiplied by the correction factor to estimate the total fruit count per 1658 1659 block/orchard. Three estimates of fruit load for each of the five orchards were made, based on use 1660 of Faster R-CNN with VGG, YOLOv3 and MangoYOLO models, with comparison to the pack-house count of harvested fruit. Percentage error on the machine vision estimate was calculated as 1661
- 1662 $\frac{SE_{A\times B}}{A\times B} \times 100$, where $SE_{A\times B} = A \times B \times \sqrt{(\frac{a}{A})^2 + (\frac{b}{B})^2}$ and A is the hidden fruit correction factor and 1663 a is its associated standard error, B is the machine vision count per tree and b is its associated RMSE.

1664 **3.3 Results**

1665 **3.3.1 Number of training images**

1666The AP of MangoYOLO(s) models in prediction of validation set 1 improved with increasing number1667of training tiles, increasing from a score of 0.925 with just 10 training images, to a plateau around

1668 0.98 after about 400 training images (Fig. 3-5). The *MangoYOLO(s)* model benchmarked to other 1669 models in this study was created using the full set of 1300 training tiles.



1670

1671Figure 3-5. Average Precision for MangoYOLO(s)-512 on validation set 1 (Table 2) plotted against the1672number of training images from Training set 1. Error bar represents standard deviation on three repeated1673assessments.

1674 **3.3.2 Model performance**

Eight models using an image resolution of 512×512 pixels were evaluated in terms of speed in training and inference, and model size. Image batch size was set at 1 (i.e., only one image loaded into memory during training). *MangoYOLO(s)/MangoYOLO(pt)* had the smallest size and a similar GPU memory usage as YOLOv2 (tiny)-512, i.e., the least of the seven models (Table 3-3).

1679Table 3-3. GPU memory consumption in training and testing time for models using 512×512 pixel images and1680final size of the trained model weight files. Values assessed on HPC hardware and using Darknet repository1681(https://github.com/AlexeyAB/darknet, accessed on 15/03/2018) in the YOLO variants.

	GPU memo	Trained weights	
Models	Training (Mb)	Inference (Mb)	Size (Mb)
Faster R-CNN(VGG)-512	2739	1759	533.948
Faster R-CNN(ZF)-512	1719	1123	230.105
SSD-512	2119	1259	92.759
YOLOv3-512	2101	2097	240.533
YOLOv2-512	1643	1639	261.957
YOLOv2(tiny)-512	849	845	61.604
MangoYOLO(s)-512	877	873	53.776
MangoYOLO(pt)-512	877	873	53.776

1682

GPU memory consumption Trained weights

	Training	Inference	
Models	(Mb)	(Mb)	Size (Mb)
Faster R-CNN(VGG)-			
512	2739	1759	533.948
Faster R-CNN(ZF)-512	1719	1123	230.105
SSD-512	2119	1259	92.759
YOLOv3-512	2101	2097	240.533
YOLOv2-512	1643	1639	261.957
YOLOv2(tiny)-512	849	845	61.604
MangoYOLO(s)-512	877	873	53.776
MangoYOLO(pt)-512	877	873	53.776

1683A further five models trained using images at original resolution were evaluated. The required1684memory allocation more than five-fold, e.g., YOLOv3 GPU memory requirement on the HPC1685increased from 2101 Mb (Table 3-3) to more than 16 GB (data not shown) for inference using

1686 512×512 and 2048×2048 pixel image resolution, respectively. This memory requirement exceeded

1687 capacity so resolution was set to 1888×1888 pixels, for which 16 GB was required (Table 3-5).

All models achieved good detection results on Validation set 1 (F1 scores > 0.9, Fig. 3-7a), with the

highest F1scores associated with *MangoYOLO(pt)-512* (0.968), MangoYOLO(s) (0.967), SSD-512

1690 (0.959) and YOLOv3-512 (0.951, Table 3-4). Tiny YOLO-original had the lowest F1 score (0.90).

1691 *MangoYOLO(s)-512* achieved the highest average precision (AP=0.986), with similar results achieved

1692 for *MangoYOLO(pt)* (AP=0.983), and SSD-300 (AP=0.982), while the lowest AP was associated with

1693 Faster R-CNN-VGG-original (AP=0.917) (Table 3-4).

1694Table 3-4. Model performance (F1-score and AP) for fruit count on Validation set 1 (512×512) images and1695average detection/inference times for different models. Values assessed on HPC hardware and using

1696 Darknet repository (https://github.com/AlexeyAB/darknet, accessed on 15/03/2018) in the YOLO variants.

Models	F1-score	Average precision	Inference time
Faster R-CNN(VGG)-original	0.936	0.918	0.067
Faster R-CNN(VGG)-512	0.945	0.953	0.067
Faster R-CNN(ZF)-original	0.929	0.933	0.037
Faster R-CNN(ZF)-512	0.939	0.950	0.037
SSD-300-original	0.950	0.983	0.046
SSD-512	0.959	0.973	0.070
YOLOv3-512	0.951	0.967	0.025
YOLOv2-original	0.916	0.945	0.020
YOLOv2-512	0.933	0.959	0.020
YOLOv2(tiny)-original	0.900	0.938	0.010
YOLOv2(tiny)-512	0.917	0.953	0.010
MangoYOLO(s)-512	0.967	0.986	0.015
MangoYOLO(pt)-512	0.968	0.983	0.015

¹⁶⁹⁷

The detection/inference times on 512×512 pixels images was lower for YOLO models than other
 models (Table 3-4). The shortest inference time was achieved by YOLOv2(tiny) (10 ms) and
 MangoYOLO(pt)-512/MangoYOLO(s)-512 (15 ms), while SSD-512 model required 70 ms (on a HPC

resource). For full canopy images (2048×2048 pixels), *MangoYOLO(s)/MangoYOLO(pt)* achieved a

1702 good detection speed of 70 ms per image (~ 14 fps), and consumed the least memory of the three

- 1703 models compared (Table 3-5). For the test set-Overall, the lowest Root Mean Square Error (RMSE)
- and highest R² (correlation coefficient of determination) was associated with the *MangoYOLO*
- 1705 models (Table 3-6). For the same network resolution, Faster R-CNN(VGG) out-performed Faster R-

1706 CNN(ZF) (Table 3-6).

1707 Table 3-5. Detection speed and memory usage for three models used with full canopy images (2048×2048).

- 1708 Values assessed on HPC hardware and using Darknet repository (https://github.com/AlexeyAB/darknet:
- accessed on 21/11/2018) in the YOLO variants. Within the GPU memory constraint the maximum network
 resolutions for YOLOv3 were 1888×1888 and 1184×1184 for HPC and Nuvo computing platforms respectively
- resolutions for YOLOv3 were 1888×1888 and 1184×1184 for HPC and Nuvo computing platforms respectively
 (note: YOLO requires the network input resolution to be multiple of 32).

	Network input resolution	HPC Detection	HPC GPU memory	Nuvo Detection	Nuvo GPU memory
Model	(pixels)	time (ms)	(Mb)	time (ms)	(Mb)
MangoYOLO(s)/MangoYOLO(pt)	512×512	<1	879	8	833
YOLOv3	512×512	<1	2103	30	2055
YOLOv2(tiny)	512×512	<1	851	5	834
MangoYOLO(s)/MangoYOLO(pt)	2048×2048	20	4513	70	4417
YOLOv3	1184×1184	-	-	123	6805
YOLOv3	1888×1888	20	16273	-	-
YOLOv2(tiny)	2048×2048	10	3279	51	3270

1712

1713

1714	Table 3-6. Model performance (R ² , Root Mean Square Error (RMSE) of prediction and Bias) for fruit counts
1715	using the overall test set (Test set 1 All) and subsets of low, medium and high frequency of fruit occlusion
1716	(n=100 tiles in each set). Units for RMSE and bias are fruit number per tile. Best result within a column is

(n=100 tiles in each set). Units for RMSE and bias are fruit number per tile. Best result within a column is
 indicated in bold. Input resolution was 416×416 pixels for YOLO original variants. Images were rescaled for

1718 the shorter side to be 600 pixels for Faster R-CNN original variants.

	1	Fest set 1	All	Te	est set 1 l	ow	Test	set 1 Me	edium	Te	est set 1	High
Model	R ²	RMSE	Bias	R ²	RMSE	Bias	R ²	RMSE	Bias	R ²	RMSE	Bias
Faster R-CNN(VGG)- original Faster R-CNN(VGG)-	0.97	2.25	-1.24	0.96	0.68	-0.14	0.92	1.67	-1.02	0.90	3.46	-2.6
512 Faster R-CNN(ZF)-	0.98	1.22	-0.43	0.98	0.44	0.01	0.96	0.97	-0.33	0.95	1.83	-1
original	0.95	2.19	-0.97	0.92	0.85	0.05	0.87	1.75	-0.74	0.89	3.25	-2.2
Faster R-CNN(ZF)-512	0.96	2.06	-0.99	0.95	0.68	-0.10	0.94	1.28	-0.63	0.90	3.25	-2.2
YOLOv2(tiny)-original	0.95	1.66	0.09	0.92	0.90	0.31	0.92	1.38	0.43	0.89	2.35	-0.5
YOLOv2(tiny)-512	0.97	1.46	0.49	0.91	1.00	0.45	0.93	1.35	0.55	0.93	1.89	0.46
YOLOv2-original	0.96	1.85	-0.90	0.94	0.76	-0.02	0.93	1.28	-0.63	0.93	2.83	-2.1
YOLOv2-512	0.97	1.33	0.13	0.95	0.70	0.27	0.92	1.21	0.05	0.93	1.82	0.08
YOLOv3-512	0.98	1.18	-0.18	0.96	0.60	0.02	0.95	0.95	-0.18	0.95	1.72	-0.4
SSD-300	0.98	1.31	-0.52	0.98	0.44	-0.01	0.94	1.19	-0.47	0.95	1.88	-1.1
SSD-512	0.98	1.48	-0.65	0.96	0.60	0.02	0.95	1.26	-0.62	0.95	2.14	-1.4
MangoYOLO(s)-512	0.98	1.09	-0.39	0.98	0.44	-0.05	0.95	1.04	-0.39	0.96	1.51	-0.7
MangoYOLO(pt)-512	0.98	0.99	0.16	0.96	0.64	0.19	0.96	0.81	0.03	0.97	1.37	0.25

1719

3.3.3 Model robustness 1720

3.3.3.1 Camera comparison and daylight imaging 1721

1722 For images acquired of the same trees under the same lighting, Canon images were the brightest 1723 and Kinect images were the darkest (Fig. 3-6). MangoYOLO(pt) and MangoYOLO(s) models were 1724 employed in estimation of fruit number per image for images of Test set 2A (17 trees in orchard A) 1725 collected using three cameras (Table 3-7). The MangoYOLO(pt) model produced higher fruit counts 1726 than the *MangoYOLO(s)* model, but with higher false positive rates and a lower R^2 . The highest false positive rate (ratio of counts of false positive detections to total detections expressed as percentage) 1727 1728 was associated with detection using MangoYOLO(pt) on Canon images. It was observed that in 1729 resizing the Canon images from 6000×4000 pixels to 2048×2048 pixels that image of some leaves 1730 took a curved shape similar to fruit. Moreover, false detections were also registered for parts of 1731 branches and tree trunks where the light intensity was high. Less fruit detection occurred for Kinect 1732 images using both *MangoYOLO(s)* and *MangoYOLO(pt)* models.



1733

(b) Canon image

(c) Kinect image

- 1734 Figure 3-6. Example of fruit detection on images of same tree (Test set 2A) for different cameras (Basler,
- 1735 Canon and Kinect), using a MangoYOLO(s) model trained on Train set 1.
- 1736

1737 Table 3-7. Regression statistics for fruit detection by MangoYOLO(s) and MangoYOLO(pt) on sets Test set 2 1738 A, Test set 2 A-can and Test set 2 A-kin captured using Basler, Canon and Kinect cameras respectively.

	Original image	Network input		ground truth	total detected				false positive
Test sets	resolution	resolution	Model	fruit #	fruit #	R ²	RMSE	Bias	rate %
Test set 2 A	2448×2048	2048×2048	MangoYOLO(s)	2163	2103	0.9	96 2.2	4 1.7	7 0.00
Test set 2 A	2448×2048	2048×2048	MangoYOLO(pt)	2163	2201	0.9	87 3.2	5 -1.1	L2 0.40
Test set 2 A-ca	n 6000×4000	2048×2048	MangoYOLO(s)	2137	2014	0.9	87 4.4	3.6	2 0.44
Test set 2 A-ca	n 6000×4000	2048×2048	MangoYOLO(pt)	2137	2455	0.9	64 10.	90 -9.3	85 6.31
Test set 2 A-kir	n 1920×1080	1024×1024	MangoYOLO(s)	1746	1512	0.9	81 7.3	9 6.8	8 0.00
Test set 2 A-kir	n 1920×1080	1024×1024	MangoYOLO(pt)	1746	1560	0.9	65 6.5	2 5.4	7 0.32

1739

1740 3.3.3.2 Bargoti and Underwood images

- 1741 MangoYOLO-512 was further benchmarked on the mango dataset created by Bargoti and
- 1742 Underwood (2017a) (Table 3-8). Poor results were obtained for the models trained on the night
- 1743 images acquired with the Basler camera when applied on the daytime/Prosilica camera images. In

- 1744 contrast, when a *MangoYOLO* model was trained using the Bargoti and Underwood (2017a) daytime
- images, it performed reasonably well on the night-Basler images. *MangoYOLO* trained using the
- 1746 Bargoti and Underwood (2017a) training set (from scratch or pretrained on the COCO data set)
- 1747 outperformed YOLOv3.

1748 Table 3-8. Prediction results for models trained and tested on the image sets used in (Bargoti and

- 1749 Underwood 2017a). Results for the Faster R-CNN(ZF) and Faster R-CNN(VGG) models are from Bargoti and
- 1750 Underwood (2017a). YOLO models sbu, and ptbu refer to models s and pt trained on bu (Bargoti and
- 1751 Underwood 2017a) dataset. Network resolution for YOLO variants was set to 512×512 pixels for training and 1752 testing.

-							
Train Set	Test set	Model	AP	F1	R ²	RMSE	Bias
Train set 1	Test set 1-Overall	MangoYOLO(s)	0.991	0.959	0.98	1.09	-0.39
Train set 1	Test set 1-Overall	MangoYOLO(pt)	0.989	0.956	0.98	0.99	0.16
Train set 2-bu	Test set 3-bu	MangoYOLO(sbu)	0.927	0.890	0.93	1.12	0.17
Train set 2-bu	Test set 3-bu	MangoYOLO(ptbu)	0.920	0.893	0.91	1.33	0.42
Train set 2-bu	Test set 3-bu	YOLOv3(sbu)	0.889	0.875	0.92	0.18	0.01
Train set 1	Test set 3-bu	MangoYOLO(s)	0.420	0.482	0.36	3.96	2.46
Train set 1	Test set 3-bu	MangoYOLO(pt)	0.463	0.535	0.39	0.39	2.38
Train set 2-bu	Test set 1-Overall	MangoYOLO(sbu)	0.834	0.868	0.86	4.60	3.13
Train set 2-bu	Test set 1-Overall	MangoYOLO(ptbu)	0.931	0.913	0.95	2.58	1.65
Train set 2-bu	Test set 3-bu	Faster R-CNN (VGG)	-	0.908	-	-	-
Train set 2-bu	Test set 3-bu	Faster R-CNN (ZF)	-	0.876	-	-	-

1753

1754 3.3.3.3 Orchard and cultivar

1755 Models trained using a set of images from one orchard and one cultivar (Calypso) only (i.e., Train set 1756 1) were employed in evaluation of fruit load of orchards varying in location, growing condition and 1757 cultivar. Despite variations in fruit shape and leaf and fruit colour between orchards and cultivars, 1758 all models performed well when applied to images of other orchards, without further training or fine 1759 tuning (Fig. 3-7, Table 3-9). The best performance metrics (highest R² and lowest RMSE) on the relationship between machine vision counts of each sides of a tree and human count was achieved 1760 1761 using the MangoYOLO(pt) model in four orchards and using the MangoYOLO(s) model in one 1762 orchard, relative to Faster R-CNN(VGG) and YOLOv3 models (Table 3-9). The poorest result for all 1763 models was associated with orchard B which contained trees with larger canopies than the other 1764 orchards.

Deep learning- fruit detection



1765

R2E2: Orchard E

1766	Figure 3-7. Example of fruit detection on images of cultivars HoneyGold and R2E2, using a MangoYOLO(s)-
1767	512 model trained on cultivar Calypso images (orchard A).

1768

1769 Table 3-9. Regression statistics of machine vision count against human count of fruit on images of sample 1770 trees (Table 3-1) for five orchards. Best result within a row is indicated in bold. Network resolution for Faster 1771 R-CNN and MangoYOLO models was set to 2048×2048 but for YOLOv3 it was 1888×1888 (maximum possible 1772 within available GPU memory of 16 Gb).

		Faster R-CNN(VGG)		YOLOv3		MangoYOLO(s)		MangoYOLO(pt)	
Orchard	Mean fruit/tree image	R ²	RMSE	R ²	RMSE	R ²	RMSE	R ²	RMSE
А	63.3	0.966	6.88	0.980	3.48	0.984	4.09	0.988	2.52
В	80.3	0.955	14.53	0.957	10.33	0.951	11.79	0.964	8.71
С	35.1	0.981	6.76	0.983	5.05	0.981	5.90	0.990	3.20
D	34.1	0.968	6.61	0.988	4.77	0.988	4.91	0.977	3.91
E	15.3	0.940	2.93	0.873	2.42	0.931	2.74	0.946	2.23

1773

3.3.4 Orchard fruit load estimation 1774

1775 A correction factor for the occluded fruit on the trees was calculated as the average ratio of ground 1776 truth human fruit count per tree image to the harvest count per tree. This value was estimated from 1777 the data of the sample trees in each orchard, following the approach of Anderson et al. (2018). The 1778 correction factors and their standard deviations were 1.69 ± 0.5 , 1.63 ± 0.46 , 2.43 ± 1.21 , 1.05 ± 0.17 1779 and 1.32 ± 0.46 for orchards A, B, C, D and E respectively. The total number of fruits detected in the 1780 images for an orchard was multiplied by the common correction factor to estimate the fruit counts 1781 (fruit load) per orchard (Table 3-10). The percentage error on the orchard estimates combined the 1782 error of the correction factor estimate and the machine vision count estimate, varying between 8 and 24% across orchards and models. 1783

1784Table 3-10. Pack-house fruit count and number of fruits detected for each orchard using dual view imaging1785six weeks before harvest, for each of five orchards. Best result within a row is indicated in bold. Netwo rk1786resolution was set to 2048×2048 for Faster R-CNN and *MangoYOLO* models, and to 1888×1888 (maximum1787possible within available GPU memory of 16 Gb) for YOLOv3.

	Orchard	А	В	С	D	E
	Packhouse fruit #	97382	26273	40837	36490	2110
Machine vision model						
Faster R-CNN(VGG)	fruit # estimate	93694	25162	41337	40151	2343
	% error estimate	13	21	24	20	21
	% difference to packhouse count	-4	-4	1	10	11
YOLOv3	fruit # estimate	100098	27102	43379	41689	2409
	% error estimate	9	17	20	14	18
	% difference to packhouse count	3	3	6	14	14
MangoYOLO(s)	fruit # estimate	98029	26453	42179	41146	2383
	% error estimate	10	19	22	15	20
	% difference to packhouse count	1	1	3	13	13
MangoYOLO(pt)	fruit # estimate	101857	27742	44501	42020	2400
	% error estimate	8	16	17	12	17
	% difference to packhouse count	5	6	9	15	14

1788

1789 The machine vision estimates based on *MangoYOLO(s)* and Faster R-CNN(VGG) were closest to the 1790 packhouse counts in two and three orchards, respectively. The higher error for the orchard D 1791 estimate was associated with small trees in which the same fruit could be seen from both sides of 1792 the tree.

1793

1794 **3.4 Discussion**

1795 **3.4.1** Architecture

In the current study, features of two YOLO variants (YOLOv3 and YOLOv2(tiny)) were combined to
create a new architecture (*MangoYOLO*). This architecture provided a processing speed appropriate
to real time operations, with use of only 33 layers for *MangoYOLO* compared to 107 layers in
YOLOv3, while the accuracy was on par to (or better than) other detectors for the task of mango
fruit detection in orchard.

1801 The convolutional and pooling layers used in YOLO gradually decrease the spatial dimension with 1802 increasing depth of the network, such that detection of some objects (e.g., small and non-obvious or 1803 dark fruits) may become difficult at lower resolutions. Predicting layers were implemented at three 1804 different resolutions of the feature maps, as in YOLOv3, concatenating meaningful semantic 1805 information from the deeper layers with fine grained information from the earlier layers for better

- 1806 detection of objects. The first detection layer was implemented at layer 42 in *MangoYOLO*,
- 1807 compared to layer 82 in YOLOv3, in an attempt to capture information from earlier layers of the
- 1808 network. The superior performance of *MangoYOLO* compared to YOLOv3 on the mango test set
- 1809 indicates that this design was successful. This result demonstrates that use of very deep networks
- 1810 may not yield better results, depending on the context of deployment, and is consistent with advice
- 1811 that architecture should be customised to the application.

1812 There is a chemometric adage that there should be 'no prediction without interpretation and no 1813 interpretation without prediction' (e.g., Herold B et al. (2009). However, it is difficult to interpret 1814 the multidimensional learning capacity of the deep-learning models. The output of convolution 1815 filters can be visualized for interpretation of the features learned and saliency/heat maps can be 1816 generated to determine the parts of the images that were more important to the model for object 1817 detection and classification tasks (Zeiler and Fergus 2014), however in the current application this 1818 merely indicates that regions of the image associated with the whole fruit were utilised in the 1819 model. As the deep learning models performed well (high F1 score and AP) in cross cultivar 1820 generalization and detection of green fruit against a green background it can be inferred that models 1821 weighted object edge and texture features rather than object colours. An infrequent error was 1822 associated with fruit covered by foliage on all side of its perimeter, thus lacking defined edges. This 1823 error is also consistent with the interpretation that the deep learning models weighted edge or

- 1824 shape features of fruit.
- 1825 **3.4.2 Training**

1826 3.4.2.1 Pre-training

Mango-YOLO(s) was trained on the mango orchard training set only, while MangoYOLO(pt) involved
 pre-training on a COCO dataset for 120K iterations before training on the mango training set. The
 transfer learning of the features learned on the larger dataset was of minor value as the validation
 results of MangoYOLO(pt) was similar to MangoYOLO(s).

Pre-training *MangoYOLO* on the COCO dataset (*MangoYOLO(pt)*) did not significantly improve
performance compared to a model that was not initialized with COCO trained weights
(*MangoYOLO(s)*). Possibly the transfer learning was not helpful as the COCO dataset does not
contain images of mango fruit. Perhaps a more likely explanation is that the detection task is

- 1835 relatively simple, and the shallower CNN architecture and heavy data augmentation techniques used
- 1836 in *MangoYOLO* allow for training of a robust model using a modest number of images.

1837 3.4.2.2 Number of training images

1838 The training image set should encompass the variation expected in prediction images. With data 1839 augmentation (hue, saturation, jitter and multiscale), the performance (AP) of *MangoYOLO(s)* 1840 models plateaued with use of >400 tiles in the training set. The use of at least 1000 images is

recommended to provide some safety margin. *MangoYOLO* models trained with the full training set
 of 1300 tiles were robust in prediction of images from other orchards and from other cameras.

1843 In contrast, Bargoti and Underwood (2017a) reported that for images acquired of the same orchard

(orchard A) in a previous year, a performance asymptote was not reached for a Faster R-CNN within
 the available set of 1154 training images. The difference between the two studies may result from

- 1846 the differences in lighting conditions. Night imaging results in suppression of background issues and
- 1847 often highlights fruit through specular reflection from the curved fruit surface (Payne et al., 2014).

- Presumably night imaging allows for the quicker convergence, using a lower number of images, withconsequent saving in the labour cost for labelling.
- 1850 The standard deviation (SD) on AP scores (Fig. 3-5) for thee repeated training exercises using a
- randomly selected 100 tiles (SD = 0.0053) was only slightly higher than that for training with 600 tiles
- 1852 (SD = 0.0041), indicative of a low level of variance in the night imaged training set. Indeed, an AP
- 1853 score of 0.925 was achieved for the validation set with use of just 10 training images. This result is
- ascribed to the heavy data augmentation used by YOLO.

1855 **3.4.3 Model performance**

1856 3.4.3.1 Comparison to previous reports

Fruit and leaf colour, shape and texture can vary with cultivar and growing condition/stage. The prediction results of handcrafted algorithms based on colour, texture and shape reported by Payne et al. (2014) and Qureshi et al. (2017) for mango fruit detection were poor relative to the results of the deep learning architectures reported in the current study, when used with images or canopies from new orchards and cultivars. For example, Qureshi et al. (2017) reported a RMSE of 11.0 fruit/tree for fruit load on prediction set images, in comparison to RMSE values < 8.7 fruit/image in the current study (Table 3-9). The result confirms previous observations of the applicability of deep

1864 learning architectures for the fruit detection task.

1865 3.4.3.2 Performance characterisation

1866 Application of deep learning models directly on higher resolution images required larger memory

allocation, with 2048×2048 images requiring more than 16 GB of memory for use of YOLOv3-512
 (Table 3-5). If GPU memory becomes a bottleneck for detection on large images, the tiling approa

(Table 3-5). If GPU memory becomes a bottleneck for detection on large images, the tiling approach
used by Bargoti and Underwood (2017a) can be adopted, wherein detection is done by sliding

1870 smaller windows over the image and finally NMS is applied collectively on the detections. However,

- 1871 this route is not required with the *MangoYOLO* model, given its low memory requirement.
- 1872 Given data augmentation, all of the 'off-the-shelf' deep learning architecture performed well in fruit
- 1873 detection, with *MangoYOLO* achieving the highest F1 score and YOLOv2(tiny) the greatest speed.
- 1874 The detection accuracy results (F1 scores) of the current study were slightly higher than those
- 1875 reported by Bargoti and Underwood (2017a) who used Faster R-CNN(VGG/ZF) with daytime images
- of orchard A of the current study. For the same network resolution, Faster R-CNN(VGG) outperformed Faster R-CNN(ZF) (Table 3-4), consistent with the report of Bargoti and Underwood
 (2017a).
- 1879 The *MangoYOLO(s)* and *(pt)* models were equally robust (i.e., similar AP and F1 scores) in prediction

1880 of fruit load per image across diverse datasets (orchard, camera), although *MangoYOLO(pt)* achieved

- the lowest RMSE of all models tested for fruit detection across different orchards (Table 3-7).
- 1882 MangoYOLO models trained on the Bargoti and Underwood (2017a) daytime images achieved
- 1883 similar performance (F1 scores) compared to the results reported by Bargoti and Underwood
- 1884 (2017a) using a Faster R-CNN framework with VGGNet and ZFNet (Table 3-8). Of practical
- 1885 significance, the *MangoYOLO* model trained on daytime ProSilica camera images performed well in
- 1886 prediction of night Basler camera images.
- Bargoti and Underwood (2017a) reported that clustered fruits represented the greatest detection
 error source in the mango fruit-on-tree detection task. In the current study, the bounding box
- 1889 approach was adequate for detection of fruits in clusters. Therefore, the labour intensive method of

- 1890 instance segmentation for data labelling was not required. False detections on the Basler images
- 1891 were associated with curved leaves with contour similar to the fruit and some over-exposed areas of
- 1892 the branches and tree trunk. There were very few cases where detections for two fruit were merged
- 1893 into one when one fruit was heavily occluded by the other fruit.

The false detection rate was high (at 6.3%) for *MangoYOLO(pt)* when applied to images from the Canon camera (Table 3-7). The false detection mostly occurred from resizing of Canon images, resulting in image distortion, with leaves taking on a curved shape similar to fruit. Compared to the number of fruit detections on the Basler images, there were fewer detections for Kinect images presumably due to the lower luminosity (under-exposure) of those images, and more detections on Canon images (higher luminosity). There were also more false detections on Canon images, resulting from over exposed regions on branches, trunks and leaves.

1901 **3.4.4 Tree and orchard fruit load estimation**

1902 The deep learning methods performed well in detection of fruit in images. However, dual view 1903 images may not reveal all fruit on a tree with a denser canopy or may result in double counting of 1904 fruits if the same fruit is visible on both images of the same tree. An attempt was made to make an 1905 estimation of the proportion of hidden fruits as a correction factor for yield prediction, based on 1906 image and total counts of the number of fruits on the sample trees of Table 3-1. MangoYOLO(pt) 1907 yield estimations between 4.6 and 15.2 % of packhouse counts were achieved, however errors on 1908 these measurements were estimated to be large (Table 3-10). The application of a single correction 1909 factor based on a few sample trees across an entire orchard constitutes a source of error for orchard 1910 yield estimation as the correction factor may vary with tree canopy density and fruit distribution 1911 within the tree canopy, even within an orchard of consistent management practice, as evident in the 1912 error term of this factor.

1913 Future studies should consider system features to improve the estimate of fruit per tree. Image 1914 masking based on the depth, e.g., using a Microsoft Kinect- time of flight sensor (Wang et al. 2017b), 1915 could be used to avoid double fruit counting by limiting the count from a given image to the near 1916 side of the imaged canopy. Stein et al. (2016) addressed the issue of hidden fruit in dual view 1917 imaging by using a multi-view approach, involving spatial localisation of fruit from approximately 25 1918 images per tree side, given input of inertial navigation system and geolocation data. Ideally a 1919 tracking algorithm would be used without input of inertial navigation system and geolocation data, 1920 to minimize operational complexity and cost.

1921 **3.5 Conclusion**

1922 This work should encourage future researches to customise deep learning models for a given 1923 application task. The multiple image sets of fruit on canopy of the current study have been made 1924 available, for use by other researchers for benchmarking performance of new algorithms.

A deep learning architecture, *MangoYOLO*, was constructed, based on YOLOv3 and YOLOv2(tiny). A
F1 score of 0.97 for fruit detection in images was achieved, which, combined with a correction factor
for hidden fruits, gave orchard yield estimates within 15% of packhouse tallies. With detection of
fruit in the image now possible in real-time, the limiting factor for accurate prediction of the block
yield using dual view imagery of trees is canopy occlusion of fruit.

Pre-training of the *MangoYOLO* model on larger datasets like ImageNet, COCO and PascalVOC is
recommended but not essential, given training with data augmentation on around 1000 tiles of fruit
on tree. The architecture is recommended for the task of estimation of mango fruit in tree images in

- 1933 comparison to YOLOv3, YOLOv2(tiny), Faster R-CNN(VGG) and Faster R-CNN(ZF) in terms of memory
- 1934 use, speed and accuracy. This model can be used with images acquired from a farm vehicle
- 1935 operated at about 6 km/hr for real time fruit detection and can be deployed for different image
- 1936 resolutions without need for re-training. The model should be tested with other fruit types and
- 1937 other imaging conditions. Increasing image resolution and use of higher illumination levels may
- 1938 further improve the fruit detection rate, and further training of the models with images from other 1939 cultivars could improve performance in use with those cultivars.
- 1940 Acknowledgement
- 1941 This work received funding support from the Australian Federal Department of Agriculture and 1942 Water, and from Horticulture Innovation Australia (project ST15005, Multiscale monitoring of
- 1943 tropical fruit production). AK acknowledges receipt of an Australian Regional Universities Network
- scholarship, and ZW was funded by a CQU Early Career Fellowship. Farm support from Chad Simpson
- and Ivan Philpott is appreciated. The assistance of Jason Bell of the CQUniversity High Performance
 Computing cluster is acknowledged. The work also benefitted from discussion with AlexeyAB
- 1947 (https://github.com/AlexeyAB).
- 1948
- 1949
- 1950
- 1951
- 1952
- 1953
- 1954
- 1955
- 1956
- 1957
- 1958
- 1959
- 1960
-
- 1961
- 1962
- 1963

1964 Chapter 4. Deep learning for mango panicle stage 1965 classification

1966 At the time of initial submission of this thesis, this chapter had been submitted for consideration by 1967 the journal 'Agronomy' - special issue "In-field Estimation of Fruit Quality and Quantity". This manuscript was subsequently accepted with minor revisions, and a revised version was accepted. 1968 1969 The published version of this chapter has been included in the Appendix-B of this thesis. 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 Abstract 1987 A light and a deep learning framework (YOLO and R²CNN) were trained and tested for categorization 1988 1989 of mango panicles into several developmental stages and compared to a segmentation method. The use of upright and rotated bounding boxes was also compared, in context of the R²CNN method. A 1990

correlation R² = 0.90 was achieved between panicle counts made using the R²CNN method and the results of a pixel-based segmentation method of a previous publication for 1,988 tree images (994

1993 trees) of an orchard. For a validation set of images from the same orchards as the training set,

1994 RMSEs on panicle counts of 16.0, 25.8 and 32.3 panicles per tree image, with Mean average

precision (mAP) scores of 69.1, 62.5 and 70.9% and weighted F1-scores of 76.1, 74 and 82,
 were achieved using YOLO, R²CNN and R²CNN-upright models (R²CNN trained on upright bounding)

boxes), respectively. For a test set of images involving a different cultivar and use of a different

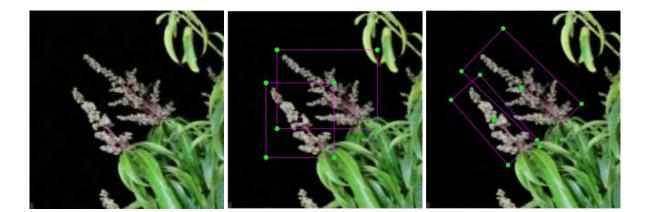
1998 camera, the R^2 for machine vision (two images per tree) to human count of panicles per tree was

- 0.803, 0.805 and 0.761 for YOLO, R²CNN and R²CNN-upright models, respectively. These predictions
 involved images from different camera hardware and cultivar to that used in training,
- 2001 indicating models can generalise well. In summary, there was no consistent benefit in use of rotated
- 2002 over upright bounding boxes in this application and, while the YOLO model was superior in terms of
- total panicle count, R²CNN proved more accurate for panicle stage classification. To demonstrate
- 2004 practical application, panicle counts were made weekly for an orchard, with a peak
- 2005 detection routine applied to document multiple flowering events.

2006 4.1 Introduction

- 2007 Mango trees produce panicles bearing hundreds of inconspicuous flowers, of which at most three or 2008 four flowers will develop fruit, although frequently only one or none will so develop. The 2009 assessment of the number of panicles on a tree thus sets a maximum potential for the crop yield of 2010 that season, while the assessment of stage of panicle development is useful for assessment of the 2011 time spread of flowering, and thus the likely time spread of the harvest period. Mapping areas of 2012 early flowering can also guide selective early harvesting. Panicle detection may also inform selective 2013 spraying operations. However, the manual assessment and recording of panicle number and stage
- 2014 is a tedious task.
- 2015 Machine vision has been applied for assessment of the level of flowering for several tree crops
- where the flowers are easily distinguishable from the background based on colour thresholding. For example, Aggelopoulou et al. (2011) reported a prediction accuracy of 82% on apple flower
- 2018 count, relative to a manual count, Dorj and Lee (2012) achieved a R² of 0.94 between machine vision
- 2019 and manual count of tangerine flowers, Horton et al. (2017) claimed an average detection rate of
- 2020 83% on peach flowers, Hočevar et al. (2014) reported a R² of 0.59 between machine vision
- and manual count of apple flower cluster counts, and Oppenheim et al. (2017) obtained a F1-score
- 2022 of 73% for tomato flower detection. Underwood et al. (2016) avoided a direct count of flowers, but
- 2023 rather characterised almond 'flowering intensity' in terms of a ratio of flower and canopy pixels,
- 2024 reporting a poor relationship ($R^2 = 0.23$) for this index for a given tree between two seasons.
- 2025 All these reports were based on segmentation routines, generally involving colour given the
- 2026 obvious colour difference for flowers of these species and background.
- 2027 There are some reports on use of machine vision for assessment of number of flowering panicles
- 2028 (i.e., inflorescences consisting of multiple flowers). For example, Diago et al. (2014) used intensity
- 2029 level in LAB colour space as an index to the number of grape flowers in inflorescences
- 2030 imaged against a black background. A R² of 0.84 against human count was achieved. Guo et al.
- 2031 (2015) used SIFT descriptors/features along with SVM to detect rice flower panicles in images.
- 2032 Recent review papers have emphasised the use of neural network and deep learning in agricultural 2033 machine vision in general (Kamilaris and Prenafeta-Boldú 2018) and for fruit detection and yield 2034 estimation (Koirala et al. 2019a). For example, Gongal et al. (2015) noted that better performance in 2035 fruit detection and localization was achieved with use of neural networks compared to traditional 2036 models based on colour thresholding and hand-engineered feature extraction methods. Koirala et al. 2037 (2019b) introduced the use of lighter weight, single shot detectors such as YOLO, which allow faster 2038 computation times. Dias et al. (2018a) used Clarifai (Zeiler and Fergus 2014) CNN architecture to 2039 extract features from the possible flower regions obtained from super-pixel segmentation followed by SVM (Cortes and Vapnik 1995) for flower detection. For apple, peach and pear datasets, this 2040 2041 method outperformed other methods of that time that were based on HSV and SVM colour 2042 thresholding methods. Extending their earlier work, Dias et al. (2018b) used a fully convolutional 2043 neural network (FCN) from Chen et al. (2018) for flower detection on tree images of apple, peach

- and pear. A region growing refinement (RGR) algorithm was implemented to refine the
- segmentation output from FCN. This method achieved F1 scores of 83, 77, 74 and 86 % on two
- 2046 apple, peach and pear flower datasets respectively, outperforming their previous Clarifai CNN
- 2047 method (Dias et al. 2018a) and the HSV-based method.
- 2048 Mango panicle size changes with developmental stage, increasing through bud break, 'asparagus',
- 2049 elongation, anthesis (flower opening) to the full bloom 'Christmas tree' stage, then decreasing with
- 2050 flower drop. Panicle structure is thus more complex than that of a single flower. Therefore, machine
- vision detection of mango panicles is more challenging compared to the detection of the single
- flowers of apple, citrus and almond trees. Deep learning methods of object detection may suit
- 2053 the task of detection and counting of panicles by developmental
- stage, through automatic learning of useful features for classification (Koirala et al. 2019a).
- 2055 There are two reports on the use of machine vision to assess mango flowering. Wang et al.
- 2056 (2016) and Wang et al. (2018b) used the traditional method of pixelwise segmentation to segment
- 2057 panicle pixels from canopy pixels, with results expressed either as panicle pixel count per tree or as
- 2058 the ratio of panicle to canopy pixel count (termed 'flowering intensity'). This procedure was
- 2059 implemented on images obtained at night using artificial lighting, processed with a colour threshold
- followed by SVM classification to refine the segmentation results. Wang et al. (2018b) also reported
- 2061 on use of a Faster R-CNN (Ren et al. 2015) deep learning technique to count panicles. This work was
- 2062 limited to estimation of the extent of flowering and the time of peak flowering event. A R^2 of 0.69,
- 2063 0.78 and 0.84 between machine-vision flowering intensity and in-field human count of panicles per
 2064 tree for 24 trees was reported for the segmentation method and a deep learning Faster R-CNN
- 2065 framework to using dual and multi-view imaging approaches, respectively.
- 2066 These earlier reports employed upright bounding boxes. However, panicles are oriented in some
- range of angles. Upright bounding boxes will therefore not fit tightly around the object
- perimeter (Fig. 4-1), and a larger amount of background signal will be included in the object class for
 training. This could adversely affect the classification accuracy of the model. Koirala et al.
- 2070 (2019a) advised use of an annotation bounding box as tight as possible around the objects to avoid
- 2071 background noise in the training image sets. R²CNN (Rotational Region CNN for Orientation Robust
- 2072 Scene Text Detection) (Jiang et al. 2017) is a modification of the Faster R-CNN object
- 2073 detection framework to incorporate training on rotated bounding box annotations for detecting
- arbitrarily-oriented objects in images. This modified framework seems suited to the task of panicledetection.
- 2076



- 2078 Figure 4-1. Left to right: original image, upright bounding box and rotated bounding box
- 2079 In the current paper, the task of panicle detection task is extended to another level through
- 2080 classification to developmental stage, with comparison of a large (R²CNN framework with
- 2081 ResNet101 CNN) and a small (YOLO framework with MangoYOLO CNN) object detection
- architecture. To the authors' knowledge, the current study is the first to classify the stage of
- flowering for an on-tree fruit crop and is the first report on use of the rotated bounding box method
- of R²CNN for flower panicle detection. The current study also utilizes the imaging hardware used
 by Wang et al. (2018b), allowing for a direct comparison of results of the traditional machine
- 2086 learning approach used by Wang et al. (2018). Field relevance is demonstrated by assessment
- 2087 of orchard flowering at regular intervals (e.g., weekly) to provide information on timing of flowering 2088 peaks, for use in estimation of harvest timing.
- 2089 4.2 Materials and methods

2090 4.2.1 Image acquisition

Tree images of orchard A (Table 4-1) were acquired at night every week from August 16 to October
18, 2018, using a 5 MP Basler acA2440-75µm RGB camera and a lighting rig (700 W LED floodlight)
mounted on a farm vehicle driven at a speed of about 5 km/hr, as described by (Wang et al.
2018b). The orchard contained 994 trees, and thus each weekly imaging event captured 1,988
images.

The image set of 24 trees from orchard B (Table 4-1; from Wang et al. (2018b) were used as a test set. In that study, images were acquired using a 24 MP Canon (DSLR 750D) camera. The number of panicles on these trees was manually counted, with categorisation to developmental stage. For both

- 2099 orchards, trees were imaged from each side, with a view from each inter-row ('dual-view' imaging).
- 2100

2101 Table 4-1. Orchard and imaging description

Orchard name	Location (lat., long.)	Variety	Camera	Image resolution
Orchard A	-23.032749, 150.620470	Honey Gold	Basler	2468×2048 pixels
Orchard B	-25.144, 152.377	Calypso	Canon	6000×4000 pixels

2102

2103 4.2.2 Data preparation

2104 4.2.2.1 Image annotation and labelling

Mango panicles in the training image sets were categorized into three stages; (i) stage X -panicles with flowers (whitish in colour) that were not fully opened (Fig. 4-2, top row); (ii) stage Y - panicles with open flowers (Fig. 4-2, middle row); and (iii) stage Z - panicles displaying flower drop (Fig. 4-2, bottom row). A four-category system was initially trialled, but human differentiation of the two early stages was problematic, and the resulting model performance was poorer than for the threecategory system (data not shown).



Stage X panicles



2112

Stage Y panicles



2113

Stage Z panicles

2114 Figure 4-2. Training examples for stages X to Z, by rows.

- 2115 Annotation was done using roLabelImg (<u>https://github.com/cgvict/roLabelImg</u>) which is a
- 2116 modification of LabelImg to incorporate rotated boxes. As the annotations prepared for training
- 2117 R²CNN contained rotated bounding boxes which cannot be used directly with the YOLO model, a
- 2118 separate python script was written to convert the rotated bounding box annotations to upright
- 2119 bounding box annotations for training of the YOLO model. A visualization of the rotated ground
- truth boxes and the transformed upright bounding box annotation is presented in Figure 4-
- 3. Separate Python scripts were written to convert the XML annotation format of roLabelImg to that
- 2122 of R²CNN and YOLO formats for model training.

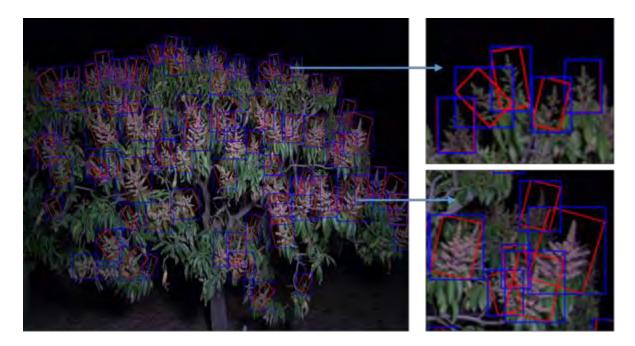


Figure 4-3. Display of ground truth bounding box original (rotated) red colour for R²CNN training and transformed (upright) blue colour for MangoYOLO training.

2126

2127 4.2.2.2 Training, validation and test sets

- 2128 Training was based on 54 images of orchard A (Table 4-2), which were drawn from different
- weeks. A validation set was assembled from images (orchard A) from one side of a single treeacquired over a six-week period.

2131 Table 4-2. Number of panicles in training and validation data sets.

			Number of panicle	25
	# images	Stage X (# panicles)	Stage Y (# panicles)	Stage Z (# panicles)
Train set (rotated/upright)	54	1007	1107	1064
Validation set (rotated/upright)	6	167	316	122

2132

The training dataset also included annotations for background (369 snips), as required for R²CNN training. R²CNN uses the background class for negative hard-data mining and was not treated as a detection class during inference. However, the YOLO object detection framework does not require a background class as all parts of the images other than those having bounding box for training are automatically treated as background. Therefore, the background class was not used for YOLO model performance assessment.

- 2139 The test set was an independent set consisting of images of orchard B, from the study of Wang et al.
- 2140 (2018b). These images were of trees of a different cultivar and from a different orchard, and
- acquired with a different camera, to that of the training and validation sets.

2142 **4.2.3** Computing

- 2143 Model training and testing was implemented on the CQUniversity High Performance Computing
- 2144 (HPC) facility graphics node with following specifications: Intel[®] Xeon[®] Gold 6126 (12 cores,
- 2145 2600MHz base clock) CPU, NVIDIA® Tesla® P100 (16 GB Memory, 1328 MHz base clock, 3584 CUDA
- cores) GPU. Red Hat Enterprise Linux Server 7.4 (Maipo) and 384GB RAM. CUDA v9.0, cuDNN v7.1.1,
- 2147 OpenCV v3.4.0, Python v2.7.14, GCC v4.8.5, scikit-learn v0.19.1, tensorflow-
- 2148 gpu v1.6.0, Keras v2.1.5, Cython v0.28.

2149 **4.2.4 Detection and classification models**

2150 *4.2.4.1 YOLO*

- A YOLO deep learning object detection framework was used with a MangoYOLO CNN classifier
- 2152 (Koirala et al. 2019b). MangoYOLO was originally conceived for on-tree mango fruit detection, and
- 2153 has an architecture based on the YOLOv3 (Redmon and Farhadi 2018) object detection framework,
- 2154 optimized for better speed and accuracy. In the current study, four classes (3 stages and 1
- 2155 background class) were used in YOLO training, however detection of the background class was
- 2156 ignored during model performance evaluation.
- 2157 The network input resolution for YOLO was set to 1024×1024 pixels following the YOLO requirement
- 2158 that the input images are square, and resolution is a multiple of 32. This resolution can be changed
- to higher values but at the cost of higher computation memory and slower train/test speed. As a
- 2160 data augmentation strategy during training, samples were rotated randomly in the range of + 40
- 2161 degrees to mimic variations in panicle orientations.
- 2162 YOLO was trained for 103.8k iterations and a batch size of 64 with data augmentation techniques
- 2163 defaulted to the YOLOv3 settings (saturation = 1.5, exposure = 1.5, hue = 0.1). The learning rate and
- 2164 momentum were set to the default values of 0.001 and 0.9 respectively. No transfer learning was
- 2165 employed, with the weights of the convolution neural network initialized at random values because
- 2166 the original MangoYOLO model training was based on mango fruit images, which are very different
- 2167 to flower images.

2168 4.2.4.2 R^2CNN

- 2169 R²CNN is basically a Faster R-CNN object detection framework with a modification to support
- 2170 training on rotated objects. The tensorflow re-implementation of R²CNN
- 2171 (<u>https://github.com/DetectionTeamUCAS/R2CNN_Faster-RCNN_Tensorflow</u>) was used for training 2172 and testing in this study.
- and testing in this study.
- All model training parameters were set to the default values of faster-RCNN for model training as
- 2174 implemented in R²CNN (<u>https://github.com/DetectionTeamUCAS/R2CNN_Faster-</u>
- 2175 <u>RCNN Tensorflow</u>). With R²CNN, as for Faster R-CNN, the input resolution can be set to be square or
- 2176 the original aspect ratio can be preserved (shorter side scaled to 800 pixels and longer side scaled
- accordingly). The original Basler images 2464×2048 pixels automatically resized to 962×800 pixels
 during training and testing. The RGB channel pixel mean values were initialized with the values (R =
- 2179 41.647, G = 41.675, B = 43.048) calculated of the training dataset.
- 2180 The R²CNN implementation supported only three CNN architectures mobilenet_v2,
- 2181 ResNet50_v1 and resnet101_v1. Given that deeper CNN models generally produce better results in
- terms of object detection and classification (Koirala et al., 2019), ResNet101_v1 CNN architecture
- 2183 was used with R²CNN framework for model training. The model was trained for 146k iterations

- with a batch size of 1 (as no support existed for batches with more than 1 image), learning
- 2185 rate of 0.0003 and momentum of 0.9. ImageNet pertained
- 2186 weights (<u>http://download.tensorflow.org/models/resnet_v1_101_2016_08_28.tar.gz</u>) were used as
- 2187 transfer learning to initialize the R²CNN model.

2188 4.2.4.3 R²CNN-upright

- 2189 To allow a comparison between rotated and upright box annotation in model training, a R²CNN-
- 2190 upright model was established. The R²CNN-upright method is the same as the R²CNN method except
- that the model was trained on the training set of upright annotation boxes, with the orientation of
- all boxes set to zero degrees, as used for training of the YOLO method. All training parameters of
- 2193 R²CNN were retained for training of the R²CNN-upright model.

2194 4.2.5 Estimation of peak of flowering

- 2195 Repeated (weekly) orchard imaging provided a time course of panicle number by
- 2196 week. The *signal.find_peaks* function from Scipy (<u>www.scipy.org</u>) packages was used to find peaks in
- the panicle numbers per tree side. Peak properties were specified as *height* = 10 and *distance* = 2.
- 2198 Height determines the minimum height of the peak which refers to the minimum number of panicles
- to consider as a peak. This parameter helps to filter noise (insignificant small peaks) in the signal.
- 2200 Distance determines the minimum horizontal distance in samples between neighbouring peaks.

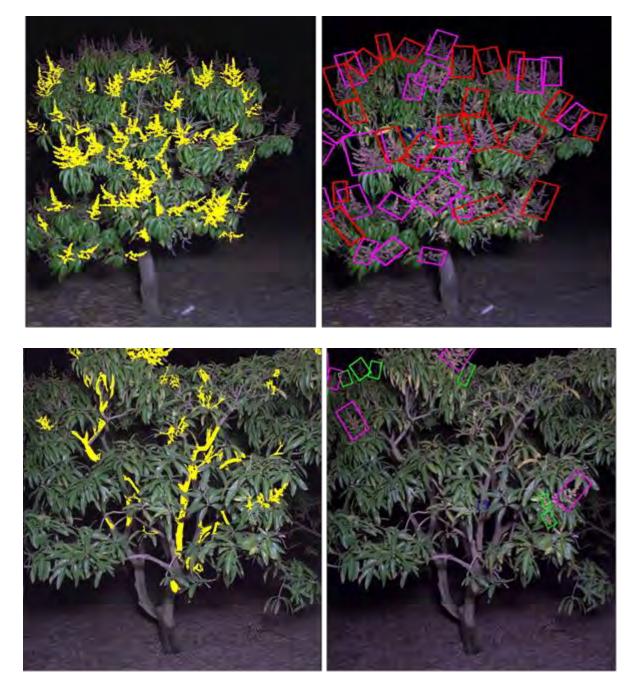
2201 **4.3 Results**

2202 4.3.1 Segmentation method

The pixel-segmentation method of Wang et al. (2018) differentiates pixels associated to panicles from background based on fixed values of colour thresholding. In poorly illuminated areas of images, panicles were not detected (false negative) while in some mages, parts of the tree such as branches or brownish/yellowish leaves were incorrectly classified as flower pixels. Two example images are presented, processed using the segmentation and deep learning R²CNN methods (Fig. 4-

2208

4).





- Figure 4-4. Pixel segmentation (left panels) and deep learning R²CNN (right panels) results for the same images. Flowers in the dark background did not segment properly (top left). Branches and leaves were erroneously segmented as flower pixels (bottom left).
- Flowering intensity (ratio of flower pixels to the canopy pixels, flowering pixels) was assessed following the method of Wang et al. (2018b) and correlated to the panicle counts per tree made
- 2217 using the R²CNN method, for all 994 trees of an orchard and each of five consecutive weeks (Table
- 4-3). Better correlation was obtained between stage Y panicle count rather than total panicle count
- in the last two weeks, as the proportion of stage Y panicles decreased (Table 4-3).
- 2220Table 4-3. Correlation R² between flowering intensity level per tree from pixel segmentation method and Y2221stage or all stages panicle counts, respectively, from the R²CNN method.

Week	1	2	3	4	5	6	7
	(16 Aug)	(23 Aug)	(30 Aug)	(6 Sept)	(13 Sept)	(20 Sept)	(27 Sept)

(Stage Y) R ²	0.903	0.896	0.892	0.788	0.853	0.871	0.708
(All stages) R ²	0.898	0.865	0.825	0.579	0.254	0.357	0.327
Avg. ratio of stage Y to total panicle count	0.376	0.390	0.395	0.361	0.416	0.320	0.147

2223 4.3.2 Deep learning methods

2224 An example of one image processed with the three methods of YOLO, R²CNN and R²CNN-upright is given as Figure 4-5. In general, RMSE for estimates of panicle count per tree of the validation set was 2225 2226 lowest with the YOLO method, and lower with use of rotated compared to upright bounding boxes 2227 for the R²CNN methods (Table 4-4). For example, RMSE on total panicle count per image was 16, 26 and 32 for the YOLO, R²CNN and R²CNN-upright methods, respectively. However, the highest 2228 2229 precision and F1 score was generally obtained with the R²CNN-upright model (Table 4-5). For 2230 example, mAP was 69.1, 62.5 and 70.9, and F1 was 76.1, 74.0 and 82.0 for YOLO, R²CNN and R²CNN-2231 upright methods, respectively.

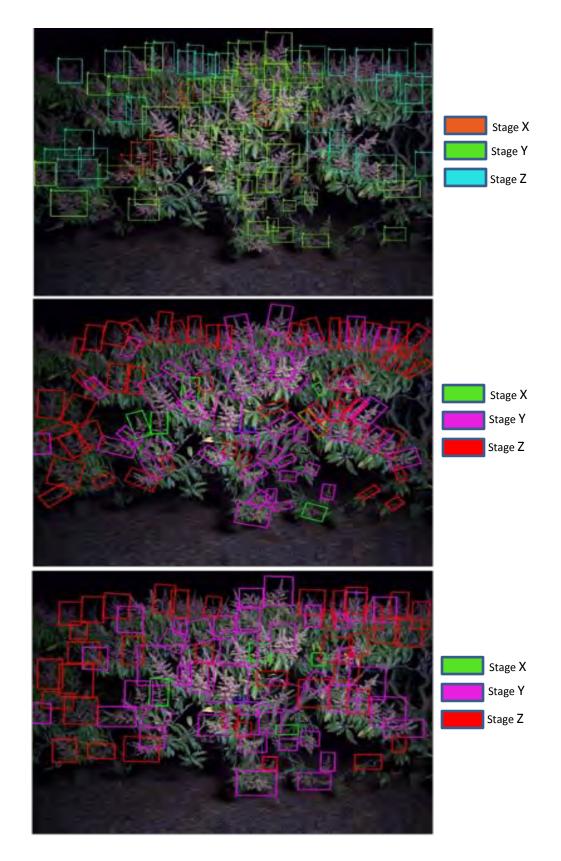
Table 4-4. Panicle stage detection results on the validation set using three methods. RMSE refers to a comparison with ground truth assessments of panicles per image. All values refer to # panicles/tree image. Lowest RMSE values are shown in bold.

Ground truth			R ² CNN		YOLO			R ² CNN -upright								
week	Х	Y	Z	total	Х	Y	Z	total	х	Y	Z	total	Х	Y	Z	total
1	1	37	72	110	1	27	64	92	0	36	73	109	1	30	46	77
2	13	67	42	122	9	54	36	99	3	65	22	90	11	41	29	81
3	28	91	8	127	9	62	15	86	24	78	6	108	22	53	7	82
4	28	69	0	97	16	49	5	70	24	65	0	89	22	46	0	68
5	46	28	0	74	38	21	1	60	45	21	0	66	36	17	0	53
6	51	24	0	75	36	16	0	52	54	17	0	71	43	18	0	61
		RMSE			11.6	16.4	5.4	25.8	4.9	6.9	8.2	16.0	6.3	21.8	11.8	32.3
		Bias			-9.7	-14.5	-1.7	-24.3	-2.8	-5.7	-3.5	-12.0	-5.3	-18.5	-6.7	-30.5

Table 4-5. Prediction statistics for the validation set using YOLO and R²CNN methods. Highest results are shown in bold.

Statistic	Average Precision			F1				
Panicle stage	Х	Y	Z	mAP	Х	Y	Z	Weighted F1
YOLO	68.7	78.1	60.5	69.1	75.4	79.0	69.5	76.1
R ² CNN	56.3	62.0	69.3	62.5	69.6	75.6	75.7	74.0
R ² CNN-upright	80.8	64.8	67.2	70.9	89.4	78.7	80.4	82.0

Deep learning- panicle assessment



2238

2239 Figure 4-5. Example images processed with three methods. Top panel: Panicle stage

detection using YOLO method. Orange, green and blue coloured boxes represent panicle stages X, Y and Z

2241 respectively. Middle panel and bottom panel: Panicle stage detection using R²CNN and R²CNN-

- upright methods, respectively. Green, pink and red coloured boxes represent panicle classes of X, Y and Z,
 respectively.
- 2244 YOLO and R²CNN methods were also compared in prediction of the test set of panicle count per tree
- from images (two per tree) collected from a different orchard, cultivar and camera to the calibration
- set (Table 4-6). The YOLO method achieved the lowest RMSE and bias of the three methods, with a
- 2247 R² similar to that achieved with R²CNN, due to a large bias on the R²CNN result (Table 4-6). The
- 2248 R²CNN-upright model returned a lower R² than the base R²CNN model, a result similar to that
- reported by Wang et al. (2018b) for a based Faster R-CNN method (which use upright boxes), for
- these trees.

Table 4-6. Comparison of several flower assessment methods on the test image set (as used by Wang et al. (2018b) in terms of the R² and RMSE between machine vision panicle (sum of two sides of a tree) count on images (two per tree) versus in-field human counts of panicles per tree.

Detection method	R ²	RMSE	Bias
YOLO	0.803	35.6	-6.4
R ² CNN	0.805	91.9	-72.2
R ² CNN upright	0.761	93.2	-72.4
Faster R-CNN (Wang et al. 2018b)	0.78	-	-

2254

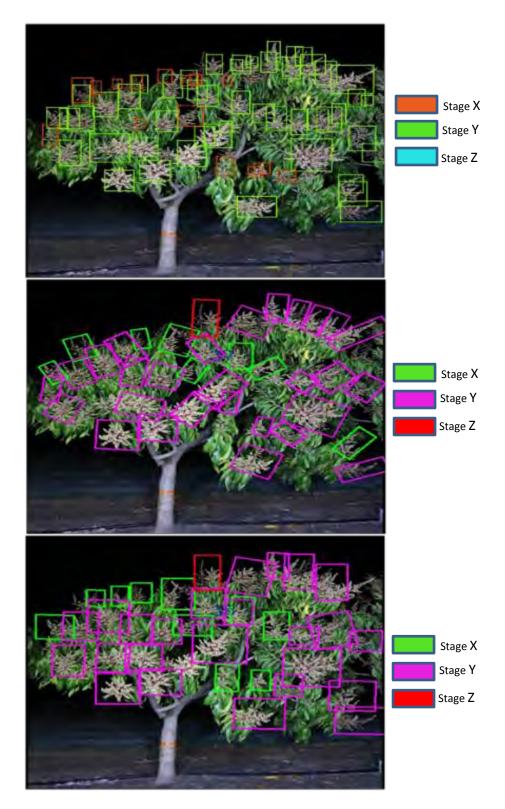
2255

2256

2257

2258

Deep learning- panicle assessment



- Figure 4-6. Example images of panicle stage detection by YOLO and R²CNN methods on Canon images
- 2262of Wang et al. (2018b). Top panel: Panicle stage detection using YOLO method. Orange, green and blue2263coloured boxes represent panicle stages X, Y and Z respectively. Middle panel and bottom panel: Panicle

stage detection using R²CNN and R2CNN-upright methods, respectively. Green, pink and red coloured boxes
 represent panicle classes of X, Y and Z, respectively.

2266 **4.4 Discussion**

2267 4.4.1 Method comparison

2268 The pixel-segmentation method of Wang et al. (2018) outputs the total flower and canopy-like pixels 2269 and does not provide estimation of the number of panicles. A correlation was obtained between 2270 flowering intensity values and panicle counts, consistent with the report of Wang et al. 2271 (2018). However, pixel number per panicle varies with stage of panicle development, and so 2272 confounds number with developmental stage. A stronger correlation is expected when all panicles 2273 are at the same stage of development. The pixel segmentation method also uses a fixed colour 2274 threshold range, but colour of panicles and canopy may vary between cultivars and with growing 2275 conditions, resulting in false positives and negatives. Use of the segmentation method was therefore 2276 discontinued in favour of a deep learning method, echoing the advice of Koirala et al. (2019).

2277 All three methods (YOLO, R²CNN and R²CNN-upright) detected a smaller number of panicles than the 2278 ground truth number (i.e., false negatives occurred). YOLO returned the lowest bias and RMSE for 2279 count of panicles per image, but R²CNN-upright returned the highest mAP and F1 score (Table 4-4 2280 and Fig. 4-5). The difference in ranking of methods is due to the difference in calculation of RMSE as 2281 a 'gross' metric compared to Precision and F1 score, with false negative offset by false positive 2282 detections in the RMSE metric, but not the Precision and F1 and metrics. Therefore, model 2283 performance was primarily assessed on their detection and classification performance using the 2284 mAP and F1 evaluation metrics (Table 4-5).

- The YOLO method (which involves use of upright boxes) produced similar mAP scores to the R2CNNupright method, but the lower weighted F1-score of YOLO. This outcome suggests that the YOLO method suffered from lower recall rates (Table 4-5). This result can be attributed to the deeper CNN classifier (ResNet101) used with R²CNN method.
- 2289 In the current application, the majority of the objects (panicles) were orientated upright, lessening
- the potential advantages for use of rotated annotation boxes for model training. However,
 surprisingly, the R²CNN-upright method achieved better result compared to R²CNN method.
- 2291 Surprisingly, the K Chikeupinght method achieved better result compared to K Chike method 2292 Presumably, this is because an object is considered detected and used for calculating
- the mAP scores only if the detected box sufficiently overlaps with the ground truth bounding box
- 2294 overlaps (to the overlap threshold set in the program). The creation of an upright box from the
- R²CNN created rotated box created a box of larger area, with greater overlap to the annotation box
- and thus a greater number of detections.
- The R²CNN, performed similarly and outperformed the Faster-RCNN method used in a previous
 publication, despite training on images from a different camera and mango cultivar to that in the
 test set (Table 4-5).
- The R²CNN-upright method returned a superior result to the Faster R-CNN in the estimation of total counts per tree. This outcome was expected as R²CNN is a Faster R-CNN framework with the added capability of training on rotated objects (bounding box). Using upright boxes with R²CNN is therefore expected to result in a result equivalent to faster-R-CNN.
- 2304 The speed of the classification by YOLO and R²CNN methods could not be
- 2305 compared because R²CNN supports standard CNN classifier architectures such as ResNet
- and MobileNet while YOLO uses a custom CNN classifier architecture, with no support for standard

- 2307 CNN classifiers. It is expected, however, that YOLO will process images faster than R2CNN as the
- object detection framework of YOLO uses a single-stage detection technique, in comparison to the
- 2309 dual-stage detection technique of R²CNN. Koirala et al. (2019b) documented speed and memory
- requirement comparisons of MangoYOLO CNN architecture used in the YOLO framework of currentstudy benchmarked against several other deep learning object detection frameworks.
- 2312 Fruit image size (in pixels) can vary due to variation in camera to tree distance or use of different
- camera lens. YOLO models can be robust for such conditions as the YOLO object
- 2314 detection framework provides multiscale image training as a part of data augmentation. Other data
- augmentation techniques such as random rotation and random change in hue, saturation and
- exposure of training samples provides robustness for YOLO model to deal with variation in lighting
- 2317 conditions and cultivars. In comparison, R²CNN like Faster R-CNN does not support multiscale
- training, and the only data augmentation during training provided is image flipping (horizontal and
 vertical flips) as. Therefore, the YOLO method of object detection is recommended for applications
- similar to that considered in the current study. Improved classification accuracy for YOLO can be
- expected from use of the full architecture of YOLOv3 (Redmon and Farhadi 2018) rather than using
- 2322 lighter MangoYOLO architecture.

2323 4.4.2 Applications

One of the challenges in precision agriculture is the display and interpretation of large data sets in a form useful to the farm manager, i.e., an appropriate decision support tool for farm management is required. Thus, the presentation of data on panicle count by stage of development for every tree in an orchard requires consideration. A key assessment is the timing of flowering, which can be used in conjunction with calendar days or thermal time to estimate harvest date. This is useful for planning harvest resourcing. Inaccurate harvest timing estimation will result either in harvest of less mature

- 2330 fruit, resulting in lowered eating quality, or over mature fruit, with reduced postharvest life.
- 2331 For example, panicle stage data can be presented on an individual tree basis by week as a line graph
- (Fig. 4-7). This figure enables identification of early flowering areas in one row of an orchard,
- 2333 but it does not scale well to consideration of an entire orchard block.

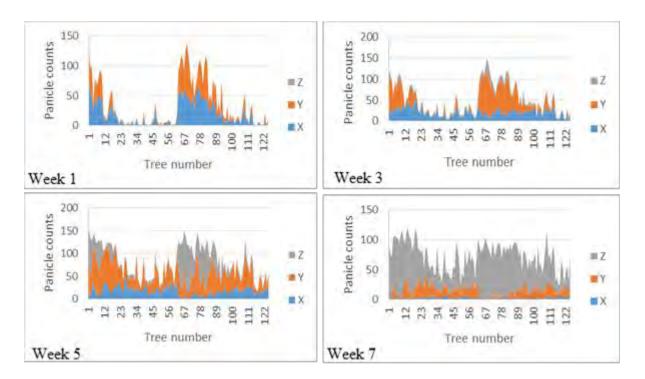
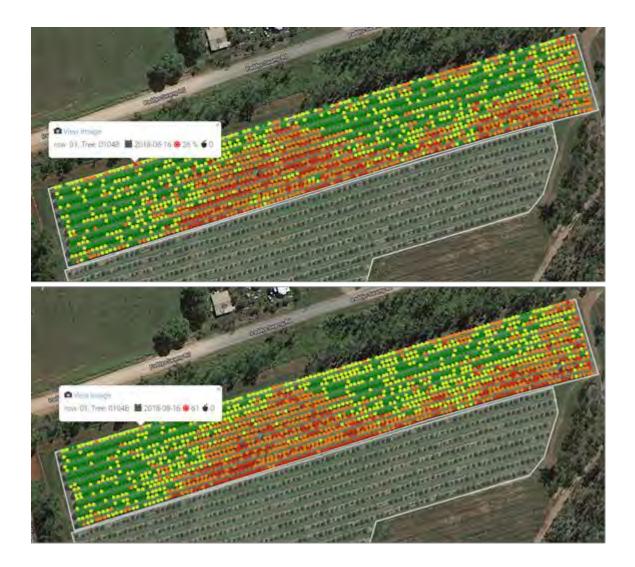


Figure 4-7. Time course (weeks 1, 3, 5 and 7) of panicle number by developmental stage per tree for a row of trees.

2337 Data can also be presented on a farm map, using a colour scale to display values, per week of

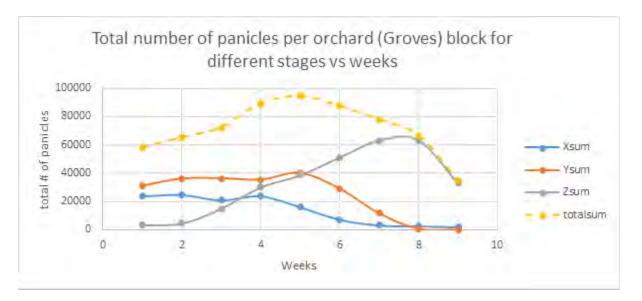
imaging (Fig. 4-8) using the web-app described by Walsh and Wang (2018). In this display, the

2339 individual image associated with each tree can be accessible.



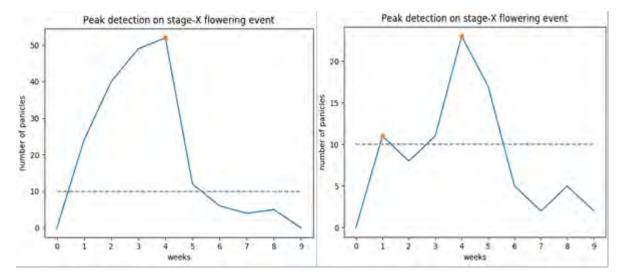
2341Figure 4-8. Flowering intensity level (green, orange and red colour corresponds to low (<30 panicles or 10%</th>2342pixels), medium (30 to 70 panicles or 10 to 25% pixels) and high (>70 panicles or 25% pixels)) (top panel) or2343panicle count (using R²CNN display) (bottom panel) of an orchard. In this software, an individual tree can be2344selected to display the flowering intensity level, tree image, image capture date and tree id.

- Alternatively, data can be summarised for an orchard block, with a tally of panicles by development
- 2346 stage by week (Fig. 4-9). In the example data, there was a shift in developmental stage from
- 2347 stage X to Y to Z over the monitored period, as expected given panicle development. A peak in total
- count occurred at week 5. This profile can be interpreted as a single sustained flowering event
- 2349 maintained over four weeks (weeks 1 to 5).
- 2350



2352 Figure 4-9. Flower stages trend analysis across weeks for an orchard

In another approach, the timing of the peak in panicle number per tree can be assessed forindividual trees given a time course of images and use of a peak detection routine (Fig. 4-10).



2355

Figure 4-10. Peak flowering event detection on stage-X panicle counts for two different trees across 9 weeks of imaging. Single peak (left) and double peak (right) marked with a coloured dot.

A display of the number of trees with peaks in X stage panicle count each week can be displayed to give a sense of when the orchard in general has a peak flowering event. For example, of 1986 images of 993 trees in the block, 168 tree-sides (8%) displayed two flowering events (peaks in stage X)

- 2361 (Fig. 4-11). The first flowering event occurred in the first 2 weeks of imaging while another peak
- flowering event occurred on the fourth week. This information on major flowering events can be
- coupled to temperature records to provide an estimated harvest maturity time based on thermal
- time (Walsh and Wang 2018). If the events are sufficiently large and temporally separated, the
- 2365 grower can consider these as separate harvest events.

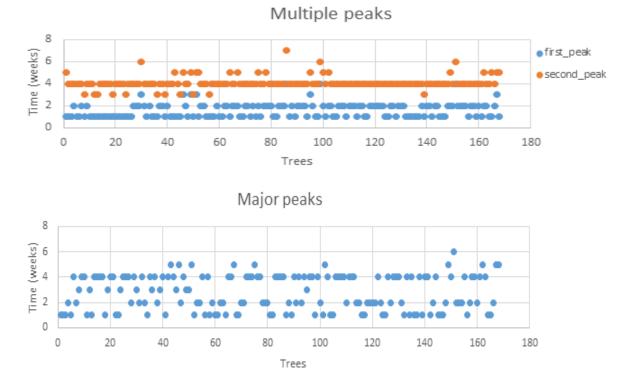


Figure 4-11. Plot displaying week in which a peak flowering event was noted (top panel) and plot displaying
 the week of the largest flowering event (bottom panel) for 168 trees in which two flowering peaks were
 recorded.

Alternatively, data can be presented on a farm map for selective display of spread of panicles per

week of imaging (Fig. 4-12). In this software (Wang and Walsh, 2018), the individual image

associated with each tree can be accessed.



2375	Figure 4-12. Flower stages X to Z (top to bottom panel) selected for one imaging run of week-1 and
2376	corresponding flowering intensity level (yellow, orange and red colour corresponds to low (<30 panicles),
2377	medium (30 to 70 panicles) and high (>70 panicles) display of an orchard.

2379 4.5 Conclusion

- 2380 Deep learning methods can automatically learn the useful features required in an application. For
- panicle detection this is likely to involve colour and shape patterns. Similar to that demonstrated
- by Koirala et al. (2019b) for the fruit detection application, a deep learning method was
- 2383 demonstrated to generalize in terms of application to an orchard of different cultivar, growing
- condition and canopy architecture, also imaged with a different camera, to that used for training
- 2385 images. Similarly, Dias et al. (2018b) reported that a deep learning FCN trained on apple flower
- images was able to generalize well for peach and pear flower detection, and to operate acrosscamera hardware. The use of deep learning models over traditional segmentation techniques for the
- 2388 application of panicle developmental stage detection is thus confirmed.
- 2389 In addition to total panicle counts, classification of panicle into several developmental stages was
- achieved. This is an important step in allowing extraction of information on the time-spread of
- flowering, and this later harvest. Counts of panicles in stage X over time were used in estimation of
- the timing of peak flowering events. Counts of panicles in stage Y demonstrated the highest
- 2393 correlation the flower intensity level per tree.
- 2394 There have been several publications on machine vision based detection and crop load estimation
- for tree fruit crops, but relatively few reports of in-field tree crop flower assessment especially in
- terms of developmental stage. There has been even less attention given to presentation and
- 2397 management of such data. The flower peak detection and display options presented in the current
- 2398 study should prompt further work in this field.

2399 Acknowledgement

- 2400 This work received funding support from the Australian Federal Department of Agriculture and
- 2401 Water, and from Horticulture Innovation Australia (project ST19009, Multiscale monitoring tools for
- 2402 managing Australian tree crops). AK acknowledges receipt of an Australian Regional Universities
- 2403 Network scholarship, and ZW was funded by a CQU Early Career Fellowship. Farm support
- 2404 from Chad Simpson and Ian Groves is appreciated. The assistance of Jason Bell of the
- 2405 CQUniversity High Performance Computing cluster is acknowledged.

2406 Chapter 5. Estimating the unseen – correction for occluded 2407 fruit in tree fruit load estimation by machine vision

This chapter has been revised over the initial thesis submission to incorporate examiner comments. This chapter will be submitted for consideration by the journal 'Computer and Electronics in Agriculture'.

2436 Abstract

2437 Methods to automatically accommodate the proportion of fruits not captured in imaging of trees 2438 from two sides due to occlusions from branches, foliage and other fruits within machine vision based 2439 estimates of fruit load per tree were considered. Five approaches (methods and architectures) were 2440 compared using data of three orchards. Several image features obtained through segmentation of 2441 fruit and canopy areas (such as the proportion of partly occluded fruit) were used in training 2442 Random forest and multi-layered perceptron (MLP) models for estimation of the correction factor 2443 per tree. In another approach, deep learning convolutional neural networks (CNNs) were used to 2444 automatically extract useful information from the input images, with each model directly trained 2445 against harvest fruit count on sample trees (n=98 trees) of all three orchards. On a training set of 2446 2017 season tree images (n=98 trees), a correlation coefficient of determination R² of 0.98 and RMSE 2447 of 18 fruits per tree was achieved between the number of fruits predicted by Random forest model 2448 and the ground truth fruit count on the trees. On a prediction set of 2018 season sample trees 2449 (n=35), a R² of 0.73 and RMSE of 69 fruits per tree was achieved for the Dual_view model, while the 2450 direct prediction models failed. Models were also validated in terms of a comparison of the machine 2451 vision and packhouse fruit counts for the entire orchards. With models trained on images of the 2452 same season, the Random forest model was the most accurate, with an estimation error of 1.6% 2453 across the three orchards (n= 880 trees).

2454 **5.1 Introduction**

2455 5.1.1 Machine vision and 'hidden' fruit

2456 For any crop, yield estimation aids harvest resourcing and market planning. Current practice for 2457 mango fruit yield estimation requires knowledge of previous yield history, visual observation and 2458 manual counting of fruit on trees. However, manual counting is labour intensive, time consuming 2459 and unreliable. For an orchard with 469 trees (Anderson et al. 2018) reported a variation in 2460 prediction error of manual count of fruit in an orchard relative to actual harvest count from 10 to 2461 31% for counts based on 33 and 5% of total trees, respectively. A ~ 27 to 93% coefficient of variation 2462 (standard deviation on tree fruit load divided by mean fruit load) was reported across ten mango 2463 orchards, highlighting the requirement for large number of samples to achieve a reliable estimate of 2464 fruit load per orchard. As the workload of manual fruit counting of such numbers is impractical, 2465 there is a need for an alternative estimation method.

Several researchers have reported on the use of machine vision for tree fruit detection and counting.
A recent review paper by Koirala et al. (2019a) reported high accuracies for deep learning methods
used in detection of fruit in canopy images, e.g., a F1 score of 0.968 was achieved for detection of
fruit in images of mango canopies using a customised deep learning MangoYOLO model.

2470 However, the proportion of fruit on a tree that are captured in an image depends on canopy 2471 architecture. A 'dense' canopy will have a higher proportion of fruit hidden from camera view than a 2472 less dense canopy, with fruit occluded by foliage or other fruit. In sparse canopies, more fruits are 2473 visible, but a given fruit may be seen twice in images from both sides of the tree row, leading to a 2474 double count. For 'dual view' imaging (one image per tree, from both sides of the row), Koirala et al. 2475 (2019b) report variation in the ratio of total fruit on tree established by harvest to machine vision 2476 count (hereafter referred to as the 'occlusion factor') ranging from 1.05 to 2.43, depending on 2477 canopy density. Payne et al. (2013), Wang et al. (2013), Stein et al. (2016) and Koirala et al. (2019b) 2478 utilized the average ratio of harvest count to the machine vision count of fruit on images of a set of

2479 'calibration' mango trees for use in correcting for the proportion of hidden fruits in estimation of2480 orchard yield from machine vision estimates.

2481 The capture of images from multiple viewpoints ('multiview') as a camera is moved past a tree 2482 results in visualization of more, but not all, fruit on each tree (Payne et al. 2013). Fruit count by 2483 multiple viewpoint imaging relies on a method to track and register individual fruit across frames, 2484 but multiple counting of the same fruit can occur if tracking fails (as caused by a sharp movement 2485 between frames, or by obscuration of the fruit in several frames, followed by reappearance). 2486 Multiple counting of the same fruit seen from other side of the row will also occur unless a method 2487 to estimate fruit position is used. Wang et al. (2013) used stereo imaging with a block matching 2488 technique to register fruit in sequential images of apple trees. Moonrinta et al. (2010) used blob 2489 tracking to track fruits from consecutive frames of a video and structure-from-motion to locate 2490 pineapple fruit in 3D. (Stein et al. 2016) used epipolar projection and the Hungarian algorithm for 2491 fruit tracking in sequential images followed by a triangulation method to locate mango fruit in 3D. 2492 Similarly Liu et al. (2018) used the Kalman filter and the Hungarian algorithm for fruit tracking on 2493 video frames followed by structure from motion (SfM) method to locate orange and apple fruit in 2494 3D. Stein et al. (2016) reported on the use of a multiview method in a mango orchard. Although a 2495 high correlation and unity slope was achieved between multiview machine vision and harvest counts 2496 of the mango fruits on trees it was reported that the number of hidden fruits were balanced by the 2497 number of over counted fruits. Such a relationship may not hold for orchards of trees with different 2498 canopy architectures, in which case even the multiview methods require an orchard or tree specific 2499 adjustment for occluded fruit.

The dual view method is simpler and computationally less expensive (only two images per tree) but less accurate (less fruit visible from a single viewpoint) compared to the multiview method which requires fruit detection on multiple images per tree, and fruit tracking. Stein et al. (2016) reported that dual view approach provided higher repeatability, if lower accuracy, for on-tree mango fruit counts compared to multi-view approach (for dual view machine vision count vs harvest estimates, $R^2 = 0.94$ and slope = 0.54, while for multi-view, $R^2 = 0.90$ and slope = 1.01, for 16 trees).

To date, estimation of an occlusion factor (harvest count to machine vision count) has required a manual fruit load estimation of a set of trees representative of the orchard. However, the occlusion factor can vary between trees for a range of reasons, including pruning history, irrigation and nutrition (related to canopy foliage density). Therefore, the selection of 'representative' trees in estimating an occlusion factor for the orchard is essential, and the error in the estimation of the orchard occlusion factor represents a limitation in the application of machine vision to fruit load estimation.

2513 **5.1.2 Dealing with occluded fruit**

Ideally, the occlusion factor should be estimated from canopy images features for each tree in an
orchard (Koirala et al. 2019a). It is likely that the features within canopy images hold clues to the
proportion of hidden fruit on the tree. For example, image characteristics such as the number of leaf
intersections or the ratio of partially occluded fruit to fully revealed fruit may hold information on
the proportion of occluded fruit.

Several reports have appeared of direct prediction of tree yield, i.e., with training against the
reference value of total tree fruit number or weight, rather than against number of fruits seen in
images. For example, Črtomir et al. (2012) trained ANN models with 4-6-1 and 5-14-1 architectures
for 'Golden Delicious' and 'Braeburn' varieties of apple, using the inputs of fruit counts obtained
from an image segmentation technique of canopy images and harvest fruit weight per tree. For

2524 single-view image of super spindle trees, the correlation (r) between model predicted and actual

- yield was improved from 0.73 to 0.83 and from 0.51 to 0.78 for 'Golden Delicious' and 'Braeburn'
- varieties, respectively, for the linear regression of fruit counts from image and direct prediction of
- 2527 total fruit load per tree.

2528 Cheng et al. (2017b) trained an ANN (4-11-1 architecture) model using image features (fruit area, 2529 fruit cluster area and canopy leaf area) along with fruit number from single-view image to predict 2530 apple var. Gala yield (kg/tree). The canopies were trained as slender spindles with 3.5×1.5 m 2531 spacing. An R² 0.82 and RMSE of 2.3 kg/tree was achieved for a test set for the season. For the 2532 Pinova variety, an ANN (4-10-1 architecture) achieved R² of 0.88 and RMSE of 2.5 kg/tree for a test 2533 set. In a parallel approach, Qian et al. (2018) trained an (4-14-1 architecture) ANN model with four 2534 image features (total fruit pixel area, circle fitted fruit pixel area, average radius of fitted circle and 2535 residual fruit pixel area after circle fitting) from dual view images of apple trees to predict individual 2536 tree yield (kg/tree). The model, trained on images of 21 trees and validated on images of five trees, 2537 achieved an R^2 of 0.996 and RMSE of 1.0 kg/tree on the training set and R^2 of 0.94 and a RMSE of 2.4 2538 kg/tree on the validation set.

2539 ANN regression can also be used with the CNN models for processing image features. Chen et al. 2540 (2017) developed a fruit counting pipeline based on deep learning to estimate fruit number in tree 2541 canopy images. A deep learning Fully Convolutional Network (FCN) blob detection network was 2542 trained for pixel-wise segmentation of fruit regions in images. This step was followed by a linear 2543 regression between the detected blobs and harvest count to estimate the final yield (total fruit 2544 count/tree) from the tree image. this study involved non-trellised orange imaged in daylight and 2545 trellised apple trees imaged at night using flash illumination. A ratio of harvest count to machine 2546 vision estimates (blob+count+regression model) of 1.03 and 1.10 was obtained for the orange and 2547 apple data sets, respectively. In comparison, this ratio was much higher for mango trees with 2548 traditional tree architectures, as reported above. Denser canopieshide more fruits and are more 2549 challenging for machine vision yield estimation. Thus, estimation of the occlusion factor is 2550 particularly important for estimation of mango crop load.

2551 In the current study, several machine learning and deep learning methods were explored for total 2552 tree fruit load estimation based on dual view imagery, i.e., avoiding the use of occlusion factor for 2553 hidden fruits based on manual counting of fruit in sample. Five models (Dual_view, Deep_yield, 2554 MLP yield, Random forest, Exception yield) were trialled for estimation of fruit load per tree based 2555 on input of dual view mango tree images (Table 5-1). MangoYOLO (Koirala et al. 2019b) was used for 2556 fruit detection and counting of visible fruit on tree images, except for the Xception_yield model. 2557 Models used in the study are tabulated with their implementation purpose (Table 5-1). "Yield 2558 estimation" in this study refers to the estimated total number of fruit for a group of trees, based on 2559 visible fruit counted using MangoYOLO adjusted t using an occlusion factor as in the dual_view 2560 model or through automated adjustment from trained models for number of hidden fruits in dual

view images of a tree or blocks of trees.

2562 Table 5-1. Fruit counting and yield estimation models and their purpose

Model name	Purpose
MangoYOLO model	Automated fruit detection and counting on tree images based on bounding-box training.

Xception_count model	A model that learns fruit distribution pattern on input tree images based on CNN-regression modelling to be utilized for canopy classification.
Xception_classification model	Automated classification of tree images into 3 categories (low, medium and high) based on fruit distribution and canopy architecture.
Dual_view model	Yield estimation based on automated counting of visible fruits using MangoYOLO model and adjusted using a correction/occlusion factor (average ratio of harvest fruit count to image fruit count for sample trees).
Deep_yield model	Automated yield estimation based on fruit numbers and canopy classification of two images, one image per sides of a tree.
MLP_yield model	Automated yield estimation based on input parameters obtained from canopy and fruit region extraction and fruit counts (fully visible and partly occluded determined through ellipse fitting) using MLP neural network.
Random_forest model	Automated yield estimation based on input parameters as for MLP_yield model but utilizing an ensemble of decision trees for regression.
Xception_yield model	Automated yield estimation based on a method similar to xception_count model but extracting canopy and fruit regions of two sides of a tree into a single image as input to the model.

5.2 Materials and Methods

2565 **5.2.1 Hardware**

Images of two sides of each tree were acquired at night time with a 5 Mp Basler acA2440 RGB
camera and a 720 W LED light panel mounted on a 5 km/h moving platform, as detailed in (Koirala et al. 2019b).

All models were compiled and run on a Red Hat Enterprise Linux Server 7.4 machine with 384GB RAM and NVIDIA® Tesla® P100 GPU (16 GB memory). CUDA v9.0, cuDNN v7.1.1, OpenCV-python v4.0.0.21, Python v2.7.14, Keras v2.2.0, Scikit-learn v0.19.1 and Tensorflow v1.8.0.

2572 5.2.2 Orchard information

2573 Images were acquired of Mangifera indica cv. Calypso trees in orchards on a farm located at (lat, long -25.144, 152.377) Queensland, Australia on 7 and 8th of December 2017, and 13 and 14th of 2574 January 2018 (Table 5-2). The orchards varied in row spacing between 9.5-12 m, with tree spacing 2575 2576 along rows at 4 m. Canopy width was approximately 4 m. The total number of trees in orchard A, B 2577 and C were 494, 121 and 265 respectively. Sample trees in each block were hand harvested for fruit 2578 count in both seasons (Table 5-2). An extended number of sample trees were hand harvested in the 2579 2017 season, resulting in 44, 19 and 35 sample trees for orchard A, B and C (termed A-x, B-x and C-x, 2580 respectively) (Table 5-2).

2581	Table 5-2. Statistics on the harvest count of sample trees (fruit per tree) for each of two seasons. ABC
2582	represent the collection of sample trees from orchards A, B and C.

		2017		2018	
Orchard	Sample tree #	Mean	SD	Mean	SD
A	17	207	86	128	99
В	6	279	148	205	134
С	12	148	75	274	126
ABC	35	199	103	191	130
A-x	44	187	76	-	-
B-x	19	253	160	-	-
C-x	35	171	90	-	-
ABC-x	98	194	105	-	-

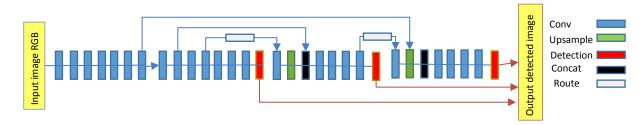
2584 5.2.3 Fruit counting and canopy classification

2585 5.2.3.1 MangoYOLO

2586 MangoYOLO (Koirala et al. 2019b) is a deep learning CNN fruit detection and localization model 2587 optimized for speed, computation, and accuracy through re-designing of YOLO object detection 2588 framework. MangoYOLO model detects mango fruit and draws bounding boxes on the detected 2589 fruits on tree images. The number of bounding boxes provides fruit number per image.

2590 Network architecture

2591 MangoYOLO consisted of total 33 layers (3 detection, 2 route, 2 up-sample and 26 convolutional 2592 layers) as shown in figure 5-1.



2593

2594 Figure 5-1. Block diagram of MangoYOLO model

2595 Training method

2596 MangoYOLO model (pre-trained on 1300 images containing 11820 mango fruits) was adopted from 2597 Koirala et al. (2019b) and implemented with the OpenCV-python v4 (added support for YOLOv3 2598 implementation). The class confidence and NMS thresholds for MangoYOLO model were set to 0.24 2599 and 0.45 respectively.

2600 5.2.3.2 Xception_count model

Xception_count model learns the fruit distribution pattern on images through training a CNN regression model on the input tree image against target fruit count (obtained from MangoYOLO) per

- tree image. The intermediate output from the trained Xception_model was then utilized for
- classifying tree images into several categories based on the learned fruit distribution pattern.
- 2605 *Network architecture*

2606 'Xception' (Chollet 2017) is a deep learning CNN model which has an input resolution of 299 × 299
 2607 pixels. The base (without the final classification layers) of the Xception model was imported from

- 2608 Keras deep learning library and used inside the Xception_count model. The input resolution of the
- 2609 Xception model was changed to accept input images of 1024 × 1024 pixels. A regression head which
- 2610 was comprised of four layers was added to this base model and a global spatial pooling layer
- 2611 (GlobalAveragePooling2D) was used on the output of the base model (Fig. 5-2). The pooling layer
- was followed by a connected (Dense) layer consisting of 1024 neurons, 'relu' activation function and
 a Dropout (0.65) layer. Dropout (Srivastava et al. 2014) helps in preventing neural networks from
- 2614 overfitting by randomly dropping out the fraction of neuron inputs to 0 at each update during
- training. The final layer is a dense layer with a 'linear' activation function.

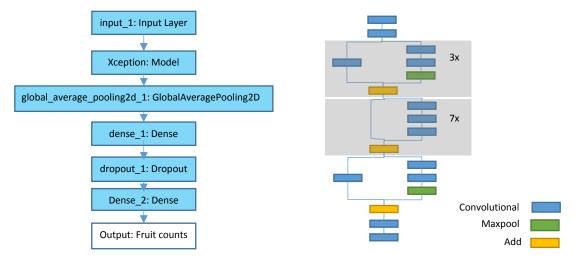


Figure 5-2. Xception_count model (left) and schematic of Xception base model (without top classification layers) (right)

2619 Training method

2620 The Xception count model was trained on 988 mango tree canopy images of 494 tress (i.e. images 2621 of two sides of each tree) from orchard A in December 2017. The original images (2448x2048 pixels) 2622 were resized (1024x1024 pixels) as input variables to the Xception_count model. Normalized fruit 2623 counts on each image obtained from the MangoYOLO model (Koirala et al. 2019b) was used as 2624 target value for model training. The Xception count model was initialized with random weights, i.e., 2625 no transfer learning was implemented. Training consisted of 50 epochs, using a batch size of 2 and a 2626 learning rate of 1e⁻⁴. The Xception count model was compiled with the MSE (Mean Squared Error) 2627 loss function and 'adam' (Kingma and Ba 2014) optimizer. MSE is a commonly used loss function for 2628 regression modelling and is computed as the mean of the squared difference between estimated 2629 and actual values.

2630 5.2.3.3 Canopy categorization/classification

The feature vector from the intermediate layer of the trained Xception_count model was utilized to classify/categorize canopies having similar fruit patterns into different clusters with the aim of accurately estimating total fruit in the tree from what is seen on the two sides. K-means clustering was applied on the output vectors from the 'global_average_pooling2d_1' layer of the Xception_count (Fig. 5-2) model (input size 1024x1024 pixels) for whole orchard A 2017-season tree images.

The K-means algorithm clusters data by separating samples to n groups of equal variances. Since there was no ground truth category/label for the images being clustered, the Silhouette coefficient (Rousseeuw 1987) was calculated to assess the usefulness of feature vector for segregating canopies

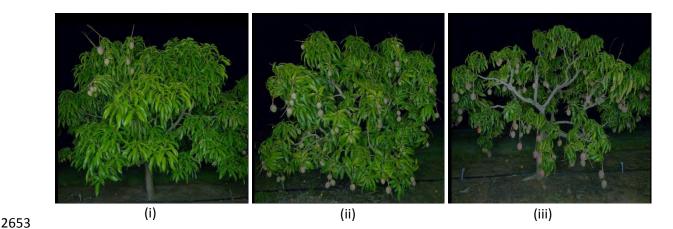
Estimating the unseen

- 2640 into clusters. The Silhouette value is a measure of similarity of an object to its own cluster compared
- to other clusters. Silhouette coefficient is calculated from the measure of mean distance between a
- sample and (i) all other points in the same cluster, (ii) all other points in the next nearest cluster.
- 2643 The Silhouette coefficient varies in range (-1, 1), with higher scores for better defined clusters.
- 2644 Default parameters from the sklearn library were used for both k-means clustering and Silhouette
- 2645 metrics calculation. The number of clusters was varied from 2 to 5 (Table 5-3). For further work,
- three clusters were chosen on all blocks because of a relatively good Silhouette coefficient and
- 2647 relatively balanced distribution on canopy number (Table 5-3). There was visual difference in the
- images of three clusters/categories (Fig. 5-3): (i) very less or even no fruit (more vegetation), (ii)
 medium number of fruits more uniformly distributed around the canopy (iii) large number of fruits
- and open canopies.

2651	Table 5-3. Categorization of tree canopy images from each block into three clusters
------	---

Orchard	Silhouette score	#trees in cluster1	#trees in cluster2	#trees in cluster3
A	0.4311	443	344	211
В	0.4622	62	104	76
С	0.4982	175	242	113

2652



- 2654 Figure 5-3. Example images from the three categories based on Silhouette score.
- 2655 Canopy classification models

2656 5.2.3.4 Xception_classification model

A trained Xception_classification model can classify input tree images into 3 categories (low,medium and high) based on fruit distribution pattern. Output from Xception_classification model

- 2659 serves as input for yield estimation models used in this study.
- 2660 Network architecture

The Xception_classification model is of the same architecture as Xception_count model (Fig. 5-2) with following changes:

Dense_2 layer of Xception_classification model consisted of 3 neurons for 3 canopy
 categories/classes and 'sigmoid' activation function compared to 1 neuron and 'linear'
 activation function for predicting continuous values in Xception_count model.

- 2666 Xception_classification model was compiled with 'categorical_crossentropy' loss function
 2667 compared to MSE loss function in Xception_count model.
- Cross-entropy is a commonly used loss function for multi-class classification task. Cross-entropy is
 based on the maximum likelihood (probability distribution across multiple classes). This function
 tries to minimize the mean difference between the actual and estimated probability distributions for
 all classes considered.
- 2672 Training method

The Xception_classification model was trained using the tree images of orchard A as input and corresponding categories (1-3) obtained from the k-means clustering algorithm as ground truth labels.

The Xception base of Xception_classification model was initialized with the learned weights from the Xception base of Xception_count model. Finally, the Xception_classification model was trained on images (1024 ×1024 pixels) of the 3 canopy categories for 50 epochs with batch size=2 and learning rate =1e⁻⁴. For training and validation, the training image data orchard A (all 494 trees) was split into 90% for training and 10% for validation.

2681 5.2.4 Canopy and fruit region extraction

2682 *Canopy extraction*

2683 After the fruit regions were extracted from the bounding box coordinates returned as detection by 2684 MangoYOLO model, a colour segmentation technique followed by contour fitting was used to 2685 extract the foliage of the canopy into a blank image. Images were converted from BGR to HSV range 2686 using the OpenCV function. Colour segmentation was done by selecting a range of green colour (HSV 2687 range lower = [33, 80, 40] and upper = [102, 255, 255]) followed by morphological operations 2688 (conversion to grayscale, median blurring, adaptive thresholding (mean) and closing). In the 2689 resultant binary image, OpenCV's contour fitting function was used to obtain the contour of the 2690 foliage from the image (Fig. 5-4).

2691 *Shape fitting*

Mango fruit shape can be approximated using an ellipse fit (Charoenpong et al. 2004; Kader 1997;
Nanaa et al. 2014). Wang et al. (2017b) and (Wang et al. 2018a) reported on the use of ellipse fitting
technique to discriminate fully exposed mango fruit from the partly occluded fruit in images of tree
for fruit size estimation. A similar approach was implemented in the current study.

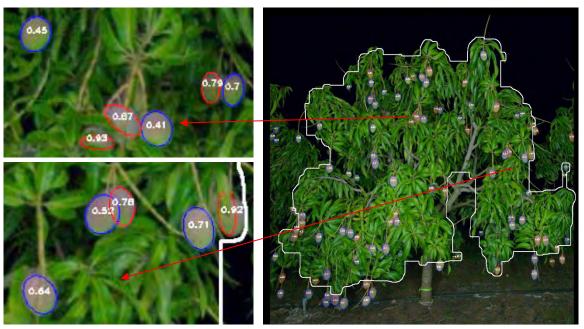
Each input image was processed using MangoYOLO for detection of the fruit region. Each Rol (region inside bounding box) was individually processed for shape fitting. ROIs were converted to grayscale and sharpened (to highlight the fruit edges) using OpenCV functions. The resultant image was converted to binary image using the adaptive thresholding (mean) algorithm followed by

- 2700 morphological closing operation. OpenCVs contour fitting function and ellipse fitting algorithm was
- used to fit an elliptical shape on the binary image.
- 2702 *Creation of training data for MLP_yield and Random_forest models*

2703 It was observed that the ellipses fitted to partially occluded fruits were more eccentric compared to

- fully visible (entire) fruit (Fig. 5-4). An eccentricity value of 0.75, asdetermined from visual
- 2705 inspection of images, was used as a threshold to discriminate fruit as partially occluded. The area of

- ellipses for full and partly occluded fruit were summed for each tree image. Similarly, the canopy
- area was calculated as the pixel area of the contour in tree images.



- Figure 5-4. Ellipse fitting and canopy extraction. Right panel: Area enclosed by the white contour around the tree represents the canopy foliage area. Left panel: Fruit enclosed by blue ellipse contour and red ellipse contour represent the fully exposed fruit and partly occluded fruit respectively.
- 2712 The eccentricity threshold may vary for cultivars which differ in fruit shape.
- 2713Table 5-4. Description of input variables used for MLP_yield and Random_forest models. Subscripts a and b2714represent data for side A and side B images of a tree, respectively.

Attributes	Description
cT _a , cT _b	count of total fruit on image from MangoYOLO model
cFa, cFb	count of exposed (fully visible) fruit
cO _a , cO _b	count of partially occluded fruit
RpFa, RpFb	ratio of total pixel area of exposed fruit to the canopy pixel area
RpO _a , RpO _b	ratio of total pixel area of partially occluded fruit to the canopy pixel area
RpC _a , RpC _b	ratio of canopy pixel area to the total image pixel area

- 2716 *Creation of training data for exception yield model*
- 2717 For each side of the tree images of the training data (images of sample trees orchard ABC-x), fruits
- 2718 and canopies were segmented and placed besides each other in a new single image (Fig. 5-5) which
- created a new dataset of images. This process resulted in half the number of images in the newly
- 2720 reconstructed dataset.
- The MangoYOLO model was used to detect fruits on the images and the coordinates of the detected bounding box were used to extract the RoI (fruit regions) into a blank image.
- 2723 Similarly, the canopy extraction contour was used as a mask to segment the foliage into a blank
- image (Fig. 5-5). Finally, the four images of extracted fruit and canopy regions (2448 × 2048 pixels
- each) were aligned into a single image (4896 × 4096 pixels) and resized (1024 × 1024 pixels) for input
- to the Xception_yield model.

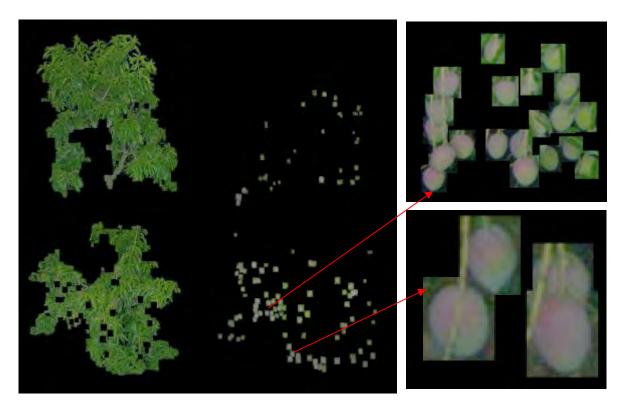


Figure 5-5. Left panel – a single image reconstructed of two sides of a canopy with fruits and canopy separated). Right panel – closeup view of the fruit cluster.

- 2730
- 2731
- 2732

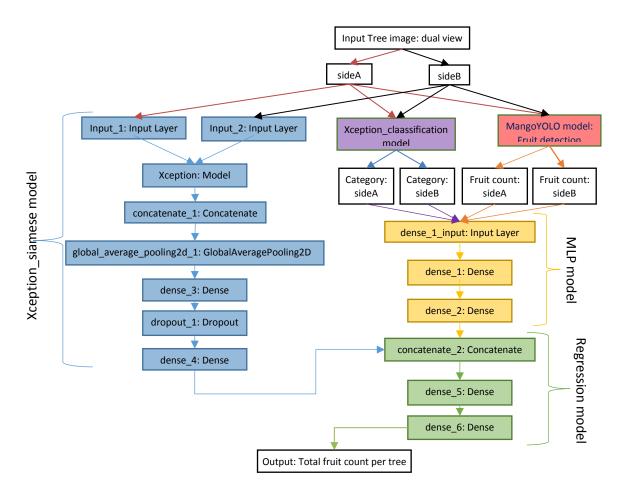
2733 5.2.5 Yield estimation

2734 5.2.5.1 Dual_view model

In order to estimate total fruit per tree, the dual_view model relied on use of a manually established
occlusion factor to adjust the fruit counts from MangoYOLO model. This occlusion factor is the
average ratio of what is counted by the machine vision (MangoYOLO) method on tree images to a
manually obtained fruit count for a set of 'calibration' trees.

The following machine learning models were created to directly predict the total fruit number per tree from the input of dual-view images. Since each yield estimation model was comprised of several small modules (different architecture, different input data types and different input resolutions), the training parameters vary across models. These parameters were tuned to obtain best results while also considering the available computational resources (e.g., GPU memory).

- 2744 5.2.5.2 Deep_yield model
- 2745 *Network architecture*
- 2746 The Deep_yield model was composed of 5 networks- MangoYOLO, MLP, Regression,
- 2747 Xception_siamese and Xception_classification (Fig. 5-6).



2749 Figure 5-6. Block diagram of the Deep_yield model

The MLP block: The Multi Layered Perceptron (MLP) consisted of a stack of multiple Fully Connected
(FC) layers of neurons (8-4 architecture). The input layer (dense_1) contained 8 neurons (6 for the 2
category vectors each having 3 elements obtained for each side of tree images from

2753 Xception classification model, and 2 neurons for the fruit count on two sides of image obtained

from MangoYOLO model). The final layer (dense_2) consisted of 4 neurons to match the number of

- 2755 input nodes (i.e., the output of the Xception_siamese network) in the model. An activation function
- 2756 'relu' was used for both dense layers.

2757 The Siamese block: With a Siamese network it is possible to train a single model for multiple inputs. 2758 The Xception_siamese network takes RGB input images (299 × 299 pixels) for each side of a tree as 2759 input to the Xception model (Fig. 5-6). The outputs for each image side from Xception model of the 2760 Xception siamese network were concatenated (concatenate 1 layer) and global average spatial 2761 pooling operation applied. The data was further processed through Fully Connected (FC)/dense 2762 layers- dense 3 and dense 4 having 1024 and 4 neurons respectively. A dropout layer (dropout 1) 2763 with dropout value =0.65 was used before the final layer to prevent the model from overfitting. All 2764 the dense layers used 'relu' activation function. Final layer (dense 4) consisted of 4 neurons to

2765 match the output nodes of the MLP model.

2766 The Regression block: The regression model consisted of 1 concatenation layer and 2 FC dense layers

- 2767 (Fig. 5-6). The concatenation layer (concatenate_2) concatenated the outputs of Xception_siamese
- and MLP models. The dense layer (dense_5) with 'relu' activation function consisted of 4 neurons to
- 2769 match the number of input nodes. The final layer (dense_6) consisted of 1 neuron and a 'linear'
- 2770 activation function to predict continuous data (total fruit counts per tree).

2771 Training method

- 2772 For training and validation, the training data (images of sample trees orchard ABC-x) was split into
- 80% for training and 20% for validation. Set ABC-x contains more images than the individual blocks A
 or B or C therefore number of images in the validation set was increased from 10% to 20%.
- 2775 The Deep_yield model was trained for 200 epochs with learning rate=1e⁻³, loss function =MSE,
- optimizer=Adam, batch size=8. The Xception network inside Xception_siamese model was initialized
 with pre-trained ImageNet weights available from the Keras library. For model training, the target
- 2778 values (harvest count) were normalized by dividing with the maximum harvest count value.

2779 5.2.5.3 MLP_yield model

- The MLP_yield model was used in predicting the total fruit number per tree form the count of fruit on the two sides of tree image with additional features (number of exposed (fully visible) and partially occluded fruit, canopy foliage area (pixel), fully visible fruits total area (pixel), partially occluded fruit total area (pixel) using MLP regression.
- 2784 Network architecture
- 2785 The MLP_yield model consisted of a stack of multiple Fully Connected (FC) layers of neurons (12-6-1
- architecture). The input layer (dense layer_1) consisted of 12 neurons to match the number of input
- variables. The intermediate layer (dense layer_2) consisted of 6 neurons. Both first and second
- 2788 layers used 'relu' activation. The final layer (output_layer) consisted of a single neuron to predict
- 2789 fruit counts and a 'linear' activation function was used.
- 2790 Training method
- 2791 The MLP_yiled network takes 12 input variables (6 for each image side of a tree- cT, cF, cO, RpF, RpO
- and RpC, Table 5-4). Fruit count data (cT_a, cF_a, cO_a, cT_b, cF_b, cO_b) and target ground truth harvest
- count per tree were normalized by dividing by 200 and 600 respectively. This value was obtained as
- the maximum values that can be expected for the training dataset.
- All 3 layers of MLP_yield model was initialized using 'uniform' weights and trained for 200 epochs
 with a batch size of 4. The model was compiled with MSE loss and Adam optimizer with learning rate
 of 1e⁻³. For training and validation, the training data (images of sample trees orchard ABC-x) was split
 into 80% for training and 20% for validation.
- 2799 5.2.5.4 Random_forest model
- The Random_forest model was used for estimating total fruit number per tree based on inputparameters same as MLP_yield model but utilizing ensemble of decision trees for regression.
- 2802 Network architecture
- 2803 Random forest (Breiman 2001) is a simple yet accurate machine learning algorithm used for
- classification and regression tasks (Liaw and Wiener 2002). It builds 'forests' from an ensemble of
- 2805 decision trees to increase predictive accuracy. Random forest adds randomness while splitting a
- 2806 node by searching for best feature from a random subset of features rather searching for most
- 2807 important features from bigger set which helps control over-fitting of the model.
- 2808 Training method

Estimating the unseen

- 2809 RandomForestRegressor from 'sklearn' library was used with default values (max_depth =None,
- n_estimators=100, criterion='mse') for model training on the same training data used by the
 MLP_yield model.
- 2812 5.2.5.5 Xception_yield model
- 2813 Network architecture
- 2814 The architecture of the Xception_yield model is the same as that of the Xception_count model (Fig.
- 5-2) but takes reconstructed image (canopy and fruit regions of both sides of a tree extracted on to asingle image) as input.
- 2817 Training method
- 2818 Since the newly reconstructed image dataset contain a smaller number of training images, this
- 2819 dataset was split into 90% for training and 10% for validation. Model was compiled with MSE loss,
- Adam optimizer and trained for 50 epochs with batch size of 2 and learning rate of 1e-4. Transfer
- learning was not used and the weights for the CNN model was initialized at random values.

2822 5.3 Results and Discussion

- 2823 In this study 'yield estimation' refers to the prediction of the total number of fruits on a group of
- 2824 mango trees based on machine vision count of fruits in dual-view images of individual trees,
- adjusted for hidden fruits through use of manual correction factor or automated model training.
- 2826 Therefore, RMSE and R² values from regression analysis were used as major metric for evaluation
- 2827 different models used in the study.

2828 5.3.1 Fruit counting

- The deep learning models e.g. Xception_count model can be used for direct prediction of fruit count on images without a need for bounding box annotation as used for training MangoYOLO but not as accurate (Tables 5-5, 5-6).
- 2832Table 5-5: Linear regression statistics for fruit counted on tree images of orchard A (17 trees, 34 images 20182833season) by Xception_count model and MangoYOLO model against human count of fruit on images.

		Xception_coun	MangoYOLO model 2018					
Orchard	Slope	Intercept	R ²	RMSE	Slope	Intercept	R ²	RMSE
А	0.715	13.69	0.93	10.1	0.915	0.08	0.98	5.3

2834Table 5-6. Linear regression statistics for fruit counted on tree images for orchard A (all 988 images of 4942835trees) by Xception_count model versus MangoYOLO model for 2 seasons.

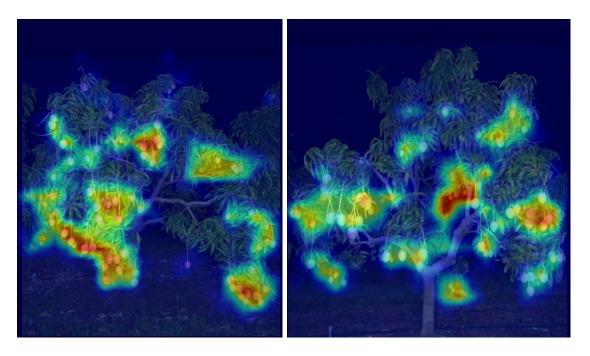
2017						2018				
Orchard	Slope	Intercept	R ²	RMSE	Slope	Intercept	R ²	RMS		
А	0.731	12.32	0.96	9.12	0.728	19.81	0.94	11.8		

2836

2837 5.3.2 Visualization

- The class activation map was visualized on the final convolutional layer (block14_sepconv2_act) of the trained Xception_count model to indicate the regions in images utilized by the model for prediction of fruit numbers. The Grad-CAM (Selvaraju et al. 2017) visualization technique revealed
- that the model was indeed activated by the fruit regions rather than other objects (Fig. 5-7).

Estimating the unseen



2842

Figure 5-7. Grad-CAM visualization of activation map of the final convolution layer of Xception_count model trained to directly predict fruit count on input images

2845 **5.3.3** Canopy categorization

The Xception_classification model was capable of classifying tree images to one of the categories it was trained for (Table 5-7).

Table 5-7. Classification results for orchard A (all 988 images of 494 trees) 2017 season used during training

		Predicted categories from Xception_classification model						
		Cat 0	Cat 1	Cat 2				
Ground truth	Cat 0 (443)	387	51	5				
categories from k-	Cat 1 (334)	2	332	0				
means clustering	Cat 2 (211)	13	0	198				

2849

The model trained on orchard A achieved precisions of 80%, 99% and 94% for classifying images in category 1, 2 and 3 respectively, on the training set orchard A. The Xception_classification model (trained on orchard A images 2017 season) was further tested on orchard C images 2017 season (Table 5-8). The model trained on orchard A achieved precisions of 65%, 88% and 97% for classifying images in category 1, 2 and 3 respectively, on the test set orchard C.

2855Table 5-8. Classification results for orchard C (all 530 images of 265 trees) 2017 season from the model2856trained on orchard A images (all 988 images of 494 trees)

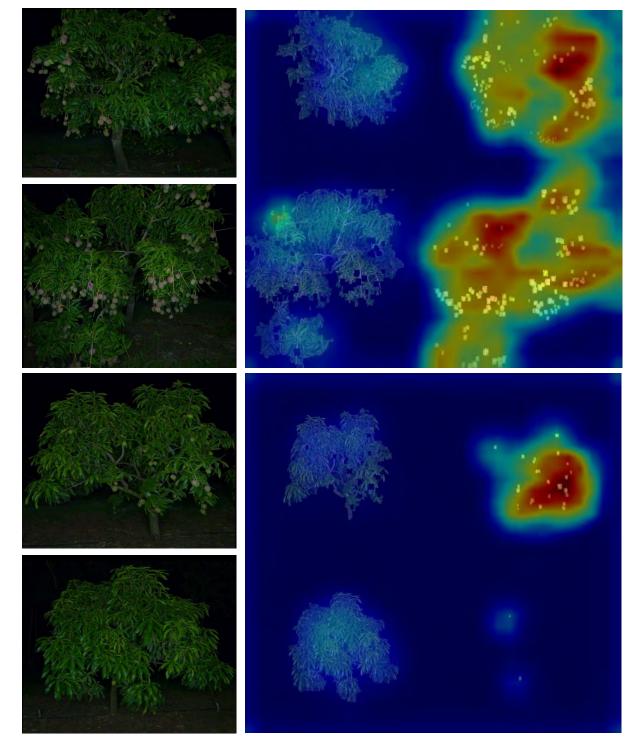
Predicted categories from Xception_classification model Cat 0 Cat 1 Cat 2 Ground truth Cat 0 (175) 114 18 43 categories from k-Cat 1 (113) 14 99 0 means clustering Cat 2 (242) 8 0 234

2858 **5.3.4 Feature importance**

2859 While the deep yield models are essentially black boxes, it is useful to gain some insight into the

attributes used, the better to train such models and to anticipate prediction failures. The
visualization of Xception_yield model showed that the model used the fruit regions in the image for

2862 prediction of the total fruit count in the canopy (Fig. 5-8).





2864

Figure 5-8. Grad-CAM visualization of the activation map of the final convolutional layer of Xception_yield model (trained to directly predict tree fruit count from input images). Left and right sub-panels present the raw images of two sides of a tree and activation heat-maps of canopy and fruit regions, respectively. Top

panels: Typical result for tree with fruits on both sides of the canopy. Bottom panels: Typical result for tree with fruits on one side of canopy.

- 2870 The Random forest regression model for prediction of harvest count placed higher weight on fruit
- count than the other canopy and fruit features (Table 5-9). The occluded fruit count (cO) wasweighted more than counts of fully visible fruits (cF).

2873Table 5-9. Feature_importance values (sums up to 1) returned by the Random Forest regressor on the2874different input variables. Refer to Table 3 for the description of variables. The variable with highest2875weighting (total fruit count) is shown in bold.

	cTa	cFa	cOa	R pF _a	RpO _a	RpC _a	cTb	cFb	cOb	RpF _b	RpO _b	RpC _b
	0.26	0.07	0.08	0.01	0.01	0.07	0.24	0.06	0.10	0.01	0.01	0.07
2876	The nu	mber of	partly oc	cluded fi	ruit was e	estimate	d throug	gh the el	ipse fitti	ng techn	ique. Of	all fruits
2877	visible	in tree in	29, nages	9 and 37	% were p	partly oc	cluded i	n the two	o sets co	nsidered	, equival	ent to 21
2878	and 31	% of tota	al fruits p	per tree,	respectiv	vely (Tab	le 5-10)	. A corre	lation R ²	>= 0.64)	was obse	erved

2879 between the number of partly occluded fruit on tree images and the harvest fruit count (Table 5-10).

2880Table 5-10. The correlation between number of partly occluded fruit to the total fruit count on images and2881to harvest count per tree.

Partly occ. vs. harvest count					Partly oc	Partly occ. vs. machine vision count			
Image set	R ²	Slope	Intercept	Ratio	R ²	Slope	Intercept	Ratio	
ABC-x	0.69	0.17	5.41	0.21	0.93	0.37	-0.19	0.37	
ABC- 2018	0.64	0.09	5.92	0.31	0.88	0.28	-0.29	0.29	

2882

The ratio of exposed fruit (cFa +cFb) to the harvest count was correlated to the proportion of hidden fruit (harvest – cTa –cTb) to the harvest count for both the seasons (Table 5-11). However, this result was of no practical significance for yield estimation as the relationship requires information on the

2886 harvest count of individual trees considered.

Table 5-11. Correlation between hidden and exposed fruits for 2 seasons obtained from MLP_yield model. Units are fruit number per tree.

Hidden f	ruit vs exposed	fruit orchard	Hidden fruit vs exposed fruit orchard			
	ABC-x 2017	7	ABC-2018			
Slope	Intercept	R ²	Slope	Intercept	R ²	
-1.3299	0.9097	0.8943	-0.7051	0.7098	0.9074	

2889

For the same orchard A used in this study, Stein et al. (2016) observed almost no relationship between the canopy volume (estimated as LiDAR voxel count) and yield (fruit count per tree). (Anderson et al. 2018) also reported a poor correlation (R² = 0.21 and 0.17, respectively) between canopy attributes (canopy volume and trunk circumference) and fruit load. These observations are consistent with the low weighting assigned to the attributes related to canopy size (e.g., RpC).

2895 5.3.5 Sample tree yield prediction

2896The comparison of number of fruits that is captured on dual view images of trees and counted using2897machine vision model (MangoYOLO) to the actual (harvest) fruit number per tree is presented in2898Table 5-12. Although a good correlation ($R^2 >= 0.69$) was achieved between machine vision count2899and harvest count, the low values of slope indicates there is a large proportion of fruits that are not2900visible in dual view imaging (Table 5-12)

2901Table 5-12. The linear regression statistics between machine vision (MangoYOLO) fruit count (without using2902occlusion ratio for correction) on images to the harvest count per tree.

Test set	Avg. #fruit per tree	St. Dev	Slope	Intercept	R ²	RMSE	Bias
ABC-x 2017	194	105	0.44	17.7	0.69	113.4	-91.8
ABC 2018	191	130	0.32	17.3	0.73	143.7	-112.0

2903

2904 The dual view method was compared to four methods for direct yield estimation in prediction of the

training set and test set (Table 5-2) (images of 2017 and 2018 seasons, respectively). The Random

forest model achieved the best results (R² = 0.98 and RMSE =17.8) between the predicted and actual fruit counts on the same set ABC-x that was used for training (Table 5-13).

2908Table 5-13. Prediction results for ABC-x 2017 season sample trees using different yield estimation methods,2909trained with the ABC-x 2017 training image set.

Model	Slope	Intercept	R ²	RMSE
				(# fruit per tree)
Deep_yield	0.94	10.25	0.92	30.4
MLP_yield	0.81	36.62	0.79	47.7
Random_forest	0.90	18.93	0.98	17.8
Xception_yield	0.71	28.8	0.94	44.8
Dual_view	0.89	36.39	0.69	65.4

2910

As expected, a better dual view method result was obtained using an occlusion factor estimated

2912 from the same season sample trees (Table 5-13). Surprisingly, this dual view result was superior to

that of the other methods (Table 5-14). The performance from the Random forest method was poor

in comparison to the dual view method but was better than other models (Table 5-14). The direct

2915 yield estimation models were therefore not robust in prediction of a new population.

2916Table 5-14. Prediction results for ABC 2018 season sample trees using different yield estimation methods,2917trained with ABC-x 2017 images.

Model	Train set	Occlusion	Test set	Slope	Intercept	R ²	RMSE
		factor					(# fruit per tree)
Deep_yield	ABC-x 2017		ABC 2018	0.29	61.19	0.34	129.1
MLP_yield	ABC-x 2017		ABC 2018	0.50	30.34	0.66	102.9
Random_forest	ABC-x 2017		ABC 2018	0.46	52.86	0.60	97.4
Xception_yield	ABC-x 2017		ABC 2018	0.42	29.00	0.72	106.3
Dual_view	-	ABC 2018	ABC 2018	0.83	44.46	0.73	69.0
Dual view	-	ABC-x 2017	ABC 2018	0.67	35.51	0.73	72.7
Dual view	-	ABC 2017	ABC 2018	0.63	33.54	0.73	77.6

2918

2919 5.3.6 Orchard block yield prediction

2920 For the combined whole orchard ABC block prediction, all direct prediction models achieved

estimates closer to the actual packhouse record than the count from the Dual_view model (Table 5-

15). Random forest model provided the best estimate for combined ABC orchards for season 2017,

with a 1.6% prediction error (Table 5-15). Note that the models were trained on images of samples

trees from the same orchard and season as used for block prediction.

Pac	khouse	Dual_\	/iew	Deep_	yield	MLP_	yield	Random	_forest	Xception	n_yield
C	ount	predicted	% err	predicted	% err						
A	97382	58074	2.57	91760	-5.77	93879	-3.60	99779	2.46	83638	-14.11
В	26273	16189	17.07	26307	0.13	32148	22.36	29307	11.54	26022	-0.96
С	40837	17329	6.09	39399	-3.52	41025	0.46	39188	-4.04	36624	-10.32
ABC	164492	91592	13.59	157466	-4.27	16705	1.56	168274	2.30	146284	-11.0
						2					

Table 5-15. Block prediction results for whole orchard A, B and C season 2017 from models trained on orchard ABC-x 2017 data. Best result (closest to packhouse) is shown in bold.

2928

The RMSE on individual tree estimates was 18 fruit per tree on an average fruit load of 200 fruit/tree, i.e., within 10%. The low prediction error of the orchard total suggests that the model has good accuracy but poor precision on individual tree count, with summation across many trees acting to improve precision.

However, some error exists on the packhouse counts, in terms of fruit left on tree or ground during commercial harvest. Also, the packhouse grader did not provide count on non-marketable fruits (small size, bruised, over-ripe, diseased etc.), and counts on this category of fruit were manually

2936 estimated from weight of the reject bin and an estimate of average fruit weight.

2937 **5.4 Conclusions**

2938 The use of machine learning models to be trained directly on fruit number per tree rather than fruit 2939 number per image would avoid the need for manual estimation of an occlusion factor every season. 2940 The supervised machine learning methods were able to improve the correlation between the 2941 number of fruits seen in the images to the harvest fruit counts, producing better result on training 2942 set data compared to the results from machine vison count corrected by occlusion factor. However, 2943 the same model when used to predict the total fruit count on a new season data (test set images) 2944 produced poor results. This indicates that the model created from just one season data was not 2945 enough for prediction on next season data for the trees considered in this study. This result can be 2946 attributed to the high variability in fruit number on individual tree between seasons. In general, the 2947 machine learning techniques try to model the underlying relations and patterns from the input data 2948 to predict on a new data. Thus, it is important to have more variation in the training data to train an 2949 accurate prediction model. Training of these models across several seasons and orchards is 2950 recommended. Moreover, the random forest regression model weighted on the fruit counts rather 2951 than the other image features based on the pixel count of fruit and canopy regions in images. It was 2952 also observed that the forest regression weighted more on the occluded fruit counts compared to

the counts of fully visible fruits.

2954 Acknowledgement

This work received funding support from the Australian Federal Department of Agriculture and
Water, and from Horticulture Innovation Australia (project ST19009, Multiscale monitoring tools for
managing Australian tree crops). AK acknowledges receipt of an Australian Regional Universities
Network scholarship, and ZW was funded by a CQU Early Career Fellowship. Farm support from Ian
Groves is appreciated.

2960

2961

2963 Chapter 6. In Field Fruit Sizing Using A Smart Phone 2964 Application

2965	This chapter was published as a journal paper on October 2018 in Sensors as:
2966 2967 2968	Wang Z, Koirala A , Walsh K, Anderson N, Verma B (2018) In Field Fruit Sizing Using A Smart Phone Application Sensors 18:3331 doi: <u>https://doi.org/10.3390/s18103331</u>
2969 2970	Responses to minor revisions as requested by the thesis examiners can be found in the Errata section.
2971	
2972	
2973	
2974	
2975	
2976	
2977	
2978	
2979	
2980	
2981	
2982	
2983	
2984	
2985	
2986	
2987	
2988	
2989	
2990	
2991	
2992	

2993 Abstract

2994 In field (on tree) fruit sizing has value in assessing crop health and for yield estimation. As the mobile 2995 phone is a sensor and communication rich device carried by almost all farm staff, an Android 2996 application ("FruitSize") was developed for measurement of fruit size in field using the phone 2997 camera, with a typical assessment rate of 240 fruit per hour achieved. The application was based on 2998 imaging of fruit against a backboard with a scale using a mobile phone, with operational limits set on 2999 camera to object plane angle and camera to object distance. Image processing and object 3000 segmentation techniques available in the OpenCV library were used to segment the fruit from 3001 background in images to obtain fruit sizes. Phone camera parameters were accessed to allow 3002 calculation of fruit size, with camera to fruit perimeter distance obtained from fruit allometric 3003 relationships between fruit thickness and width. Phone geolocation data was also accessed, allowing 3004 for mapping fruits of data. Under controlled lighting, RMSEs of 3.4, 3.8, 2.4, and 2.0 mm were 3005 achieved in estimation of avocado, mandarin, navel orange, and apple fruit diameter, respectively. 3006 For mango fruit, RMSEs of 5.3 and 3.7 mm were achieved on length and width, benchmarked to 3007 manual caliper measurements, under controlled lighting, and RMSEs of 5.5 and 4.6 mm were 3008 obtained in-field under ambient lighting.

3009 6.1 Introduction

In-field sizing of fruit on tree can be used to provide information on rate of fruit growth and thus
timing of harvest maturity, for estimation of pack-house packaging resource requirement and to
inform marketing decisions (Moreda et al. 2009). For example, (Zude et al. 2011) reported on use of
manual in-field fruit sizing for the estimation of cherry fruit harvest maturity. (Wang et al. 2017a)
reported in-field sizing of mango fruit to gauge packing tray size requirements for the crop.

3015 Current best practice for estimation of fruit size in orchard involves measurement by calipers, fruit 3016 sizing rings or circumference tapes. These measures require a certain level of operator attention, 3017 particularly when manual transcription of results is required. Relatively low cost (< USD \$1000) 3018 digital fruit sizers with data logger functionality are available (e.g., from Guss Manufacturing, Strand, 3019 South Africa), with transfer of data to a connected lightweight computer or tablet possible, e.g., as 3020 utilised in studies by References (Green et al. 1990; Morandi et al. 2007)). However, the available 3021 tools lack inbuilt geolocation and wireless communications capacity, and inherently rely on the 3022 operator purchasing and carrying specialist hardware.

3023 The use of machine vision for fruit sizing on packing lines began in the 1970's and is now well 3024 established (van Eck et al. 1998; Walsh 2018). In this application, a light box installed over the 3025 conveyor allows uniform lighting, a background of contrasting colour, fixed camera to fruit distance 3026 and angle, and use of multiple cameras. For example, Spreer and Müller (Spreer and Müller 2011) 3027 used two colour cameras with a fixed camera-to-fruit distance to assess mango fruit length, width, 3028 and thickness in a pack line scenario. Machine vision can also be applied to in-field estimation of 3029 fruit size, given knowledge of camera to object (fruit) distance. This can be achieved by inclusion of a 3030 scale bar in the object plane, e.g., as done by Reference (Cheng et al. 2017a). To avoid this 3031 requirement, Regunathan and Lee (Murali and Won Suk 2005) employed four ultrasonic depth 3032 sensors, however, the method produced an average distance to canopy rather than distance to 3033 individual fruit. In consequence the root mean square error (RMSE) on the size estimate of citrus 3034 fruit on tree was 19 mm (calculated from neural network estimate in Table 2 of (Murali and Won Suk 3035 2005)). The use of a RGB-D (depth) camera was suggested by Reference (Dellen and Rojas Jofre 3036 2013) for the measurement of object volume (i.e., boxes), while (Kongsro 2014) proposed use of the 3037 low cost Kinect RGB-D camera (Microsoft, Redmond, WA, USA) in the estimation of pig weight. In a

parallel work, our group has used a Microsoft Kinect V2 RGB-D camera mounted to a farm vehicle to
measure mango fruit length and width of non-occluded fruit on tree, with RMSE of 4.9 and 4.3 mm
achieved, respectively (Wang et al. 2017a). The disadvantage of this system is the requirement for
relatively bulky, specialist equipment.

3042 The 'smartphone' is now a ubiquitous handheld communication and computing device, effectively 3043 carried at all times by all farm staff. (Seifert et al. 2015) used the data storage and communication 3044 capability of the smartphone for management of manual measurements of cherry fruit dimensions. 3045 However, these devices possess cameras, geolocation capability, capacity for real time image 3046 processing and the ability to transfer data to shared data structures. Therefore, a smartphone 3047 application is proposed for the assessment of fruit size, capitalising on existing phone features to 3048 achieve a low-cost solution in a product carried by the user at all times, and thus accessible for use 3049 at any time. Future developments can include image analysis to provide a defect assessment in 3050 parallel to sizing estimates. The development of such an application is relatively straightforward, 3051 with novelty in being the first to achieve this practical solution to in-field fruit measurement, and in 3052 addressing the technical challenge of accommodating variation in fruit thickness (i.e., camera to fruit 3053 perimeter distance).

3054 6.2 Materials and Methods

A description of operation of the application is given, followed by a description of the experimentalcharacterisation exercises.

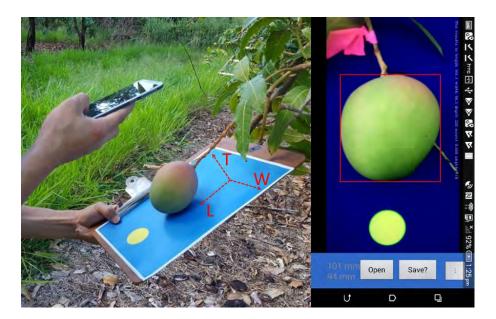
3057 6.2.1 The 'FruitSize' Application

3058 6.2.1.1 Description of Overall Operation

3059 An application ('FruitSize') was written in Java and C/C++ language using the OpenCV library for 3060 image processing, for use on an Android phone (Android 4.4 or above platforms). The application 3061 can read parameters such as camera maximum resolution, camera focal length, camera horizontal, 3062 and vertical view angles for different phones, and output length and width of an imaged object 3063 (fruit). A HTC Desire 820 mobile phone was used for the reported trial work. In this phone the in-3064 built primary camera has a focal length of 3.81 mm, a calculated pixel-bin size of 2.377 μm and 4224 3065 × 3136 pixels (13 Megapixels), with a resolution of 1920 × 1080 pixels utilised as this resolution is 3066 supported by most smartphones. A Samsung Galaxy S6 with 16 Megapixel rear camera (Android 7.0) 3067 was used to illustrate the use of the application on other mobile phones.

3068Operation consisted of holding a board with a blue coloured A4 paper sheet behind fruit on tree3069(Figure 6-1a). A yellow circle on the backing paper was used as a reference for the estimation of3070camera to reference plane distance. The blue background was chosen to facilitate segmentation of3071green, red, and yellow fruit (e.g., Figure 6-2). The camera to fruit distance was maintained greater3072than 120 mm (to ensure image coverage of the whole fruit and reference circle) and less than 3003073mm (as an operator convenience, with backboard held in one hand and camera in other hand).

Fruit sizing app



3074

Figure 6-1. Application scenario (left) and main user interface (right), illustrating fruit real dimensions of length (L), width (W), and thickness (T).



3077

Figure 6-2. Image processing: image acquisition (left); circle identification (middle); and fruit
 segmentation (right).

3080 6.2.1.2 Detection of Fruit

3081 Two regions in the image are of interest: The reference circle and the fruit. For in-field acquired 3082 images, object segmentation is a challenging task because of high variation in lighting condition, 3083 including strong shadows on fruits, and variance in fruit traits such as size, shape, colour, and 3084 texture. As cameras in different mobile phones differ in colour rendering, a device independent 3085 segmentation method was sought. Camera RGB output was converted to XYZ space and then to CIE 3086 L*a*b* space using OpenCV functions. A blue background provided contrast to fruit which are 3087 typically green, red, or yellow, using the b* channel of CIE L*a*b* colour space. The Otsu's 3088 automatic thresholding (Otsu 1975) was applied on the b* channel of the image to segment fruit and 3089 reference object from the background. Otsu's method seeks a global optimum threshold such that

- the method can adapt to different light conditions. Thus, thresholding was accurate even with strongdirect s4unshine and partial shadowing of the fruit (Figure 6-3).
- 3092 Morphological operations were then conducted to remove small objects (peduncle, twigs, leaves,
- 3093 etc.), and a stalk (fruit peduncle) filter based on the priori knowledge that fruit are much larger than
- the stalks (Wang et al. 2017a), was used to remove the stalks which connect with the fruit. Finally, a
- dilation operation with a disk structure of one pixel radius was used to generate a smooth curve
- shape to outline the imaged fruit (examples in Figures 6-2 and 6-3). Images were rejected in which
- the fruit and reference circle made contact.

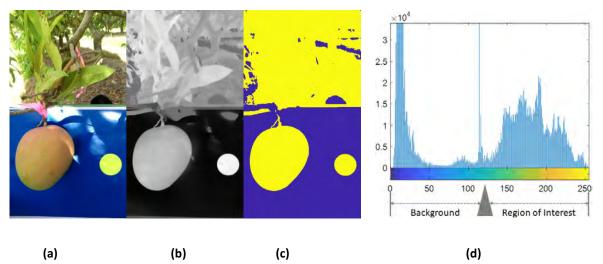


Figure 6-3. Panels from left to right display (a) RGB image of fruit against a blue board and canopy
 background, in conditions of partial direct sunlight; (b) gray scale in the b* channel for the same image, (c)
 the segmented image; and (d) a histogram of b* channel values from the image, with the Otsu's method
 optimum threshold for separation of background from Region of Interest pixels indicated by the grey arrow.

Finally, object eccentricity and area were used to define the reference circle and the fruit. The

3103 findContours function in the OpenCV library was used to find the shapes of all isolated objects and

the fitEllipse function was used to judge if the object image shape was close to a circle, using the

- 3105 criterion of eccentricity (ε) calculated from the length (ellipse major axis, a) and width (minor axis, b)
- 3106 of a bounding box (Equation (1)).

$$\varepsilon = \sqrt{1 - \left(\frac{b}{a}\right)^2} \tag{1}$$

This parameter was estimated for both the reference circle and the fruit. With the phone camera held parallel to the clipboard, the reference circle is imaged as a perfect circle with $\varepsilon = 0$.

- 3109 The eccentricity and area size (in pixels) of the fruit image were used to judge the type of fruit, as
- 3110 introduced in Reference (Wang et al. 2017a). Briefly, mango and avocado fruit approximate an
- 3111 ellipse shape while apple and citrus fruit approximate a circle shape. This categorisation was used for
- 3112 selection of the appropriate fruit allometric relationship between width and thickness.

3113 6.2.1.3 Camera to Reference Plane Distance Estimation

Pixel size (*p*) (in mm) can be estimated from the camera focal length, *f* (in mm), the total number of horizontal (or vertical) pixels, W_m , and the camera horizontal (or vertical) angle of field of view, θ (in degrees) (Equation (2)).

$$p = \frac{2f}{W_m} \tan \frac{\theta}{2} \tag{2}$$

Usually each cell in a CMOS image sensor is square, so solving p in one dimension is sufficient. The
reference circle diameter (in pixels, denoted by ø) was estimated of the segmented image, and its
actual size was calculated as product of ø and p. Using the thin lens formula, the distance u (in mm)

3120 from the reference plane to the camera was estimated from the known diameter (*D*) of the

3121 reference circle (Equation (3)):

$$u = \frac{fD}{\phi p} \tag{3}$$

3122 6.2.1.4 Estimating Fruit Dimensions

Fruit length, width and thickness were defined in terms of real dimensions (L_r , W_r , T_r , in units of mm) and image dimensions (L_i , W_i , in units of pixels). A bounding box was drawn around an object recognised as a fruit. Box vertical and horizontal dimensions represent fruit image length L_i and width W_i , respectively, in units of pixels. The thin lens formula (Equations (4) and (5)) allows calculation of the fruit real length L_r and width W_r (in mm):

$$L_r = \frac{u}{f} p L_i \tag{4}$$

$$W_r = \frac{u}{f} p W_i \tag{5}$$

3128 where \boldsymbol{u} is the distance from the backboard to the camera (in mm).

However, the fruit is a three-dimensional object, with the imaged perimeter of the fruit occurring

some distance above the reference plane. The reference plane was projected to the fruit perimeter

plane using an estimate of the fruit half-thickness. Fruit thickness, T_r (in mm), was iteratively

3132 estimated using an experimentally determined relationship with fruit width (see later section), as

3133 $T_r = cW_r$ (where c is constant) until W_r converged. With each iteration, a revised estimate of T_r 3134 was used to calculate new values for L_r and W_r by replacing u with $\left(u - \frac{T_r}{2}\right)$. Empirically, this

3135 process converged to a stable value of W_r within five iterations.

3136 6.2.1.5 Operating Features

3137 The user requires a simple and interactive interface, with the application reacting to potential errors,

e.g., the acquired image is rejected if the camera to object plane angle or the camera to object

distance is out of specification. An accepted image is processed and displayed with a green circle

overlaying the reference spot and a red bounding box overlaying the target fruit. Time stamp and

- 3141 geolocation data from the smartphone is saved along with fruit size into a CSV file in the phone
- 3142 'Download' folder, with user input required to transfer the file to local computer or cloud server.

3143 6.2.2 Experimental Exercises

3144 6.2.2.1 Reference Circle Size

A larger reference circle allows for a more accurate assessment of diameter, given limitations of camera resolution. The impact on fruit sizing of size of reference circles was empirically evaluated by mounting the phone on an optical stand with distance to the background varied between 120 and 300 mm in steps of 20 mm as assessed by manual tape measure (accuracy of approximately 1 mm), covering the range of anticipated working distances. Camera to reference plane distance *u* was calculated from Equation (3) for images acquired of reference circles of a range of diameters (10–70 mm) at each distance.

3152

3153 6.2.2.2 Camera Tilt

The camera plane should be parallel to the object plane for the most accurate measurement of
object size. A circle object will appear as an ellipse in the image if the camera is tilted. To evaluate
this source of error on distance estimation, the mobile phone was held on an adjustable optics

- 3157 mount with a fixed distance of 200 mm from camera lens to the backboard, tilted up (+) or down (-)
- 3158 relative to the reference plane.

3159 6.2.2.3 Fruit Allometrics and Sizing

As a reference measure, fruit length and width were assessed using a digital caliper (DCLR-1205,

3161 Clockwise Tools Inc., Palo Alto, CA, USA). Measurement repeatability was assessed as standard

deviation (SD) of 20 repeated measurements of a mango fruit. To establish fruit allometrics, the

3163 length, width, thickness, and weight of 387 mango fruits of four cultivars (Kensington Pride (KP),

Calypso, Honey gold and Keitt) were measured. Fruit were of a range of maturities.

3165 To test the performance of the 'FruitSize' application, four exercises were undertaken. Exercise 1: 3166 Images were acquired of 20 fruit each of apple (Royal Gala), avocado (Hass), mandarin (Imperial), orange (Navel), and mango (Honey Gold) in a laboratory setting. Exercise 2: Images were acquired of 3167 3168 176 mango (cultivars Kensington Pride and Honey Gold) fruit on tree. Acquisition occurred under a 3169 range of lighting conditions, although strong shadows were avoided. Exercise 3: Twenty-one mixed 3170 fruit (mandarin, apple and orange) were measured by calipers, and by two operators, using separate 3171 phones (Samsung and HTC). Exercise 4: Twenty mango (Kensington Pride) fruit were tagged on tree 3172 at stone hardening stage, and assessed weekly over two months, until harvest, using both calipers 3173 and the FruitSize application.

3174 6.3 Results and Discussion

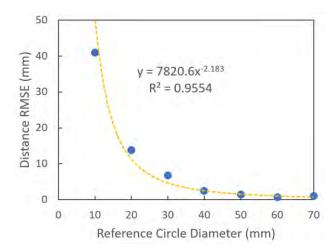
3175 6.3.1 Effect of Reference Circle Size on Distance Estimation

Increased reference circle size should result in an improved estimation of the distance from camera
to image plane, as the relative uncertainty in the estimation of the diameter of the reference circle is
decreased. For example, there is a blur between yellow and blue areas of approximately four pixels
at a camera to image plane distance of 200 mm. At this distance, a line of 10 mm is imaged by 80
pixels, and thus an uncertainty of 4 pixels represents 5% of the measurement.

3181 The measured RMSE of the camera to reference plane distance estimate decreased as circle size 3182 increased, with improvement decreasing for larger diameters (Figure 6-4). A reference circle size of

Fruit sizing app

- 40 mm, associated with RMSE of 2.4 mm (Figure 6-4), was adopted in further work in a compromise
- between precision and the available space on an A4 sized background. An RMSE of 2.4 mm in
- distance estimation will introduce a 1.2% error in estimation of fruit size, given a camera to image
- 3186 plane distance of 200 mm (calculation using Equation (3)).



3187

Figure 6-4. RMSE of camera to object distance estimation (for 10 replicate measurements) as influenced by reference circle size.

3190 6.3.2 Effect of Camera Tilt Angle

- 3191 As the angle of the camera to reference plane was increased by tilting the camera long axis, the
- error of the camera to reference plane distance estimation increased non-linearly (Figure 6-5)
- 3193 because of distortion of the imaged reference circle. A limit of 14°, associated with an eccentricity of
- 0.3, was set on the camera-object plane angle for further processing of the image. The RMSE of 12
- 3195 mm, associated with a 14° tilt, will introduce 6% error on estimation of fruit size for a fruit-to-camera
- distance of 200 mm (from Equation (3)).

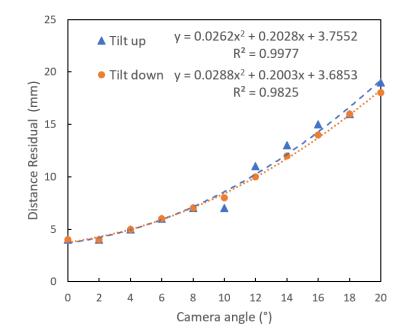


Figure 6-5. Absolute residual of camera to object distance estimation as influenced by angle between camera (phone major axis) and object planes, for tilt forward (down) and back (up). Camera lens held at a set distance to object plane, while phone body was tilted.

3201 In field practice, there was greater variation in user positioning of the major axis of the phone

relative to the backboard than for the minor axis of the phone. Use of the reference circle minor axis

- 3203 (axis parallel to the minor axis of the phone) to estimate camera to reference plane distance
- therefore achieved better accuracy than use of major axis (data not shown), and therefore this axis was used in scaling the image.

3206 6.3.2.1 Fruit Allometrics

Fruit can be characterized in terms of their lineal dimensions of length, width, and thickness. For some commodities, width and thickness may be similar, and all three parameters are similar for a

3209 spherical fruit. An allometric relation between the lineal dimensions of fruit length, width, thickness,

3210 and the weight of Chok Annan variety mango fruit was established by Reference (Spreer and Müller

3211 2011), and for a range of other cultivars by (Anderson et al. 2017). Similar allometric relationships

3212 between fruit lineal dimensions and weight hold for other fruit. Allometric relations can also be

3213 established between lineal parameters, e.g., between length and width.

3214 In holding the backing board behind fruit on a tree, the long axis of the fruit (*L*_{*r*}) naturally aligns with

3215 long axis of the backing board. For mango fruit, the wider minor axis (fruit 'width', W_r) aligns with

3216 the board minor axis, with the dimension of the fruit in the camera to reference plane direction

3217 representing fruit 'thickness'. An allometric relationship established between mango fruit thickness

3218 (T_r) and width was stronger than that for thickness and length (Table 6-1). The relationship $T_r = 0.88$

- 3219 $\times W_r$ was adopted in the operation of the application. The same relationship was applied to avocado
- 3220 fruit. Mandarin, apple, and citrus were assumed to be spheres, i.e., $T_r = W_r$.

Table 6-1. Allometric relationships based on linear regression with intercept of zero for fruit real thickness (Tr) to length (Lr) and width (Wr) of mango fruit (n given in brackets).

Cultivar	T_r vs. W_r Re	lationship	T _r vs. L _r Rel	ationship
Cultival	Slope	R ²	Slope	R ²
Honey Gold (42)	0.82	0.72	0.74	0.50
KP (43)	0.87	0.66	0.73	0.47
Calypso (242)	0.90	0.88	0.75	0.65
Keitt (60)	0.82	0.93	0.57	0.92
All	0.88	0.67	0.74	0.60

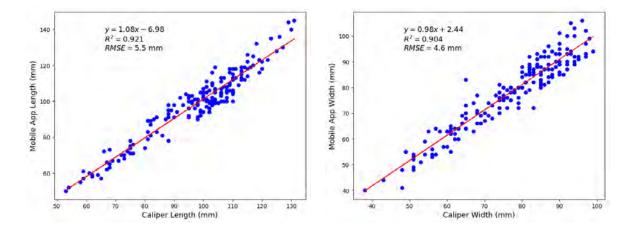
The RMSE values on length and width estimation of 20 mango fruit were decreased from 8.8 to 5.3 mm and 6.8 to 3.7 mm by the depth correction step. Obviously, the procedure does not give a

perfect estimate of camera to fruit perimeter distance, and so represents a source of error in fruitsizing estimation.

3227 6.3.3 Size Estimation Results

Repeatability of the reference (caliper) method was assessed at 1.2 mm for mango fruit (SD of 20 measurements).

- 3230 Exercise 1: For avocado fruit, RMSE on fruit length and width was 3.4 and 1.6 mm, respectively,
- while for apple, mandarin and orange diameter, RMSE was 2.0, 3.8 and 2.4 mm, respectively. The 3231 3232 RMSE on mango fruit length and width measurements were 5.3 and 3.7 mm, respectively.
- 3233 Exercise 2: For in orchard measurements of mango fruit on tree (n = 176), the linear correlation of
- 3234 machine vision estimated fruit length and width against caliper measurements was characterised by
- 3235 a R² = 0.921 and 0.904 and RMSE of 5.5 mm and 4.6 mm (bias = +1.0 and +1.1) for length and width
- 3236 estimation, respectively (Figure 6-6). This RMSE result is equivalent to that achieved in a companion
- 3237 study on in field estimation of mango fruit size using a Kinect time of flight camera (Wang et al.
- 3238 2017a). The lower RMSE of the assessments made indoors compared to the field based mango
- 3239 measurements is likely due to better operator performance in terms of holding the phone parallel to
- 3240 the backing board and less segmentation error associated with uniform lighting in an indoor setting.

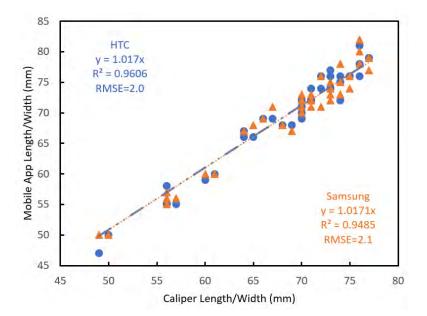


3241 Figure 6-6. Correlation between mobile application and caliper measurement of fruit length (left panel) and 3242 width (right panel).

3243 The bias is attributed to an error either in the lens and pixel specifications accessed from the camera 3244 software, or an over-estimate of fruit depth from fruit allometry. However, bias can be empirically 3245 corrected in future measurements.

- 3246 Exercise 3: To illustrate use of the application across mobile phones, two mobile phones were bench 3247 marked to caliper measurements, with similar results achieved (RMSE = 2.0 and 2.1 mm for the HTC 3248 and Samsung phones, respectively) (Figure 6-7). The slight difference is attributed to operation error 3249 (i.e., different camera tilt angles). The performance consistency of the application on different 3250 mobile phones is ascribed to the measurement principle, being based on thin lens theory. A 3251 manufacturer/model calibration or bias-correction procedure could increase accuracy, but this
- 3252 approach adds additional complexity (e.g., a list of supported phones).

Fruit sizing app

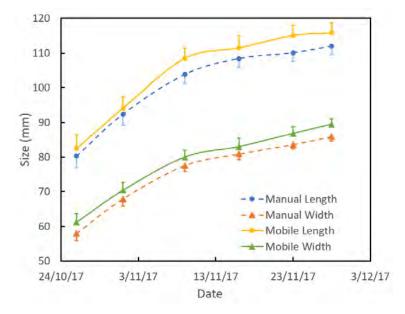


3253

3254 Figure 6-7. Fruit length and width estimated using the mobile application on two phone types (HTC, 3255 Samsung), relative to caliper measurement, for a set of mandarin, orange and apple fruit (n = 21 fruit).

3256 Exercise 4: To illustrate use of the application, machine vision and caliper estimates of 20 tagged 3257 fruits on tree were acquired weekly over two months (three fruit dropped before the last reading) 3258 (Figure 6-8). The average growth rate over this period was 1.0 mm/day for length and 0.9 mm/day 3259 for width, as estimated using either caliper or machine vision. Changes in the rate of growth can be 3260 interpreted in terms of growth conditions, e.g., water availability and fruit maturity (Anderson et al. 3261 2017). Size estimates can be converted to fruit weight estimates based on fruit allometry, as

3262 undertaken in the companion study (Wang et al. 2017a).



3264 Figure 6-8. Time course of average mango fruit lineal dimensions (n = 17 fruit) for length (top line pair) and 3265 width (bottom line pair), as assessed using calipers (dashed line) or machine vision (FruitSize application; 3266 solid line). Error bars represent the standard error of the mean.

3267 6.3.4 Fruit Size Application Use

A typical assessment rate of four fruit per minute or 240 fruit per hour was achieved. The FruitSize application can be used for estimation of population statistics on fruit size, to inform packing and marketing decision making. The required sample for such a task is dependent of the level of population variation. For the population of assessed fruit reported in Figure 6-5, the standard

deviation (SD) on the machine vision length and width estimates was 18.7 mm and 14.2 mm,

respectively. Using the t-statistic relationship $n = \left(\frac{t \times SD}{e}\right)^2$ and t = 1.96 (confidence level of 95%, n > 15) and acceptable error of 5 mm, a sample size of 54 (for length) and 31 (for width) samples is

3275 warranted. This is a manageable workload, per orchard block.

3276 6.4 Conclusions

3277 In a parallel study we describe in-field fruit sizing based on use of an RGB-D camera mounted to a 3278 farm vehicle. The current paper introduces a low-cost solution, based on use of a smartphone. The 3279 application utilises the thin lens formulae and a reference marker to estimate camera to background 3280 distance, adjusted for camera to mid-fruit plane by an iterative correction from the relationship 3281 between fruit length and thickness. The application employs several internal checks of image quality 3282 features that impact on size estimation, rejecting images taken outside of the camera to fruit 3283 distance of 120 mm to 300 mm, and at a tilt angle between camera plane and object plane of 3284 greater than 14 degrees. The eccentricity and area of the imaged fruit was used to allocate the 3285 image to a fruit type and associated allometric relationship between fruit width and thickness. 3286 Measurement accuracy allowed estimation of fruit growth rate from weekly assessments, and 3287 allowed estimation of fruit size distribution. Phone geolocation data allows for field collected data to 3288 be automatically assigned to user defined orchard management units, with associated calculations 3289 to render weight distribution or projected time to reach a desired size.

Possible improvements include a lightweight frame to hold the camera at a fixed distance and angle relative to the object, with compromise to ease of storage and transport. Use of more than one reference circle (e.g., in the four corners of the backboard) would also be useful in calculating a correction for camera tilt and distance. New phone technology employing a true-depth camera (e.g., iPhone X) or stereo cameras (HTC Evo 3D, iPhone 7 Plus) could be used to obtain the fruit perimeter to camera distance, replacing the need for a reference object. Alternatively, a third-party plug-in ToF-distance measurement device (e.g., Ryobi ES9400) could be used.

3297 The application is available for download at https://www.fruitmaps.com.au.

Funding: Funding support (grant ST15005) from Hort Innovation Australia and the Australian Government Department of Agriculture and Water Resources as part of its Rural R&D Profit program is acknowledged. Z.W. and A.K. acknowledge support of a CQU Early Career Research Fellowship and a RUN scholarship, respectively.

- Acknowledgments: We thank growers of the Central Queensland region, particularly Groves Grownproduce, for provision of fruit.
- 3304
- 3305
- 3306
- 3307

3308 Chapter 7. Conclusion

3309 7.1 Summary

Tree fruit crop yield estimation is a valuable input to orchard management activity and into decision support tools for planning marketing, transport and harvest labour needs. Advances in machine vision hardware, image analysis and data management (cloud computing) are allowing new opportunities in a wide range of applications. In this thesis, machine vision has been applied to infield assessment of the extent of flowering, fruit size and fruit number. An accurate automated fruit recognition and counting is also a precursor to robotic fruit harvesting.

- A machine vision fruit detection system was designed, evaluated and implemented for real-time
 mango panicle and fruit detection and counting in the orchard. The system is based on night time
- dual view imaging, and is comprised of an RGB camera, LED floodlight, GNSS receiver and a field
- 3319 computer mounted on a farm vehicle. Deep learning model training and testing was implemented on
- the CQUniversity High Performance Computing (HPC) platform running Red Hat Linux operating
- 3321 system and NVIDIA[®] Tesla[®] P100 (16 GB memory) GPU. For in-field use, a Nuvo-6108GC industrial-
- 3322 grade computer running Windows operating system and NVIDIA GeForce GTX 1070 Ti (8GB memory)
- 3323 GPU was used. Deep learning algorithms on both hardware were compiled with CUDA and cuDNN
- major versions 9 and 7, respectively. Similarly, OpenCV and Python major versions 3 and 2 were
- installed on both computing resources, respectively. As a general recommendation, a computingplatform with GPU memory of 8 GB or more is recommended for deep learning model training and
- 3327 testing. For larger CNNs and higher network input resolution, the memory requirement can scale to
- 3328 more than 16 GB (Koirala et al. 2019b). Therefore, for practical in-field implementation, deep
- 3329 learning architectures must be optimized for lower computational resources.
- The thesis addressed both flowering and fruit load estimation. There have been several publications on automated fruit detection but relatively few publications address in-field tree crop flower
- assessment. The flower peak detection and display options presented in the current study should
- 3333 prompt further work in this field. Another area poorly addressed in the literature is the issue of fruit
- 3334 occlusion. The structure of canopy, camera viewpoint and other occlusions pose a challenge for
- machine vision systems in terms of visualization of all fruits on tree and thus for accurate yield
- 3336 estimation. Attempts to accommodate the occlusion factor inside a trained model as presented in
- the current study should also provide guidance for future work in this field.
- 3338 The following objectives, as set at the start of the thesis, have been addressed:
- 3339 Objective 1: Define applications for machine vision in mango culture for precise estimation of crop3340 yield
- As initially conceived, this thesis addressed a need for mango fruit load estimation in orchard, with estimation made at least four weeks before harvest. However, additional applications for machine vision were defined and addressed, including the in-field count of panicle number to assist in characterisation of the time and spread of harvest and the size of fruit, to complement the estimate of fruit number. Fruit detection and estimation of fruit size, which is based on an estimate of camera to fruit distance, is also fundamental to the development of automated harvesting.
- 3347 Objective 2: Contribute to the design of a low-cost imaging system for use on mango orchards
- 3348This thesis work occurred in context of a team activity (Sensors Group, Institute for Future Farming3349Systems, CQU). In the design of a low-cost imaging systems for practical implementation on mango

- orchards my primary contribution, as recorded in this thesis, was the development of a deep
- 3351 learning based detection framework. An existing object detection framework was re-designed for
- 3352 reduced computational cost (Chapter 3). This new detection framework outperformed other
- 3353 frameworks in terms of operational speed, memory requirement and detection accuracy, enabling it
- to be used for real-time in-field operation. This framework was extended to detection and count of
- several developmental stages of mango panicles for flowering assessment (Chapter 4). Several
 machine learning methods and their combinations were also explored to predict the total crop yield
- 3357 from the fruit counts and tree images obtained from using machine vision system (Chapter 5).
- Additionally, a smartphone app based on machine vision algorithms was developed for fruit sizing and estimation of fruit weight (Chapter 6). The app can be downloaded to any android phone (from <u>www.fruitmaps.com.au</u>) and operated without the need for extra imaging hardware and computing resource, makes this a cost-effective application.
- 3362 *Objective 3: Establish a database of images with different mango varieties and associated actual* 3363 *flowering/fruit count collected at different times of season and from different orchards. These image*
- 3364 sets will be useful for anyone who wish to train and test their models
- 3365 Deep learning methods benefit from the use of large training datasets (big data) and the availability 3366 of such datasets for mango flower and fruit is rare. Tree flower and fruit images were collected on 3367 several commercial orchards located in different geographical regions within Australia, with images 3368 of different cultivars acquired every season for three years. Ground truth manual counting of fruit 3369 per tree was collected for calibration trees. Different camera hardware was also used in imaging in 3370 some locations. This activity created a large dataset of images and ground truth fruit counts. The 3371 labour-intensive task of ground truth data annotations and labelling of images for training machine 3372 learning algorithms was undertaken on approximately 2600 images, across fruit and flowering tasks. 3373 This annotated training data (1730 annotated images for fruit detection) was made publicly available 3374 for benchmarking other machine vision systems at http://hdl.cqu.edu.au/10018/1261224.
- 3375 Objective 4: Compare and benchmark the state-of-the-art machine learning methods for estimation3376 mango flower and fruit detection and yield estimation
- State-of-art machine learning methods were reviewed and overviewed for the use in fruit detection
 and yield estimation (Chapter 2). An existing object detection framework was re-designed to create
 a new model, and performance benchmarked against several state-of-the art machine learning
 methods, for the task of fruit detection and yield estimation (Chapter 3). Similarly, the new model
 was extended for flower assessment and benchmarked against the machine vision systems of
 previously published works (Chapter 4).
- The better generalization capabilities of deep learning methods across cultivars, lighting conditions
 and imaging hardware, were demonstrated in this current study. Moreover, deep learning
 algorithms proved robust against varying level of occlusions, which is an unavoidable case for in-field
 applications.
- 3387 Objective 5: Contribute to the development of a decision support tool using output of the machine3388 vision analysis
- 3389 One of the challenges in precision agriculture is the display and interpretation of large data sets in a
- form useful to the farm manager, i.e., an appropriate decision support tool for farm management.
- All the machine vision systems, established form this study, for application in flower assessment,
- 3392 fruit detection, fruit sizing and yield estimation, were integrated into a common resource the
- 3393 FruitMaps website (<u>www.fruitmaps.com.au</u>), which was developed by the greater CQU Sensors

team. The machine vision system is capable of uploading the data (images, fruit counts, panicle stage
counts, fruit size etc.) into the web server for the display in an easy to interpret form (graphics and
plots). Users can view the data at an individual tree level on the satellite maps of their orchard and
use the information to manage farm and for market planning and farm resource management
(Chapter 4).

3399 7.2 Future directions

3400 7.2.1 General trends

The general public is aware of rapid progress in array of technologies, with the media carrying coverage of 'buzzword' topics such as machine vision, AI and IoT. Key drivers for adoption of automated sensors and technologies on farm include the shortage of farm labour, the falling price and increasing availability of computing and communication resources. In particular, orchard applications of machine vision are enabled by the rapid implementation of machine vision technologies in a range of other applications, which underpins lowered cost and improved reliability of the technology.

3408 7.2.2 Technology compromises

The adoption of a technology relies on a synergy of technical, social and political factors. The best technical solution is not necessarily the solution that will be adopted, e.g., in the videotape format war, Betamax was said to have been better technical solution, but VHS won the war. An obvious compromise exists in the balance of cost, complexity and performance of a technology.

3413 The suite of technologies that will be commercially adopted for orchard imaging remains to be seen. 3414 For the orchard imaging task, night time imaging was preferred in this study over daytime imaging. 3415 The night imaging solution provided reduced background noise and better image quality, with use of 3416 lower cost and less specialised equipment (e.g., LED lighting system instead of high cost Xenon 3417 strobe lights which have been used in daytime imaging, with attendant need for short exposure 3418 cameras). However, 'higher end' solutions can provide amazing visualisations of orchard scenes. For 3419 example, Stein et al. (2016) utilised RGB and LiDAR with IMU and GNSS tracking to create a 3D 3420 representation of a mango orchard, locating every fruit in XYZ space. However, as noted earlier,

- 3421 commercial adoption will rest on a balance of cost, complexity and performance.
- 3422 The issue of lighting deserves further attention. A Flash-No-Flash (FNF) technique has been 3423 proposed to mitigate the effect of strong background lights for a picking robot operating in 3424 structured glasshouse environment (Arad et al. 2019). The issue of hidden fruits especially for large 3425 and dense tree canopies also deserves further attention. The issue is partly addressed using multi 3426 view imaging (e.g., sequence of images or videos) which exposes more fruit than use of a single or 3427 dual view. Recent advances in the field of complex tracking, e.g., DeepSORT (Wojke et al. 2017) 3428 holds promise for this implementation. However, multi view imaging requires tracking of multiple 3429 fruit across consecutive frames to prevent double counts, and the added computational complexity
- 3430 will add to the computing hardware requirements of this application.

The use of machine vision and associated automation of farm tasks is lagging for orchard applications because of the constraints of the unstructured farm environment. Developments in several fields are often required for the adoption of a technology. For example, the uptake of the automobile was concurrent with developments in road engineering. The uptake of machine vision into orchard tasks is complemented by a trend to high-density small tree plantings in many tree crops, including mango. In the case of apple trees, trellised production is now common. The narrow 3437 canopies have promise for better light interception, for machine vision and for penetration by3438 harvest arms.

3439 7.2.3 From cloud to in-field computing

3440 The expansion of cloud computing resources provides opportunities for implementation of orchard 3441 machine vision applications. The Amazon AWS and Microsoft Azure provide pre-configured 3442 environments in the cloud supporting popular deep-learning tools such as Caffe, Keras, MXnet, 3443 PyTorch, and Tensorflow for deploying machine learning and deep learning AI applications. 3444 However, uploading large number of images to the cloud server can be time consuming and in many 3445 cases the orchards are in relatively remote areas with no or very poor internet speed. As an 3446 example, a commercial orchard (approximately 55k mango trees) in the Northern Territory of 3447 Australia is using a satellite link to upload farm images because of lack of landline or 3-G mobile 3448 network. Approximately 7000, 5 MP images of 3,500 trees requires about 8 hours of upload time.

- Having a local computing resource can avoid the need to store or transfer images and provides
- 3450 reduced system complexity and operation time. Industrial-grade ruggedized GPU computers suitable
- for in-field operation are becoming available, servicing a market in autonomous vehicles. For
- 3452 example, the Nuvo from Neousys Technology (<u>www.neousys-tech.com</u>) provides GPU 'grunt' for the
- 3453 running of state-of-art deep learning models. These computers have been used in autonomous farm
- 3454 robots (<u>https://www.swarmfarm.com/</u>) and CQUniversity's mango harvester.
- 3455 Reduced computational time allows for cheaper hardware. There exist some light weight deep 3456 neural networks (e.g., MobileNet) for mobile and embedded vision applications. Smart phones are 3457 equipped with various sensors including RGB and ToF cameras and can run machine vision and light 3458 weight deep learning algorithms. Many mobile apps have already integrated machine learning 3459 technology to provide better and accurate services through learning from huge pool of data. The 3460 application of mango fruit sizing from this research work can be extended to disease/defect 3461 detection on fruits in future. Such embedded machine vision systems can find several applications in 3462 an orchard, such as for insect and weed classification. For example, Leafsnap (Kumar et al. 2012) 3463 mobile app identifies tree species from the image of tree leaves (leafsnap.com). The Croptracker 3464 'Harvest Quality Vision'app allows imaging of apple fruit in a harvest bin, with analysis of quality 3465 attributes before the fruit reaches the packhouse (<u>https://www.croptracker.com/product/harvest-</u>
- 3466 <u>quality-vision.html</u>, doa 5/11/2019).
- FLIR Systems (https://www.flir.com.au/) recently launched the Firefly-DL machine vision camera that supports deep learning inference hardware onboard the camera. This current generation of hardware can support small CNN models such as LeNet, SqueezeNet and SSD MobileNet_v1, and cannot handle the more accurate and popular CNN architectures such as VGG, ResNet, Inception and YOLO are not yet supported. No doubt hardware advances will come, enabling 'edge computing' applications in the field of precision agriculture and in pack-line grading.

3473 7.2.4 Advancement in AI

Deep learning CNN architectures have shown state-of-art performance on object detection and
classification tasks. There are several other deep learning techniques beyond classification task that
can be exploited for the use in precision agriculture. There has been a huge progress in generative
AI, for example, Google's BigGAN is able to generate high resolution synthetic images that are
precisely like real images. Moreover, deep learning techniques has been used for tracking objects
with higher accuracies in the video frames. Current application of deep learning methods for fruit
detection and yield estimation such as, training deep models on synthetic fruit datasets (saving the

- 3481 cost of data acquisition) and training deep regression models for direct prediction of fruit numbers
- on input tree images (saving the time and cost for object annotation) are promising. Long Short-
- 3483 Term Memory (LSTM) methods of deep learning have been widely used for prediction on time-series
- 3484 data for example, weather forecast. Such technologies along with IoT have a big promise for a3485 sustainable precision farming.

3486 7.2.5 IoT

3487 The application purpose of machine vision estimation of flowering is to support estimation of 3488 harvest timing, but it is only one half of the solution. The other requirement is for temperature 3489 monitoring, ideally of individual blocks of trees, to inform heat sum models of fruit maturation. This 3490 can be done with manually logged thermistors, but the logging task is tedious. Given that 3G 3491 connection is often poor on mango farms, and connections are relatively expensive, other solutions 3492 are needed to deploy sensors for crop and orchard environment monitoring. As an example, 3493 LoRaWAN (Long Range Wide Area Network) solutions from Advantech (www.advantech.com) 3494 provides IoT sensor nodes of low cost, long range (~18 km in rural areas) and low energy 3495 consumption (https://doi.org/10.3390/s18030772) . A wireless gateway can support hundreds of

3496 sensor nodes to efficiently collect and upload sensor data to the cloud for forecasting.

3497 7.2.6 Autonomy on farms

Autonomous vehicles could be used to monitor the orchards for tree crop health and yield. Among others, SwarmFarm Robotics (<u>https://www.swarmfarm.com/</u>) deploys small semi-autonomous machines for spraying and weeding tasks on broad acre crops. Such platforms are the basis for autonomous farm machine vision systems. The use of auto-steer tractors is common in the grains industry and nascent in tree crops use, with a level of human intervention, e.g., for turning at row ends. Thus, orchards need to be designed with suitable navigation of field automated systems in mind.

3505 In the field work of this thesis, GNSS signal loss was experienced in some parts of orchard blocks,

- usually the end rows close to wind-breaking tall trees. The use of GNSS system for navigation of
- 3507 autonomous farm vehicles is thus unreliable for existing orchards. Self-driving cars can
- autonomously navigate using LIDAR or RGB data and machine learning algorithms. LIDAR and/or
 RGB can be used for mapping trees in the orchard to support navigation using SLAM techniques,
- 3510 with the downside of additional computational complexity.

3511 The machine vision applications of flowering and fruit load estimation were explored in this thesis. 3512 The same technology can be used to support auto-harvesting technology. Several attempts have 3513 been made to apply robotics to fruit harvest. Most of these attempts involve the use of standard 3514 industrial robotic arms with five to seven degrees of freedom in a protected crop environment (e.g., 3515 the Harvey capsicum harvester; https://research.gut.edu.au/future-farming/projects/harvey-the-3516 robotic-capsicum-sweet-pepper-harvester/, doa 5/11/2019). Such industrial robotic arms are 3517 expensive and operate slowly and may not be suitable for tree crops bearing hundreds of fruits per 3518 tree. More appropriate is the use of multiple linear actuators or picking arms arranged in some grid 3519 pattern specific to the crop application. For example, the AGROBOT strawberry harvester offers 3520 higher speed from multiple arms with limited (1 or 2 degrees of freedom) flexibility. The citrus 3521 harvester from Energid (www.energid.com) uses a multiple low-cost hydraulic picking mechanism 3522 arranged in a grid fashion. The FFRobotics (www.ffrobotics.com) apple harvester uses multiple linear 3523 actuators for picking of apple fruits.

Conclusion

3524 As a relatively high cost, large, firm fruit, hanging freely at the end of a long peduncle, the mango 3525 fruit is well suited to robotic harvesting. There is also a strong pull factor for this application in 3526 Australia, given shortage of farm labour and the harsh harvest time environment, with up to 40 °C 3527 ambient temperatures, and the release of an acidic sap from fruit on harvesting that can cause 3528 human skin damage. The CQUniversity team has recently developed the world's first mango 3529 harvesting robot (www.cqu.edu.au/cquninews/stories/general-category/2019/world-first-mango-3530 auto-harvester-from-cquni). This machine, designed to operate in existing Australian commercial 3531 orchards, is 3 meters tall and equipped with 9 linear arms that can extend 1.7 m into the canopy. A 3532 75% harvest efficiency on fruits visible to the system has been achieved to date. The eyes of the 3533 system are an RGB and ToF camera. The brain of the system is the deep learning MangoYOLO model 3534 developed in this thesis. This technology has great promise to replace human harvest labour, but 3535 numerous challenges remain. In the machine vision area, it is necessary to implement a selective 3536 harvesting capability, based on harvest maturity classification.

3537 Quantum fieri non multo – much has been done but there is much to do.

3539 List of References

3540 Aggelopoulou A, Bochtis D, Fountas S, Swain KC, Gemtos T, Nanos G (2011) Yield prediction in apple 3541 orchards based on image processing Precision Agriculture 12:448-456 3542 Anderson N, Underwood J, Rahman M, Robson A, Walsh K (2018) Estimation of fruit load in mango 3543 orchards: tree sampling considerations and use of machine vision and satellite imagery. 3544 Precision Agriculture doi: https://doi.org/10.1007/s1119-018-9614-1 3545 Anderson NT, Subedi PP, Walsh KB (2017) Manipulation of mango fruit dry matter content to 3546 improve eating quality Scientia Horticulturae 226:316-321 3547 Annamalai P, Lee WS, Burks TF Color vision system for estimating citrus yield in real-time. In: ASAE 3548 Annual Meeting, 2004. American Society of Agricultural and Biological Engineers, p 1. 3549 doi:https://doi.org/10.13031/2013.16714 Arad B, Kurtser P, Barnea E, Harel B, Edan Y, Ben-Shahar O (2019) Controlled Lighting and 3550 3551 Illumination-Independent Target Detection for Real-Time Cost-Efficient Applications. The 3552 Case Study of Sweet Pepper Robotic Harvesting Sensors 19:1390 3553 doi:https://doi.org/10.3390/s19061390 3554 Bargoti S, Underwood J Deep fruit detection in orchards. In: Proceedings - IEEE International 3555 Conference on Robotics and Automation, 2017a. pp 3626-3633. doi:https://doi.org/10.1109/ICRA.2017.7989417 3556 3557 Bargoti S, Underwood JP (2017b) Image Segmentation for Fruit Detection and Yield Estimation in 3558 Apple Orchards. Journal of Field Robotics 34:1039-1060 3559 doi:https://doi.org/10.1002/rob.21699 3560 Bay H, Tuytelaars T, Van Gool L Surf: Speeded up robust features. In: European Conference on 3561 Computer Vision, 2006. Springer, pp 404-417. doi:<u>https://doi.org/10.1007/11744023_32</u> 3562 Breiman L (2001) Random forests Machine learning 45:5-32 3563 Charoenpong T, Chamnongthai K, Kamhom P, Krairiksh M Volume measurement of mango by using 3564 2D ellipse model. In: Industrial Technology, 2004. IEEE ICIT'04. 2004 IEEE International 3565 Conference on, 2004. IEEE, pp 1438-1441 3566 Chen L, Papandreou G, Kokkinos I, Murphy K, Yuille AL (2018) DeepLab: Semantic Image 3567 Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs 3568 IEEE Transactions on Pattern Analysis and Machine Intelligence 40:834-848 3569 doi:https://doi.org/10.1109/TPAMI.2017.2699184 3570 Chen SW, Shivakumar SS, Dcunha S, Das J, Okon E, Qu C, Taylor CJ, Kumar V (2017) Counting apples and oranges with deep learning: A data-driven approach. IEEE Robotics and Automation 3571 3572 Letters 2:781-788 doi: https://doi.org/10.1109/LRA.2017.2651944 3573 Cheng H, Damerow L, Sun Y, Blanke M (2017a) Early Yield Prediction Using Image Analysis of Apple 3574 Fruit and Tree Canopy Features with Neural Networks Journal of Imaging 3:6 3575 Cheng H, Damerow L, Sun Y, Blanke M (2017b) Early yield prediction using image analysis of apple 3576 fruit and tree canopy features with neural networks. Journal of Imaging 3:6 3577 Choi D, Lee WS, Ehsani R, Schueller JK, Roka F Machine vision system for early yield estimation of 3578 citrus in a site-specific manner. In: ASABE Annual International Meeting, 2015. American 3579 Society of Agricultural and Biological Engineers, p 1. 3580 doi:https://doi.org/10.13031/aim.20152181863 3581 Chollet F Xception: Deep Learning with Depthwise Separable Convolutions. In: Proceedings of the 3582 IEEE Conference on Computer Vision and Pattern Recognition, 2017. pp 1251-1258 3583 Cortes C, Vapnik V (1995) Support-vector networks. Machine learning 20:273-297 3584 doi:https://doi.org/10.1007/BF00994018 3585 Črtomir R, Urška C, Stanislav T, Denis S, Karmen P, Pavlovič M, Marjan V (2012) Application of Neural Networks and Image Visualization for Early Forecast of Apple Yield. Erwerbs-Obstbau 54:69-3586 3587 76 doi:<u>https://doi.org/10.1007/s10341-012-0162-y</u>

3588	Dalal N, Triggs B Histograms of oriented gradients for human detection. In: IEEE Computer Society
3589 3590	Conference on Computer Vision and Pattern Recognition (CVPR), 2005. pp 886-893. doi: <u>https://doi.org/</u> 10.1109/CVPR.2005.177
3591	Dellen B, Rojas Jofre IA Volume measurement with a consumer depth camera based on structured
3592	infrared light. In: Proceedings of the 16th Catalan Conference on Artificial Intelligence,
3593	poster session, 2013. pp 1-10
3594	Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L Imagenet: A large-scale hierarchical image database.
3595	In: Proceedings - IEE conference on Computer Vision and Pattern Recognition, 2009. pp 248-
3596	255. doi: <u>https://doi.org/10.1109/CVPR.2009.5206848</u>
3597	Diago MP, Sanz-Garcia A, Millan B, Blasco J, Tardaguila J (2014) Assessment of flower number per
3598	inflorescence in grapevine by image analysis under field conditions Journal of the Science of
3599	Food and Agriculture 94:1981-1987 doi: <u>https://doi.org/10.1002/jsfa.6512</u>
3600	Dias PA, Tabb A, Medeiros H (2018a) Apple flower detection using deep convolutional networks
3601	Computers in Industry 99:17-28 doi: <u>https://doi.org/10.1016/j.compind.2018.03.010</u>
3602	Dias PA, Tabb A, Medeiros H (2018b) Multispecies Fruit Flower Detection Using a Refined Semantic
3603	Segmentation Network IEEE Robotics and Automation Letters 3:3003-3010
3604	doi: <u>https://doi.org/10.1109/LRA.2018.2849498</u>
3605	Dorj U-O, Lee M (2012) A new method for tangerine tree flower recognition. In: Computer
3606	Applications for Bio-technology, Multimedia, and Ubiquitous City, vol 353. Springer, pp 49-
3607	56. doi: <u>https://doi.org/10.1007/978-3-642-35521-9_7</u>
3608	Everingham M, Van Gool L, Williams CK, Winn J, Zisserman A (2010) The pascal visual object classes
3609	(voc) challenge. International Journal of Computer Vision 88:303-338
3610	doi: <u>https://doi.org/10.1007/s11263-009-0275-4</u>
3611	Girshick R Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision,
3612	2015. pp 1440-1448. doi: <u>https://doi.org/10.1109/ICCV.2015.169</u>
3613	Girshick R, Donahue J, Darrell T, Malik J Rich feature hierarchies for accurate object detection and
3614	semantic segmentation. In: Proceedings of the IEEE Computer Society Conference on
3615	Computer Vision and Pattern Recognition, 2014. pp 580-587.
3616	doi: <u>https://doi.org/10.1109/CVPR.2014.81</u>
3617	Gongal A, Amatya S, Karkee M, Zhang Q, Lewis K (2015) Sensors and systems for fruit detection and
3618	localization: A review. Computers and Electronics in Agriculture 116:8-19
3619	doi: <u>https://doi.org/10.1016/j.compag.2015.05.021</u>
3620	Green A, McAneney K, Astill M (1990) An instrument for measuring kiwifruit size New Zealand
3621	journal of crop and horticultural science 18:115-120
3622	Guo W, Fukatsu T, Ninomiya S (2015) Automated characterization of flowering dynamics in rice using
3623	field-acquired time-series RGB images Plant Methods 11:7
3624	doi: <u>https://doi.org/10.1186/s13007-015-0047-9</u>
3625	Habaragamuwa H, Ogawa Y, Suzuki T, Shiigi T, Ono M, Kondo N (2018) Detecting greenhouse
3626	strawberries (mature and immature), using deep convolutional neural network. Engineering
3627	in Agriculture, Environment and Food 11:127-138
3628	doi: <u>https://doi.org/10.1016/j.eaef.2018.03.001</u>
3629	He K, Gkioxari G, Dollar P, Girshick R Mask R-CNN. In: Proceedings of the IEEE International
3630	Conference on Computer Vision, 2017. pp 2980-2988.
3631	doi: <u>https://doi.org/10.1109/ICCV.2017.322</u>
3632	He K, Zhang X, Ren S, Sun J Spatial pyramid pooling in deep convolutional networks for visual
3633	recognition. In: European Conference on Computer Vision, 2014. Springer, pp 346-361.
3634	doi: <u>https://doi.org/10.1007/978-3-319-10578-9_23</u>
3635	He K, Zhang X, Ren S, Sun J Deep residual learning for image recognition. In: Proceedings of the IEEE
3636	Computer Society Conference on Computer Vision and Pattern Recognition, 2016. pp 770-
3637	778

3638	Herold B, Kawano S, Sumpf B, Tillmann P, Walsh KB (2009) Chapter 3. VIS/NIR Spectroscopy. In: Zude
3639	M (ed) Optical Monitoring of Fresh and Processed Agricultural Crops. CRC Press, Boca Raton,
3640	USA, pp 141-249
3641	Hočevar M, Širok B, Godeša T, Stopar M (2014) Flowering estimation in apple orchards by image
3642	analysis Precision Agriculture 15:466-478 doi: <u>https://doi.org/10.1007/s11119-013-9341-6</u>
3643	Horton R, Cano E, Bulanon D, Fallahi E (2017) Peach flower monitoring using aerial multispectral
3644	imaging Journal of Imaging 3:2 doi: <u>https://doi.org/10.3390/jimaging3010002</u>
3645	Hu J, Shen L, Sun G (2017) Squeeze-and-excitation networks. arXiv preprint arXiv:170901507 7
3646	Hung C, Underwood J, Nieto J, Sukkarieh S A feature learning based approach for automated fruit
3647	yield estimation. In: Field and Service Robotics, 2015. Springer, pp 485-498.
3648	doi: <u>https://doi.org/10.1007/978-3-319-07488-7_33</u>
3649	Jiang Y, Zhu X, Wang X, Yang S, Li W, Wang H, Fu P, Luo Z (2017) R2cnn: Rotational region cnn for
3650	orientation robust scene text detection arXiv preprint arXiv:170609579
3651	Jiang Z, Liu C, Hendricks NP, Ganapathysubramanian B, Hayes DJ, Sarkar S (2018) Predicting County
3652	Level Corn Yields Using Deep Long Short Term Memory Models. arXiv preprint
3653	arXiv:180512044
3654	Jimenez A, Ceres R, Pons J (2000) A survey of computer vision methods for locating fruit on trees.
3655	Transactions of the ASAE 43:1911-1920 doi: <u>https://doi.org/10.13031/2013.3096</u>
3656	Kader AA Fruit maturity, ripening, and quality relationships. In: International Symposium Effect of
3657	Pre-& Postharvest factors in Fruit Storage 485, 1997. pp 203-208
3658	Kadir MFA, Yusri NAN, Rizon M, bin Mamat AR, Makhtar M, Jamal AA (2015) Automatic mango
3659	detection using texture analysis and randomised hough transform. Applied Mathematical
3660	Sciences 9:6427-6436 doi:https://doi.org/10.12988/ams.2015.53290
3661	Kamilaris A, Prenafeta-Boldú FX (2018) Deep learning in agriculture: A survey. Computers and
3662	Electronics in Agriculture 147:70-90 doi:https://doi.org/10.1016/j.compag.2018.02.016
3663	Kestur R, Meduri A, Narasipura O (2019) MangoNet: A deep semantic segmentation architecture for
3664	a method to detect and count mangoes in an open orchard. Engineering Applications of
3665	Artificial Intelligence 77:59-69 doi: <u>https://doi.org/10.1016/j.engappai.2018.09.011</u>
3666	Kingma DP, Ba J (2014) Adam: A Method for Stochastic Optimization. arXiv e-prints
3667	Koirala A, Walsh K, Wang Z, McCarthy C Mobile device machine vision estimation of mano crop load.
3668	In: International Tri-Conference for Precision Agriculture, New Zealand, 2017.
3669	doi:https://doi.org/10.5281/zenodo.895382
3670	Koirala A, Walsh KB, Wang Z, McCarthy C (2019a) Deep learning – Method overview and review of
3671	use for fruit detection and yield estimation Computers and Electronics in Agriculture
3672	162:219-234 doi: <u>https://doi.org/10.1016/j.compag.2019.04.017</u>
3673	Koirala A, Wang Z, Walsh K, McCarthy C (2019b) Deep learning for real-time fruit detection and
3674	orchard fruit load estimation: benchmarking of <i>'MangoYOLO'</i> . Precision Agriculture 20:1107-
3675	1135 doi:https://doi.org/10.1007/s11119-019-09642-0
3676	Kongsro J (2014) Estimation of pig weight using a Microsoft Kinect prototype imaging system
3677	Computers and Electronics in Agriculture 109:32-35
3678	doi:https://doi.org/10.1016/j.compag.2014.08.008
3679	
	Krizhevsky A, Sutskever I, Hinton GE ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, 2012. pp 1097-1105
3680	
3681	Kumar N, Belhumeur PN, Biswas A, Jacobs DW, Kress WJ, Lopez IC, Soares JVB Leafsnap: A Computer
3682	Vision System for Automatic Plant Species Identification. In, Berlin, Heidelberg, 2012.
3683	Computer Vision – ECCV 2012. Springer Berlin Heidelberg, pp 502-516
3684	Kurtulmus F, Lee WS, Vardar A (2011) Green citrus detection using 'eigenfruit', color and circular
3685	Gabor texture features under natural outdoor conditions. Computers and Electronics in
3686	Agriculture 78:140-149 doi: <u>https://doi.org/10.1016/j.compag.2011.07.001</u>

3687 3688	Kuwata K, Shibasaki R Estimating Crop Yields with Deep Learning and Remotely Sensed Data. In: IEEE International Geoscience and Remote Sensing Symposium (IGARSS), 2015. pp 858-861.
3689	doi: <u>https://doi.org/10.1109/IGARSS.2015.7325900</u>
3690	LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document
3691	recognition. Proceedings of the IEEE 86:2278-2323 doi: <u>https://doi.org/10.1109/5.726791</u>
3692	Liakos K, Busato P, Moshou D, Pearson S, Bochtis D (2018) Machine learning in agriculture: A review.
3693	Sensors 18:2674 doi:https://doi.org/10.3390/s18082674
3694	Liang Q, Zhu W, Long J, Wang Y, Sun W, Wu W A Real-Time Detection Framework for On-Tree Mango
3695	Based on SSD Network. In: International Conference on Intelligent Robotics and
3696	Applications, 2018. Springer, pp 423-436. doi:https://doi.org/10.1007/978-3-319-97589-
3697	4 36
3698	Liaw A, Wiener M (2002) Classification and regression by randomForest R news 2:18-22
3699	Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S Feature pyramid networks for object
3700	detection. In: IEE Conference on Computer Vision and Pattern Recognition, 2017a. pp 936-
3701	944. doi: <u>https://doi.org/10.1109/CVPR.2017.106</u>
3702	Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL Microsoft coco:
3703	Common objects in context. In: European conference on computer vision, 2014. Springer, pp
3704	740-755
3705	Lin TY, Goyal P, Girshick R, He K, Dollar P Focal Loss for Dense Object Detection. In: Proceedings of
3706	the IEEE International Conference on Computer Vision, 2017b. pp 2999-3007.
3707	doi:https://doi.org/10.1109/ICCV.2017.324
3708	Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC Ssd: Single shot multibox detector.
3709	In: European conference on computer vision, 2016. Springer, pp 21-37.
3710	doi: <u>https://doi.org/10.1007/978-3-319-46448-0_2</u>
3711	Liu X, Chen SW, Aditya S, Sivakumar N, Dcunha S, Qu C, Taylor CJ, Das J, Kumar V (2018) Robust Fruit
3712	Counting: Combining Deep Learning, Tracking, and Structure from Motion. arXiv preprint
3713	arXiv:180400307
3714	Long J, Shelhamer E, Darrell T Fully convolutional networks for semantic segmentation. In:
3715	Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2015. pp
3716	3431-3440
3717	Lowe DG Object recognition from local scale-invariant features. In: Proceedings of the Seventh IEEE
3718	International Conference on Computer Vision, 1999. pp 1150-1157.
3719	doi:https://doi.org/10.1109/ICCV.1999.790410
3720	Margetts J (2014) Australian Mango Industry Strategic Investment Plan 2014/15 - 2018/19. Plant &
3721	Food RESEARCH, Australia
3722	McCool C, Sa I, Dayoub F, Lehnert C, Perez T, Upcroft B Visual detection of occluded crop: For
3723	automated harvesting. In: IEEE International Conference on Robotics and Automation 2016.
3723	pp 2506-2512. doi:https://doi.org/10.1109/ICRA.2016.7487405
3724	Mitchell P (1986) Pear fruit growth and the use of diameter to estimate fruit volume and weight.
3725	HortScience 21:1003-1005
3720	Moonrinta J, Chaivivatrakul S, Dailey MN, Ekpanyapong M Fruit detection, tracking, and 3D
3728	reconstruction for crop mapping and yield estimation. In: 11th International Conference on
3729	Control Automation Robotics & Vision, 2010. pp 1181-1186.
3730	doi: <u>https://doi.org/10.1109/ICARCV.2010.5707436</u>
3731	Morandi B, Manfrini L, Zibordi M, Noferini M, Fiori G, Grappadelli LC (2007) A low-cost device for
3732	accurate and continuous measurements of fruit diameter HortScience 42:1380-1382
3733	Moreda GP, Ortiz-Cañavate J, García-Ramos FJ, Ruiz-Altisent M (2009) Non-destructive technologies
3734 3735	for fruit and vegetable size determination – A review Journal of Food Engineering 92:119- 136 doi:http://dx.doi.org/10.1016/i.jfoodeng.2008.11.004
3/33	100 U01.11(10.//UX.U01.01g/10.1010/1.11000efig.2008.11.004

3736	Murali R, Won Suk L (2005) Citrus Fruit Identification and Size Determination Using Machine Vision
3737	and Ultrasonic Sensors. Paper presented at the ASAE Annual International Meeting, St.
3738	Joseph, Mich.,
3739	Mureşan H, Oltean M (2018) Fruit recognition from images using deep learning Acta Universitatis
3740	Sapientiae, Informatica 10:26-42
3741	Naik S, Patel B (2017) Machine Vision based Fruit Classification and Grading-A Review. International
3742	Journal of Computer Applications (0975-8887) 170:22-34
3743	Nanaa K, Rizon M, Rahman MNA, Ibrahim Y, Aziz AZA Detecting mango fruits by using randomized
3744	hough transform and backpropagation neural network. In: Proceedings of the International
3745	Conference on Information Visualisation, 2014. pp 388-391.
3746	doi: <u>https://doi.org/10.1109/IV.2014.54</u>
3747	Norris K (1996) History of NIR. Journal of Near Infrared Spectroscopy 4:31-37
3748	doi: <u>https://doi.org/10.1255/jnirs.941</u>
3749	Ojala T, Pietikäinen M, Harwood D (1996) A comparative study of texture measures with
3750	classification based on featured distributions. Pattern Recognition 29:51-59
3751	doi: <u>https://doi.org/10.1016/0031-3203(95)00067-4</u>
3752	Oppenheim D, Edan Y, Shani G (2017) Detecting Tomato Flowers in Greenhouses Using Computer
3753	Vision World Academy of Science, Engineering and Technology, International Journal of
3754	Computer, Electrical, Automation, Control and Information Engineering 11:104-109
3755	Otsu N (1975) A threshold selection method from gray-level histograms Automatica 11:23-27
3756	Payne A, Walsh K (2014) Chapter 16. Machine vision in estimation of fruit crop yield. In: Ibaraki Y,
3757	Gupta SD (eds) Plant Image Analysis: Fundamentals and Applications. CRC Press, Boca Raton,
3758	FL, USA, pp 329-374
3759	Payne A, Walsh K, Subedi P, Jarvis D (2014) Estimating mango crop yield using image analysis using
3760	fruit at 'stone hardening' stage and night time imaging. Computers and Electronics in
3761	Agriculture 100:160-167 doi:https://doi.org/10.1016/j.compag.2013.11.011
3762	Payne AB, Walsh KB, Subedi P, Jarvis D (2013) Estimation of mango crop yield using image analysis-
3763	segmentation method. Computers and Electronics in Agriculture 91:57-64
3764	doi: <u>https://doi.org/10.1016/j.compag.2012.11.009</u>
3765	Peng H, Huang B, Shao Y, Li Z, Zhang C, Chen Y, Xiong J (2018) General improved SSD model for
3766	picking object recognition of multiple fruits in natural environment. Transactions of the
3767	Chinese Society of Agricultural Engineering 34:155-162
3768	doi:https://doi.org/10.11975/j.issn.1002-6819.2018.16.020
3769	Pothen ZS, Nuske S Texture-based fruit detection via images using the smooth patterns on the fruit.
3770	In: IEEE International Conference on Robotics and Automation 2016. pp 5171-5176.
3771	doi:https://doi.org/10.1109/ICRA.2016.7487722
3772	Qian J, Xing B, Wu X, Chen M, Wang Ya (2018) A smartphone-based apple yield estimation
3773	application using imaging features and the ANN method in mature period Scientia Agricola
3774	75:273-280 doi: <u>https://doi.org/10.1590/1678-992X-2016-0152</u>
3775	Qureshi WS, Payne A, Walsh KB, Linker R, Cohen O, Dailey MN (2017) Machine vision for counting
3776	fruit on mango tree canopies. Precision Agriculture 18:224-244
3777	doi:https://doi.org/10.1007/s11119-016-9458-5
3778	Rahman M, Robson A, Bristow M (2018) Exploring the Potential of High Resolution WorldView-3
3779	Imagery for Estimating Yield of Mango. Remote Sensing 10:1866
3780	doi:https://doi.org/10.3390/rs10121866
3781	Rahnemoonfar M, Sheppard C (2017) Deep count: Fruit counting based on deep simulated learning.
3782	Sensors 17 doi:https://doi.org/10.3390/s17040905
3783	Redmon J (2018) Darknet: Open Source Neural Networks in C. <u>https://pjreddie.com/darknet/</u> .
3784	Accessed 23/03/2018

3785	Redmon J, Divvala S, Girshick R, Farhadi A You only look once: Unified, real-time object detection. In:
3786	Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern
3787	Recognition, 2016. pp 779-788. doi: <u>https://doi.org/10.1109/CVPR.2016.91</u>
3788	Redmon J, Farhadi A YOLO9000: Better, Faster, Stronger. In: Proceedings of the IEEE Conference on
3789	Computer Vision and Pattern Recognition, 2017. pp 7263-7271.
3790	doi: <u>https://doi.org/10.1109/CVPR.2017.690</u>
3791	Redmon J, Farhadi A (2018) YOLOv3: An Incremental Improvement. arXiv preprint arXiv:180402767
3792	Ren S, He K, Girshick R, Sun J Faster R-CNN: Towards real-time object detection with region proposal
3793	networks. In: Advances in Neural Information Processing Systems, 2015. pp 91-99
3794	Rousseeuw PJ (1987) Silhouettes: A graphical aid to the interpretation and validation of cluster
3795	analysis Journal of Computational and Applied Mathematics 20:53-65
3796	doi: <u>https://doi.org/10.1016/0377-0427(87)90125-7</u>
3797	Sa I, Ge Z, Dayoub F, Upcroft B, Perez T, McCool C (2016) Deepfruits: A fruit detection system using
3798	deep neural networks. Sensors 16 doi: <u>https://doi.org/10.3390/s16081222</u>
3799	Sa I, McCool C, Lehnert C, Perez T On visual detection of highly-occluded objects for harvesting
3800	automation in horticulture. In: IEEE International Conference on Robotics and Automation,
3801	2015.
3802	Seifert B, Zude M, Spinelli L, Torricelli A (2015) Optical properties of developing pip and stone fruit
3803	reveal underlying structural changes Physiologia plantarum 153:327-336
3804	Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D Grad-CAM: Visual Explanations from
3805	Deep Networks via Gradient-Based Localization. In: IEEE International Conference on
3806	Computer Vision (ICCV), 22-29 Oct. 2017 2017. pp 618-626.
3807	doi: <u>https://doi.org/10.1109/ICCV.2017.74</u>
3808	Sengupta S, Lee WS (2014) Identification and determination of the number of immature green citrus
3809	fruit in a canopy under different ambient light conditions. Biosystems Engineering 117:51-61
3810	doi: <u>https://doi.org/10.1016/j.biosystemseng.2013.07.007</u>
3811	Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R, LeCun Y (2013) Overfeat: Integrated
3812	recognition, localization and detection using convolutional networks. arXiv preprint
3813	arXiv:13126229
3814	Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition.
3815	arXiv preprint arXiv:14091556
3816	Song Y, Glasbey C, Horgan G, Polder G, Dieleman J, Van der Heijden G (2014) Automatic fruit
3817	recognition and counting from multiple images. Biosystems Engineering 118:203-215
3818	doi:https://doi.org/10.1016/j.biosystemseng.2013.12.008
3819	Spreer W, Müller J (2011) Estimating the mass of mango fruit (Mangifera indica, cv. Chok Anan) from
3820	its geometric dimensions by optical measurement Computers and electronics in agriculture
3821	75:125-131
3822	Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M (2014) Striving for simplicity: The all
3823	convolutional net. arXiv preprint arXiv:14126806
3824	Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: A Simple Way to
3825	Prevent Neural Networks from Overfitting. The Journal of Machine Learning Research
3826	15:1929-1958
3827	Stajnko D, Čmelik Z (2005) Modelling of apple fruit growth by application of image analysis.
3828	Agriculturae Conspectus Scientificus 70:59-64
3829	Stajnko D, Rakun J, Blanke M (2009) Modelling apple fruit yield using image analysis for fruit colour,
3830	shape and texture. European Journal of Horticultural Science 74:260-267
3831	Stein M, Bargoti S, Underwood J (2016) Image based mango fruit detection, localisation and yield
3832	estimation using multiple view geometry. Sensors 16 doi:10.3390/s16111915
3833	Syal A, Garg D, Sharma S (2013) A survey of computer vision methods for counting fruits and yield
3834	prediction. International Journal of Computer Science Engineering 2:346-350

3835 Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and 3836 3837 Pattern Recognition, 2015. pp 1-9. doi:https://doi.org/10.1109/CVPR.2015.7298594 Szegedy C, Reed S, Erhan D, Anguelov D, Ioffe S (2014) Scalable, high-quality object detection. arXiv 3838 3839 preprint arXiv:14121441 3840 Tao Y, Zhou J, Wang K, Shen W Rapid detection of fruits in orchard scene based on deep neural network. In: ASABE 2018 Annual International Meeting, 2018. 3841 3842 doi:https://doi.org/10.13031/aim.201801055 3843 Turk M, Pentland A (1991) Eigenfaces for recognition. Journal of Cognitive Neuroscience 3:71-86 3844 doi:https://doi.org/10.1162/jocn.1991.3.1.71 Uijlings JR, Van De Sande KE, Gevers T, Smeulders AW (2013) Selective search for object recognition. 3845 International Journal of Computer Vision 104:154-171 doi:https://doi.org/10.1007/s11263-3846 3847 013-0620-5 3848 Underwood JP, Hung C, Whelan B, Sukkarieh S (2016) Mapping almond orchard canopy volume, 3849 flowers, fruit and yield using lidar and vision sensors Computers and Electronics in 3850 Agriculture 130:83-96 doi:https://doi.org/10.1016/j.compag.2016.09.014 3851 Underwood JP, Rahman MM, Robson A, Walsh KB, Koirala A, Wang Z (2018) Fruit load estimation in 3852 mango orchards - a method comparison. Paper presented at the ICRA 2018 Workshop on 3853 Robotic Vision and Action in Agriculture, Brisbane, Australia, 3854 van Eck JW, van der Heijden GWAM, Polder G (1998) Accurate Measurement of Size and Shape of 3855 Cucumber Fruits with Image Analysis Journal of Agricultural Engineering Research 70:335-343 doi:http://dx.doi.org/10.1006/jaer.1998.0285 3856 3857 Viola P, Jones M Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 3858 3859 2001. doi:https://doi.org/10.1109/CVPR.2001.990517 3860 Wachs J, Stern H, Burks T, Alchanatis V, Bet-Dagan I Apple detection in natural tree canopies from 3861 multimodal images. In: Proceedings of the 7th European Conference on Precision 3862 Agriculture, 2009. pp 293-302 3863 Walsh K, Wang Z (2018) Monitoring fruit quality and quantity in mangoes. In: Galán Saúco V, Lu P 3864 (eds) Achieving sustainable cultivation of mangoes. Burleigh Dodds Science Publishing, Cambridge, UK, pp 313-338. doi:http://doi.org/10.19103/AS.2017.0026.14 3865 Walsh KB (2018) Fruit and Vegetable Packhouse: Technologies for Assessing Fruit Quantity and 3866 Quality. In: Chen G (ed) Advances in Agricultural Machinery and Technologies. CRC Press, 3867 3868 Boca Raton, pp 367 – 395 3869 Wang Q, Nuske S, Bergerman M, Singh S (2013) Automated crop yield estimation for apple orchards. 3870 In: Experimental Robotics, vol 88. Springer, pp 745-758. doi:https://doi.org/10.1007/978-3-3871 319-00065-7 50 3872 Wang Z, Koirala A, Walsh K, Anderson N, Verma B (2018a) In Field Fruit Sizing Using A Smart Phone 3873 Application Sensors 18:3331 doi:https://doi.org/10.3390/s18103331 3874 Wang Z, Underwood J, Walsh KB (2018b) Machine vision assessment of mango orchard flowering 3875 Computers and Electronics in Agriculture 151:501-511 3876 doi:https://doi.org/10.1016/j.compag.2018.06.040 3877 Wang Z, Verma B, Walsh KB, Subedi P, Koirala A Automated mango flowering assessment via 3878 refinement segmentation. In: International Conference on Image and Vision Computing New 3879 Zealand (IVCNZ) 2016. IEEE, pp 1-6. doi: <u>https://doi.org/10.1109/IVCNZ.2016.7804426</u> 3880 Wang Z, Walsh K, Koirala A (2019) Mango Fruit Load Estimation Using a Video Based MangoYOLO-3881 Kalman Filter—Hungarian Algorithm Method Sensors 19:2742 3882 doi:https://doi.org/10.3390/s19122742 3883 Wang Z, Walsh KB, Verma B (2017a) On-tree mango fruit size estimation using RGB-D images Sensors 3884 17:2738

3885	Wang Z, Walsh KB, Verma B (2017b) On-Tree Mango Fruit Size Estimation Using RGB-D Images.
3886	Sensors 17 doi: <u>https://doi.org/10.3390/s17122738</u>
3887	Wojke N, Bewley A, Paulus D Simple online and realtime tracking with a deep association metric. In:
3888	2017 IEEE International Conference on Image Processing (ICIP), 2017. IEEE, pp 3645-3649
3889	Xiong J, Liu Z, Tang L, Lin R, Bu R, Peng H (2018) Visual Detection Technology of Green Citrus under
3890	Natural Environment. Nongye Jixie Xuebao/Transactions of the Chinese Society for
3891	Agricultural Machinery 49:45-52 doi: <u>https://doi.org/10.6041/j.issn.1000-1298.2018.04.005</u>
3892	You J, Li X, Low M, Lobell D, Ermon S Deep Gaussian Process for Crop Yield Prediction Based on
3893	Remote Sensing Data. In: 31st AAAI Conference on Artificial Intelligence, AAAI 2017, 2017.
3894	pp 4559-4565
3895	Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. vol 8689 LNCS.
3896	doi: <u>https://doi.org/10.1007/978-3-319-10590-1_53</u>
3897	Zeng X, Ouyang W, Yan J, Li H, Xiao T, Wang K, Liu Y, Zhou Y, Yang B, Wang Z (2018) Crafting gbd-net
3898	for object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence
3899	40:2109-2123 doi: <u>https://doi.org/10.1109/TPAMI.2017.2745563</u>
3900	Zhao Y, Gong L, Zhou B, Huang Y, Liu C (2016) Detecting tomatoes in greenhouse scenes by
3901	combining AdaBoost classifier and colour analysis. Biosystems Engineering 148:127-137
3902	doi: <u>https://doi.org/10.1016/j.biosystemseng.2016.05.001</u>
3903	Zhou R, Damerow L, Sun Y, Blanke MM (2012) Using colour features of cv.'Gala'apple fruits in an
3904	orchard in image processing to predict yield. Precision Agriculture 13:568-580
3905	doi: <u>https://doi.org/10.1007/s11119-012-9269-2</u>
3906	Zude M, Pflanz M, Spinelli L, Dosche C, Torricelli A (2011) Non-destructive analysis of anthocyanins in
3907	cherries by means of Lambert–Beer and multivariate regression based on spectroscopy and
3908	scatter correction using time-resolved analysis Journal of Food Engineering 103:68-75
3909	doi: <u>https://doi.org/10.1016/j.jfoodeng.2010.09.021</u>
3910	
3911	
3912	
3913	
2212	
3914	
3915	
3916	
3917	

3918 Appendix

3919 Other Published Research

3920 The main body of this thesis contained the key publications made in the context of the use of 3921 machine learning and deep learning CNN techniques for mango flower detection and assessment, 3922 fruit detection, counting and fruit load estimation. Some of the research follows preliminary work 3923 conducted for fruit detection and flower assessment using traditional object segmentation methods 3924 which were also published. Moreover, some methods extend fruit detection method developed in 3925 this thesis for use with tracking algorithms as a major component of yield estimation framework 3926 which were also published. This section contains these papers listed in chronological order as 3927 follows: 3928 Wang Z, Walsh K, Koirala A (2019) Mango Fruit Load Estimation Using a Video Based MangoYOLO— 3929 Kalman Filter—Hungarian Algorithm Method Sensors 19:2742 3930 doi:https://doi.org/10.3390/s19122742 3931 3932 Underwood JP, Rahman MM, Robson A, Walsh KB, Koirala A, Wang Z (2018) Fruit load estimation in 3933 mango orchards - a method comparison. Paper presented at the ICRA 2018 Workshop on 3934 Robotic Vision and Action in Agriculture, Brisbane, Australia. 3935 3936 Koirala A, Walsh K, Wang Z, McCarthy C Mobile device machine vision estimation of mano crop load. 3937 In: International Tri-Conference for Precision Agriculture, New Zealand, 2017. 3938 doi:https://doi.org/10.5281/zenodo.895382 3939 3940 3941 Wang Z, Verma B, Walsh KB, Subedi P, Koirala A Automated mango flowering assessment via 3942 refinement segmentation. In: International Conference on Image and Vision Computing New 3943 Zealand (IVCNZ) 2016 IEEE, pp 1-6. doi:https://doi.org/10.1109/IVCNZ.2016.7804426 3944 3945 3946 3947 3948 3949 3950 3951 3952 3953 3954 3955

3956 Appendix A. Published version of Chapter 3

Mango Fruit Load Estimation Using a Video Based MangoYOLO-Kalman Filter-Hungarian Algorithm Method

- Wang Z, Walsh K, Koirala A (2019) Mango Fruit Load Estimation Using a Video Based MangoYOLO—
 Kalman Filter—Hungarian Algorithm Method Sensors 19:2742
- doi:https://doi.org/10.3390/s19122742



Article

MDPI

Mango Fruit Load Estimation Using a Video Based MangoYOLO—Kalman Filter—Hungarian Algorithm Method

Zhenglin Wang ^{1,*0}, Kerry Walsh ^{2,*0} and Anand Koirala ²

- ¹ Centre for Intelligent Systems, Central Queensland University, Rockhampton North 4701, Australia
- ² Institute of Future Farming Systems, Central Queensland University, Rockhampton North 4701, Australia; a.koirala@cqu.edu.au
- * Correspondence: z.wang@cqu.edu.au (Z.W.); k.walsh@cqu.edu.au (K.W.)

Received: 11 May 2019; Accepted: 14 June 2019; Published: 18 June 2019



Abstract: Pre-harvest fruit yield estimation is useful to guide harvesting and marketing resourcing, but machine vision estimates based on a single view from each side of the tree ("dual-view") underestimates the fruit yield as fruit can be hidden from view. A method is proposed involving deep learning, Kalman filter, and Hungarian algorithm for on-tree mango fruit detection, tracking, and counting from 10 frame-per-second videos captured of trees from a platform moving along the inter row at 5 km/h. The deep learning based mango fruit detection algorithm, MangoYOLO, was used to detect fruit in each frame. The Hungarian algorithm was used to correlate fruit between neighbouring frames, with the improvement of enabling multiple-to-one assignment. The Kalman filter was used to predict the position of fruit in following frames, to avoid multiple counts of a single fruit that is obscured or otherwise not detected with a frame series. A "borrow" concept was added to the Kalman filter to predict fruit position when its precise prediction model was absent, by borrowing the horizontal and vertical speed from neighbouring fruit. By comparison with human count for a video with 110 frames and 192 (human count) fruit, the method produced 9.9% double counts and 7.3% missing count errors, resulting in around 2.6% over count. In another test, a video (of 1162 frames, with 42 images centred on the tree trunk) was acquired of both sides of a row of 21 trees, for which the harvest fruit count was 3286 (i.e., average of 156 fruit/tree). The trees had thick canopies, such that the proportion of fruit hidden from view from any given perspective was high. The proposed method recorded 2050 fruit (62% of harvest) with a bias corrected Root Mean Square Error (RMSE) = 18.0 fruit/tree while the dual-view image method (also using MangoYOLO) recorded 1322 fruit (40%) with a bias corrected RMSE = 21.7 fruit/tree. The video tracking system is recommended over the dual-view imaging system for mango orchard fruit count.

Keywords: crop load; Kalman filter; deep learning; YOLO; Hungarian assignment; tree fruit load

1. Introduction

A mango harvest yield prediction is ideally made six weeks before harvest, after the period of fruit drop in fruit development, to inform harvesting and marketing decisions [1]. However, mango production varies between seasons due to a biennial bearing habit [2] and to environmental (e.g., temperatures) and management (e.g., pruning) variables, and thus the relationship between historical production and the current season can be poor. Further, mango tree fruit load is variable within a given orchard and season, such that a statistically valid estimate of pre-harvest crop load requires the assessment of a large number of trees [3].

Given the required number of trees for a statistically valid estimate of average fruit load, the use of field counts by human operators is impractical [3]. The use of machine vision (MV) based on RGB

Sensors 2019, 19, 2742; doi:10.3390/s19122742

www.mdpi.com/journal/sensors

images of tree canopies has been trialed for tree fruit yield estimation by a number of research groups. When fruit colour is distinct to foliage, a simple segmentation method using colour features can be successful for fruit detection. For example, Zaman et al. [4] used colour features in the estimation of blueberry yield, with a high R^2 of 0.99 achieved, likely due to the distinct blue colour of the fruit. Zhou et al. [5] used red colour features in the detection of mature 'Gala' apples, with a reported R^2 of 0.8. However, in other applications, fruit may be green or have varied colours. Also, perceived object colour is influenced by ambient light. Annamalai and Suk Lee [6] achieved a R^2 of only 0.76 on human count of images using a MV technique based on colour to count mature citrus fruit in images of tree canopies.

A number of researchers have given attention to the shape and texture features in the detection of fruit. Low level feature detection algorithms exploit features such as object edges [7], corners [8], blobs [9], and ridges [10]. Various high-level feature extraction algorithms have also been used to evaluate small regions of interest, generating a set of feature descriptors (or feature vectors). Popular algorithms include Scale Invariant Feature Transform (SIFT) [11], Speeded Up Robust Features (SURF) [12], Histogram of Oriented Gradients (HOG) [13], and Local Binary Patterns (LBP) [14]. For example, Kurtulmus et al. [15] extracted circular Gabor texture features for green citrus yield estimation. Zhou et al. [5] used the shape features in the estimation of grape yield. Linker et al. [16] and Qureshi et al. [17] utilised arc features to estimate the number of green apples and mango fruit, respectively, in an orchard. In these reports, colour features can be a supporting, but not dominant, criterion.

Deep learning neural networks have an object-recognition ability that can outperform humans, as demonstrated in [18–21]. The models are essentially black box in that it is unclear what image features are used. Deep learning algorithms have been recommended for tree fruit detection, as reviewed by Koirala, et al. [22]. The first report was made by Sa, et al. [23] using the Faster Region-Convolutional Neural Networks (Faster-RCNN) algorithm [24] to detected multi-coloured (green, red, or yellow) capsicum fruit. Chen et al. [25] used a Fully Convolutional Network [26] to count fruit in apple and orange orchards. Bargoti and Underwood [27] used Faster-RCNN and transfer learning to estimate the yield of apple, mango, and almond orchards.

The use of MV in the estimation of mature mango fruit yield was first reported by Payne, et al. [28], based on a segmentation approach that relied on colour features. A R² of 0.74 and bias corrected RMSE of 7.7 fruit per tree was achieved. The authors reported on the impact of varying ambient light on detection accuracy, and night imaging with artificial lighting was adopted in their subsequent research [29]. They also reported the use of SIFT features to improve detection accuracy (R² improved from 0.78 to 0.85) [17]. The problems of imaging in daylight were addressed using Xe flash lamps and very short exposure times [30]. These authors employed Faster-RCNN and 'multi-view' imagery to obtain a R² > 0.9 on mango fruit count per tree. In a continuation study, Bargoti and Underwood [27] reported a F1-score of 0.908 and precision of 0.958 achieved for the estimation of mango yield per tree. Anderson et al. [3] compared the estimates for several orchards, and concluded that the method based on multi-view imaging and Faster-RCNN gave a better result than a method based on satellite imagery [31] or methods based on manual counts of a sample of trees. Subsequently, Koirala et al. [32] reported improved results for mango fruit load estimation using a You Only Look Once (YOLO) based deep learning architecture [33,34] termed 'MangoYOLO', relative to the Faster R-CNN architecture (F1 score of 0.968 and 0.945 for MangoYOLO and Faster-RCNN, respectively, on the same image set). In the current study, we adopt MangoYOLO as the state-of-art detection and localization algorithm for mango fruit

A number of the previous MV based imaging systems for orchard assessment have employed a dual-view imaging approach in which one image from each side of each tree, centered on the tree trunk, is acquired (e.g., [35]). However, dual-view imaging gives only one perspective of each side of the tree, with a proportion of fruit hidden from view. The proportion of 'hidden' fruit varies with canopy density, and thus Koirala et al. [32] relied on the use of a correction factor based on a human

count of fruit on a sample of trees for each orchard. The highest factor reported by Koirala, et al. [32] for a given mango orchard was 2.43, associated with large trees and dense foliage, and the lowest factor was 1.05, associated with open canopies and sparse foliage. However, 'dual view' imaging can result in a high number of unobserved fruit per tree (i.e., occluded fruit), and the ultimate estimate of orchard fruit count is sensitive to the estimate of the correction factor.

Imaging of each tree from multiple perspectives, for example from a camera on a vehicle moving down the orchard row, allows for detection of a greater proportion of fruit on a tree. In this case there is less sensitivity to error in the estimate of the correction factor, but a system is required to track fruit between images to avoid multiple counting of fruit. Liu et al. [36] summarized three major issues causing over counting in video based fruit yield estimation—(1) double count of the same fruit detected in consecutive frames; (2) double count of a fruit which is tracked across several frames, not detected in a frame due to occlusion or temporary failure of detection, and then detected and tracked again; and (3) double count of fruit seen from both sides of a tree. Another category can be added—(4) mis-counting of new fruit appearing close to the position of a fruit that was present in the previous frame but absent in the current frame, with mis-assignment of the new fruit to that fruit.

Several methods have been applied to address the above issues. The first approach is to register fruit based on a 3D point cloud. The 3D point cloud can be developed using combinations of Light Detection and Ranging (LiDAR) sensors, high precision Global Navigation Satellite System (GNSS), stereo imaging, and Structure from Motion (SfM). Other methods such as Kernelized Correlation Filters (KCF) extract features of the target object in the initial frame and then seek the best approximation in the sequent frames via linear or nonlinear regression, so as to track the object [37]. This method is effective in tracking a single distinctive object in a scene. In the proposed application of tree fruit load estimation, each image contains many objects (fruit) with similar features, such that the method is unlikely to work well. A more common approach is to use a state-of-the-art object detection method such as Faster-RCNN or YOLO to detect multiple fruit, and then apply a tracking algorithm such as optical flow [38] or Kalman filter [39]. A correlation may be needed to correlate the same fruit in successive frames, as provided by the Hungarian algorithm.

A number of studies have attempted to acquire 3D information on fruit position. Using a high-performance GNSS to improve the 3D reconstruction accuracy, Moonrinta et al. [40] utilised SfM to generate a point cloud for pineapple orchards to track and count pineapple fruit. Wang et al. [41] employed stereo imaging and high-precision GNSS geolocation to generate a 3D structure of apple canopies and estimate apple yield, with a registration threshold of 5 cm for fruit imaged in sequential views on the same side of the tree, and 16 cm for fruit imaged from the opposite side of the row. The platform travelled at a very low speed of 0.9 km/h.

Stein et al. [30] and Bargoti and Underwood [27] used a platform travelling at approximately 5 km/h to acquire images at 5 fps ('multi-view' imaging) and employed the Hungarian assignment to track fruit. A GNSS-inertial navigation system (GNSS/INS) was used to provide the absolute and relative camera poses for each image, and then the change of camera position (termed 'epipolar') was used to describe the translation of the whole image. The 3D localisation of each fruit achieved using eipipolar projection was then projected onto a LiDAR tree mask to provide assignment of fruit to individual trees.

Gan et al. [42] utilised a system consisting of GNSS, inertial measurement unit (IMU), wheel encoders, and LiDAR to achieve Simultaneous Localization And Mapping (SLAM), with an extended Kalman filter to improve the reliability and accuracy of localization. The system tracks fruit in a real-world coordinate system. Halstead et al. [43] used a simple Intersection over Union (IoU) technique to correlate (track) fruit in frame sequence for capsicum fruit quantity estimation, with the prerequisite that the video was acquired at a high speed (30 fps) and the platform (robot arm) moved at a low speed, such that the fruit had little movement between successive frames.

Use of a 3D localisation system in which each fruit is assigned a universal world coordinate position can ease the tracking issue. However, issues remain—(i) each sensor will carry a measurement

error and the accumulating errors can be significant; (ii) fusion of measurements from different sensors can be an issue, such as the registration of LiDAR depth to RGB images, or the use of IMU information to enhance GPS accuracy and camera position; (iii) the optical flow method assumes that the pixel intensity of the target object remains similar in the frame sequence, and the neighbouring pixels have similar motion, such that the optical flow method is vulnerable to large motions, occlusion, illumination changes, and changes of the appearance of the objects; and (iv) SfM employs various feature extraction methods to reconstruct a 3D point cloud, but difficulties occur in choice of the best match features, the exclusion of outliers, and there may be an inadequate number of feature points if the movement between two frames is large; and (v) the combined set of high precision sensors results in an expensive system.

A lower cost, simplified system for tracking objects across image frames must, however, cope with camera rotation. For the orchard application, the moving platform will introduce some rotation to the camera. Stein et al. [30] described this issue as platform oscillation, and assumed that the translation of the camera could be applied to individual fruit in the image. However, camera rotation will impact the position of fruit across the image differently, with increasing effect further from the centre of the image. We propose that a Kalman filter can be used to predict fruit movement, accommodating camera rotation, and change in scale (due to camera-to-fruit distance changing), as well as the linear translation of position. The rotation and scale can be treated as a process noise, with the Kalman filter used to obtain optimal predictions of individual fruits in future frames based on their trajectories in previous frames. In such a method, fruit at different locations within the image are assigned different horizontal and vertical speeds, rather than a universal global translation. If a stable Kalman filter model is absent for a given fruit (e.g., if it was not detected in previous frames), the fruit can be assigned a model from the neighbouring fruit. The prediction error from the Kalman filter can be then mitigated by a Hungarian assignment algorithm.

In summary, the estimation of fruit load using the dual view approach suffers from sensitivity to the estimation of a correction factor for hidden fruit, related to the magnitude of this factor. Use of a tracking count using video will decrease the proportion of unseen fruit, and can remove the need for accurate spatial location of every tree as required for dual view imaging. In the current study we couple MangoYOLO, as the state-of-the-art detection and localisation algorithm, to the use of the Hungarian algorithm with a Kalman filter to track and count mango fruit in a video of mango trees. The novelty of this approach lies in context of a lower complexity and cost solution contrast to the epipolar projection procedure proposed by Bargoti and Underwood [27] or the high-precision GNSS based registration proposed by Wang, et al. [41].

2. Materials and Methods

2.1. Imaging

Mango orchards of cultivar Calypso^{fm} (latitude -25.14, longitude 152.37) and Honey Gold^{fm} (latitude -23.02, longitude 150.63) were imaged. These are orchards GE1 and HG B, respectively, in the study of [44] and [32] and are described in those papers in terms of tree density and size.

The imaging platform was previously reported in [32,35] and evolved from the systems used by Payne, et al. [29]. Briefly, the automated imaging platform (Figure 1) consisted of a frame mounted to a farm vehicle carrying a 720 W LED flood light and a Basler acA2440-75um machine vision camera (maximum 75 fps, operated at 10 fps, 5 Mp, 2448 \times 2048 pixels) with a Goyo (GM10HR30518MCN) lens (5 mm focal length). The camera was set at a P/# of 1.8 with an exposure of 2.5 ms for the capture of both images and video. The distance from the camera to canopy was typically 2 m. As a one-off task, every tree was geo-located to an accuracy of within 2 cm using a Leica GS-14 unit operating on a Continuously Operating Reference Stations (CORS) network. For tree imaging, the camera was triggered with reference to the pre-acquired GNSS position of trees. The images and video are available as a supplementary data file to this manuscript. The imaging rig operated after sunset at a speed of

about 5 km/hr, such that there were around 30 frames per tree and a shift of approximately 20 pixels per frame. The orchard land was relatively flat, so there was minor vertical movement of the vehicle, and the vertical speed was assumed to be zero.



Figure 1. Automated imaging platform, involving a GNSS device (on pole), a 720 W LED floodlight, and a camera, powered by an inverter-generator and mounted on a farm vehicle ((**a**) buggy or (**b**) utility), operated at 5 km/h.

Images and video frames (10 fps) were scaled to 1024 × 1024 pixels for the use in MangoYOLO. The MangoYOLO architecture and model weights were adopted from Koirala, et al. [32]. The confidence threshold was set at 0.24 and the NMS threshold at 0.45, based on the previous work of Koirala, et al. [32].

The MangoYOLO deep neural network was trained on a CQUniversity High Performance Computing (HPC) facility graphics node with the following specifications—Intel[®] Xeon[®] Gold 6126 (12 cores, 2600 MHz base clock) CPU, NVIDIA[®] Tesla[®] P100 (16 GB Memory, 1328 MHz base clock, 3584 CUDA cores) GPU, Red Hat Enterprise Linux Server 7.4 (Maipo) and 384 GB RAM, CUDA v9.0, cuDNN v7.1.1, OpenCV v3.4.0, Python v2.7.14, and GCC v4.8.5.

The MangoYOLO detector was run on a common laptop (Acer Aspire VN7-593G, New Taipai, Taiwan), with the specification of—Intel[®] Core™ i7-7700HQ CPU @ 2.80 GHz, 32 GB RAM, NVIDIA GeForce GTX 1060 GPU (1506 MHz GPU clock) with 6 GB dedicated memory, 64 bit Windows 10 Pro, CUDA v10.0, cuDNN v7.0.5, and OpenCV v4.0. The algorithm was implemented based on the OpenCV library and programmed in C/C++ language. The processing speed was 3.7 s/frame (407 s for 110 frames).

2.2. Fruit Counting

2.2.1. Dual-View Fruit Counting

For 'dual-view' estimates, the method of Koirala, et al. [32] was adopted. Briefly, two dual-view images (one per tree side) were acquired for each tree. An auto segmentation method was used to delimit trees by exploiting their contours. The Otsu's method [45] was applied to the grayscale images

to remove pixels (set the pixel values to zeros) below the optimum threshold. Then, a sum of the intensity of each column was projected onto the horizontal dimension (see blue line in Figure 2). The points close to zero were considered as the delimiters of two neighboring trees. Thus, trees were segmented if they were separated by a gap or only slightly touched each other; otherwise, images were cropped with a fixed margin. The MangoYOLO model developed by Koirala, et al. [32] with a reported F1 score of 0.968 and Average Precision of 0.983, was then used to detect the fruit in the segmented images, per tree. Finally, individual tree counts were summed to obtain a final yield for the tree row.

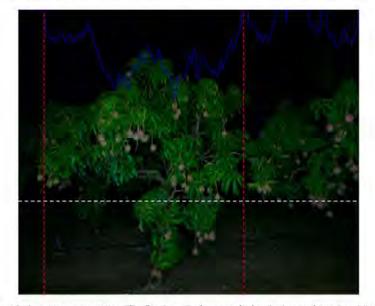


Figure 2. Auto tree segmentation. The Otsu's method was applied to the image above the white line. The sum of intensity of each pixel column x -1 is illustrated as the blue line (zero at image top). Moving away from the image center in each direction, two near zero minima were identified (illustrated with red lines) and used to segment each tree. Canopy segmentation was based on the top two thirds of the image (as denoted by the white horizontal dash line, approximately 1.5 m above ground), as this part of the image carried the sky silhouette of the canopy and the max width of the canopy.

2.2.2. Video Based Fruit Tracking and Counting

MangoYOLO and Kalman Filter.

The MangoYOLO approach was again used to detect fruit location, with the centre point of the output bounding box (x, y) used as the fruit location. The Kalman filter was used to track and predict fruit position for fruit which were not present in a subsequent frame. The variation of vehicle speed and the minor scaling and rotation variations of the image were considered as process noise. A simple model was established involving four measurements—fruit location (x, y) and travel horizontal and vertical speeds $(v_x \text{ and } v_y)$. The transition state matrix *F* and the measurement matrix *H* are:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
(1)

3968

 $\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$ (2)

where *dt* is the time (frame) interval, which was set at 1. The choices of the process and measurement noise covariances are empirical. A small covariance of the process noise leads to accurate estimation, but requires more updates, resulting in a long lag time to build a precise prediction model. To provide for a rapid response, a relatively large covariance of 1.0 was chosen for the process noise. The measurements of fruit locations are fairly precise, and thus a relatively small covariance **R** of 0.1 was set for the measurement noise. The state vector is (x, y, v_x, v_y) , denoted by $\mathbf{x_k}$.

The Kalman filter has two major steps—predict and update. At the prediction stage, a priori state prediction is modelled by:

$$\mathbf{x}_{k|k-1} = \mathbf{F}\mathbf{x}_{k-1|k-1} \tag{3}$$

where $\mathbf{x}_{k-1|k-1}$ is previous location of a fruit. $\mathbf{x}_{k|k-1}$ is the intermediate predicted location without considering the Kalman gain. Meanwhile, a priori predicted error covariance can be calculated as:

$$\mathbf{P}_{\mathbf{k}|\mathbf{k}-1} = \mathbf{F}\mathbf{P}_{\mathbf{k}-1|\mathbf{k}-1}\mathbf{F}^{\mathsf{T}} + \mathbf{Q} \tag{4}$$

where Q is the covariance of the process noise. The Kalman gain can be calculated as:

$$\mathbf{K}_{\mathbf{k}} = \mathbf{P}_{\mathbf{k}|\mathbf{k}-1}\mathbf{H}^{\mathrm{T}} \left(\mathbf{H}\mathbf{P}_{\mathbf{k}|\mathbf{k}-1}\mathbf{H}^{\mathrm{T}} + \mathbf{R}\right)^{-1}$$
(5)

If the fruit is detected in the current frame, an update process is needed. Given the fruit new measured location is z_k , the measurement residual r_k is:

$$\mathbf{r}_{\mathbf{k}} = \mathbf{z}_{\mathbf{k}} - \mathbf{H}\mathbf{x}_{\mathbf{k}|\mathbf{k}-1} \tag{6}$$

Then a posteriori state estimate is updated as:

$$\mathbf{x}_{\mathbf{k}|\mathbf{k}} = \mathbf{x}_{\mathbf{k}|\mathbf{k}-1} + \mathbf{K}_{\mathbf{k}}\mathbf{r}_{\mathbf{k}} \tag{7}$$

and the posteriori error covariance is

$$\mathbf{P}_{\mathbf{k}|\mathbf{k}} = (\mathbf{I} - \mathbf{K}_{\mathbf{k}}\mathbf{H})\mathbf{P}_{\mathbf{k}|\mathbf{k}-1} \tag{8}$$

where I is an Identity matrix.

However, even if a large covariance of process noise is chosen to promote the fast convergence of modelling, the Kalman filter still requires at least four continuous updates (four measurements) to obtain a relatively accurate location prediction model. In practice, some fruit only appear a few (< 4) times in the whole video, and thus the above standard Kalman filter is not applicable to those fruit. In such a case, a speed $\begin{pmatrix} v_x \\ v_y \end{pmatrix}$ can be borrowed from a neighbouring fruit (where that fruit has been updated four or more times and a stable location prediction model has been established). Using the borrowed $\begin{pmatrix} v_x \\ v_y \end{pmatrix}$, an artificial measurement can be provided as:

$$\mathbf{z}_{\mathbf{k}} = \mathbf{H}\mathbf{x}_{\mathbf{k}-1|\mathbf{k}-1} + \begin{pmatrix} v_{\mathbf{k}} \\ v_{\mathbf{y}} \end{pmatrix}$$
(9)

If no "stable" fruit can be found, as occurs with the first four frames, a global speed $\begin{pmatrix} v_x = 20 \\ v_y = 0 \end{pmatrix}$ was used, based on the assumption that the platform travels at a speed of 5 km/h on flat ground.

3969

A process was required to avoid repeat counts for the situation where a fruit was not detected in one or more frames, due to occlusion by leaves or branches or a false negative, but then reappears. Once a stable location prediction model is built, the location of a given fruit can be anticipated in subsequent frames. A fruit reappearing in the expected position of a fruit within 15 frames of its last detection was assumed to be that fruit.

Hungarian Filter.

It is a challenge to match individual fruit between neighbouring frames, when there are many fruit in each video frame. The classic Hungarian method is a combinatorial optimisation algorithm that solves this assignment problem. The Euclidean distance of a fruit in two successive frames is used for the registration of a fruit if it occurs in both. The cost matrix can be then constructed with the element equaling:

$$C_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$
(10)

The Hungarian algorithm aims to find a minimum total cost of assignment, which can be modelled as:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} C_{i,j} x_{i,j}$$
s.t. $\sum_{i=1}^{m} x_{i,i} = 1$, $\sum_{j=1}^{n} x_{i,j} = 1$ and $x_{i,j} \in \{0, 1\}$

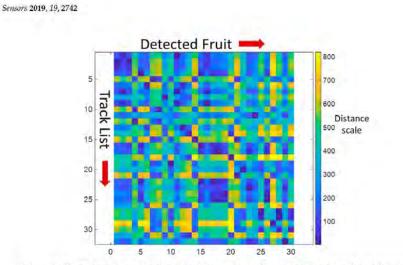
$$(11)$$

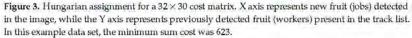
where m and n are the numbers of tracked and new fruit.

For the fruit count application, the algorithm was used to calculate the minimum total cost to correlate fruit between two subsequent frames. Each job (new fruit) was assigned to one worker (tracked fruit) and each worker was assigned one job (known as 'bijection correspondence'). The number of new fruits can be either more or less than the number of (continuing) tracked fruit, although in most case it should be less. Given a travel speed of the imaging platform of 5 km/h, there is a horizontal shift of 20 pixels per frame. Therefore, a maximum distance threshold (redundancy considered) was applied to each assignment to judge if it is a valid assignment. If a fruit in the first frame was associated to a fruit in the second frame, the fruit was considered tracked, and its location in Frame 2 was used to update the Kalman filter. Otherwise, the predicted location from the Kalman filter was used as its new location and an invisible counter related to the fruit was increased by one. Unassigned fruit in Frame 2 were identified as new fruit and a new Kalman filter was created and appointed to the fruit.

In the example provided in Figure 3, there are 32 fruit in the track list and 30 fruit detected in the new image. The colour scale refers to the distance (in pixels) between the centroid of each new fruit and tracked fruit. The red dots signify the assignments of tracked fruit to new fruit, based on a minimum sum cost (Equation (11)). Tracked fruit # 26 and 31 failed to match a new fruit, so a trajectory will be borrowed from a neighbouring fruit to anticipate position in future frames, or these fruit will be removed from the track list if not detected in the past 15 frames (see Results for the rationale for choice of the value 15). As the individual cost for all tracked fruit was less than the threshold value (60, see Results), all assignments were considered to be valid.

If the distance of a fruit in Frame 2 to a fruit in the previous frame was large (>60 pixels), the Frame 1 fruit will fail assignment. In the case shown in Figure 4, fruit A and B are detected in Frame 1 (as A_1 and B_1) and are tracked. In Frame 2, fruit A is not detected, fruit B is detected in position B_2 , and a new fruit C is detected at position C_2 . To obtain the global optimized (lowest) cost, the Hungarian algorithm will assign B_2 to A_1 and C_2 to B_1 . This incorrect assignment results in an under-count of fruit, as noted in Table 1, column iv. Further, if the distance between B_1 and C_2 is greater than the distance threshold, the assignments is rejected. In this example, A_1 was wrongly assigned to B_2 , the prediction models for fruit A and B wrongly updated, and a missed count result.





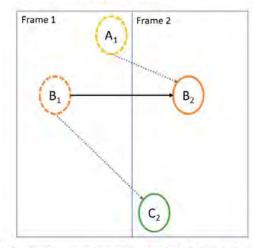


Figure 4. Example of multiple-to-one assignment for three fruit (A, B, C): both A_1 and B_1 are assigned to B_2 , but finally only the assignment of B_1 -to- B_2 is valid as it has a smaller distance.

To solve the issue described above, an improved Hungarian algorithm is proposed, in which:

- (i) the Hungarian algorithm is applied to tracked and new fruit to obtain one-to-one assignments;
- (ii) the maximum distance threshold is applied to decorrelate the assignments with large distances;
- (iii) the Hungarian algorithm is applied a second time to unassigned tracked fruit and new fruit;
- (iv) where two tracked fruit have been assigned the same new fruit (a 'multiple-to-one assignment'), only the assignment with smaller cost (distance) is retained.

As in the case of Figure 4, the improved Hungarian algorithm can fail to count a new fruit in the situation that the new fruit appears very close to a tracked fruit that is not detected in the current frame, (i.e., smaller than 60 pixels). However, such cases were found to be rare.

10 of 17

The fruit correlation between frames can be lost due to an incorrect Hungarian assignment, temporary occlusion, or failure of detection. The Kalman filter was used to mitigate these issues by predicting the fruit location across subsequent frames. However, the Kalman prediction will lose accuracy in the prediction of fruit position as the frame number increases. Therefore, the maximum number of continuous unobserved frames that the fruit was carried on the track list was empirically set to 15.

The workflow of the proposed method is illustrated in Figure 5. For an input video frame, MangoYOLO was used to detect fruit, then the adapted Hungarian algorithm was used to correlate to fruit detected in the previous frame. If a fruit was in the track list but not detected in the current frame, and if its continuous unobserved frames was less than a specified value 'a' (where a = 15, see Results), the location predicted by the improved Kalman filter was registered as the new location of the fruit. If the count was >a, the fruit was removed from the track list. If a fruit detected in the current frame was not in the track list, it was considered to be a new fruit, a new Kalman filter was created and the fruit was added to the track list.

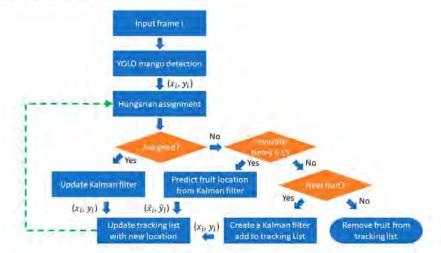


Figure 5. Workflow of the proposed method.

A tracking example is illustrated in Figure 6. Each fruit was assigned a unique tracking number by its order of appearance. If a correlation (assignment) was found for a tracked fruit in the current frame, the fruit bounding box was coloured red. If there was no assignment for a tracked fruit in the current frame, the projected position of the missing fruit was shown with a pink bounding box. New fruit detections were shown in blue bounding boxes. Most new fruit occurred on the left side of the image, which is the direction of camera travel. However, fruit that were heavily occluded by foliage or other fruit could become visible from a specific angle, appeared in the centre or right of the image, i.e., fruit 103. If a fruit passed out of view for more than 15 frames or its predicted location become inaccurate relative to its actual location within 15 frames, it would be treated as a new fruit (i.e., double counted), as occurred for fruit 86/99.

To reduce the number of fruit seen twice in imaging of the tree from both inter rows, a threshold was placed on fruit size, as fruit on the far side of the canopy in a given image appear smaller. Specifically, bounding boxes with width <12 pixels or height <15 pixels were excluded from detection.

Sensors 2019, 19, 2742

11 of 17



Figure 6. A tracking example: red bounding boxes indicate tracked fruit with valid assignments; pink indicates predicted location for fruit on the track list that were not detected in the current frame; blue indicates a new fruit detection. Numbering on fruit is in order of detection. Red arrows point to fruit mentioned in text.

2.2.3. Human Count

For reference values, fruit in the images were counted by a human, and the fruit number on trees was counted following harvest. For videos, the tracking results were examined frame by frame, and the count difference between the estimates of a target tree from its neighbouring trees were recorded as the fruit count of the tree. Errors in the association of fruit to individual trees will impact individual tree count but not total row count.

3. Results and Discussion

3.1. MangoYOLO Performance

The MangoYOLO model was trained on images from a mango orchard of the Calypso variety, collected in 2017 [32]. When tested on images from the same orchard during the subsequent season (2018), a $R^2 = 0.988$, RMSE = 5.0, and bias = -4.0 fruit/image was achieved, relative to the human count (Figure 7). For the MangoYOLO model used in the assessment of the 110 video frames of cultivar HoneyGold, $R^2 = 0.665$, RMSE = 2.1 and bias = 0.0 fruit/image (Figure 7). Thus the model performed well, irrespective of cultivar. The lower R^2 for the video result was a result of the lower variation in fruit load number in these images, compared to that in the Calypso images. For the HoneyGold video, there was an average of 31 fruit detected per frame, with 1.4 false positives and 1.4 false negatives. The false positive errors of MangoYOLO resulted in an over-count while the false negative errors resulted in an under-count. The numbers of false positives and false negatives were similar, resulting in a low bias.

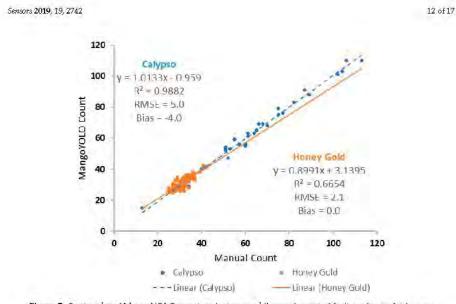


Figure 7. Scatter plot of MangoYOLO count against manual (human) count of fruit per image for images of Calypsotm trees in 2018 (blue symbols), and video of HoneyGoldtm trees in 2018 (orange symbols).

3.2. Choice of Maximum Unobserved Times and Threshold for Hungarian Assignment

A range of values from 0 to 50 was considered for the number of frames a fruit was not detected in before the fruit was dropped from the tracked list ('maximum unobserved times'), in the context of the impact on prediction accuracy (Figure 8). At a setting of 0, the Kalman filter was not used and only the Hungarian algorithm was used to correlate fruit. In this case, once a fruit loses its assignment in the subsequent frame, it was counted as a new fruit if detected in subsequent frames. This setting resulted in a high count (n = 357), compared to the human count of 192. Count numbers stabilised at a threshold of 15 unobserved frame times, with a MV count (n = 191) close to the human count (n = 192), and this value was adopted in subsequent work.

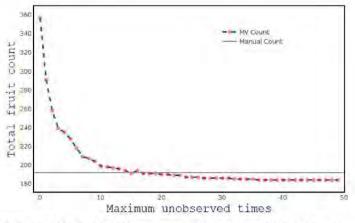


Figure 8. Impact of the value of maximum unobserved times on MangoYOLO fruit count in a video frame. The horizontal line denotes the human count of fruit in frame.

The distance threshold used in assessing the validity of a Hungarian assignment was optimised by consideration of a range of values from 20 to 100 pixels, with a step of five (Figure 9). In principle, a smaller value should result in an increased repeat count, whereas a larger value should result in an increased incorrect assignment of new fruit to tracked fruit, leading to an underestimation of count. For example, if the platform speed was higher than the assumed 20 pixels per frame, at a setting of 20 pixels re-appearing fruit would be considered as new fruit, leading to over-counting (count of 956 fruit). The estimation was close to human count at a setting of 60 pixels (Figure 9).

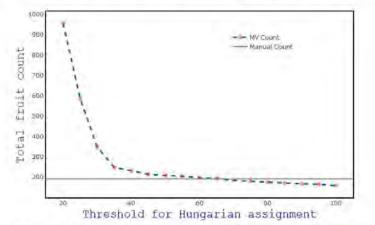


Figure 9. Impact of the value of threshold for Hungarian assignment on fruit count in the video. The horizontal line denotes the human count of fruit in frame.

3.3. Frame by Frame Comparison Between Human and Proposed Method

A frame-by-frame assessment of the fruit tracking and counting results of the MangoYOLO-Kalman filer-Hungarian algorithm method was undertaken by a human, for a video of 110 frames. Several kinds of errors were noted (Table 1)—(i) repeat count associated with temporary occlusion within a 15 frame interval, in which the predicted position is inaccurate; (ii) false prediction due to an inaccurate prediction model or the platform suffering an abrupt movement; or (iii) missing of a previous detection. These errors result in a fruit count overestimate. Another case can result in an underestimate—(iv) when a new fruit appears close to a tracked fruit and the tracked fruit does not exist in the current frame, the new fruit is assigned to the tracked fruit and a count is missed. Over the 110 frames assessed, the repeat fruit count was 9.7%, while the miss count was 7.1%, resulting in an over-estimate of 2.6% compared to the human assessment. Overall, the proposed method resulted in an estimate of "non-hidden" fruit that was 102.6% of the count of the harvest tally.

Table 1. Error attributions based on human assessment for fruit detection in 110 video frames using the tracking (MangoYOLO-Kalman filter-Hungarian algorithm) method.

Hionan Count	(i) Repeat Due to Occlusion in Previous Frames	(ii) Repeat Due to FN in Previous Frames	(iii) False Prediction of Position	Total Repeat Count	(iv) Missed Count Due to New Fruit Assigned to Old Fruit Position	Estimated Count
192	3	1.	15	19	-14	197
100%	1.5%	0.5%	7.8%	9.9%	-7.3%	102.6%

3.4. Fruit Count From Video

A comparison was undertaken of the harvest count to the dual-view image and MangoYOLO—Kalman filter—Hungarian algorithm method estimate, for a continuous row of 21 trees (Table 2). The dual view method achieved a count equivalent to 40% of the harvest count with bias corrected RMSE = 21.7 fruit/tree, while the proposed method achieved a count equivalent to 62% of the harvest count with bias corrected RMSE = 18.0 fruit/tree.

Table 2. Fruit count of each of 21 trees based on the harvest fruit tally, dual view imaging count (as per [32]), and tracking (MangoYOLO-Kalman filter-Hungarian algorithm) method count. Correction factors refers to the ratio of MV count to harvest tally.

	Harvest	Dual-View Imaging	Tracking
Total (#fruit/21 trees)	3286	1322	2050
Average (#fruit/tree)	156.5	63,0	97.6
Bias (#fruit/tree)	14	-93.5	-58.9
% MV/harvest	-	40.2%	62,3%
RMSE-bc	-	21.7	18.0
Correction factor	- R	2.5	1.6

3.5. Orchard Application

Any imaging system based on a 'drive by' imaging platform will suffer a failure to 'see' a proportion of fruit on the tree. As expected, the video based MangoYOLO—Kalman filter—Hungarian algorithm tracking method improved the detection of fruit per tree, by adding additional imaging perspectives. The ratio of total (harvest) count to machine vision count was decreased from 2.5 using the dual view method to 1.6 using the video based method.

Thus the MV based estimation of mango fruit load, and ultimately automated harvest, is suited to orchards in which canopies are managed to have all fruit visible on the external wall of the canopy. This involves a move to canopies akin to the high density apple production systems, a trend that is already underway in the mango industry, e.g., [46]. The orchard imaged in the current study was a 'difficult' case, with a high proportion of 'hidden' fruit due a relatively thick, dense canopy, pruned to a continuous hedge. Future work should evaluate the technique in context of narrower canopy architectures, where hidden fruit becomes less of an issue and double counting of fruit from imaging from the two sides of the canopy becomes a greater issue. Another solution involves the generation of a correction factor for hidden fruit per free, or indeed, per image based on characters within the image (e.g., foliage overlap).

4. Conclusions

The work of this study supports an automated imaging system for the estimation of mango orchard fruit load, extending a progression of studies [28,29]. The use of MangoYOLO [32] for mango detection and counting is validated in the current study. Video based Kalman tracking was demonstrated to provide an improved estimate of total fruit load over the dual view imaging procedure. Video based fruit yield estimation reduces the number of hidden fruit by providing more viewpoints of the canopy.

Video based methods for fruit tracking and counting have been proposed by several contemporary research groups. These imaging systems are complex and costly, due to the use of GNSS/IMU or LiDAR to track camera movement or obtain precise real-word coordinates of individual fruit. Further, registration errors can impact the calculation of real-world coordinates and worsen the tracking of fruit. The practical adoption of an in-field machine vision solution to fruit load estimation requires a compromise between cost, ease of use, and accuracy/precision.

The multi-view method presented in this study employed only LED lighting and a camera, and does not require differential GNSS if orchard rather than per tree estimates are required. The proposed method thus has lower hardware complexity and cost (e.g., the hardware cost of the current imaging

rig of LED panel, camera and computer was <US\$ 4000, while a LiDAR and dGNSS system can easily add >US\$ 30,000). The system acquired 10 frames per second while that of Stein, et al. [30] employed 5 fps at a similar vehicle speed, providing more view-points per tree. The video based system does not allow for localisation within the orchard or for segmentation of trees, but could be used in the production of orchard maps given the use of GNSS or row locators, e.g., barcodes. The proposed method could be applied to other crops, such as apples and citrus.

Supplementary Materials: The following are available online at http://www.mdpi.com/1424-8220/19/12/2742/s1, Video S1: Fruit Tracking Count.

Author Contributions: Conceptualization, Z.W.; Data curation, Z.W. and A.K.; Funding acquisition, K.W.; Investigation, K.W.; Methodology, Z.W. and A.K.; Project administration, K.W.; Software, Z.W.; Supervision, K.W.; Validation, Z.W.; Visualization, Z.W.; Writing—original draft, Z.W. and K.W.; Writing—review & editing, A.K.

Funding: This work received funding support from the Australian Federal Department of Agriculture and Water, and from Horticulture Innovation (project ST15005, Multiscale Monitoring of Tropical Fruit Production). AK acknowledges receipt of an Australian Regional Universities Network scholarship, and ZW was funded by a CQU early career fellowship.

Acknowledgments: We thank Groves Grown orchards for orchard access and moral support.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Walsh, K.; Wang, Z. Monitoring fruit quality and quantity in mangoes. In Achieving Sustainable Cultivation of Mangoes; Galán Saúco, V., Lu, P., Eds.; Burleigh Dodds Science Publishing Limited: Cambridge, UK, 2018; pp. 313–338.
- 2. Monselise, S.P.; Goldschmidt, E. Alternate bearing in fruit trees. Hortic. Rev. 1982, 4, 128–173.
- Anderson, N.T.; Underwood, J.P.; Rahman, M.M.; Robson, A.; Walsh, K.B. Estimation of fruit load in mango orchards: Tree sampling considerations and use of machine vision and satellite imagery. *Precis. Agric.* 2018. [CrossRef]
- Zaman, Q.; Schumann, A.; Percival, D.; Gordon, R. Estimation of Wild Blueberry Fruit Yield Using Digital Color Photography. *Trans. Asabe* 2008, 51, 1539–1544. [CrossRef]
- Zhou, R.; Damerow, L.; Sun, Y.; Blanke, M.M. Using colour features of cv. 'Gala' apple fruits in an orchard in image processing to predict yield. *Precis. Agric.* 2012, 13, 568–580. [CrossRef]
- Annamalai, P.; Lee, W.S. Citrus Yield Mapping System Using Machine Vision. In Proceedings of the Annual International Conference of The American Society of Agricultural Engineers, Las Vegas, NV, USA, 27–30 July 2003.
- Canny, J. A Computational Approach to Edge Detection. In Readings in Computer Vision; Fischler, M.A., Firschein, O., Eds.; Morgan Kaufmann: San Francisco, CA, USA, 1987; pp. 184–203. [CrossRef]
- Jianbo, S.; Tomasi, C. Good features to track. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 21–23 June 1994; pp. 593–600.
- Lindeberg, T. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. Int. J. Comput. Vis. 1993, 11, 283–318. [CrossRef]
- Damon, J. Properties of ridges and cores for two-dimensional images. J. Math. Imaging Vis. 1999, 10, 163–174. [CrossRef]
- Lowe, D.G. Object recognition from local scale-invariant features. In Proceedings of the 7th IEEE International Conference on Computer Vision, Kerkyra, Greece, 20-27 September 1999; pp. 1150–1157.
- Bay, H.; Tuytelaars, T.; Van Gool, L. Surf: Speeded up robust features. In European Conference on Computer Vision; Springer: Berlin/Heidelberg, Germany, 2006; pp. 404–417.
- Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
- Ahonen, T.; Hadid, A.; Pietikäinen, M. Face recognition with local binary patterns. In European Conference on Computer Vision; Springer: Berlin/Heidelberg, Germany, 2004; pp. 469–481.

- Kurtulmus, E; Lee, W.S.; Vardar, A. Green citrus detection using 'eigenfruit', color and circular Gabor texture features under natural outdoor conditions. *Comput. Electron. Agric.* 2011, 78, 140–149. [CrossRef]
- Linker, R.; Cohen, O.; Naor, A. Determination of the number of green apples in RGB images recorded in orchards. Comput. Electron. Agric. 2012, 81, 45–57. [CrossRef]
- Qureshi, W.S.; Payne, A.; Walsh, K.B.; Linker, R.; Cohen, O.; Dailey, M.N. Machine vision for counting fruit on mango tree canopies. *Precis. Agric.* 2017, 18, 224–244. [CrossRef]
- Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
- He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
- Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
- Zeiler, M.D.; Fergus, R. Visualizing and understanding convolutional networks. In European Conference on Computer Vision; Springer: Cham, Switzerland, 2014; pp. 818–833.
- Koirala, A.; Walsh, K.B.; Wang, Z.; McCarthy, C. Deep learning—Method overview and review of use for fruit detection and yield estimation. *Comput. Electron. Agric.* 2019, 162, 219–234. [CrossRef]
- Sa, I.; Ge, Z.; Dayoub, F.; Upcroft, B.; Perez, T.; McCool, C. DeepFruits: A Fruit Detection System Using Deep Neural Networks. Sensors 2016, 16, 1222. [CrossRef] [PubMed]
- Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In Advances in Neural Information Processing Systems; The MIT Press: Cambridge, MA, USA, 2016; pp. 91–99.
- Chen, S.W.; Shivakumar, S.S.; Dcunha, S.; Das, J.; Okon, E.; Qu, C.; Taylor, C.J.; Kumar, V. Counting Apples and Oranges with Deep Learning: A Data-Driven Approach. *IEEE Robot. Autom. Lett.* 2017, 2, 781–788. [CrossRef]
- Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
- Bargoti, S.; Underwood, J. Deep fruit detection in orchards. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 3626–3633.
- Payne, A.B.; Walsh, K.B.; Subedi, P.P.; Jarvis, D. Estimation of mango crop yield using image analysis-Segmentation method. *Comput. Electron. Agric.* 2013, 91, 57–64. [CrossRef]
- Payne, A.; Walsh, K.; Subedi, P.; Jarvis, D. Estimating mango crop yield using image analysis using fruit at 'stone hardening' stage and night time imaging. *Comput. Electron. Agric.* 2014, 100, 160–167. [CrossRef]
- Stein, M.; Bargoti, S.; Underwood, J. Image based mango fruit detection, localisation and yield estimation using multiple view geometry. Sensors 2016, 16, 1915. [CrossRef] [PubMed]
- Robson, A.; Rahman, M.; Muir, J. Using worldview satellite imagery to map yield in avocado (*Persea americana*): A case study in Bundaberg, Australia. *Remote Sens.* 2017, 9, 1223. [CrossRef]
- Koirala, A.; Walsh, K.B.; Wang, Z.; McCarthy, C. Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of 'MangoYOLO'. Precis. Agric. 2019, 1–29. [CrossRef]
- Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27-30 June 2016; pp. 779–788.
- Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21-26 July 2017; pp. 7263–7271.
- Wang, Z.; Underwood, J.; Walsh, K.B. Machine vision assessment of mango orchard flowering. Comput. Electron. Agric. 2018, 151, 501–511. [CrossRef]
- Liu, X.; Chen, S.W.; Liu, C.; Shivakumar, S.S.; Das, J.; Taylor, C.J.; Underwood, J.; Kumar, V. Monocular Camera Based Fruit Counting and Mapping with Semantic Data Association. *IEEE Robot. Autom. Lett.* 2019, 4, 2296–2303. [CrossRef]
- Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-Speed Tracking with Kemelized Correlation Filters. IEEE Trans. Pattern Anal. Mach. Intell. 2015, 37, 583–596. [CrossRef]

17 of 17

- 38. Horn, B.K.P.; Schunck, B.G. Determining optical flow. Artif. Intell. 1981, 17, 185-203. [CrossRef]
- Zarchan, P.; Musoff, H. Fundamentals of Kalman Filtering: A Practical Approach; American Institute of Aeronautics and Astronautics, Inc.: Reston, VA, USA, 2013.
- Moonrinta, J.; Chaivivatrakul, S.; Dailey, M.N.; Ekpanyapong, M. Fruit detection, tracking, and 3D reconstruction for crop mapping and yield estimation. In Proceedings of the 2010 11th International Conference on Control Automation Robotics & Vision, Singapore, 7–10 December 2010; pp. 1181–1186.
- Wang, Q.; Nuske, S.; Bergerman, M.; Singh, S. Automated Crop Yield Estimation for Apple Orchards. In Experimental Robotics, Proceedings of the 13th International Symposium on Experimental Robotics, Quebec City, QC, Canada, 18–21 June 2012; Desai, P.J., Dudek, G., Khatib, O., Kumar, V., Eds.; Springer International Publishing: Heidelberg, Germany, 2013; pp. 745–758. [CrossRef]
- Gan, H.; Lee, W.S.; Alchanatis, V. A Prototype of an Immature Citrus Fruit Yield Mapping System. In Proceedings of the 2017 ASABE Annual International Meeting, Spokane, WA, USA, 16–19 July 2017; pp. 1–6.
- Halstead, M.; McCool, C.; Denman, S.; Perez, T.; Fookes, C. Fruit Quantity and Ripeness Estimation Using a Robotic Vision System. *IEEE Robot. Autom. Lett.* 2018, 3, 2995–3002. [CrossRef]
- Underwood, J.; Rahman, M.; Robson, A.; Walsh, K.; Koirala, A.; Wang, Z. Fruit load estimation in mango orchards—A method comparison. In Proceedings of the ICRA 2018 Workshop on Robotic Vision and Action in Agriculture, Brisbane, Australia, 21–25 May 2018; pp. 1–6.
- 45. Otsu, N. A threshold selection method from gray-level histograms. Automatica 1975, 11, 23–27. [CrossRef]
- Menzel, C.M.; Le Lagadec, M.D. Can the productivity of mango orchards be increased by using high-density plantings? Sci. Hortic. 2017, 219, 222–263. [CrossRef]

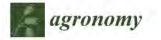


© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).

3981 Appendix B. Published version of Chapter 4

3982 Deep Learning for Mango (*Magnifera indica*) Panicle Stage Classification

- 3983 Koirala, A.; Walsh, K.B.; Wang, Z.; Anderson, N. Deep Learning for Mango (*Mangifera indica*) Panicle
- 3984 Stage Classification. *Agronomy* **2020**, *10*, 143. Doi: <u>https://doi.org/10.3390/agronomy10010143</u>



Article



Deep Learning for Mango (*Mangifera indica*) Panicle Stage Classification

Anand Koirala *10, Kerry B. Walsh , Zhenglin Wang and Nicholas Anderson

Institute for Future Farming Systems, Central Queensland University, Building 361, Bruce Highway,

Rockhampton, QLD 4701, Australia; k.walsh@cqu.edu.au (K.B.W.); z.wang@cqu.edu.au (Z.W.);

nicholas.anderson@cqumail.com (N.A.)

* Correspondence: anand.koirala@cqumail.com; Tel.: +61-411096239

Received: 11 December 2019; Accepted: 15 January 2020; Published: 18 January 2020



Abstract: Automated assessment of the number of panicles by developmental stage can provide information on the time spread of flowering and thus inform farm management. A pixel-based segmentation method for the estimation of flowering level from tree images was confounded by the developmental stage. Therefore, the use of a single and a two-stage deep learning framework (YOLO and R^2 CNN) was considered, using either upright or rotated bounding boxes. For a validation image set and for a total panicle count, the models MangoYOLO(-upright), MangoYOLO-rotated, YOLOv3-rotated, R^2 CNN(-rotated) and R^2 CNN-upright achieved weighted F1 scores of 76.5, 76.1, 74.9, 74.0 and 82.0, respectively. For a test set of the images of another cultivar and using a different camera, the R^2 for machine vision to human count of panicles per tree was 0.86, 0.80, 0.83, 0.81 and 0.76 for the same models, respectively. Thus, there was no consistent benefit from the use of rotated over the use of upright bounding boxes. The YOLOv3-rotated model was superior in terms of total panicle count, and the R^2 CNN-upright model was more accurate for panicle stage classification. To demonstrate practical application, panicle counts were made weekly for an orchard of 994 trees, with a peak detection routine applied to document multiple flowering events.

Keywords: bounding box; deep learning; Mangifera indica; panicle classification; rotation; segmentation

1. Introduction

Mango (*Mangifera indica*) trees produce panicles bearing hundreds of inconspicuous flowers, of which at most three or four flowers will develop fruit, although frequently only one or none will so develop. The assessment of the number of panicles on a tree thus sets a maximum potential for the crop yield of that season, while the assessment of the stage of panicle development is useful for assessment of the time spread of flowering, and thus the likely time spread of the harvest period. Mapping areas of early flowering can also guide selective early harvesting and panicle detection can inform selective spraying operations. However, the manual assessment and recording of panicle number and stage is a tedious task that relies on the experience of the observer. An automated approach for detection and classification of flowering stage could thus aid orchard management.

Machine vision has been applied for the assessment of the level of flowering for several tree crops where the flowers are easily distinguishable from the background based on colour thresholding. For example, [1] reported a prediction accuracy of 82% on apple flower count, relative to a manual count, [2] achieved a coefficient of determination (R^2) of 0.94 between machine vision and manual count of tangerine flowers, [3] claimed an average detection rate of 84.3% on peach flowers, [4] reported a R^2 of 0.59 between machine vision and manual count of apple flower cluster counts, [5] obtained a F1-score of 73% for tomato flower detection, and [6] avoided a direct count of flowers, pervise a machine vision and ratio of flower and canopy pixels, reporting a

Agronomy 2020, 10, 143; doi:10.3390/agronomy10010143

www.mdpi.com/journal/agronomy

poor relationship ($R^2 = 0.23$) for this index for a given tree between two seasons. However, all these reports were based on segmentation routines, generally involving colour given the obvious colour difference for flowers of these species and background. This approach is expected to perform poorly for tree crops for which the colour difference between vegetative and reproductive tissue is not marked.

Furthermore, apple, peach and tomato flowers are relatively conspicuous. A mango tree produces panicles with pale cream flowers a few mm in size. The panicle size changes with developmental stage, increasing through the stages of bud break, 'asparagus', elongation, anthesis (flower opening) to the full bloom 'Christmas tree' stage, then decreasing with flower drop. Panicle structure is thus more complex than that of a single flower. Therefore, machine vision detection of mango panicles is more challenging compared to the detection of the single flowers of apple, citrus and almond trees.

There are few reports on use of machine vision for the assessment of the number of flowering panicles involving a branched inflorescence with many inconspicuous flowers. For example, [7] used intensity level in CIE LAB colour space in assessment of the number of grape flowers in inflorescences imaged against a black background. A R^2 of 0.84 against human count was achieved. In [8], researchers used Scale-invariant Feature Transform (SIFT) descriptors/features along with a Support Vector Machine (SVM) to detect rice flower panicles in images, with a R^2 of 0.67 against a human count achieved.

Recent review papers have emphasised the use of neural network and deep learning in agricultural machine vision in general [9] and for fruit detection and yield estimation [10]. For example, [11] noted that better performance in fruit detection and localization was achieved through the use of neural networks compared to traditional models based on colour thresholding and hand-engineered feature extraction methods. The use of lighter weight, single shot detectors which allow faster computation times is introduced in [12]. Our group designed a modified You Only Look Once (YOLO, [13]) architecture, 'MangoYOLO' [12] which was demonstrated to be superior to YOLOv3 [14] for the application of fruit detection and count. In [15], researchers used Clarifai [16] Convolutional Neural Network (CNN) architecture to extract features from the possible flower regions obtained from super-pixel segmentation followed by SVM [17] for flower detection. For apple, peach and pear datasets, this method outperformed other methods of that time that were based on SVMs and Hue-Saturation-Value (HSV) colour thresholding methods. Extending their earlier work, [18] used a Fully Convolutional Neural Network (FCN) from [19] for flower detection on tree images of apple, peach and pear. A region growing refinement (RGR) algorithm was implemented to refine the segmentation output from FCN. This method achieved F1 scores of 83.3%, 77.3%, 74.2% and 86% on two apple, peach and pear flower datasets respectively, outperforming their previous Clarifai CNN method [15] and the HSV colour model. Deep learning methods of object detection may suit the task of detection and counting of mango panicles by developmental stage, through automatic learning of useful features for classification [10].

Our group has previously considered the use of machine vision to assess mango flowering. In [20,21], researchers used the traditional method of pixelwise segmentation to segment panicle pixels from canopy pixels, with results expressed either as panicle pixel count per free or as the ratio of panicle to canopy pixel count (termed 'flowering intensity'). This procedure was implemented on images obtained at night using artificial lighting, processed with a colour threshold followed by SVM classification to refine the segmentation results. The use of a Faster *R*-CNN [22] deep learning technique to count panicles was reported by [21]. This work was limited to estimation of the extent of flowering and the time of peak flowering event. A R^2 of 0.69 between machine-vision flowering intensity and in-field human count of panicles per tree for 24 trees was reported for the segmentation method, while a R^2 of 0.78 and 0.84 was reported for deep learning Faster *R*-CNN framework with dual and multi-view imaging approaches, respectively.

These earlier reports employed upright bounding boxes. The use of an annotation bounding box as tight as possible around the objects has been recommended to avoid background noise in the training image sets [10]. However, panicles are oriented in some range of angles. Upright bounding boxes will therefore not fit tightly around the object perimeter (Figure 1), and a larger amount of

background signal will be included in the object class for training. This could adversely affect the classification accuracy of the model.

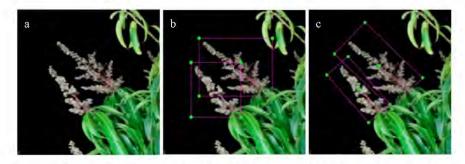


Figure 1. (a) Original image, (b) upright bounding box and (c) rotated bounding box.

There are several methods that involve use of rotated bounding boxes. The two-stage detector R^2 CNN (Rotational Region CNN for Orientation Robust Scene Text Detection) [23] is a modification of the Faster *R*-CNN object detection framework used to incorporate training on rotated bounding box annotations for detecting arbitrarily-oriented objects in images. This modified framework seems suited to the task of panicle detection. Alternatively, the single stage detector YOLO allows a number of data augmentation measures, including the random rotation of the labelled bounding boxes during training by a range (degrees) specified by the user in the configuration file.

In the current paper, the task of panicle detection task and count is extended to another level through classification to developmental stage, with comparison of a large (R^2 CNN framework with ResNet101 CNN) and a small (YOLO framework with Mango YOLO CNN) object detection architecture, and consideration of the use of rotated, compared to upright, bounding boxes. To the authors' knowledge, the current study is the first to classify the stage of flowering for an on-tree fruit crop and is the first report on use of the rotated bounding box method of R²CNN for flower panicle detection. The current study utilized the imaging hardware used by [21], allowing for a direct comparison of results of the traditional machine learning approach used by [21]. Field relevance was demonstrated by assessment of orchard flowering at regular intervals (e.g., weekly) to provide information on timing of flowering peaks, for use in estimation of harvest timing. 'The major contributions of this study are thus (i) the comparison of single and double-staged deep learning object detection frameworks for flowering panicle detection and development stage classification; (ii) the comparison of deep learning methods with traditional segmentation method; (iii) the comparison of model performance with respect to the use of rotated and upright bounding box annotations; (iv) the application of a peak-detection routine to assist in the identification of a flowering event in a time series; (v) an approach to assess/visualize flowering data to inform farm management.

2. Materials and Methods

2.1. Image Acquisition

Tree images of orchard A (Table 1) were acquired at night every week from 16 August to 18 October 2018, using a 5 MP Basler acA2440-75 μ m RGB camera and a lighting rig (700 W LED floodlight) mounted on a farm vehicle driven at a speed of about 5 km/h, with triggering of the camera at every tree position from a GNSS device, and the display of data on a web-app farm map as described by [21]. The orchard contained 994 trees, and thus each weekly imaging event captured 1,988 images. This time window covered the flowering period of this orchard, from panicle formation to fruit set.

The image set of 24 trees from orchard B (Table 1; from [21]) were used as a test set. In that study, images were acquired using a 24 MP Canon (DSLR 750D) camera. The number of open, flowering

panicles on these trees was manually counted. For both orchards, trees were imaged from each side, with a view from each inter-row ('dual-view' imaging).

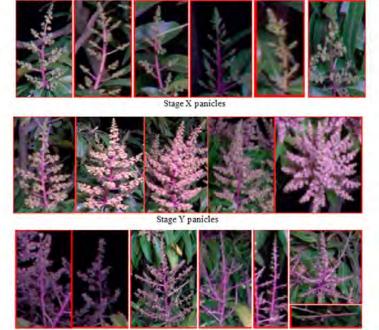
Table 1. Orchard and	imaging description.
----------------------	----------------------

Orchard Name	Location (lat., long.)	Cultivar	Camera	Image Resolution		
Orchard A	-23.032749, 150.620470	Honey Gold	Basler	2464 × 2048 pixels		
Orchard B	-25.144, 152.377	Calypso	Canon	6000 × 4000 pixels		

2.2. Data Preparation

2.2.1. Image Annotation and Labelling

Mango panicles in the training image sets were categorized into three stages; (i) stage X—panicles with flowers (whitish in colour) that were not fully opened, i.e., flower opening to inflorescence branching (Figure 2, top row); (ii) stage Y—panicles with open flowers, colloquially known as 'Christmas tree' stage (Figure 2, middle row); (iii) stage Z—panicles displaying flower drop and fruit set (Figure 2, bottom row). A four-category system was initially trialled, but human differentiation of the two early stages was problematic, and the resulting model performance was poorer than for the three-category system (data not shown).



Stage Z panicles

Figure 2. Examples for stages X to Z, by rows.

Annotation was done using roLabelImg (https://github.com/cgvict/roLabelImg) which is a modification of LabelImg (a graphical image annotation tool) to incorporate rotated boxes. Images were opened in roLabelImg software and a rectangular bounding box was drawn and rotated by dragging to fit nicely around each panicle. Each box was labelled with their respective class name (X or Y or Z). For each

image, annotations were saved as XML files which contained the label (class name or category) and the position (height, width and centre co-ordinates of the box) of each annotated box along with the orientation angle.

As the annotations prepared for training R^2 CNN contained rotated bounding boxes which cannot be used directly with the YOLO model, a separate python script was written to convert the rotated bounding box annotations to upright bounding box annotations for training of the YOLO model. The *boundingRect* function from open source computer vision library (OpenCV) [24] was utilized to obtain an upright bounding box from the vertices of a rotated bounding box as contour input to the function. A visualization of the rotated ground truth boxes and the transformed upright bounding box annotation is presented in Figure 3. Separate Python scripts were written to convert the XML annotation format of roLabelImg to that of R^2 CNN and YOLO formats for model training.

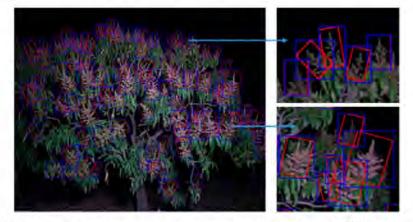


Figure 3. Visualization of ground truth bounding box original (rotated) red colour for *R*²CNN training and transformed (upright) blue colour for MangoYOLO training.

2.2.2. Training, Validation and Test Sets

Training was based on 54 images (3178 labelled panicles) of trees from orchard A (Table 2), with images drawn from different weeks. A validation set was assembled from images of one side of a single orchard A tree, acquired over a six-week period (Table 2). These images were not included in the training set.

Table 2. Of panicles in training, validation and test image sets. The test set consisted of manual panicle counts per tree from [21] and Canon camera images of those trees (no panicle stage categorization was available).

	Number of Lands	Nu	Number of Panicles						
	Number of Images	Stage X	Stage Y	Stage Z	Total				
Training set	54	1007	1107	1064	3178				
Validation set	6	167	316	122	635				
Test set	48			-	2853				

The training dataset also included annotations for background (369 snips), as required for R^2 CNN training. R^2 CNN uses the background class for negative hard-data mining and was not treated as a detection class during inference. However, the YOLO object detection framework does not require a background class, as all parts of the images other than those having bounding box for training are automatically treated as background. Therefore, the background class was not used for YOLO model performance evaluation.

The test set (Table 2) was an independent set consisting of images of orchard B, from the study of [21]. These images were of trees of a different cultivar and from a different orchard, and acquired with a different camera, to that of the training and validation sets.

2.2.3. Computing

Model training and testing was implemented on the CQUniversity High Performance Computing (HPC) facility graphics node with following specifications: Intel[®] (Santa Clara, California, U.S.A.) Xeon[®] Gold 6126 (12 cores, 2600MHz base clock) CPU, NVIDIA[®] (Santa Clara, California, U.S.A.) Tesla[®] (San Carlos, California, U.S.A) P100 (16 GB Memory, 1328 MHz base clock, 3584 CUDA cores) GPU. Red Hat (Raleigh, North Carolina, U.S.A.) Enterprise Linux Server 7.4 (Maipo) and 384GB RAM. CUDA v9.0, cuDNN v7.1.1, OpenCV v3.4.0, Python v2.7.14, GCC v4.8.5, scikit-learn v0.19.1, tensorflow-gpu v1.6.0, Keras v2.1.5, Cython v0.28.

Images (of two sides) of each tree were fed to the model for detection and classification. For each image of a tree, the results (number of each stage panicles) were saved as a text file and later uploaded along with the GNSS position to the 'fruitMaps' website for display on a farm map as described in [21].

2.3. Detection and Classification Models

The pixel segmentation method used in [21] was employed using the settings of [20]. Mango panicle classification at three developmental stages was attempted using the following architectures:

YOLOv3-rotated: The single shot object detection framework YOLOv3 [14] is deeper but more accurate than its previous versions [13,25]. The configuration file data augmentation feature to allow rotation of images during training was enabled for this training exercise. Images were rotated randomly in the range of 40 degrees to mimic the natural range of panicle orientations in tree images. Network input resolution was set to 1024×1024 pixels following the YOLO requirement that the input images are square, and resolution is a multiple of 32. This resolution can be changed to higher values but at the cost of higher computation memory and slower train/test speed. Four classes (3 stages and 1 background class) were used in training, however the detection of the background class was ignored during model performance evaluation. The model was trained for 35.7 k iterations with a batch size of 32 and data augmentation techniques defaulted to the YOLOv3 settings (saturation = 1.5, exposure = 1.5, hue = 0.1). The learning rate and momentum were set to the default values of 0.001 and 0.9 respectively. No transfer learning was employed, with the weights of the convolution neural network initialized at random values. Finally, saved model weights from 33 k iteration was used for testing and validation.

MangoYOLO-rotated: MangoYOLO was originally conceived for on-tree mango fruit detection, and has an architecture based on the YOLOv3 [14] object detection framework, optimized for better speed and accuracy. In this architecture, a YOLO deep learning object detection framework is used with a MangoYOLO CNN classifier [12]. The training parameters for YOLOv3-rotated were retained for MangoYOLO-rotated model training. Images were rotated randomly in the range of 40 degrees. The model was trained for 77.5 k iterations with a batch size of 32. Finally, the saved model weights from 77.4 k iteration were used for testing and validation.

MangoYOLO(-upright): MangoYOLO(-upright) is same as MangoYOLO-rotated except the random image rotation feature in the MangoYOLO-rotated configuration file was turned off, as was the case in the fruit detection work of [12]. The model was trained for 89.5 k iterations with a batch size of 32. Finally, saved model weights from the 66 k iteration were used for testing and validation.

 R^2 CNN(-rotated): R^2 CNN(-rotated) is basically a Faster R-CNN object detection framework with a modification to support training on rotated objects. The R^2 CNN(-rotated) implementation supported only three CNN architectures - mobilenet_v2, ResNet50_v1 and resnet101_v1. Given that deeper CNN models generally produce better results in terms of object detection and classification, ResNet101_v1 CNN architecture was used with R^2 CNN(-rotated) framework for model training. The tensorflow re-implementation of R^2 CNN (https://github.com/DetectionTeamUCAS/R2CNN_Faster-RCNN_Tensorflow) was used for training and testing in this study. All model training parameters

were set to the default values. With R^2 CNN(-rotated), as for Faster *R*-CNN, the input resolution can be set to be square or the original aspect ratio can be preserved (shorter side scaled to 800 pixels and longer side scaled accordingly). The original Basler images 2464×2048 pixels automatically resized to 962×800 pixels during training and testing. The RGB channel pixel mean values were initialized with the values (R = 41.647, G = 41.675, B = 43.048) calculated of the training dataset. The model was trained for 146 k iterations with a batch size of 1 (as no support existed for batches with more than 1 image), learning rate of 0.0003 and momentum of 0.9. ImageNet weights (http://download.tensorflow.org/models/resnet_v1_101_2016_08_28 tar.gz) were used as transfer learning to initialize the R^2 CNN(-rotated) model. Finally, saved model weights from the 146 k iteration were used for testing and validation.

 R^2 CNN-upright: To allow a comparison between rotated and upright box annotation in model training, a R^2 CNN-upright model was established. The R^2 CNN-upright model was trained using a training set of upright annotation boxes. This was achieved by setting the orientation of all boxes to 0 degrees, as used for training of the YOLO method. The training parameters used for the R^2 CNN(-rotated) model were retained for training of the R^2 CNN-upright model. The model was trained for 146 k iterations and the final weight was used for testing and validation.

2.4. Estimation of Peak of Flowering

Repeated (weekly) orchard imaging provided a time course of panicle number by week. The *signal_find_peaks* function from Scipy (www.scipy.org) packages was used to find peaks in the panicle numbers per tree side. Peak properties were specified as *height* = 10 and *distance* = 2. Height determines the minimum height of the peak which refers to the minimum number of panicles to consider as a peak. This parameter helps to filter noise (insignificant small peaks) in the signal. Distance determines the minimum horizontal distance in samples between neighbouring peaks.

3. Results

3.1. Segmentation Method.

The pixel-segmentation method of [21] differentiates pixels associated to panicles from background based on fixed values of colour thresholding. In poorly illuminated areas of images, panicles were not detected (false negative); while in some images, parts of the tree such as branches or brownish/yellowish leaves were incorrectly classified as flower pixels. Two example images are presented, processed using the segmentation [21] and deep learning R^2 CNN methods (Figure 4).

Flowering intensity (ratio of panicle pixels to the canopy pixels) was assessed following the method of [21] and correlated to the panicle counts per tree made using the R^2 CNN(-rotated) method, for all 994 trees (1988 images) of an orchard for each of seven consecutive weeks (Table 3). Better correlation was obtained between the stage Y panicle counts rather than the total panicle count in the last two weeks, as the proportion of stage Y panicles changed (Table 3).

Table 3. Flowering intensity level per tree from pixel segmentation method and Y stage or all stages panicle counts, respectively, from the R^2 CNN(-rotated) method, and the average ratio of stage Y to total panicle count per image, for each week (n = 1988).

Week	1 16 Aug	2 23 Aug	3 30 Aug	4 6 Sep	5 13 Sep	6 20 Sep	7 27 Sept
(Stage Y) R ²	0.903	0.896	0.892	0.788	0,853	0.871	0.708
(Stage $X = Y$) R^2	0,914	0.900	0.777	0.496	0.791	0.846	0.671
(Stage $X + Y + Z$) R^2	0.898	0.865	0.825	0.579	0.254	0.357	0.327
Ratio of stage Y to total panicle count	0.376	0,390	0.395	0.361	0.416	0.320	0.147

Agronomy 2020, 10, 143

8 of 21



Figure 4. Pixel segmentation (**a**,**c**) and deep learning R^2 CNN(-rotated) (**b**,**d**) results for the same images. Flowers in the dark background did not segment properly (**a**), and branches and leaves were erroneously segmented as flower pixels (**c**).

3.2. Deep Learning Methods

An example of one image processed with the three methods of MangoYOLO-rotated, R^2 CNN(-rotated) and R^2 CNN-upright is given as Figure 5.

The Root Mean Square Error (RMSE) for estimates of total panicle count per tree of the validation set was lowest with the YOLOV3-rotated method, while the lowest RMSE for count of stages X, Y and Z was achieved with the MangoYOLO-rotated and R^2 CNN-rotated methods (Table 4). Low RMSE values were associated with low bias values. RMSE and bias were generally lower with use of rotated compared to upright bounding boxes for both YOLO and R^2 CNN methods (Table 4). The highest mean average precision and Weighted F1 score was obtained with MangoYOLO and R^2 CNN-upright models respectively (Table 4).

Agronomy 2020, 10, 143

9 of 21

Table 4. Stage detection results on the validation set using three methods. RMSE refers to a comparison with ground truth assessments of panicles per image. All values refer to number of panicles per tree image. Best results for a given metric and panicle stage are shown in **bold**.

	-	Groun	d Tru	th	- 9	R ² CNN(-Rotated	0		R ² CNN-	Uprigh	t	Ma	ngoYOL	O(-Upri	ght)	Ma	ngoYOI	O-Rota	ited	1	YOLOVS	-Rotate	d
Wk.	X	Y	Z.	total	X	Ŷ	Z	total	X	Ŷ	Z	total	Х	Y	Z	total	х	Y	Ż	total	X	Ŷ	Z	tota
6	1	37	72	110	1	27	64	92	1	30	46	77	0	27	69	96	0	36	73	109	.0	25	100	125
5	13	67	42	122	9	54	36	-99	11	41	29	81	0	48	23	71	3	65	22	90	0	67	40	107
4	28	.91	8	127	-9	62	15	86	22	53	7	82	16	73	11	100	24	78	6	108	11	75	16	102
3	28	69	0	97	16	49	5	70	22	46	0	68	21	57	0	78	24	65	0	89	22	60	1	83
2	46	28	0	74	38	21	1	60	36	17	0	53	43	25	0	68	45	21	0	66	39	23	0	62
1	51	24	0	75	36	16	0	52	43	18	0	61	55	18	0	73	54	17	0.	71	54	20	0	74
	1	RMSE			11.6	16.4	5.4	25.8	6,3	21,8	11.5	32,3	8.0	12.7	7.9	25.6	4.9	6,9	8.2	16.0	9.6	9,3	11.9	15.4
		Bias			-9.7	-14.5	-1.7	-24.3	-5.3	-18.5	-6.7	-30.5	-5.3	-11.3	-3.2	-19.8	-2.8	-5.7	-3.5	-12.0	-6.8	-7.7	15.8	-8.
- 1	Averag	e Prec	iston ¹		56,3	62.0	69,3	62.5	80.8	64.8	67.2	70.9	74,1	76.7	65.6	72.2	68.7	78.1	60.5	69.1	65.4	74.0	55,4	65,4
		F12			69.6	75.6	75.7	74.0	89.4	78.7	80.4	82.0	77.8	77.8	71.4	76.5	75.4	79.0	69.5	76.1	76.5	77.1	67.2	74.

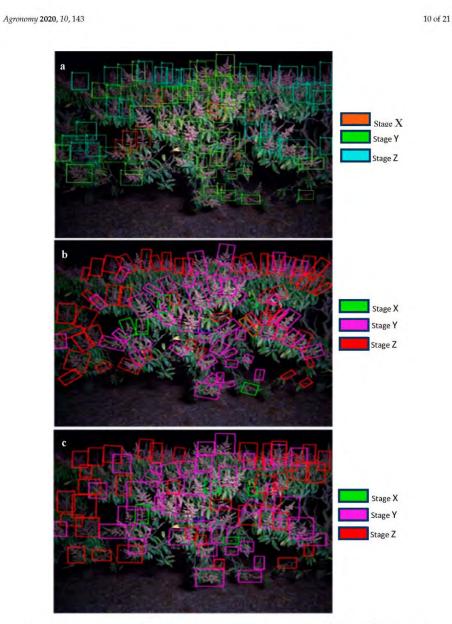


Figure 5. Image processed with three methods for panicle stage detection. (a) MangoYOLO-rotated method. Orange, green and blue coloured boxes represent panicle stages X, Y and Z respectively. (b) R^2 CNN(-rotated) and (c) R^2 CNN-upright methods, respectively. Green, pink and red coloured boxes represent panicle classes of X, Y and Z, respectively.

YOLO and R^2 CNN methods were also used in prediction of the test set images (two images per tree), collected from a different orchard, cultivar and camera to the calibration set (Table 5, Figure 6). Predicted counts were compared to human counts of panicle stages per tree. The MangoYOLO-rotated method achieved the lowest RMSE and bias of the five methods, while the MangoYOLO(-upright)

method returned the highest R^2 but suffered a high bias (Table 5). The R^2 CNN-upright model returned a lower R^2 than the base R^2 CNN(-rotated) model. The R^2 CNN-upright model result was similar to that reported by [21] for a Faster *R*-CNN (VGG-16) method (which uses upright boxes) for the same trees from test set (Table 2).

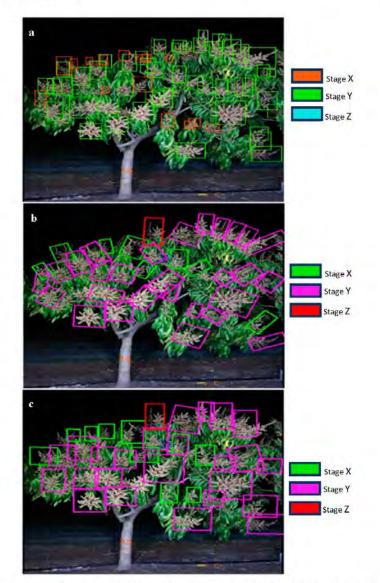


Figure 6. Images of panicle stage detection on Canon images of [21]. (a) MangoYOLO(-upright) method: orange, green and blue coloured boxes represent panicle stages X, Y and Z respectively. (b) R^2 CNN(-rotated) and (c) R^2 CNN-upright methods, respectively: green, pink and red coloured boxes represent panicle classes of X, Y and Z, respectively.

3995

Table 5. Comparison of panicle assessment methods on the test image set (from [21]) in terms of the R^2 and RMSE between machine vision panicle (sum of two sides of a tree) count on images (two per tree) and in-field human counts of panicles per tree.

Detection Method	\mathbb{R}^2	RMSE	Bias
R ² CNN(-rotated)	0.81	91,9	-72.2
R ² CNN-upright	0,76	93.2	-72,4
MangoYOLO(-upright)	0.86	50.7	-30.3
MangoYOLO-rotated	0.80	35.6	-6.4
YOLOv3-rotated	0.83	53.5	-33.6
Faster R-CNN [21]	0.78	1.1	

4. Discussion

4.1. Method Comparison

4.1.1. Pixel Segmentation

The pixel-segmentation method of [21] outputs the total flower and canopy-like pixels and does not provide an estimation of the number of panicles. A correlation was obtained between flowering intensity values and panicle counts, consistent with the report of [21]. However, pixel number per panicle varies with each stage of panicle development, and so confounds the number with the developmental stage. A stronger correlation between flowering intensity and panicle count is expected when all the panicles are at the same stage of development.

The pixel segmentation method also uses a fixed colour threshold range, but colour of panicles and canopy may vary between cultivars and with growing conditions, resulting in false positives and negatives. Use of the segmentation method was therefore discontinued in favour of a deep learning method for object detection, echoing the advice of [12].

4.1.2. Defining Flowering Stages

Mango panicle development is typically described in terms of eight stages: quiescent bud, bud swelling, inflorescence axis elongation, inflorescence sprout, flower opening and inflorescence branching, well-developed inflorescence, and inflorescence starting to set fruit [26]. The early stages are difficult to differentiate from vegetative structures at a distance, and thus, the classification of these stages was not attempted. Initially a four-stage classification was attempted, adding an early stage of development, but the level of uncertainty in the human labelling of the images was high (data not shown). Even with the three-class model, there was uncertainty in annotation, between late X-stage and early Y-stage panicles and late Y-stage and early Z-stage panicles. This uncertainty will adversely affect the classification accuracy of the trained models on the development stage, but it will not affect the model estimate of total panicle number. Figures 7 and 8 illustrate misclassification and the ambiguous classification of panicle stages.

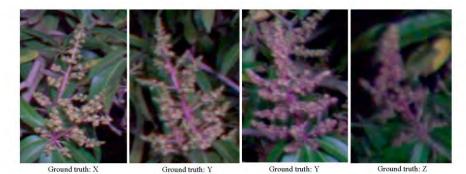
4.1.3. Deep Learning Methods: Use of Rotated Bounding Boxes

 R^2 CNN(-rotated) is a Faster *R*-CNN framework with the added capability of training on rotated objects (bounding boxes). It was hypothesised that that use of rotated bounding boxes in training and prediction would improve results for panicle detection.

Models trained using image rotation and rotated bounding boxes produced numbers closer to the total panicle count per image, i.e., showed lower bias (Table 4). However, lower mAP and average weighted *F1*-scores for the rotated models suggest that there was more error in the classification of detected panicles compared to the models trained with the upright bounding box (Table 4). This result indicated that rotated models were better in detecting panicles of different orientations and suited for applications involving a single class; for example, for the task of detecting and counting panicles.

Agronomy 2020, 10, 143

However, models for classifying panicles to different classes did not benefit from this approach. Thus, upright models are recommended for applications involving multi-class classification.



Predicted: YOLO: X, R²CNN: Y Predicted: YOLO: X, R²CNN: Y Predicted: YOLO: X, R²CNN: Z

Figure 7. Representative examples of differences in classification from YOLO and R^2 CNN models. Ground truth and predicted results by YOLO and R^2 CNN, respectively.

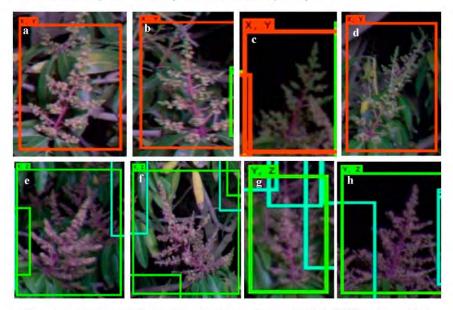


Figure 8. Ambiguity in classification. Examples of the panicle stages which the YOLO models predicted to 50% probability for each class. The top (**a**–**d**) and bottom (**e**–**h**) panels depict panicles classified to both classes X and Y (a,b,e,f), and Y and Z (c,d,g,h), respectively.

We conclude that there was no benefit from use of random panicle rotation as a data augmentation technique for panicle stages classification.

3997

4.1.4. Deep Learning Methods: A Comparison

The five methods (MangoYOLO(-upright), MangoYOLO-rotated, YOLOv3-rotated, R^2 CNN(-rotated) and R^2 CNN-upright) ranked differently in terms of RMSE, bias, Precision and F1 score. Such differences are expected, given the difference in calculation of these metrics. RMSE is a 'gross' metric compared to Precision and F1 score, as false negatives can be offset by false positive detections to yield a low RMSE metric. Therefore, model performance was primarily assessed on the mAP and F1 evaluation metrics (Table 5).

As a rule of thumb, a deeper CNN can yield superior prediction accuracy. However, performance is also related to architecture (connecting layers, etc.) as well as depth [27]. A comparison of the R^2 CNN-upright method with its deeper CNN classifier (ResNet101) to the single- shot MangoYOLO(-upright) method is useful in this respect. In the validation exercise, the R^2 CNN-upright method produced a higher weighted *F1*-score, but a lower mAP score compared to the MangoYOLO(-upright) method (Table 4). This outcome suggests that the YOLO method suffered from lower recall rates.

The performance of the deeper single shot detector, YOLOv3, was also compared to that of the MangoYOLO method. The better results (mAP and *F1*-score) of the MangoYOLO method were consistent with previous results in fruit detection [12], and suggest that the MangoYOLO architecture is better suited to this application than that of YOLOv3.

The YOLOv3-rotated model returned the lowest bias and RMSE for count of panicles per image, but R^2 CNN-upright returned the highest *F1* score (Table 4 and Figure 5) and the MangoYOLO(-upright) model achieved the highest mAP score (Table 4). For the test exercise, the standout result was a low bias and associated low RMSE for the MangoYOLO-rotated model (Table 5).

Panicle image size (in pixels) can vary due to variation in camera to tree distance or use of different camera lens. YOLO models can be robust for such conditions, as the YOLO object detection framework provides multiscale image training as a part of data augmentation. Other data augmentation techniques such as random rotation and random change in hue, saturation and exposure of training samples provides robustness for the YOLO model to deal with variation in lighting conditions and cultivars. In comparison, *R*²CNN like Faster *R*-CNN does not support multiscale training, and the only data augmentation during training provided is image flipping (horizontal and vertical flips). Therefore, the YOLO method of object detection is recommended for applications similar to those considered in the current study.

The speed of classification by YOLO and R^2 CNN methods could not be compared because R^2 CNN supports standard CNN classifier architectures such as ResNet and MobileNet while YOLO uses a custom CNN classifier architecture, with no support for standard CNN classifiers. It is expected, however, that YOLO will process images faster than R^2 CNN as the object detection framework of YOLO uses a single-stage detection technique, in comparison with the dual-stage detection technique of R^2 CNN. In [12], researchers documented speed and memory requirement comparisons of several deep learning object detection frameworks. For example, it took 10, 25 and 67 ms for mango fruit detection on tree images (512 \times 512 pixels)—requiring 873, 2097 and 1759 Mb of GPU memory for MangoYOLO, YOLOv3 and Faster-RCNN (VGG16) models, respectively.

4.2. Applications

One of the challenges in precision agriculture is the display and interpretation of large data sets in a form useful to the farm manager, i.e., an appropriate decision-support tool for farm management is required. Thus, the presentation of data on panicle count by stage of development for every tree in an orchard requires consideration. A key assessment is the timing of flowering, which can be used in conjunction with calendar days or thermal time to estimate harvest date. This is useful for planning harvest resourcing. Inaccurate harvest liming estimation will result either in harvests of less mature fruit, resulting in lowered eating quality, or over-mature fruit, with reduced postharvest life.

15 of 21

For example, panicle stage data can be presented on an individual tree basis by week as a line graph (Figure 9). This figure enables the identification of early flowering areas in one row of an orchard, but it does not scale well to consideration of an entire orchard block.

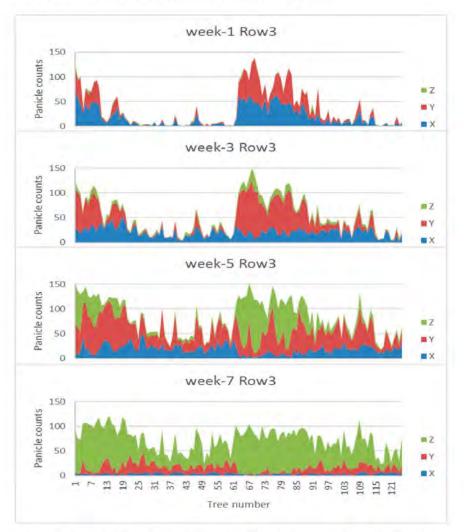


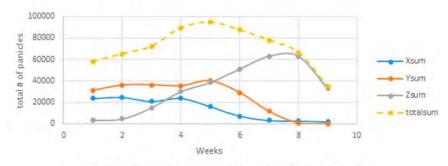
Figure 9. Course (weeks 1, 3, 5 and 7) of panicle number by developmental stage per tree for a row of trees.

Data can also be presented on a farm map, using a colour scale to display values, per week of imaging (Figure 10) using the web-app described by [28], which allows display of the individual images associated with each tree.



Figure 10. (a) Intensity level and (b) panicle count (using R^2 CNN(-rotated)) of an orchard with tree rows (994 trees, 1988 images). The colours green, orange and red colour corresponds to low, medium and high flowering levels, per tree with each coloured dot representing the image of one side of a tree. In the top panel the categories are in terms of flower pixels as a percentage of canopy pixels (<10%, 10% to 25% and >25%). In the bottom panel, the categories are in terms of total panicle number (<30, 30 to 70 and >70 panicles per tree image).

Alternatively, data can be summarised for an orchard block, with a tally of panicles by development stage by week (Figure 11). In the example data, there was a shift in developmental stage from stage X to Y to Z over the monitored period—as expected given panicle development. A peak in total count occurred at week 5. This profile can be interpreted as a single sustained flowering event maintained over four weeks (weeks 1 to 5).





17 of 21

In another approach, the timing of the peak in panicle number per tree can be assessed for individual trees given a time course of images and use of a peak detection routine (Figure 12).

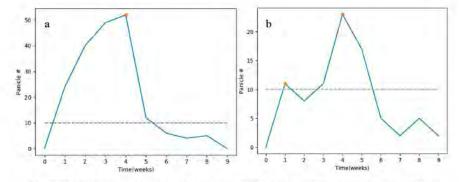


Figure 12. Flowering event detection on stage-X panicle counts for two different trees across 9 weeks of imaging. (a) Single peak and (b) double peaks marked with a coloured dot.

A display of the number of trees with peaks in *X* stage panicle count each week can be displayed, to give a sense of when the orchard generally has a peak flowering event. For example, of 1986 images of 993 trees in orchard A, 168 tree-sides (8%) displayed two flowering events (peaks in stage X) (Figure 13). The first flowering event occurred in the first two weeks of imaging while another peak flowering event occurred on the fourth week. This information on major flowering events can be coupled to temperature records to provide an estimated harvest maturity time based on thermal time [28]. If the events are sufficiently large and temporally separated, the grower can consider these as separate harvest populations.

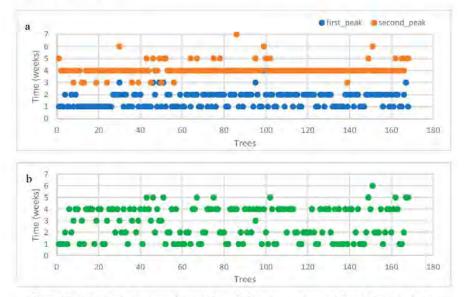


Figure 13. (a) Plot displaying the weeks in which a peak flowering event was noted and (b) plot displaying the week of the largest flowering event for 168 trees in which two flowering peaks were recorded.

Alternatively, data can be presented on a farm map for the selective display of the spread of panicles per week of imaging (Figure 14). In this software [21], an individual tree can be selected to display the flowering level, tree image, image capture date and tree identity (not shown).

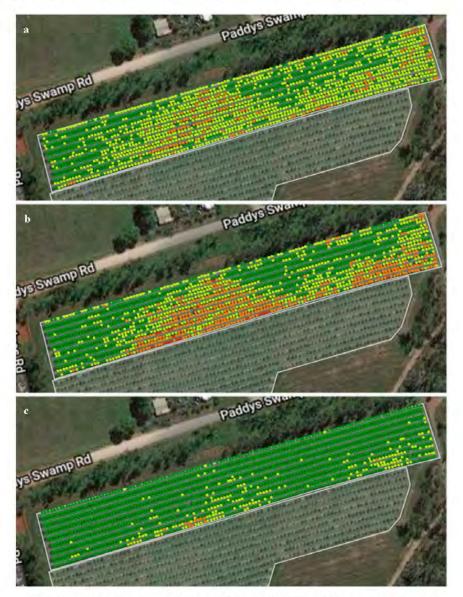


Figure 14. Data of the imaging run of week-1 of Orchard A, in terms of number of panicles at stages (a) X, (b) Y and (c) Z. Dark green, light green, orange and red colours correspond to <30; 30 to 70, 70 to 100 and >100 panicles per tree image.

4002

5. Conclusions

A method is presented in this paper for the estimation of panicle number and stage of development. A method is also presented for the detection of a peak flowering event in a time series. This gives a point from which to calculate a thermal or calendar time to fruit maturation, a critical estimate for harvest management. The extent of flowering also provides an index to potential crop yield (fruit number) although in previous studies we have shown a poor correlation between the number of panicles and the fruit numbers (i.e., variable and high flower and fruit drop). The spatial information on the spread of flowering also provide growers with wealth of information for agronomic treatments.

Deep learning methods can automatically learn the useful features required in an application. For panicle detection, this is likely to involve colour and shape patterns. Similar to that demonstrated by [12] for the fruit detection application, a deep learning method was demonstrated to generalize in terms of application to an orchard of different cultivar, growing conditions and canopy architecture, and also to images acquired with a different camera to that used for the training images. Similarly, [18] reported that a deep learning FCN trained on apple flower images was able to generalize well for peach and pear flower detection, and to operate across camera hardware. The use of deep learning models over traditional segmentation techniques for the application of panicle developmental stage detection is thus confirmed.

In addition to total panicle counts, classification of panicle into several developmental stages was achieved. This is an important step in allowing the extraction of information on the time-spread of flowering, and thus of the future harvest. Counts of panicles in stage X over time were used for estimating the timing of peak flowering events. Counts of panicles in stage Y demonstrated the highest correlation with the flower intensity level per tree.

There has been continuous evolution of CNN architectures in terms of increased depth (number of layers) for better performance, but recent improvements in the representational capacity of deep CNNs has been attributed to the restructuring of processing units (e.g., having multiple paths and connections) rather than just increasing depth [27]. The better performance of MangoYOLO compared to deeper CNN architecture of YOLOv3 for panicle detection is consistent to the report of [12] for fruit detection, and can be ascribed to CNN design considerations.

The YOLO method of object detection is recommended for total panicle classification and count, and similar applications. The procedure offers a large set of data augmentation features (e.g., image rotation, HSV channel values, jitter, image scale etc.) that can be specified by the user to create robust models. Moreover, YOLO is a single shot detection technique and provides a faster detection rate compared to two-stage detection methods (e.g., Faster *R*-CNN) which allows YOLO models to be operated in real-time.

The use of rotated training images improved the detection of panicles and thus this approach is recommended for applications that require counting the instances of objects in natural scenes and having different orientations. However, for applications that require more emphasis on the multi-class classification of the objects, it is recommended to use upright bounding boxes.

There have been several publications on machine-vision-based detection and crop load estimation for tree fruit crops, but relatively few reports of in-field tree crop flower assessment especially in terms of developmental stage. There has been even less attention given to the presentation and management of such data. The flower peak detection and display options presented in the current study should prompt further work in this field.

Author Contributions: Conceptualization, A.K. and K.B.W.; image acquisition, N.A., methodology, A.K.; software, A.K.; validation, A.K., K.B.W. and Z.W.; formal analysis, A.K. and K.B.W.; investigation, A.K.; data curation, A.K. and N.A.; writing—original draft preparation, A.K.; writing—review and editing, K.B.W.; Z.W. and N.A.; visualization, A.K. and Z.W.; supervision, K.B.W.; project administration, K.B.W.; funding acquisition, K.B.W. All authors have read and agreed to the published version of the manuscript.

Agronomy 2020, 10, 143

Funding: This work received funding support from the Australian Federal Department of Agriculture and Water through Horticulture Innovation (project ST19009, Multiscale monitoring tools for managing Australian tree crops).

Acknowledgments: Z.W. acknowledge support of a CQU Early Career Fellowship and A.K. acknowledges a CQU RUN scholarship. Farm support from Chad Simpson and Ian Groves, and manual panicle count by Bryony Wilcox is appreciated. The assistance of Jason Bell of the CQUniversity High Performance Computing cluster is acknowledged.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Aggelopoulou, A.; Bochtis, D.; Fountas, S.; Swain, K.C.; Gemtos, T.; Nanos, G. Yield prediction in apple orchards based on image processing. *Precis. Agric*, 2011, 12, 448–456. [CrossRef]
- Dorj, U.-O.; Lee, M. A new method for tangerine tree flower recognition. In Computer Applications for Bio-Technology, Multimedia, and Ubiquitous City; Springer: Heidelberg, Germany, 2012; pp. 49–56.
- Horton, R.; Cano, E.; Bulanon, D.; Fallahi, E. Peach flower monitoring using aerial multispectral imaging. J. Imaging 2017, 3, 2. [CrossRef]
- Hočevar, M.; Širok, B.; Godeša, T.; Stopar, M. Flowering estimation in apple orchards by image analysis. Precis. Agric. 2014, 15, 466–478. [CrossRef]
- Oppenheim, D.; Edan, Y.; Shani, G. Detecting Tomato Flowers in Greenhouses Using Computer Vision. Int. J. Comput. Electr. Autom. Control Inform. Eng. 2017, 11, 104–109.
- Underwood, J.P.; Hung, C.; Whelan, B.; Sukkarieh, S. Mapping almond orchard canopy volume, flowers, fruit and yield using lidar and vision sensors. *Comput. Electron. Agric.* 2016, 130, 83–96. [CrossRef]
- Diago, M.P.; Sanz-Garcia, A.; Millan, B.; Blasco, J.; Tardaguila, J. Assessment of flower number per inflorescence in grapevine by image analysis under field conditions. J. Sci. Food Agric. 2014, 94, 1981–1987. [CrossRef]
- Guo, W.; Fukatsu, T.; Ninomiya, S. Automated characterization of flowering dynamics in rice using field-acquired time-series RGB images. *Plant Methods* 2015, 11, 7–23. [CrossRef]
- Kamilaris, A.; Prenafeta-Boldú, F.X. Deep learning in agriculture: A survey. Comput. Electron. Agric. 2018, 147, 70–90. [CrossRef]
- Koirala, A.; Walsh, K.B.; Wang, Z.; McCarthy, C. Deep learning—Method overview and review of use for fruit detection and yield estimation. *Comput. Electron. Agric.* 2019, 162, 219–234. [CrossRef]
- Gongal, A.; Amatya, S.; Karkee, M.; Zhang, Q.; Lewis, K. Sensors and systems for fruit detection and localization: A review. Comput. Electron. Agric. 2015, 116, 8–19. [CrossRef]
- Koirala, A.; Wang, Z.; Walsh, K.; McCarthy, C. Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of 'MangoYOLO'. Precis. Agric. 2019, 20, 1107–1135. [CrossRef]
- Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 779–788.
- Redmon, J.; Farhadi, A. YOLOV3: An Incremental Improvement. Available online: https://arxiv.org/abs/1706. 09579 (Accessed on 10 December 2019).
- Dias, P.A.; Tabb, A.; Medeiros, H. Apple flower detection using deep convolutional networks. Comput. Ind. 2018, 99, 17–28. [CrossRef]
- Zeiler, M.D.; Fergus, R. Visualizing and Understanding Convolutional Networks; Springer: Basel, Switzerland, 2014; pp. 818–833.
- 17. Cortes, C.; Vapnik, V. Support-vector networks. Mach. Learn. 1995, 20, 273-297. [CrossRef]
- Dias, P.A.; Tabb, A.; Medeiros, H. Multispecies Fruit Flower Detection Using a Refined Semantic Segmentation Network. *IEEE Robot. Autom. Lett.* 2018, 3, 3003–3010. [CrossRef]
- Chen, L.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal.* 2018, 40, 834–848. [CrossRef] [PubMed]
- Wang, Z.; Verma, B.; Walsh, K.B.; Subedi, P.; Koirala, A. Automated mango flowering assessment via refinement segmentation. In Proceedings of the International Conference on Image and Vision Computing New Zealand (IVCNZ), Palmerston North, New Zealand, 21–22 November 2016; pp. 1–6.

Agronomy 2020, 10, 143

- Wang, Z.; Underwood, J.; Walsh, K.B. Machine vision assessment of mango orchard flowering. Comput. Electron. Agric. 2018, 157, 501–511. [CrossRef]
- Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern. Anal.* 2017, 39, 91–99. [CrossRef] [PubMed]
- Jiang, Y.; Zhu, X.; Wang, X.; Yang, S.; Li, W.; Wang, H.; Fu, P.; Luo, Z. R2CNN: Rotational Region CNN for Orientation Robust Scene Text Detection. Available online: https://arxiv.org/abs/1901.06032 (accessed on 10 December 2019).
- 24. OpenCV. OpenCV Library. Available online: http://opencv.org/ (accessed on 4 November 2019).
- Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
- Yeshitela, T.; Robbertse, P.; Stassen, P. The impact of panicle and shoot pruning on inflorescence and yield related developments in some mango cultivars. J. Appl. Hort. 2003, 5, 69–75.
- Khan, A.; Sohail, A.; Zahoora, U.; Qureshi, A.S. A Survey of the Recent Architectures of Deep Convolutional Neural Networks. Available online: https://arxiv.org/abs/1901.06032 (accessed on 10 December 2019).
- Walsh, K.; Wang, Z. Monitoring fruit quality and quantity in mangoes. In Achieving Sustainable Cultivation of Mangoes; Galán Saúco, V., Lu, P., Eds.; Burleigh Dodds Science Publishing: Cambridge, UK, 2018, pp. 313–338.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).

4008 Appendix C. Published version of Chapter 6

4009 Mobile device machine vision estimation of mango crop load

- 4010 **Koirala A**, Walsh K, Wang Z, McCarthy C Mobile device machine vision estimation of mano crop load.
- 4011 In: International Tri-Conference for Precision Agriculture, New Zealand, 2017.
- 4012 doi:<u>https://doi.org/10.5281/zenodo.895382</u>
- 4013



7th Asian-Australasian Conference on Precision Agriculture

Mobile device machine vision estimation of mango crop load

Anand Koirala¹¹, Kerry B. Walsh¹, Zhenglin Wang¹, Cheryl McCarthy² ¹ Central Queensland University, Rockhampton, Australia ² University of Southern Queensland, Toowoomba, Australia

Abstract

The application of machine vision in orchard was considered in context of mango crop load (fruit number and fruit size). An algorithm for automatic detection and counting of fruits in images of trees in orchard was developed. RGB images were acquired of two sides of mango trees ('dual view'). Fruit count per tree was obtained by harvest of trees, and by manual count of fruit in images. The R² and slope between dual-view and harvest count varied between 0.74 and 0.92, and 0.34 and 0.55, respectively, depending on canopy structure. The fruit counting model involved: (i) fruit-like object detection using HAAR cascade classifier using an AdaBoost technique; (ii) classification of detected region using a multilayer Convolutional Neural Network (CNN). The machine vision count achieved a precision = 0.94, recall= 0.89, and F1 score = 0.9 against a backing board (with a circular scale printed on a blue background), with an RMSE of 3.6 mm for lineal dimension measurement achieved.

Background

Knowledge of fruit size and counts can help to make decisions on agronomic treatments, resource management and market planning. Estimation of crop load and quantity is typically based on previous yield history, and manual fruit count and size measuring of a sample of orchard fruit. Manual in-field counting of fruits is time consuming and inconsistent. Similarly, manual measurement of fruit dimensions using callipers is slow. In practice, few farm managers make these measurements consistently, due to the labour requirement.

Machine vision can be applied to assessment of fruit number and size on tree, given consideration of:

- 1. variation in apparent size of fruit vary within canopy with distance from camera
 - variation in lighting conditions within the canopy
 - variation in colour of fruits and foliage
 - occlusion of fruit by other fruits, branches and foliage

Computer vision and machine learning has been applied to automatic fruit detection and sizing (Sethy *et al.*). High detection rates have been achieved through the combination of various sensor technologies and machine vision techniques. In particular, convolutional neural networks (CNN) have been successfully implemented in many image classification challenges in recent years.

However, the cost of the equipment, the long processing times, and complexity of use are shortcomings. A mobile device running a machine vision application could be used to capture images of a representative number of trees and fruits for estimation of fruit number and lineal dimensions, speeding the current human based sampling protocol. A mobile device based method is also suitable for small farms. A mobile device can also store images, record location and allow for connection to farm networked devices and upload/download data to servers or cloud. However, low processing load is required. The paper reports on the development of mobile applications for crop load estimation through sizing and counting of fruits in digital images.

zenodo.org/communities/pa17



7th Asian-Australasian Conference on Precision Agriculture

2

Methods

Equipment

Images were acquired of mango tree canopies using a Canon DSLR 750D camera (2448x2048 pixels).

Estimating tree crop load from dual view images

Images were captured of both sides of 18 trees in each of three mango orchards. Manual counts were made of fruit visible in images. The trees were strip harvested and fruit counted.

Training image set

20 RGB colour images of trees from a row were randomly selected for training. Fruits and backgrounds (leaves, trunks, branches, and sky) were annotated using the OpenCV annotation tool. This included fruits of various levels of occlusion. Annotated snips (n=2000) of each class (fruit and background) were cropped from the training image set. These snips were used to train both stages (cascade classifier and neural network).

Detection

The open- source computer vision and machine vision library OpenCV (OpenCV, 2017) was used for training and testing of the cascade classifier.

HAAR (Viola *et al.*, 2001) like features were extracted into a pool of features from all the training images and a boosting algorithm AdaBoost (Papageorgiou *et al.*, 1998) used to build a cascade classifier object detection model from the weak separate classifiers. In this experiment three basic HAAR like features (Figure 1) were used. Each feature is a value obtained by subtracting sum of pixels in black region from sum of pixels in white region.

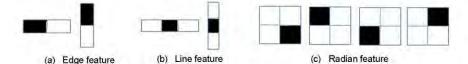


Figure 1. Basic HAAR like features

Multiscale detection from OpenCV was used to detect fruit of all possible sizes in the image. The minimum object size that this model can detect is the height and width parameters provided during the training. The smallest fruit size in the input images (2448x2048 pixels) was 28x28 pixels. A scale factor of value 1.1 was used, meaning the full image is reduced by 10 % in each step in multiscale detection from a single scale model. There can be many detection for the same object at different scales therefore the parameter 'minNeighbors' was set to 4, i.e. at least 4 detection windows must exist in order to consider the object as fruit. A Region of Interest (ROI) was generated for all objects being detected as fruit by the cascade classifier. These regions also contained many false positives such as leaves and branches. A validation process involving a convolutional neural network was used to remove the false positives.

Classification stage

The CNN model used in this paper follows LeNet (LeCun *et al.*, 1998) architecture with some modifications in the parameters. This model features a series of convolutional layers followed by maxpooling layers. Six layers were used (Fig. 2). The CNN model was implemented and trained in DL4J (Gibson *et al.*, 2016), an open source, distributed, deep learning library. The model was trained with 2000 snips each of fruits and background cropped from full canopy images. All three channels (RGB) were used as input to the neural network.

zenodo.org/communities/pa17

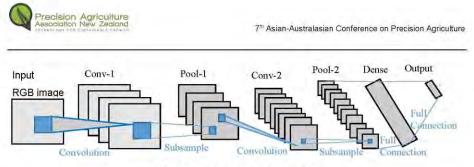


Figure 2. The LeNet architecture

The ROIs generated from the cascade classifier were resized to 28x28 pixels and fed as inputs to the CNN model. An activation function 'ReLU' was used for convolution and dense layers. The model passes the input through a series of convolution and subsampling (max-pool) layers and finally a probability score for each class (fruit/background) is generated using a logistic regression function 'softmax'. An optimum threshold value of 0.79 on the score was selected based on the precision-recall curve. Therefore any ROI with score > 0.79 was considered to be a fruit and included in the count.

Fruit sizing

RGB image captured from a mobile phone were converted to CIE L*a*b* colour space and the b* channel extracted. The colour of background was chosen blue and that of marker yellow because these two colours are opposing colours in b* channel. The fruit and scale was segmented from background using Otsu's (Otsu, 1979) thresholding algorithm. Morphological operations were performed on the resultant binary image to segment the fruit stalk, an unavoidable feature for in-field images. An upright rectangular bounding box was drawn around the fruit perimeter in the image and pixel dimensions for length and width of the fruit calculated. The ratio of diameter (pixel) of marker to the known physical diameter (4 cm) of circular marker was used as a scale to allow estimation of the lineal dimension of fruit from the images.

Results

Estimating tree crop load from dual view images

Though all the fruits on a tree are not visible in the dual view images, the correlation of dual view image count to harvest counts exceed 0.74 in all cases.

		Dual view image count vs	Harvest Count
farm	Number of trees	R ²	Slope
A	18	0.79	0.34
В	18	0.74	0.35
С	18	0.77	0.41
D	18	0.84	0.55
E	18	0.92	0.46

Table 1. Correlation between dual view image count and in-field harvest count for several orchards

Fruit detection and count

In-field tree canopy images contain a varying number of fruits as well as much background (leaves, branches, sky, etc.). The cascade structure of the classifier was successful in discarding most of negative samples in a few early stages, based on the evaluation of a small set of features (Fig. 3). A precision = 0.94, recall =0.89, and F1 score = 0.9 assessed against human image count was achieved with the combination of the HAAR classifier and the CNN.

zenodo.org/communities/pa17



7th Asian-Australasian Conference on Precision Agriculture



Figure 3. Fruit detection in the images

Fruit sizing Imaging fruit against the background and scale (Fig. 4), a $R^2 = 0.95$ and RMSE = 3.6 mm was achieved (n = 40 fruit).



Figure 4. In-field imaging (left) and the result of fruit sizing mobile app (right)

Discussion

Estimating tree crop load from dual view images

The slope of the correlation between the dual view and the actual tree count varied with orchard, with denser canopies having more occluded, non-visible fruit. This was also true between trees within an orchard, with more variation accounting for a decreased R² between dual view and the actual tree count (data not shown). Future work will consider measures of foliage density as indices of the proportion of occluded fruit, to be used in fruit count correct per tree.

4

zenodo.org/communities/pa17



7th Asian-Australasian Conference on Precision Agriculture

Fruit detection and count

In-field tree canopy images contain a varying number of fruits as well as much background (leaves, branches, sky, etc.). The cascade structure of the classifier was successful in discarding most of negative samples in a few early stages, based on the evaluation of a small set of features. The CNN classifier stage was needed to further reduce false positives, but it was applied to only a subset of the original image. Thus the time for detection was greatly reduced. For an image having an average of 103 detections, the algorithm running on a laptop (CPU 2.4 GHz, RAM 8 GB, 64 bit Windows OS) took an average of 2.2 seconds for detection and 1.6 seconds for classification of all objects detected. A couple of seconds delay in processing the image is all right for the sampling approach as the user is expected to spend some time while moving through the orchard to acquire the next sample.

The fruits occluded by other fruit in clusters were sometimes not detected by the HAAR cascade classifier and if the CNN model was not trained with sufficient samples of images having adequate variation in lighting conditions, model performance was degraded. Model results plateaued for training set sizes > 1500 in this case, but the image number for training will depend on the variation expected in the validation sets. The use of scene specific knowledge and constraints should be considered throughout the model training process. The proposed model for fruit counting should generalize well to other tree fruit crops through transfer learning, after some fine tuning of CNN model parameters.

Conclusion and future work

Use of mobile device applications for yield estimation have potential as a low-cost and easily adoptable solution for farmers/growers, feeding into decision support tool for timely resource planning and orchard management.

Further work is required to (i) develop a canopy fruit occlusion index, for correction of dual view to total fruit count; (ii) collect images from further orchards, varying in canopy shape and fruit and foliage colour, (iii) documentation of the effect of test set size on CNN model performance and (iv) addition of a feature to allow cropping of the image on the screen of the mobile device, allowing for fruit load estimation of only one tree canopy at a time. For the fruit sizing mobile device application, addition of a distance measuring capability (e.g. time of flight laser range meter) would remove the need for use of a scale bar.

Acknowledgments

We acknowledge Groves and Simpsons farm for orchard access and support through HIA project MT14018. Anand Koirala acknowledges support of a RUN scholarship provided by CQ University.

References

Gibson A, Patterson J, Nicholson C 2016. Deeplearning4j: Open-source, Distributed Deep Learning for the JVM [Online]. Available: https://deeplearning4j.org [Accessed 16 June 2017].

Lecun Y, Bottou L, Bengio Y, Haffner P 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE 86: 2278–2324.

OPENCV 2017. OpenCV library [Online]. Forge. Available: http://opencv.org/ [Accessed 16 June 2017].

Otsu N 1979. A threshold selection method from gray-level histograms. IEEE transactions on systems, man, and cybernetics 9: 62–66.

Papageorgiou CP, Oren M, Poggio T 1998. A general framework for object detection. Computer vision, 1998. Sixth international conference on. IEEE 555–562.

Sethy PK, Panda S, Behera SK, Rath AK. On tree detection, counting & post-harvest grading of fruits based on image processing and machine learning approach - A review.

Viola P, Jones M 2001. Rapid object detection using a boosted cascade of simple features. Computer Vision and Pattern Recognition. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, 2001. IEEE, I-I.

5

zenodo.org/communities/pa17

4018

4020 Appendix D. Method comparison

4021 Fruit load estimation in mango orchards- a method comparison

- 4022 Underwood JP, Rahman MM, Robson A, Walsh KB, Koirala A, Wang Z (2018) Fruit load estimation in
- 4023mango orchards a method comparison. Paper presented at the ICRA 2018 Workshop on4024Robotic Vision and Action in Agriculture, Brisbane, Australia.

Fruit load estimation in mango orchards - a method comparison

J. P. Underwood¹, M.M. Rahman², A. Robson², K. B. Walsh³, A. Koirala³, Z. Wang³

Abstract-The fruit load of entire mango orchards was estimated well before harvest using (i) in-field machine vision on mobile platforms and (ii) WorldView-3 satellite imagery. For in-field machine vision, two imaging platforms were utilized, with (a) day time imaging with LiDAR based tree segmentation and multiple views per tree, and (b) night time imaging system using two images per tree. The machine vision approaches involved training of neural networks with image snips from one orchard only, followed by use for all other orchards (varying in location and cultivar). Estimates of fruit load per tree achieved up to a R² = 0.88 and a RMSE = 22.5 fruit/tree against harvest fruit count per tree (n = 18 trees per orchard). With satellite imaging, a regression was established between a number of spectral indices and fruit number for a set (n=18) of trees in each orchard (example: R² = 0.57, RMSE = 22 fruit/tree), and this model applied across all tree associated pixels per orchard. The weighted average percentage error on packhouse counts (weighted by packhouse fruit numbers) was 6.0, 8.8 and 9.9% for the day imaging system, night imaging machine vision system and the satellite method, respectively, averaged across all orchards assessed. Additionally, fruit sizing was achieved with a RMSE = 5 mm (on fruit length and width). These estimates are useful for harvest resource planning and marketing and set the foundation for automated harvest.

I. INTRODUCTION

The management of a tree crop harvest can be aided by pre-harvest estimations of fruit load in context of harvest logistics (e.g. labour, field equipment and packing material requirements), transport and storage requirements and marketing activity (e.g. forward selling). There is potential for yield estimation of tree fruit crops using remote sensing or machine vision. Fruit load per tree could be proportional to canopy volume or foliage health, with canopy volume potentially indexed by canopy area as assessed in satellite imagery, and canopy health potentially indexed by a spectral index from the satellite multispectral data. Alternately, fruit number and size could be assessed by machine vision using imagery collected from a mobile ground based platform. At the very least, these techniques could provide a categorization of trees by level of crop load, allowing for a reduced manual sampling effort. In the current study we report on a parallel investigation of ground and satellite based estimation for mango crop load, with application across multiple orchards.

A WorldView-3 satellite imagery based system for estimation of fruit load based on canopy health and area has been previously described for macadamia and avocado by [1] (e.g. $\mathbb{R}^2 = 0.81$, 0.68 and 0.72 on three avocado blocks, respectively, for 18 trees per block). This remote sensing approach allows for a yield forecast of large areas. However, Robson *et al.* [1] reported that the VI used and the relationship slope between that VI and fruit load varied with orchard, necessitating field-work for calibration at each orchard.

Machine vision systems for estimation of mango fruit load on tree have been previously reported by Payne et al. [2] and Stein et al. [3]. The former study [2] used a night imaging, dual-view approach (two images of each tree, from the two inter rows), with a report of a coefficient of determination (R2) of only 0.74 relative to human count, with fruit number under estimated due to canopy occlusion. Stein et al. [3] employed a faster regional convolutional neural network (R-CNN) detector with both a dual-view and a multi-view (multiple images of each tree, from the different angles and two inter rows) approach, with the latter using trajectory data to track fruits between frames to avoid repetitive counts. Detected fruits were ascribed to individual trees by reference to a LiDAR-projected 2D image mask and 3D locations of every fruit were calculated by triangulation. The multi-view count was precise $(R^2=0.90)$ and accurate (slope = 1.01) while the dual-view approach was more precise $(R^2=0.94)$ but less accurate (slope = 0.54) for the assessed set of 18 sample trees A unity slope implies a correction factor and associated field calibration work may not be required, but denser canopies still occlude some fruit, necessitating calibration even for the multi-view approach. In the current study, image training for machine learning was based on snips from one site only, with the model used on other fields. Correction factors (harvest to machine vision count) were calculated per orchard. The use of in-field machine vision to estimate fruit size was also demonstrated, extending the work of Wang et al. [4].

11. MATERIALS AND METHODS

A. Field material and harvest

Field work was undertaken in one orchard in the 2016/17 season and five orchards in the 2017/2018 seasons, involving different farms and mango cultivars. Orchard 1 was a commercial mango (cultivar CalypsoTM) block near Bundaberg, Australia. This site was also utilized in the study of Stein *et al.* [3] in a previous year. Orchard 2 was an adjacent block of the same cultivar, with larger trees. Orchard 3 was on the same farm and of the same cultivar, with yet larger trees with intertwined neighbouring tree canopies. This orchard was located about 1 km from Orchard 1. Orchards 4 and 5 were on different farms in the same district. Orchard 4 featured the R2E2 cultivar with one large completely intertwined block and

¹ J. P. Underwood is with Anstralian Centre for Field Robotics, University of Sydney, Sydney, Australia (e-mail: j.underwood @acft.usyd.edu.au).
⁷ M. M. Rahman and A. Robson are with Precision Agriculture Group,

University of New England, Armidale, Australia, ³ K. B. Walsh, A. Koirala and Z. Wang are with the Institute for Future Farming Systems, Central Queensland University, Rockhampton, Australia.

a second small block with well separated trees, while Orchard 5 featured relatively small, well separated trees of Honey Gold™ cultivar. Orchard 6 was located some 400 km distant, and involved large trees with intertwined canopies of the cultivar Honey Gold.

For each orchard, a set of ea.18 'calibration' trees were selected on the basis of canopy NDVI (six each of high, medium and low values; based on WV3 satellite imagery, see description in next section). Fruit load on the calibration trees of Orchard 1 was assessed by a human in-field count at the time of in-field imaging. For all orchards, the calibration trees were harvested and fruit count recorded. Each block was commercially harvested on the following day, with fruit passed over a commercial fruit grader which provided total orchard yield (# of fruit and # of trays). Fruit of consistent size were packed to trays of 7.5 kg weight, with tray sizes ranging from 12 to 28 (fruit per tray). Harvest occurred during December -January in 2017-18.

In current best farm practice, a manual count of fruit per tree is undertaken of a sample of trees to obtain an orchard yield estimate. This practice was mimicked using the calibration tree harvests to estimate total crop yield.

B. In-field machine vision imaging

Two imaging platforms were employed (Figure 1). Platform A (based on that of Stein *et al.* [3]) is based on an autonomous platform equipped with a high-precision GPS/INS device, LiDAR and a 12 MP RGB camera. The platform was operated in daylight hours, using high intensity Xe flash lamps to enable short exposure times to avoid motion blur and background light variation issues. Platform B (based on that of Payne *et al.* [2]) was mounted on a farm vehicle and was equipped with a GPS device utilized for triggering of imaging based on previously mapped tree locations, a 5 MP RGB camera, a depth camera (Microsoft Kinect v2) and 720 W of LED lighting, with imaging undertaken at night. Both platforms were operated at a speed of approximately 5 km/h, with imaging of a block of 1000 trees completed in 2 h.



Figure 1. Imaging platform A (left) and B (right).

<u>Platform A</u>. The techniques involved in LiDAR based estimates of canopy volume and multi-view machine vision estimates of fruit load per tree are described by Stein *et al.* [3] and Bargoti and Underwood [5]. A faster region convolutional neural network (FR-CNN) algorithm was trained on >1500 snips from images collected of a CalypsoTM orchard (Orchard 1) in the previous 2016-7 season, with the trained model applied to all orchards imaged in the 2017-18 year. An example of detection is shown in Figure 2.

Fruit detection was undertaken both for a single image per side of the tree ('dual-view') and for tracked fruit in ca. 25 images per side of each tree ('multi-view'). Fruit were detected from multiple images to reduce the effect of occlusion from any single viewpoint, and were tracked to minimise redundantly counting the same fruit several times. This was achieved by associating detections from frame to frame, using GPS/INS to estimate the camera trajectory. Fruit detections within each image were associated to a specific tree by projecting segmented LiDAR data of each tree to the corresponding image frames to form image masks. To mitigate double counting of fruit apparent in the images from opposing sides of the tree, the method of Stein *et al.* [3] was modified by clipping the LiDAR masks at a vertical plane at the geometric centroid of each tree.

The performance per-tree was assessed by comparison to a manual count of fruit on each of 18 trees in Orchard 1, in December 2017 with fruit still on the tree, six weeks prior to harvest. For block-level estimates, a calibration factor was obtained from post-harvest counts of 18 trees per block from the previous year. Consequently, the reported block estimates for platform A were true predictions that were provided to farm management 6 weeks prior to harvest. The system was used as a 'black box' with no labour-intensive training, calibration or refinement using data from the current year.

Platform B. With night-time imaging, the dark background of the target tree was easily segmented using Otsu's thresholding [6] technique. This technique also enabled segmentation of tree canopies except for orchards where neighboring tree canopies were overlapped, in which case a fixed margin crop was applied. Images of trees in two rows of Orchard 1 (excluding calibration trees) were used for deep learning training, with ground truth bounding boxes. The deep learning technique of Faster R-CNN (VGG) and YOLO v3 [7] were trained with 11820 annotated fruit in 1,300 tiles, with tuning based on a further 861 fruit in 130 tiles. The trained model was then used to localize and count fruit per image for all orchards, with the fruit number per tree estimated as the sum of count for images of both tree sides ('dual-view'). However, not all fruit in the canopy can be captured in the dual-view images, so a correction factor was required. This can come from the historical yield results or from a manual fruit counts on representative sample trees. In this study the correction factor was based on the calibration tree harvest undertaken for each orchard. In addition, camera depth information enabled on-tree fruit sizing as described in [4]. A color-based thresholding was used to remove the background (e.g. leaves) from YOLO v3 output, and an elliptical fitting method was then applied to identify well separated fruit. Image size and depth information was used to estimate fruit size, which is correlated to fruit mass [8, 9].



Figure 2. Examples of imaging and fruit detection from Platform A (top) and B (bottom)

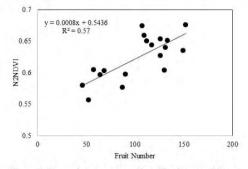
C. Satellite imaging

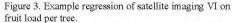
WV3 satellite imagery of Orchard 1 was obtained on August 18, 2017, some 5 months before harvest. The method of Robson *et al.* [1, 10] was employed. Briefly, an unsupervised classification of trees to high, medium and low Normalised Difference Vegetation Index (NDVI) categories (NDVI = (NIR1 – R)/(NIR1 + R)) was undertaken, with random selection of six trees from each class for calibration activities. A 1.5 m radius area around the central point of each calibration tree was segmented using ArcGIS 10.2 (Environmental Systems Research Institute, Redlands, CA, USA), and 18 indices relevant to crop biomass calculated for each tree [1]. Fruit number per tree was regressed against these indices. The index with the highest regression coefficient of determination for each orchard was adopted, with the model applied to the canopy associated average index value for the whole orchard. Tree canopy pixels were segmented using a 2D scatter plot (NIR1 against Red). The estimated average yield per tree for the orchard was multiplied by tree number to provide an estimate of orchard fruit yield.

III. RESULTS AND DISCUSSION

A. Satellite image calibration

Of the 18 spectral indices calculated from the satellite imagery, the best relationship was obtained with the N2RENDVI = ((NIR2-RedEdge)/(NIR2 + RedEdge)) for orchard 1, 2 and 4; N1RENDVI = ((NIR1-RedEdge)/(NIR1 + RedEdge)) for orchard 3; and N2NDVI = ((NIR2-R)/(NIR2 + R)) for orchard 5. The R² of the linear regression between harvested fruit number per tree and VI ranged between 0.10 and 0.57, while RMSE varied between 22 and 151 fruit per tree across the assessed orchards (data not shown, see Figure 3 for example).





B. In-field machine vision calibration

Platform A. The FR-CNN algorithm was trained on images from the previous season. Correction factors calibrated by comparing machine vision counts to post-harvest individual tree counts from the previous year resulted in a value of 0.96 for separated Calypso trees (Orchard 1), 0.95 for a young R2E2 block (Orchard 4) and 0.875 for a mature R2E2 block (Orchard 5). The linear regression of multi-view machine vision counts to pre-harvest manual field counts in Orchard 1 in the current year was described by a similar slope of 0.97, R² of 0.88 and RMSE of 22 fruit (see Figure 4a), which confirmed the appropriateness of using calibration data from the previous year (i.e. adopting field calibration from the current year did not significantly affect block-level estimates). Calibration factors were similar to that reported by Stein et al. [3] for Orchard 1, meaning factors did not change significantly over three consecutive seasons, which is encouraging for using the system as a 'black box'.

<u>Platform B.</u> For the YOLO model, the best F1 score (0.951) was obtained for the training set with a Non-maximum

suppression (NMS) threshold of 0.4 and class confidence cut-off threshold of 0.257 (data not shown). The correlation between human count of fruit per image and the machine vision count was strong (e.g. $R^2=0.97$, slope=1.01 for the Orchard 1 calibration tree set, Figure 4b), indicative of high accuracy in the machine vision count. This result was comparable to that obtained with the VGG F-RCNN for these night images, but the YOLO model operated at an order of magnitude higher speed (data not shown).

However, the correlation of dual view machine vision estimate number of fruits to harvest number was characterized by a \mathbb{R}^2 between 0.64 and 0.85 and a slope between 0.37 and 0.69 for the orchards considered (data not shown). Thus, while the technique was successful in detecting exposed and partially occluded fruit, a significant proportion of fruit on the tree remain unseen in the dual view technique. Thus a correction factor for occluded fruit was employed, based on total fruit numbers of the calibration trees.

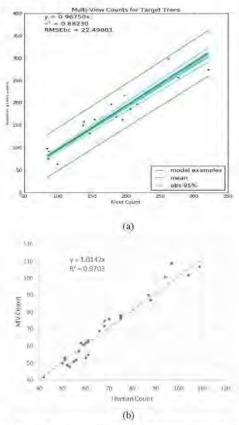


Figure 4. Regression of (a) fruit counts estimates per tree from platform A machine vision system against manual field counts in Orchard 1; (b) platform B YOLO v3 estimate against human count of fruit in tree images.

C. Orchard yield estimate validation

The orchard fruit load estimates were compared to harvest packline data and to an estimate based on the count of a subsample of (calibration) trees (Table 1). The subsample method involved a high human labour input, and the estimates varied between 97 and 150% of packhouse totals.

The weighted average percentage error (weighted by packhouse fruit numbers) was 6.0, 8.8 and 9.9% of packhouse counts for the day imaging system, night imaging machine vision system and the satellite method, respectively, averaged across all orchards assessed (data of Table 1).

All imaging methods achieved a good result on Orchard 1, with poorer results obtained on other orchards. The size and open canopy structure of Orchard 1 particularly suited in-field imaging, in comparison to other orchards with either smaller canopies (more chance for double counting of fruit) or larger canopies (more occlusion of fruit).

The satellite imagery procedure suffered a high uncertainty in the calibration of tree load to VI (Fig. 3), but the accuracy of the total orchard predictions was consistently high (Table I). Note that estimates were based on imagery collected five months before harvest. To guide farm management decisions, calibration could be based on human count of fruit on tree at a time well before harvest.

Dual-view platform B delivered orchard predictions that over-estimated actual harvest, with underestimation due to occluded fruit exceeded by overestimation due to double counting from the two sides of the tree. The multi-view machine vision method decreased count errors associated with occluded fruit and double counting of fruit from imaging each tree from both inter-row sides, thus reducing the reliance on a manually estimated correction factor per orchard. The model was trained using image snips from one orchard only, and that from the previous season, with the predictions thus demonstrating model robustness across season, orchard and location.

With an industry move to higher density, smaller tree plantings, this method offers potential for use without need to estimate a correction factor per orchard. Additional to orchard fruit load estimation, these methods can monitor the yield of individual trees over seasons, enabling the identification of elite trees.

Orchard	Tree #	Packhouse count	Sum-sample	wviii	Multi-view (Platform A)	Dual view (Platform A)	Dual-view (Platform B)
1 Calypso	494 (44)	97,382	94,700 (97%)	94,293 (97%)	94,115 (97%)	92119 (95%)	100,217 (103%)
2 Calypso	386 (54)	67.110	75,810 (113%)	59,552 (89%)	61,279 (91%)	54912 (82%)	74,435 (111%)
3 Calypso	980 (39)	175,431	171,794 (98%)	201,036 (115%)	163,709 (93%)	149488 (85%)	1.1.0
4 R2E2	78 (18)	2,110	3,159 (150%)		2,693 (128%)	3,860 (183%)	2,701 (128%)
4 R2E2	224 (18)	21,180	24.086 (114%)	20,891 (99%)	21033 (100%)	27003 (128%)	
5 HG A	616 (18)	36,490	44,557 (122%)	43,814 (107%)	1		47,248 (130%)
6 HG B	994 (23)	143,544	153,162 (107%)				152,756 (106%)

Table 1. Harvest packhouse count of fruit and estimation from several methods involving count of a sample of trees or satellite imagery VL and multi-view/dual-view machine vision of the entire or-chardl. Values in percentage are estimated fruit muches as a % of harvest mumber. The value in frackets in the Tese # column is the mutaber of trees harvests for the Sub-sample stimate. The Sub-sample method refers to use of the average of fruit numbers for the calibration trees, multiplied by tree number fer or/chard.

D. In field fruit sizing

Size of un-occluded fruit was estimated from pixel dimensions and estimated camera to fruit distance using Platform B (imaged one day before harvest). The RMSE on estimated length and width was 5 mm relative to caliper measurements (data not shown). The length and width dimensions were converted to weight estimates using an allometric relationship (W = $0.49^{*}L^{*}W^{2}$), and to tray size based on use of a 7.5 kg tray. The resulting population distribution, derived from all non-occluded fruit in images of all trees in the orchard, was comparable to that of the packline data (Figure 5). Practical use would involve measurement of fruit size some weeks before harvest, with an assumed rate of increase applied to estimate size distribution at harvest.

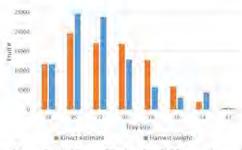


Figure 5. Packhouse (blue) and in-field image (orange) based estimates of the population distribution based on tray size of fruit, for Orchard 1.

IV. CONCLUSION

The three studied approaches were all able to provide sufficiently accurate estimates to support farm management

and decision making. Satellite imagery allows for area wide assessments. Estimation was based on imagery collected five months before harvest in this study, although orchard level calibration is required using fruit counts of calibration trees at a time closer to harvest. Reliability of in-field machine vision estimates of fruit load was improved with use of the multi-view technique, with need for a correction factor for occluded fruit likely to be removed for orchards with narrow canopies. Note that for in-field machine vision counts, the whole orchard need not be assessed, but rather a sample size consistent with the required precision. Estimates of fruit load per tree accumulated across years may be useful for selective breeding and to inform management interventions. Fruit size estimation complements fruit load estimation to inform harvest management decisions. These developments in fruit detection, localization and sizing also set the scene for a robotic harvest operation.

ACKNOWLEDGMENT

Support of Simpsons and Philpots farms, Childers, Queensland, Groves Grown farm, Bungundara, is appreciated. This work was supported by the Australian Centre for Field Robotics (ACFR) at The University of Sydney, a CQUniversity RUN scholarship to AK (with co-supervision of C. McCarthy of Uni Southern Qld) and CQUmi fellowship to ZW, and the Precision Agriculture Group at University of New England. Funding support (grants ST15002, ST15005 and ST15006) from Hort. Innovation Australia and the Australian Government Department of Agriculture and Water Resources as part of its Rural R&D Profit program is acknowledged.

REFERENCES

 A. Robson, M. Rahman, and J. Muir, 'Using worldview satellite imagery to map yield in avocado (Persea americana): a case study in Bundaberg, Australia," Remote Sensing, vol. 9, no. 12, p. 1223. 2017.

- [2] A. Payne, K. Walsh, P. Subedi, and D. Jarvis, "Estimating mango A. Payne, K. Walsh, P. Subedi, and D. Jarvis, "Estimating mutgo crop yield using image analysis using fruit at "stone hardening" stage and hight time imaging," *Computers and Electronics in Agriculture*, vol. 100, pp. 160-167, 1// 2014. M. Stein, S. Bargoti, and J. Underwood, "Image based mango fruit detection, localisation and yield estimation using multiple view geometry," *Sensors*, vol. 16, no. 11, p. 1915, 2016. Z. Wang, K. B. Walsh, and B. Verma, "On-tree mango fruit size estimation using RGH-D images," *Sensors*, vol. 17, no. 12, p. 2738, 2017.
- [3]
- [4]
- [5]
- [6]
- [7]
- [8]
- [9]
- estimation using RGH-D images." Sensors, vol. 17, no. 12, p. 2738, 2017.
 S. Bargoti and J. Underwood, "Deep fluit detection in orchards," in 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 3626-3633.
 N. Otsu, "A threshold selection method from gray-level histograms," Automatica, vol. 11, no. 285-296, pp. 23-27, 1975.
 J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv preprint arXiv;1804.02767, 2018.
 W. Spreer and J. Muller, "Estimating the mass of margo fluit (Mangifera indica, ev. Chok Anan) from its geometric dimensions by optical measurement," Computers and Electronics in Agriculture, vol. 75, no. 1, pp. 125-131, 10' 2011.
 N. T. Anderson, P. P. Subedi, and K. B. Walsh, "Manipulation of margo fluit dry matter content to improve eating quality," Scientia Horticulturae, vol. 226, pp. 316-321, 2017.
 A. Robson, J. Patty, D. Joyce, J. Marques, and P. Hofman, "High resolution remote sensing, GIS and Google Earth for avocado fluit quality mapping and tree number auditing," in Proceedings of the 29th International Horticultural Congress, Brisbane Convention and Exhibition Centre, Brisbane, Australia, 2014, pp. 17-22. [01]

4032 Appendix E. Automated Mango Flowering Assessment via Refinement 4033 Segmentation

4034Wang Z, Verma B, Walsh KB, Subedi P, Koirala A Automated mango flowering assessment via4035refinement segmentation. In: International Conference on Image and Vision Computing New4036Zealand (IVCNZ) 2016 IEEE, pp 1-6. doi:https://doi.org/10.1109/IVCNZ.2016.7804426

Automated Mango Flowering Assessment via Refinement Segmentation

Zhenglin Wang, Brijesh Verma Centre for Intelligent Systems School of Engineering and Technology Central Queensland University, Australia z.wang@cqu.edu.au

Abstract—An automated flowering assessment system for mango orchards was proposed. Segmentation of flowers from a complex background (i.e. leaves, branches and ground) was achieved based on (i) colour correction via adjustment of the brightness and contrast to a reference level, to rectify the illumination variability spatially within and between images; (ii) colour thresholding with fixed thresholds to separate flowers, although with some branches and trunks; and (iii) SVM classification to refine the segmentation results, removing the branch and trunk errors. Mango tree canopy images (n=160) were acquired during a five-week flowering period, with 15 of the images used in calibration and 145 used in validation. The proposed method had a good correlation with human scoring, with coefficient of determination (R^2) of 0.87.

Keywords—thresholding; segmentation; classification; mango orchard; flowering assessment

I. INTRODUCTION

Mango flowering is a critical physiological event in the production of fruit [1]. The floral developmental pathway includes induction, floral meristem initiation, elongation, bud opening, anthesis, pollination and fruit set [2]. Several factors, principally low temperatures, promote flowering, and many factors (e.g. rain, water stress) can impact flower and fruitlet retention.

Mango trees may flower asynchronously between trees and even between individual branches on a single tree, resulting in variability in time of fruit set and maturation for harvest [3]. Various measures have been proposed to reduce flowering variability for harvest synchronization [1, 3]. In practice, however, it is useful to monitor time of flowering as an input to fruit maturation models, with several harvests conducted of the same orchard if flowering is spaced over several weeks. Knowledge of anticipated harvest date can inform farm logistics and crop marketing [4]. Currently farmers may manually flag early flowering trees using tree ribbons, which is tedious and labor intensive for an orchard with thousands of fruit trees.

Assessment of flowering can also be helpful to guide crop management, including flower or fruitlet thinning and variable rate spraying (halting spraying of trees with no flowers and thus no fruit) [5, 6]. Therefore, flower detection

978-1-5090-2748-4/16/\$31.00 @2016 IEEE

Kerry B. Walsh, Phul Subedi, Anand Koirala Institute for Future Farming Systems Central Queensland University Rockhampton, Queensland, Australia

and monitoring has attracted increasing interest in the machine vision field. For example, Bulanon *et al.* proposed to use aerial multispectral imaging to monitor peach flowering [7] and Nielsen *et al.* developed a stereo vision based system for automated blossom thinning of peach [8]. In some crops, flowering density can correlate with crop yield [9]. Diago *et al.* developed a computer vision-based system to count the flowers of grapevine inflorescence for grape yield estimation [10], achieving a coefficient of determination (\mathbb{R}^2) of 0.8 in correlation with manual counting. Aggelopoulou *et al.* utilized image processing techniques to predict apple orchard yield by detecting the flower density (white pixels) [11]. In both studies, a black screen (i.e., cloth) was used to provide uniform background and better colour contrast.

Previous work was largely based on colour features of flowers. However, mango flowers are inconspicuous, borne on inflorescences (panicles), with colour varying through the flowering season. The management task is to assess the percentage of shoot terminals that become reproductive (ie. produce inflorescences) rather than count individual flowers per se. Automated assessment of extent of flowering has management value to:

- Inform harvest timing of individual mango trees (given order of tree flowering);
- Inform crop agronomy (e.g. control of variable rate sprayers, with no chemical spraying for nonflowering trees);

Colour feature based segmentation is susceptible to varying light condition and imaging noise, as well as difficult to differentiate objects with similar colours. Therefore, a refinement segmentation method is proposed, utilizing both a color based segmentation and a support vector machine (SVM) [12] based classification with speeded up robust features (SURF) to identify false segmentation.

II. MATERIALS AND METHODS

A. Overall Approach

The manual assessment of mango flowering involves assessment of the proportion of branch terminals with reproductive growth (i.e., presence of panicles). However, the vegetative terminals are difficult to detect by machine vision based methods. Alternatively, we initially assumed the number of pixels within a canopy associated with flowers or leaves to be positively correlated to the proportion of the terminals in reproductive growth. Hence, the proposed method targeted segmentation of flowers and leaves from the input images, and investigated the correlation between the ratio of flower and leaf pixels and human assessment. The flowchart of the proposed method is shown in Figure 1.

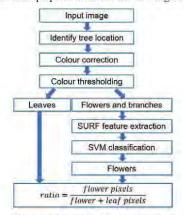


Figure 1. Flowchart of proposed method

B. Site Description, Image Acquisition and Pre-processing

Images were collected from a commercial mango (cultivar Kensington Pride, KP) orchard located at Kabra (23°31'03"S, 150°25'33"E), Central Queensland, Australia. Tree spacing was 4 m between trees and 3 m between rows. Sixteen trees were chosen for observation, with trees varying in the extent of flowering from nil to flowering of every terminal on the first day of imaging. Images were acquired from both sides of each tree at night (between 19:00 to 20:00) to reduce background issues. Images were collected weekly for five consecutive weeks (from July 29 to August 26, 2016) to track a flowering period. Hence, 32 images were acquired each week and finally 160 images were collected.

Images were acquired using a Canon DSLR EOS 50D (2352×1568 pixels) with a wide-angle 10~18 mm lens operated at 10 mm to cover the whole canopy of each tree. An external flash (Canon Speedlite 430EX II) was used for illumination. The distance between the camera and tree base was maintained at 2 m. The camera was using 'Auto mode' so f-stop, shutter and ISO speeds varied. In addition, sampled images varied in angle and light condition between weekly sampling events. Two sample images are shown in Figure 2.

At preprocessing stage, a symmetric Gaussian lowpass filter with size of 9 pixels and standard deviation of 0.5 was used to filter out high frequency imaging noise and to slightly expand the inflorescence image, to improve detection. The parameters for the Gaussian filter were obtained empirically.





(b)

Figure 2. Two sampled images: (a) medium and (b) low flowering trees.

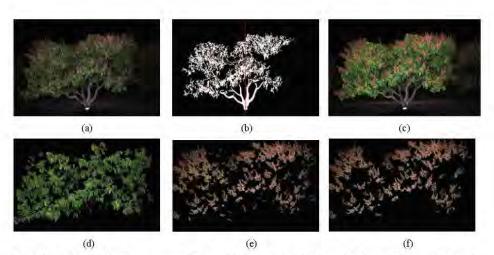
C. Identify Tree Location

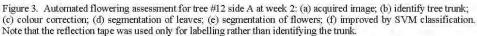
In the mango orchard, some tree canopies were clearly separated from their neighbors while others were not. In night imagery, the background is dark and this facilitates the separation of a target canopy from others. However, when the canopies of two neighboring trees were heavily overlapping, a boundary had to be imposed on the image to locate a valid assessment area per tree. This estimate was based on tree trunk localization and tree spacing. It was observed that the canopy of leaves for most the trees were rectangles with horizontal length being around 1.5 times of canopy height. An algorithm was developed to identify the tree trunk, and the target canopy delimited within a rectangle using the trunk as central axis.

The tree trunk was identifiable in images as the trunk was well illuminated and high grayscale values. A global image threshold using Otsu's method was applied to the grayscale image to obtain a binary image such as Figure 3(b). A search algorithm for a white rectangle of $a \times b$ pixels was used to seek the tree trunk. 'a=20' denoted the desired height and '30 $\leq b \leq 100'$ denoted the trunk width. The proposed algorithm could identify most the trunks including some special cases where the mango tree had two trunks. When the algorithm failed, the central line of the image was regarded as the trunk location.

D. Colour Correction

Tree leaves varies in colour from pinkish-red when young to dark-green or yellow-green when mature. Flowers also vary in colour, from light red to dark pink. Lighting





condition is another factor which may damage colour fidelity. Overexposed objects appear white, whereas underexposed objects appear dark. These factors degrade the performance of sequent colour based segmentation methods.

Several colour correction algorithms have been proposed for stereo [13] and multi-view [14] imaging to increase the similarity of pairing points. In these applications, one image was used for reference, and the remainder corrected region by region by referring to the reference. A global colour correction usually achieves worse results. However, in the current study, several images were chosen to compute reference brightness and contrast values, and a global colour correction algorithm was adopted.

Ten well illuminated images were selected to calculate reference mean colour values and intervals in red (R), green (G) and blue (B) channels respectively. A preprocessing operation was performed on these images to remove background objects via a simple colour based thresholding method, such that only leaves, flowers and other objects with similar colour to leaves or flowers remained to participate in the calculation. Then, the overall brightness of all the images was adjusted by scaling the mean colour value to the reference, and the contrast adjusted such that 0.1% of data was saturated to the reference low and high intensities. An example after colour correction is shown in Figure 3(c).

E. Segmentation of Leaves and Flowers

Segmentation of mango leaves and flowers are difficult due to their non-uniform colour and irregular shapes. Leaf colour can be yellow-green, purple, deep green and copper dependent on their maturity and nutrition. Panicle colour changes from green at initiation, to yellow at maturity and then to pink at anthesis, while shape changes from a swollen bud through an 'asparagus' stage to a mature pyramidal panicle, with maximum visibility as flowers open. By the fifth week, fruits are set, while unpollinated flowers turn brown and abort.

In contrast with shape features, the colour features of leaves and flowers are more favorable to segmentation. There are many colour spaces that approach human colour perception, of which Hue-saturation-value (HSV) is a popular one. A combination of the three channels of H, S and V was adopted in the proposed study. As mango leaves mostly lie in the range of light-green to green and flowers reside in the range of yellow to red, hue is the main criterion for differentiation. The detection of green flowers is currently a limitation of the proposed method. S and V channels were used to remove dark and white background. The colour thresholds were manually fine-tuned and finally fixed as given in Table-I.

Target	Channel	Threshold
	Н	700~1500
Leaves	S	0.2~0.95
	V	0.2~0.95
	Н	00~600
Flowers	S	0.35 ~ 0.95
	V	0.35~0.95

TABLE I. THRESHOLDS FOR LEAVES AND FLOWERS SEGMENTATION

A mature flower cluster is around 6-40cm in length which corresponds to at least 300 pixels in the image. So, a

Appendix

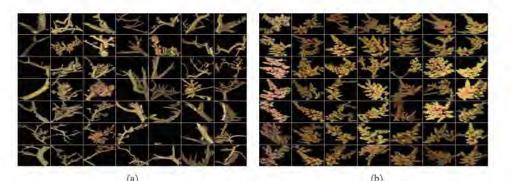


Figure 4. Selected labelled example images: (a) branches and leaves; (b) flowers.

region-size based filter was used to remove patches less than 200 pixels. As several panieles could be joined, there was no upper bound for region size filter. Two examples for leaves and flowers segmentation are shown in Figure 3(d, e).

F. SVM Classification

There were some detection errors in the colour-based segmentation results. The leaf canopy could be well separated from the background through colour thresholding, with some false positives (green weed) and false negative (non-green leaves). However, more commonly branches, trunks, dried leaves and land were mistaken for flowers. This affected the assessment result severely when there were few or no flowers. Therefore, a SVM based machine learning method was employed to further reduce the flower detection errors.

Three segmented images of canopies with low, medium and high levels of flowering were drawn from each week of sampling database for use in training (a total of 15 images, comprising 9.3% of total samples). The images were divided into many patches according to their connected components and labelled manually. Small neighboring patches were combined and overly large patches were divided, such that all patches had similar size. A resize operation was then applied to convert the patches into the same size (120×180 pixels). A few selected training images are shown in Figure 4. The speeded up robust features (SURF) [15] algorithm was used to extract visual words and a histogram based approach [16] was used to oncode the target image with visual words. Finally, a linear SVM with 10-fold cross validation was used to obtain the classifier and then predict if the test patch was flower or not.

III. EXPERIMENTS & RESULTS

The proposed method was implemented using Matlab R2016a. Colour thresholding used the hard thresholds given in Table I. Classification was performed based on the thresholding results. Fifteen images (397 branch patches) and 321 flower patches) were chosen for training and 145 images were used for testing. The classification parameters were

based on the Matlab default values, except the vocabulary size which was set at 500. The expected classification accuracy as around 86.6%.

A. Human Assessment

In order to validate the performance of the machine vision algorithm, all images were assessed by two human operators, in terms of the proportion of terminals with and without panieles in each image. Because many terminals were occluded by leaves or flowers and difficult to recognize, the results of the two assessors were differed up to 20%. The two scores were averaged for comparison to the machine vision results.

B. Segmentation Accuracy

The segmentation accuracies of different methods were compared. Images (n=32) acquired at the third week were chosen for observation since many trees were nearly fully flowering at that time. A manual operation was conducted using the tool of Microsoft Paint to separate the flowers precisely, which was then compared with the segmentation results from colour thresholding and SVM classification respectively.

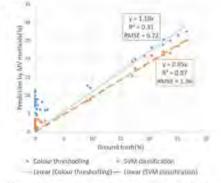


Figure 5. Proportions of flower pixels in total images

First, the results were evaluated as the proportion of flower pixels in the total image (Figure 5). The colour thresholding method overestimated the proportions of flower pixels in all images. The average overestimation rate was approximately 10%. This result was expected as loose colour thresholds were selected. The segmentation errors were improved by subsequent SVM classification, although it failed to identify all the branches or dried leaves and sometimes mistook flowers for branches. Specifically, the proportion of flower pixels for images of canopies with low flowering rate was still overestimated, but at lower rate; and the proportion for highly flowering images were underestimated. Overall, the R^2 was improved from 0.31 to 0.97.

TABLE II. CONFUSION MATRIX FOR SVM CLASSIFICATION

		Pred	ict class
	\sim	Flower	Non-flower
Actual	Flower	591	155
class	Non-flower	128	1121

The results were then evaluated in terms of the number of image patches. Colour thresholding recognized 1995 patches as flowers, of which 746 were true positive identifications, so the colour thresholding method achieved an accuracy of 37.4%. Afterwards, the SVM classification method correctly classified 1712 patches, achieving an accuracy of 85.8%.



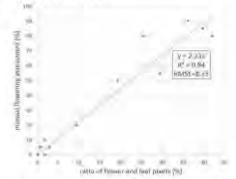
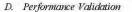


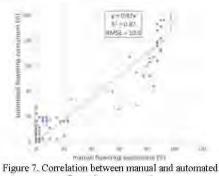
Figure 6. Correlation between proportion of reproductive terminals and ratio of flower and leaf pixels

The output of the proposed method is a ratio of flower and leaf pixels, which is assumed to linearly correlate with the proportion of reproductive terminals. In order to establish the relationship between automated and manual assessment methods, the 15 images chosen for SVM training were used as calibration data to build a linear correlation model (Figure 6, coefficient of determination (\mathbb{R}^2) of 0.94 and root-meansquare error (RMSE) of the residuals of 8.73). Therefore, equation (1) was used to map the ratio of flower and leaf pixels calculated by the proposed method to a score, simulating human assessment,

y=2.23x (1)

The intercept is ignored since the model should pass through the origin, that is, there should be of nil flowering when there is no flower pixel. As the flowering is scored in percentage, y is bounded in the interval [0, 100]. Equation (1) also indicates that the tree is fully blooming if the flower pixels occupy more than 45% of the tree canopy.





flowering assessment Thereafter, all images were input to the proposed method to calculate the ratios of flower and leaf pixels.

method to calculate the ratios of flower and leaf pixels. Equation (1) was used to convert the ratios into automated assessment scores which were then compared to manual assessment results. We acknowledge that this automated flowering assessment method suffers from a drawback that when mango flowering enters into fruit set stage, flowers turn brown and drop. This change does not impact flowering panicle number, but results in decrease in flower associated pixels detected by the proposed method. Flower drop began from Week 4. Therefore, only image data from Week 1 to 3 were used for statistical analysis (Figure 7). The proposed method underestimated the actual flowering situation (regression slope of 0.87), which is consistent with the segmentation results. An \mathbb{R}^2 of 0.87 between manual and automated flowering assessment was achieved.

E. Application Significance

The proposed method has potential to provide flowering information to guide harvest timing decisions or manipulation of flowering. Figure 8 shows a record of flowering of the tree (No 12, side A). It reveals that the tree was fully blooming (Figure 8(b)) at Week 3, and started fluit set at Week 5 (Figure 8(c)) when the flower pixel count declined.

Appendix



Figure 8. Record of automated flowering assessment for tree #12 side A: (a) flowering assessment over 5 weeks; (b) full flowering at Week 3; (c) fruit set at Week 5 (zoom in to see small fruits).

IV. CONCLUSION

This paper presented an automated flowering assessment method, aiming to provide mango growers an efficient way to collect and analyze mango flowering information. The proposed method consisted of three major steps: colour correction, colour thresholding and SVM based classification. The colour correction normalized the sampled images to the similar levels in terms of brightness and contrast, enabling a colour thresholding with fixed thresholds to be applied to obtain a rough segmentation of flowers and leaves. Then, a SVM based classification was employed to further refine the segmentation accuracy. The experimental results demonstrated that the proposed method could yield objective assessment results in good correlation with human subjective assessment.

ACKNOWLEDGMENT

Support from a CQU ECF Grant, the Centre for Intelligent Systems and Multiscale monitoring HIA project ST15005 is acknowledged. The authors would like to thank Mr. Tim & Mrs. Jenna Keogh for allowing access to their mango farm for data collection.

REFERENCES

- F. Ramírez and T. L. Davenport, "Mango (Mangifera indica L.) flowering physiology," Scienzia Horticulturae, vol. 126, pp. 65-72, [1] 2010.
- T. L. Davenport, "Processes influencing floral initiation and bloom: the role of phytohormones in a conceptual flowering model," *HortTechnology*, vol. 10, pp. 733-739, 2000 [2]
- S. Oosthuyse and G. Jacobs, "Flowering synchronization of "Sensation" mango trees by winter pruning," in *P International Marga Symposium* 455, 1996, pp. 422-430.
 P. Ding and K. B. Darduri, "Morphology of Chok Anan mango flower grown in Malaysia," *African Journal of Agricultural Research*, vol. 8, pp. 1877-1880, 2013. [3]
- [4]
- J L. Guardiola, "Overview of flower bud induction, flowering and [5]
- J. L. Guardiola, "Overview of lower but induction, itowering and fruit set," in *Proceedings of Citrus Flowering and Fruit Short Course*, University of Florida, 1997, pp. 5-21.
 T. Yeshitela, P. Robbertse, and P. Stassen, "Effects of pruning on flowering, yield and fruit quality in mango (Mangifera indica)," *Animal Production Science*, vol. 45, pp. 1325-1330, 2005. [6]
- [7] M. B. Duke, C. Esteban, H. Ryan, and F. Esmaeil, "Peach flower monitoring using aerial multispectral imaging," 2016 ASABE Annual International Meeting, pp 1, 2016.

- M. Nielsen, D. C. Slaughter, and C. Gliever, "Vision-based 3D peach tree teconstruction for automated blossom thinning," I. Transactions on Industrial Informatics, vol. 8, pp. 188-196, 2012.
- E. Stover, "Relationship of flowering intensity and cropping in fruit species," *HortTechnology*, vol. 10, pp. 729-732, 2000. [9]
- [10] M. P. Diago, A. Sanz-Garcia, B. Millan, J. Blasco, and J. Tardaguila, "Assessment of flower number per inflorescence in grapevine by image analysis under field conditions," *Journal of the Science of Food and Agriculture*, vol. 94, pp. 1981-1987, 2014.
- [11] A. D. Aggelopoulou, D. Bochtis, S. Fountas, K. C. Swain, T. A. Gentos, and G. D. Nanos, "Yield prediction in apple orchards based on image processing," *Precision Agriculture*, vol. 12, pp. 448-456, 2021. 2011
- [12] N. Cristianini and J. Shawe-Taylor, An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, 2000.
- [13] Q. Wang, P. Yan, Y. Yuan, and X. Li, "Robust color correction in stereo vision," in 18th IEEE International Conference on Image Processing, 2011, pp. 965-968.
- [14] M. Panahpour Tehrani, A. Ishikawa, S. Sakazawa, and A. Koike, "Iterative colour correction of multicamera systems using corresponding feature points," *Journal of Visual Communication and Image Representation*, vol. 21, pp. 377-391, 2010.
- [15] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust. features (SURF)," *Computer Vision and Image Understanding*, vol. 110, pp. 346-359, 2008.
- [16] G. Csurka, C. Dance, L. Fan, J. Willarnowski, and C. Bray, "Visual categorization with bags of keypoints," in Workshop on Statistical Learning in Computer Vision, BCCV, 2004, pp. 1-2.

4043 4044

4046	Addendum
4047 4048	This section includes materials that provide additional information for some part of the thesis to incorporate reviewers' suggestions.
4049	
4050	
4051	
4052	
4053	
4054	
4055	
4056	
4057	
4058	
4059	
4060	
4061	
4062	
4063	
4064	
4065	
4066	
4067	
4068	
4069	
4070	
4071	
4072	
4073	
4074	

WEEK ENDING (FRIDAY)	JUL	JUL	10r	26 JUL A	2 AUG A	9 16 AUG AUG	6 23 JG AUG		30 6 AUG SEP	P 36P	3 20 EP SEP	0 27 EP SEP	7 4 EP OCT	T OCT	1 18 17 OCT	т 25 т ост	T NOV	v NOV	15 NOV	22 NOV	29 NOV	6 DEC	13 DEC	20 DEC	27 DEC	3 JAN	10 JAN	17 JAN	24 JAN	31 JAN	7 FEB	14 FEB	21 FEB	28 FEB	6 MÅR	13 MAR	20 MAR	TOTAL
CLASS 1 TOTAL	-	1	ß	9	8	5 1	10 16	6 31	1 54	4 62	2 83	3 127	27 240	0 254	4 364	4 397	230	0 415	481	466	358	289	231	286	370	247	152	106	131	35	22	27	29	25	M	м	-	5700
CLASS 2 TOTAL	0	0	+	м	m	M	5 11	1 20	0 32	2 42	2 48	8 82	2 143	3 152	2 173	3 202	2 129	9 140	146	102	107	122	107	81	89	59	26	13	23	6	7	12	H.	H.	÷	÷	0	2148
OTHER TOTAL	0	-	0	-	R	5	9 2	8	3 15	5 20	11 0	1 12	2 24	4 19	11 6	7 21	7	16	2	÷	+	1	10	10	0	0	IJ	1	0	16	0	0	- 4	0	0	6	0	266
TOTAL VOLUME* (7 KG TRAYS 000S)	-	7	ø	2	14	13 24	4 29	6 23	101 6	1 124	142	12 221	21 407	7 425	5 554	4 620	526	571	629	569	466	412	348	377	459	306	183	122	154	60	59	39	57	36	4	12	-	8114
DARWIN (2.8 million trays)	on trays	(s																																				
WEEK ENDING (FRIDAY)	5 JUL	JUL	19 JUL	26 JUL A	2 AUG A	9 16 AUG AUG	6 23 JG AUG	3 30 IG AUG	0 6 1G SEP	13 P SEP	3 20 EP SEP	0 27 EP SEP	7 4 EP OCT	T oct	1 18 5T OCT	25 T OCT	T NOV	v NOV	/ NOV	22 NOV	29 NOV	6 DEC	13 DEC	20 DEC	27 DEC	3 JAN	10 JAN	17 JAN	24 JAN	31 JAN	7 FEB	14 FEB	21 FEB	28 FEB	6 MAR	13 MAR	20 MAR	gure 1 gion a
KP CLASS 1	0.1%	0.1%	0.4%	0.5% 0	0.7% 0	0.4% 0.7	0.7% 1.3%	% 2.5%	5% 4.4%	% 4.7%	7% 5.8%	8% 8.8%	8% 14.1%	15.8%	8% 17.0%	0% 12.6%	% 7.4%	% 2.3%	6 0.5%	6 0.0%																		
R2E2 CLASS 1	0.0%	%0.0	0.1%	0.0% 0	0.2% 0.	0.0% 0.1	0.7% 0.0%	0.5%	5% 0.5%	% 2.5%	5% 6.1%		6.7% 23.8%	8% 7.3%	% 8.1%	% 35.6%	3.7%	% 1.7%	6 1.5%	.0.9%																		
CALYPSO CLASS 1												3.0	3.0% 7.3%	% 10.7%	7% 21.4%	1% 32.5%	5% 25.0%	%																				
VOLUME* (7 KG TRAYS 000S)	-	2	w	5	14	13 2	24 29	6 23	101	1 124	4 142	12 221	21 405	5 395	5 477	7 502	2 216	83	8	w	+	+																
KUNUNURRA (150 thousand trays)	thousa	and tra	(s)																																			
WEEK ENDING (FRIDAY)	JUL	JUL	10L	26 JUL A	2 AUG A	9 16 AUG AL	16 23 AUG AUG	3 30 G AUG	0 6 IG SEP	P SEP	3 20 EP SEP	0 27 EP SEP	7 4 EP OCT	T OCT	1 18 CT OCT	25 T OCT	T NOV	V NOV	/ NOV	22 NOV	29 NOV	6 DEC	13 DEC	20 DEC	27 DEC	3 JAN	10 JAN	17 JAN	24 JAN	31 JAN	7 FEB	14 FEB	21 FEB	28 FEB	6 MAR	13 MAR	20 MAR	RED -
KP CLASS 1												1.0	0.1% 0.2%	% 9.1%	% 24.5%	5% 8.6%	% 14.8%	3% 15.3 %	% 23.7%	% 3.6%																		For
R2E2 CLASS 1												0.0	0.0% 0.0%	% 0.0%	17.3%	3% 9.1%	% 34.8%	3% 34.6%	% 2.3%	1.9%																		eca
HG CLASS 1																							14.3%	42.9%	42.9%													st w
VOLUME* (7 KG TRAYS 000S)												0	•	80	28	14	5	33	30	4	0	0	۰	-	-													veekl
KATHERINE (1.8 million trays)	illion tr	(sve																												-								y vol
WEEK ENDING (FRIDAY)	JUL	12 JUL	19 JUL	26 JUL A	2 AUG A	9 16 AUG AL	16 23 AUG AUG	3 30 IG AUG	0 6 IG SEP	13 P SEP	g 20 EP SEP	0 27 EP SEP	7 4 EP OCT	T OCT	1 18 5T OCT	25 T OCT	T NOV	V NOV	/ NOV	22 NOV	29 NOV	6 DEC	13 DEC	20 DEC	27 DEC	3 JAN	10 JAN	17 JAN	24 JAN	31 JAN	7 FEB	14 FEB	21 FEB	28 FEB	6 MAR	13 MAR	20 MAR	ume
KP CLASS 1													0.3%	% 3.0%	% 6.4%	%6.8 %1	% 13.1%	% 25.6%	% 24.4%	% 9.0%	9.4%																	of t
R2E2 CLASS 1																9.2%	% 30.2%	2% 28.4%	% 11.2%	6 7.9%	7.9%	5.3%																rays
CALYPSO CLASS 1																	8.9%	% 12.1%	6 28.6%	% 28.6%	5 21.8%																	s dis
HG CLASS 1																	3.8%	% 3.8%	6 16.0%	6 43.1%	6.3%	25.2%	%61															pat
VOLUME* (7 KG TRAYS 000S)													3	22	2 46	88	239	9 373	430	321	226	lin	12	9														ched
BOWEN/BURDEKIN (1.6 million trays)	N (1.6 r	million	trays)																																			fror
WEEK ENDING (FRIDAY)	JUL	12 JUL	19 JUL	26 JUL A	AUG A	9 16 AUG AL	16 23 AUG AUG	3 30 IG AUG	0 6 IG SEP	13 P SEP	3 20 EP SEP	0 27 EP SEP	7 4 EP OCT	T OCT	1 18 17 OCT	25 T OCT	T NOV	v NOV	/ 15 NOV	22 NOV	29 NOV	6 DEC	13 DEC	20 DEC	27 DEC	3 JAN	10 JAN	17 JAN	24 JAN	31 JAN	7 FEB	14 FEB	21 FEB	28 FEB	6 MAR	13 MAR	20 MAR	n the
KP CLASS 1																	0.7%	% 3.2%	6 12.7%	6 20.8%	6 19.0%	23.2%	11.5%	3.0%	3.6%	2.4%												reg
R2E2 CLASS 1															0.6%	3% 2.3%	% 5.1%	% 13.1%	% 9.7%	5 14.1%	10.9%	13.5%	13.2%	9.8%	7.1%	0.8%												ion
HG CLASS 1																						14.3%	9.5%	19.0%	19.0%	19.0%	19.0%											acr
VOLUME* (7 KG TRAYS 000S)															4	16	40	114	148	233	222	291	217	122	94	41	20											oss a
MAREEBA/DIMBULAH (1.8 million trays)	LAH (1.	8 millik	on tray	(s							-																											ll var
WEEK ENDING (FRIDAY)	5 JUL	JUL	JUL .	26 JUL A	2 AUG A	9 16 AUG AL	16 23 AUG AUG	3 30 IG AUG	0 6 IG SEP	P SEP	3 20 EP SEP	0 27 EP SEP	7 4 EP OCT	T oct	1 18 T OCT	25 T OCT	T NOV	V NOV	/ 15 / NOV	22 NOV	29 NOV	6 DEC	13 DEC	20 DEC	27 DEC	3 JAN	10 JAN	17 JAN	24 JAN	31 JAN	7 FEB	14 FEB	21 FEB	28 FEB	6 MAR	13 MAR	20 MAR	ieties
KP CLASS 1																			0.4%	6 0.5%	2.3%	9.2%	16.3%	20.9%	24.3%	20.1%	5.8%	0.2%										5)
R2E2 CLASS 1																				2.1%		3.0%	2.8%	30.4%	39.9%	9.8%	11.5%	0.6%		0.0%								

Figure 1-1. 2019/2020 Mango crop forecast- dispatch to the market (*BLACK - Actual weekly volume of trays dispatched from the

0.8%

%0.0 10.0% 13

%0.0 10.0% 4

16.4% 15.9% 15.1% 36

15.9% 23.4% 39

5.1% 10.0% 29

21.2% %0.0 154

15.5% 9.7%

1.7% 122

0.0% 265

3.0% 364

244

611 69

11 ø ю

VOLUME* (7 KG TRAYS 000S)

OTHER CLASS 1 KEITT CLASS 1 HG CLASS 1

CALYPSO CLASS 1

9.7% 19.4% 8.6% %0.0 60

25.8% 16.0%

16.0%

20.0% %0.0 0.7% %0.0 163

20.0% 10.3%

18.0% 25.2%

10.0%

26.8% 21

-

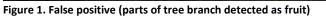
4104 **Representative images showing the failure detection by Mango-YOLO.**

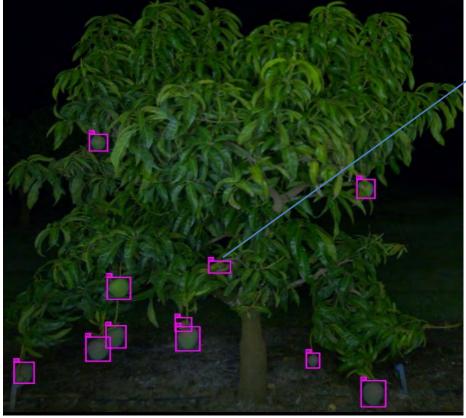




True positive = 25 False positive = 1 False negative = 0 Precision = 0.96 Recall = 1 F1 = 0.98

4105 4106



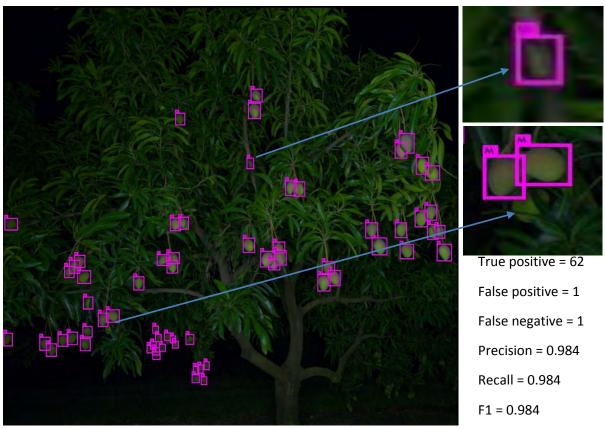




True positive = 10 False positive = 1 False negative = 0 Precision = 0.96 Recall = 1 F1 = 0.979

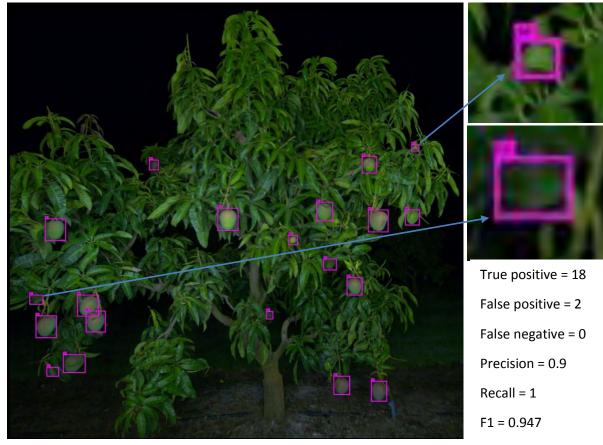
4107 4108

8 Figure 2. False positive (parts of tree branch detected as fruit)



4109 4110

Figure 3. False positive (parts of tree branch detected as fruit) and false negative (fruit not detected)



4111 4112

Figure 4. False positive (parts of tree leaves detected as fruit)

4113	
4114	
4115	
4116	
4117	
4118	
4119	
4120	
4121	
4122	THANK YOU
4123	
4124	
4125	
4126	
4127	
4128	
4129	
4130	
4131	
4132	
4133	
4134	
4135	
4136	
4137	
4138	