Penerbit Akademia Baru

Journal of Advanced Research Design

Journal homepage:
https://akademiabaru.com/submit/index.php/ard
ISSN: 2289-7984

# Real-Time Object Detection using Convolution Neural Network on Raspberry Pi Embedded Systems

Ahmad Luqman Abd Wahab[1], Noor Huda Ja'afar[1,*], Ernie Mazuin Mohd Yusof[1], Sahar Wahab Khadim[2]

[1] Instrumentation and Control Engineering Section, Universiti Kuala Lumpur Malaysian Institute of Industrial Technology, Bandar Seri Alam, Masai, 81750 Johor Bahru, Johor, Malaysia
[2] Ministry of Education, Karkh Second Directorate of Education, Iraq

**ARTICLE INFO**

***Article history:***
Received 27 January 2025
Received in revised form 21 February 2025
Accepted 2 May 2025
Available online 23 May 2025

**ABSTRACT**

Object detection through colour filtering of pixel hue and saturation values is effective for objects lacking consistent distinguishable features. Traditional object recognition frameworks, however, are often passive and neglect the relationship between detection configuration and recognition performance, as well as the importance of feedback for image quality improvement. This limitation, combined with the inherent challenges of human colour detection, particularly for those with colour vision deficiencies, can impact tasks in sorting areas. To address these challenges, this project focuses on developing a real-time object detection system using convolutional neural networks (CNNs). The system integrates a camera detector, Raspberry Pi, servo motor, 5 V DC motor, infrared obstacle avoidance sensor and Liquid-Crystal Display (LCD) display. The dataset comprises 660 augmented chili images in red, green and yellow, used for both training and testing. During the final training epoch, the system achieved 100 % accuracy for both the training and validation datasets, demonstrating the efficacy of CNNs in real-time object detection and classification. To enhance the dataset, four types of image augmentations were applied: rotation along the centre, brightness increments and decrements and image sharpening. These augmentations increased the dataset size and improved the robustness of the neural network. The validation accuracy was only 3 % lower than the training accuracy, indicating minimal overfitting. This project highlights the advantages of using CNNs for precise colour-based object identification, addressing the inefficiencies of traditional methods and the limitations of human visual perception. By integrating advanced image processing techniques and machine learning algorithms, the system provides a robust and reliable solution for real-time object detection, applicable in various industrial sectors. This approach not only improves the accuracy and efficiency of object detection but also offers a practical solution for individuals with colour vision deficiencies, making it a valuable tool for diverse applications.

***Keywords:***
Real-time detection; convolution neural networks; embedded system

[*] *Corresponding author*
*E-mail address: noorhuda.jaafar@unikl.edu.my*

https://doi.org/10.37934/ard.132.1.7890

## 1. Introduction

With the advent of the Industrial Revolution 4.0 (IR 4.0), advanced information and communication technologies (ICT) such as the Internet of Things (IoT), mechanical robots and artificial intelligence (AI) have become integral parts of our lives. These technologies have rapidly evolved over time and are expected to shape a future where virtual environments seamlessly integrate with human reality [1]. In various industries, including automotive and medicine, AI has been introduced and continues to advance, enhancing efficiency and innovation. Additionally, AI has become a ubiquitous presence in everyday life, with billions of people worldwide using AI voice assistants on their smartphones.

Creating intelligent machines that can perform tasks that typically demand human energy and intelligence is the goal of the broad discipline of AI in computer science. Furthermore, prior studies have noted that AI encompasses learning, language, comprehension, object or sound recognition and problem-solving capabilities [2]. AI systems typically complete tasks swiftly and with few errors, particularly in repetitive, meticulous tasks such as analysing large volumes of legal documents to be accurately categorized, as demonstrated in earlier findings [3]. The availability of AI also benefits humans in their daily activities. Examples include applications that can anticipate user behaviour, maps and GPS that show users the routes to their destinations and Siri, which functions as a human personal assistant. In addition, the use of AI in the field of medical engineering aids in the management of many medical issues, including depression. For instance, an AI robotic companion was developed to assist individuals suffering from depression and allergies to animal fur.

Machine learning (ML), which is a subset of AI, plays a crucial role in the rapidly expanding field of data science, as stated in prior research [4]. Learning ML algorithms is essential for data analysis. Instead of manually programming codes with specific instructions, algorithms are trained to produce classifications or predictions using statistical approaches, which help uncover valuable insights in data mining initiatives. On the other hand, deep learning (DL), which is a further subset of ML, is based on neural networks, as explained in the literature [5]. Although these terms can be confusing, there are differences in how the algorithms learn. Both DL and neural networks have been instrumental in accelerating progress in areas such as computer vision, natural language processing and speech recognition, as seen in existing studies by Jiang *et al.,* [6]. Despite these advantages, this technology also comes with drawbacks. One significant limitation is the high cost, including maintenance and repair, as noted in earlier assessments [7]. The software must be regularly updated to meet changing requirements. Additionally, all machines are susceptible to failure and even small calculation errors can result in significant consequences, including the potential loss of critical data. Moreover, there are concerns that automation may lead to unemployment if machines begin to replace human workers.

On the other hand, computer vision still faces significant challenges, especially in colour detection and object recognition. Traditional object detection methods often struggle in real-world scenarios where objects lack consistent, distinguishable features. Colour detection is particularly difficult, as the human eye itself has trouble differentiating between similar colours, as discussed in related works [7,8]. This issue becomes even more pronounced for individuals with colour vision deficiencies, commonly known as colour blindness. In industrial settings—where precise colour recognition is crucial, such as in sorting and grading processes—such limitations can cause inefficiencies, errors and reduced productivity, [9,10]. Furthermore, a series of previous works have highlighted that tracking and detection systems have become critical applications that deserve attention [11–19]. Additionally, many existing object recognition frameworks remain passive and fail to dynamically adjust their detection configurations based on real-time feedback, thus limiting their effectiveness in improving image quality and detection accuracy.

In tackling these challenges, this project presents an innovative solution: an advanced real-time object detection and tracking system powered by convolutional neural networks (CNNs). Our primary aim is to develop a functional prototype for a real-time object detection and tracking system using a Raspberry Pi. This system will proficiently identify and track objects in live video feeds, with a specific focus on colour recognition. Moreover, it will be capable of precisely monitoring and categorizing the colour of chili plants by leveraging CNNs to differentiate between red, green and yellow variations. The system will also dynamically adjust its tracking and detection parameters based on real-time feedback, significantly enhancing accuracy and minimizing errors.

## 2. Methodology

In this project, a Raspberry Pi acts as the central processing unit, interfacing with a camera detector to capture live video feeds. The hardware setup also includes servo motors, a 5 V DC motor, infrared obstacle avoidance sensors and a Liquid-Crystal Display (LCD) display, which collectively enable the system to detect, identify and track objects in real-time. On the software side, the project relies on Python and utilizes libraries such as NumPy and OpenCV for image processing and TensorFlow for implementing the CNN algorithm. The system uses the Hue, Saturation and Value (HSV) colour space instead of the traditional Red, Green and Blue (RGB) model due to its closer alignment with human colour perception, making it more effective for tasks involving colour detection. Additionally, the system implements data augmentation techniques to improve the robustness of the CNN by increasing the size and variability of the training dataset, which includes 660 augmented images of chili plants in different colours—red, green and yellow.

### *2.1 CNN Model Design*

As stated by Ullah *et al.,* [20], CNN has emerged as one of the most impactful and widely adopted techniques in the field of deep learning. Figure 1 illustrates the CNN model in this project's three-dimensional (3D) visualised model. The model consists of convolutional blocks, pooling blocks, as well as fully connected blocks. Python code is written to implement the designed convolutional neural network into an executable file. Additionally, the entire augmented dataset was duplicated and resized. The process was done using the cv2.resize() function which can be found in the OpenCV python library to produce a new dataset with the same augmentations but resized into 28x28 dimensions from the original cropped 18x28 dimension. The product of this process is three folders that are used to contain the dataset images of each corresponding chili character.

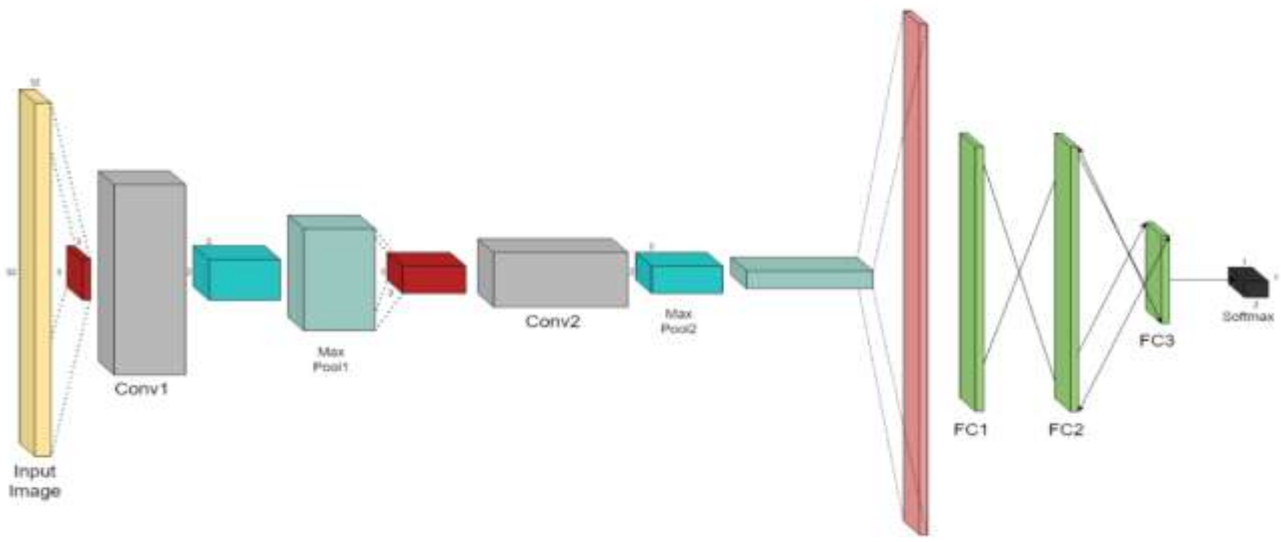


**Fig. 1.** CNN model

The convolutional layers in this model use a Rectified Linear Unit (ReLU) activation function and digital image padding to ensure that the output image is the same size as the input image. Each convolutional layer employs a 3x3 kernel window for image convolution. The pooling layers utilize the max pooling method, which is popular and reliable in many convolutional neural network models. The pooling layer has a 3x3 pooling kernel size with a stride of three. Its purpose is to consolidate the learned features from the images into a more generalized format, reducing overfitting during training. Once constructed, the convolutional neural networks are saved into an H5-type file, which stores the trained model, its learned parameters and trained weights. This enables saving time by only needing to train the model once, as well as exporting to other machines for classification. The saved model can be loaded into any operating system with compatible Python, Tensorflow and H5py versions.

## *2.2 Experiment Setup*

A project workflow as illustrated in Figures 2 and 3, outlining the steps and decisions involved in the process. The project begins when the power supply activates the entire system. Once powered up, the conveyor starts moving and the infrared obstacle avoidance sensor detects chili within a predefined range. The Raspberry Pi 4, running code in the Thonny Python IDE, attempts to establish a connection; if no connection is found, it continuously retries until successful. The camera and image processing component are crucial, with the camera acting as a virtual sensor and the infrared sensor counting the total number of chilies passing through. Image processing involves various techniques to extract image information, with pre-processing steps stored in the model.h5 file. These steps include acquisition, rotation, compression, brightness adjustment and sharpening. When the first infrared sensor detects an object, the camera initializes the chili data. If an object is detected, the image processor converts the image to the HSV colour space and identifies the colour range that matches the captured image. Next, the system uses a CNN for object recognition, classifying the type of chili. The processor identifies the contours of the image, focusing on the largest contour area. After passing through the camera section, the chili is detected by a second infrared sensor, which determines the chili type and displays the total count on the LCD screen. The system has three output containers: container 1 is set to 0° for red chili, container 2 is set to 45° for yellow chili and container 3 is set to 90° for green chili.

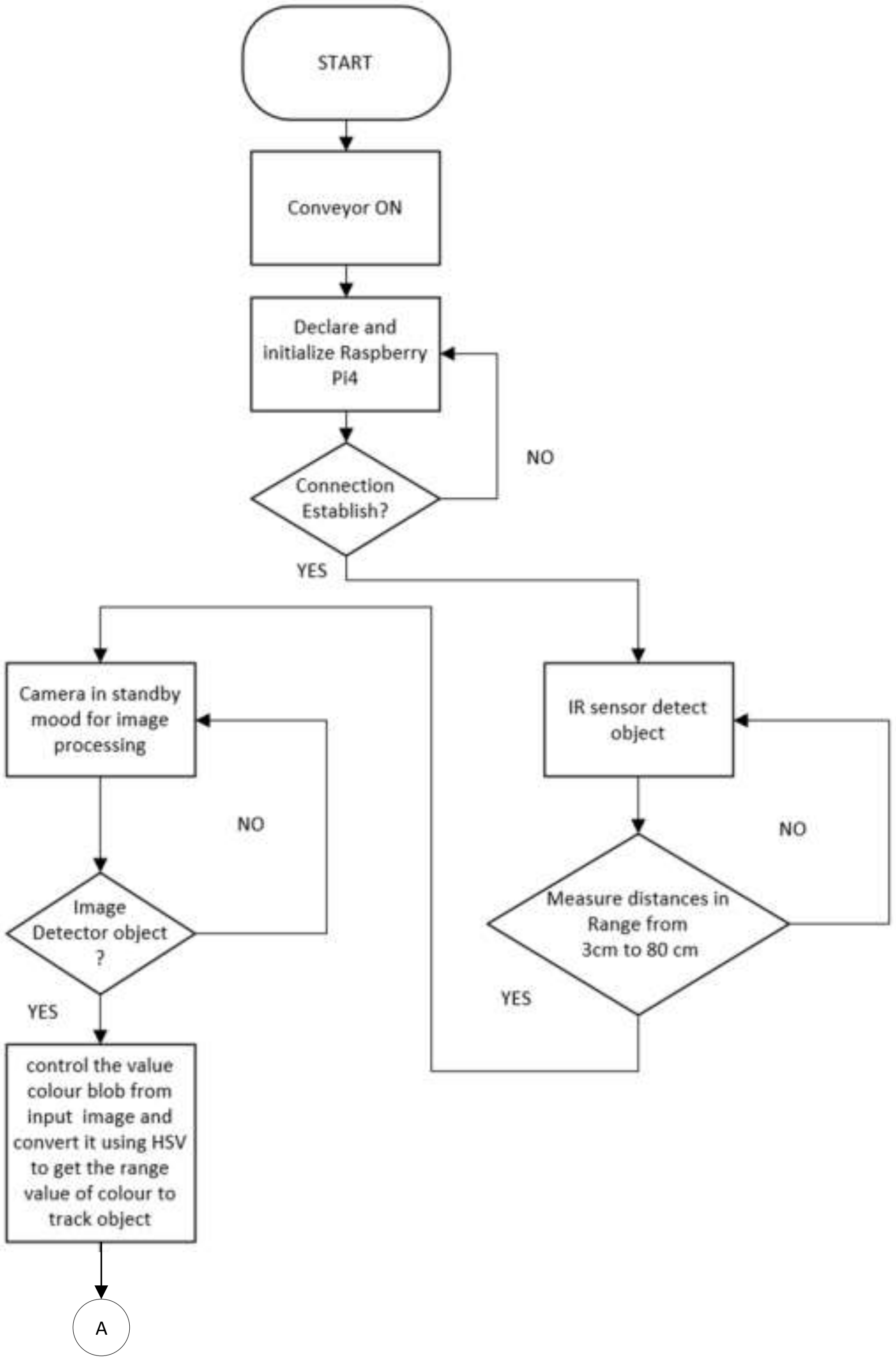

START
Conveyor ON
Declare and initialize Raspberry Pi4
Connection Establish?
NO
YES
Camera in standby mood for image processing
NO
Image Detector object ?
YES
control the value colour blob from input image and convert it using HSV to get the range value of colour to track object
A
IR sensor detect object
NO
Measure distances in Range from 3cm to 80 cm
YES

**Fig. 2.** Project workflow for the conveyer

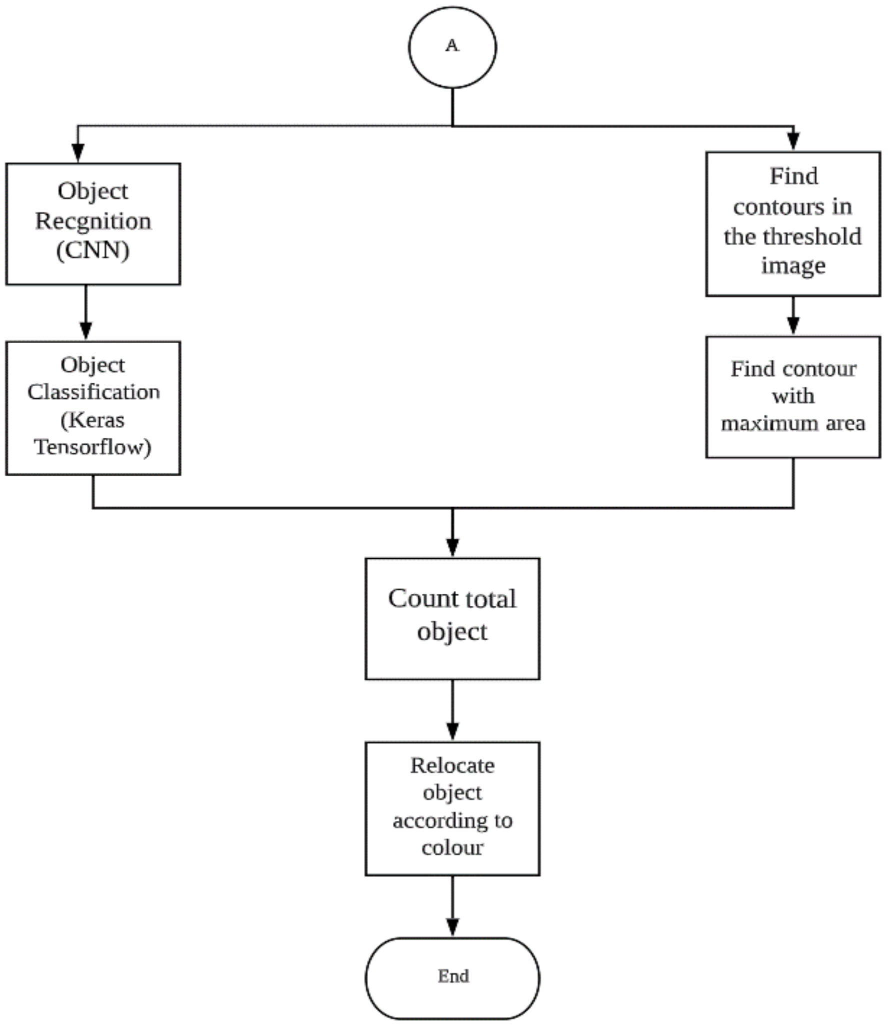

A
Object Recgnition (CNN)
Object Classification (Keras Tensorflow)
Find contours in the threshold image
Find contour with maximum area
Count total object
Relocate object according to colour
End

**Fig. 3.** Project workflow for the image processing

## *2.3 Project and Circuit Design*

The prototype design incorporates an innovative design pattern used in software development, including Tinkercad, AutoCAD or Solidworks. Tinkercad software was specifically utilized to create the project drawing, as depicted in Figure 4.

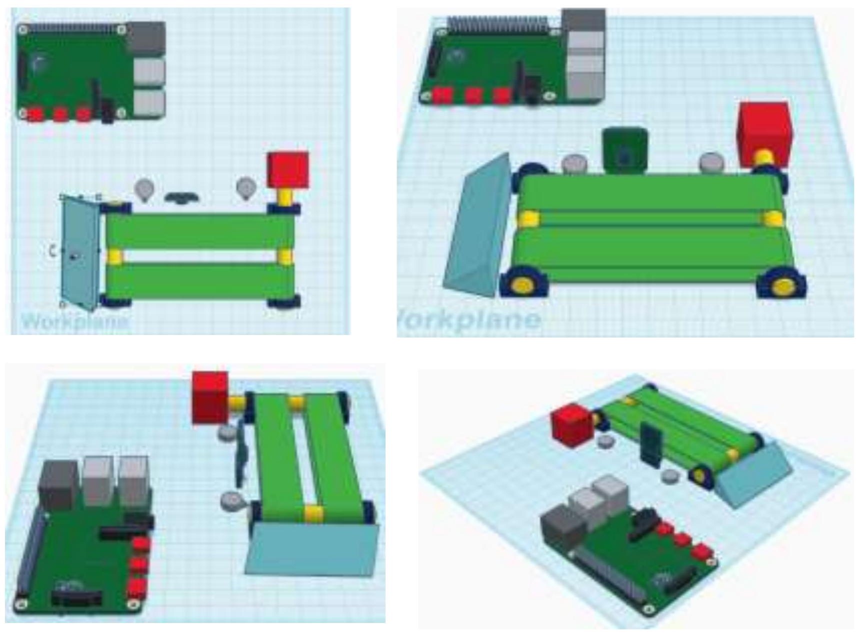


**Fig. 4.** Project design

Furthermore, Figure 5 presents a circuit diagram that was meticulously designed before developing the prototype to ensure seamless system development. This circuit diagram serves as a critical reference for the user during the development phase. Additionally, the meticulous circuit design helps to avert incorrect wiring, thereby minimizing the risk of short circuits or damage to electronic components due to lapses in concentration or human error.

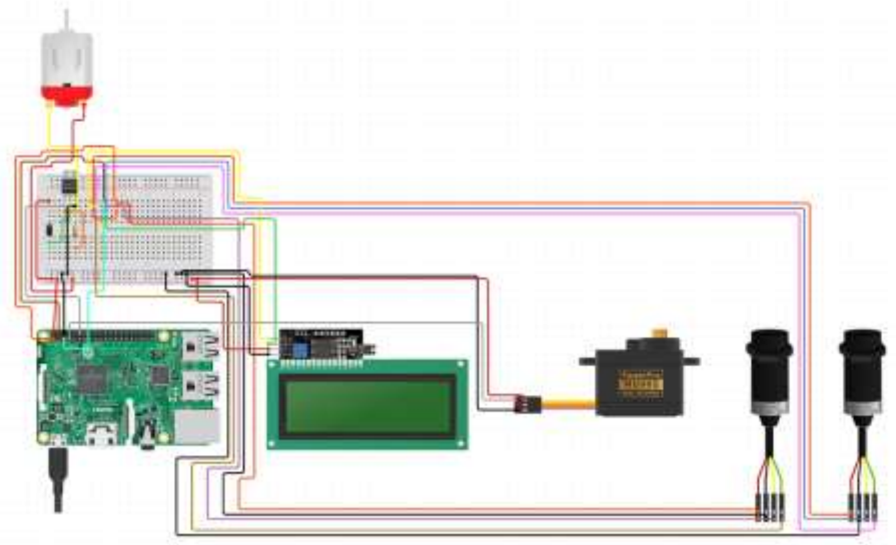


**Fig. 5.** Circuit design

## 3. Results
### *3.1 Hardware Development*

Figure 6 illustrates the prototype of the tracking and sorting device. A power bank provides a 5 V supply to power the entire system. The prototype incorporates two types of sensors: a camera and an IR sensor, both strategically placed on the conveyor to detect objects. The camera, a Logitech model, streams video at a resolution of 720 pixels, making it suitable for this project. The IR sensor, capable of detecting distances between 20 and 300 mm, calculates these distances using OpenCV. The Raspberry Pi 4B was selected as the main controller due to its powerful 1.5 GHz Central Processing Unit (CPU) and 4 GB Random Access Memory (RAM), ensuring high processing speed. At the end of the conveyor, a servo motor is installed to sort and position the detected objects accurately.

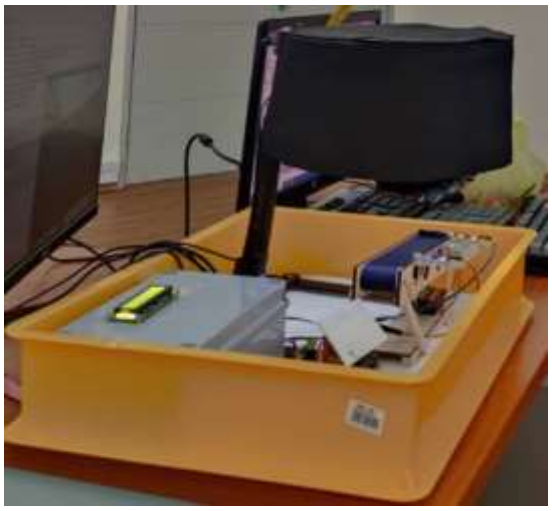


**Fig. 6.** Prototype of the tracking and sorting device

### 3.2 Software Development

The entire project complete programming was done using Python version 3.7 inside the Visual Studio 2019 Community IDE. Several Python scripts were written independently to allow for faster and more convenience during testing and debugging phases. For example, the dataset augmentation and neural network training codes were written in separate Python text files. All Python files are saved into the local machine initially, which is then uploaded to GitHub.

#### 3.2.1 Variable colour masking HSV

A masking task requires upper and lower boundary conditions, all the pixels with values between the boundaries are kept one while others are kept zero in a new binary image. These boundaries values, depend on required segmentation and the colour model of the image. For this project it is focusing on three colour that represent three type of chili character red colour, green colour and yellow. In HSV model, hence this value is varied in each frame. Mean and standard deviation of the image is used to train a regression model, which gives a linear equation to calculate the lower boundary of the mask. Figure 7 shows the example of the chili dataset. The collected data is composed of three folders, each with 660 augmented images. After augmentation, the data is split into training and validation sets. The 660 images for each chili character are divided 1:4, with 495 images in the training set and 165 in the validation set. The original images are 18x28x3 pixels and the augmented images have the same dimensions.

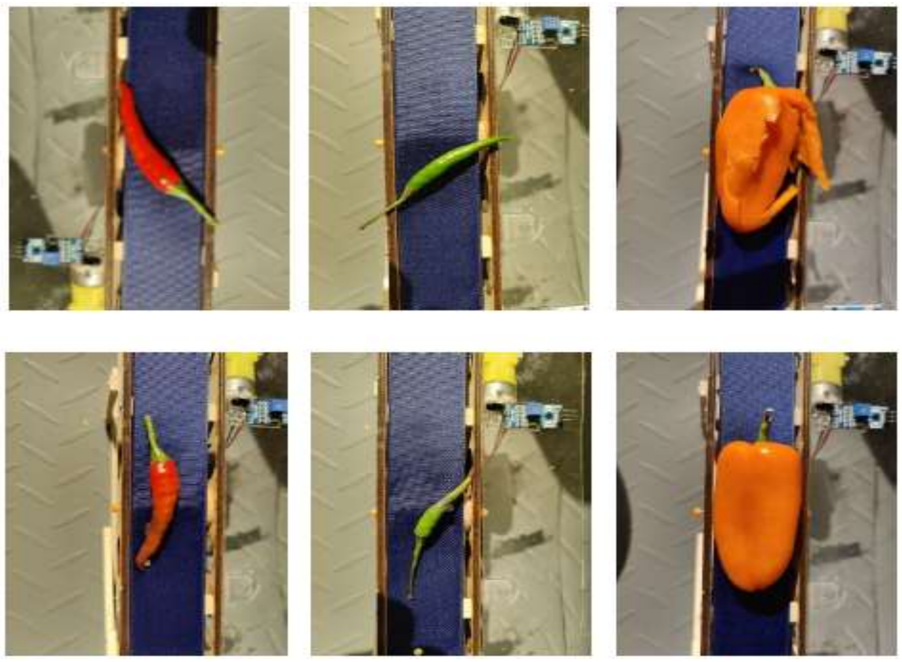
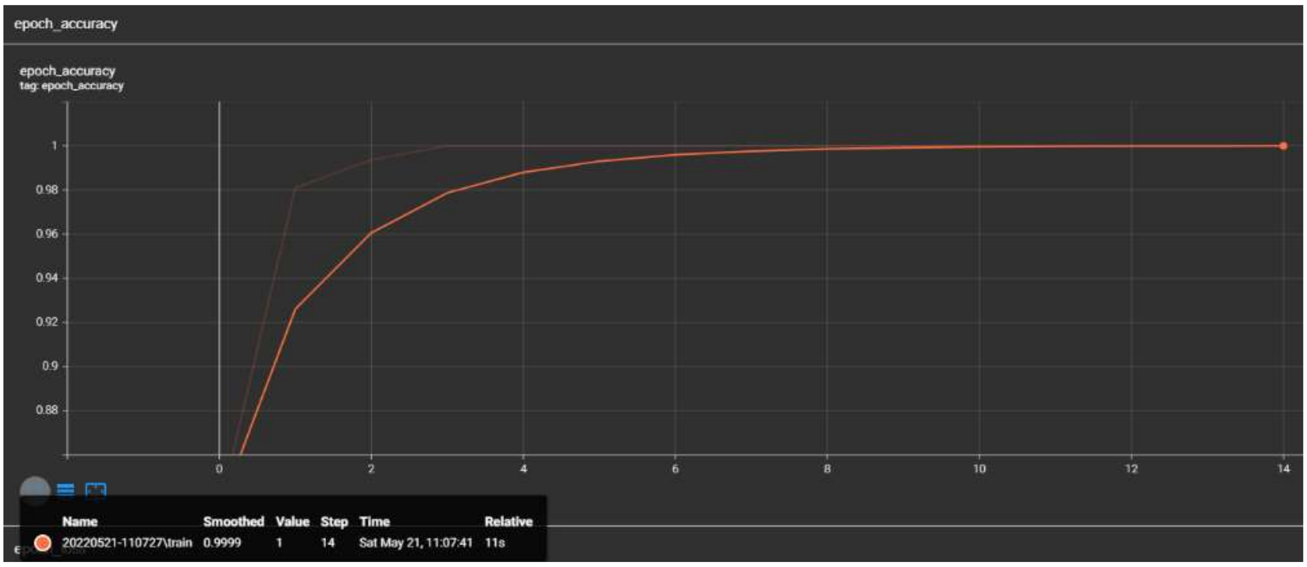


**Fig. 7.** Chili dataset

### *3.2.2 Operating CNN*

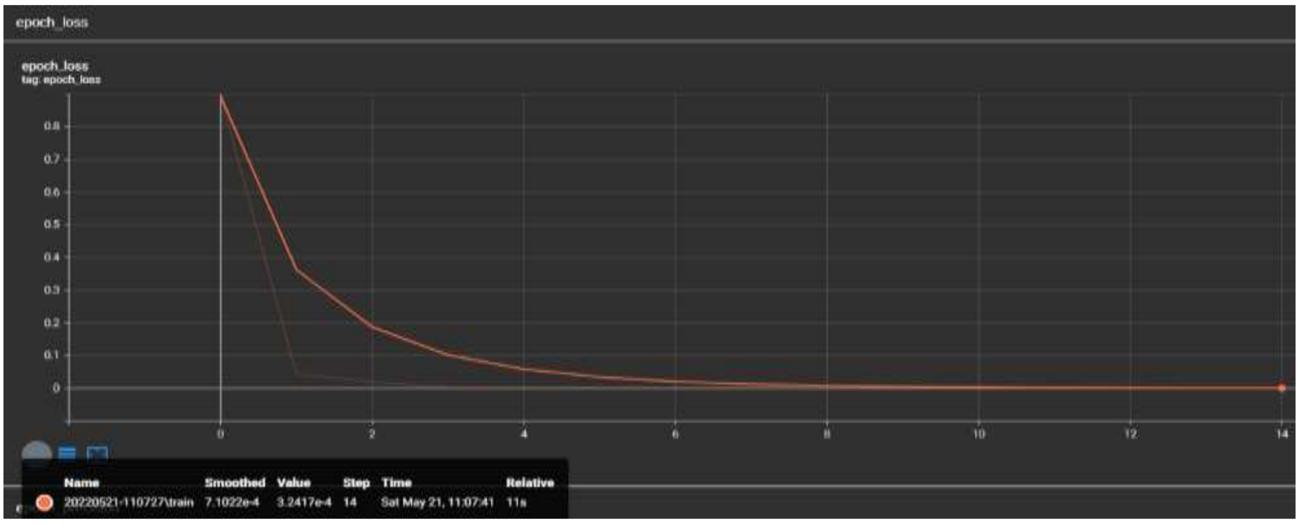
For initial testing, the neural network's accuracy is measured using basic ratio of correct and incorrect predictions for each of the datasets. The conclusive measurement is taken from the last training epoch results, which is the 5th epoch. During this final training epoch, the accuracy value is found to be 1.000 (100 %) and 1.000 (100 %) for both the training dataset as well as validation dataset respectively. Figure 8 shows the graph of the epoch accuracy. An epoch accuracy is used to measure the dataset performance in an interpretable way. The accuracy of a model is using to determine the model parameters. It is to measure of accuracy model prediction is compared to the true data.



| Name | Smoothed | Value | Step | Time | Relative |
|---|---|---|---|---|---|
| ● 20220521-110727\train | 0.9999 | 1 | 14 | Sat May 21, 11:07:41 | 11s |

**Fig. 8.** Epoch accuracy

Figure 9 illustrates the epoch loss. The epoch loss is calculated on training and validation and its interpretation. It is the sum of errors made for each example in training or validation sets. Loss value implies how poorly or well a model behaves after each iteration of optimization.



| | Name | Smoothed | Value | Step | Time | Relative |
|---|---|---|---|---|---|---|
| ● | 20220521-110727\train | 7.1022e-4 | 3.2417e-4 | 14 | Sat May 21, 11:07:41 | 11s |

**Fig. 9.** Epoch loss

After implementing padding, the model's performance improved to 100 % by the 15th epoch. Despite the model consisting of a single long short-term memory (LSTM) layer, the padded data size is significantly larger compared to previous methods by Ullah *et al.,* [20], as it is padded to the length of the longest review. This approach ensures a nearly fully populated array with data rather than zeroes. While increasing the number of epochs slightly reduced the model's performance, it did not lead to overfitting. The training loss reflects how well the model fits the training data, whereas the validation loss shows how well the model generalizes to new data. To develop effective CNN models, the validation accuracy should continue to rise and the error rate should continue to fall in conjunction with the training error.

### *3.2.3 Colour chili recognition*

The operating system for sorting devices uses a camera to evaluate the accuracy of the system's classification by using HSV. Several types of chilies, such as red, yellow and green chili, need to be tested for accuracy using the camera. There are 30 chilis to be tested. The camera is positioned facing down at a 90° angle with support from an O-ring light to ensure a fixed light intensity in the area. Table 1 records the accuracy test between the camera and colour chili.

**Table 1**
The accuracy test between camera and colour chili

| Object | Reaction (Red) | Reaction (Yellow) | Reaction (Green) |
|---|---|---|---|
| Chili 1 (Red) | Detected | Undetected | Undetected |
| Chili 2 (Red) | Detected | Undetected | Undetected |
| Chili 3 (Green) | Undetected | Undetected | Detected |
| Chili 4 (Yellow) | Undetected | Detected | Undetected |
| Chili 5 (Yellow) | Undetected | Detected | Undetected |
| Chili 6 (Green) | Undetected | Undetected | Detected |
| Chili 7 (Red) | Detected | Undetected | Undetected |
| Chili 8 (Yellow) | Undetected | Detected | Undetected |
| Chili 9 (Red) | Undetected | Detected | Undetected |
| Chili 10 (Yellow) | Detected | Undetected | Undetected |
| Chili 11 (Green) | Undetected | Undetected | Detected |
| Chili 12 (Red) | Undetected | Detected | Undetected |
| Chili 13 (Green) | Undetected | Undetected | Detected |
| Chili 14 (Yellow) | Undetected | Detected | Undetected |
| Chili 15 (Green) | Undetected | Undetected | Detected |
| Chili 16 (Red) | Detected | Undetected | Undetected |
| Chili 17 (Yellow) | Undetected | Detected | Undetected |
| Chili 18 (Green) | Undetected | Undetected | Detected |
| Chili 19 (Green) | Undetected | Undetected | Detected |
| Chili 20 (Yellow) | Undetected | Detected | Undetected |
| Chili 21 (Yellow) | Detected | Undetected | Undetected |
| Chili 22 (Green) | Undetected | Undetected | Detected |
| Chili 23 (Red) | Detected | Undetected | Undetected |
| Chili 24 (Red) | Undetected | Detected | Undetected |
| Chili 25 (Yellow) | Undetected | Detected | Undetected |
| Chili 26 (Red) | Detected | Undetected | Undetected |
| Chili 27 (Green) | Undetected | Undetected | Detected |
| Chili 28 (Green) | Undetected | Undetected | Detected |
| Chili 29 (Yellow) | Detected | Undetected | Undetected |
| Chili 30 (Red) | Undetected | Detected | Undetected |

Figure 10 shows the results of the accuracy test for chili recognition. The numbers indicate the accuracy of image processing in sorting chilies using this system. According to the data, green chilies are more accurately recognized compared to other colours, as the HSV concentration range for green is set within an ideal range. There were no errors reported for green chilies. However, the system is also able to recognize red and yellow chilies, but the camera's sensitivity and the colour transition between red and yellow can lead to some inaccuracies in detecting these colours. The analysis also indicates that the camera's angle in relation to the object is crucial for accurate detection. If the camera is placed too close to the object, it may not be able to autofocus properly. Therefore, maintaining a fixed angle for the camera can reduce errors in assessing an object.

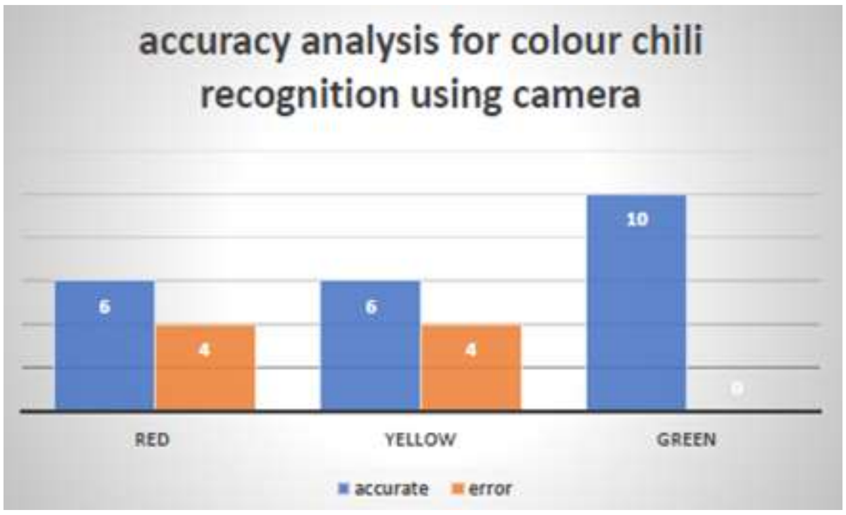


**Fig. 10.** Bar chart of the accuracy test chili recognition

## 4. Conclusions

The prototype of image processing model for real-time video tracking system using Raspberry pi is developed in this project. In addition, the experiment data of the chili plant using CNN algorithms also presented here. More experiments using bigger number of epochs are required to verify the complicated algorithms.

These future improvements will help this project to perform even better, efficiently and effectively. Although the obtained performance is good, but the model can still be improved. The first improvement is by using a Recurrent Neural Network (RNN) it has a multiple Long short-term Memory (LSTM) layer due to its better performance as it builds a nice hierarchical level of features. Next, using multiple LSTM layers from RNN to create a dropout layer should be considered as it can help prevent overfitting. Lastly, create an automated conveyor for consistencies in subject movement from point A to point B. The significance of this project would be the contribution of combining two types of methods used. The combination of CNN as well as using colour detection algorithm commonly known as HSV. Other than that, they use Kaggle.com as a data set storage which can be easily accessed publicly by anyone, they can apply the data set for personal use or improve it further depending on their use case.

## Acknowledgement
The authors would like to thank the Universiti Kuala Lumpur Malaysian Institute of Industrial Technology for providing facilities for this project. This project was not funded by any grant.

## References
[1] Cho, Dae-Yong and Min-Koo Kang. "Human gaze-aware attentive object detection for ambient intelligence." *Engineering Applications of Artificial Intelligence* 106 (2021): 104471. https://doi.org/10.1016/j.engappai.2021.104471
[2] Ranjan, Rajeev, Vishal M. Patel and Rama Chellappa. "A deep pyramid deformable part model for face detection." In *2015 IEEE 7th international conference on biometrics theory, applications and systems (BTAS)*, pp. 1-8. IEEE, 2015. https://doi.org/10.1109/BTAS.2015.7358755
[3] Burns, E., Laskowski, N. and Tucci, L. "What is Artificial Intelligence?" (2021). https://searchenterpriseai.techtarget.com/definition/AI-Artificial-Intelligence
[4] Dakshinamoorthy, Prabakar, Gnanajeyaraman Rajaram, Prabhu Murugan, A. Manimaran and Ramesh Sundar. "Artificial intelligence algorithms for object detection and recognition in video and images." *Multimedia Tools and Applications* (2025): 1-18. https://doi.org/10.1007/s11042-025-20635-2

[5] Alzubaidi, Laith, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, José Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie and Laith Farhan. "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions." *Journal of big Data* 8 (2021): 1-74. https://doi.org/10.1186/s40537-021-00444-8

[6] Jiang, Weiwen, Jinjun Xiong and Yiyu Shi. "A co-design framework of neural networks and quantum circuits towards quantum advantage." *Nature communications* 12, no. 1 (2021): 579. https://doi.org/10.1038/s41467-020-20729-5

[7] Kumar, P., Raj, R. and Vishwakarma, S. "Novel Camera Security Alert IOT System." International Journal of Innovative Technology and Exploring Engineering (IJITEE). 8(12). (2019): 1660-1662. https://doi.org/10.35940/ijitee.L3164.1081219

[8] Zarkasi, Ahmad, Siti Nurmaini, Deris Stiawan and Cora Deri Amanda. "Implementation of fire image processing for land fire detection using color filtering method." In *Journal of Physics: Conference Series*, vol. 1196, no. 1, p. 012003. IOP Publishing, 2019. https://doi.org/10.1088/1742-6596/1196/1/012003

[9] Zhang, Yan, Jian Lian, Mingqu Fan and Yuanjie Zheng. "Deep indicator for fine-grained classification of banana's ripening stages." *EURASIP Journal on Image and Video Processing* 2018, no. 1 (2018): 1-10. https://doi.org/10.1186/s13640-018-0284-8

[10] Waghmare, Prajakta Mukund and Swati Shankar Shetkar. "Guided Filter for Grayscale, RGB and HSV color space." *International Journal of Engineering and Computer Science* 5 (2016): 12.

[11] Honeycutt, Wesley T. and Eli S. Bridge. "Uncanny: Exploiting reversed edge detection as a basis for object tracking in video." *Journal of Imaging* 7, no. 5 (2021): 77. https://doi.org/10.3390/jimaging7050077

[12] Liu, Ying. "RETRACTED: Human motion image detection and tracking method based on Gaussian mixture model and CAMSHIFT." (2021): 103843. https://doi.org/10.1016/j.micpro.2021.103843

[13] Lyu, Ye, Michael Ying Yang, George Vosselman and Gui-Song Xia. "Video object detection with a convolutional regression tracker." *ISPRS journal of photogrammetry and remote sensing* 176 (2021): 139-150. https://doi.org/10.1016/j.isprsjprs.2021.04.004

[14] Margapuri, Venkat. "Smart motion detection system using raspberry pi." *arXiv preprint arXiv:2006.06442* (2020).

[15] Mas, Juan, Teodoro Panadero, Guillermo Botella, Alberto A. Del Barrio and Carlos García. "CNN Inference acceleration using low-power devices for human monitoring and security scenarios." *Computers & Electrical Engineering* 88 (2020): 106859. https://doi.org/10.1016/j.compeleceng.2020.106859

[16] Hussien, Habib Mohammed. "Detection and tracking system of moving objects based on MATLAB." *International Journal of Engineering Research and Technology (IJERT)* 3, no. 10 (2014).

[17] Parasher, Rhitivij and Preksha Pareek. "Event triggering Using hand gesture using open cv." *International Journal Of Engineering And Computer Science* 5 (2016). https://doi.org/10.18535/ijecs/v5i2.4

[18] Patil, Suraj Pramod. "Techniques and methods for detection and tracking of moving object in a video." *J. Innov. Res. Comput. Commun. Eng.* 4, no. 5 (2015): 8116-8120.

[19] Zarkasi, Ahmad, Siti Nurmaini, Deris Stiawan and Cora Deri Amanda. "Implementation of fire image processing for land fire detection using color filtering method." In *Journal of Physics: Conference Series*, vol. 1196, no. 1, p. 012003. IOP Publishing, 2019. https://doi.org/10.1088/1742-6596/1196/1/012003

[20] Ullah, Md Aman, Abdul Aziz K. Abdul Hamid, Muhamad Safiih Lola, R. U. Gobithaasan and Habiba Sultana. "GF-CNN: A Hybrid Approach for Pollen Recognition Combining Gabor Filters and Convolutional Neural Networks." *Journal of Advanced Research Design* 127, no. 1 (2025): 120-136. https://doi.org/10.37934/ard.127.1.120136