ON TREE GUAVA FRUIT DETECTION AND YIELD ESTIMATION

Gift Knowledge Wuzor, Nancy Chinyere Woods

Abstract— Fruit detection is a branch of image processing that aims to automatically detect various fruits, despite their varied nature. Guava fruits are peculiar in nature since the fruits closely resemble the leaves even when ripe and can be easily unnoticed, while on the tree. Unfortunately, these fruits that are rich in Vitamin C are easily perishable. Manually detecting the fruits on the tree to estimate yield can be cumbersome and labour intensive. This work developed a model for automatic detection of on-tree guava fruits and estimation of guava fruit yield. Digital images of on-tree guava fruits were first pre-processed to remove noise and reduce computational cost. K means clustering algorithm was then used to segment the fruit region from the background in the pre-processed images. Watershed Segmentation was used alongside some morphological operations, to separate joined and clustered fruits into individual fruit objects. The fruit yield was then estimated by counting the number of individual fruit object in each image using connected component labelling algorithm. A precision and recall of 94.5% and 84.4% respectively was obtained for fruit detection, while an accuracy of 92.4% was obtained for fruit counting. It was observed that the new system detected ripe guava fruits correctly but missed some unripe guava fruits.

Index Terms— Image Segmentation, watershed, fruit detection, k-means clustering, fruiting counting, guava fruit

1 INTRODUCTION

Image processing enables researchers to get more information from a digital image. The application of image processing in agriculture has increased tremendously in recent years, since it provides substantial information about the nature and attributes of the produce, reduces costs, guarantees the maintenance of quality standards and provides useful information in real time. Fruit detection is a branch of image processing that has gained attention over the years, due to the application of computer vision in robots that work in the agricultural sector, especially to harvest produce.

Guava tree, scientifically known as *psidium guajava L.*, is a tropical tree which is grown essentially for its fruit. The fruit is fleshy and can take the shape of ovoid, oblong or round. Guava is widely cultivated in Africa, and incorporated into agroforestry system in India. [1]. The outer skin of guava fruits are either rough or soft and with varying thickness and are usually green prior to maturity but may be either yellow, green or maroon when ripe. [2]. This peculiarity of some ripe guava fruits makes the fruit closely resemble the leaves and can be easily unnoticed, while on the tree. Unfortunately, these fruits that are rich in Vitamin C are easily perishable.[3]. It is therefore needful for growers to inspect the fruits on tree to determine their ripeness on time and estimate the yield in order to plan the market. This however cannot be done without first detecting the fruits. Farmers usually employ manual means to detect their guava fruits and estimate the yield which are time consuming and labour intensive. This work presents a system for automatic detection and yield estimation of on-tree guava fruits.

The first part of the work contains the review of some related literatures. The second part gives a detailed description of our methodology. Part three shows our results and part four contains the conclusion and recommendation.

2 RELATED WORKS

Much work has been done by researchers towards fruit and vegetable inspection systems. These works includes fruit detection, yield estimation, fruit quality evaluation and sorting. For instance, for citrus fruits, Blasco *et al.* [4] used Bayesian discriminant analysis to segment fruits from their backgrounds in their work which was geared towards evaluation of the efficiency of machine vision techniques for online estimation of apples. Features such as size, colour, stem location and blemish detection were evaluated. They obtained an accuracy of 96% for size estimation and an accuracy of 86% for blemish detection.

Sudkahara *et al.*, [5] developed a model for sorting, grading of apples fruits online using features such as colour shape and size. The images used for their work was captured in RGB colour space using CCD camera and frame grabber card. The images were then converted to HIS colour model and analyzed using some advanced image processing algorithms. They achieved an accuracy of 98% using median density of the Hue component of the HIS model as the grading criterion. A model for apple grading was proposed in [6]. The model involved steps such as colour

- *Wuzor Gift Knowledge is a Postgraduate Student of Computer Science in the University of Ibadan, Nigeria.. E-mail: wuzor.gift@ust.edu.ng*
- *Nancy Chinyere Woods is a lecturer in the department of Computer Science, University of Ibadan.. E-mail: chyn.woods@gmail.com*

classification, defect segmentation and stem recognition. The model was tested on the Golden Delicious and jobagold varieties of apple fruits, and a classification rate of 78% and 72% was obtained for both Golden delicious and Jobagold apples, respectively. A method for detecting and counting mature and immature on-tree papaya fruits in images was developed in [7]. The authors used Texture Analysis, Morphological Operations and Randomized Hough Transform to carry out their work. They were able to obtain an accuracy of 89.2% for fruit detection and an accuracy of 100% for counting of mature and immature fruits. In [8], a method for defect segmentation based on hyperspectral images was presented. Their goal was to develop a technique for early detection of Penicillium fungi on citrus fruits using mango as a case study. The images were classified using Artificial Neural Networks and Decision Trees and an accuracy of 98% was achieved. For orange fruits in particular, Bama *et al*., [9] developed a technique for inspecting orange fruits using texture and colour features. Histogram based segmentation was adopted in the segmentation of the input images to identify defective and non-defective regions in the image. Texture features were extracted using 3D co-occurrence distribution and sum of squared distance was calculated between texture features of test and training data. They obtained a classification accuracy of 93%. Li, Rao and Ying [10], proposed a technique for common defects detection in oranges using Principal Component Analysis and band ratio with a simple threshold method. They used hyperspectral imaging system to acquire the reflectance images from the orange samples in the spectral region 400 and 1000nm. The hyperspectral images were evaluated using PCA to identify the defect region in the image. This approach performed well except that it was unable to differentiate the step from the defect regions. To overcome this, representative regions of interest (ROIs) reflectance spectra of samples with different types of skin conditions were visually analyzed. The researches revealed that a two-band ratio image could be used to differentiate stems from defects effectively.

3 METHODOLOGY

This section contains an overview of the methodology used in this work. It also explains in detail each step carried out in the methodology. The methodology has five major steps as shown in figure. 1.

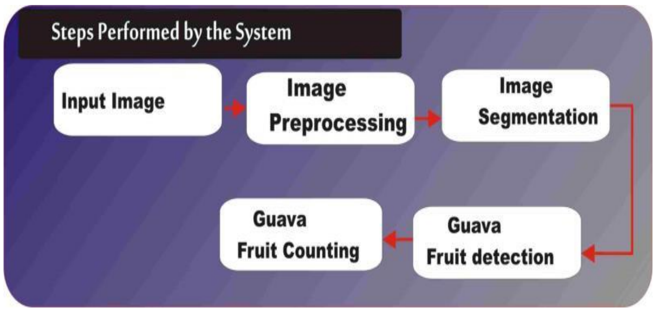


Figure 1: Methodology

The system accepts an on-tree image of guava fruits as input. Image pre-processing involved image resize, noise removal, shadow removal and image blurring; these were first carried out on the input image. The image was then converted from RGB colour space to L*a*b colour space and segmented using k-means clustering algorithm which separates the fruit image into fruit region, leaf region and background. The cluster with the fruit region, derived after k-means clustering, was converted to binary image. Morphological operations such as image filling and morphological open, image dilation and image erosion operations were also carried out on the binary image to fill up small holes and remove isolated pixels which are too small to be considered as fruit regions.

The common challenge of most on-tree fruit recognition systems is that of occlusion which occurs when leaves or fruits cover some part of a fruit region. Another problem is the cluster of fruits in a fruit image. When images of cluster fruits are converted to binary images, they appear as single fruit objects.

To overcome this challenge watershed algorithm was applied on the binary image to separate clustered fruits into separate fruits. After this, the individual components representing fruit regions were then counted using component labelling and region properties. Each region in the image identified as a fruit region was then localized using rectangular boundary box.

3.1 Image Dataset

A total of Eighty (80) on-tree guava fruit images, fetched from the internet are contained in the dataset. Some of the images in our dataset were of different sizes, had varied brightness, and contained noises and shadows, therefore they needed to be pre-processed. Some of the images in the dataset are presented in Figure 2.

Figure 2: Sample images in dataset

3.2 Image Pre-processing.

Image pre-processing entails removing low frequency background noise in images, normalizing the intensity of each pixel in the image, removing reflections and masking portions of the images. It is the technique for enhancing image data before computational processing. [11]. The Image pre-processing techniques applied in this work include; image resize, image noise removal and image shadow removal. The larger the size of an image, the more the computational time required to process the image and the more the memory capacity required to save it. To save the cost of processing and image storage, the images were first resized to a fixed size of 620 x 480 pixel resolution using image resize function in MATLAB. Then median and Gaussian blur filters were used to remove noises and smoothen images. To remove shadows in the images used for this work, the images were first converted from RGB colour space to HIS colour space. Then the saturation component of the HIS colour space, S, was extracted and normalised. All saturation component pixels with values greater than 0.6 were replaced with the value 0.6, while those having lesser values retained their normal values. The image was then converted again from HIS colour space to RGB colour space.

3.3 Image Segmentation

In this work, colour segmentation using K-means clustering algorithm was used to separate regions of images of on-tree guava fruits into leave regions, fruit regions, and backgrounds. K-means clustering can be seen as a segmentation algorithm and as well as a classification algorithm. It belongs to a type of classification algorithm called unsupervised classification. Unsupervised classification algorithms, unlike their supervised counterpart, does not require training of dataset but classifies images based on certain parameters.

K-means clustering algorithm seeks to segment and classify image pixels into clusters based on their distant similarity. In this work, the RGB image obtained after pre-processing was first converted to L*a*b colour space. The L channel represents the luminosity of the image whereas the 'a' and 'b' channels carry the chromaticity property of the image. After the conversion from RGB to L*a*b colour space, the L*a*b image was then separated into its constituent channels. The 'a' and b channels representing the colour component of the image were used as input data for the k-means clustering.

The image was separated into four (4) clusters, where one of the clusters contained the fruit region. The cluster containing the fruit region was then saved and used for further processing.

3.4 Fruit Region Detection

To detect the individual guava fruits in an image, the cluster containing the fruit region, obtained after segmentation using K-means clustering, was first converted to a grayscale image and then to binary image using the inbuilt functions of MATLAB rgb2gray() and im2bw respectively. Morphological operations such as:- image filling, erosion and dilation were performed on the binary image. Image filling is carried out to fill up tiny holes within the binary image while image erosion and image dilation are done to remove smaller white regions which are too small to be considered as guava fruits.

In most on-tree fruit detection systems, fruits tend to join together or form a cluster such that when segmented into binary images, they appear as a single component in the binary images. This makes it difficult for counting to be carried out. For instance, four fruits that are joined together could be recognized by the system as one fruit object, making the fruit recognition model inefficient. Most algorithms try to use image erosion and dilation operation to separate joined fruits, but that is never sufficient to separate fruits joined closely.

To solve this problem, after image erosion and dilation operations have been carried out on the binary image, we used watershed segmentation [14], shown in Figure 3, to separate joined fruit objects into constituent objects. The combination of algorithms based on watershed transform for image segmentation is called Watershed Segmentation.

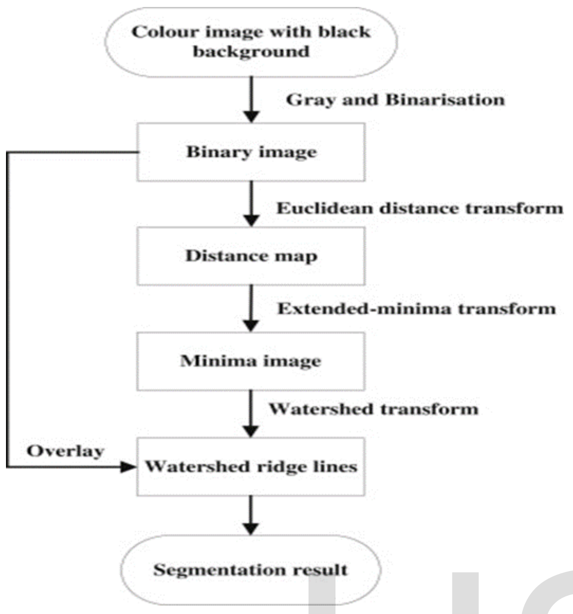


Fig. 3 Watershed Segmentation [12]

The term watershed metaphorically refers to the geological watershed, which is a divide that separates adjacent catchment basins. The watershed transform decomposes an image into catchment basins and then separate basins from each other. Applying watershed transform to an image often results to a large number of smaller regions. This is known as the problem of over-segmentation. [14]. For this problem, watershed transform alone cannot segment an image to the taste of the user. It is most times combined with other algorithms in other for users to get their expected results. To enhance the performance of watershed transform we combined it with Euclidean Distance Transform and Extended Minima Transform. The Euclidean Distance transform converts our image into another image whose catchment basins are the objects we want to segment, in this case, Guava fruits. The distance transform of a binary image is the distance from every pixel to the nearest nonzero-valued pixel. In this one work, we applied the distance transform on the complement of our binary image, as it gave us better results than when we applied it directly.

After applying the distance transform on the binary image, we then applied the Extended Minima Transform. This algorithm helps to remove tiny local minima which results into smaller catchment basins and consequently over-segmentation in binary images. After the tiny minima has been removed, we then modify the distance transform so that no minima occur at the locations where the minima were removed. This is called "minima imposition" and is implemented using the inbuilt MATLAB function for minima imposition. The combination of these three transforms makes up our watershed segmentation as shown in Figure 3.

3.5 Fruit Counting

Connected component labelling algorithm [13] was used to count/estimate the number of individual guava fruits in each image. The algorithm accepts the binary image obtained after watershed segmentation as input and returns the number of individual 'objects' in the image. The Connected components labelling algorithm works by scanning an image (usually binary images or grey level images) pixel by pixel, grouping them into components based on pixel connectivity. Pixels of a connected component share similar intensity values (*V)* and are connected to each other. Once all groups have been determined, each pixel is labelled with a grey level or a colour (colour labelling) according to the component it was assigned to.

In this work, we applied 8-connectivity on the binary image. The connected components labelling operator scans the binary image by moving along a row until it comes to a point p (where p denotes the pixel to be labelled at any stage in the scanning process) for which *V*={1}. When this is true, it examines the four neighbours of p which have already been encountered in the scan. Based on this information, the labelling of p occurs as follows:

- If all four neighbours are 0, assign a new label to p, else
- If only one neighbour has *V*={1}, assign its label to p, else
- If more than one of the neighbours have *V*={1}, assign one of the labels to p and make a note of the equivalences.

After completing the scan, the equivalent label pairs are sorted into equivalence classes and a unique label is assigned to each class. As a final step, a second scan is made through the image, during which each label is replaced by the label assigned to its equivalence classes. Having obtained the number of individual fruits on the image, we place a rectangular bounding box on each fruit region in the image using the MATLAB function boundingbox().

4 RESULTS AND DISCUSSION

The model was evaluated first by comparing the detected fruit regions with a manually created ground truth dataset for the same set of images. A hit or true positive is recorded whenever a region identified as fruit region in the ground truth image is also identified by our algorithm as a fruit region. A false Hit or false positive is recorded whenever a region counted as fruit region by our model does not match any region identified as fruit region in our manually created ground truth data. A miss or false negative is recorded for any ground truth region for which no hit was obtained; that is, a fruit region that was not identified as such by our model. A Merge is recorded when more than one fruit region identified in the ground truth image is identified as a single fruit region by our model. The following results analysis focuses on guava fruit detection rates, counting rates and counting error rates.

4.1 Image Pre-processing Result

As stated in our methodology, each image was first pre-processed to remove noise, shadow and save computation time. Figure 4 shows the input image and the corresponding output image, after that image has gone through resizing, noise removal and shadow removal.

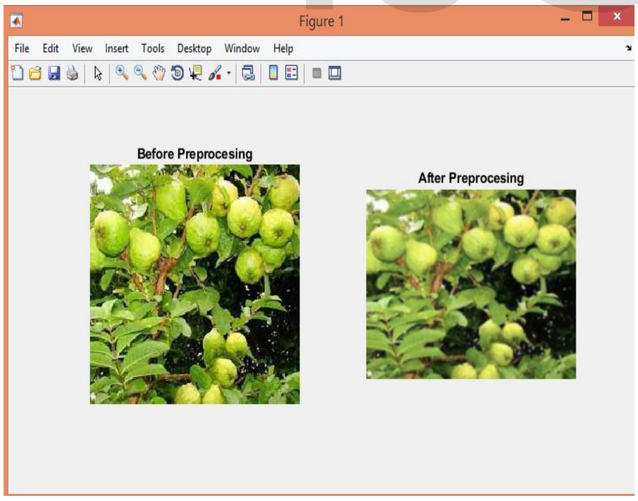
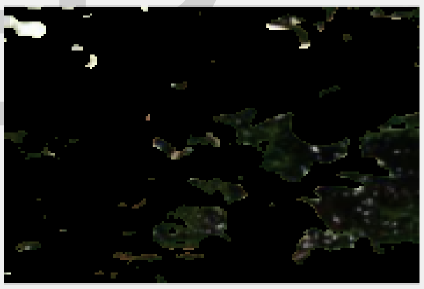
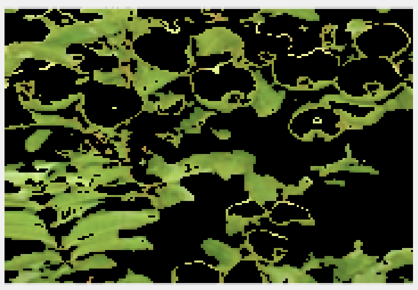


Fig. 4. Image pre-processing result

4.2 Image Segmentation Results.

Segmenting our images using *k* means clustering algorithm was successful and the best results was achieved when *k* was set to four (i.e four clusters) compared to when it was set to three (3). Figure 5(a-e) shows the results obtained by applying *k* means clustering on our pre-processed image. Figure 5 (a) shows the pre-processed image, figure 5(b) shows the cluster containing the background region, figure 5 (c) shows the cluster containing the leave area, figure 5(d) shows the cluster containing the stem regions of the image and finally, figure 5(e) shows the cluster containing the fruit region, which is our cluster of interest. K means algorithm was efficient in separating the fruit region from all other regions based on colour difference. However, some non-fruit pixels in the image that are of similar colour with the fruit regions were miss classified as fruit regions.

a	
b	
c	

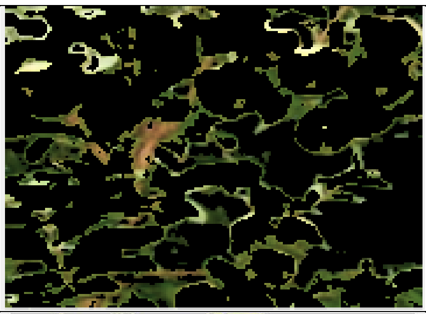
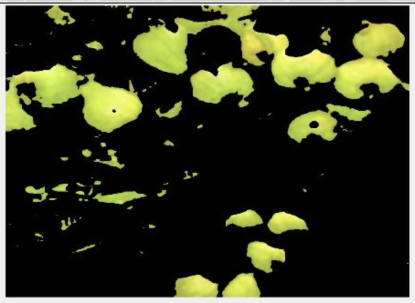
d	
e	

Fig. 5 (a –e): Segmentation results after applying k means clustering

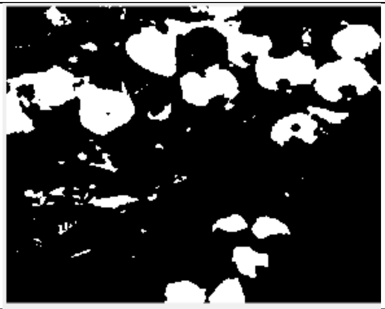
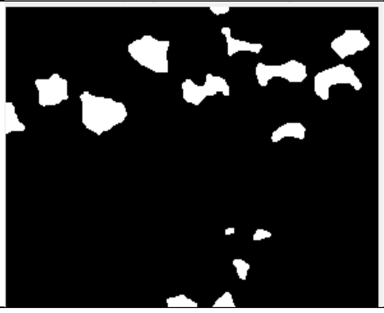
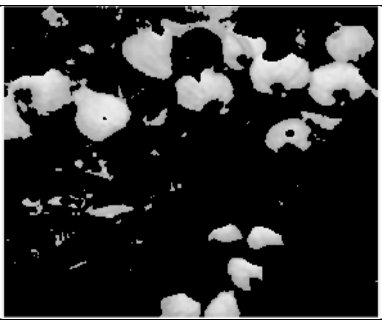
b	
c	

Fig. 6: Results of Watershed Segmentation

4.3 Image Segmentation Result

Figure 6(a-c) shows the result obtained after applying watershed segmentation on the cluster with the fruit region shown in figure 5e. Note that figure 5e was first converted to a grayscale image (figure 6a), then to a binary image (figure 6b), using functions in MATLAB as specified in our methodology. Figure 6(c) shows the result after applying watershed segmentation described in our methodology. From figure 6c we see clearly, that after the watershed segmentation, smaller regions in figure 6(b) which were not large enough to be considered as fruit regions were removed, likewise, joined fruit regions were also separated into individual fruit region.

a	

4.4 Fruit Detection Results

Figure 7 contains pictures of some of the detection results of our system. The fruit detection results were classified into Hits, Misses, Merges and False alarms as mentioned earlier. Figure 7(a) in particular is the final output of the test image in figure 6. From figure 7a, we see that the fifteen guava fruits in the image were correctly identified as fruits while a leaf area was wrongly identified as fruit region, giving us 15 hits and 1 false alarm. In figure 7(b), the two guava fruits were recognized as 1 fruit giving us 1 Merge and 2 hits. Out of the five guava fruits in figure 7(c) four were correctly identified resulting into 4 hits and 1 miss. In Figure 7(d) the two guava fruits were identified correctly but a leaf region was identified as a fruit region giving us 2 hits and 1 false alarm.

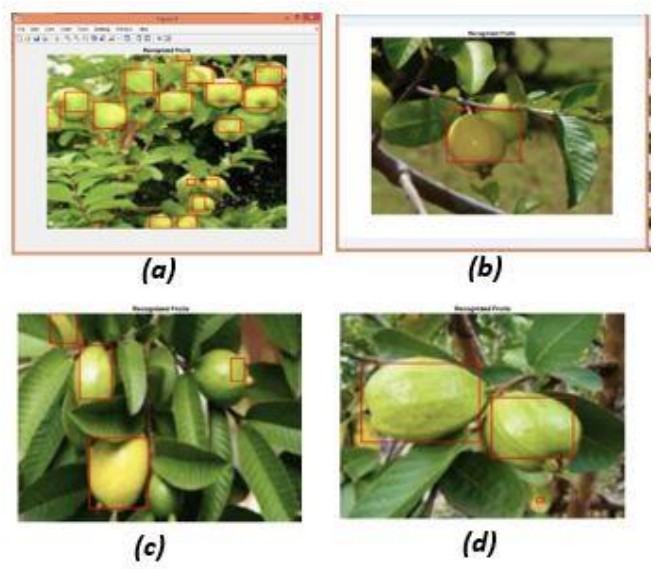


Fig. 7. Guava Fruit detection results.

4.5 Evaluation of Detection Results

Table 1 shows the summary of results obtained for on-tree guava fruit detection using our system. . The results were classified under four major headings as Hits, Misses, Merges and False alarms.

Table 1: Summary of results

Hits/True Positives	Misses/False Negatives	Merges	False Alarms/ False Positives
84.37%	15.625%	1.5625%	4.48%

Table 4.1 contains the percentage values after the results of the detection were classified into hits (true positives), misses (false negative), false alarms (true negatives) and merges. The formula used in the percentage analysis are:

$$\boldsymbol{T_P = \frac{\sum h}{\sum f} \times 100} \quad (1)$$

$$\boldsymbol{F_N = \frac{\sum m}{\sum f} \times 100} \quad (2)$$

$$\boldsymbol{T_N = \frac{\sum a}{\sum f} \times 100} \quad (3)$$

$$\boldsymbol{M_g = \frac{\sum m_s}{\sum f} \times 100} \quad (4)$$

Where :

T_P = Hit rate or rate of true positives. It is given by the sum of all hits in all our images divided by the sum of the total number of fruits in our images.

F_N = Miss rate or Rate of False negatives. It is given by the sum of all misses in all our images divided by the sum of the total number of fruits in our images.

T_N = Rate of False alarms or False positives. It is given by the sum of all false alarms in all our images divided by the sum of the total number of fruits in our images.

M_g = Rate of Merges. It is given by the sum of all hits in all our images divided by the sum of the total number of fruits in our images.

h	=	number of hits in each image
m	=	number of misses in each image
a	=	number of false alarms in each image
m_s	=	the number of merges in each image
f	=	the total number of guava fruits in each image.

4.6 Precision and Recall

Precision refers to the percentage of our results which are relevant and it tells us the percentage of guava fruits in our detected objects. While recall refers to the percentage of total relevant results correctly recognized by an algorithm, it tells us the percentage of detected guava fruits in all our images. After counting the misses, hits and false alarms in each results, we summed them up to get total hits (true positives), total misses (False negatives) and total false alarms (False positives). From the values obtained we calculated the precision and recall of our results.

Precision (P) is given by the formula

$$P = \frac{T_P}{T_P + F_p} \times 100 \quad (5)$$

The Recall (R) was calculated using the formula:

$$R = \frac{T_P}{T_P + F_N} \times 100 \quad (6)$$

We obtained a precision of **94.5%** and a recall of **84.4%** as shown in table 2.

Table 2 Precision and Recall

Precision	Recall
94.5%	84.4%

4.7 Automated Guava Fruit Counting Result Using Connected Components Labelling Algorithm.

To evaluate our automated yield estimation model using Connected Component Labelling Algorithm, we first counted the number of fruits in each on-tree Guava fruit image manually and then compared it to the results we got using our counting algorithm on the same set of images. The percentage error of the counting was defined as the percentage error between the number of guava fruits counted using the connected component labelling algorithm, and the number of guava fruit counted manually.

Percentage Error of Image= $\frac{A_c - M_c}{Mc} \times 100$ (7)

Where

A_c = number of guava fruit counted by our automated counting algorithm (i.e Automated count)

M_c = number of guava counted manually (i.e Manual count).

The accuracy (A) of our automated counting algorithm was estimated using the formula.

= $A = \frac{A_c}{Mc} \times 100$ (8)

We obtained an accuracy of 92.4% and percentage of 8.6% for our entire dataset

5. CONCLUSION

Fruit detection and yield estimation is an aspect of image processing that has been greatly applied to automate agricultural processes. This work was geared towards the detection and estimation of guava fruit yield in a guava tree via an image of the tree.

The new system was able to detect guava fruits on guava tree images as well as count the number of guava fruits in the image. It was discovered that the *k* means clustering gave the best results when the value of *k* was set to 4 (i.e when the image was segmented into four clusters). The watershed segmentation also gave the best result when enhanced with some morphological operation such as erosion and dilation. The model performed better in detecting fruits that were mature and ripe but immature guava fruits constituted the misses incurred. When compared with manual count, the performance of the connected component labelling algorithm used in counting the guava fruits was impressive.

REFERENCES

[1] CABI. 2013. "Wallingford, UK: CAB International | Feedipedia." http://www.cabi.org/isc (May 7, 2019).

[2] Morton, Julia Frances, and Curtis F. Dowling. (1987.) *Fruits of Warm Climates. Florida Flair Books Miami*. J.F. Morton.

[3] SELFNutritionData. (2010). "Guavas, Common, Raw Nutrition Facts &amp; Calories." https://nutritiondata.self.com/facts/fruits-and-fruit-juices/1927/2 (May 7, 2019).

[4] Blasco, J., N. Aleixos, and E. Moltó. (2003). "Machine Vision System for Automatic Quality Grading of Fruit." *Biosystems Engineering* 85(4): 415–23.

[5] P Sudhakara, A Gopal, R Revathy, K Meenakshi. (2009). Analysis of fruits using machine vision system for Automatic Sorting and Grading, *J. Instrum. Soc. India* 34 (4) 284-291.

[6] Vincent Leemans, Hugo Magein, Marie-France Destain. (2002). On-line Fruit Grading according to their External Quality using Machine Vision, Biosystems Engineering 83 (4), 397–404, doi:10.1006/bioe.2002.0131,*Journal of Automation and Emerging Technologies*, Belgium.

[7] Sethy Prabira Kumar, Jyoti Vihar, and Jyoti Vihar. (2016). "On-Tree Detection and Counting of Mature and Immature Fruit of Carica Papaya Using Image Processing Technique." 156(8): 16–21.

[8] Gómez-Sanchís Juan, José David Martín-Guerrero, Emilio Soria-Olivas, Marcelino Martínez-Sober, José Rafael Magdalena Benedito, José Blasco. (2012). "Detecting Rottenness Caused by Penicillium Genus Fungi in Citrus Fruits Using Machine Learning Techniques." *Expert Systems with Applications* 39(1): 780–85.

[9] B. Sathya Bama, T. Harinie, I. Janani Chellam, V. Abhaikumar, Sangeetha Raju (2011). 3D Colour Co-occurance texture features as tool to evaluate quality of fruits, *Journal of Scientific and Industrial Research*,Vol 70, pp. 912-917.

[10] Li, Jiangbo, Xiuqin Rao, and Yibin Ying. (2011). "Detection of Common Defects on Oranges Using Hyperspectral Reflectance Imaging." *Computers and Electronics in Agriculture* 78(1): 38–48. http://dx.doi.org/10.1016/j.compag.2011.05.010.

[11] Deshpande, Tejal, Sharmila Sengupta, and K S Raghuvanshi. 2014. "Grading & Identification of Disease in Pomegranate Leaf and Fruit." *International journal of computer science and information Technologies* 5(3): 4638–45

[12] Yao Qing, Liu Qingjie, Dietterich Thomas G, Todorovic Sinisa, Lin Jeffrey, Diao Guangqiang, Yang Baojun,Tang Jian "Segmentation of touching insects based on optical flow and NCuts," *Biosyst. Eng.*, vol. 114, no. 2, pp. 67–77, 2012

[13] André Ødegårdstuen (2019). Connected-component

labeling (https://www.mathworks.com/matlabcentral/fileexchange/45480-connected-component-labeling.html), MATLAB Central File Exchange. Retrieved May 14, 2019.

[14] Steve Eddins(2002). The Watershed Transform: Strategies for Image Segmentation. (https://www.mathworks.com/company/newsletters/articles/the-watershed-transform-strategies-for-image-segmentation.html). Mathworks Technical Articles and News Letters.