



Embedded System

DESIGN AND DEVELOPMENT OF A SORTING SYSTEM FOR RIPENESS AND DAMAGE IN AVOCADO (*Persea americana* Mill.) BASED ON *MACHINE LEARNING*

Aldi Nuari ¹, Budi Rahmadya ^{*2}, Fadli Hafizulhaq ³

^{1,2} Department of Computer Engineering, Faculty of Information Technology, Andalas University, Padang, West Sumatra, 25163, Indonesia

³ Department of Agricultural dan Biosystem Engineering, Faculty of Agricultural Technology, Andalas University, Padang, West Sumatra, 25163, Indonesia

ARTICLE INFORMATION

Received: September 10, 2025
Revised: October 15, 2025
Available online: October 31, 2025

KEYWORDS

Avocado, YOLOv8n, Raspberry Pi 4, Google Coral, Real Time, Object Detection.

CORRESPONDENCE

E-mail: budi-r@it.unand.ac.id

ABSTRACT

Avocado is an important commodity that requires proper postharvest handling, particularly in sorting based on ripeness levels and fruit condition. Conventional methods that still rely on manual inspection have limitations in terms of accuracy and efficiency. This study proposes a real-time sorting system based on YOLOv8n running on a Raspberry Pi 4 Model B with a Google Coral accelerator to detect avocados and classify their ripeness levels as well as damage. The model was trained with a custom dataset and achieved a precision of 77.2%, recall of 84.4%, and a macro-average F1-Score of 80.6%. The system was integrated with a camera for image acquisition and a mini conveyor for the sorting process. Experimental results demonstrated reliable detection with an inference speed of 11.7 ms per image, and field testing successfully classified avocados according to their categories. This system has proven to be effective, cost-efficient, and supports the improvement of avocado postharvest quality.

INTRODUCTION

Indonesia is a country with abundant natural resources, including marine, mineral, and agricultural wealth. Agriculture is a human activity that encompasses farming, livestock, fisheries, and forestry. The majority of the Indonesian population earns a living as farmers, making agriculture the largest sector in the development of Indonesia's economy. [1]

Avocado plants have long been known to the Indonesian people. This fruit is one of the horticultural commodities with relatively high economic value. Avocado cultivation has spread widely across Indonesia. There are three major groups of avocado varieties, namely Mexican, West Indian, and Guatemalan. [2]

According to a report by the Central Bureau of Statistics (BPS), in 2022 the production of avocados in Indonesia reached 865,780 tons. This amount increased by 27.7% compared to the previous year, which was 669,260 tons. The production of avocados is not only marketed domestically but also exported abroad. The Director General of Horticulture, Ministry of Agriculture of the Republic of Indonesia, Prihasto Setyanto, stated that avocado

exports reach 400 tons per year. Based on this export volume, Indonesia ranks fifth in the world as one of the largest avocado-exporting countries. [3]

The origin of avocado is Central America, specifically Mexico, Guatemala, and Honduras. This fruit was cultivated from the Rio Grande to Central Peru before the arrival of Europeans. It is now cultivated in other tropical and subtropical regions around the world (Lim, 2012). Early Europeans discovered the avocado, cultivated it, and distributed it throughout Central America and northern South America during the 16th century. Evidence of this can be found in the original names given to the avocado in various languages and archaeological findings. Carbon dating indicates that Mexican avocados have been used as food since 9,000–10,000 years ago. Early records show that avocados were not cultivated in the Caribbean islands during the pre-Columbian period and were introduced to Jamaica by the Spanish around 1650. The spread to tropical regions of Africa and Asia occurred during the 1700s and 1800s. The first recorded import into Florida was in 1833, into California in 1848, and into Hawaii in the early 19th century. By 1855, avocado trees were commonly found in gardens in Oahu and distributed to other islands in the Hawaiian archipelago. Today, they are widely grown throughout tropical and subtropical regions, although the uses of the fruit vary across

different areas (Janick and Paull, 2008). Avocado plants are believed to have been introduced to Indonesia in the 18th century and are now widely spread across almost all regions of the country (Syah, 2018). [4]

METHOD

Detail cara kerja sistem, dijelaskan pada gambar berikut.



Figure 1 General Design of the Tool Environment

The general design of this system consists of two main components, namely a scanner box and a mini belt conveyor, which are designed to sort avocados (*Persea americana* Mill.) based on their level of ripeness using the Convolutional Neural Network (CNN) method.

1. Scanner Box

The scanner box functions as the main component for analyzing the ripeness level of avocados in real-time. This component is equipped with:

- Raspberry Pi Camera V2, used to capture images of avocados entering the scanner box.
- Raspberry Pi 4B, serving as the data processing unit that runs the Convolutional Neural Network (CNN) algorithm to analyze the ripeness level of the fruit based on its color and texture.
- Servo motor, which controls the scanner box lid to ensure the fruit remains stable during the analysis process. Once the process is complete, the servo opens the lid, allowing the avocado (*Persea americana* Mill.) to continue its path on the conveyor.

2. Mini Belt Conveyor

The mini belt conveyor functions to transport avocados into the scanner box and deliver the analyzed fruit to the sorting container based on its ripeness level. The conveyor consists of:

- DC motor, which drives the belt conveyor continuously.
- L298N motor driver, used to control the motor speed.
- Additional sensor (optional), which can detect the position of the fruit before entering the scanner box.

After the ripeness level is identified, the fruit is directed to one of the sorting containers according to the specified category. The machine learning (CNN)-based analysis process ensures high accuracy in avocado sorting. Communication between the scanner box and conveyor is automatically integrated to ensure process efficiency.

With the scanner box design equipped with a servo-controlled lid, image stability and analysis results are maintained. In addition,

the conveyor is designed to transport avocados one by one, ensuring optimal analysis. This system is designed to help farmers sort avocados efficiently, reduce losses caused by damage or spoilage, and increase the market value of their products.

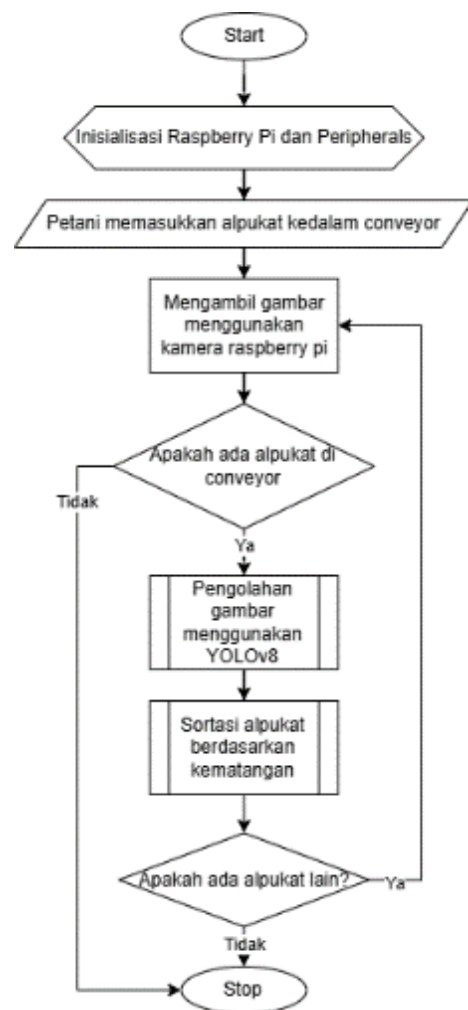


Figure 2 General System Flowchart

The flowchart in Figure illustrates the overall working process of the system. The process begins when the farmer first powers on the Raspberry Pi 4B, the mini belt conveyor, and other peripherals. Once the Raspberry Pi and the mini belt conveyor are activated, the next step is data acquisition from the avocado placed on the mini belt conveyor. The conveyor then transports the fruit to the scanner box. Inside the scanner box, the camera captures an image of the avocado, and the Raspberry Pi 4B analyzes the image using the Convolutional Neural Network (CNN) method to determine the ripeness level (unripe, ripe, or overripe). Based on the analysis results, the system instructs the conveyor to direct the fruit to the appropriate sorting container according to its classification, namely unripe, ripe, or rotten. The

process is repeated for each fruit until all avocados are completely sorted.

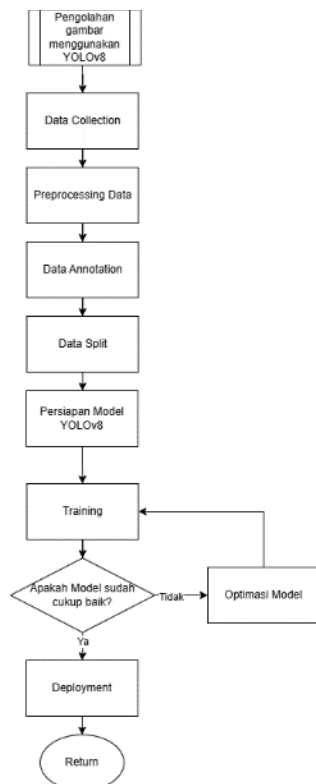


Figure 3 The distance setting process is dangerous

In Figure 3, the flowchart illustrates the sub-process of the YOLOv8 computational algorithm. Data Collection is performed to gather image data. Data Preprocessing represents the steps that include image resizing, data augmentation, and normalization. Data Annotation refers to the labeling process of avocado images using bounding boxes. Data Splitting involves dividing the dataset into training, validation, and test sets.

YOLOv8 Model Preparation consists of configuring the model, including loading pretrained weights and adjusting hyperparameters. Model Training utilizes the training data to optimize the model. Model Evaluation is carried out using the validation data to assess the model's performance.

At the Decision-Making stage, the process determines whether the model is sufficiently accurate or needs retraining. If the model performance is adequate, the process continues to Deployment; if not, the workflow returns to the training phase. Model Optimization may also be performed when evaluation results indicate insufficient performance. Finally, Deployment involves implementing the trained model into the application to perform avocado (*Persea americana* Mill.) detection, after which the process is completed.

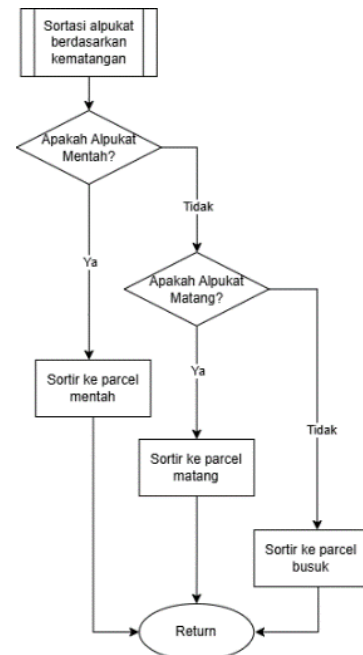


Figure 4 The process of detecting children going to dangerous place

In Figure 4, the flowchart illustrates the sub-process of sorting based on ripeness. The system performs classification of avocado ripeness levels into three categories: unripe, ripe, and rotten. The process then continues with sorting the fruit into the designated parcel according to the classification result.

RESULTS AND DISCUSSION

Hardware Implementation

The following are details of the hardware implementation found in each system that will later be connected to each other.

1. AvoSmart (Avocado Smart Sorting System)



Figure 5 System Appearance

In the implementation of the ripeness and damage sorting system, a main module is used, which consists of three integrated components: the detection unit for image acquisition (camera), the fruit driving mechanism (mini belt conveyor), and the sorting unit (servo motor arm). These components work in a continuous and interconnected manner.



Figure 12 Anotation Dataset

The model consists of four object classes: raw, cooked, rotten, and moldy. Labeling was performed one by one, with a total of 978 labels. The training data split was divided into 70% training data, 20% validation data, and 10% testing data.

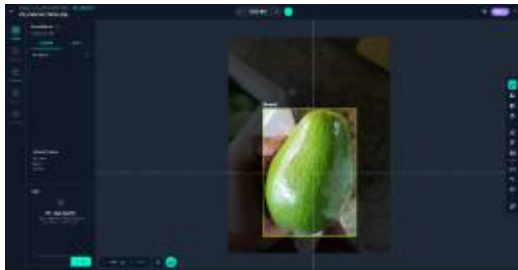


Figure 13 Anotation Raw

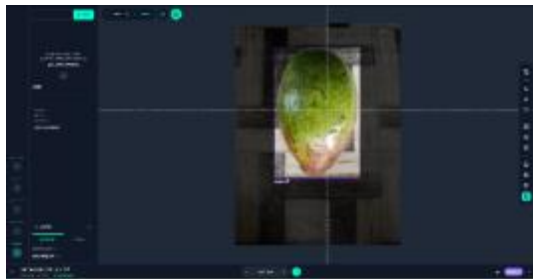


Figure 14 Anotation Cooked

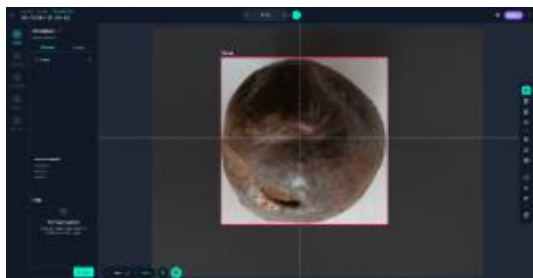


Figure 15 Anotaton Rotten

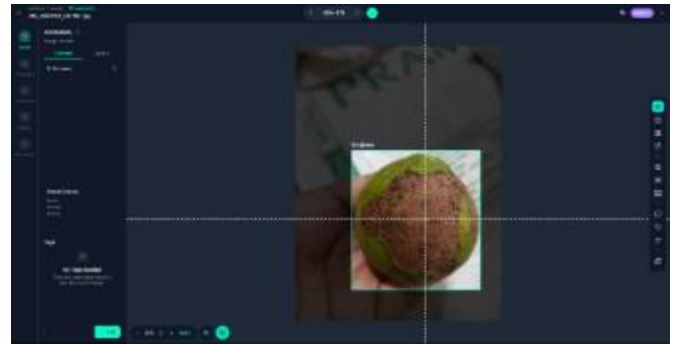


Figure 16 Anotation Moldy

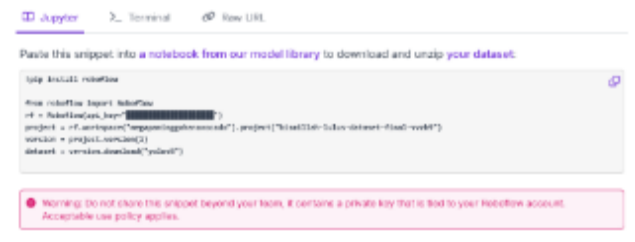


Figure 17 Roboflow Dataset API Key

After the Dataset is completed in Roboflow, the next stage is to create a Machine Learning model using Google Colab with the Dataset API provided by Roboflow.

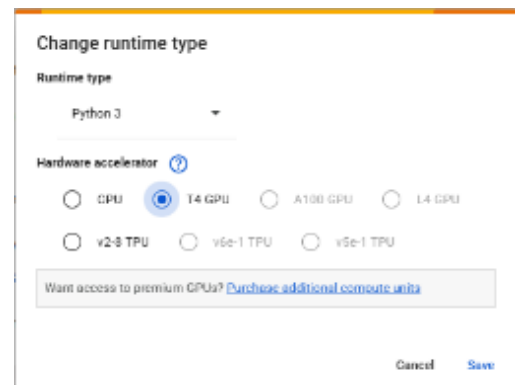


Figure 18 T4 GPU [8]

Before building the model, we change the runtime type to T4 GPU for training, as shown in Figure 18. NVIDIA® T4 GPUs accelerate a wide range of cloud workloads, including high-performance computing, deep learning training and inference, machine learning, data analytics, and graphics. Based on the new NVIDIA Turing™ architecture and packaged in a small, energy-efficient 70-watt PCIe form factor, T4 is optimized for mainstream computing environments and features multi-precision Turing Tensor Cores and new RT Cores.[8]



Figure 19 Instalation API Dataset YOLOv8 on Google Colab

Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
100/100	2.97G	0.4327	0.2897	1.828	2	Size: 1000 117/117 [100:21000000, 0.0001/s]
Class	Images	Instances	Instances	Instances	Instances	Size: 1000 117/117 [100:21000000, 0.0001/s]
all	99	99	99	99	99	Size: 1000 117/117 [100:21000000, 0.0001/s]
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
200/100	3.00G	0.4327	0.2897	1.828	2	Size: 1000 117/117 [100:21000000, 0.0001/s]
Class	Images	Instances	Instances	Instances	Instances	Size: 1000 117/117 [100:21000000, 0.0001/s]
all	99	99	99	99	99	Size: 1000 117/117 [100:21000000, 0.0001/s]
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
300/100	3.00G	0.4327	0.2897	1.828	2	Size: 1000 117/117 [100:21000000, 0.0001/s]
Class	Images	Instances	Instances	Instances	Instances	Size: 1000 117/117 [100:21000000, 0.0001/s]
all	99	99	99	99	99	Size: 1000 117/117 [100:21000000, 0.0001/s]
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
400/100	3.00G	0.4327	0.2897	1.828	2	Size: 1000 117/117 [100:21000000, 0.0001/s]
Class	Images	Instances	Instances	Instances	Instances	Size: 1000 117/117 [100:21000000, 0.0001/s]
all	99	99	99	99	99	Size: 1000 117/117 [100:21000000, 0.0001/s]

Figure 20 Training Proses

The training process of 100 epochs took approximately 0.627 hours or the equivalent of ≈ 38 minutes using Google Colab.

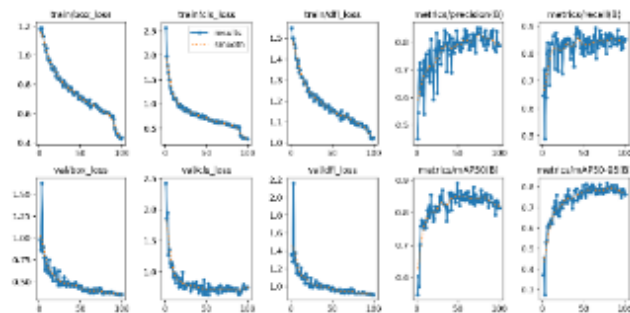


Figure 21 Grafik Hasil Pelatihan Model

The graph in Figure 4.18 shows the results of the object detection model training and validation process over 100 epochs, displaying the loss progression and evaluation metrics. The top left shows the train/box_loss consistently decreasing from around 1.2 to nearly 0.4, indicating that the model is improving at predicting bounding box positions. A similar trend is seen in the train/cls_loss, which initially hovered above 2.5 and then steadily decreased to near 0.5, indicating significant improvement in object classification. Meanwhile, the train/dfl_loss also shows a decreasing pattern from around 1.5 to 1.0, indicating an improvement in the distribution of predicted bounding box locations.

In the validation phase, the decline pattern is similar. Val/box_loss, val/cls_loss, and val/dfl_loss decrease sharply at the beginning of the epoch and then stabilize near a low value,

indicating that the model not only learns well on the training data but is also able to generalize to the validation data.

Evaluation metrics showed a positive trend. Precision increased rapidly and stabilized around 0.8–0.9 after about 20 epochs, demonstrating the model's ability to minimize false positive predictions. Recall also increased and stabilized around 0.8, indicating the model successfully detected most of the correct objects. The mAP50 value, which measures detection accuracy at an IoU threshold of 0.5, increased from around 0.6 to over 0.9, reflecting excellent detection performance. Meanwhile, mAP50-95, which uses a wider IoU range, also showed a steady increase to nearly 0.8, confirming the model's consistent performance across different accuracy levels.

Overall, this graph shows that the training process is running well, marked by a decrease in the loss value and an increase in the evaluation metrics, and there are no signs of significant overfitting because the validation results are in line with the training results.



Figure 22 Model yang diperoleh setelah training

After the training process is complete, a best.pt file will appear, as shown in Figure 4.19. Saved automatically when performance is optimal, the .pt format is PyTorch. This file contains the YOLOv8n model architecture and configuration information such as the number of classes, input size, and so on.



Figure 23 Konversi model

Google Coral must use the TFLite model because Edge TPU only supports the TensorFlow Lite (.tflite) format which has been compiled specifically for TPU accelerators. Therefore, we must convert the model from best.pt to .tflite format but because of the framework differences from PyTorch to TensorFlow it must go through ONNX first as seen in Figure 4.20, ONNX (Open Neural Network Exchange) is a neutral format across frameworks (PyTorch, TensorFlow, etc.)

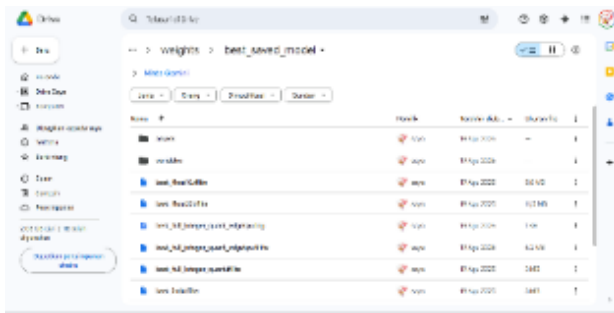


Figure 24 Hasil Konversi Model

System Testing

System testing was conducted to determine the extent of the capability or performance of the implementation of the avocado (Persea americana Mill.) ripeness and damage sorting tool. This testing is expected to provide an evaluation of the system's effectiveness in carrying out its duties.

CPU and memory usage testing on the RaspberryPi 4B

Tests were conducted to determine CPU and memory usage on the RaspberryPi 4B when running the previously created machine learning model.

The testing steps are to run the machine learning model using the python program and monitor CPU usage by running 'htop' on the terminal, and observing memory usage in Task Manager.

Hasil Pengujian 1

Table1 Hasil pengujian CPU dan Memory pada RaspberryPi+Coral

No	Parameter	Hasil	Bukti
1	Penggunaan CPU	Core 0: 50% Core 1: 38,5% Core 2: 45,1% Core 3: 47,3%	
2	Memory	1.89G/7.63G	
3	Suhu saat idle	Temp=39.9°C	
4	Suhu saat model dijalankan	Temp=67.2°C	

Figure 25 Pengujian penggunaan CPU di 'htop'

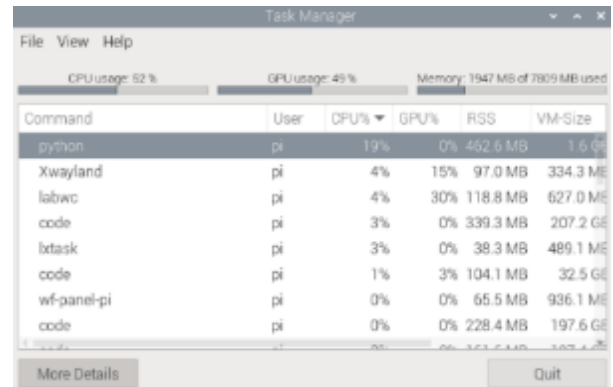


Figure 26 Pengujian penggunaan memory di Task Manager

Test Analysis 1

In Figure 25 htop says: Load average: 2.08 2.47 2.12. This shows the average number of processes currently active/waiting on the CPU in the last 1 minute, 5 minutes, and 15 minutes. The uptime of 02:09:52 is the length of time the Raspberry Pi has been on since the last boot. So it has been running for 2 hours 9 minutes 52 seconds when the screenshot was taken. For CPU usage, there are 4 cores on the Raspberry Pi 4B.

Core 0: 50% used

Core 1: 38.5% used

Core 2: 45.1% used

Core 3: 47.3% used

The average CPU operates at around 45%–50% per core. The Python process or machine learning model being run uses 83% of a core, making it the primary load.

2. Testing camera resolution based on distance and light intensity in detecting avocado (Persea americana Mill.)

This test was conducted to determine the effect of camera resolution on system performance in detecting the level of ripeness and damage of avocados.

Testing Step 2

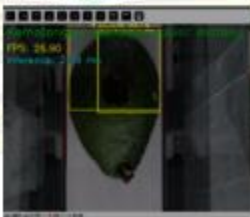
The testing steps are done by setting the resolution in the program, the resolutions tested are 160x120, 320x240, 640x480, and 1280x720. The parameters observed include FPS (Frames per Second), Inference (ms), and confidence score (%).

Test Results 2

Table 2 Tabel pengujian pengaruh resolusi kamera

Resolusi	FPS	Inference (ms)
160x120	41.86	Undefined
320x240	34.23	20,5 ms
640x480	34.86	21,6 ms
1280x720	11.37	18,9 ms

Table 3 Pengaruh Intensitas Cahaya dan Jarak

No	Nilai Lux	Bukti	Keterangan
1	50		Objek salah deteksi (Jarak < 35 cm)
2	475		Objek terdeteksi dengan benar
3	850		Objek terdeteksi dengan benar

Test Analysis 2

Analysis in Table 4.1 At 160x120 resolution, the highest FPS achieved was 41.86 FPS with a confidence score of 91.6%, but the inference time value was undefined. 320x240 resolution produced an FPS of 34.23 FPS, an inference time of 20.5 ms, and a confidence score of 94.4%. 640x480 resolution had an FPS of 34.86 FPS, an inference time of 21.6 ms, and the highest confidence score of 95.4%. 1280x720 resolution produced the lowest FPS of 11.37 FPS, the fastest inference of 18.9 ms, but the confidence score actually dropped to 84.9%. As for the test results from the light intensity, which used 50 Lux, 475 Lux, and 850 Lux. In conclusion, the system cannot be used at low light intensity, but can run normally in both normal light and outdoor light.

3. Belt Conveyor Testing on Fruit Size and Weight

This test aims to determine the effect of the number and weight of fruit on the time required by the Minibelt Conveyor to move avocados.

Testing Step 3

The test was carried out by counting the number of fruits, measuring the total weight of the fruits using a digital scale, measuring the time using a stopwatch, and setting the test speed on the minibelt conveyor to 140 rpm.

Test Results 3

Table 4 Testing the effect of size and weight on Minibelt Conveyor

No	Jumlah	Berat (gram)	Waktu Selesai (detik)
1	1 Buah	238g	07.52
2	1 Buah	288g	07.49
3	3 Buah	914g	07.60
4	6 Buah	1542g	07.80

Test Analysis 3

Based on the test results, as can be seen in Table 4.2, the time required by the minibelt conveyor is relatively stable at around 7.49–7.80 seconds, even though the number and weight of the fruit vary. This indicates that the weight of the fruit does not significantly affect the performance of the Minibelt Conveyor. However, during testing, the fruit must not exceed the tolerance limit of 8–12 cm for the belt size because it can cause the fruit to fall or get caught on the track. Pengujian Performa Model YOLOv8n

Langkah Pengujian 4

Testing was performed on the confusion matrix in Figure 4.38 using calculation formulas to determine accuracy, precision, recall, and f1-score. The calculation formulas are as follows:[9]

$$Accuracy (Class i) = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Precision (Class i) = \frac{TP}{TP + FP}$$

$$Recall (Class i) = \frac{TP}{TP + FN}$$

$$F1 - Score (Class i) = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

Test Results 4

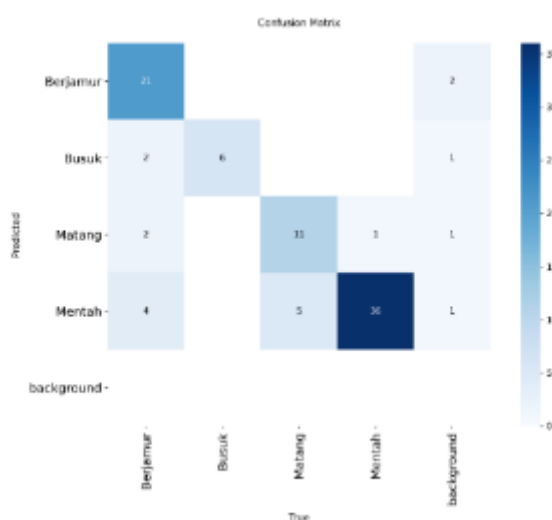


Figure 27 Confusion Matrix

Test Analysis 4

1. Kelas Berjamur

Table 5 Kelas Berjamur

Berjamur	21 (TP)	0 (FP)	0 (FP)	0 (FP)	2 (FP)
Busuk	2 (FN)	6 (TN)	0 (TN)	0 (TN)	1 (TN)
Matang	2 (FN)	0 (TN)	11 (TN)	1 (TN)	1 (TN)
Mentah	4 (FN)	0 (TN)	5 (TN)	36 (TN)	1 (TN)
Background	0 (FN)	0 (TN)	0 (TN)	0 (TN)	0 (TN)
Predicted/True	Berjamur	Busuk	Matang	Mentah	Background

$$TP = 21$$

$$TN = 6 + 1 + 11 + 1 + 1 + 5 + 36 + 1 = 62$$

$$FP = 2$$

$$FN = 2 + 2 + 4 = 8$$

$$Accuracy (Berjamur) = \frac{21 + 62}{21 + 2 + 62 + 8} = 0,892 \approx 89,2\%$$

$$Precision (Berjamur) = \frac{21}{21 + 2} = 0,913 \approx 91,3\%$$

$$Recall (Berjamur) = \frac{21}{21 + 8} = 0,724 \approx 72,4\%$$

$$F1 - Score (Berjamur) = \frac{2 \times 0,724 \times 0,913}{0,724 + 0,913} = 0,807 \approx 80,7\%$$

2. Kelas Busuk

Table 6 Kelas Busuk

Berjamur	21 (TN)	0 (FN)	0 (TN)	0 (TN)	2 (TN)
Busuk	2 (FP)	6 (TP)	0 (FP)	0 (FP)	1 (FP)
Matang	2 (TN)	0 (FN)	11 (TN)	1 (TN)	1 (TN)
Mentah	4 (TN)	0 (FN)	5 (TN)	36 (TN)	1 (TN)
Background	0 (TN)	0 (FN)	0 (TN)	0 (TN)	0 (TN)
Predicted/True	Berjamur	Busuk	Matang	Mentah	Background

$$TP = 6$$

$$TN = 21 + 2 + 2 + 11 + 1 + 1 + 4 + 5 + 36 + 1 = 84$$

$$FP = 2 + 1 = 3$$

$$FN = 0$$

$$Accuracy (Busuk) = \frac{6 + 84}{6 + 3 + 84 + 0} = 0,967 \approx 96,7\%$$

$$Precision (Busuk) = \frac{6}{6 + 3} = 0,666 \approx 66,6\%$$

$$Recall (Busuk) = \frac{6}{6 + 0} = 1 \approx 100\%$$

$$F1 - Score (Busuk) = \frac{2 \times 1 \times 0,666}{1 + 0,666} = 0,799 \approx 80\%$$

3. Kelas Matang

Table 7 Kelas Matang

Berjamur	21 (TN)	0 (TN)	0 (FN)	0 (TN)	2 (TN)
Busuk	2 (TN)	6 (TN)	0 (FN)	0 (TN)	1 (TN)
Matang	2 (FP)	0 (FP)	11 (TP)	1 (FP)	1 (FP)
Mentah	4 (TN)	0 (TN)	5 (FN)	36 (TN)	1 (TN)
Background	0 (TN)	0 (TN)	0 (FN)	0 (TN)	0 (TN)
Predicted/True	Berjamur	Busuk	Matang	Mentah	Background

$$TP = 11$$

$$TN = 21 + 2 + 2 + 6 + 1 + 4 + 36 + 1 = 73$$

$$FP = 2 + 1 + 1 = 4$$

$$FN = 5$$

$$Accuracy (Matang) = \frac{11 + 73}{11 + 4 + 73 + 5} = 0,903 \approx 90,3\%$$

$$Precision (Matang) = \frac{11}{11 + 4} = 0,73 \approx 73\%$$

$$Recall (Matang) = \frac{11}{11 + 5} = 0,68 \approx 68\%$$

$$F1 - Score (Matang) = \frac{2 \times 0,68 \times 0,73}{0,68 + 0,73} = 0,703 \approx 70,3\%$$

4. Kelas Mentah

Table 8 Kelas Mentah

Berjamur	21 (TN)	0 (TN)	0 (TN)	0 (FN)	2 (TN)
Busuk	2 (TN)	6 (TN)	0 (TN)	0 (FN)	1 (TN)
Matang	2 (TN)	0 (TN)	11 (TN)	1 (FN)	1 (TN)
Mentah	4 (FP)	0 (FP)	5 (FP)	36 (TP)	1 (FP)
Background	0 (TN)	0 (TN)	0 (TN)	0 (FN)	0 (TN)
Predicted/True	Berjamur	Busuk	Matang	Mentah	Background

$$TP = 36$$

$$TN = 21 + 2 + 2 + 6 + 1 + 2 + 11 + 1 = 46$$

$$FP = 4 + 5 + 1 = 10$$

$$FN = 1$$

$$Accuracy (Mentah) = \frac{36 + 46}{36 + 10 + 46 + 1} = 0,88 \approx 88\%$$

$$Precision (Mentah) = \frac{36}{36 + 10} = 0,782 \approx 78,2\%$$

$$\text{Recall (Mentah)} = \frac{36}{36 + 1} = 0,972 \approx 97,2\%$$

$$\begin{aligned} F1 - \text{Score (Mentah)} &= \frac{2 \times 0,972 \times 0,782}{0,972 + 0,782} = 0,866 \\ &\approx 86,6\% \end{aligned}$$

5. Macro-averaging

Berikut rumus untuk merata-ratakan presisi dan recall di semua kelas:

Precision (Macro – Average)

$$= \frac{\text{Precision}(\text{Class A}) + \text{Precision}(\text{Class B}) + \dots \text{Precision}(\text{Class N})}{N}$$

Recall (Macro – Average)

$$= \frac{\text{Recall}(\text{Class A}) + \text{Recall}(\text{Class B}) + \dots \text{Recall}(\text{Class N})}{N}$$

F1 – Score (Macro – Average)

$$= \frac{2 \times \text{Recall (Macro – Average)} \times \text{Precision (Macro – Average)}}{\text{Recall (Macro – Average)} + \text{Precision (Macro – Average)}}$$

Karena jumlah kelas ada 4 yaitu Berjamur, Busuk, Matang, dan Mentah, maka nilai N = 4

Precision (Macro – Average)

$$= \frac{0,913 + 0,666 + 0,73 + 0,782}{4}$$

$$= 0,772 \times 100\% = 77,2\%$$

$$\text{Recall (Macro – Average)} = \frac{0,724 + 1 + 0,68 + 0,972}{4}$$

$$= 0,844 \approx 84,4\%$$

$$F1 - \text{Score (Macro – Average)} = \frac{2 \times 0,844 \times 0,772}{0,844 + 0,772}$$

$$= 0,806 \approx 80,6\%$$

CONCLUSIONS

Based on the results of implementation, testing, and analysis carried out in this final project, the following conclusions can be drawn:

1. The device has been able to function according to its main purpose, namely detecting avocados based on ripeness and damage, and performing automatic sorting.
2. The fruit sorting process can only be carried out one by one.
3. The Raspberry Pi is not capable of performing machine learning processing in real time, thus requiring additional hardware such as an Edge TPU.

REFERENCES

- [1] E. K. a. S. S. Ali Maddinsyah, “PENYULUHAN MANAJEMEN PEMANFAATAN SUMBER DAYA ALAM UNTUK MENINGKATKAN PEREKONOMIAN KELUARGA KAMPUNG

CIBOLEGER LEBAK - BANTEN,” Jurnal Pengabdian Dharma Laksana, vol. 1, no. 1, 2018.

- [2] D. S. a. T. B. Lukitariati Sadwiyanti, Budidaya alpukat, Sumatera Barat: Balai Penelitian Tanaman Buah Tropika, 2009.
- [3] Indra Lintang [Online] Available: <https://www.inilah.com/daerah-penghasil-alpukat-terbanyak-di-indonesia>
- [4] Alpokat (Persea Americana Mill.), Taman Digital Fakultas Pertanian Universitas Andalas, Kampus Unand, Kota Padang. [Online]. Available: <https://tamandigital.faperta.unand.ac.id/index.php/deskripsi/tanaman/alpokat-persea-americana-mill>
- [5] A. H. Makarim, "Menggunakan *Convolutional Neural Network* (CNN) untuk Klasifikasi Gambar Produk," Medium, May 15, 2024. [Online]. Available: <https://medium.com/@21611129/menggunakan-convolutional-neural-network-cnn-untuk-klasifikasi-gambar-produk-c9f75c2446cb>. [Accessed: Nov. 18, 2024].
- [6] Xukyo, “Using a PCA9685 module with Raspberry Pi,” AranaCorp, 27 Sep. 2020. [Online]. Available: <https://bit.ly/3Qm5P6Y>. [Accessed: May. 13, 2025].
- [7] Google LLC, “USB Accelerator datasheet,” v.1.4, Aug. 2020. [Online]. Available: <https://coral.ai/docs/accelerator/datasheet/>. [Accessed: May. 13, 2025].
- [8] NVIDIA, "NVIDIA T4 Tensor Core GPU," NVIDIA. [Online]. Available: <https://www.nvidia.com/en-us/data-center/tesla-t4/>. [Accessed: Aug. 4, 2025].
- [9] D. H. Seol, J. E. Choi, C. Y. Kim, and S. J. Hong, “Confusion matrix: Precision, Recall, Accuracy, and F1 score,” Jan. 2023. [Online]. Available: https://www.researchgate.net/figure/Confusion-matrix-Precision-Recall-Accuracy-and-F1-score_fig4_367393140. [Accessed: Aug. 4, 2025].

AUTHOR(S) BIOGRAPHY

Aldi Nuari



Born in Solok City, West Sumatra, on January 14 2003, then studied at the Department of Computer Engineering, Faculty of Information Technology, Andalas University.

Dr. Eng. Ir. Budi Rahmadya, M.Eng



Born in Bukittinggi, West Sumatra, on December 22 1981, Associate Professor at the Department of Computer Engineering, Faculty of Information Technology, Andalas University

Dr. Fadli Hafizulhaq, S.T



Fadli Hafizulhaq is a Senior Lecturer in the Department of Agricultural dan Biosystem Engineering, Faculty of Agricultural Technology, Universitas Andalas, Indonesia. He finished his doctoral study with a research focus on biocomposite from agricultural products and waste in 2019 when he was still 27 y.o. Fadli is one of hundreds of PMDSU Awardees (Program Magister Menuju Doktorat untuk Sarjana Unggul) by the Ministry of Research, Technology, and Higher Education (Kemenristekdikti). In the past 7 years, Fadli has published many research articles in various journals. Most of his published articles discuss the topic of bioplastics and biocomposites. Currently, he is also interested in researching control systems and data science in post-harvest engineering.