Pertanika J. Sci. & Technol. 34 (5): 3109 - 3130 (2026)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Mobile-Based Inventory Counting Using 2D Object Detection and Layer-Based Inference

Gan Heng Lai[1], Goh Eg Su[1,2]*, and Johanna Ahmad[1]

[1]Faculty of Computing, Universiti Teknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia
[2]Media and Game Innovation Centre of Excellence (MaGICX), Institute of Human Centred Engineering (iHumEn), Universiti Teknologi Malaysia, 81310 Johor Bahru, Johor, Malaysia

ABSTRACT

The term "2D" refers to layer-based quantity inference from a spatially stacked arrangement and does not indicate direct 2D bounding-box regression or depth reconstruction. YOLOv8 is a deep learning framework that enables the development of exceedingly fast computer vision applications. It is adept at object detection and classification, including multi-object detection. Computer Vision explores methods for reconstructing, interpreting, and understanding a scene's three-dimensional structure from its two-dimensional images, leveraging the scene's inherent structure. It involves modelling and simulating human vision using both hardware and software. In this project, YOLOv8 was used as the processing algorithm, integrated with other tools and libraries to build a mobile-based inventory-counting application using 2D object detection and layer-based inference. It is a design that will be used for stock checking or dispatching goods from stock. The input will be the mobile phone's front camera for capturing the boxes. The scene will be captured from the surrounding area of the stock area, where the rack is located. To train the model, a custom dataset of 1,500 images containing three cardboard box sizes was utilised, annotated via the Roboflow platform. Separate YOLOv8 models were trained for the Small, Middle, and Big box datasets. Bounding-box detection was evaluated on labelled validation images using mAP@0.5 and mAP@0.5:0.95, while the complete mobile inventory-counting application was evaluated separately through 250 controlled trials for each box-size model. Finally, the system demonstrated its effectiveness in application. It accurately recognised and measured various box types, assuming there were no vacant spaces within a 3 × 3 × 3 stacking configuration. The variation in box size enables an in-depth evaluation of the device's effectiveness across a range of in-stock items.

Keywords: Artificial intelligence, computer vision, inventory system, object detection, YOLOv8

ARTICLE INFO

Article history:
Received: 30 July 2026
Accepted: 05 October 2026
Published: 30 October 2026

DOI: https://doi.org/10.47836/pjst.34.5.23

E-mail addresses:
ganhenglai1030@gmail.com (Gan Heng Lai)
eg.su@utm.my (Goh Eg Su)
johanna@utm.my (Johanna Ahmad)
* Corresponding author

e-ISSN: 2231-8526

© Universiti Putra Malaysia Press

INTRODUCTION

The application of vision technology for computers across many businesses has brought a revolutionary change to traditional methods. It has provided novel solutions for processes such as inventory management, object detection, and visual data analysis over the past few years (Javaid et al., 2022). Using advanced algorithms to process camera-captured visual data, computer vision systems can automate and improve processes that were previously reliant on human intervention and error-prone (Radanliev et al., 2021). This technology was widely used across industries such as retail, logistics, manufacturing, and healthcare. Its integration with automated stock management systems has brought about significant changes and is having a major impact on how businesses manage, monitor, and control their inventory levels (Verhoef et al., 2021; Koh & Yuen, 2022).

There are real-world applications demonstrating the impact of applying computer vision technology in managing stock and inventory for large retail and logistics companies. They could be demonstrated by the example of one of the world's retail giants, Walmart, which is famous for its technological advancement and passion for innovation. Automated systems have been implemented at Walmart using computer vision technology in-store to scan inventory levels, monitor shelf elements, and automate restocking (Collins, 2026). It could improve operating efficiency by detecting stock shortages, identifying shelf gaps, and optimising product placement through the deployment of cameras with machine learning algorithms (Toneguzzi, 2022; Gama et al., 2025).

Dalsey, Hillblom and Lynn (DHL), like many other businesses, has suffered from poor stock management because they are holding onto a lot of inventory (DHL Supply Chain, n.d.). This had been especially challenging for the organisation due to low visibility and low stock accuracy. This led to the duplication of valuable commodities across multiple sites (DHL Supply Chain, n.d.). It will lead to excess costs in terms of finance and consumer satisfaction. Thus, DHL chose to apply computer vision technology to address these challenges, recognising the need for a more precise and effective approach to managing their stock (DHL Supply Chain, n.d.). DHL improved the accuracy of its stock counting method, reduced product duplication, and gained better visibility into its stock levels by using high-speed cameras and enhanced image recognition technology (Villegas-Ch et al., 2024).

LITERATURE REVIEW

The integration of digital transformation technologies into inventory management represents a major shift toward automated, error-free operational environments (Sengupta & Dreyer, 2023). This literature review highlights and discusses technologies and studies relevant to the current inventory system's operation and to the limitations of human operation. Central to this transformation is Computer Vision, a field of artificial intelligence

that enables computers to derive meaningful information from digital images and videos. Specifically, the You Only Look Once (YOLO) architecture, and its recent YOLOv8 iteration, have revolutionised real-time object detection by predicting bounding boxes and class probabilities in a single neural network evaluation (Li et al., 2022; Jocher et al., 2023). Exploring these technologies provides the necessary context for identifying the existing gaps in manual inventory systems and evaluating the efficacy of proposed automated alternatives (Gao et al., 2020; Terven & Cordova-Esparza, 2023).

The Process of the Inventory System

The inventory system process begins with purchasing goods and delivering those goods to the company's storage facility. This will start the inventory flow by addressing physical goods, which involve acquiring products needed for operations (Büyüközkan & Göçer, 2018). Determining the appropriate order quantity can be tricky, as the quantity recorded in one's own system may lead to excess stock due to overordering (Fildes et al., 2009). Once purchased, goods are sent directly to the company's warehouse area.

The next step in the inventory system involves evaluating, sorting and storing goods in the assigned stock area. Carefully measuring the number of products purchased was essential at this stage to maintain the accuracy of inventory management (Chen & Patel, 2024; Villegas-Ch et al., 2024). Sorting is grouping products by size, type, and demand frequency, among other criteria. To maintain optimal stock levels and avoid operational disruptions, inventory must be closely monitored. Through periodic physical inventory counts, businesses can closely track stock quantities in sequence, identify stock discrepancies in a timely manner, such as deadstock or stockouts, and then take swift corrective actions.

When it comes to order placement, businesses rely on a mix of internal processes and external suppliers. Once orders are placed, businesses use their tracking systems to monitor shipment status. The key to this phase was attentively checking the number of orders made to ensure the order was completed accurately. When goods were delivered from the warehouse to the customer, inventory records were promptly updated to reflect the latest stock levels. Its purpose was to identify potential stock shortages and adjust their replenishment strategies accordingly (Belyh, 2022).

Limitation of Current System

One major drawback of the current inventory system is the additional complexity it imposes on the employees responsible for managing it (Liboni et al., 2019). Many businesses underestimate how long it takes for new inventory management procedures to be completely understood and accepted by their employees. Inventory management becomes difficult when process leaders are unfamiliar with the new rules, instruments, and frameworks.

Businesses will be required to invest much time and resources in training employees on the specifics of order processing and inventory updating (Nguyen & Tran, 2024).

Another significant limitation is the lack of efficient order and stock management practices. The responsibilities of order management include receiving, monitoring, and updating stock levels in response to customer orders. Businesses risk overselling products if they don't establish an effective order management process. In the absence of appropriate supervision and management protocols, companies would find it difficult to sustain sufficient stock levels. It could result in client discontent and decreased earnings (Belyh, 2022).

Comparison Between Existing Systems and Proposed System

Inventory systems in use today predominantly rely on stationary arrangements, such as fixed barcode scanners or permanently affixed factory cameras along conveyor belts. A primary limitation of these current approaches is that they capture stock only from a single, direct line of sight and usually only the front face of a pallet. Consequently, users are compelled to manually estimate the remaining hidden stock behind the initial item, a process that is highly error-prone and prone to record inconsistencies. However, this proposed system solution will be utilised with small components that can be integrated into any inventory system, offering flexibility and compatibility across various operational setups.

Table 1 presents a comparison between the proposed system with existing systems regarding mobility and cost. It is crucial to study existing systems like Walmart Canada's Out-of-Stock Detection and CountThings to establish a comparative baseline for performance, cost, and mobility in modern inventory operations. Walmart Canada represents a large-scale, enterprise-grade deployment utilising fixed infrastructural cameras, highlighting the high costs and lack of mobility inherent in permanent setups. Conversely, CountThings exemplifies a mobile, template-based software approach that introduces subscription costs and dimensional limitations. Analysing these opposing ends of the technological spectrum justifies the need for the proposed mobile-based inventory-counting

Table 1.
Comparison of The Proposed System with Existing Systems.

System	Mobility	Cost
Walmart Canada's Out-of-Stock Detection (Wassel, 2022)	An operation that operates in a fixed location cannot be migrated.	Depending on the size of the retail operation
CountThings from Photos (CountThings, n.d.)	Count on one side only; it cannot compute quantities in three dimensions	$120 per month or RM1200 per year
2D Object Detection Used for an Inventory System	High flexibility; can be installed on most Android mobile phones.	Low-cost device.

application, which uses 2D object detection and layer-based inference to merge the mobility of smartphone applications with the robust, two-dimensional detection capabilities typically reserved for enterprise AI, while remaining cost-effective for smaller businesses (Redmon et al., 2016; Wang et al., 2023).

Walmart Canada implemented an automated artificial intelligence system in all its locations to provide consumers and associates with a better shopping experience. As they recognise that product unavailability can be frustrating for customers, they utilise Focal Systems to solve the problem by automating out-of-stock detection with cameras. Focal Systems is a technique that utilises the most advanced and proven AI technology in retail, known as FocalOS. It automates and optimises all current human-made decisions. Focal Systems has developed a durable and cost-effective Wi-Fi camera that scans shelves hourly to provide real-time counts of shelf contents throughout the day. Focal can boast accurate computer vision technology because it is trained on over a billion labelled images from global businesses. (*Focal Systems*, n.d.). This system was installed in high-traffic areas throughout Walmart stores. The camera integrated Walmart's existing inventory system to automatically detect stock issues and then provide alerts for out-of-stock items (Toneguzzi, 2022).

CountThings is an automatic counting application for industrial and digital transformation. It consists of techniques for counting objects from photos and videos. This will automatically count the items in the photo on mobile devices and recognise, categorise, and initiate actions using computer vision. This development aims to reduce the error-prone task of manual counting in industrial operations and make it an efficient, effective process. This application requires a model template to allow the app to count the items. It does not work without a template for the things we want to detect. The procedure for this application starts by inserting the counting template before taking a new picture, and then tapping the "count" button summarises the detection result for the user. There is a constraint in this application: the camera's position. The user must make sure the objects are directly across from the camera, not angled toward it. This application only allows free access for the top seven days of a new account. The user must pay in USD for the license on a daily, monthly, or annual basis (CountThings, n.d.).

This system aims to address and complement the weaknesses of existing solutions, such as Walmart's fixed-location camera system and the CountThings application. This proposed solution offers greater versatility than Walmart's system, as it can be deployed anywhere in the inventory area, whereas Walmart's system is limited to fixed locations. Even though CountThings was good at identifying and counting objects when viewed from a single angle, it was much more expensive when used in isolation from inventory management systems in small businesses or companies and was unable to calculate numbers when viewed from three dimensions. This proposed solution can reduce costs and integrate easily into the current inventory system.

The "You Only Look Once" (YOLO) framework is a state-of-the-art family of object detection models in computer vision. Unlike region-based convolutional neural networks (R-CNNs), which require thousands of network evaluations per image, YOLO is a single-shot algorithm that directly classifies objects and predicts bounding boxes in a single forward pass. This project utilises YOLOv8, which introduces an anchor-free detection mechanism. By eliminating predefined anchor boxes, YOLOv8 improves generalisation and learning speed for custom datasets. Furthermore, YOLOv8 incorporates a C2f module and a decoupled head, which significantly enhances the model's ability to distinguish densely packed objects.

METHODOLOGY

The extended waterfall model was chosen in this case because it requires all features or functions to be completed before testing. It eliminates the need for user interaction or communication throughout the project's development stage. Additionally, clarification will be provided on the current state of development for each phase of the extended waterfall methodology. There are five stages in the waterfall methodology: requirements, design, implementation, testing, and maintenance. However, the process in this project was covered only up to the testing phase, as shown in Figure 1. The difference between an extended waterfall and a standard waterfall methodology is that the system is adjusted if accuracy is not met at the evaluation stage. It was evaluated again after the adjustment was completed.

Analysis Phase

The project starts with the Analysis and Requirements Gathering Phase, which discusses case studies related to the project's inventory system. Companies face significant challenges due to inventory discrepancies, including inaccurate item counts and operational inefficiencies. The application was designed for inventory managers and warehouse workers, with a focus on capturing photos from multiple angles to ensure precise counts. The scope was restricted to a single user using the device's front camera to detect three distinct box sizes and shapes within the marked space on a single rack level. There was a user requirement collection process in

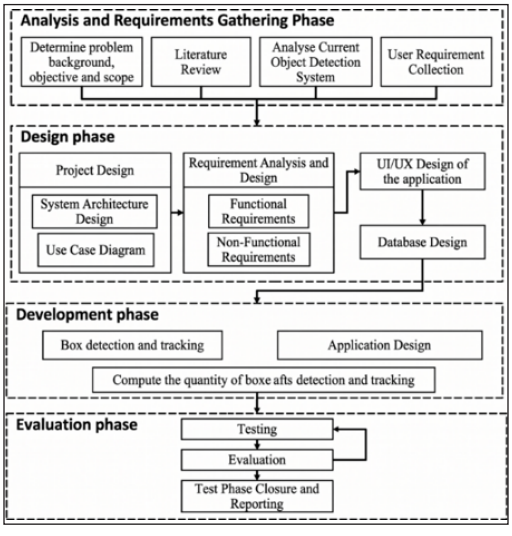


Figure 1. Project methodology phases overview

this analysis stage. A structured approach using Google Forms was used to collect user requirements from the staff members of Purple Fashion Sdn. Bhd. to guide the inventory management application's development.

Design Phase

The interaction between the mobile application and the backend program is the main emphasis of the user-device architecture. The application will be developed using the Flutter framework to capture images of cardboard boxes via the device's camera. The images will be stored and used for object detection processing. The detection result will be stored again in a cloud-based database, Firebase, to store the inventory data. Each inventory record will contain details such as the item ID, the quantity detected, and timestamps. The primary use case is user authentication, allowing users to log in via the mobile app. Image capture and upload is another important use case that allows customers to take pictures of box stores and process them for object detection. The program can identify and count boxes in real-time, showing the user the findings.

Development Phase

The first step in the development stage will be detecting cardboard boxes using YOLOv8, implemented in Python within a Jupyter Notebook environment. The detection process begins with the YOLOv8 backbone extracting low-, medium-, and high-level feature maps from the captured warehouse images. The model then utilises anchor-free detection to predict the midpoint of each cardboard box, drawing precise bounding boxes around them. The C2f module integrates contextual information with high-level features to improve detection accuracy and model performance. By processing the spatial coordinates of these bounding boxes, the system categorises the detected objects into varying hierarchical layers. It was decided to use only the essential object detection and tracking components of YOLOv8 (Zhao et al., 2019; Vijayakumar & Vairavasundaram, 2024). It achieves greater accuracy with deep learning techniques than traditional computer vision algorithms. This improved method dispenses with the need for supplementary algorithms that were to be developed in C++.

Testing Phase

This stage included evaluating the system's functionality and assessing the detector's accuracy across a range of samples. The project primarily focused on accuracy percentage testing, as system accuracy is highly dependent on the arrangement of boxes and the angles at which images are captured. A specialised user testing session will be conducted in collaboration with Purple Fashion Sdn. Bhd. to evaluate the system in an active retail environment. A total of 5 participants from the inventory staff will be invited to test the

application. The specific use case focuses on autonomously managing the remaining stock on the racks, eliminating the need for manual counting. The number of participants was limited to one individual to maintain strict control over the restricted stock arrangement order, ensuring the box patterns were placed sequentially without unwanted repetition that could skew the algorithm's performance data. These participants will be given specific use cases and guidance to arrange the boxes and capture images to test the system's detection capabilities.

DESIGN

The various modules in the use case model of this research-prototype are designed to handle specific functionalities in Figure 2. The main modules were the User Authentication module, Image Processing module and Object Detection module, which consist of several use case functions that establish the interactions between the user and the system.

The primary use case within the User Authentication module is the Login Account function. This scenario enables users to verify their identity by inputting their username and password. The Image Processing module manages all aspects associated with capturing and processing box images. It involves various essential use cases such as Select Template, Take Photo, and Check Result. The Select Template use case lets you choose from a range of box detection templates offered by the system. The system provided multiple detection algorithms designed to accommodate different box types, ensuring flexibility in box detection.

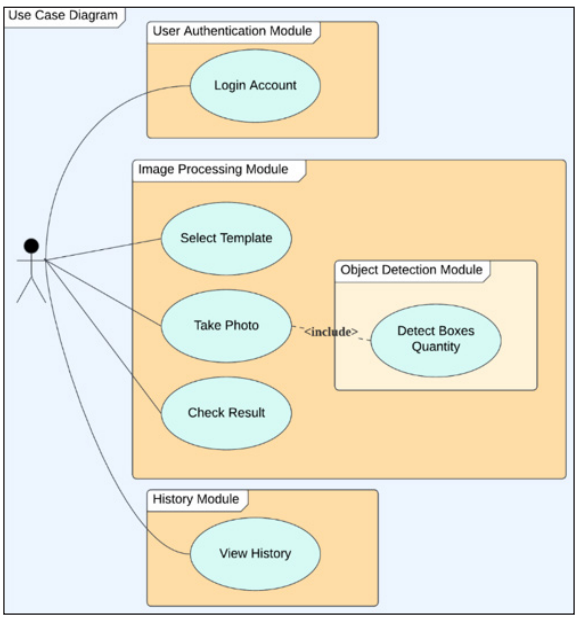


Figure 2. Use case diagram of the proposed research prototype

The third module was the Object Detection module, which is responsible for detecting and computing cardboard boxes in images. It consists of a use case, Detect Boxes Quantity, to perform actual detection and counting of boxes in the captured image. By leveraging the YOLOv8 algorithm's capabilities, the system was able to analyse the image and accurately identify and count the boxes. The system processes the image with the chosen detection template and calculates the number of boxes based on the detected features. The last module was the History module, used for the history review display to the users. The primary use case for this module is the view history function. This function was integrated to allow the user to review the box detection result saved from the previous operation. This allowed multiple users to check the latest updated stock quantity status.

IMPLEMENTATION

This section describes the justification for Open-Source Computer Vision Library (OpenCV) techniques and the implementation of a new box detection algorithm, model training, and application development.

Justification For Deprecating OpenCV Techniques

Two standard OpenCV methods were tested for the box computation algorithm: Harris Corner Detection and FLANN-based feature matching (Bradski & Kaehler, 2008). The Harris Corner Detection algorithm was first created to detect the corners of cardboard boxes (OpenCV, n.d.-b). It served as a clear marker for counting boxes and marked the corners with blue-coloured points. However, its use within real warehouse environments showed a few major shortcomings. The algorithm showed trouble distinguishing true corners on the box from bordering background features in Figure 3. While human viewers can easily notice the high visual contrast between the container edges and backgrounds, the algorithm frequently struggled to distinguish corners from neighbouring features. Real cardboard containers in warehouse environments led to high false-detection rates for corners. This Harris Corner Detection algorithm will also mistakenly detect scores of areas that were not corners, especially within backgrounds containing complex textures.

The FLANN was tested as one possible technique for detecting identical boxes imaged from different camera angles, as shown in Figure 4 (*OpenCV*, n.d.-a). However, the experiment's outcome showed significant limitations to its practical application. While aligning features from



Figure 3. Harris corner detection result

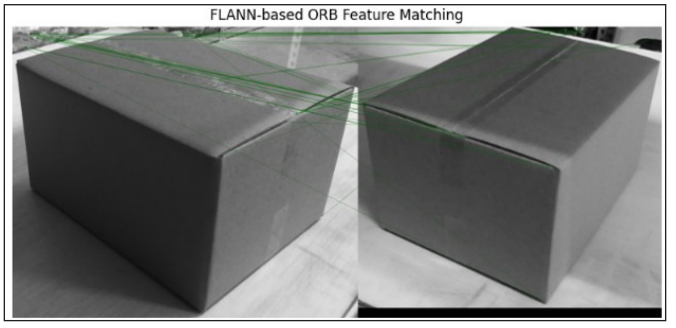


Figure 4. FLANN-based matching result

two images of the same box taken from different angles, the algorithm often misidentified corners as features that were anything but corners, generating false correspondences. Features from one box often ended up being matched to features from different positions on the box, or even from the background, creating erroneous box identifiers.

The comparison between Harris Corner Detection and FLANN-Based Matching showed that, though both are theoretically grounded, they did not meet the practical requirements for accurate identification and counting of boxes in warehouse environments. Their high rate of false positives and variability in feature correspondences would have led to significant errors in the procedure for counting the boxes. These preliminary tests were used for feasibility assessment rather than as formal performance baselines because the OpenCV methods do not directly generate equivalent box-level and layer-level detection outputs.

Implementation of Enhanced Box Detection Algorithm

Based on previous studies on the existing algorithms for both Harris Corner Detection and FLANN-based algorithms, an enhanced algorithm is implemented by eliminating the need for corner detection and feature matching by focusing on a crucial insight into the holistic layout of boxes in warehouse environments. The basic premise of the suggested algorithm is to detect and count boxes based on their spatial position within a hierarchical system. The proposed algorithm classifies the detected boxes into three different layers: Layer 1, Layer 2, and Layer 3, rather than attempting to find every individual box. These layers correspond to ground-level, middle-level, and top-level boxes, respectively, within the hierarchical system. The merit of this proposed algorithm lies in its ability to accurately detect the topmost layer of boxes, which follow stacking principles, thereby enabling the overall estimation of the total number present within the entire stack.

The algorithm implements a structured decision-making process based on predefined layer classes. When the arrangement has 9 or fewer boxes, they are classed as Layer 1 boxes, and the system then calculates the total number of objects detected directly. When

the arrangement involves about 10 to 18 boxes, they were classed as Layer 2 boxes, which then indicates the presence of an overlying layer. In this model, the total count of boxes was calculated by adding recognised Layer 2 boxes to a base of 9, the minimum count for the stacked first-layer configuration. For example, it calculates a total count of 11 boxes if the system recognises two Layer 2 boxes (9 base boxes and 2 Layer 2 boxes). If the arrangement ranges from about 19 to 27 boxes, the system detects Layer 3 boxes, which indicate the presence of a third tier of stacking. When the system detects Layer 3 boxes, it first assumes an overall total of 18, then adds the number of detected Layer 3 boxes to this total to determine the final count. For instance, the system calculates an overall total of 22 when detecting four Layer 3 boxes (18 base boxes plus 4 detected Layer 3 boxes). The minimal pseudocode is: if Layer 3 is detected, set total to 18 + Layer 3 count. Else if Layer 2 is detected, total = 9 + Layer 2 count. Else, total = Layer 1 count.

However, several assumptions are considered to guarantee the precision of this calculation, as shown in Figure 5. Firstly, it was necessary to arrange the boxes so that the bottom layer was fully occupied before placing any boxes on the upper layers. Secondly, there should be no gaps between the boxes, as they may lead to inaccurate volume calculations. Lastly, the boxes should be arranged in rows to avoid isolated boxes in a corner. The boxes should be arranged to connect to each other and cannot be a box standing alone in a corner. The 3 × 3 × 3 configuration represents the supported operating domain of the current prototype and should not be interpreted as a general assumption about all warehouse arrangements. Irregular stacks, vacant positions, isolated boxes, and partially filled lower layers are outside the scope of the present layer-based calculation.

Implementation of Model Training

During the implementation phase, the YOLOv8 model was trained specifically for box detection and the corresponding box labels, but not for detecting arbitrary features or corners. This was done to enhance the model by distinguishing the boxes placed at varying levels, even when they had the same appearance and shape. During training, the model was

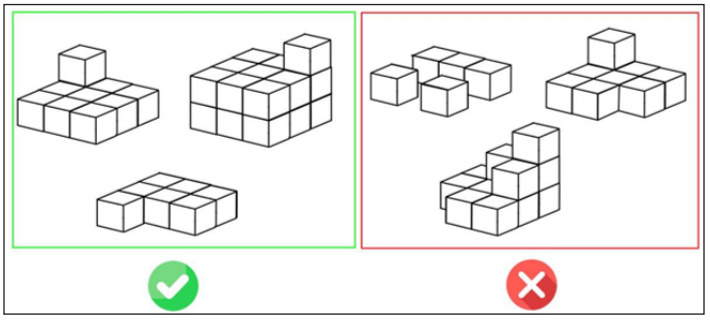


Figure 5. Box arrangement assumptions example

trained on images containing different box configurations, with each box labelled based on its location across multiple layers. It helped the neural network detect ground-level and higher-level placed boxes.

The model was trained on three different datasets using the same methodology. The same code structure and training pipeline were applied to the three models, with the only difference being the dataset download step. There are 500 pictures for each different type of cardboard box model training. Those dataset images were recognised and labelled using the Roboflow platform (*Roboflow*, n.d.). Each box-sized dataset was divided into 80% training images and 20% validation images. The validation split was used to select the best.pt checkpoint and to assess bounding-box detection; it was not an independent test set. The small variant of YOLOv8 was initialised using the pretrained yolov8s.pt checkpoint and trained separately on each of the three box-size datasets. The dataset paths and Layer 1–Layer 3 class definitions were specified in the corresponding data.YAML files. Each model was trained for a maximum of 100 epochs using an input resolution of 640 pixels, a batch size of 16, automatic optimiser selection (optimizer=auto), an initial learning-rate parameter (lr0) of 0.01, a weight decay of 0.0005, and a random seed of 0. Training was capped at 100 epochs. Although the recorded training configuration set patience=100, this setting did not provide an effective early stopping condition within the 100-epoch schedule. The checkpoint with the best validation performance, best.pt, was retained for application testing.

During application testing, the best. The PT checkpoint was used to process input images with a nominal image size of 640 pixels. No custom confidence threshold was specified in the prediction configuration; therefore, the default Ultralytics version 8.3.39 confidence threshold of 0.25 was applied. The non-maximum-suppression IoU threshold was 0.70, and the maximum number of detections per image was 300. Predictions were generated for all three-layer classes without class filtering.

Roboflow was used to annotate and export the three datasets in YOLOv8 format. No additional offline augmentation was applied during the Roboflow export process. Because no custom augmentation parameters were specified in the model-training command, the default online augmentation pipeline of the installed Ultralytics YOLOv8 version was used. This pipeline applied transformations such as HSV colour variation, translation, scaling, horizontal flipping, and mosaic augmentation to the training images.

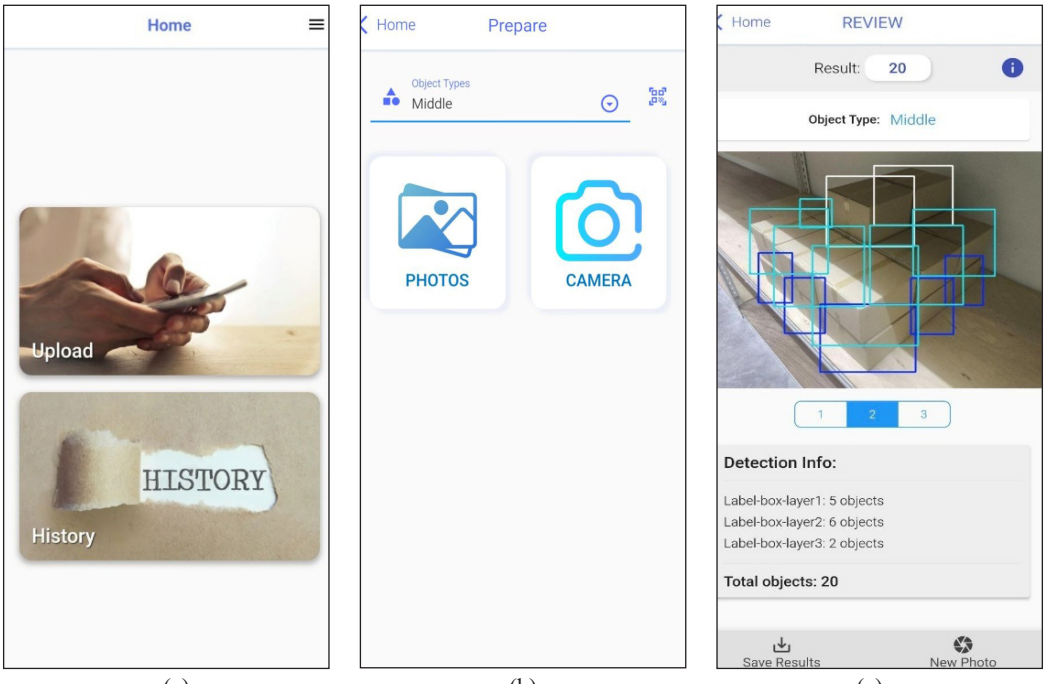
The use of three separate models was an experimental design choice intended to isolate performance across the predefined box dimensions. It is not proposed as a scalable production strategy for arbitrary SKUs. A future implementation could train a unified multi-class model on a substantially larger dataset that includes multiple packaging types, dimensions, and warehouse environments.

Implementation of Application Interface

The home page provided two operational options: Take Photo and View History. The photo option allows the user to compute the number of boxes by capturing images, while the view history option provides a history of the detection process the user has made. The template selection interface provides three options corresponding to the three different box sizes to be detected. After selecting either template, the system proceeded to the image capture page. After the user captures images of the stock area, the user proceeds to the next interface, the result page, by clicking the proceed button. The result interface displayed the detection outcome and the computed box count. The number of boxes detected is displayed at the top of the screen, followed by the captured image. The user can click the save button to store the result in the database. The system interfaces are shown in Figure 6.

Implementation of Database

The box detection and counting process involves creating a database specifically to store important information about users, images, and detection outcomes, as shown in Figure 7. Below is a breakdown of the information provided, including the primary key and the foreign key that will be stored in the database.



(a) (b) (c)

Figure 6. Application interface: (a) Home page, (b) Template selection page, (c) Result page

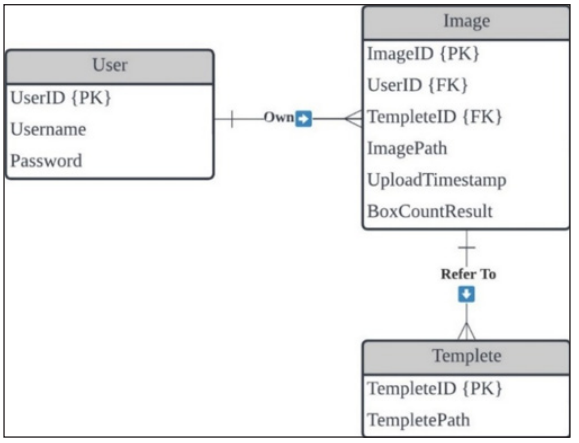


Figure 7. Database diagram

The User table was created to store crucial user details, such as UserID, Username, and Password. UserID serves as a unique identifier for each user within the system and database. The Image table stores information about the images uploaded by users and the results processed by the system. The primary key for this table was ImageID as a unique identifier for each detection. There were two foreign keys in this table: UserID and TemplateID, which are references from the User table and the Template table, respectively. These were the linking of images to their uploaders and the indication of the template used to process the images. The Template table was used to store details about the various templates accessible for box detection. The primary key of this table was TemplateID, which serves as a unique identifier assigned to each template. TemplatePath was the location of the file where the image processing template was stored.

RESULT AND DISCUSSIONS

The experiment was conducted in a warehouse stock room with goods on the racks and on the floor, creating a visually complex background for object detection. An empty shelf was used to arrange the test boxes. During testing, the operator stood approximately 0.5–1.0 m from the shelf and captured images at an angle of approximately 45°, as shown in Figure 8. Maintaining the same position throughout the experiment ensured consistent testing conditions and reflected the typical posture warehouse workers use during stock checking. All available room lights were switched on to provide sufficient illumination for the smartphone camera.

This experiment had only a single tester perform 250 trials for each model trained, rather than distributing testing across multiple individuals. This decision was rooted in the unique nature of this research. This procedure aligns with the nature of the research, in which performance measures are influenced by patterns and the sequential placement of

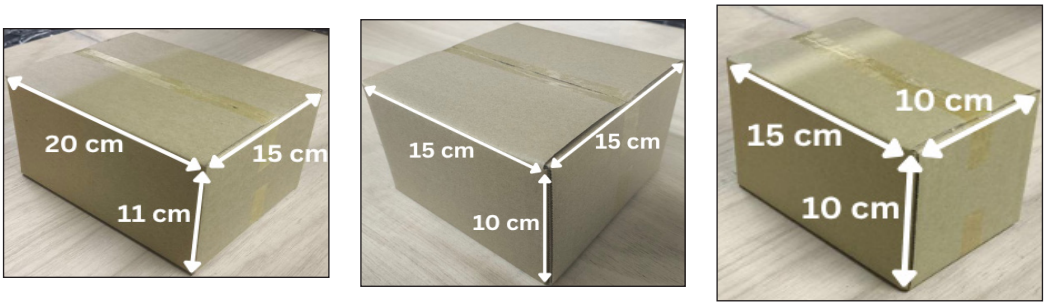


Figure 8. Experimental environment setup

boxes on the rack. The use of a single tester not only provided a deeper understanding of the earlier boxes' arrangements but also enabled more purposeful adjustments to prevent unwanted pattern repetition and further reduce the risk of compromising the authenticity of the experimental data. The resulting images were grouped into three categories: true positives (cases in which boxes were identified with accurate layer detection), false positives (cases in which boxes were found but with incorrect layer categorisations), and false negatives (cases in which boxes were missed by the model). True negative cases are considered irrelevant in this experimental setup due to the nature of the detection problem.

Experiment Record

The experiment required an Android phone with the proposed research prototype installed and 27 cardboard boxes in three sizes: 20 × 15 × 11 cm, 15 × 15 × 10 cm, and 15 × 10 × 10 cm, as shown in Figure 9. Testing was conducted in an operational warehouse stockroom to evaluate the proposed system's performance in a realistic environment.



(a) (b) (c)

Figure 9. Cardboard box type: (a) Big size, (b) Middle size, (c) Small size

Bounding-Box Detection Validation

The selected checkpoint for each box-size model was assessed on its labelled validation split. Table 2 reports the mean average precision at an intersection-over-union threshold of 0.50 (mAP@0.5) and averaged across thresholds from 0.50 to 0.95 (mAP@0.5:0.95). These metrics evaluate predicted bounding boxes against annotated boxes. They are reported separately from the subsequent 250 application-level counting trials for each model.

Two evaluations were conducted. Bounding-box detection was assessed on labelled validation images using mAP@0.5 and mAP@0.5:0.95, as reported in Table 2. The integrated mobile application was assessed separately through 250 controlled counting trials for each box-size model. Figures 10 to 14 report the recorded application-trial outcomes and their derived measures. These trial results illustrate how the complete counting workflow is carried out. They do not measure bounding-box localisation and are not a substitute for detector-level mAP results.

	A	B	C	D	E
1	Box Types	Iteration	Actual number	Number of boxes detected	Accuracy Status (✓/✗)
2	Middle	1	13	14	×
3		2	13	14	×
4		3	15	15	×
5		4	22	21	×
6		5	23	21	×
7		6	26	27 (Layer Error)	×
8		7	26	27 (Layer Error)	×
9		8			
10		9			
11		10			
12	Big	1	4	19	×
13		2	4	11	×
14		3	5	11	×
15		4	5	11	×
16		5	15	14	×
17		6	17	16	×
18		7	22	22 (Layer Error)	×
19		8	22	21	×
20		9	22	22 (Layer Error)	×
21		10	23	24	×
22		11	23	21	×
23		12	23	23 (Layer Error)	×
24		13	24	24 (Layer Error)	×
25		14	25	25 (Layer Error)	×
26		15	25	25 (Layer Error)	×
27		16	25	25 (Layer Error)	×
28		17	26	26 (Layer Error)	×
29	Small	1	11	12	×
30		2	14	15	×
31		3	15	16	×
32		4	21	20	×
33		5	21	20	×
34		6	25	25 (Layer Error)	×
35		7	22	22 (Layer Error)	×
36		8	22	22 (Layer Error)	×
37		9	22	22 (Layer Error)	×
38		10	23	24	×
39		11	23	23 (Layer Error)	×
40		12	25	25 (Layer Error)	×
41		13	24	24 (Layer Error)	×
42		14	22	21	×

Figure 10. Fail record

Figure 10 presents the table used to record unsuccessful trials during the experiment. Because 250 trials were conducted for each model, this table facilitated the identification and subsequent analysis of counting and layer-classification errors.

The recorded failures were primarily associated with incorrect layer classification, over- or undercounting due to viewing angle, and violations of the assumed gap-free stacking arrangement. Low illumination may further reduce the visibility of box boundaries.

Table 2
Bounding Box Detection Results on Validation Splits

Box-size model	Variant	mAP@0.5	mAP@0.5:0.95
Small	YOLOv8s	0.99500	0.99273
Middle	YOLOv8s	0.99500	0.99159
Big	YOLOv8s	0.99500	0.99345

Result of Middle Box Type Model Experiment

The Middle box type model showed significant performance across the 250 test iterations. As shown in Figure 11, it achieved 243 true positives, 7 false positives, and no false negatives. This distribution indicated a strong detection capacity, but with the presence of occasional errors in its classification.

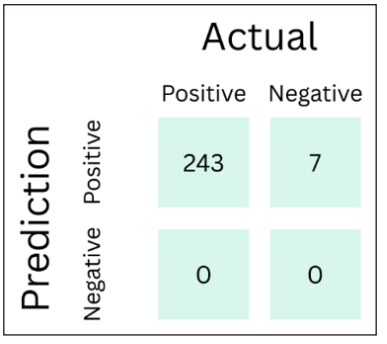


Figure 11. Middle type results classification

The performance metrics for the middle box model are calculated as follows:

$$\text{Accuracy} = \frac{243}{250} \times 100\% = 97.2\% \quad [1]$$

$$\text{Precision} = \frac{TP}{TP + FP} \times 100\% = \frac{243}{250} \times 100\% = 97.2\% \quad [2]$$

$$\text{Recall} = \frac{TP}{TP + FN} \times 100\% = \frac{243}{243} \times 100\% = 100\% \quad [3]$$

$$\text{F1 Score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% = 2 \times \frac{0.972}{1.972} \times 100\% = 98.6\% \quad [4]$$

The Middle box model exhibits excellent overall performance with errorless recall. These performance measurement parameters were near to 100%. This proved that the middle-box-type model could label all boxes in the test cases.

Result of Big Box Type Model Experiment

The Big box model showed significant performance across the 250 test iterations. It achieved 233 true positives, 14 false positives, and 3 false negatives, as shown in Figure 12. The distribution reflects good overall performance with periods of both misclassification and missed instances.

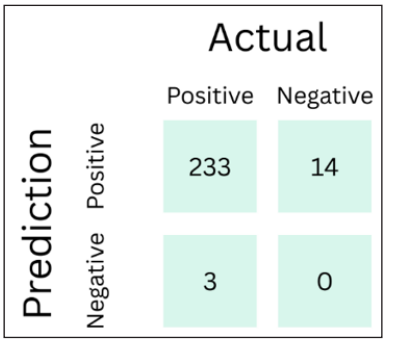


Figure 12. Big type results classification

The performance metrics for the big box model are calculated as follows:

$$\text{Accuracy} = \frac{233}{250} \times 100\% = 93.2\% \quad [5]$$

$$\text{Precision} = \frac{TP}{TP + FP} \times 100\% = \frac{233}{247} \times 100\% = 94.3\% \quad [6]$$

$$\text{Recall} = \frac{TP}{TP + FN} \times 100\% = \frac{233}{236} \times 100\% = 98.7\% \quad [7]$$

$$\text{F1 Score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% = 2 \times \frac{0.931}{1.930} \times 100\% = 96.5\% \quad [8]$$

The big box model achieves optimal performance, but it performs marginally worse than the box box model. The performance measurement parameters near 100% proved that this big box model could label all boxes in the test cases.

Result of Small Box Type Model Experiment

The Small box model showed high performance across the 250 test iterations. As shown in Figure 13, it achieved 236 true positives, 14 false positives, and no false negatives. This trend indicates high detection quality, even with some false positives.

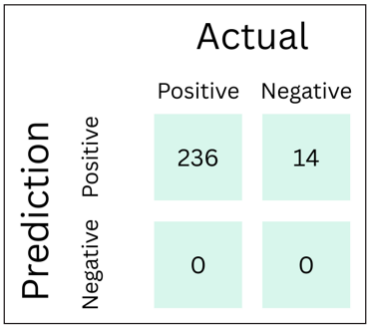


Figure 13. Small type results classification

The performance metrics for the small box model are calculated as follows:

$$\text{Accuracy} = \frac{236}{250} \times 100\% = 94.4\% \quad [9]$$

$$\text{Precision} = \frac{TP}{TP + FP} \times 100\% = \frac{236}{250} \times 100\% = 94.4\% \quad [10]$$

$$\text{Recall} = \frac{TP}{TP + FN} \times 100\% = \frac{236}{236} \times 100\% = 100\% \quad [11]$$

$$\text{F1 Score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% = 2 \times \frac{0.944}{1.944} \times 100\% = 97.1\% \quad [12]$$

The small box-type model exhibits excellent overall performance, with optimal recall. These performance measurement parameters were near to 100%. This proved that the small box-type model can label all boxes in the test cases, and that there are opportunities to improve classification accuracy.

DISCUSSION

The analysis of the experimental results reveals significant performance trends across the three model types, as shown in Figure 14. Table 2 presents detector-level bounding-

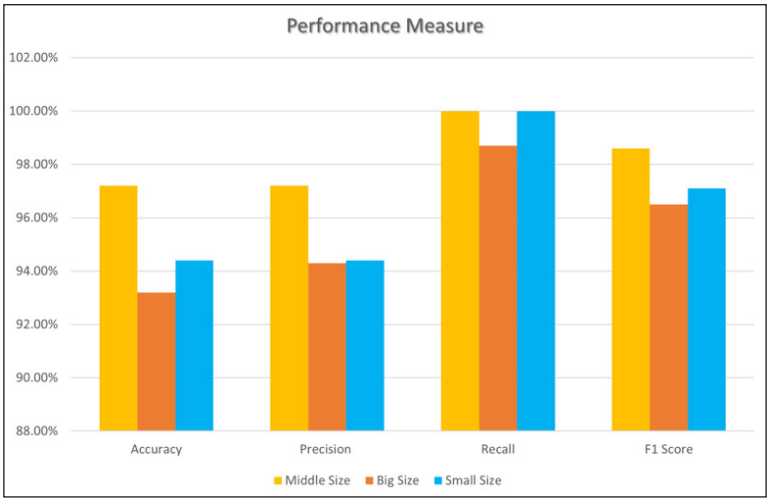


Figure 14. Model performance measure

box performance, whereas Figure 14 presents application-level trial outcomes; the two evaluations measure different aspects of the system. The highest-performing model was the Middle type, with 97.2% accuracy and an F1 score of 98.6%. It maintained an excellent balance between precision and recall. The Small-type model followed closely, with an accuracy of 94.4% and an F1 score of 97.1%. But it faces slightly greater difficulty with classification than the Middle type. This model correctly picked up most of the positive instances but missed a few, as evidenced by the recall being less than 100%. The Big type of model, with its relatively satisfactory result of 93.2% in terms of accuracy, along with an F1 of 96.3%. That is, the model accurately identifies all actual positive cases without omitting any and is highly accurate in its positive predictions.

Study Limitations and Future Work

This system is a proof of concept evaluated under a constrained 3 × 3 × 3 stacking arrangement. Its counting rule assumes that lower layers are fully occupied and boxes are placed in contiguous rows. Consequently, the present method cannot reliably count stacks with vacant positions, partially filled lower layers or irregular arrangements. This fixed-layout assumption is a key barrier to deployment in general warehouse settings. Future work must revise the counting logic to accommodate such arrangements and validate it using more diverse warehouse scenes.

Lack of Multi-Operator Usability Testing

One operator performed 250 trials for each of the three box-size models. Using one operator reduced variation in image-capture conditions during this initial system test, but the results

do not establish whether other warehouse workers would achieve comparable performance. Future work should assess multiple operators with different experience levels, capture styles, and report variation between operators.

CONCLUSION

The proposed solution successfully identified and enumerated the number of boxes. It has high potential to solve the major challenges faced by warehouse and stock management systems. The achievements indicated that by employing advanced computer vision methods and machine learning algorithms, the process of warehouse stock enumeration can be automated and simplified. There are some limitations that have been recognised as potentially influencing the system's performance and usability in certain settings. The system is currently confined to recognising three-dimensional square cardboard boxes. The application's detection accuracy is heavily dependent on environmental conditions, specifically lighting and image-taking angles. Extremely low-light conditions might immediately reduce detection performance. There must also be certain prerequisites to get the best results. The layer-based sorting algorithm also takes several important assumptions that must be maintained for accurate counting. The algorithm also requires the boxes to be positioned without gaps and arranged in touching rows rather than discrete locations. There are multiple areas where the system's effectiveness can be improved and its limitations overcome. The system's current tri-layer detection system can be expanded to encompass five or six layers. This aims to support a wider range of stacking scenarios commonly found in today's warehouses. The accuracy of detection results can be significantly improved by collecting and integrating large training sets across varied lighting conditions, including dim lighting, artificial light common in warehouses, natural light, and combinations thereof. The additional dataset should also include images that represent complex backgrounds with varied warehouse equipment, shelving units, and other inventory items. Future improvements should focus on models skilled at detecting various types of stock items. For example, cylindrical containers, plastic containers, metal objects, fabric containers, and unusual items are commonly found in warehouses. The development of these multi-format detection capabilities would transform the system from a box-counting device to a comprehensive inventory management system. Future work should remove the fixed stacking constraint by incorporating training examples with gaps, partially occupied layers, irregular arrangements, and larger stacking configurations.

ACKNOWLEDGMENT

This research is supported by Media and Game Innovation Centre of Excellence (MaGICX), Institute of Human Centred Engineering (iHumEn), Universiti Teknologi Malaysia (UTM) and Faculty of Computing, UTM.

REFERENCES

Belyh, A. (2022, December 6). *What is inventory management process? Flow, challenges, implementation*. FounderJar. https://www.founderjar.com/inventory-management-process-flow/

Bradski, G., & Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media.

Büyüközkan, G., & Göçer, F. (2018). Digital supply chain: Literature review and a proposed framework for future research. *Computers in Industry, 97*, 157-177. https://doi.org/10.1016/j.compind.2018.02.010

Chen, Y., & Patel, K. (2024). The impact of automated sorting on warehouse logistics and inventory accuracy. *International Journal of Supply Chain Automation, 15*(2), 200-215.

Collins, G. (2026, March 1). *Walmart's AI automation push sets new retail benchmark*. AI Magazine. https://aimagazine.com/news/walmarts-ai-automation-push-sets-new-retail-benchmark

CountThings. (n.d.). *CountThings from photos*. https://countthings.com/

DHL Supply Chain. (n.d.). *Improving field inventory management for a medical devices provider*. https://www.dhl.com/content/dam/dhl/global/csi/documents/pdf/glo-csi-inventory-optimization-for-global-medical-devices-customer-2023.pdf

Fildes, R., Goodwin, P., Lawrence, M., & Nikolopoulos, K. (2009). Effective forecasting and judgmental adjustments: An empirical evaluation and strategies for improvement in supply-chain planning. *International Journal of Forecasting, 25*(1), 3-23. https://doi.org/10.1016/j.ijforecast.2008.11.010

Focal Systems. (n.d.). *Focal Systems*. https://focal.systems/

Gama, L., et al. (2025). Transforming enterprise structures through digitalization and advanced machine learning. *Technology in Society, 72*, 101-115.

Gao, F., He, Y., Wang, J., Hussain, A., & Zhou, H. (2020). Anchor-free convolutional network with dense attention feature aggregation for ship detection in SAR images. *Remote Sensing, 12*(16), Article 2619. https://doi.org/10.3390/rs12162619

Javaid, M., Haleem, A., Singh, R. P., & Suman, R. (2022). Artificial intelligence applications for industry 4.0: A literature-based study. *Journal of Industrial Integration and Management, 7*(1), 83-111. https://doi.org/10.1142/s2424862221300040

Jocher, G., Chaurasia, A., & Qiu, J. (2023). *Ultralytics YOLOv8* (Version 8.0.0) [Computer software]. Ultralytics. https://github.com/ultralytics/ultralytics/tree/v8.0.0

Koh, L. Y., & Yuen, K. F. (2022). Emerging competencies for logistics professionals in the digital era: A literature review. *Frontiers in Psychology, 13*, Article 965748. https://doi.org/10.3389/fpsyg.2022.965748

Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., ong, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., & Wei, X. (2022). *YOLOv6: A single-stage object detection framework for industrial applications* [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2209.02976

Liboni, L. B., Cezarino, L. O., Jabbour, C. J. C., Oliveira, B. G., & Stefanelli, N. O. (2019). Smart industry and the pathways to HRM 4.0: Implications for SCM. *Supply Chain Management: An International Journal, 24*(1), 124-146. https://doi.org/10.1108/scm-03-2018-0150

OpenCV. (n.d.-a). *Feature matching with FLANN*. https://docs.opencv.org/4.10.0/d5/d6f/tutorial_feature_flann_matcher.html

OpenCV. (n.d.-b). *Harris corner detection*. https://docs.opencv.org/4.10.0/dc/d0d/tutorial_py_features_harris.html

Radanliev, P., De Roure, D., Nicolescu, R., Huth, M., & Santos, O. (2021). Artificial intelligence and the Internet of Things in industry 4.0. *CCF Transactions on Pervasive Computing and Interaction, 3*(3), 329-338. https://doi.org/10.1007/s42486-021-00057-3

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 779-788). IEEE. https://doi.org/10.1109/CVPR.2016.91

Roboflow. (n.d.). *Roboflow*. https://roboflow.com/

Sengupta, S., & Dreyer, H. (2023). Realizing zero-waste value chains through digital twin-driven S&OP: A case of grocery retail. *Computers in Industry, 148*, Article 103890. https://doi.org/10.1016/j.compind.2023.103890

Terven, J., & Cordova-Esparza, D. (2023). *A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond* [Preprint]. arXiv. https://arxiv.org/abs/2304.00501v1

Toneguzzi, M. (2022, September 21). *Walmart Canada introduces AI tech to avoid out-of-stock in stores [interview]*. Retail Insider. https://retail-insider.com/retail-insider/2022/09/walmart-canada-introduces-ai-tech-to-avoid-out-of-stock-in-stores-interview/

Verhoef, P. C., Broekhuizen, T., Bart, Y., Bhattacharya, A., Dong, J. Q., Fabian, N., & Haenlein, M. (2021). Digital transformation: A multidisciplinary reflection and research agenda. *Journal of Business Research, 122*, 889-901. https://doi.org/10.1016/j.jbusres.2019.09.022

Vijayakumar, A., & Vairavasundaram, S. (2024). YOLO-based object detection models: A review and its applications. *Multimedia Tools and Applications, 83*(35), 83535-83574. https://doi.org/10.1007/s11042-024-18872-y

Villegas-Ch, W., Maldonado Navarro, A., & Sanchez-Viteri, S. (2024). Optimization of inventory management through computer vision and machine learning technologies. *Intelligent Systems with Applications, 24*, Article 200438. https://doi.org/10.1016/j.iswa.2024.200438

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 7464-7475). IEEE. https://doi.org/10.1109/CVPR52729.2023.00721

Wassel, B. (2022, September 20). *Walmart Canada rolls out AI-powered out-of-stock detection to all locations*. Retail TouchPoints. https://www.retailtouchpoints.com/news/walmart-canada-rolls-out-ai-powered-out-of-stock-detection-to-all-locations/117730/

Zhao, Z.-Q., Zheng, P., Xu, S.-T., & Wu, X. (2019). Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems, 30*(11), 3212-3232. https://doi.org/10.1109/tnnls.2018.2876865