| Team Number : | apmcm2309504 |
|---|---|
| Problem Chosen : | A |

2023 APMCM summary sheet

# Image Recognition for Fruit-Picking Robots

By using the detection algorithm YOLOv5(you only look once), the problem of identifying the number of apples in each image can be solved through labelling part of the images which are afterwards used to train the specific.By using the Contour algrithm in the OpenCV Library to identify and draw the contours of each apple, the problem of identifying the position of the apples in each image can be solved.By establishing a maturity-calculation model based on the proportion of specific pixels, the problem of calculating the maturity of apples in each image can be solved.By establishing a mass-estimation model based on the rough volume of apples and its density, the problem of estimating the masses of the apples can be solved.By using the detection algorithm YOLOv5 and training the model using the labelled images, the problem of identifying the apples among the various fruits can be solved.

For problem one, firstly, 30 percent of images from source are selected to be labelled using the software Labelimg to construct a fruit image dataset. Secondly, the YOLOv5s training method is given to obtain an apple quantity detection and classification network model with excellent performance, and the results show that the average detection accuracy of YOLOv5s for fruit quantity detection can reach 94.5%.By applying the trained model, the numbers of each image to be tested are acquired and stored in the text file.

For problem two, firstly, it is necessary to change the original color space RGB to the color space HSV for the purpose of protrude the images of apples.And then, it is important to establish a color-filter model to filter the irrelevant color in the OpenCV Library for the purpose of emphasizing the specific images. In order to find more precise contours of each apples, the algorithms of dilate and erode are chosen.There is a remarkable effect of reducing the noise after using this method.In the end, by using the Contour algorithm, the position of each apple can be obtained.

For problem three, firstly, color space is changed from RGB to HSV for the purpose of protrude the images of apples. And then, we use the algorithm boundingRect in the OpenCV Library to draw each contour’s bounding rectangle.A threshold is set here for detecting the apples as the bounding rectangle of apple tends to be a square.If the ratio of length to width is within the threshold which means the ratio is no more than 1.2 and no less than 0.8, it is reasonable to consider it as an apple. In the end, a pixel-estimate model is established to calculate the number of specific pixel and quantify the maturity of each apple. The more specific pixels there is, the more possibility there is for an apple to be mature.

For problem four, firstly, it is also important to change the color space from RGB to HSV. And then, it is important to establish a color-filter model to filter the irrelevant color applying the algorithm of dilate and erode in the OpenCV Library for the purpose of emphasizing the specific images. In the end, by establishing a mass-estimate model which contains contourArea algorithm in the OpenCV Library, the masses of apples can be obtained.

For problem five, to improve the precision and speed of detection,propose a fruit detection and classification scheme which uses the detection algorithm YOLOv5 based on fruit images. Firstly，five kinds of fruit images including apples，carambolas，pears, plums and tomatoes are obtained, and fruit images are labeled by the LabelImg tool to construct a fruit image dataset. Secondly， the YOLOv5x training method is given to obtain a fruit detection and classification network model with excellent performance. The results show that the average detection accuracy of YOLOv5x for fruit detection and classification can reach 98.3%, which has a certain degree of advancement. The use of YOLOv5 algorithm to detect fruit can provide new ideas for fruit quality detection in actual agricultural scenarios[1].

**Keywords:** YOLOV5, OpenCV,object detection,deep learning

# Contents

# 1 Introduction

## 1.1 Problem Background

China is a major producer of apples, and the total area of apple cultivation is increasing year by year. With the further optimization of China's agricultural structure, the planting environment for apples has been improved, and their production efficiency has also been significantly improved. In the apple production process, fruit picking accounts for 40-50% of the total workload, making it the most concentrated, labor-intensive, complex, and dangerous link in the production process. The quality of fruit picking directly affects the storage and sale of apples, thereby affecting their market value and economic benefits. Due to the implementation of the family planning policy, a large amount of labor has also been transferred to the service and manufacturing industries, leading to an increase in labor costs and increasing the difficulty of timely apple picking. From a development perspective, achieving mechanized, automated, and intelligent apple picking is beneficial for ensuring fruit quality, reducing costs during apple production, and improving the market competitiveness of harvested apples. It has significant practical significance for the overall level of agriculture in China. So designing and implementing an automatic method for picking apple fruits is the trend, and the key link is how to use machine vision technology to achieve precise recognition and positioning of the fruits.

Nowadays, due to the increase in human and material costs, **machine vision based automated apple fruit picking technology** has become a focus of attention. In the natural environment, the growth status of apples varies greatly due to the influence of natural conditions such as seasons, growing regions, and weather. Fruits are often obstructed by branches and leaves, and the background of apple images taken is complex, which may include sky, branches and leaves, land, etc. In response to the above problems, this article proposes an OpenCV and YOLOv5 based method for identifying and locating apple targets in complex backgrounds, which can effectively reduce the impact of natural conditions on apple fruit recognition and positioning, effectively solve the problem of occlusion overlap in the recognition and positioning process, and improve the speed and accuracy of recognition and positioning.

## 1.2 Problem Requirements

(1)Question 1: Based on the image dataset of harvest-ready apples provided in Attachment 1, extract image features, establish a mathematical model, count the number of apples in each image, and draw a histogram of the distribution of all apples in Attachment 1.

(2)Question 2: Based on the image dataset of harvest-ready apples provided in Attachment 1, identify the position of the apples in each image with the left bottom corner of the image as the coordinate origin, and draw a two-dimensional scatter diagram of the geometric coordinates of all apples in Attachment 1.

(3)Question 3: Based on the image dataset of harvest-ready apples provided in Attachment 1, establish a mathematical model, calculate the maturity of apples in each image, and draw a histogram of the maturity distribution of all apples in Attachment 1.

(4)Question 4: Based on the image dataset of harvest-ready apples provided in Attachment 1, calculate the two-dimensional area of the apples in each image with the bottom left corner of the image as the coordinate origin, estimate the masses of the apples, and draw a histogram of the mass distribution of all apples in Attachment 1.

(5)Question 5: Based on the dataset of harvested fruits' images provided in Attachment 2, extract image features, train an apple recognition model, identify the apples in Attachment 3, and draw a distribution histogram of the ID numbers of all apple images in Attachment 3.

# 2 Problem Analysis

Five questions are related to the object detection with complicated background such as leaves, sky, trees and so on of the essence, while each question possesses different research focuses.First, it is prominent to process each image before conducting the detection. We consider obtaining the images of apples and enhancing the image by the approaches such as transforming the color space and so on.And then, the contours of each apple is needed for further task, and OpenCV Library is widely used as it show absolutely efficient when processing images.We could segment the images of apples using the contours.Finally, fit circles corresponding to all the apples are determined,and through these circle we could solve all these five questions. However, while conducting the aforementioned process we found that certain questions is not suitable or less suitable than using the YOLOv5 algorithm. When it comes to the question five, this point is clearly proved as the trained model based on YOLOv5 manifest a remarkable performance with an accurancy of over 98 percent. Hence, we decide to utilize YOLOv5 algorithm to address question 1 and question 5, and solve question 2,3,4 mainly based on OpenCV Library.

## 2.1 Analysis of Question 1

Question 1 is a typical object detection task with complicated background such as sky, leaves, trees and so forth. We consider using YOLOv5 algorithm is relatively suitable as it has a well-performed accuracy and speed and most importantly[2], it has better performance on small objects. We need a considerable datasets with all kinds of apples. We choose to select 30 percent of images in Attachment 1 to train the model and that is 60 images, and 10 percent of images to test the model.By using the software tool, a fruit image dataset is constructed.Then we consider start training the model with default parameters.After we acquire the trained model, the data in Attachment 1 is then tested using this model and manually, we select some of the results to verify the accuracy. Taking Precision and Recall as the measurement, we tend to acquire the total numbers of apples in each image by adding some code into the detect file.By using the Pyecharts, we could get the histogram of the numbers of apples in each image.

## 2.2 Analysis of Question 2

Question 2 is to acquire the position of each apple. We intend to solve this problem mainly based on the OpenCV Library. As the background is complicated, we need some Pre-processing approached to extract the features of apple.Firstly, changing the color space from original RGB to HSV will help as it protrude the image of apples.And then, binarization is applied to process the image by setting the threshold to change the image, inRange algorithm is needed for this purpose.After that, it is important to reduce the noise of the images by adapting some algorithms such as erode and dilate.When finishing the pre-processing, we look to find the contours of the apple through the algorithm of findContours in the OpenCV Library.When each contours corresponding to apples is defined, we could obtain the position of each apple by calculating the centroid of each circle.Finally, the scatter diagram could be drawn with these data.

## 2.3 Analysis of Question 3

Question 3 is to predict the maturity of each apples and draw the histogram of the maturity distribution of all apples.The pre-processing is similar to question 2, the only difference is that question 3 is inclined to get the maturity of each apple after determining the position of apples.We consider that the higher the maturity of an apple, the more red pixels it contains.So we base on this assumption, we could calculation the proportion of red pixels within all the different pixels.To achieve that, we use inRange algorithm and adapting the corresponding thresholds.Finally, we can draw the histogram of the maturity distribution through these data.

## 2.4 Analysis of Question 4

Question 4 is to predict the masses of the apples.There is an important assumption:the mass of apple is directly proportional to its size in the image.Similarly, the pre-processing is equal to question 2.We could get the contours of each apple, so we could use contourArea algorithm in OpenCV Library to calculate its area.Once the area of one apple is determined, its volumn is also known.From the Internet we could get the rough density of apple,and then, we could correspondingly obtain the mass of the very apple.Finally, the histogram of the mass distribution of all apples can be drawn through these data.

## 2.5 Analysis of Question 5

Question 5 is to identify the apples within different fruit types which contain apple,pear,carambola,plum,tomato. This question is suitable to utilize YOLOv5 algorithm. Firstly, five kinds of fruit images are collected, and fruit images are labeled by the LabelImg tool to construct a fruit image dataset. Secondly，the YOLOv5x training method is given to obtain a fruit detection and classification network model with excellent performance. By training the model, we could identify and classify the fruit in images from Attachment 3.Finally, by adding some subsidiary code we could obtain the numbers of apples in all images.And distribution histogram of the ID numbers of all apple images can be drawn.

# 3 Model Assumption

(1) Assume that the mass of an apple is directly proportional to its size in the image.

(2) Assume that a apple with higher maturity possesses a higher ratio of red pixels to the whole pixels.

(3) Assume that the green apples from Attachment 1 are all immature.

# 4 Symbol Description

| Symbol | Explain |
|---|---|
| $R$ | *Red Channel Brightness* |
| $G$ | Green *Channel Brightness* |
| $B$ | Blue *Channel Brightness* |
| $R'$ | *Red Channel Brightness* After *Normalized* |
| $G'$ | *Green Channel Brightness* After *Normalized* |
| $B'$ | *Blue Channel Brightness* After *Normalized* |
| $C_{max}$ | *The maximum pixel value in channels R, B, and G* |
| $C_{min}$ | *The minimum pixel value in channels R, B, and G* |
| $H$ | *Hue* |
| $S$ | *Saturation* |
| $V$ | *Value* |

| Symbol | Explain |
|---|---|
| $TP$ | $Correct\ Detection$ |
| $FP$ | $Error\ Detection$ |
| $FN$ | $Positive\ Class\ Judged\ As\ Negative\ class$ |
| $AP$ | $Average\ Accuracy$ |
| $mAP$ | $Mean\ of\ the\ Average\ Accuracy$ |
| $z$ | $Aspect\ Ratio$ |
| $w$ | $Width$ |
| $h$ | $Height$ |
| $m$ | $Maturity$ |
| $red$ | $Number\ Of\ Red\ Pixels$ |
| $green$ | $Number\ Of\ Green\ Pixels$ |
| $S$ | $Area$ |
| $r$ | $Radius$ |
| $v$ | $Volume$ |
| $W$ | $Weight$ |
| $\rho$ | $Density$ |

# 5 Model Establishment

## 5.1 Image Pre-processing

During extracting the features of each image, images have complex background such as trees, leaves, sky and so on. So we need the pre-processing to better extract the image of apples for futher researches on specific question[3].

### 5.1.1 Convert Color Space

In machine vision technology research, in order to achieve better processing results, images are usually converted between RGB, HSV, and Lab color spaces to select suitable color space models, achieving the goal of fast and accurate recognition of targets or other areas. After comparison, we think that the HSV color space is most suitable for apple detection.

*Step 1: Normalize the various channel components of apple images*

$$R' = R/255$$
$$G' = G/255$$

$$B' = B/255$$

*Step 2: Let Cmax be the maximum pixel value in channels R, B, and G, and Cmin be the minimum pixel value in channels R, B, and G, Δ The difference between the maximum and minimum values*

*Step 3: Calculate the H, S, and V components of the converted apple image based on the above parameters*

$$H = \begin{cases} 0 & \Delta = 0 \\ 60 * \left(\frac{G - B}{\Delta} + 0\right) & C_{max} = R \\ 60 * \left(\frac{B - R}{\Delta} + 2\right) & C_{max} = G \\ 60 * \left(\frac{R - G}{\Delta} + 4\right) & C_{max} = B \end{cases}$$

$$S = \begin{cases} 0 & C_{max} = 0 \\ \frac{\Delta}{C_{max}} & C_{max} \neq 0 \end{cases}$$

$$V = C_{max}$$

The image transforming from color space RGB to color space HSV can be obeserved from Figure 1 and Figure 2.

Figure 1 Original RGB image

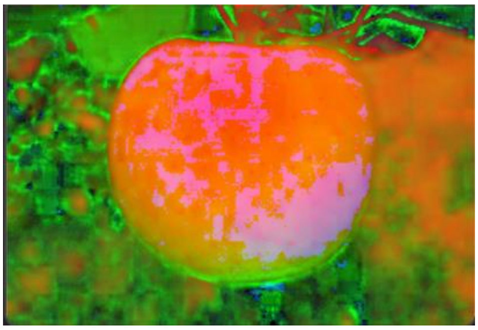

Figure 2 Converted HSV image

### 5.1.2 Image Binarization Processing

After HSV conversion, the color of the apple becomes more prominent, but it is still difficult for the computer to extract the outline. Therefore, in order to make the outline of the apple more prominent, the image is binarized. First, a color range is set, and the pixels that meet this color range are turned into white, while the remaining pixels are turned into black. In this question, it is to turn the red pixels into white and the remaining pixels into black.Observe Figure 3 and Figure 4 we could see the effect of this processing.

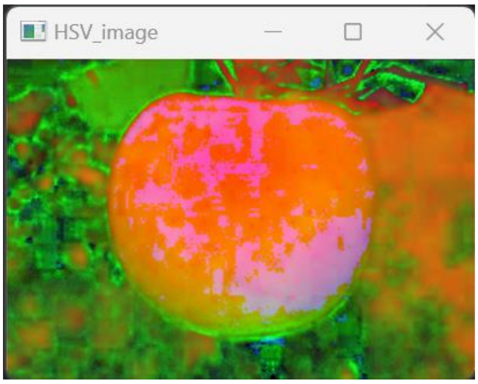


Figure 3 HSV Color Space Image

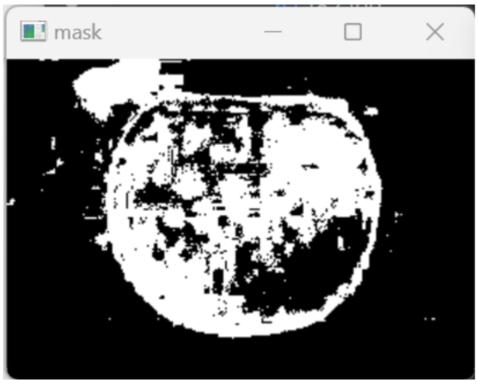


Figure 4 Image after Binarization Processing

### 5.1.3 Image Morphological Processing

The specific idea of morphological processing is to delete, modify, and extract the corresponding shaped structures in the image through fixed structural elements, in order to achieve the purpose of image recognition and analysis.

(1) **Corrosion**: In morphological operations of binary images, corrosion is used to eliminate boundary points and connected areas with smaller dimensions than structural elements in the image. It can also remove some granular noise.

$$p\_e\ (x,y) = (f \ominus b)(x,y)$$

(2) **Inflation**: Inflation is the process of merging all pixels that can be touched by the target area in an image into that area, while reducing the size of the holes in the target area. It is generally used to eliminate the holes inside the apple target area.

$$p\_d\ (x,y) = (f \oplus b)(x,y)$$

Observe Figure 5 and Figure 6 we could see the effect of this process.

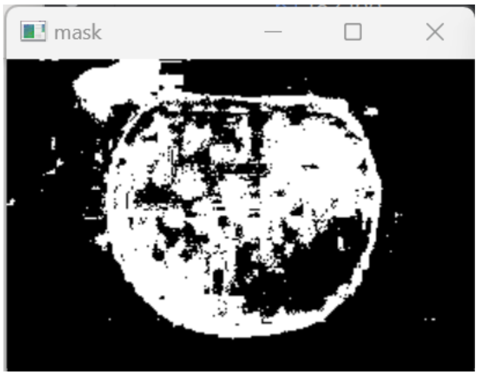


Figure 5 Image after Binarization Processing

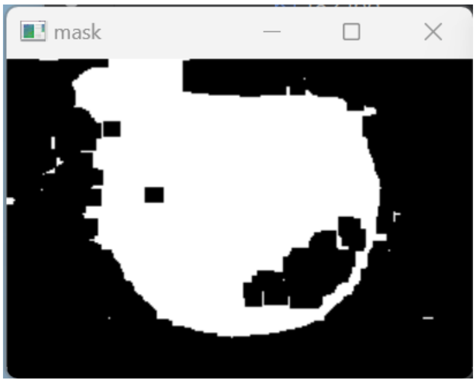


Figure 6 Image After Morphological processing

## 5.2 Contour Detection

The findContours function in OpenCV is used to locate and identify the contours within an image. It operates by first binarizing the input image, converting the pixel values to either 0 or 255. Then, it utilizes edge detection algorithms such as Canny edge detection to identify the contours within the image. Each contour is represented as a set of points. Finally, the function returns all the identified contours, which can be further processed, sorted, and filtered as needed. This function is widely used in various image processing and computer vision applications such as object detection, shape analysis, and image segmentation.

Figure 7 Original Image

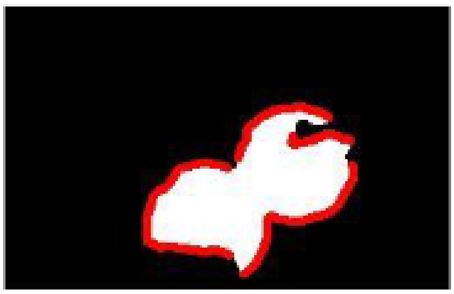

Figure 8 images processed by findContour method

# 6 Problem Solving

## 6.1 For Question 1

### 6.1.1 Parameter setting

The experiment used a Stochastic Gradient Descent (SGD) optimizer to train the network, modified the target type parameter nc (number of classes) to 1, and named it apple. Input an image size of 384 × 384, with 300 iterations, batch size of 8, momentum factor of 0.937, and weight decay coefficient of 0.000 5. Using automatic anchor detection and Mosaic data augmentation strategy, the initial learning rate is 0.001.

### 6.1.2 Measurement Index

Using Precision and Recall as evaluation indicators for fruit classification, expressed as:

$$\begin{cases} Precision = \dfrac{TP}{TP + FP} \\ Recall = \dfrac{TP}{TP + FN} \end{cases}$$

In the formula: TP is the positive class judged as positive (correct detection), FP is the negative class judged as positive (incorrect detection), and FN is the positive class judged as negative. Use the average precision mean (mAP:0.5) when IoU is set to 0.5, the average mAP (mAP:0.5:0.95) with the same IoU threshold (from 0.5 to 0.95, step size 0.05), accuracy, and recall as model evaluation metrics. The average precision (AP) represents the area under the Pre precision Recall curve. The average precision for each class of the image is calculated as mAP, using the following formula:

$$mAP = \frac{\sum AP}{N}$$

In the formula, N represents the total sum of different categories of the image.

### 6.1.3 Experiment Results

Train the model in the above parameter environment and visualize the training results, including model accuracy, recall, and mAP_ 0.5, mAP_ 0.5:0.95, as shown in Figure 9 and Figure 10. It can be seen that when the number of iterations is less than 200, the accuracy, recall, and mAP are improved_ 0.5 and mAP_ 0.5:0.95 both increase rapidly with the increase of iteration times. When the number of iterations exceeds 200, various performance indicators gradually stabilize, and the model accuracy and recall remain stable at over 90%, with an average accuracy of 95.3%. The model performs relatively well during the training phase.

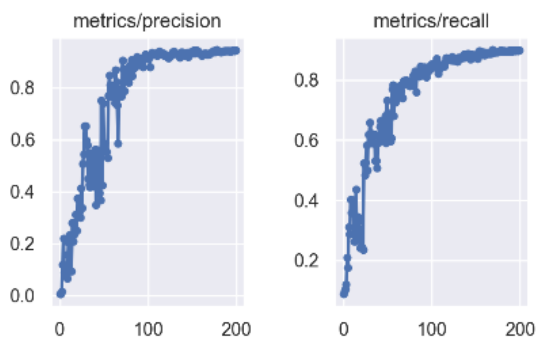


Figure 9 Precision and Recall

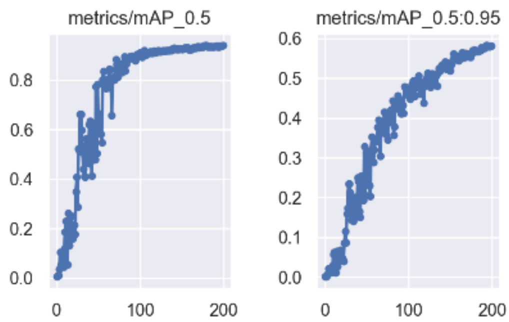


Figure 10 mAP_0.5 and mAP_0.5:0.95

After training a model with high perfomances, we used it to count the number of apples in each image,and the histogram of the distribution of all apples can be observed through Figure 11.

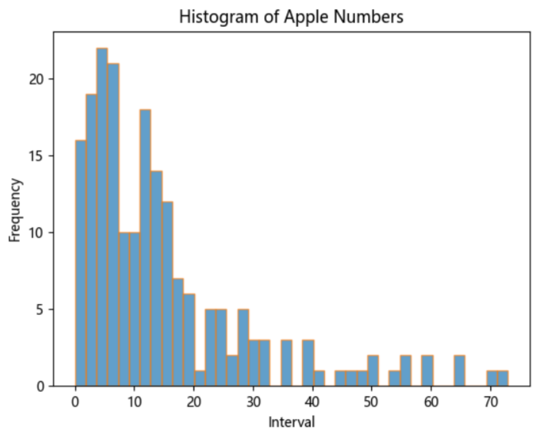


Figure 11 Histogram of the Distribution of All Apples

## 6.2 For Question 2

### 6.2.1 Image Pre-processing

We assume that a apple with higher maturity possesses a higher ratio of red pixels to the whole pixels.For the purpose of extracting the very images of each apple better, we processed the image precedentedly using approaches such as converting the color space and so on.

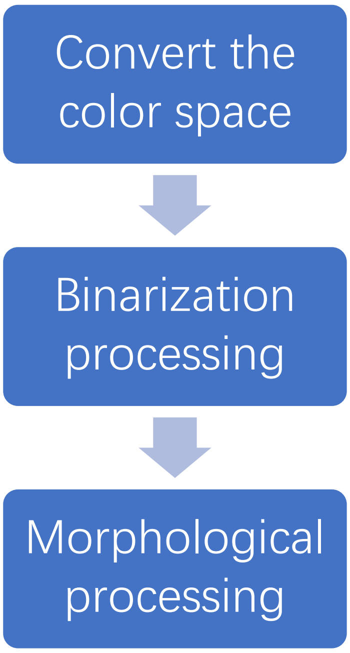


Figure 12 The Flow Chart of Pre-processing

The change of image after pre-processing can be observed through Figure 5 and Figure 6.

### 6.2.2 Identify the Position

*Step 1*:

**Contour detection**. Use the findContours function in opencv to find contours on the processed mask. The principle is to find continuous points in the image that share the same color and connect them to form a curve[4].

*Step 2:*

**Centriod estimation**.For each contour, use the moments function to calculate the centroid of the contour, which is the center point of the apple. Use the centroid to calculate the coordinates of the contour in the image.

Due to the requirement for the coordinate origin to be the bottom left corner in the question, and the default coordinate origin for computer image processing is the top left corner, coordinate system conversion is required to extend the Y-axis upwards.

*Step 3:*

Collect geometric coordinates and draw scatter plots.

Observe Figure 13 we can obtain the position of all the apples.

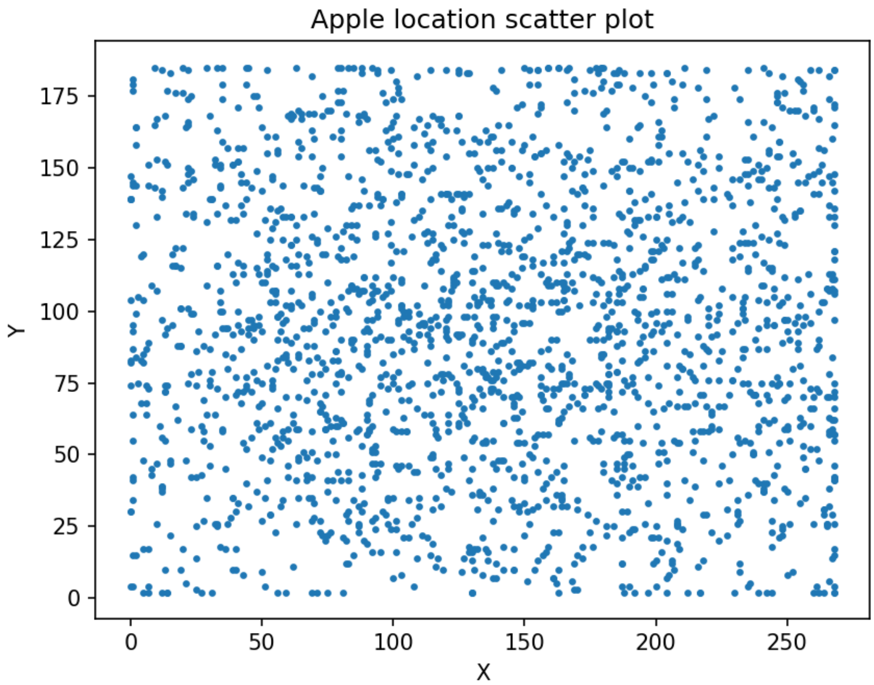


Figure 13 Scatter Diagram of the Geometric Coordinates of All Apples

## 6.3 For Question 3

### 6.3.1 Image Pre-processing

The pre-processing for the question 3 is as same as that of the question 2.

### 6.3.2 Maturity Estimation

The main characteristic of apple ripening is the color turning red, so when calculating maturity, this question mainly considers the proportion of apple red pixels in the image.

*Step 1*:

**Excluding interference**: Due to the presence of a small amount of green pigment in mature apples, the green part is easily confused with the leaves, so it is necessary to eliminate interference from the leaves as much as possible. Considering the significant difference between the shape of apples and their leaves, apples are more rounded and three-dimensional, while leaves are more slender. Therefore, this question uses morphological parameters of contours to distinguish between leaves and apples. This question establishes an bounding rectangle for the contour, with a width of w and a height of h. Describe the shape of the contour by calculating the aspect ratio z.

$$z = \frac{w}{h}$$

When the width and height of the bounding rectangle are approximately equal, the probability of the inscribed image being circular is higher and can be considered as an apple. When the width to height ratio of the bounding rectangle is large or small, it indicates that the shape is slender, usually consisting of leaves.Here we assume that if z is no more than 1.2 and no less than 0.8, the object in this rectangle is an apple.

*Step 2*:

**Maturity Estimation**:Read the HSV values of all pixels in the image using OpenCV, and use the sum function in the numpy library to sum the number of pixels for a specific color value. Obtain the number of red pixels r and the number of green pixels g, and use the formula to calculate the maturity m.

$$m = \frac{red}{red + green}$$

Step 3:

Bring the image in Attachment 1 into the model and use Python programming to solve it. Obtain the distribution histogram of apple ripeness as shown in the Figure 14.

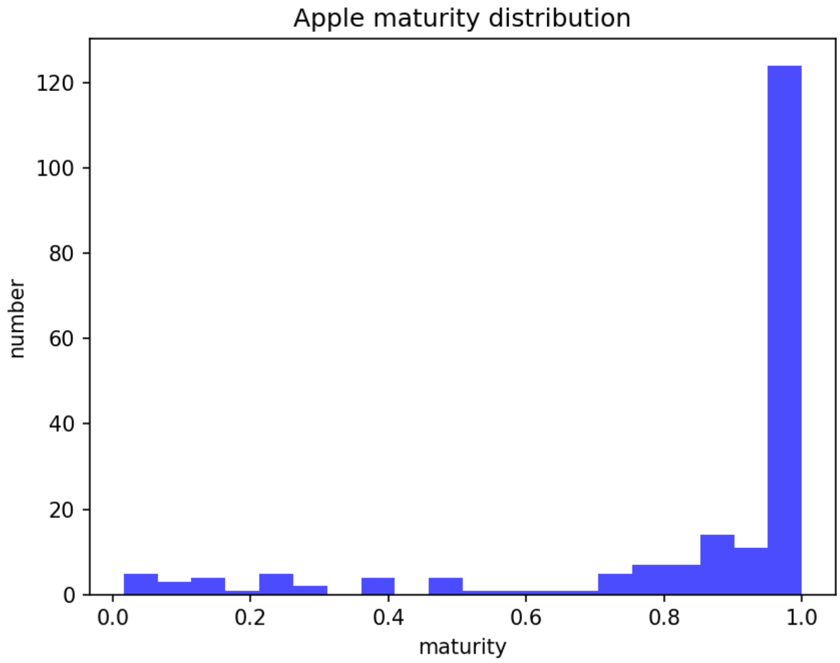


Figure 14 Histogram of the Maturity Distribution of All Apples

## 6.4 For Question 4

### 6.4.1 Image Pre-processing

The pre-processing for the question 4 is as same as that of the question 2.

### 6.4.2 Mass Estimation

The estimation of apple quality mainly relies on the volume and density of the apple, so it is necessary to process the two-dimensional area identified in the image to obtain a three-dimensional apple volume. Mature apples have a plump appearance and can be approximately spherical. Adopt

$$S = \pi r^2$$

$$r = \sqrt{\frac{S}{\pi}}$$

Obtain the 2D contour radius r of the apple on the image, and then use the

$$V = \frac{4}{3}\pi r^3$$

Calculate the volume of the three-dimensional sphere approximated by the contour, and multiply the volume by density to obtain the mass

$$W = V\rho$$

According to the query, the density of apples is 0.9g/cm^3 on average. By inputting the image in Attachment 1 into the model and using Python programming to solve, the histogram of apple quality distribution is obtained as Figure 15:

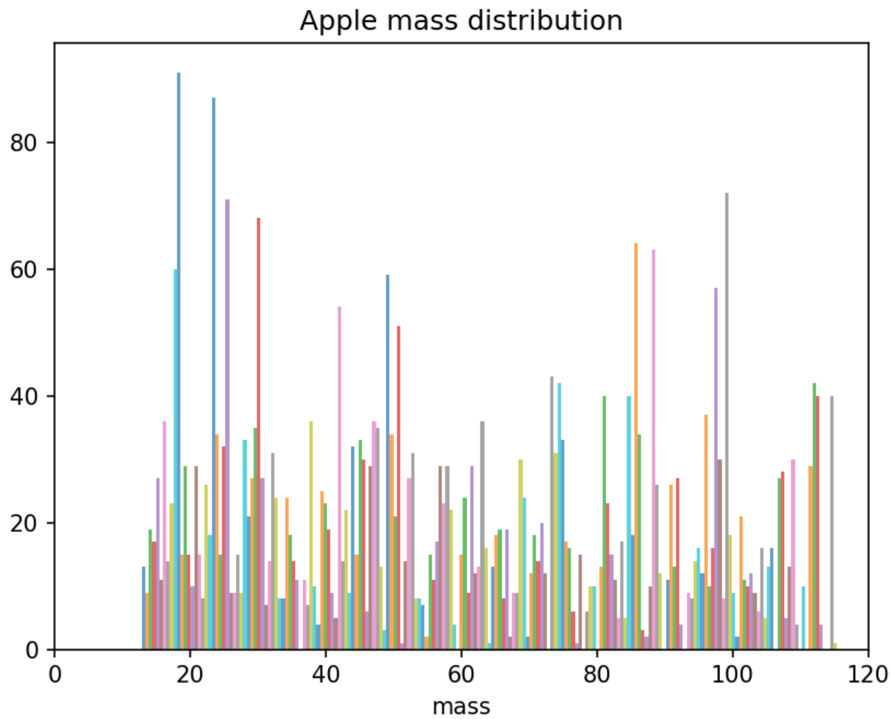


Figure 15 Histogram of the Mass Distribution of All Apples

**Note that** this question is solved based on the **assumption** that mass of an apple is directly proportional to its size in the image.Though it might sound unreasonable as a same apple in reality with different distance to the camera, through this model, will be result in various mass estimation, sometimes this deviation is even unaccepted.

**However,** we incline to view that this assumpion take effects under certain circumstances. For exaple, the Fruit-Picking Robots only pick the apples on a same tree in one task, which means that all the apples on this tree nearly have a same distance to the robot, that is to say, the effect of distance could be ignored roughly. By setting the parameter or using the empirical approach, it is still useful to estimate one apple's mass through its size in the camera of robot.

## 6.5 For Question 5

### 6.5.1 Parameter setting

The experiment used a Stochastic Gradient Descent (SGD) optimizer to train the network, modified the target type parameter nc (number of classes) to 1, and named it apple. Input an image size of 384 × 384, with 300 iterations, batch size of 8, momentum factor of 0.937, and weight decay coefficient of 0.000 5. Using automatic anchor detection and Mosaic data augmentation strategy, the initial learning rate is 0.001.

### 6.5.2 Measurement Index

Using Precision and Recall as evaluation indicators for fruit classification, expressed as:

$$\begin{cases} Precision = \dfrac{TP}{TP + FP} \\ Recall = \dfrac{TP}{TP + FN} \end{cases}$$

In the formula: TP is the positive class judged as positive (correct detection), FP is the negative class judged as positive (incorrect detection), and FN is the positive class judged as negative. Use the average precision mean (mAP:0.5) when IoU is set to 0.5, the average mAP (mAP:0.5:0.95) with the same IoU threshold (from 0.5 to 0.95, step size 0.05), accuracy, and recall as model evaluation metrics. The average precision (AP) represents the area under the Pre precision Recall curve. The average precision for each class of the image is calculated as mAP, using the following formula:

$$mAP = \frac{\sum AP}{N}$$

In the formula, N represents the total sum of different categories of the image.

### 6.5.3 Experiment Results

Train the model in the above parameter environment and visualize the training results, including model accuracy, recall, and mAP_ 0.5, mAP_ 0.5:0.95, as shown in Figure 12 and Figure 13. It can be seen that when the number of iterations is less than 200, the accuracy, recall, and mAP are improved_ 0.5 and mAP_ 0.5:0.95 both increase rapidly with the increase of iteration times. When the number of iterations exceeds 200, various performance indicators gradually stabilize, and the model accuracy and recall remain stable at over 90%, with an average accuracy of 96.7%. The model performs relatively well during the training phase.

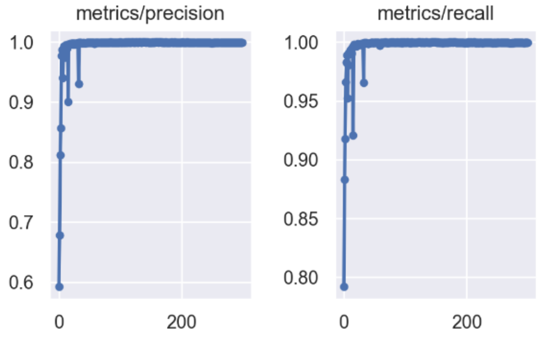


Figure 16 Precision and Recall

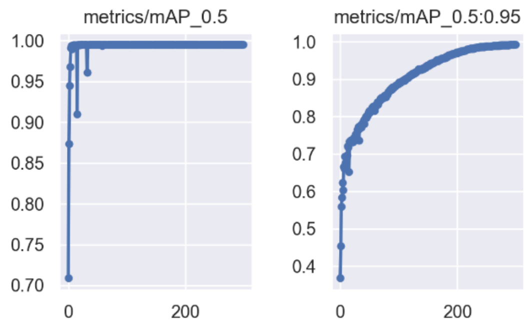


Figure 17 mAP_0.5 and mAP_0.5:0.95

After training a model with high perfomances, we used it to identify the apples in Attachment 3,and the distribution histogram of the ID numbers of all apple images can be observed through Figure 14.

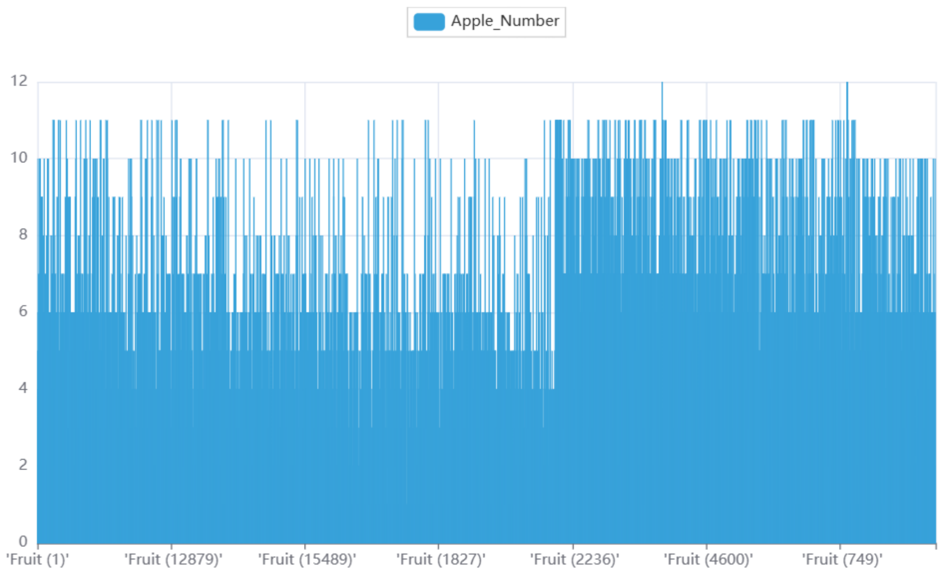


Figure 18 Distribution Histogram of the ID Numbers of All Apples

# 7 Evaluation and promotion of models

## 7.1 Advantages of the models

(1)Adopting various image processing methods including color space conversion, bin arization processing, morphological processing and other image processing methods to process the image, it is remarkable to improve the model’s ability to extract the ap ple contour in the image, which makes the model more accurate in recognizing apple s.

(2) Using a deep learning framework to construct a fruit recognition model, it can realize high precision, fast and accurate recognition of various fruits in the image.

(3) By setting the color threshold and morphological discrimination methods, we can distinguish apples from leaves and ripe apples from green apples, which improves the recognition accuracy of ripe apples and enhances the picking efficiency of the robot.

## 7.2 Disadvantages of the models

(1) Models face some barriers when the background of apple images is relatively complex, which may include sky, branches and leaves, land, etc. And due to factors such as branch and leaf occlusion, fruit overlap, etc., it seriously interferes with the accurate identification of fruits, the accurate positioning of picking points, and the progress of picking activities.

(2) Considering the practical application needs of the algorithm, further research is needed on how to complete the recognition and localization of apple targets in a three-dimensional coordinate system.

## 7.3 Promotion of the models

(1) In order to simplify the model, it is assumed that green apples belong to unripe apples, but in fact some green apples are also ripe, but due to the model's low sensitivity to shape, it is not possible to make an accurate distinction between the green apples and the contours of the leaves.

(2) By setting appropriate parameters, the model can accurately recognize the location and number of ripe apples, ripeness, and quality in different situations, which can be very effectively applied to apple picking to ensure the quality of picking.

# 8 References

[1] Luo Jiamei; Wang Min. Research on fruit quality detection and classification method based on YOLOv5 [J]. Software Guide, 2023, 22(09): 190-195.

[2] Jiang, Y. (2021). Research on the recognition and positioning method of apple target in complex background based on OpenCV [Doctoral dissertation, Harbin University of Science and Technology].

[3] Chen, B.W., & Chen, G. (2022). Overview of fruit classification recognition and maturity detection technology. Computer Era, (07), 62-65.

[4] Weikuan J, Dean Zh, Chanli H. Fast Recognition of Overlapping Fruit Based on Maximum Optimisation for Apple Harvesting Robot[J]. International Journal of Collaborative Intelligence,

2015, 1(2): 124-136.

# Appendix

The Software of our using:

1.Pycharm

2.Python

3.Anaconda

4.Pytorch

5.OpenCV

The computer source code for question 1 and question 5 is **stored at the attachment** while it is too large to show in this article.

## Code for question 2:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
import zipfile
import os
import unidecode

import pip
from numpy.array_api import astype


def find_mature_apple_centers_and_size(image_path):

    image_path = (image_path)
    image = cv2.imread(image_path)
    hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

    lower_red = np.array([0, 50, 50])
    upper_red = np.array([10, 255, 255])
    mask = cv2.inRange(hsv_image, lower_red, upper_red)


    kernel = np.ones((5, 5), np.uint8)
    mask = cv2.dilate(mask, kernel, iterations=1)
    mask = cv2.erode(mask, kernel, iterations=2)
    contours, _ = cv2.findContours(mask, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
    size = []
    centers=[]
    image_height=image.shape[0]
```

```
    apple_count = 0
    for contour in contours:
        M=cv2.moments(contour)
        if M["m00"]!=0:
            cX=int(M["m10"]/M["m00"])
            cY = image_height-int(M["m01"] / M["m00"])
            centers.append((cX,cY))
            area = cv2.contourArea(contour)/20
            size.append(area)
    return centers,size




all_centers =[]
all_transformed_centers=[]
all_sizes=[]
paths=[]
root_dir='Attachment/Attachment 1'
start_number,end_number=1,200
for number in range(start_number,end_number+1):
    path=os.path.join(root_dir,f'{number}.jpg')
    paths.append(path)

for path in paths:
    centers,size= find_mature_apple_centers_and_size(path)
    all_centers.extend(centers)
    all_sizes.extend(size)

for path in paths:
    image_height=cv2.imread(path)
    transformed_center=[(x,image_height-y) for (x,y) in centers]
    all_transformed_centers.extend(transformed_center)

fig=plt.figure()
ax=fig.add_subplot(111)
x_coord,y_coord = zip(*all_centers)
si=all_sizes

plt.scatter(x_coord,y_coord,s=5)

plt.title('Apple location scatter plot')
plt.xlabel('X')
plt.ylabel('Y')
```

```
plt.show()
```

## Code for question 3:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
import zipfile
import os
import unidecode

import pip
from numpy.array_api import astype


def calculate_apple_ripeness_with_shape_exclusion(image_path):

    image_path = (image_path)
    image = cv2.imread(image_path)
    image_rgb=cv2.cvtColor(image, cv2.COLOR_BGR2RGB)


    lower_red = np.array([100, 0, 0])
    upper_red = np.array([255, 100, 100])
    lower_green = np.array([0, 100, 0])

    upper_green = np.array([100, 255, 100])
    mask_red = cv2.inRange(image_rgb, lower_red, upper_red)
    mask_green = cv2.inRange(image_rgb, lower_green, upper_green)

    contours, _ = cv2.findContours(mask_green, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    for contour in contours:
        area = cv2.contourArea(contour)
        x,y,w,h = cv2.boundingRect(contour)
        if w>0 and h>0:
            aspect_radio=w/h
            if aspect_radio>0.8 and aspect_radio<1.2 and area>100:
                cv2.drawContours(mask_green,[contour],-
1,(255,255,255),-1)
            else:
```

```
                cv2.drawContours(mask_green, [contour], -1, (0, 0, 0),
-1)

    red_count=np.sum(mask_red>0)
    green_count=np.sum(mask_green>0)

    ripness_score=red_count/(red_count+green_count+0.00000001)

    return ripness_score
paths=[]
ripeness_with_shape_exclusion=[]
root_dir='Attachment/Attachment 1'
start_number,end_number=1,200
for number in range(start_number,end_number+1):
    path=os.path.join(root_dir,f'{number}.jpg')
    paths.append(path)

for path in paths:
    score=calculate_apple_ripeness_with_shape_exclusion(path)
    ripeness_with_shape_exclusion.append(score)


plt.hist(ripeness_with_shape_exclusion, bins=20, color='blue',
alpha=0.7)
plt.title('Apple maturity distribution')
plt.xlabel('maturity')
plt.ylabel('number')
plt.show()
```

## Code for question 4:

```
import math

import cv2
import numpy as np
import matplotlib.pyplot as plt
import zipfile
import os
import unidecode

import pip
from numpy.array_api import astype
```

```
def find_mature_apple_quality(image_path):

    image_path = (image_path)
    image = cv2.imread(image_path)
    hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

    lower_red = np.array([0, 50, 50])
    upper_red = np.array([10, 255, 255])
    mask = cv2.inRange(hsv_image, lower_red, upper_red)


    kernel = np.ones((3, 3), np.uint8)
    mask = cv2.dilate(mask, kernel, iterations=1)
    mask = cv2.erode(mask, kernel, iterations=2)
    contours, _ = cv2.findContours(mask, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)

    size = []
    centers=[]
    radiuss=[]
    qualitys=[]
    image_height=image.shape[0]
    apple_count = 0
    for contour in contours:
        M=cv2.moments(contour)
        if M["m00"]!=0:
            cX=int(M["m10"]/M["m00"])
            cY = image_height-int(M["m01"] / M["m00"])
            centers.append((cX,cY))
            area = cv2.contourArea(contour)/100
            radius=math.sqrt(area/3.1415926)
            size.append(area)
            radiuss.append(radius)
            quality=0.9*1.33333*radius*radius*radius*3.1415926
            qualitys.append(quality)
    return qualitys




all_centers =[]
all_transformed_centers=[]
all_sizes=[]
```

```
paths=[]
all_qualitys=[]
root_dir='Attachment/Attachment 1'
start_number,end_number=1,200
for number in range(start_number,end_number+1):
    path=os.path.join(root_dir,f'{number}.jpg')
    paths.append(path)

for path in paths:
    qualitys= find_mature_apple_quality(path)
    all_qualitys.append(qualitys)


plt.hist(all_qualitys, bins=20, alpha=0.7)
plt.xlim(0,120)
plt.title('Apple mass distribution')
plt.xlabel('mass')
plt.ylabel('number')
plt.show()
```