









Inventory automation using computer vision models and techniques

Automatización de inventario usando modelos y técnicas de visión por computadora

Davalos-Nava, Gilberto*^a & Reyes-Nava, Adriana^b

^a  TecNM: Tecnológico de Estudios Superiores de Jocotitlán •  LMO-8932-2024 •  0009-0002-1490-9592 •  2073713

^b  TecNM: Tecnológico de Estudios Superiores de Jocotitlán •  LNP-9046-2024 •  0000-0002-4440-909X •  786639

CONAHCYT classification: DOI: <https://doi.org/10.35429/H.2024.12.13.27>

Area: Engineering
Field: Technological sciences
Discipline: Computer technology
Subdiscipline: Real-time systems

Key Handbooks

The main contributions of this research to science and technology lie in the application of advanced computer vision tools and the development of modern web solutions. These technologies are integrated to create systems capable of performing automated sensing, tracking and estimation tasks in real environments, contributing to the advancement of automation and digitization of industrial and commercial processes. In addition, the development of an efficient web platform allows the visualization and management of data in real time, facilitating the remote monitoring and control of complex systems. To apply this knowledge to the generation of universal knowledge, it is essential to understand the fundamental principles of computer vision and the use of modern web development techniques. These key aspects are the basis for the design of intelligent systems that can process and analyze large volumes of visual data and present them in an accessible way through interactive interfaces. The authors of the paper come from technological universities, which highlights the academic collaboration between institutions dedicated to technological education and innovation. Key words used throughout the development of the research include “computer vision” and “modern web development”.











Citation: Davalos-Nava, Gilberto & Reyes-Nava, Adriana. 2024. Inventory automation using computer vision models and techniques. 13-27. ECORFAN.

* ✉ [\[2020150480247@tesjo.edu.mx\]](mailto:2020150480247@tesjo.edu.mx)

Handbook shelf URL: <https://www.ecorfan.org/handbooks.php>

Abstract











Inventory taking is essential for managing production or warehouse operations, as it allows for precise control of the quantity, location, and status of products. However, this task can be demanding, requiring constant staff involvement and breaks in activities. Computer vision technology facilitates automatic inventory monitoring by detecting and recognizing objects, allowing for accurate quantification of boxes stacked on a platform using distance estimation algorithms. This project is divided into two phases: the development of software for inventory monitoring and counting, and the creation of an autonomous prototype for monitoring in production environments. Currently, work is focused on the first phase, which includes training the detection model, setting up an inference server, and integrating it into a web interface for inventory consumption and monitoring.

Inventory automation using computer vision models and techniques		
Objectives	Methodology	Contribution
<p>Implement a computer vision model for the automatic detection, recognition and quantity estimation of stored products. Create a robust software architecture that integrates the computer vision system with a server that allows the efficient collection, processing and storage of inventory data.</p> 	<p>Scrum is an agile methodology that facilitates the management of complex projects in software development.</p> <div>Planning</div> <div>Implement</div> <div>Review</div> <div>Feedback</div> <div>Deploy</div>	<p>In Warehouse enviroment:</p> <div>+ Productivity</div> <div>Reduce Costs</div> <div>Maximize efficiency</div> <div>+ Audits Accuracy</div>
		<div>Results</div> <div>Quantity estimation.</div> <div>Functional web software.</div>

Process monitoring, Computer vision, Modern web development

Resumen

La realización de inventarios es esencial para gestionar operaciones de producción o almacén, ya que permite un control preciso de la cantidad, ubicación y estado de productos. Sin embargo, esta tarea puede ser demandante, requiriendo participación constante del personal y pausas en las actividades. La tecnología de visión por computadora facilita el monitoreo automático del inventario mediante la detección y reconocimiento de objetos, permitiendo cuantificar con precisión cajas apiladas sobre una plataforma usando algoritmos de estimación de distancia. Este proyecto se divide en dos fases: el desarrollo del software para supervisión y conteo de inventario, y la creación de un prototipo autónomo para el monitoreo en entornos de producción. Actualmente, el trabajo se centra en la primera fase, que incluye el entrenamiento del modelo de detección, la configuración de un servidor de inferencia y la integración de este en una interfaz web para el consumo y monitoreo del inventario.

Automatización de inventario usando modelos y técnicas de visión por computadora		
Objetivos	Metodologia	Contribucion
<p>Implementar un modelo de visión por computadora para la detección, reconocimiento y estimación de cantidad automática de productos almacenados. Crear una arquitectura de software robusta que integre el sistema de visión por computadora con un servidor que permita la recolección, procesamiento y almacenamiento eficiente de datos de inventario.</p> 	<p>Scrum es una metodología ágil que facilita la gestión de proyectos complejos en el desarrollo de software.</p> <div>Planeacion</div> <div>Implementar</div> <div>Revisar</div> <div>Retroalimentar</div> <div>Desplegar</div>	<p>En un entorno de almacen:</p> <div>+ Productividad</div> <div>Reducir Costos</div> <div>Maximizar Eficiencia</div> <div>+ Precision de auditorias</div> <div>Results</div> <div>Estimacion de cantidad. Software Web Funcional.</div>

Monitoreo de procesos, Visión por Computadora, Desarrollo web moderno

Introduction

In production environments, process automation has always been a priority, not only to minimize human effort, but also to speed up processes and reduce operating costs. Recent advances in artificial intelligence (AI) have radically transformed activity monitoring, allowing the implementation of models that process images in real time. The combination of these technologies has opened up new opportunities to optimize data collection and improve efficiency in various industries.

One of the most promising fields within this trend is computer vision, a branch of AI that allows machines to interpret and understand the content of real-world images. However, developing efficient models in this area is often a significant challenge due to the need for large volumes of training data and the associated computational cost. Computer vision relies on algorithms that repeatedly analyze data to identify patterns and recognize objects or differences within images (IBM, 2021). Despite these challenges, advancements in tools and platforms have made it easier to access, allowing companies and developers to take advantage of its potential without requiring large investments in computing resources.

This project focuses on the implementation of an efficient system to automate inventory taking in production environments. In a warehouse, finished products can be organized in a variety of ways, which represents an additional challenge when monitoring and managing inventories. Two key factors directly influence this task: the type of product, given the diversity of items in a warehouse; and the arrangement of these, since they are stored in three-dimensional structures, such as platforms or shelves.

A practical example of these challenges can be seen in an industrial plant that manages two main warehouses: one for raw materials and another for materials. The inventory process in these warehouses takes approximately one work week, during which no inputs or outputs of materials can be made. This not only consumes the time of the personnel in charge, but also requires the support of other teams, which affects the overall operation of the plant.

Another example occurs in a plant where finished products are stored in boxes stacked on platforms. Staff must perform periodic counts of the material on each pallet, which, although necessary, consumes a considerable amount of time and human resources.

Optimizing inventory taking could not only be achieved through automated pallet monitoring, but also through more precise control of product inputs and outputs in the warehouse. For this task, technological solutions already exist, although most are based on software with local databases. This is where modern web development comes into play, offering the possibility of centralizing information through remotely accessible servers, improving real-time inventory management and allowing users to access updated data from any location.

Modern web development is characterized by asynchronous processing, which allows for multiple tasks to be performed simultaneously without blocking processes, improving efficiency even in inventory management systems with large volumes of data. Technologies such as Server-Side Rendering (SSR) and Client-Side Rendering (CSR) optimize both performance and user experience; SSR reduces loading times and improves SEO, while CSR offers greater interactivity by processing data in the browser. In addition, CI/CD (Continuous Integration/Continuous Deployment) tools automate code integration and deployment, ensuring fast and secure updates. Technologies such as Docker create isolated and scalable environments, facilitating application deployment. In terms of communication between the frontend and backend, REST APIs and GraphQL allow for efficient resource management, with GraphQL being especially useful for requesting only the necessary data in applications with multiple dependencies.

A prominent example where computer vision and modern web development work together is Intenseye, a platform designed for monitoring activities in industrial environments with the aim of preventing accidents. Using advanced object detection, pose estimation and segmentation techniques, Intenseye continuously monitors operations to identify potential risks. A success story is its application at Coats, a company that was facing serious road safety issues at its industrial facilities, especially in India, a country with a high rate of traffic accidents. Thanks to the implementation of Intenseye's software, which integrates with existing CCTV infrastructure and uses artificial intelligence to identify risks such as speeding, Coats was able to apply immediate corrective measures, such as driver training.

In just one week, they managed to reduce speed violations by 20%, reaching a 50% decrease with the continuous enforcement of safety regulations ([Intenseye, 2019](#)).

Another success story was in 2022, when a major global food manufacturer experienced a 61% reduction in hazard detection within 9 months. This meat producer, with over 78,000 employees and exposed to risks from heavy machinery and dangerous instruments, already had robust safety protocols in place, however, the incorporation of Intenseye's technology significantly boosted its ability to identify hazards in multiple areas simultaneously, improving speed of response and risk mitigation ([Intenseye, 2022](#)).

On the other hand, the autonomous robot Tally, used in stores and supermarkets for inventory management, is another example of how computer vision is combined with modern web technologies. Tally uses CV and technologies such as RFID/Digimarc to navigate shelves and generate automatic reports on product availability. Its advanced detection system enables it to perform inventory audits three times a day with 99% accuracy, far surpassing the efficiency of manual processes, which take a week and achieve only 65% accuracy. In addition, Tally uses hybrid cloud and edge computing, allowing it to process and transmit data in real time with low bandwidth consumption, facilitating the integration of information with existing IT systems ([Simbe, 2024](#)).

This work is organized in five sections:

- Section 1. Introduction, in this section we present the rationale of the project, the most relevant topics and a review of background in similar applications.
- Section 2. Theoretical Foundation, addresses the key concepts of computer vision, supervised learning and its workflow. In addition, essential notions about modern web development are included.
- Section 3. Development, details the steps implemented for object detection and distance estimation, along with the description of the architecture and techniques used in the development of the web application.
- Section 4. Results, presents the results obtained using the classification and estimation techniques, highlighting the advantages of the web architecture used compared to traditional methods.
- Section 5. Conclusions, discusses possible improvements, feedback and future work to optimize the application.

2. Theoretical Foundation

2.1. Machine Learning and Computer Vision

In recent years, artificial intelligence (AI) has experienced a remarkable boom. Although the concept of AI is broad, it refers to applications that mimic human intelligence. Not all AI-based solutions use machine learning (ML), but AI, in general, seeks to perform complex tasks efficiently. There are several methods within AI, such as those based on rules, neural networks, and computer vision, among others. On the other hand, Machine Learning is a specific methodology within AI. All ML solutions are AI solutions, but ML focuses on identifying patterns in large data sets to solve specific problems. In this approach, humans manually select and extract features from raw data and assign weights to train the model. ([AWS, 2023](#))

Within the branch of machine learning. There are three main types of learning:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

2.2. Supervised Learning

Supervised learning is a group of algorithms that require a data set composed of example input and output pairs. Each pair consists of a data sample used to make predictions and an expected outcome known as a label. The term "supervised" comes from the fact that a human supervisor must assign these labels to the data.

During the training process, samples are iteratively fed to the model. For each sample, the model uses the current state of its parameters and generates a prediction. This prediction is compared to the corresponding label, and the difference between the two is called the error. The error acts as feedback, telling the model what went wrong and how it should adjust its parameters to decrease this error in future predictions. In this way, the model updates its values according to the algorithm on which it was designed as seen in Figure 1. (Medium, 2018)

Box 1

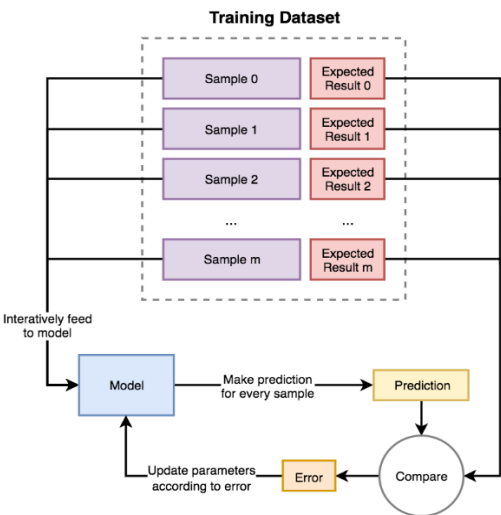


Figure 1
Supervised learning flow

Source: obtained from (Medium, 2018)

2.3. Modern Web Development

In recent years, web development has seen great technological advances, which have allowed for improved development and end-user experience, as well as significantly reduced delivery times. One of the main trends in modern web development is client-server architecture.

Web development has seen significant advancements, with client-server architecture becoming a cornerstone for efficient application delivery. In this model, tasks are divided between clients (devices requesting services) and servers (providers of data or processing). The client sends a request over the network, and the server processes it, returning the necessary data. This architecture allows for a smooth user experience, especially through techniques like asynchronous programming, which enables the system to handle long-running tasks without disrupting other processes.

Rendering in web applications can be approached in two main ways: Client-Side Rendering (CSR) and Server-Side Rendering (SSR). CSR involves generating the user interface directly in the client browser using JavaScript. It allows for dynamic, interactive web applications, particularly useful for single-page applications (SPAs) where content is constantly updated. However, CSR may lead to slower initial loading times and challenges with search engine optimization (SEO), as the HTML delivered to the client is minimal and requires JavaScript to fully render.

On the other hand, SSR generates complete HTML on the server before sending it to the client. This approach ensures faster loading times and improved SEO, as the browser receives a fully rendered page. It is often used in applications where quick access and SEO are crucial, such as e-commerce sites. However, SSR can lead to a heavier load on the server and may not be as interactive as CSR, which excels in dynamic component handling.

A balanced solution often involves combining CSR and SSR. Using SSR for initial rendering ensures faster page loads and better SEO, while CSR can enhance interactivity and dynamic content updates after the page has loaded. This hybrid approach provides a seamless user experience, merging the benefits of both methods: quick load times, search engine optimization, and a dynamic, engaging interface.

2.4. Agile Scrum Methodology

The Agile Scrum methodology is widely used to manage complex projects, especially in software development, by encouraging a collaborative approach based on adaptability and continuous improvement. Scrum organizes work into short iterations called "sprints," lasting one to four weeks, where the team delivers tangible project increments. Regular meetings ensure alignment and progress. This framework allows teams to quickly adapt to changing requirements, prioritizing flexibility and efficiency in delivering high-quality products (NimbleWork, 2022).

The Scrum methodology is developed through a series of events in each sprint, as described by (NimbleWork, 2022):

- Sprint Planning: At the beginning of each Sprint, a meeting is held to define the Sprint goal and select the Product Backlog items to be worked on.
- Daily Scrum: Short daily meeting in which the team synchronizes its activities, shares progress, and discusses possible obstacles.
- Sprint Review: After the end of the Sprint, an event is held in which the completed work is presented and feedback is received from stakeholders.
- Sprint Retrospective: Reflective meeting that takes place after the review, in which the team evaluates its processes and proposes improvements for future Sprints.
- Backlog Refinement: Continuous process of reviewing and adjusting the Product Backlog to ensure that the items are ready for the next Sprints.

3. Development

This section details the steps taken to develop the software, including analysis of the environment, training of the model, and application of each stage of the Scrum methodology to create the web application.

3.1. Environmental Analysis

The environmental analysis addresses how products are arranged on pallets within warehouses, where boxes are typically stacked on shelves or pallets aligned in aisles. To effectively count items on these platforms, challenges like depth perception and hidden contents need to be considered. Two solutions were proposed:

- Two-point detection: Utilizing sensors (e.g., cameras or LIDAR) placed at elevated positions to scan items from above, allowing for accurate identification of dimensions and detection of irregularities. While this approach provides precise measurements, it complicates the prototype design and reduces adaptability across different scenarios.
- Distance estimation: Calculating object distances using constants like focal length and known item measurements. This approach simplifies the design and reduces hardware requirements but may be less accurate due to reliance on estimations.

Given its simplicity and adaptability, the distance estimation method was chosen. It involves identifying each item for dimensional data and calculating storage capacity by recognizing each platform. Two approaches were considered for this identification:

- Object detection: Using a trained model to detect pallets and boxes, suitable for general item identification but requiring a large labeled dataset due to the visual similarity between items.
- Label-based identification: Implementing labels (text, barcodes, or QR codes) with information about pallet ID and stored items. Although this approach needs an additional reading module, QR codes provide quick and easy identification.

The project opted for the label-based method due to its faster and simpler implementation.

3.2 Model Training

This chapter describes in detail the complete process to prepare and configure an object detection system using YOLOv8, the most recent version of the algorithm developed by Ultralytics. The main objective was to detect boxes and pallets in a warehouse environment, allowing for more accurate and efficient counting of products. To achieve this, the first step was the generation of the dataset, which is one of the most critical stages. A series of images were collected that included variations in lighting, viewing angles, and resolutions, thus ensuring that the model could learn to recognize objects in various real-world conditions. These images were uploaded to the Roboflow platform, which facilitated the task of labeling and preprocessing. Using Roboflow's "Annotate" tool, accurate labels were ensured, paying special attention to not including partially visible objects, as this could lead to erroneous detections and affect the efficiency of the system.

Once the labeling phase was completed, the images were preprocessed. This included two essential steps: auto-orientation, which preserves metadata and ensures that bounding boxes remain correct even if images are resized or rotated, and adjusting the dimensions of all images to 640x640 pixels, which is the size required by YOLOv8. After these adjustments, data augmentation techniques were applied, such as rotation, variations in saturation and brightness, exposure, and blurring. These transformations increased the variety of the dataset without the need to capture more images, which helped the model to generalize better and be more robust to different lighting scenarios and angles in a production environment.

The final dataset was divided into three sets: training (75%), validation (15%), and test (10%). This separation allowed the model's performance to be evaluated on data that was not seen during training, thus providing a more accurate measure of its ability to generalize. During training, the YOLOv8n version was chosen, which balances accuracy and speed. This version features a lower number of parameters compared to larger models, but still offers adequate performance for the needs of the project. Training was carried out in Google Colab using an Nvidia T4 GPU, completing in 16 minutes for 100 epochs with a dataset of approximately 600 images. The results of the training process included performance graphs showing the evolution of the model, as well as the final model saved in PyTorch format.



Figure 2
Supervised learning flow

To evaluate the effectiveness of the model, prediction tests were performed on videos that were not part of the original dataset as seen in Figure 2. This allowed validating the model's ability to detect objects in new situations and ensuring that detections were accurate and consistent. Finally, the trained model was exported in its original PyTorch format, as it is compatible with the current environment running on a computer with ARM architecture. Although YOLOv8 offers the possibility to export to other formats such as TensorFlow, ONNX and TensorFlow.js, the choice to keep PyTorch makes it easier to implement and use in the context of the project, optimizing the detection of boxes and pallets in warehouse environments with efficiency and precision.

3.3 Estimation Software

This chapter describes the development and implementation of the system that uses the trained detection model to make accurate estimates in a real environment. This software is key to managing the model's predictions, allowing tasks such as counting objects and assessing the placement of boxes on platforms.

The software uses the OpenCV library to capture images from various sources (real-time cameras, pre-recorded videos, static images) and process them frame by frame. Each frame is analyzed by the model trained with YOLOv8n, generating bounding boxes, labels of detected objects, and confidence scores as output.

For object tracking throughout the video, Roboflow Supervision is used, and specifically the ByteTrack algorithm, which assigns unique identities to detected objects, allowing their movement to be tracked between frames. ByteTrack associates the most confident detections in the first phase and, in a second phase, uses box similarities (IoU) and appearance to associate lower confidence detections. This ensures accurate tracking even under difficult conditions, such as occlusions.

Once the objects have been identified, a specific area (polygon) is defined for counting. This area is delimited so that only the objects detected within it are considered, filtering out irrelevant detections. In this way, the total number of boxes within the area of interest is counted. Each pallet on the platform carries a QR code that stores key information, such as the ID of the pallet and the boxes it contains. Using the pyzbar library, these QR codes are decoded to obtain detailed data on the count and dimensions of boxes and pallets. QR codes and polygon definition is seen in figure 3.

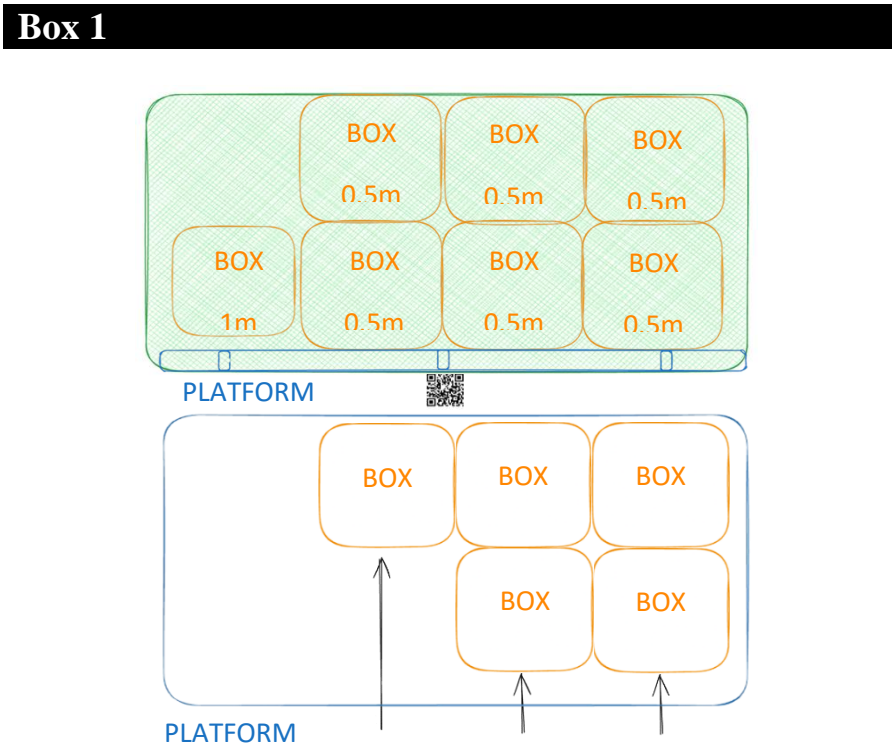


Figure 3
Software Operation

OpenCV tools are used to draw boxes, labels, and confidence scores directly on the processed frames. These modified frames are then converted to a JPG format for easy transmission or storage. In addition, a generator was implemented that produces and returns these frames continuously, allowing real-time integration with other systems or user interfaces.

3.4 Web application

The chapter describes the development of the web application to visualize and manage the estimation and inference system, implemented with a client-server architecture that allows real-time data transmission. An Agile Scrum methodology was used, facilitating continuous feedback and organization in well-defined Sprints.

The database selected was SQLite, known for being light, fast and easy to deploy. SQLAlchemy was used as an ORM (Object-Relational Mapper) to facilitate data connection and manipulation, automating the creation of tables and queries through abstract models. Two main models were defined: item and platform, interrelated to manage the inventory.

The server was developed with FastAPI, a modern and fast web framework, ideal for building APIs. Unlike other frameworks such as Django, FastAPI offers more flexibility in its structure, allowing efficient integration with the estimation module written in Python. The RESTful architecture used facilitates client-server communication using HTTP methods (GET, POST, PUT, DELETE), where each operation is performed through specific endpoints, and responses are sent in JSON format. Additionally, FastAPI includes a Swagger UI interface to interactively test endpoints as seen in figure 4.

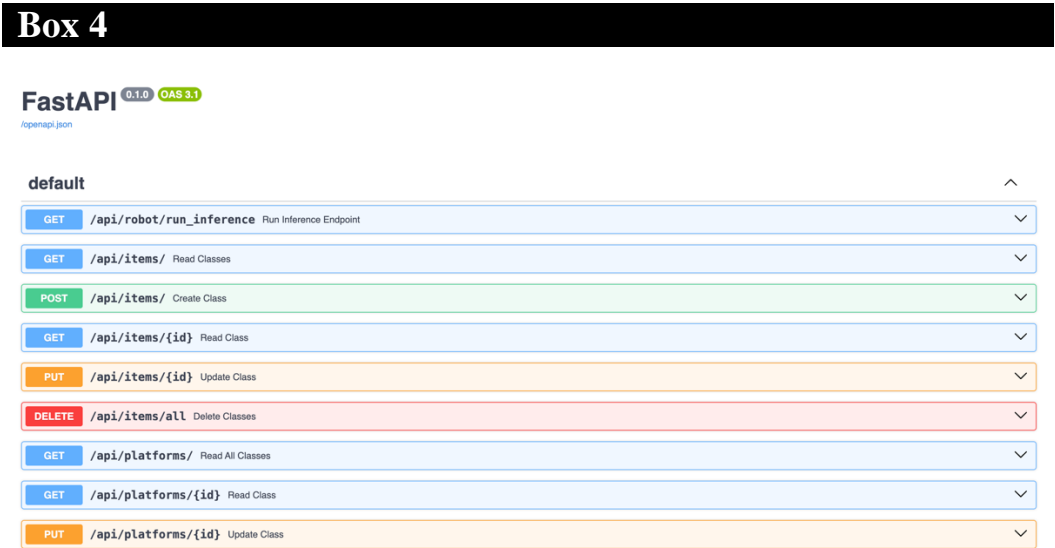


Figure 4
Swagger UI Server

On the client side, Astro was used, a web framework that improves performance by rendering components on the server and sending lightweight HTML to the browser, resulting in faster loading times. Astro uses SSR (Server-Side Rendering) to deliver fully rendered content from the server, improving user experience compared to traditional frameworks like React, which use CSR (Client-Side Rendering).

The development of the user interface (UI) was done by combining Astro and React, allowing for a modular and efficient design. Although React runs on the client side, content is first rendered on the server, ensuring fast delivery. The web application includes:

- Dashboard: Visualizes the count of objects in the warehouse and provides detailed statistical data.
- Control Panel: Allows monitoring the inference process in real time, including a map with the real-time position of the prototype, which will be expanded in future versions of the project.

The combination of Astro and React provides a modern, fast and scalable solution, optimizing both development and the end-user experience.

4 Results

4.1 YOLOv8 Custom Model

Starting with the YOLOv8-based module, the F1 curve is a key indicator that combines the precision and recall of the model. As can be seen in Figure 5, the "box" class shows high performance with an F1 score of 0.9 for detections up to 95% confidence. However, beyond this threshold, the F1 decreases, suggesting that the model's recall decreases as confidence increases. On the other hand, the "platform" class performs slightly worse, with an F1 score of 0.85, which starts to drop at 80% confidence. This behavior may be due to the difference in the number of labels, since "box" has a total of 1404 annotations compared to "platform" 143.

Box 5

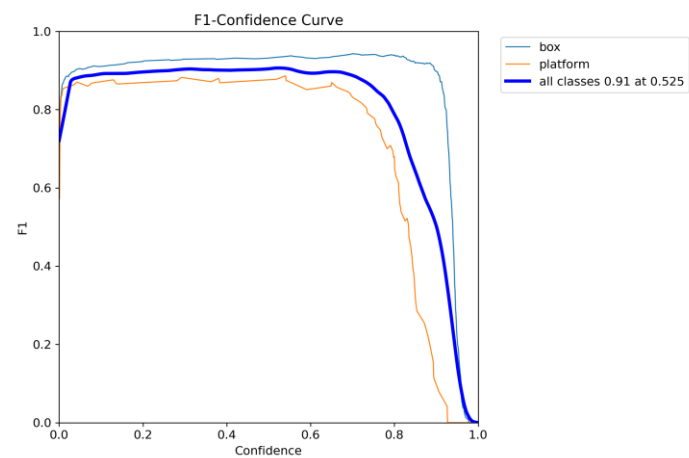


Figure 5
Curve F1 - Confidence

In the confusion matrix (Figure 6), we can analyze the detailed performance of the model. The Y-axis represents the predicted classes and the X-axis the actual classes. It is observed that the model has an accuracy of 94% for “platform” and 98% for “box”. The “background” class reflects false positives and negatives: false positives occur when the model detects a non-existent object, while false negatives occur when it does not detect a present object. According to the results, the model presents false positives in “box” (80%) and “platform” (20%), which can be attributed to the labeling process, where only complete objects were annotated. This could explain the detections of partially visible objects that were not labeled.

Box 6

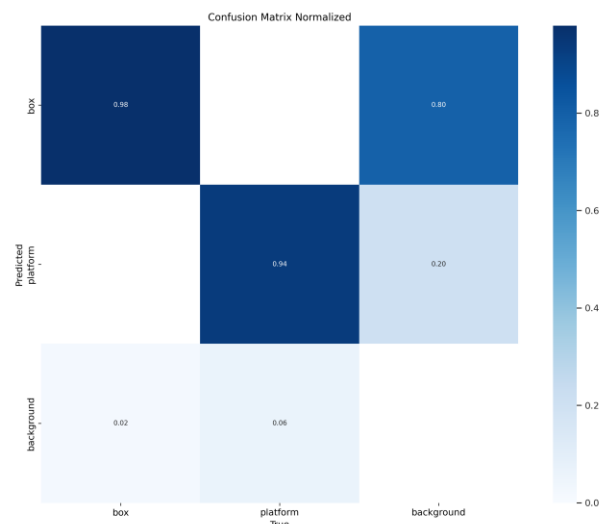


Figure 6
Confusion Matrix

4.2 Inventory Estimation

The estimating software also met the requirements set. Figure 7 shows that upon detecting a pallet, the box count is executed correctly, allowing for remarkable efficiency in processing. Proper labeling has been crucial for this functionality.

Box 7

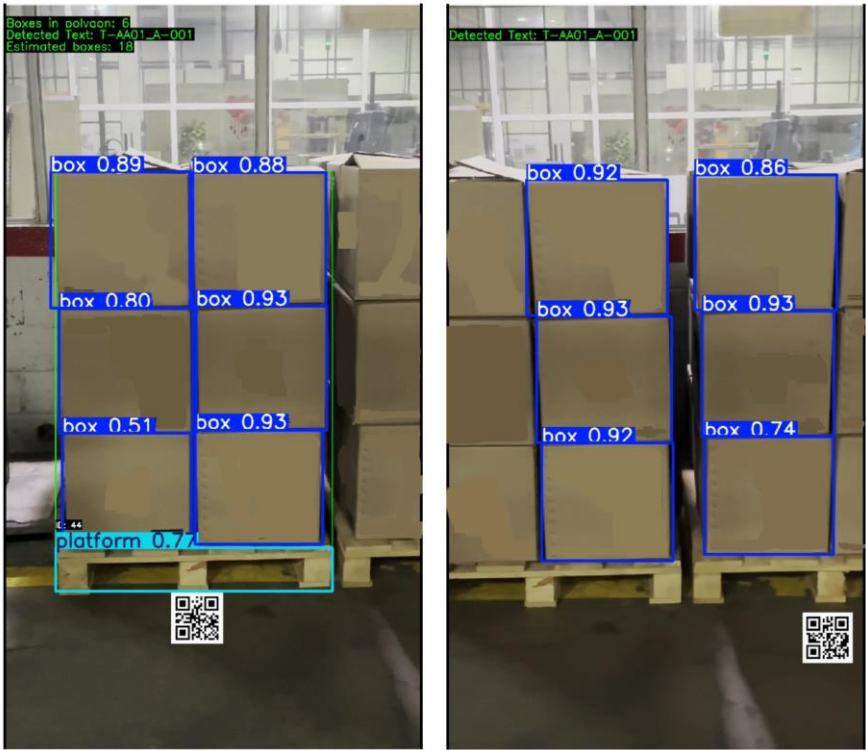


Figure 7
Estimation and Detection

4.3 Web Application

Regarding the web application, Figure 8 shows a dashboard where warehouse statistics are displayed: number of entries and exits, products with the highest stock and those with the highest turnover. All this data is presented in graphs to facilitate its interpretation.

In addition, Figure 8 shows the control interface, which allows the estimation software process to be viewed in real time, as well as monitoring and managing the prototype status. A section called "camera1" is also included, designed to view the prototype path.

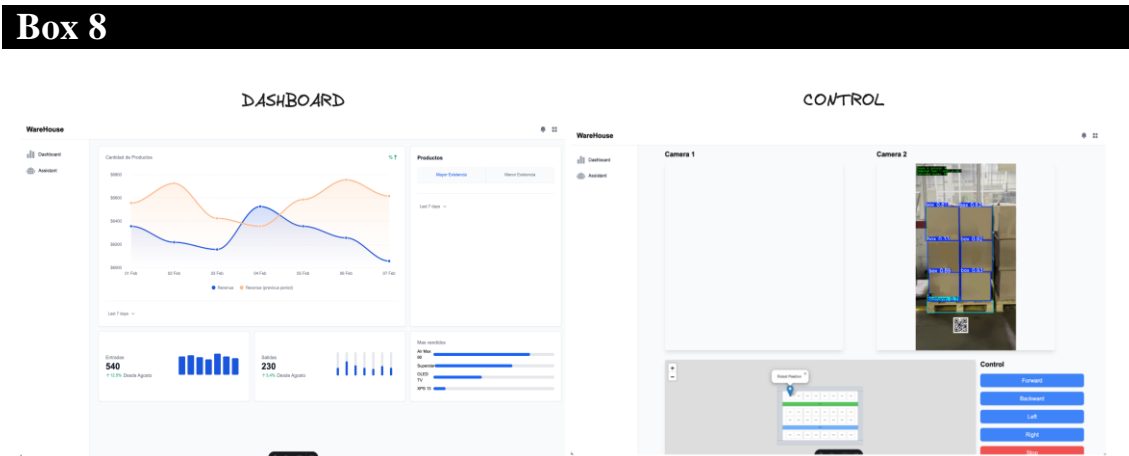


Figure 8
Dashboard and Control view

Conclusions

During the development of this first phase of the project, several challenges arose, especially related to the environment. The environment of a warehouse is highly variable, and this variability can represent a significant limitation. The diversity in the arrangement of boxes adds complexity to the system, making the design of a non-intrusive solution a considerable challenge.

Training the model also presented challenges. Typically, a large number of images is required to achieve good performance, which involves finding an appropriate balance between model parameters and the number of images per class. Despite these difficulties, the model achieved good performance on the validation and test datasets, although there is room for improvement.

As for the web application, it is possible to optimize both the design and the functionalities, especially with regard to the display of statistical data of the items. It is also advisable to continue improving the results obtained in the Lighthouse tests, with the aim of achieving scores close to 100% in all metrics.

The following actions are necessary to improve the project in a comprehensive manner:

- Labeling optimization: It is essential to improve and expand the dataset to optimize the performance of the model, as well as to improve the detection of objects based on their class and not just on the QR code.
- Adaptation to the environment: It is crucial to adjust the work cycle without making invasive modifications to the environment, thus achieving a more adaptable solution.
- Improvements to the web application: Both the design and the functionalities must be optimized to facilitate interaction and data management.
- Increase in the Lighthouse score: Work on optimizing performance, accessibility and best practices until reaching scores close to 100%.
- Development of a CI/CD system: Implement a Continuous Integration and Deployment pipeline that automates these processes, improving efficiency and reducing errors.
- Development of an autonomous prototype: Move towards a prototype that can perform detections autonomously in the warehouse environment.

As for the future of the project, the points mentioned above mark a clear path for the optimization and scalability of the system. These improvements will allow saving time and performing continuous audits without wasting resources.

Declarations

Conflict of interest

The authors declare that they have no conflict of interest. They have no financial interests or personal relationships that could have influenced this book.

Authors' contribution

Dávalos-Nava Gilberto: Design of the project idea. Choice of tools, methods, architecture and system techniques. Software development.

Reyes-Nava Adriana: Support in the idea of the project, Distribution of parts of the project, Review of the article.

Availability of data and materials

All data used for this research were derived from our own data analysis, no information from third parties was used.

Funding

This research did not receive any type of funding.

Acknowledgments

We thank the Tecnológico de Estudios Superiores de Jocotitlán and the entire academic community for their support during the development of this project. Their contribution in technical resources and academic guidance was essential to carry out this research.

Abbreviations

AI	Artificial Intelligence
API	Aplication Programming Interface
ARM	Advanced Risc Machine
CCTV	Closed Circuit Television
CD	Continuous Deployment
CI	Continuous Integration
CSR	Client Side Rendering
CV	Computer Vision
GPU	Graphic Processing Unity
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ID	Identification
IOU	Intersection Over Union
IT	Information Technology
JPG	Joint Photographic Experts Group
JSON	Java Script Object Notation
LIDAR	Light Detection And Ranging
ML	Machine Learning
ONNX	Open Neural Network Exchange
ORM	Object-Relational Mapping
QR	Quick Response Code
REST	Representational State Transfer
RFID	Radio-Frequency Identification
SEO	Search Engine Optimization
SPA	Single Page Application
SQL	Structured Query Language
SSR	Server Side Rendering
UI	User Interface
YOLO	You Only Look Once

References

Background.

[IA y machine learning: diferencia entre inteligencia artificial y machine learning.](#) AWS. (2023). Retrieved September 27, 2024, from Amazon Web Services, Inc.

IBM. (2021, October 6). [¿Qué es la inteligencia artificial \(IA\)?](#) | IBM. Retrieved September 27, 2024, from Ibm.com

[AI for Computer Vision Applications | NetApp.](#) (2023). Retrieved September 27, 2024, from Netapp.com

Basic

[Intenseye - Transforming Workplace Safety with AI](#). (2023). Retrieved September 27, 2024, from Intenseye.com

Simbe. (2023). [Meet Tally: Autonomous Inventory Robot](#) | Simbe Robotics. Retrieved September 27, 2024, from Simbe

Yadav, D. (2022, March 22). [DIFFERENT MACHINE LEARNING MODELS](#) - Deepika Yadav - Medium. Retrieved September 27, 2024, from Medium

GeeksforGeeks. (2019, October 23). [ClientServer Model](#). Retrieved September 27, 2024, from GeeksforGeeks

[Introducción a JavaScript asíncrono - Aprende desarrollo web](#) | MDN. (2023, September 8). Retrieved September 27, 2024, from MDN Web Docs

Support

Nefe Emadamerho-Atori. (2024, May 7). [Client-side Rendering \(CSR\) vs. Server-side Rendering \(SSR\)](#). Retrieved September 27, 2024, from Prismic.io

[What is Scrum?](#) (2022, December 23). Retrieved September 27, 2024, from NimbleWork

Torres, J. (2024, January 15). [YOLOv8 Architecture; Deep Dive into its Architecture -Yolov8](#). Retrieved September 27, 2024

[Roboflow: Computer vision tools for developers and enterprises](#). (2024). Retrieved September 27, 2024, from Roboflow.com

Vina, A. (2024, August 21). [What is ByteTrack? A Deep Dive](#). Retrieved September 27, 2024, from Roboflow Blog

[SQLite Home Page](#). (2024). Retrieved September 27, 2024, from Sqlite.org website:

[FastAPI](#). (2024). Retrieved September 27, 2024, from Tiangolo.com website:

Differences

[What is RESTful API? - RESTful API Explained - AWS](#). (2022). Retrieved September 27, 2024, from Amazon Web Services, Inc.

Discussions

[Evaluating Deep Learning Models: The Confusion Matrix, Accuracy, Precision, and Recall](#) | DigitalOcean. (2024). Retrieved September 27, 2024, from Digitalocean.com