A

Mini Project Report

On

FRUIT RIPENESS DETECTION USING OPENCV

Submitted to JNTU HYDERABAD

In Partial Fulfilment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

Submitted By

MORAVANENI DHEERAJ	**218R1A1241**
NARAYANA GUNA PRIYA	**218R1A1242**
KOLLI MAHESH	**218R1A1236**
PATHIGARI VIKAS	**218R1A1248**

Under the Esteemed guidance of

Mrs.T.SWATHI

Assistant Professor, Department of IT

Department of Information Technology

CMR ENGINEERING COLLEGE

(UGC AUTONOMOUS)

(Accredited by NAAC&NBA, Approved by AICTE NEW DELHI, Affiliated to JNTU, Hyderabad)

(Kandlakoya, Medchal Road, R. R. Dist. Hyderabad-501 401)

(2024-2025)

CMR ENGINEERING COLLEGE

(UGC AUTONOMOUS)

(Accredited by NAAC&NBA, Approved by AICTE NEW DELHI, Affiliated to JNTU, Hyderabad)

(Kandlakoya, Medchal Road, R. R. Dist. Hyderabad-501 401)

Department of Information Technology

CERTIFICATE

This is to certify that the project entitled “**FRUIT RIPENESS DETECTION USING OPENCV”** is a bonafide work carried out by

MORAVANENI DHEERAJ	**218R1A1241**
NARAYANA GUNA PRIYA	**218R1A1242**
KOLLI MAHESH	**218R1A1236**
PATHIGARI VIKAS	**218R1A1248**

in partial fulfilment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in **INFORMATION TECHNOLOGY** from CMR Engineering College, affiliated to JNTU, Hyderabad, Under our guidance and supervision.

The results presented in this project have been verified and are found to be satisfactory. The results embodied in this project have not been submitted to any other university for the award of any other degree or diploma.

Internal Guide	Head of the Department
Mrs. T.SWATHI	**Dr. MADHAVI PINGILI**
Assistant Professor	Professor & HOD
Department of IT	Department of IT
CMREC, Hyderabad	CMREC, Hyderabad

DECLARATION

This is to certify that the work reported in the present project entitled **“fruit ripeness detection using opencv”** is a record of bonafide work done by us in the Departmentof Information Technology, CMR Engineering College, JNTU Hyderabad. The reports are based on theproject work done entirely by us and not copied from any other source. We submit our project for further development by any interested students who share similar interests to improve the project in thefuture.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

MORAVANENI DHEERAJ	**218R1A1241**
NARAYANA GUNA PRIYA	**218R1A1242**
KOLLI MAHESH	**218R1A1236**
PATHIGARI VIKAS	**218R1A1248**

ACKNOWLEDEGEMENT

We are extremely grateful to **Dr. A. Srinivasula Reddy**, Principal and **Dr. Madhavi Pingili**, Professor & HOD, **Department of IT**, **CMR Engineering College** for their constant support.

We are extremely thankful to **Mrs. T.Swathi**, Assistant Professor, Internal Guide, Department of IT, for her constant guidance, encouragement and moral support throughout the project.

We will be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred in this Project.

We express our thanks to all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time.

Finally, we are very much thankful to our parents who guided us for every step.

MORAVANENI DHEERAJ	**218R1A1241**
NARAYANA GUNA PRIYA	**218R1A1242**
KOLLI MAHESH	**218R1A1236**
PATHIGARI VIKAS	**218R1A1248**

CONTENTS

ABSTRACT

The agricultural industry faces challenges in optimizing fruit harvesting processes, where accurate and timely ripeness detection is crucial for ensuring product quality and reducing waste. This study proposes a novel approach to fruit ripeness detection using OpenCV, a popular computer vision library. The methodology involves the development of a robust algorithm that leverages image processing techniques to analyze fruit images and determine their ripeness status. The process begins with the acquisition of high-resolution images of fruits in various stages of ripeness. These images serve as the input data for the proposed algorithm, which employs color- based segmentation techniques to isolate the fruit from the background. Subsequently, feature extraction methods are applied to capture key characteristics associated with ripeness, such as color intensity, texture, and shape. OpenCV's machine learning capabilities are then utilized to train a model capable of classifying fruits into distinct ripeness categories. The model is trained on a labeled dataset, incorporating images of fruits at different ripeness stages. The trained model is subsequently applied to new images, providing real-time ripeness assessments. To enhance the system's adaptability and accuracy, the algorithm incorporates dynamic thresholding techniques to account for variations in lighting conditions and fruit types. Additionally, the implementation includes a user-friendly interface for easy integration into existing agricultural workflows. The proposed approach demonstrates promising results in terms of accuracy and efficiency, showcasing its potential to revolutionize fruit harvesting practices. By automating the ripeness detection process, farmers and agricultural professionals can make more informed decisions about the optimal timing for harvesting, leading to improved yield quality and reduced post-harvest losses.

Keywords: Ripeness detection, OpenCV, Color-Based Segmentation, Dynamic Thresholding, Accuracy, Reduced Post-Harvest Losses.

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 Introduction

Fruit ripeness detection is a crucial task in the agricultural and food industry. The ability to accurately determine the ripeness of fruits can impact various stages of the supply chain, from harvesting to distribution and consumption. One effective way to perform fruit ripeness detection is by leveraging computer vision techniques, and OpenCV (Open-Source Computer Vision Library) is a powerful tool for this purpose.

The flexibility of OpenCV facilitates the integration of additional machine learning components, allowing for continuous improvement and adaptation to diverse fruit types and environmental conditions. This adaptability is particularly valuable in addressing the variability in fruit appearance due to factors such as species, cultivar, and ripening conditions. As technology continues to advance, fruit ripeness detection using OpenCV holds the potential to enhance productivity, reduce waste, ensure the delivery of high-quality produce to consumers. The development and refinement of such systems contribute to the broader field of precision agriculture and smart food processing, paving the way for more sustainable and efficient practices in the agri-food sector.

It has several uses like Object Detection and Video Processing. Computer Vision overlaps with fields like Image Processing, Photogrammetry, and Pattern Recognition. A popular wrapper Emgu CV is used to run OpenCV using C#. OpenCV has always aimed to support commercial computer vision development by providing a common infrastructure to build on. Advancing vision- based commercial applications by making portable, performance-optimized code available for free increases the need for fast processors.

1.2 Project Objectives

The Fruit Ripeness Detection System revolutionizes agricultural practices by automating and accurately assessing fruit ripeness using OpenCV and computer vision. Its image preprocessing, color space conversion, and adaptable machine learning components ensure accuracy across various fruits and environmental conditions. A Beyond efficiency gains, this system significantly reduces food waste, aligning with sustainability goals. And its dynamic, data-driven approach fosters continuous improvement, aiding precision agriculture by enabling real-time decision-making for optimized resource allocation and arvest timing. This not only enhances fruit quality but also boosts market competitiveness. From this project we can ensure that the loss of crop and the wastage of the crop is minimized and can be useful for many of the farmers to do the harvest of the crop in the correct time which will be very helpful for them.

1.3 Purpose of the Project

Food waste causes numerous issues for the agriculture sector, such as picking crops before they are ready. So, this may result in food waste and financial loss for farmers. Our project, fruit ripeness detection, aims to address this issue. The basis for this project is OpenCV. It will function by utilizing computer vision techniques and algorithms. Therefore, before fruit is picked, this project helps to determine if it is ripened or not. In order to prevent food waste and to ensure that farmers do not suffer losses.

1.4 Existing System with Disadvantages

Current fruit ripeness detection systems typically use traditional methods that rely on basic image features and shallow machine learning models. These systems often utilize features such as saturation, hue,and RGB and apply simple classification algorithms to detect ripeness states. They may alsouse pre-defined image categories and rely on limited images.

Disadvantages:

- **Limited Generalization:** ML models struggle to generalize across diverse fruit varieties due to differences in size, shape, and color.
- **Dependency on Training Data:** ML model accuracy relies heavily on the quality and diversity of training data, affecting real-world applicability.
- **Security and Privacy Concerns:** Transmitting sensitive ripeness data over networks raises concerns about data security and privacy on IoT devices.

1.5 Proposed System with Features

The new system we are proposing is like a smart detective for fruits, making sure they are at the right level of ripeness. Our system uses a powerful tool called OpenCV to look at pictures of fruits and figure out if they're ready to be eaten. OpenCV helps us do this by turning the pictures into a form it can understand better. the system gets to work, separating the fruits from the background and picking out important details like color and shape. The system can be helpful in many areas, like helping farmers know when to harvest their crops or making sure the fruits you see at the store are just right for you to take home.

It is a basic computer vision script for fruit ripeness detection using the OpenCV library in Python. It captures frames from a camera feed, converts them to the HSV color space, and defines color ranges to create masks for identifying ripe and unripe regions based on color characteristics.

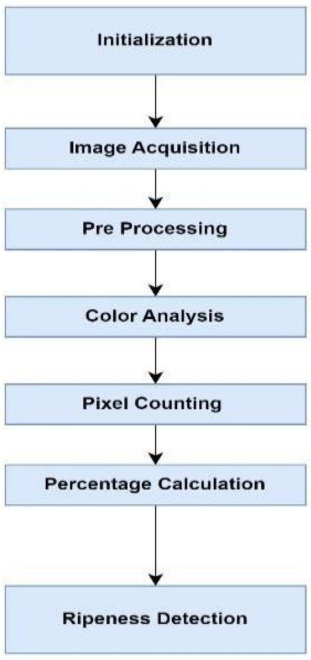


Figure 1.5.1: Block diagram of proposed system.

Advantages:

- Acurate visual feature detection: By learning OpenCV's image processing capabilities, the system can extract key features like colour,texture and shape, enabling accurate detection of fruit ripeness across different types of fruits.
- Real time processing: OpenCV allows for efficient, real time processing of images, making the system suitable for applications where rapid quality assessment is critical, such as in agricultural fields or supermarkets.
- Scalability and Flexibility: The system is designed to handle large datasets and can be easily scaled or modified to incorporate additional images or more complex models, making it adaptable to various use cases.
- Cost effective solution: Using the OpenCV with the standard cameras or smartphones provides a low cost solution for automating fruit ripeness detection as it eliminates the need of specialized hardware.

1.6 Input And Output Design

Input Design

In a fruit ripeness detection system, the process typically begins with Initialization, where the system is set up and calibrated. Image Acquisition involves capturing pictures of the fruit using cameras or other imaging devices.It entails cleaning, enhances the image to improve the accuracy of subsequent

Color Analysis focuses on examining the color characteristics of the fruit, as ripeness often correlates with changes in color. Pixel Counting involves quantifying the number of pixels associated with specific color ranges or features relevant to ripeness. Percentage Calculation then determines the proportion of ripe areas based on the pixel count.

Finally, Ripeness Detection involves interpreting the calculated percentages to classify the fruit as either ripe or unripe. This comprehensive approach integrates image processing techniques and color analysis to automate the assessment of fruit ripeness

Objectives:

- To build and refine a machine learning model capable of accurately identifying and classifying a wide range of images from the environment, leveraging advanced feature extraction techniques such as OpenCV, RGB features.
- To design and implement a standardized input data processing pipeline that ensures high-quality, consistent, and error-free image data for detecting the images, thereby maximizing the model's performance and reliability.
- To create a clear, guided process for operating personnel to prepare, validate, and input image data, ensuring that the OpenCV system is easy to use and reduces the likelihood of errors during data capturing and processing.

Output Design

A quality output in the Fruit ripeness detection system is essential for ensuring that users receive information accurately and effectively. The accuracy of outputs determines howresults are passed to users and other systems, playing a crucial role in the system's utility and user decision-making.

- Conveying Classification Results: Clearly communicate the classification of fruit ripeness, including visual indicators and the relevant metrics such as accuracy or confidence levels.This help users understand the system’s performance and make decisions.
- Signaling Important Events: Alert users to important stages in fruit ripeness, such as when fruit is overripe or ready for the harvest. The system should also notify the users of any issues, such as inconsistent image quality or detection errors, allowing quick corrections.
- Triggering Actions: Based on the ripeness classification, enable actions as sorting fruits, sending notifications, or logging the results for the inventory management. Ensure that these actions are clearly defined and easy to execute, making the system efficient for practical use in agriculture and the commercial sectors or the retail settings helps to ease the work. This ease of work will help both in agricultural and food sector when the results are accurate.

2. LITERATURE SURVEY

A. M. de Oca, L. Arreola, A. Flores, J. Sanchez and G. Flores, "Lowcostmultispectral imaging system for crop monitoring,"2018 International Conference on Unmanned Aircraft Systems (ICUAS).

The Research Article work presents the design and development of a multispectral imaging system to precision agriculture tasks. The imaging system features two small digital cameras controlled by a microcomputer embedded in a drone.

R. F. Maia, I. Netto and A. L. H. Tran, "Precision agriculture using remote monitoring systems in Brazil", 2017 IEEE Global Humanitarian Technology Conference (GHTC).

This paper describes a real-time, in-situ agricultural internet of things (IoT) device designed to monitor the state of the soil and the environment.

N. Kitpo and M. Inoue, "Early Rice Disease Detection and Position Mapping System using Drone and IoT Architecture", 2018 12th South East Asian Technical University Consortium (SEATUC).

For a better rice productivity, this paper presents the implementation system of drone based on Internet of Things (IoT) architecture using real-time information including data acquisition and data analysis using image processing technique.

K. R. Aravind and P. Raja, "Design and simulation of crop monitoring robot for green house", 2016 International Conference on Robotics: Current Trends and Future Challenges (RCTFC).

Incidence of disease in horticultural crops is one of the important problems affecting the production of fruits, vegetables and flowers. Regular monitoring of crops for early diagnosis and treatment with pesticides or removal of the affected crop is some of the solution to minimize the crop loss.

A. Agarwal, A. K. Singh, S. Kumar and D. Singh, "Critical analysis of classification techniques for precision agriculture monitoring using satellite and drone", 2018 IEEE 13th International Conference on Industrial and Information Systems (ICIIS)

Satellite imagery is used in various application such as metrological, change detection, disaster migration, agriculture development etc. With the changing habits of the agriculture practice, there is a need to estimate the vegetated area with the non-vegetated area for the yield estimation.

P. Senthil and I. S. Akila, "Automated Robotic Moisture Monitoring in Agricultural Fields", 2018 International Seminar on Intelligent Technology and Its Applications (ISITIA)

The main objective of this project is to use a robotic kit in collaboration with the on-field moisture sensor circuits, thereby creating an efficient and economical moisture monitoring system.

M. B. Garcia, S. Ambat and R. T. Adao, "Tomayto Tomahto: A Machine Learning Approach for Tomato Ripening Stage Identification Using fruits, colour Pixel-Based Color Image Classification", 2019 IEEE 11th International Conference on Humanoid Nanotechnology Information Technology Communication and Control Environment and Management (HNICEM).

This study proposes an automatic tomato ripeness identification using Support Vector Machine (SVM) classifier and CIELab color space via a machine learning approach.

R. Hamza and M. Chtourou, "Design of fuzzy inference system for apple ripeness estimation using gradient method", IET Image ProcessingThis

In this study, a fuzzy classification approach based on colour features has been investigated to estimate the ripeness of apple fruits according to three maturity stages; unripe, turning-ripe and ripe.

Renann G Balvindo;Raphael Antoine U. Lim;Patrick Reylie R. Salvador;Euri Andre P. Tiamzon "Real-Time Banana Ripeness Detection and Classification using YOLOv8",2024 9th International Conference on Mechatronics Engineering (ICOM).

Due to farmers in the Philippines having to utilize manual methods to determine the grade and quality of freshly harvested bananas, post-harvest classification and segregation accounts for between 3 % and 30% of food waste that is linked to bananas. This study focused on real-time banana object recognition and classification via a webcam using neural networks. The principles of computer vision and object recognition and how they relate to fast algorithms like CNN and YOLO are a major emphasis of this study. The researchers used the YOLOv8 algorithm to develop their model since the appropriate selection of the YOLO model determines both the accuracy and processing speed of the photographs**.**

G. Saranya;N. Dineshkumar;A. S. Hariprasath;G. Jeevanantham Fruit ripeness detection", 2023 International Conference on Computer Communication and Informatics (ICCCI).

Being an agricultural country, India's economy is heavily dependent on the growth of agricultural products and related market goods. In India, rain, which is incredibly unpredictable, has a major impact on agriculture. The development of farming also depends on a variety of requirements, especially nitrogen, phosphorus, potassium, plant growth, moisture, surface temperature, and meteorological conditions such as temperature, rain, and so on. India is now making rapid technical progress. Innovation will therefore be beneficial to agriculture, improving plant efficiency and providing farmers with significantly higher yields. The proposed employment provides a service for wise farming by recommending plants to the agricultural sector, which can greatly increase farmers' productivity. Weather report information acquired from IMD (Indian Metrological Division) such as temperature level and rains and specifications database provides an understanding of which crops are appropriate to be cultivated.

Dilip Kumar Jang Bahadur Saini;Sk Hasane Ahammad;Purushottam Das;Ashutosh Bhatt;Pragati Malusare;Sandeep Dwarkanath Pande "Grading Using Arduino in IoT",2023 International Conference on Sustainable Emerging Innovations in Engineering and Technology (ICSEIET)

Ripening is the process by which the growing fruit will get the required quality and palatable nature. Grading fruits play an essential role in the food processing industry because by knowing the quality of the fruit, the vendor or the farmer can decide when he must sell his product to his customer. If the farmer or the fruit vendor knows the correct quality of the product, then he can sell his product at a better price, and he may get some profit. In this work, we supervise the quality of climacteric fruits like bananas and mango. It has been proposed to permit customers to purchase healthy fruits without actively contacting products and to benefit a predetermined set of individuals. The sensing element is the one that helps to detect the quality of the fruit, and here the sensing element is a gas sensor that is connected to the Arduino Uno. The information about the fruit will be sent to the user Via NodeMCU using Arduino Uno**.**

Nurulaqilla Khamis;Hazlina Selamat;ShuwaibatulAslamiah Ghazalli;Nurul Izrin Md Saleh;Nooraini Yusoff "Comparison of Palm Oil Fresh Bunches (FFB) Classification Technique using Deep Learning Method",2022 13th Asian Control Conference (ASCC)

The ripeness of palm oil fruit is currently determined through manual visual inspection by palm oil estate workers that could result inconsistent and inaccurate fruit grading. Moreover, the manual inspection is time-consuming and exhausting duty for humans to complete the daily repetitive task. To overcome this issue, this paper proposes an automatic fruit grading classification by utilizing computer vision technologies. A comparison using image classification (ResNet50) and object detection (YOLOv3) technique is analysed in this work. It is clearly demonstrated that object detection model is remarkable in improving ripeness category based on the finer level of feature that has been extracted during the convolutional process.

Yvhan Jiao;Qi Zhang;Xinyao Luo;Zhaohua Liang " based on miniature spectral sensor", 2021 International Symposium on Computer Technology and Information Science (ISCTIS)

Due to the problems of short shelf life and poor taste of immature fruits, a reasonable judgment of fruit maturity in the process of fruit picking and transportation can not only promote the development of China's fruit industry, but also is very important for China's fruit export trade. This device is a miniature spectrometer designed with C12666MA as the spectral sensor. It can collect and process the spectral signal of fruit surface pigment, communicate with Android smart phone through ESP32, draw on the smart phone APP and analyze the spectral data, so as to achieve the purpose of detecting fruit maturity. In the process of the experiment, the spectral signal of chlorophyll in the epidermis of different fruit varieties was detected. The detector could meet the measurement requirements and realize nondestructive testing. It also has the advantages of easy to carry, low cost and high precision.

Sheng Tan;Linghan Zhang;Jie Yang “Sensing Fruit Ripeness Using Wireless Signals”,2018 27th International Conference on Computer Communication and Networks (ICCCN)

This paper presents FruitSense, a novel fruit ripeness sensing system that leverages wireless signals to enable non-destructive and low-cost detection of fruit ripeness. Such a system can reuse existing WiFi devices in homes without the need for additional sensors. It uses WiFi signals to sense the physiological changes associated with fruit ripening for detecting the ripeness of fruit. FruitSense leverages the larger bandwidth at 5GHz (i.e., over 600MHz) to extract the multipath-independent signal components to characterize the physiological compounds of the fruit. It then measures the similarity between the extracted features and the ones in ripeness profiles for identifying the ripeness level. We evaluate FruitSense in different multipath environments with two types of fruits (i.e, kiwi and avocado) under four levels of ripeness. Experimental results show that FruitSense can detect the ripeness levels of fruits with an accuracy over 90%.

S.Gayathri;T.U. Ujwala;C.V. Vinusha;N. Rebecca Pauline;D.B. Tharunika “ Papaya Using Deep Learning Approach”,2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)

Papaya is a berry fruit with nutritional as well as real worth because to its non-seasonality and short harvesting period. The fiscal year 2020 statistics shows that the volume of papaya production increased to over six million metric tons in India. The grading of papayas is done by hand by human operators, which might lead to misclassifications. The identification of the ripeness of a fruit is important in case of distributing the classified papaya packages as well as in purchasing them by customers. Many projects were proposed earlier for classifying fruits and vegetables, however they were done using machine learning algorithms while the proposed system focuses on deep learning algorithm, especially using Convolution Neural Network (CNN). Convolution Neural Network is a deep learning technique that identifies features without the need for manual absorption. The papaya dataset which is used for this system consist of 300 images, in which each class (ripe, unripe and partially ripe) has 100 images. The proposed model is expected to have a maximum accuracy.

Akshay A Menon;Arun K Nair;Ananthu Vasudevan;Krishna Das K S;Anjali T “RipId App: Estimator and Object Recognition Application” ,2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)

In the agricultural sector, India is one of the largest nations. This is because India has a population of 1.3 billion, providing food for these people is a huge task. The cultivation and production of agricultural products in huge amounts will certainly have its own boon and bane. Producing huge amounts of food more than what the human society needs is a common problem, but if compared with the huge population that India has it is certainly a major problem. The ratio of supply and demand for the agricultural sector is not balanced at all. The survey conducted by the Central Institute of Post-Harvesting Engineering and Technology (CIPHET) states that 16% of agricultural food products are wasted because of the overproduction of the supply chain. The food products over produced are to be preserved but these preserving methods are viable for normal farmers who cannot afford these expensive preserving methods. Perishable products such as fruits and vegetables are induced with artificial chemicals which preserve them longer but are unhealthy for human consumption.

Siti Azizah;Wahidin;Margaretha Padang;Ledya Novamizanti;Sofia Sa'idah "Identifying the Ripeness and Quality Level of Strawberries Based on YOLOv7-EfficientNet",2024 International Conference on Data Science and Its Applications (ICoDSA)

Strawberries (Fragaria x ananassa) are fruits that hold significant economic importance in Indonesia due to their high demand and profitability. These fruits are highly beneficial for daily consumption and are rich in nutrients. However, traditional harvesting methods reliant on visual assessment can lead to inaccuracies and inefficiencies. Evaluating strawberry ripeness is crucial to ensure optimal harvest quality. To address this challenge, Deep Learning emerges as a promising solution for accurately assessing strawberry ripeness and quality visually. This study compares the performance of the basic YOLOv7 model with the YOLOv7-EfficientNetV2S model in detecting strawberry ripeness and quality levels. In the Basic YOLOv7 model, YOLOv7 detects and classifies strawberry ripeness and quality levels. In contrast, in the YOLOv7-EfficientNet model, YOLOv7 is utilized for detection, while EfficientNetV2S serves as the classification object. The research methodology employs a structured approach, including dataset collection, model architecture design, image pre-processing, model implementation, and classification. The final evaluation measures accuracy, precision, recall, and F1-Score.

Venkatesh;Prajwal K P;Preeti Patil;Priyanka G;Prajwal Poojary;Satish B Basapur "CRipS: Coconut ripeness Stage detection System" ,2023 International Conference on Network, Multimedia and Information Technology (NMITCON)

Manual Coconut harvesting requires tree-climbing skill, it is hazardous because of the tree's hard shell, height and high occlusions on the tree. Further, it is difficult to determine whether a coconut is ripe or not because of the outside illumination and various background. The development of a coconut ripen stages detection system with the help of deep learning techniques is of great interest to farmers.

Luiz E.P. Reis;Ingrid B.D. Souza;Jorge D.R. da Silva;Celso B. Carvalho;Waldir S.S. Júnior;Andrey R.R. Bessa;Diego A. Amoedo;Edma V.C. Urtiga "Wireless IoT System for Detection of Fruit Maturation by Color and Ethylene Gas",2023 Symposium on Internet of

Things (SIoT)

Over the past decade, Brazil has seen an increase in fresh fruit consumption, intensifying during the pandemic. De-spite this positive trend, there's a significant wastage challenge in markets. Leveraging advancements in Wireless Communication, the Internet of Things, and new technologies, this paper proposes a system to assess fruit ripeness using ethylene gas measurement and color analysis.

Che-Wei Hsu;Yu-Hsiang Huang;Nen-Fu Huang "Real-time Dragonfruit's ripeness Classification System with Edge Computing Based on Convolution Neural Network",2022 International Conference on Information Networking (ICOIN)

In recent years, planted area and production of dragonfruits in Taiwan increased three times in the past decade. As the improvement of agricultural research, it also develops lots of brand-new species. However, grading on dragonfruits still needs to improve. Our research use the appearance of dragonfruit to predict its ripeness by using Convolution Neural Network(CNN) model, which helps to reduce labor and time costs and increase the profit for farmers.

B Lakshmi Sirisha;B Jayendra Nayak;B sai Sndhya;J Eswara Rao;M Devisri " fruit Quality detection Using Deep Learning for Sustainable Agriculture" ,2024 International Conference on Recent Advances in Electrical, Electronics, Ubiquitous Communication, and Computational Intelligence (RAEEUCCI)

Because of the agriculture sector's vital role in global food production and sustainability, fruit quality is critical. Traditional methods of fruit quality evaluation are time consuming and subjective, resulting in inefficiencies in the supply chain and significant food loss. Our research provides a novel solution. By evaluating fruit photos, our algorithm detects fresh versus suboptimal-quality fruits using cutting-edge machine learning techniques. Our model correctly identifies ripeness, flaws, and deformities across multiple fruit varieties using deep learning with CNNs at its heart and transfer learning.

3. SOFTWARE REQUREIMENTS ANALYSIS

3.1. Problem Statement

This project aims to propose a real-time object detection system based on deep learning architectures. Our approach leverages state-of-the-art deep neural network models such as Faster R-CNN, YOLO (You Only Look Once), or SSD (Single Shot MultiBox Detector) to achieve real-time performance. We focus on optimizing the computational efficiency of these models without compromising accuracy, making them suitable for deployment in real-world scenarios. The goal of the project fruit ripeness detection using opencv is to detect the ripeness of the fruits based on the saturation values and by analyzing the hue values with respect to the colours red, green and the blue.

3.2. Modules and Their Functionalities

Data Preprocessing

After collecting datasets from various sources, the dataset must undergo preprocessing before training the model. Data preprocessing involves several stages, beginning with reading the collected dataset and continuing to data cleaning. In data cleaning, datasets containing redundant attributes are removed, as these attributes are not considered for object detection. Missing values in the datasets are also dropped to ensure better accuracy.

Key steps in data preprocessing include:

- **Getting the dataset**: Load or capture the images and extract features.
- **Importing libraries**: Use libraries such as OpenCV for capturing the images and libraries like the NumPy for handling the data.
- **Importing datasets**: Load the image of the fruits from the specified path.
- **Training the classifier**: Train the model using a SVM or CNN depending on the complexities.
- **Testing the classifier**: Evaluate the model's performance on the test data to classify ripeness.
- **Evaluate**: Evaluate the model using metrics like accuracy, precision, and recall to assess the models effectiveness in detecting the fruit ripeness.

Feature Extraction:

In this stage, features such as color histograms, texture patterns, and shape descriptors are extracted from fruit images using the OpenCV library. These features represent the visual properties of the fruit, capturing key aspects like color, surface texture, and size, which are crucial for determining the ripeness of the fruit by representing the ripeness percentage of the fruit in the integer format in accurate manner. For fruit ripeness detection using OpenCV, feature extraction plays a crucial role in identifying key attributes that indicate the fruit’s ripeness. The features commonly extracted in such tasks involve color, texture, and shape since these factors change as a fruit ripens.

Splitting the Data set into the Training set and Test set

In machine learning data preprocessing, the dataset is divided into a training set and a test set. This is a crucial step to enhance the model’s performance. Training the model on one part of the data and testing it on another helps to evaluate how well the model generalizes to new data.

- Training Set: A subset of the dataset used to train the machine learning model. The outputs are known and used to learn the model parameters.
- Test Set: A subset of the dataset used to test the model after training. The model's predictions on this set help evaluate its performance.

Training the Classifier

The classifier is trained using the training data. In the fruit ripeness detection system, a Random Forest Classifier (or another suitable algorithm) is used to distinguish between different stages of fruit ripeness. The classifier learns from the extracted features, such as color, texture, and shape, along with the corresponding labels, allowing it to accurately classify fruits as ripe, underripe, or overripe.

Testing the Classifier

After training, the classifier is tested using the test data. The model predicts the ripeness of the images, and these predictions are compared with the images to assess accuracy.

Evaluation

The classifier's performance is evaluated using a confusion matrix and metrics such as accuracy, precision, and recall. These metrics help in understanding the model's strengths and weaknesses.

3.3. Functional Requirements

Functional requirements describe the essential capabilities that the system must provide to meet the user's needs. The primary functions of the emotion recognition system include:

- **Feature Extraction**: The system should extract meaningful features from images, whichwill be used for ripeness detection.
- **Model Training**: The system must train a machine learning model using the extracted features and their corresponding labels.
- **Ripeness Prediction**: The system should accurately predict the ripeness for the newly captured images.
- **Camera**: A simple and intuitive camera should be provided for users to capture the images.
- **Multi fruit detection**: The system shall support detection and classification of multiple fruits in a single image.

3.4. Non-Functional Requirements

Non-functional requirements address how the system performs its functions and include:

- **Performance**: The system should process and predict ripenesss from images efficiently.
- **Scalability**: The system should handle increasing numbers of users or larger datasets without significant performance degradation.
- **Usability**: The user model should be easy to use, allowing users to interact with the system effortlessly.
- **Reliability**: The system should produce consistent results and handle errors gracefully.

3.5. Feasibility Study

Economic Feasibility

The system is designed to be cost-effective by utilizing open-source libraries such as OpenCV. This reduces development costs as most tools required for the project are freely available.

Technical Feasibility

The technical requirements are minimal, as the system is developed using commonly available Python libraries that require only modest computational resources. This ensures that the system can be deployed on standard hardware without the need for specialized equipment.

Social Feasibility

The system aims to be user-friendly, with an interface designed to be accessible to non-technical users. By providing clear feedback and predictions, the system encourages user acceptance and engagement. Training sessions or guides may be provided to help users get familiar with the system, ensuring a high level of user confidence and satisfaction.

4. SOFTWARE AND HARDWARE REQUIREMENTS

4.1. Software Requirements

The software requirements provide an overview of the essential tools and technologies required for developing the emotion recognition system. This includes the operating system, coding language, development environment, database, and server specifications.

Operating system	:	Windows 10
Coding Language	:	Python
Tool	:	Jupiter notebook
Framework	:	OpenCV
Image processing library	:	OpenCV

4.2. Hardware Requirements

Minimum hardware requirements are highly dependent on the specific software being developed in Visual Studio Code. Applications that need to store large arrays or objects in memory will require more RAM, while applications that perform numerous calculations or tasks more quickly will benefit from a faster processor.

System	:	Intel i3 or above
Hard Disk	:	64 GB
Monitor	:	15” LED
Input Devices	:	Keyboard, Mouse
Ram	:	4 GB

5. SOFTWARE DESIGN

5.1. System Architecture

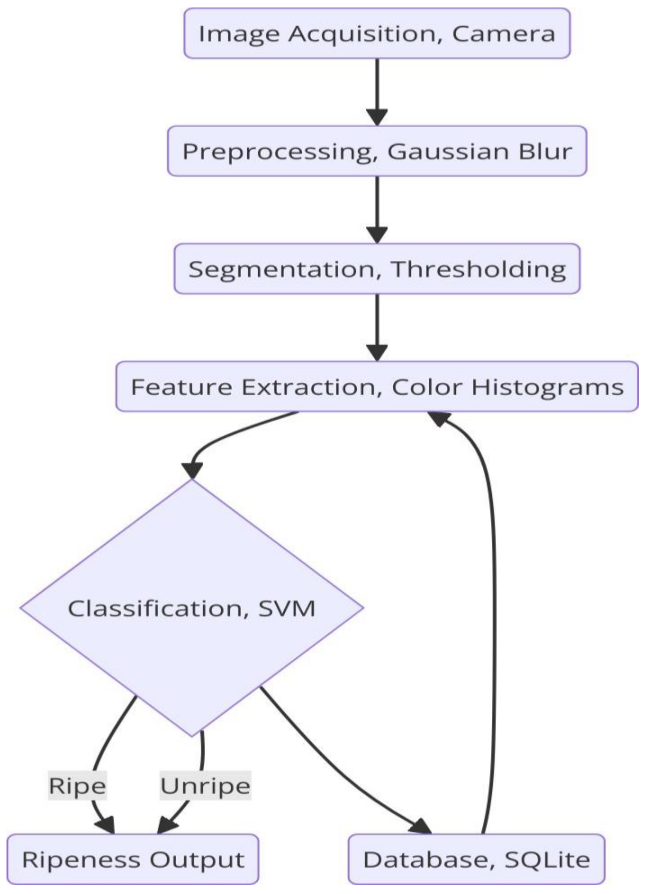


Image Acquisition, Camera
Preprocessing, Gaussian Blur
Segmentation, Thresholding
Feature Extraction, Color Histograms
Classification, SVM
Ripe
Unripe
Ripeness Output
Database, SQLite

The system architecture for a fruit ripeness detection system using OpenCV involves several components working together to achieve the goal. Below is an overview of a typical system architecture for such a system. These features are then used to train a image Classifier to classifyripeness, with the data split into training and test sets. The model's performance is evaluated, and boththe trained model and label encoder are saved for future predictions, enabling ripeness detections from new images inputs without retraining.

5.2. Dataflow Diagram

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system. A Data Flow Diagram (DFD) is a graphical representation of how data moves through a system, showing the inputs, processes, and outputs, as well as how the data is stored.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

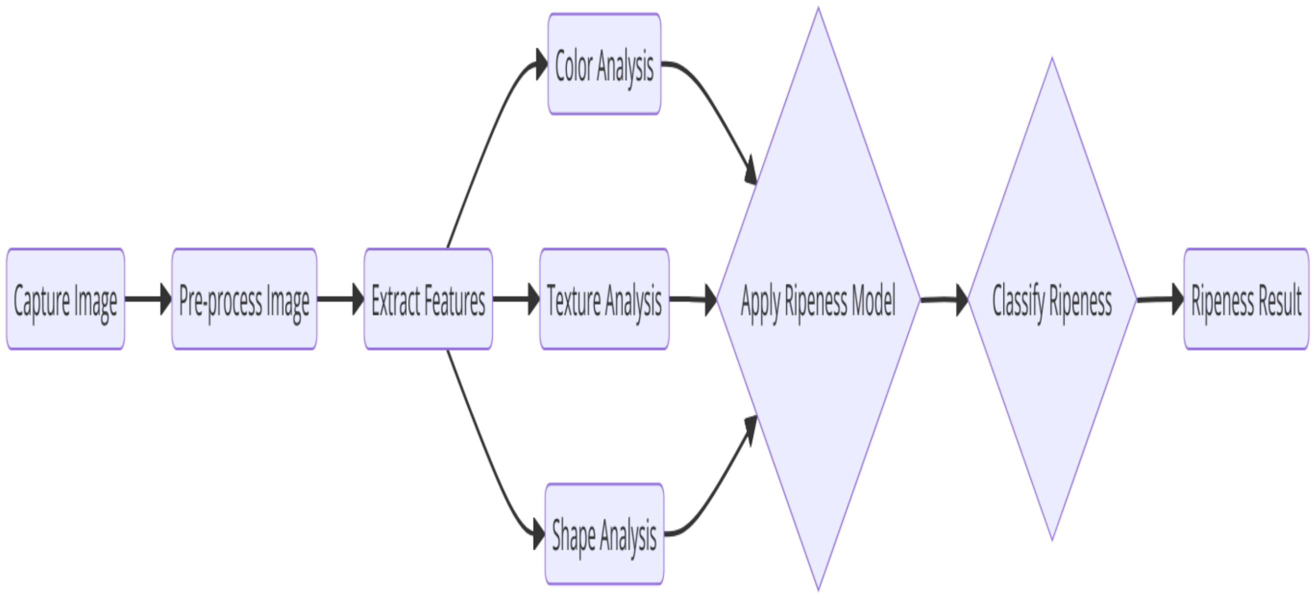


Figure: 5.2. Dataflow Diagram

5.3 UML Diagrams

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

There are several types of UML diagrams and each one of them serves a different purpose regardless of whether it is being designed before the implementation or after (as part of documentation). UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard. The two broadest categories that encompass all other types are behavourial uml diagram and the structural uml diagram, both the types analyses the structure of the process and also describes the behavior of the system. The diagram focuses on the static aspects that what components exist and how the relate and the flow.

- Behavioural UML diagram and
- Structural UML diagram.

As the name suggests, some UML diagrams try to analyses and depict the structure of a system or process, whereas other describe the behavior of the system, its actors, and its building components.

Goals: The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns andcomponents.
- Integrate best practices.

The different types are as follows:

- Sequence diagram
- Use case diagram
- Activity diagram
- Class diagram
- Collaboration diagram

Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

List of actions

User : User need to press button (i.e., Predict) then user will get the output accordingly.

System : System will give the output as he enters according to the given data.

Result : As per user enters the data it will give emotion of the speech.

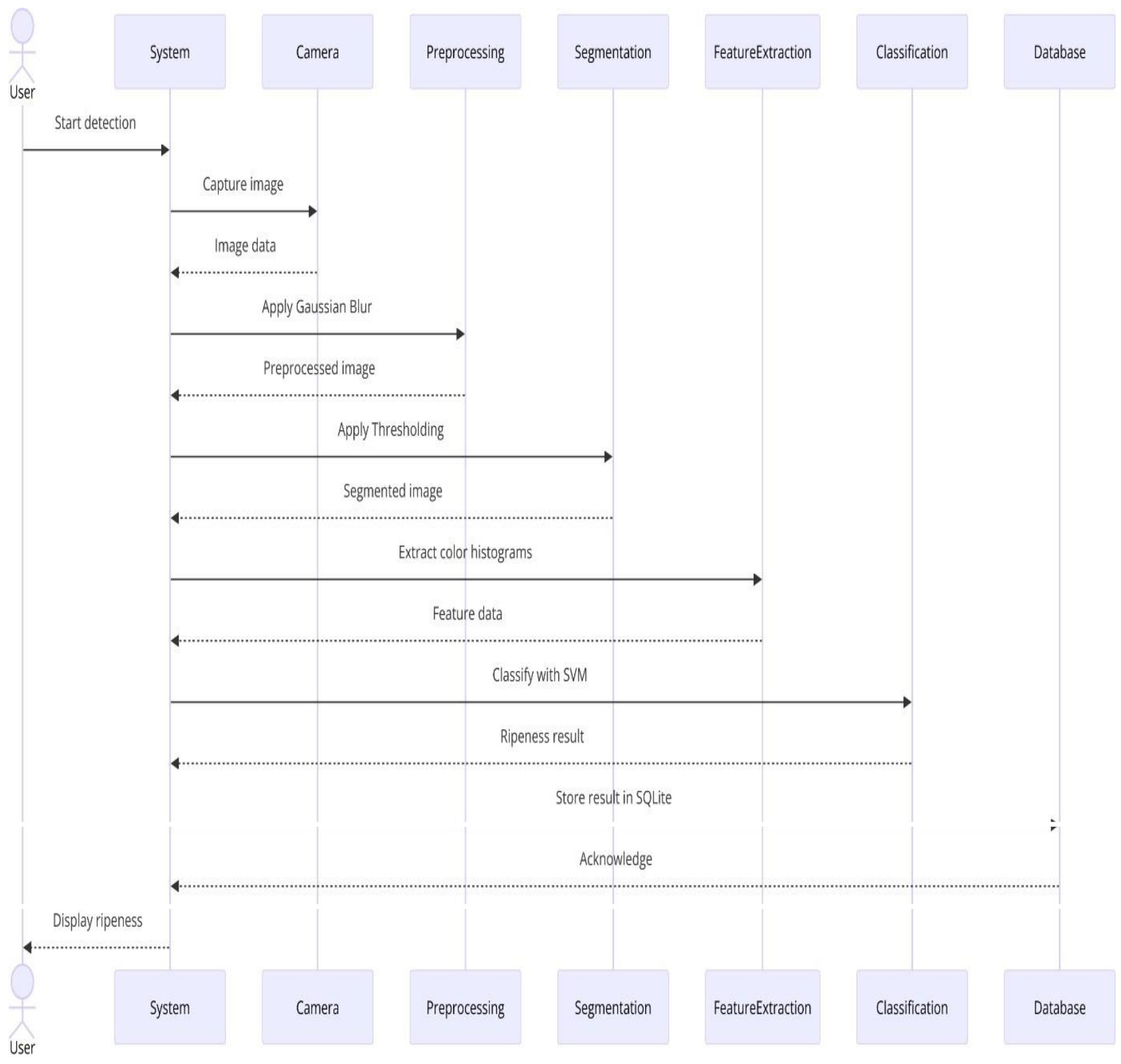


Figure:5.3.1 Sequence Diagram

Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use case in which the user is involved. A use case diagram is used to structure of the behavior thing in a model. The use cases are represented by either circles or ellipses. A use case diagram is a type of behavioral diagram defined by the Unified Modeling Language (UML) that visually represents the functionality of a system from the user's perspective. It illustrates the different interactions users (actors) have with the system, and the specific services or actions (use cases) provided by the system. Use case diagrams are useful for understanding system requirements and the relationships between the system and external entities. A use case diagram is a type of behavioral diagram defined by the Unified Modeling Language that visually represents the functionality of a system from the user's perspective.

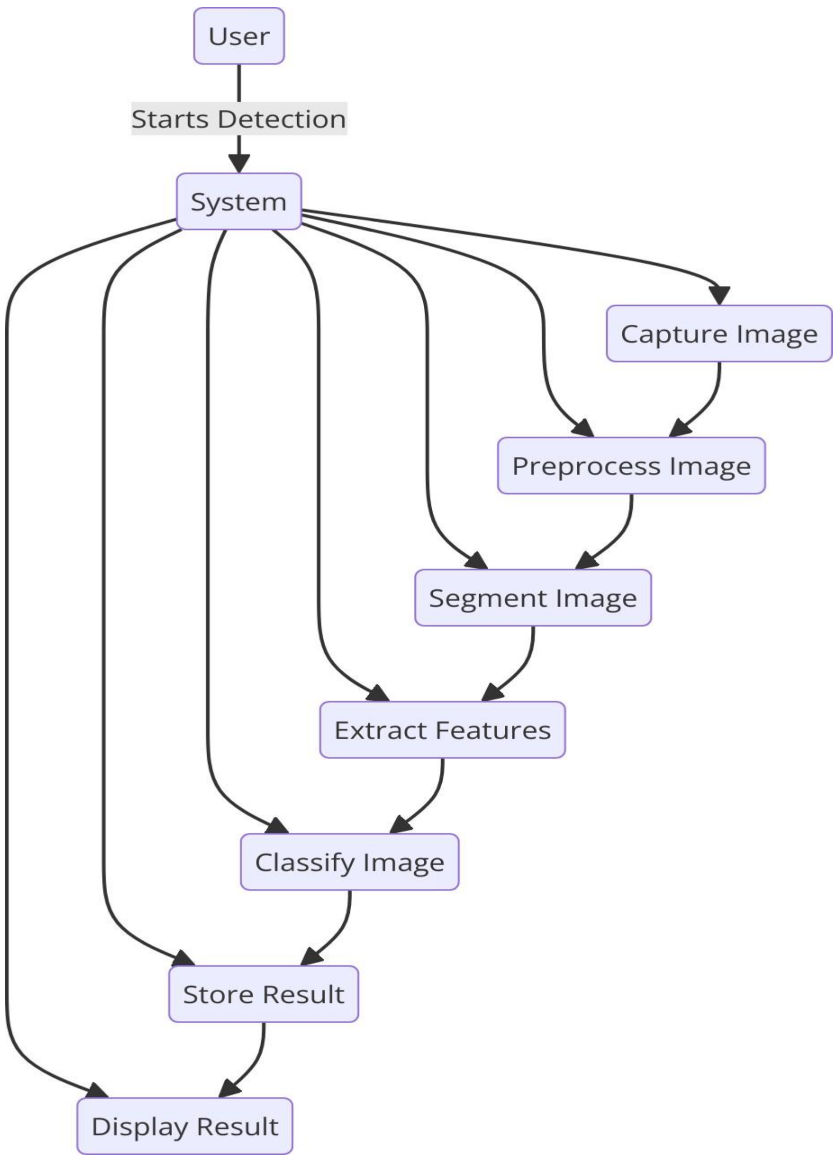


Figure: 5.3.2 Use Case Diagram

The Use Case Diagram provides a high-level overview of the system’s functionalities and interactions between different actors and the system. It helps in understanding what the system is supposed to do from the user’s and administrator’s perspectives and outlines the major operations within the fruit ripeness detection system. A use case diagram is a visual representation in UML (Unified Modeling Language) that illustrates the interactions between users (actors) and the system to achieve a specific goal. It helps stakeholders understand the functional requirements of a system by focusing on "what" the system does rather than "how" it performs it. Use case diagrams are useful during the early stages of software design to outline system functionality.

It’s primarily used to capture the functional requirements of a system and helps in understanding the overall behavior from a user's perspective.

Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

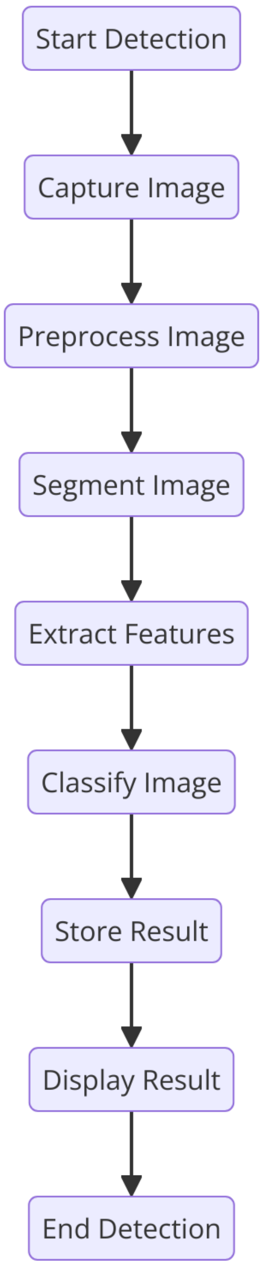


Figure 5.3.3. Activity Diagram

Class Diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of staticstructure diagram that describes the structure of a system by showing the system's classes attributes, operations (or methods), and the relationships among the classes. It explains which class contains information. Class diagrams are useful during the design phase of a project to visualize and model the system's structure, making it easier to understand how different components of the system will interact. It is one of the key diagrams in UML (Unified Modeling Language) that represents the static structure of a system. It shows the system's classes, their attributes, methods, and the showing relationships between the classes.

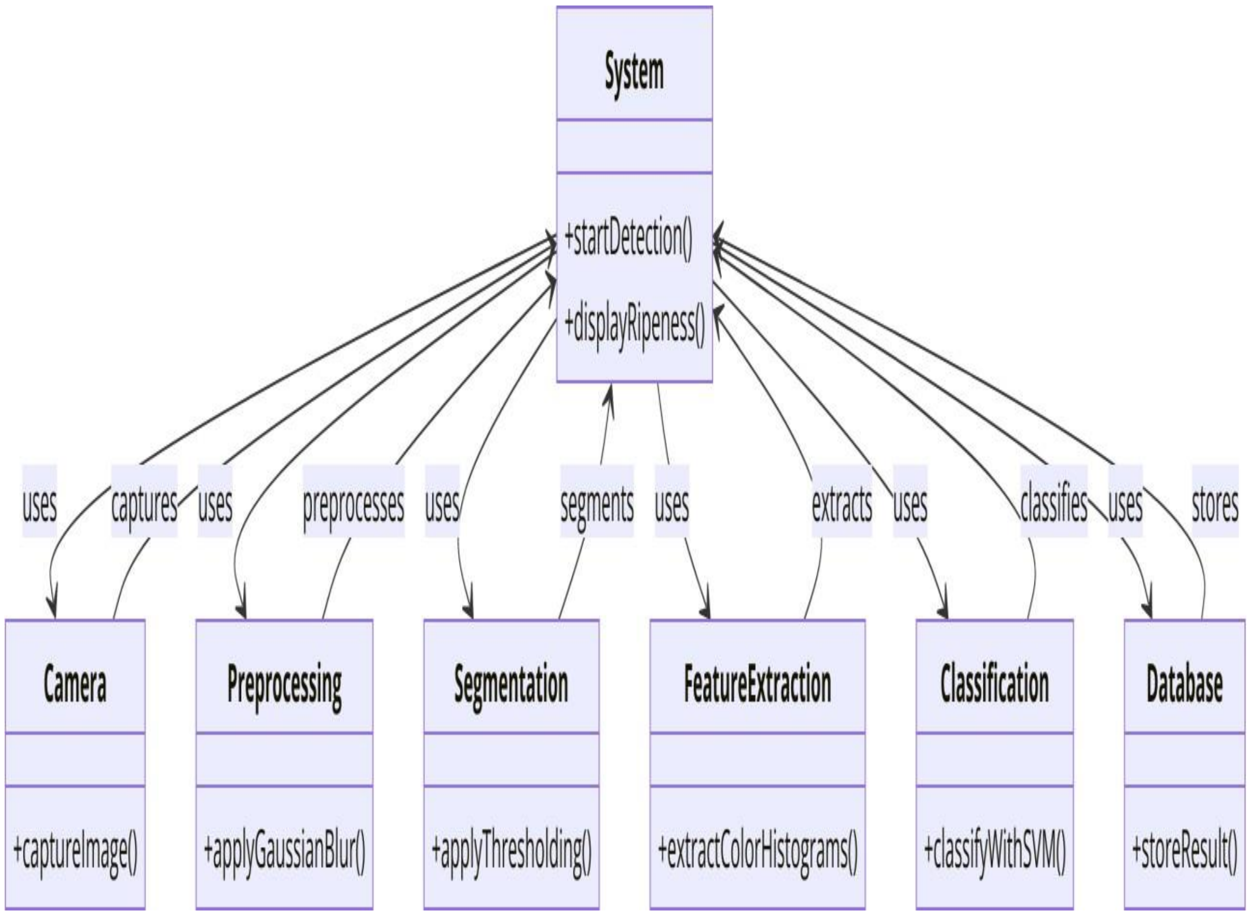


Figure: 5.3.4 Class Diagram

6. CODING AND ITS IMPLEMENTATION

6.1 Source Code

```
from _future_ import
division import io
import os
import
random
import cv2
import numpy as np
import time
from copy import deepcopy

kernelOpen = np.ones((5,
5)) kernelClose =
np.ones((20, 20))

# The name of the image file to annotate
i = time.strftime("%d-%m-%y_%H-%M-%S")

# Capture image
camera = cv2.VideoCapture(0)
return_value, image =
camera.read() cv2.imwrite(i +
'.jpeg', image) del(camera)
frame = image
edge_img = deepcopy(image)

# Finds edges in the input image and marks them in the output map
edges edged = cv2.Canny(edge_img, 50, 100)
edged = cv2.dilate(edged, None,
iterations=1) edged = cv2.erode(edged,
None, iterations=1)

# Find contours in the edge map
cnts, h = cv2.findContours(edged.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

max_cont = max(cnts,
key=cv2.contourArea) x, y, w, h =
cv2.boundingRect(max_cont)
cv2.rectangle(edge_img, (x, y), (x + w, y + h), (0, 0,
255), 2) croppedk = frame[y:y + h, x:x + w]

# Display the edges
cv2.imshow('Edges',
edge_img)

frame = edge_img

# Converting BGR to HSV
```

```
hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

# Define range of red color in HSV
lower_red1 = np.array([0, 50, 50])
upper_red1 = np.array([10, 255, 255])
lower_red2 = np.array([170, 50, 50])
upper_red2 = np.array([180, 255, 255])
reddmask2 = cv2.inRange(hsv, lower_red2, upper_red2)
redmask = redmask1 + redmask2
maskOpen = cv2.morphologyEx(redmask, cv2.MORPH_OPEN, kernelOpen)
maskClose = cv2.morphologyEx(maskOpen, cv2.MORPH_CLOSE,
kernelClose)

maskFinal = maskClose
cv2.imshow('Red Mask',
maskFinal)

# Calculate redness using contours
cnt_r, _ = cv2.findContours(maskFinal, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE) cnt_r_area = sum(cv2.contourArea(c) for c in cnt_r)
print("Redness:", cnt_r_area)

lower_green = np.array([50, 50, 50])
upper_green = np.array([70, 255, 255])
greenmask = cv2.inRange(hsv, lower_green, upper_green)
cv2.imshow('Green Mask', greenmask)
cnt_g = cv2.countNonZero(greenmask)
print("Greenness:", cnt_g)

lower_yellow = np.array([20, 50, 50])
upper_yellow = np.array([50, 255, 255])
yellowmask = cv2.inRange(hsv, lower_yellow, upper_yellow)
cv2.imshow('Yellow Mask', yellowmask)
cnt_y = cv2.countNonZero(yellowmask)
print("Yellowness:", cnt_y)

# Calculate ripeness
tot_area = cnt_r_area + cnt_y +
cnt_g rperc = cnt_r_area / tot_area
yperc = cnt_y /
tot_area gperc =
cnt_g / tot_area

# Adjust the limits for your fruit
glimit = 0.0
ylimit_low = 0.0
ylimit_high = 0.0
if yperc * 100 < 10 or gperc * 100
< 10: print("Fruit not detected")
else:
if gperc > glimit:
   print(f"Unripe ({gperc * 100:.2f}%
Unripe)") else:
print(f"Ripe ({yperc * 100:.2f}% Ripe)")

# Wait for any key to close
while True:
k = cv2.waitKey(5) & 0xFF
 if k != 255:  # If any key is pressed
break

 # Calculate redness using contours
cnt_r, _ = cv2.findContours(maskFinal, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE) cnt_r_area = sum(cv2.contourArea(c) for c in cnt_r)
print("Redness:", cnt_r_area)


lower_yellow = np.array([20, 50, 50])
upper_yellow = np.array([50, 255, 255])
yellowmask = cv2.inRange(hsv, lower_yellow, upper_yellow)
cv2.imshow('Yellow Mask', yellowmask)
cnt_y = cv2.countNonZero(yellowmask)
print("Yellowness:", cnt_y)


    # Adjust the limits for your
fruit glimit = 0.0
ylimit_low = 0.0
ylimit_high = 0.0
 if yperc * 100 < 10 or gperc * 100
< 10: print("Fruit not detected")
else:
if gperc > glimit:
    print(f"Unripe ({gperc * 100:.2f}%
Unripe)") else:
print(f"Ripe ({yperc * 100:.2f}% Ripe)")

    # Calculate ripeness
tot_area = cnt_r_area + cnt_y +
cnt_g rperc = cnt_r_area / tot_area
yperc = cnt_ytot_area
gperc =
cnt_g/tot_area


# De-allocate any associated memory usage
cv2.destroyAllWindows()
```

6.2. Implementation

6.2.1. Python

Python is a versatile, high-level programming language renowned for its readability and ease of use, making it an ideal choice for a wide range of applications, including machine learning and audio processing. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its use of significant whitespace, which allows developers to write clean and maintainable code.

Python is instrumental in developing a speech emotion recognition system through its robust ecosystem of libraries and frameworks. OpenCV is used for image processing. In addition to basic image operations, OpenCV provides advanced feature detection methods, including edge detection with algorithms like Canny and Sobel, and corner detection using Harris and Shi-Tomasi algorithms. It also offers template matching for object recognition. Another key functionality of OpenCV is its ability to detect objects, such as faces, using pre-trained classifiers like Haar cascades, or identifying contours and shapes in an image for segmentation and analysis. These capabilities make OpenCV highly versatile and useful in real-world applications.

OpenCV also integrates machine learning algorithms, including k-nearest neighbors (k-NN), decision trees, and support vector machines (SVM), which can be used for classification tasks. Additionally, OpenCV can work with deep learning frameworks like TensorFlow, PyTorch, and Caffe, enabling tasks such as object detection, image classification, and segmentation using pre-trained models like YOLO, SSD, and MobileNet. It supports both traditional machine learning and deep learning workflows, making it a powerful tool for various industries.

For video processing, OpenCV can capture and manipulate frames from cameras or video files in real time, which is useful for applications such as motion detection, video editing, and object tracking. It also provides optical flow algorithms for tracking the movement of objects across frames in a video. In real-time applications, OpenCV can track moving objects, apply augmented reality techniques by overlaying graphics on real-world scenes, or recognize gestures for interactive control.

Modules Used In Project

OpenCV (cv2):

- **Purpose**: The core module for image processing. It is used for reading images, resizing, filtering, color space conversion, and detecting features like color and texture in the images.

- **Functions**:
    - cv2.imread(): To load images.
    - cv2.resize(): To resize images.
    - cv2.cvtColor(): To convert images between color spaces like RGB and HSV.
    - cv2.inRange(): To create masks for detecting specific color ranges.

NumPy:

- **Purpose**: NumPy is used for handling arrays and performing mathematical operations. In image processing, an image is treated as a multi-dimensional array, so NumPy is essential for operations on pixel values.
- **Functions**:
    - np.array(): To create and manipulate arrays (images).
    - np.mean(), np.std(): To compute statistical metrics from the pixel values for color and texture analysis.

Matplotlib:

- **Purpose**: Matplotlib is primarily used for visualizing the images and results during the development and testing of the system.
- **Functions**:
    - plt.imshow(): To display images.
    - plt.plot(): To visualize color histograms or feature distributions.

scikit-learn:

- **Purpose**: This module is used for applying machine learning algorithms for classification or regression tasks. It is useful for training and testing classifiers that determine fruit ripeness based on extracted features.

7. SYSTEM TESTING

The purpose of testing in your emotion recognition project is to ensure that all components work as expected and to identify and rectify any errors or issues. Testing involves verifying the functionality of different parts of the system, including feature extraction, model training, and predictions.

7.1. Types Of Testings

Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.
Functional testing is centered on the following items:

Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures	: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, system testing ensures the known and the predictable results before conducting the defined and the testing should be done perfectly for each and every module to enhance the model more accurately that it can give good results for any type of the fruit that will be captured using the camera, so that the all predefined processes, and successive processes must be considered for testing. Before the functional testing is complete, additiotnal tests are identified and the effective value of current tests is determined.

System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Objective: To verify that individual components of the Emotion Recognition system function correctlywhen tested in isolation.

Test Strategy and Approach:

- **Manual Testing:** Conducted detailed checks on each function to ensure correct operation.
- **Automated Testing:** Where feasible, automated tests were used to assess core functionalitiesand edge cases.

Key Areas Tested:

- Feature Extraction: Ensured the extract_features function accurately processes images and extracts features like RGB, saturation and the hue values.
- Model Training: Verified that the train_model function successfully trains the OpenCV and handles dataset processing without errors.

- Error Handling: Confirmed that the system provides appropriate error messages for issues such as failed ripeness extraction or model training problems.

Results: All test cases passed successfully. No defects encountered.

Integration Testing

Objective: Integration testing aims to confirm that all components of the Emotion Recognition system, including feature extraction, model training, and prediction, function cohesively. The objective is to ensure that the complete workflow operates smoothly from audio file upload through to emotion prediction and result display.

Test Strategy and Approach:

- Incremental Integration: Tested the integration of various functions, such as feature extraction, model training, and prediction.
- End-to-End Testing: Evaluated the entire workflow from image file upload through to ripeness prediction and result display.

Key Areas Tested:

- Complete Workflow: Ensured that the system processes an image from capture, through feature extraction, model prediction, and displays the result accurately.
- Image Extraction: Verified that the system can handle and extractt various images formats to colour image processing for prediction, and that this process integrates smoothly.
- Error Management: Checked that errors during processing are properly managed and that users receive clear and helpful feedback.

Results: All test cases passed successfully. No defects encountered.

Functional Testing

Objective: To ensure that the system meets its functional requirements and performs as expected from a user perspective.

Test Strategy and Approach:

- Manual Interaction Testing: Simulated user interactions with the camera popup interface to verify functionality.
- Scenario-Based Testing: Executed typical user scenarios to confirm the system’s behavior under various conditions.

Key Areas Tested:

The following are the key areas tested:

- User Interface: Validated that users can capture images, receive accurate ripeness predictions, and view results without issues.
- System Performance: Assessed that the system responds quickly and accurately to user inputs.
- File Format Support: Ensured that the system supports and processes various images formats correctly and handles unsupported formats appropriately.

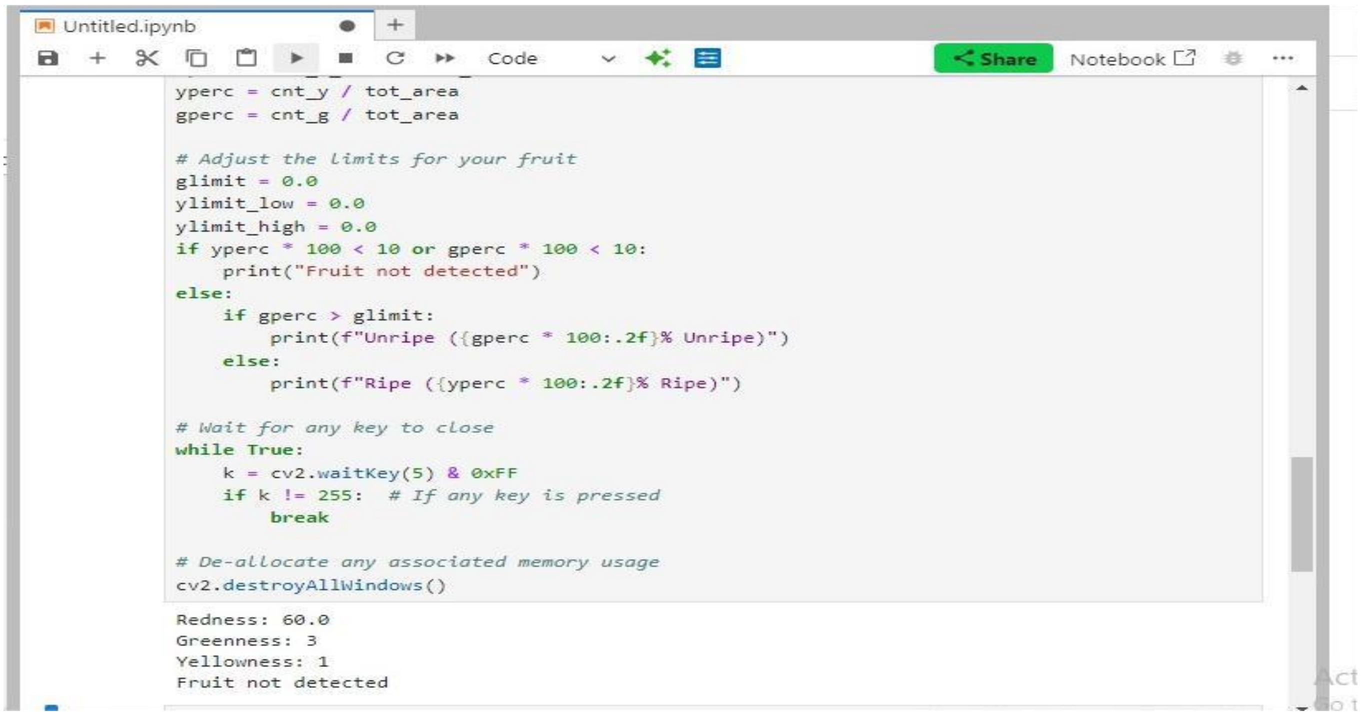
Results: All test cases passed successfully. No defects encountered.

7.2. Test Cases

S. No	Test Case	Expected Result	Result	Remarks (If Fails)
1	Image capturing	The image should be successfully uploaded.	Pass	Executed successfully.
2	Feature Extraction	The system should extract features from the picture.	Pass	Features extracted and stored correctly.
3	Model Training	The model should be trained on the captured dataset without errors.	Pass	Training completed successfully.
4	Colour conversion	The system should correctly classify colurs.	Pass	The colours were correctly predicted as RGB.
5	Colour prediction (Invalid Format)	The system should flash an error for unsupported objects (e.g., .water bottle).	Fail	The system did not properly handle the invalid image format and did not flash an error.
6	Wrong Prediction	The system should predict the correct ripeness for a valid fruit.	Fail	The system predicted the wrong ripeness for the given fruit.

Table 7.2 Test Cases

Test Case 1:

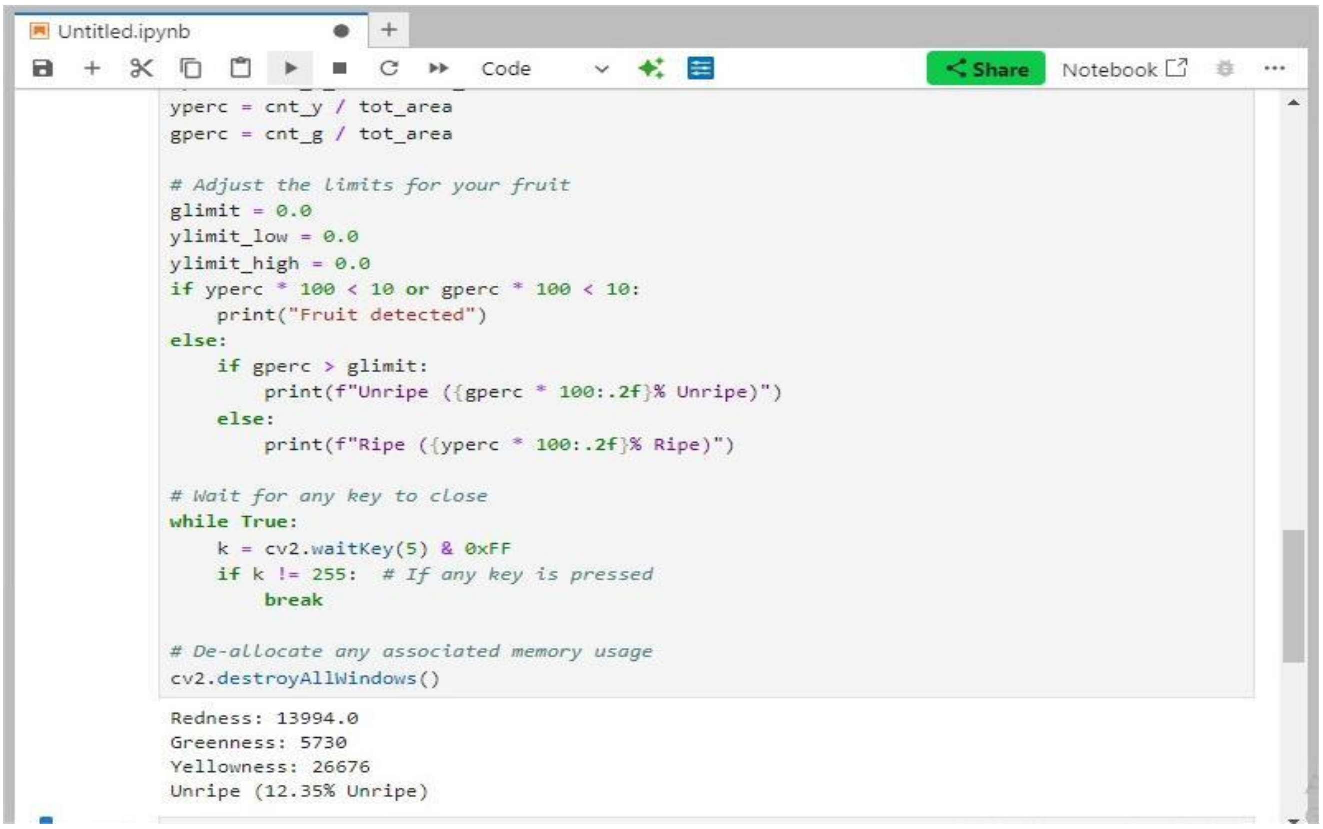


```
yperc = cnt_y / tot_area
gperc = cnt_g / tot_area

# Adjust the limits for your fruit
glimit = 0.0
ylimit_low = 0.0
ylimit_high = 0.0
if yperc * 100 < 10 or gperc * 100 < 10:
    print("Fruit not detected")
else:
    if gperc > glimit:
        print(f"Unripe ({gperc * 100:.2f}% Unripe)")
    else:
        print(f"Ripe ({yperc * 100:.2f}% Ripe)")

# Wait for any key to close
while True:
    k = cv2.waitKey(5) & 0xFF
    if k != 255:  # If any key is pressed
        break

# De-allocate any associated memory usage
cv2.destroyAllWindows()
```

```
Redness: 60.0
Greenness: 3
Yellowness: 1
Fruit not detected
```

Figure 7.2.1: Test Case 1

Test Case 2:

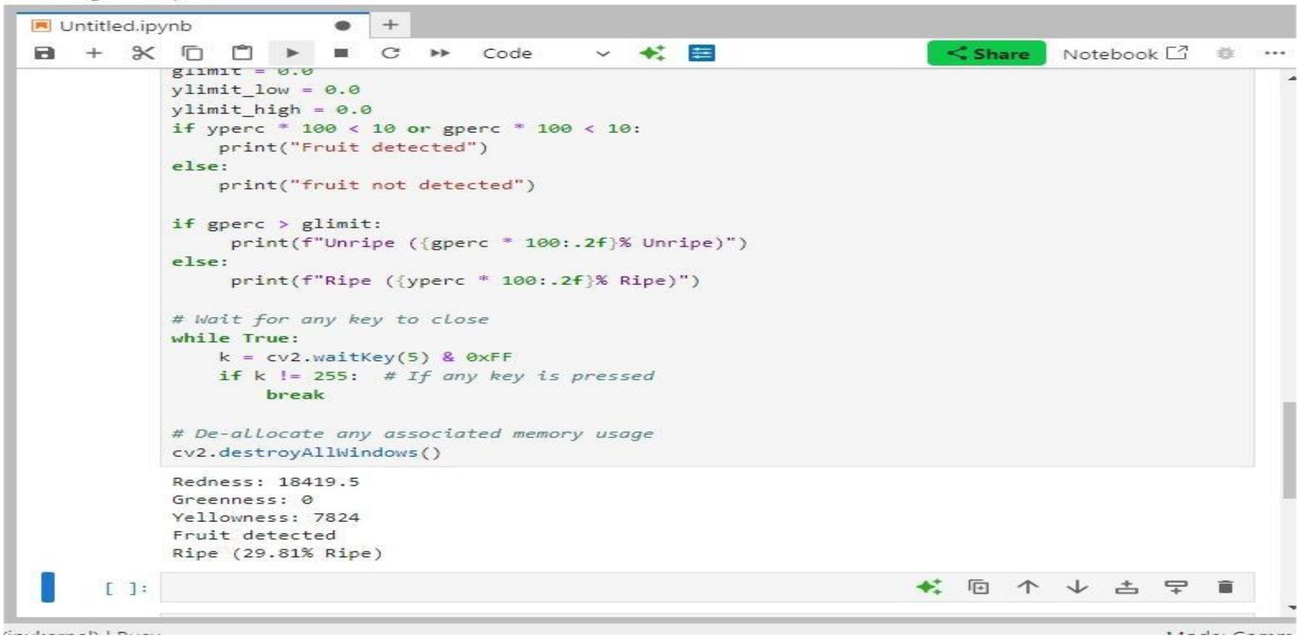


```
yperc = cnt_y / tot_area
gperc = cnt_g / tot_area

# Adjust the limits for your fruit
glimit = 0.0
ylimit_low = 0.0
ylimit_high = 0.0
if yperc * 100 < 10 or gperc * 100 < 10:
    print("Fruit detected")
else:
    if gperc > glimit:
        print(f"Unripe ({gperc * 100:.2f}% Unripe)")
    else:
        print(f"Ripe ({yperc * 100:.2f}% Ripe)")

# Wait for any key to close
while True:
    k = cv2.waitKey(5) & 0xFF
    if k != 255:  # If any key is pressed
        break

# De-allocate any associated memory usage
cv2.destroyAllWindows()
```

```
Redness: 13994.0
Greenness: 5730
Yellowness: 26676
Unripe (12.35% Unripe)
```

Figure 7.2.2: Test Case 2

Test Case 3:



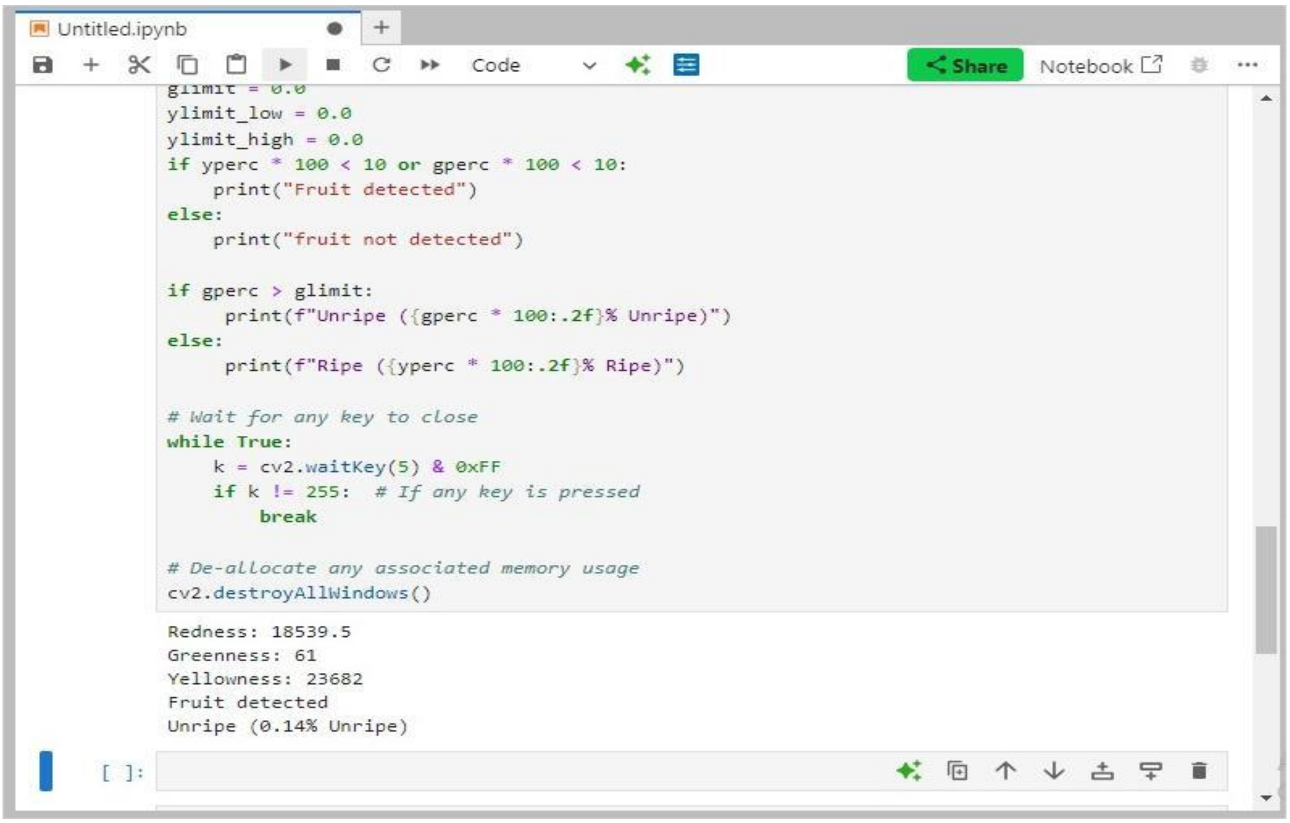
```
glimit = 0.0
ylimit_low = 0.0
ylimit_high = 0.0
if yperc * 100 < 10 or gperc * 100 < 10:
    print("Fruit detected")
else:
    print("fruit not detected")

if gperc > glimit:
     print(f"Unripe ({gperc * 100:.2f}% Unripe)")
else:
     print(f"Ripe ({yperc * 100:.2f}% Ripe)")

# Wait for any key to close
while True:
    k = cv2.waitKey(5) & 0xFF
    if k != 255:  # If any key is pressed
        break

# De-allocate any associated memory usage
cv2.destroyAllWindows()
```

```
Redness: 18419.5
Greenness: 0
Yellowness: 7824
Fruit detected
Ripe (29.81% Ripe)
```

```
[ ]:
```

Figure 7.2.3: Test Case 3

Test Case 4:



```
glimit = 0.0
ylimit_low = 0.0
ylimit_high = 0.0
if yperc * 100 < 10 or gperc * 100 < 10:
    print("Fruit detected")
else:
    print("fruit not detected")

if gperc > glimit:
     print(f"Unripe ({gperc * 100:.2f}% Unripe)")
else:
     print(f"Ripe ({yperc * 100:.2f}% Ripe)")

# Wait for any key to close
while True:
    k = cv2.waitKey(5) & 0xFF
    if k != 255:  # If any key is pressed
        break

# De-allocate any associated memory usage
cv2.destroyAllWindows()
```

```
Redness: 18539.5
Greenness: 61
Yellowness: 23682
Fruit detected
Unripe (0.14% Unripe)
```

```
[ ]:
```

Figure 7.2.4: Test Case 4

Test Case 5:

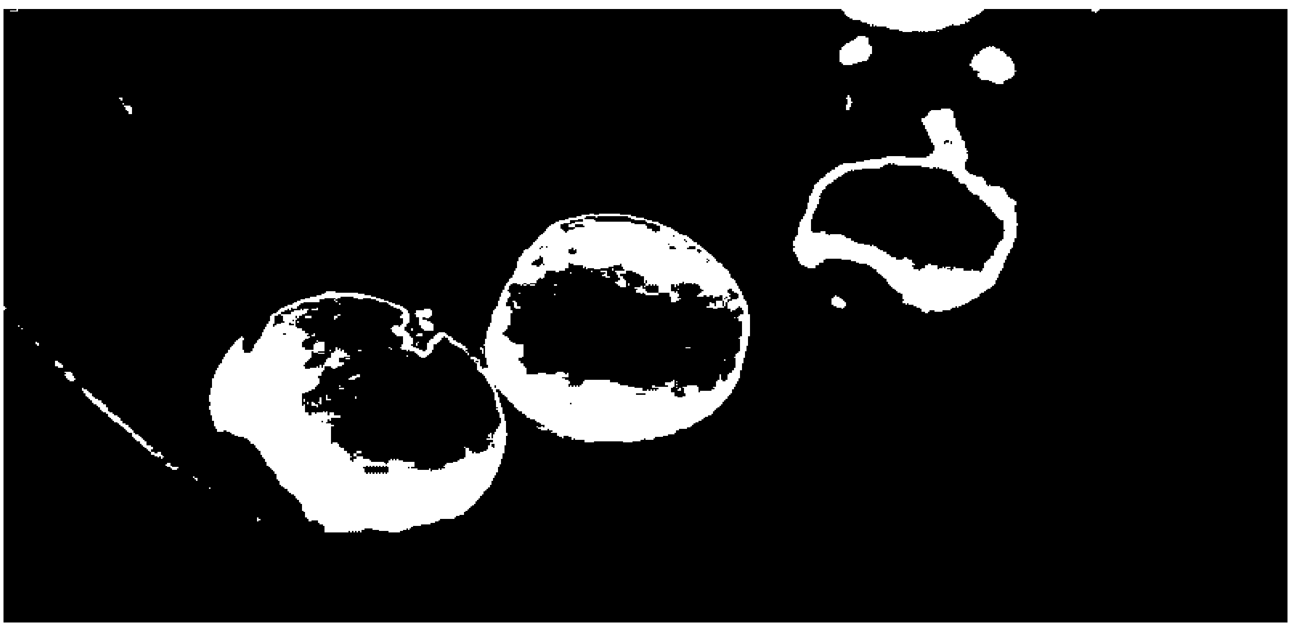


Figure 7.2.5: Test Case 5

Test Case 6:

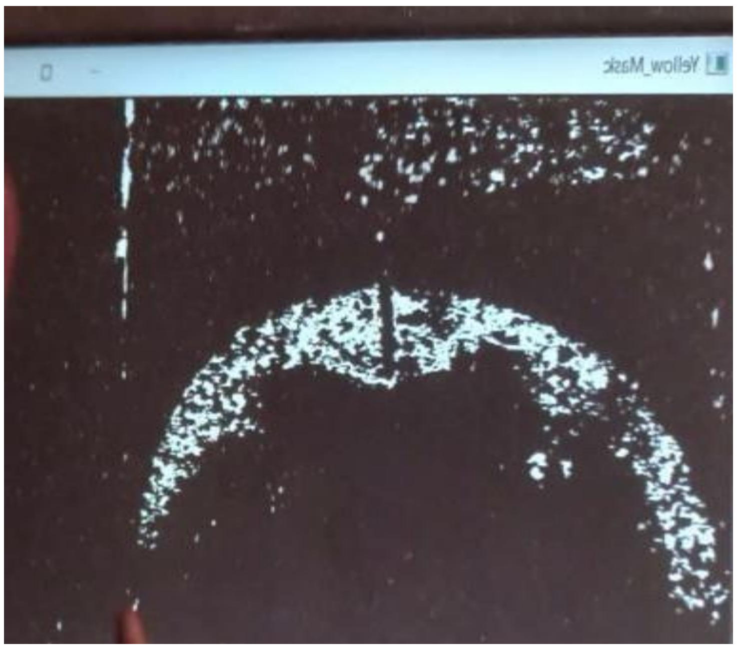
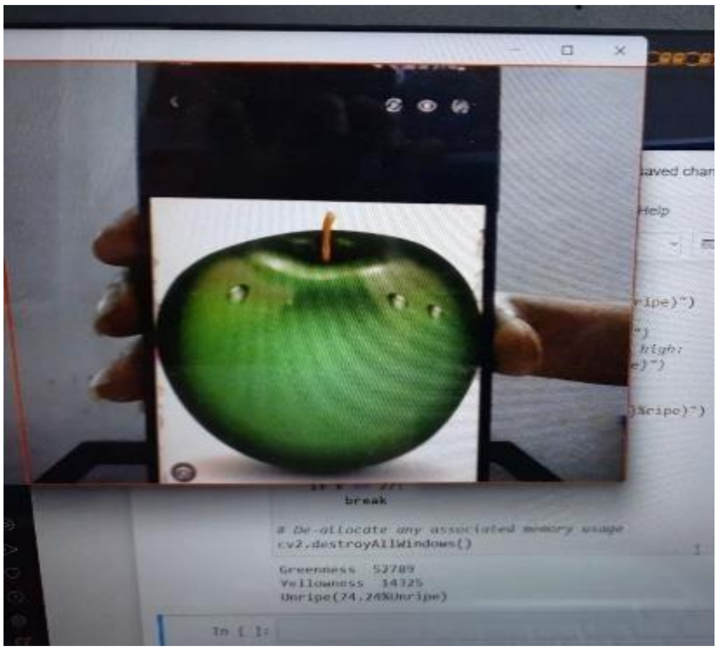


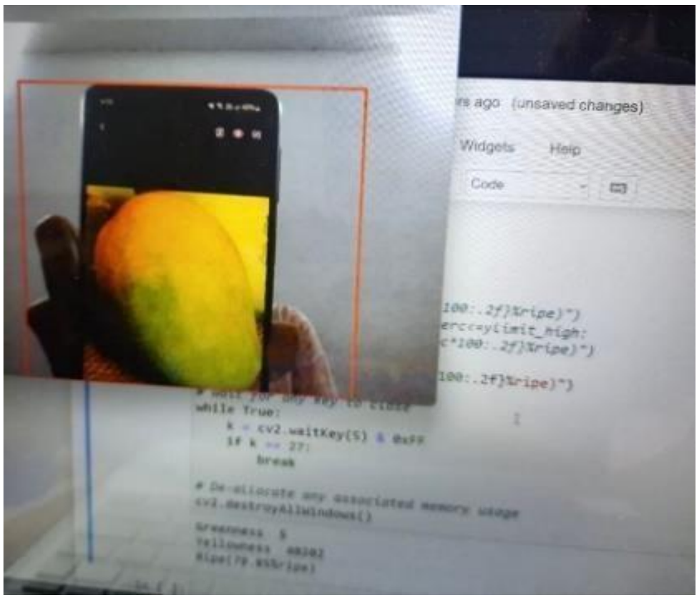
Figure 7.2.6: Test Case 6

8. OUTPUT SCREENS



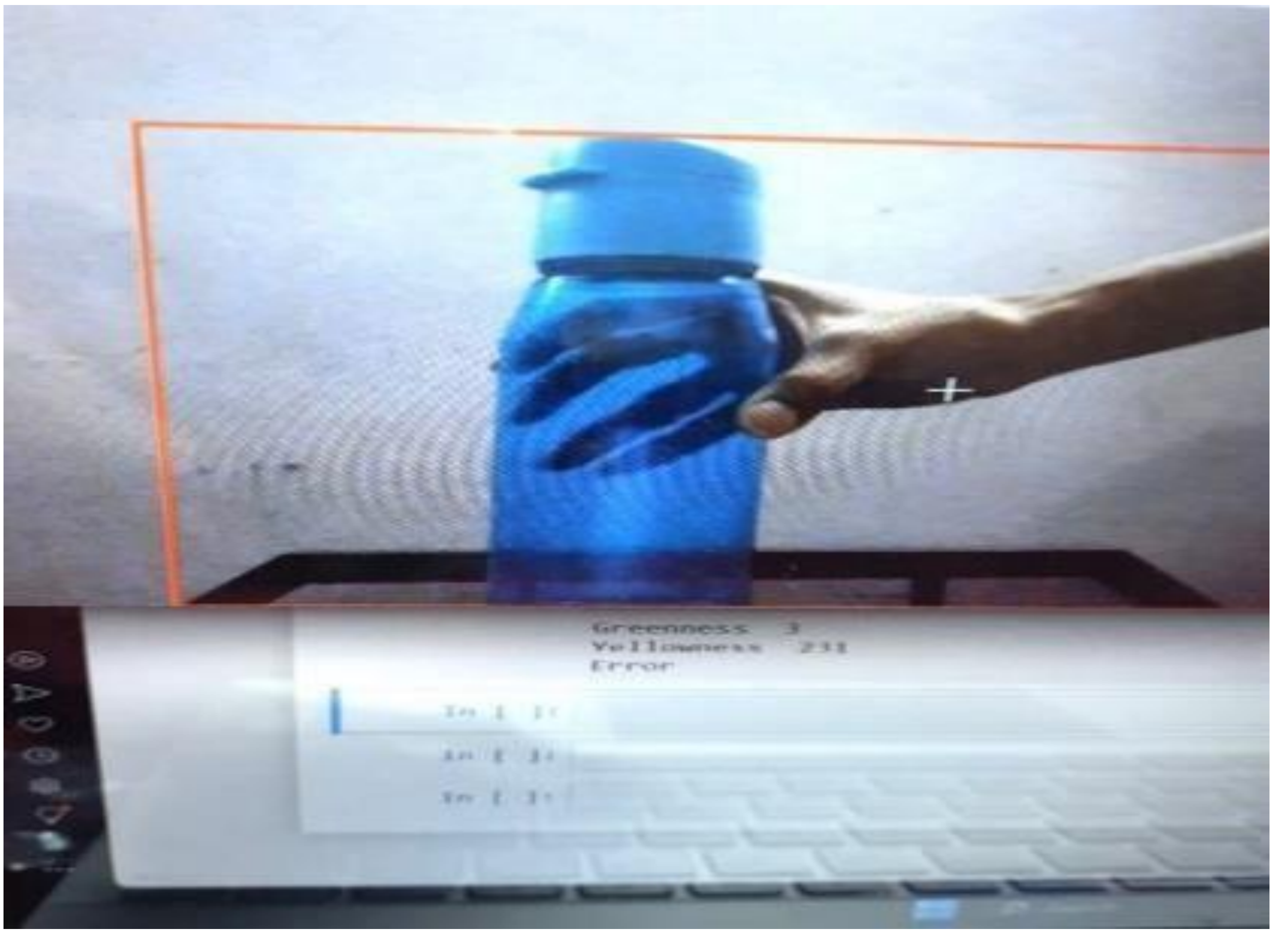
```
        break

# De-allocate any associated memory usage
cv2.destroyAllWindows()

Greenness  52789
Yellowness  14325
Unripe(74.24%Unripe)

In [ ]:
```

Figure 8.1: predicting
ripeness of apple



```
100:.2f}%ripe)")
erc<=ylimit_high:
c*100:.2f}%ripe)")

100:.2f}%ripe)")

while True:
    k = cv2.waitKey(5) & 0xFF
    if k == 27:
        break

# De-allocate any associated memory usage
cv2.destroyAllWindows()

Greenness  5
Yellowness  48202
Ripe(79.85%ripe)
```

Figure 8.2: predicting ripeness of Mango



```
Greenness  3
Yellowness  231
Error

In [ ]:

In [ ]:

In [ ]:
```

Figure 8.3: Output showing model doen’t work for objects

9. CONCLUSION

In conclusion, the trajectory of fruit ripeness detection through OpenCV is poised for significant advancements and widespread application. Integrating cutting-edge technologies such as deep learning models, real-time processing optimizations, and IoT integration holds the promise of revolutionizing fruit management systems. The future of this technology lies in its ability to transcend singular fruit types, leveraging adaptable and generalized models to discern diverse fruit characteristics accurately. This evolution necessitates robust dataset curation encompassing various environmental conditions and ripeness levels. Th e augmentation of ripeness assessment metrics, beyond conventional color and shape analyses, signifies a shift toward holistic evaluation methodologies. Simultaneously, user-centric design principles drive accessibility, empowering stakeholders across industries. Robustness against environmental variables, integration of additional sensory inputs, and optimization for diverse operational landscapes emerge as pivotal focal points for further development. Ultimately, these advancements aim to democratize access to accurate and efficient fruit ripeness detection, catering to both industrial and agricultural sectors, and contributing to enhanced productivity and quality in fruit management practices. As OpenCV continues to evolve, its adaptability and comprehensive toolset are instrumental in propelling the frontiers of fruit ripeness detection technology.

10. FUTURE ENHANCEMENTS

The evolution of fruit ripeness detection via OpenCV presents an exciting trajectory poised for multifaceted advancements. Integrating cutting-edge deep learning models, notably Convolutional Neural Networks (CNNs), stands as a pivotal avenue for elevating the system's capabilities. This integration promises heightened precision in feature extraction and classification, allowing for nuanced assessments of fruit ripeness levels.

Efforts to streamline real-time processing form another critical frontier, where optimization techniques and hardware acceleration are focal points. Such enhancements aim to significantly boost the system's speed and efficiency, thereby enabling swift analyses even in dynamic environments.

Expanding the system's purview beyond singular fruit types represents a fundamental shift toward inclusivity. This necessitates the development of adaptable and generalized models capable of discerning and categorizing various fruit characteristics accurately. Comprehensive and diverse datasets play a crucial role in this pursuit, enabling robust model training by encompassing a spectrum of lighting conditions, fruit sizes, and ripeness gradients.

Augmenting ripeness assessment metrics beyond traditional color and shape analyses marks a strategic leap forward. Exploring additional factors like texture analysis or spectroscopy promises a more holistic evaluation of fruit ripeness, potentially unlocking deeper insights into quality. Ensuring the system's resilience in diverse environmental conditions—adapting to varying lighting scenarios or irregular fruit shapes—remains a cornerstone of further development. Supplementing visual analysis with additional sensory inputs, such as ethylene gas sensors or chemical indicators, holds promise for refining accuracy and precision in ripeness determination.

Lastly, optimizing models for deployment in resource-constrained environments is pivotal for democratizing access to this technology. Aligning with OpenCV's adaptable nature, such optimizations aim to ensure the viability of the system across a spectrum of operational settings, from sophisticated facilities to remote agricultural landscapes.

11. REFERENCES

[1]A. M. de Oca, L. Arreola, A. Flores, J. Sanchez and G. Flores, "Lowcostmultispectral imaging system for crop monitoring,"2018 International Conference on Unmanned Aircraft Systems (ICUAS).

[2] R. F. Maia, I. Netto and A. L. H. Tran, "Precision agriculture using remote monitoring systems in Brazil", 2017 IEEE Global Humanitarian Technology Conference (GHTC).

[3]N. Kitpo and M. Inoue, "Early Rice Disease Detection and Position Mapping System using Drone and IoT Architecture", 2018 12th South East Asian Technical University Consortium (SEATUC).

[4]K. R. Aravind and P. Raja, "Design and simulation of crop monitoring robot for green house", 2016 International Conference on Robotics: Current Trends and Future Challenges (RCTFC).

[5]A. Agarwal, A. K. Singh, S. Kumar and D. Singh, "Critical analysis of classification techniques for precision agriculture monitoring using satellite and drone", 2018 IEEE 13th International Conference on Industrial and Information Systems (ICIIS)

[6]P. Senthil and I. S. Akila, "Automated Robotic Moisture Monitoring in Agricultural Fields", 2018 International Seminar on Intelligent Technology and Its Applications (ISITIA)

[7]M. B. Garcia, S. Ambat and R. T. Adao, "Tomayto Tomahto: A Machine Learning Approach for Tomato Ripening Stage Identification Using Pixel-Based Color Image Classification", 2019 IEEE 11th International Conference on Humanoid Nanotechnology Information Technology Communication and Control Environment and Management
(HNICEM).

[8] Raphael Antoine U. Lim;Patrick Reylie R. Salvador;Euri Andre P.Tiamzon "Real-Time Banana Ripeness Detection and Classification using YOLOv8",2024 9th International Conference on Mechatronics Engineering (ICOM)

[9]G. Saranya;N. Dineshkumar;A. S. Hariprasath;G. Jeevanantham fruit ripeness detection", 2023 International Conference on Computer Communication and Informatics (ICCCI).

[10]Dilip Kumar Jang Bahadur Saini;Sk Hasane Ahammad;Purushottam Das;Ashutosh Bhatt;Pragati Malusare;Sandeep Dwarkanath Pande "Grading of fruit ripeness Using Arduino in IoT",2023 International Conference on Sustainable Emerging Innovations in Engineering and Technology (ICSEIET)

[11]Nurulaqilla Khamis;Hazlina Selamat;ShuwaibatulAslamiah Ghazalli;Nurul Izrin Md Saleh;Nooraini Yusoff "Comparison of Palm Oil Fresh Bunches (FFB) Classification Technique using Deep Learning Method",2022 13th Asian Control Conference (ASCC)

[12]Yvhan Jiao;Qi Zhang;Xinyao Luo;Zhaohua Liang “ based on miniature spectral sensor”, 2021 International Symposium on Computer Technology and Information Science (ISCTIS).

[13]Sheng Tan;Linghan Zhang;Jie Yang “Sensing Fruit Ripeness Using Wireless Signals”,2018 27th International Conference on Computer Communication and Networks (ICCCN)

[14]S.Gayathri;T.U. Ujwala;C.V. Vinusha;N. Rebecca Pauline;D.B. Tharunika “ Papaya Using Deep Learning Approach”,2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)

[15]Akshay A Menon;Arun K Nair;Ananthu Vasudevan;Krishna Das K S;Anjali T “RipId App: Estimator and Object Recognition Application” ,2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)

[16]Siti Azizah;Wahidin;Margaretha Padang;Ledya Novamizanti;Sofia Sa'idah “Identifying the Ripeness and Quality Level of Strawberries Based on YOLOv7-EfficientNet”,2024 International Conference on Data Science and Its Applications (ICoDSA)

[17]Venkatesh;Prajwal K P;Preeti Patil;Priyanka G;Prajwal Poojary;Satish B Basapur “CRipS: Coconut ripeness Stage detection System” ,2023 International Conference on Network, Multimedia and Information Technology (NMITCON)

[18]Luiz E.P. Reis;Ingrid B.D. Souza;Jorge D.R. da Silva;Celso B. Carvalho;Waldir S.S. Júnior;Andrey R.R. Bessa;Diego A. Amoedo;Edma V.C. Urtiga “Wireless IoT System for Detection of Fruit Maturation by Color and Ethylene Gas”,2023 Symposium on Internet of Things (SIoT)

[19]Che-Wei Hsu;Yu-Hsiang Huang;Nen-Fu Huang “Real-time Dragonfruit’s ripeness Classification System with Edge Computing Based on Convolution Neural Network”,2022 International Conference on Information Networking (ICOIN)

[20]B Lakshmi Sirisha;B Jayendra Nayak;B sai Sndhya;J Eswara Rao;M Devisri “ fruit Quality detection Using Deep Learning for Sustainable Agriculture” ,2024 International Conference on Recent Advances in Electrical, Electronics, Ubiquitous Communication, and Computational Intelligence (RAEEUCCI)