Robust Apple Detection in Orchard Environments Using Machine Vision and SVM

No.	Name	Project Contribution	Report Contribution	Signature
1	Devansh Kopra	**Pipeline Implementation:** Developed the core image processing logic (HSV conversion, Morphological ops) and main loop.	Authored **Section 2.1** (Algorithm Design) and **Section 2.2** (Image Processing).	
2	Tarun Dandekar	**Feature Engineering:** Implemented HOG feature descriptor extraction and managed data preprocessing constraints.	Authored **Section 1** (Introduction), **1.1** (Assumptions), and **1.2** (Literature Review).	
3	Gogulnath	**Performance Evaluation:** Coded the metric calculations (Precision, Recall) and generated performance graphs.	Authored **Section 3** (Results and Evaluation).	
4	Bhanu Neelam	**Classification Model:** Implemented the SVM classifier training and geometric filtering logic.	Authored **Section 2.3** (Machine Learning) and **Section 2.1.3** (Dataset).	

Contents

1 Introduction

The reliable identification and counting of fruits is essential for the implementation of precision farming because both yield prediction and the management of crop production systems depend upon it. The orchard environment, however, presents various visual complications when it comes to fruit detection, for example, the variable illumination levels and the presence of tree shadows. [1] Researchers studying apple yield data believe that natural light and the shading effects of leaves and tree branches significantly impact the ability to accurately count the number of apples growing within an orchard or farm location leading to missed detections of some apple fruit or miscounting the number of apples that actually exist. [2]

Figure 1: Sample images from the Fuji-Sfm dataset showing apples under natural orchard conditions.

This project develops a single integrated detection and counting algorithm that detects, count apples in orchard images reliably across a variety of apple cultivars and lighting conditions.Two datasets are used to evaluate the generalization of the system Fuji-SfM, Figure 1 which primarily includes images of red apples that were captured under relatively bright lighting conditions; and MinneApple, Figure 2 which contains images of green and yellow apples that tend to be primarily blended in with the foliage of the trees, making them more challenging to detect. [3] To address these challenges, a hybrid approach that combines machine learning and image processing is adopted.

Figure 2: Sample images from the datasets showing apples under natural orchard conditions. [4].

1.1 Assumptions

Images of apples in orchard environments during daytime construction will have enough detail to capture the image clearly. A small percentage of leaves and branches obstructing the fruit will likely obscure some apples in images, but an extreme amount (entirely obstructed by branches/leaves), taking photos at night, and extreme motion blur do not fall within the focus of this research. Apples are assumed to have a round, symmetrical shape around a center point and to be located in an area relative to the overall image dimensions. These assumptions depict realistic input conditions as they exist within most orchards, but maintain the problem scope as defined by the researcher's defined boundaries.

1.2 Literature Review

1.2.1 Image Processing

Fruit detection and yield estimation were traditionally done using methods that included color thresholding, shape analysis, and morphological techniques. These methods can be implemented using the ability to generate computer algorithms, and as such, are very attractive for agricultural information systems because they are both efficient and easy to understand by end users. While studies have demonstrated the ability of these methods to accurately identify fruit in controlled or simplified environments, in natural environments, fruit detection accuracy falls off significantly due to factors like changes in sunlight and background clutter. [1], In particular, orchard-based studies have reported that uncontrolled natural illumination and occlusion from foliage lead to missed detections and inconsistent fruit counts when using traditional vision-based approaches. [2]

1.2.2 Machine Learning Methods

In recent years, machine learning techniques have been frequently employed to solve fruit detection issues due to the limitations of traditional, strictly rule-based methods. Learning-based approaches rely on translating texture and shape; thus, they are able to differentiate between fruit and those objects that appear similarly to the fruit within the same environment (backgrounds) based solely on their colour. As a means of supporting the evaluation of higher quality and more reliable detection methodologies within particularly difficult fruit detection environments (i.e., green apples in thick foliage), benchmark datasets such as MinneApple have been created [3]. From the above developments, we are encouraged to explore the utilization of hybrid systems that combine machine learning with image processing for enhanced robustness.

2 Methodology

2.1 Algorithm design

2.1.1 Problem Formulation & Overview

It is hard to find and count apples in outdoor orchards because the lighting changes, tree canopies cast shadows, leaves block the view, and apples are the same colour as the plants around them. Colour thresholding and shape analysis are two common image processing techniques that work well in controlled environments but not so well in real-world situations. Even though pure deep learning methods are very powerful, they are like black boxes, which makes it hard to understand and reproduce results. This is very important in agriculture.

To overcome these constraints, we propose a hybrid algorithm that combines traditional image processing with machine learning. The method has two steps: (i) quickly finding candidate regions that are likely to have apples, and (ii) sorting these candidates to tell the difference between real apples and background elements like leaves and tree trunks. This two-stage framework is very efficient and very accurate, and it also makes it clear how features are extracted and decisions are made.

2.1.2 Algorithm Pipeline

This is achieved using an integrated detection algorithm Figure 3 that works in four successive stages, modeled after what can be called a "Scout and Judge" system:

Stage 1 : Stage one quickly identifies potential regions using color-space transformation and morphological operations. Images are converted from BGR to HSV, which separates color information from brightness, enabling reliable apple detection under varying lighting and shadows. Red apples remain within a hue range of 0–15 regardless of illumination. Pixels in this range are used to create a binary mask of candidate regions. Morphological opening and closing with a circular structuring element remove noise and refine apple regions. Connected-component analysis is then applied to the cleaned mask to generate axis-aligned bounding boxes, producing dozens to hundreds of candidate regions per image.

Stage 2: Instead of using raw pixel data, candidate regions are represented using the Histogram of Oriented Gradients (HOG) descriptor, a manually designed feature extractor that captures shape and texture information by focusing on edge gradient orientations. Gradients in the x and y directions are computed using the Sobel operator. Each region is divided into fixed, non-overlapping cells (e.g., 8×8 pixels), and a histogram of gradient directions is accumulated for each cell using 9 orientation bins (e.g., 0° to 160°). For robustness to illumination changes, overlapping blocks (e.g., 2×2 cells) are normalized. The normalized histograms from all blocks are concatenated to form a HOG feature vector of approximately 1,000–2,000 dimensions, effectively capturing edge distributions such as the smooth curves of apples compared to the rugged vertical edges of tree trunks.

Stage 3: A linear Support Vector Machine (SVM) with a linear kernel is used for the binary classification of candidate regions into apple or non-apple regions such as background, trunk, or leaves. The linear SVM with a linear kernel is used because it is efficient and prevents overfitting, which may occur when a nonlinear classifier is used with a small amount of data. The linear SVM computes a confidence level where a positive output classifies a region as an apple patch, and the magnitude of the output determines how confident the output is.

Stage 4: After classification by SVM, several filters further refine the detections. Aspect ratio filtering removes non-applicable objects by maintaining the bounding boxes within a small range, say from 0.4 to 2.5, which corresponds to the almost spherical shape of an apple. Area filtering removes detections that are either too large-for example, an entire tree-or too small to reliably constitute a detection. Confidence thresholding retains only those above a set confidence in SVM detection. This would allow tuning the balance between precision and recall-for example, only the one with $> 50\%$ confidence in their detection. Finally, Non-Maximum Suppression resolves overlapping proposals, where the highest-confidence detection is retained and all other detections with an IoU above a threshold are discarded, returning a final set of unique detections of apples.

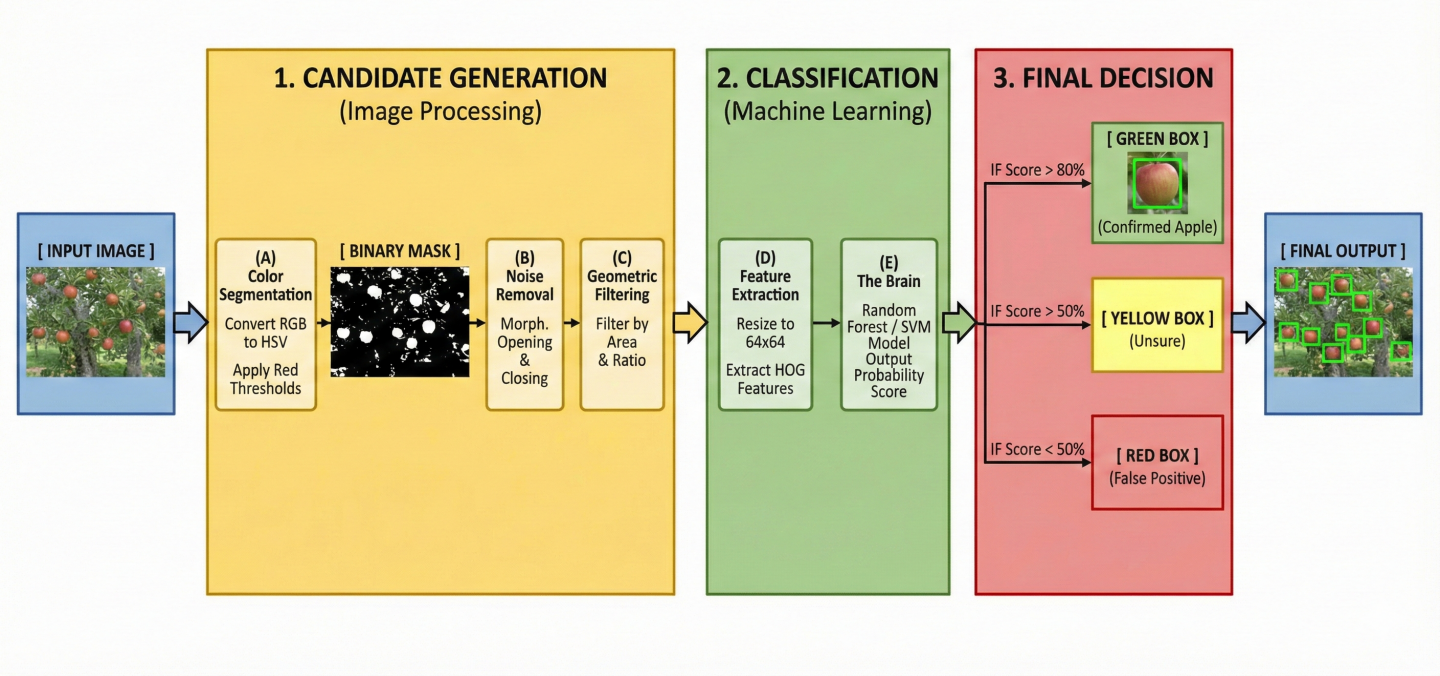

Figure 3: Hybrid Integrated Pipeline for Apple Detection

2.1.3 Dataset

Fuji-SfM Dataset

The Fuji-SfM dataset consists of about 288 high-resolution images of Fuji apple cultivars (red apples) taken in an open orchard environment under sunny conditions. This dataset is divided into 231 training images and 57 validation images. The main challenge here is the changing lighting condition; the apples that are in direct sunlight are bright red, while those which are in shadow are almost black. Annotations are given as polygon coordinates by tracing the exact outline of each apple. These are converted to axis-aligned rectangular bounding boxes by computing the minimum and maximum x and y coordinates encompassing each polygon.

MinneApple Dataset: Dataset This MinneApple training data for object detection comprises 670 challenging images of Granny Smith apples that are green versus Golden Delicious apples that appear yellow to green. These pictures individually feature 50-100 apples with substantial overlap among the fruit. The biggest issue here is that apples look very similar to the foliage with which they appear in the same picture, so HOG and SVM features were used instead of ones dependent on color. The ground truth annotations are given in YOLO file (.txt with normalized ground truth boxes) or JSON files with additional information.

Data Partitioning:

These datasets are divided into three disjoint sets:

Training Set: In this case, a total of 901 source images are presented, of which 231 come from Fuji-SfM and 670 come from MinneApple. Furthermore, based on these images, a total of 25,363 individual patch samples are obtained, consisting of 2,369 red apple samples from Fuji-SfM, 19,946 green/yellow apple samples from MinneApple, and 3,048 background/negative samples of leaves, trunks, and sky. Such a negative-positive imbalance is common in nature, given the class distribution of an orchard, which is approximately 22,315 for positive samples and 3,048 for negative samples.

Validation Set: In full resolution, coming from 57 source images in the Fuji-SfM folder validation images and annotations, which is used for the evaluation and calibration of hyper-parameters, thresholds, and post-processing constraints, including aspect ratio range, area bounds, and confidence thresholds.

Test Set: 20 source images high-resolution images sampled from the Fuji-SfM dataset that were held out fully during training and validation to ensure reported performance metrics truly generalize to unseen data.

2.2 Image processing

Colour Space and Illumination Robustness "BGR to HSV" is the preprocessing step that resolves the major issue of outdoor apple fruit detection, which is illumination variation. As opposed to BGR, where illumination varies the value of all three components, HSV segregates the colour information of the image (denoted by H) from the brightness information (denoted by V). This is a very useful characteristic for outdoor applications like apple fruit gardens, where the influence of shadow areas, canopies, and variations in the time of day cause major illumination variations.

It is empirically observed that the hue channel is robust to lighting conditions that would confound RGB or BGR-based detectors. When a red apple looks crimson in the presence of bright lighting, dark magenta in partial shadow, and black in intense shadow, the hue components of the apple remain in the red spectrum (0-15° in the standard HSV color wheel). This makes thresholding in HSV more robust than naive RGB or BGR thresholding in outdoor lighting conditions.

Morphological Operations for Noise Suppression: Hue-based thresholding inevitably produces noisy binary masks due to similarly colored non-fruit objects in the orchard environment including reddish foliage, weathered wood, and incidental colored objects. The best implemented by morphological operations as part of the process to preserve true apple region structure. Opening-erosion followed by dilation removes small isolated bright regions. Therefore, it filters the sporadic false positives coming from single pixels or small clusters matching the hue threshold but not corresponding to apples. Closing-dilation followed by erosion fills small dark holes, therefore improving region connectedness and compactness. This representation of the size of a structuring element is a trade-off: the smaller elements capture fine detail but do not remove noise; larger elements remove noise more aggressively but risk eroding small or thin apples. A circular structuring element with an empirically determined radius (e.g., 5–10 pixels depending on the image resolution) is an effective compromise.

2.3 Machine learning

Dimensionality and Feature Extraction:

The HOG descriptor turns raw image patches into high-dimensional feature vectors, usually from 1,000 to 2,000 features, encoding shape and textural information. What gives HOG its force is the explicit encoding of edge structure and orientation patterns: smooth, rounded edges characteristic of apple curvature produce distinctive HOG patterns, while rough, vertically-oriented bark texture produces markedly different signatures. This makes it possible for the SVM to learn a decision boundary that effectively separates apples from confounding objects-a task considerably harder using raw pixels or simple color features.

Support Vector Machines Classification:

The linear kernel-based linear SVM is able to distinguish apples from non-apples by identifying the optimal boundary between the extracted HOG features of both classes. Its choice of a linear kernel is of crucial importance since it helps in the following ways: Interpretability: The weight vector w learned contains direct information about the importance of features, allowing researchers to derive understandings of which edge orientation or features distinguish apples.

Computational Efficiency: Both training and running a linear SVM are much faster than their non-linear kernel counterparts, making it easier to quickly test or prototype their design application with the possibility of real-time execution.

Generalization: Based on the empirical evidence, it is clear that the linear SVM is able to classify with a similar level of accuracy to other kernels on HOG features without the risk of overfitting that is inherent in more complex kernels.

The decision function of the SVM yields a confidence score that is continuous. f(x) quantifies levels of prediction certainty. Varying the threshold value of classification allows for trade-off between sensitivity (recall) and specificity (precision) for a given prediction task.

Post-processing for semantic refinement.

Aspect ratio, area, confidence thresholding, and non-maximum suppression post-processing filters encode apple form and expected detection patterns. Even if these kinds of restrictions might be learnt from start to finish, writing them out clearly has useful effects. Each filter is based on a clear biological or geometric assumption, which makes things more transparent and allows for domain-driven adjustment. Separating learned parts from rule-based limits makes things more stable and makes it easier to find bugs. These operations are quick to compute, making them possible to use in real time. Non-maximum suppression gets rid of overlapping detections by keeping the forecast with the highest confidence. This makes sure that the apple counts are correct and not duplicated.

3 Results and Evaluation

The performance of the integrated apple counting algorithm was evaluated with classification metrics from the confusion matrix in Figure 4. The classification metrics are accuracy, precision, recall and the F1-score. Overall, this particular model achieved an accuracy of 73.08 %. Individual class percentage were determined and described below. In the apple class, the precision and the recall were 0.74 Which is described in Table 1.This clearly demonstrates that the model is not aggressive and gave equal importance to both false positives and missed detections. The training was done with the combined datasets consisting of both MinneApple and Fuji-Sfm. During the testing, 20 resolution images are taken from the Fuji-Sfm dataset. The F1-score is also the same as the precision and accuracy. This clearly explains that the integration of image processing and SVM is stable. The color similarity between the red apples and the background trunks affects the decrease in the F1 score. The Figure 5 shows the apple detection with three different color boxes, namely red, green, and yellow. The degree of robustness is clearly differentiated with these boxes. The green boxes shows high confidence, followed by yellow boxes with medium confidence and low confidence. The differentiation shows the reliability of the integrated model with image processing and the SVM.

Table 1: Classification Performance Metrics

Class	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.73	0.72	0.72	587
Apple	0.74	0.74	0.74	613
Accuracy			0.73	1200
Macro Avg	0.73	0.73	0.73	1200
Weighted Avg	0.73	0.73	0.73	1200

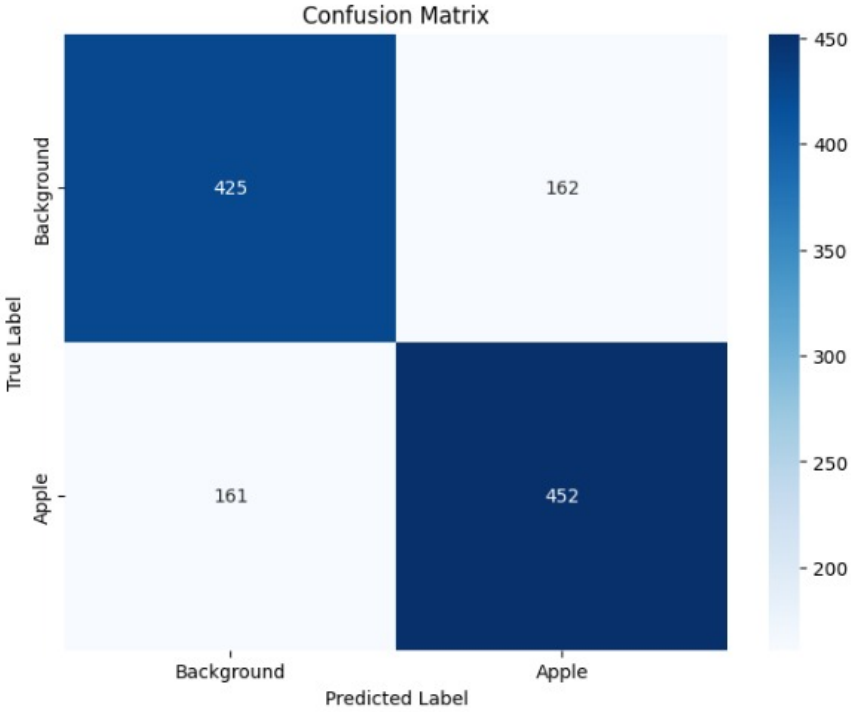

Figure 4: Confusion Matrix

Figure 5: Final Detection Result

3.1 Referencing

References

[1] A. Gongal, S. Amatya, M. Karkee, Q. Zhang, and K. Lewis, "Sensors and systems for fruit detection and localization: A review," *Computers and electronics in agriculture*, vol. 116, pp. 8–19, 2015.

[2] Q. Wang, S. Nuske, M. Bergerman, and S. Singh, "Automated crop yield estimation for apple orchards," vol. 88, 06 2012.

[3] N. Häni, P. Roy, and V. Isler, "Minneapple: A benchmark dataset for apple detection and segmentation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 852–858, 2020.

[4] A. van Meekeren, M. Aghaei, and K. Dijkstra, "Exploring the effectiveness of dataset synthesis: An application of apple detection in orchards," 06 2023.