Automatic Parcel Damage Recognition Module for an Inspection Robot

Wiktor Goszczyński, Szymon Wałęga, Janusz Chmiel, Bartłomiej Gawęda, Katarzyna Grobler-Dębska
AGH University of Krakow
Dep.of Automatic Control and Robotics
al. Adama Mickiewicza 30
30-059 Kraków, Poland
Email: {wiktorg, swalega, januszchm}@student.agh.edu.pl,{bargaw, grobler}@agh.edu.pl

***Abstract*—This paper describes our work on expanding machine learning hardware and algorithm solutions for a damage detection inspection robot in order to create a digital twin. Building on previous projects—one featuring an autonomous mobile robot and another focused on computer vision for package damage detection—we combine these efforts to improve clarity and speed in real-world inspections. We developed a dataset of over 6,800 images and applied a tailored data augmentation process to better capture the variability found in operational environments. Our approach refines a YOLOv11n-cls-based model, achieving 98.50% accuracy, 97.% precision, and 99.74% recall on validation data. By optimizing the model for deployment on widely available hardware via CoreML, we reached inference speeds exceeding 251 FPS, ensuring rapid processing. An interactive dashboard was also created to monitor performance and facilitate comparisons between model iterations.**

I. Introduction

A digital twin of a warehouse is a virtual replica of the physical warehouse environment, including its operations, inventory, and infrastructure. This dynamic model is continuously updated with real-time data, enabling warehouse managers to monitor, analyze, and optimize processes without disrupting actual operations. The creation of such a sophisticated system is inherently complex and comprises two primary components:

- Data acquisition: This includes a robot that navigates the warehouse, avoids obstacles, inspects the condition of packages, and records their data.
- Data processing: This component is responsible for analyzing data received from the robot, storing it, and presenting it in an accessible manner.

Data processing part relies heavily on big data solutions, such as technologies for data transmission, storage, and visualization. In contrast, the data acquisition requires the implementation of various specialized algorithms, including:

- Localization algorithms: To determine the robot's precise position within the warehouse.
- Path-planning algorithms: For defining the robot's optimal navigation routes.
- Obstacle-avoidance algorithms: To ensure safe and efficient movement.
- Machine vision systems: For identifying obstacles and recognizing packages in real-time.

From another perspective, a warehouse inspection robot serves as a valuable tool for testing and validating these algorithms. It bridges the gap between theoretical concepts and practical implementation. This synergy is the rationale behind initiating such a project within the Industrial Data Science (IDS) student research group. This document delves into the process of constructing such a robot and implementing a computer vision algorithm for package damage detection.

Traditional methods of identifying damaged packages are highly dependent on manual inspection, which is time- consuming, inconsistent, and impractical given the high number of packages processed daily. Computer vision-based automated inspection systems offer a promising solution to this challenge by providing a scalable, consistent, and efficient alternative. Our work contributes to the field of logistics automation by demonstrating a practical implementation of deep learning for quality control in shipping operations. The developed system not only achieves high classification accuracy, but is also optimized for real-time applications, making it suitable for integration into existing logistics workflows.

A. Related Works

Modern warehouses are under increasing pressure to optimize logistics processes, which has led to growing interest in automation technologies based on autonomous mobile robots (AMRs). Unlike traditional automated guided vehicles (AGVs), AMRs are capable of making independent decisions and navigating dynamic environments without the need for physical navigation infrastructure [3]. Thanks to advances in sensor technologies, control algorithms, and artificial intelligence, AMRs are increasingly being used not only for transporting goods but also for inspection tasks in warehouses and logistics centers [9].

One of the key applications of AMRs in this context is automatic package identification and inspection. In [4], the authors present the use of a MicroAir Vehicle (MAV), equipped with a SLAM localization system and an onboard camera, to autonomously inspect a warehouse. Operating in a GPS-denied environment, the robot was able to detect QR codes on packages placed on shelves. Inspection data were transmitted to a ground control station, enabling full documentation and tracking of resources.

A more advanced approach is presented in [7], where the authors evaluate the effectiveness of object detection models—such as YOLO—used by AMRs for inspection of logistics infrastructure. These systems support not only navigation but also real-time package recognition and condition assessment. Field tests confirmed the high effectiveness of AI-based solutions in automating inspection and maintenance, demonstrating significant potential for deployment in real warehouse environments.

Another notable implementation is found in [6], which describes a ground-based AMR built with a differential drive, depth camera, and LiDAR sensor, capable of autonomously navigating warehouse environments. The robot's task was to detect and count boxes on shelves, then compare the results with stored inventory data, enabling automatic stock verification. Deep learning-based object detection models further increased inspection accuracy.

In addition, the review paper [8] offers a broader technological context by highlighting the transformative role of artificial intelligence in warehouse automation, including its application in AMRs. The study emphasizes how AI-driven systems can enhance the perception and decision-making capabilities of robots, with computer vision algorithms enabling functions such as object recognition, quality control, and autonomous visual inspection. The paper also discusses emerging trends such as edge computing and reinforcement learning, both of which contribute to the adaptability and real-time responsiveness of AMR-based inspection systems.

The literature emphasizes the growing importance of autonomous mobile robots in tasks related to package control and inspection in warehouse environments. The integration of advanced sensors, vision systems, and AI algorithms with mobile robotic platforms lays a solid foundation for further developments toward fully automated warehouse processes and high-quality logistics services.

II. Custom Inspection Robot

Due to budget constraints and the lack of commercially available robots that met our specific requirements, we developed a custom inspection platform. This robot serves as the core of our package inspection system, incorporating hardware needed for the parcel damage recognition system.

A. Robot Requirements

The design was driven by several key requirements:

- **Compact size and modularity:** Facilitates easy transport and customization.
- **RGB Camera:** Enables visual inspection and data collection.
- **LiDAR:** Provides precise spatial awareness and obstacle detection.
- **Ultrasonic Distance Sensors:** Offer additional proximity sensing for enhanced safety.
- **Omnidirectional Drive:** Ensures agile maneuvering in constrained environments.
- **Battery Power Supply:** Supports portability and extended operation.

B. Mechanical Design and Drive System

To achieve omnidirectional movement, the robot is equipped with mecanum wheels driven by 360-degree hobby servos. These were chosen for their cost efficiency and simplicity, as they operate using PWM signals and don't require additional control circuitry. Two independent, galvanically isolated power circuits, along with optocouplers are used to ensure that signal interference between the motor system and control electronics is minimized.

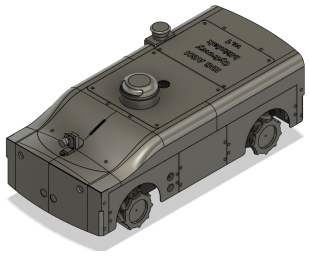

Fig. 1. Conceptual Model of the Inspection Robot

C. Control System Architecture

The onboard control system comprises two main units:

- **NVIDIA Jetson Nano:** Oversees external communication, processes sensor data from the RGB camera and LiDAR, and supports machine learning algorithms.
- **Arduino Microcontroller:** Handles motor control via PWM signals and gathers wheel position data from magnetic sensors.

These units communicate via USB using JSON over a serial connection at 115200 baud, ensuring a data exchange cycle of approximately 30 ms.

D. Environmental Sensing and Vision System

To effectively inspect the environment, the robot integrates an RGB camera and LiDAR, both connected to the Jetson Nano. The camera, mounted on a hobby servo, can rotate to cover the required field of view—making it possible to detect obstacles and inspect packages simultaneously. Future improvements may include replacing the servo with dedicated cameras for forward and side views.

E. Prototyping and Testing

The initial design was first modeled and then brought to life through 3D printing. The physical prototype was rigorously tested in the lab to validate its functionality and ensure that it meets the design requirements.

III. Dataset Development

The foundation of our parcel damage classification system developed for the inspection robot is a comprehensive, high-quality dataset that accurately represents the problem. This section details our approach to dataset construction, exploration, and augmentation.

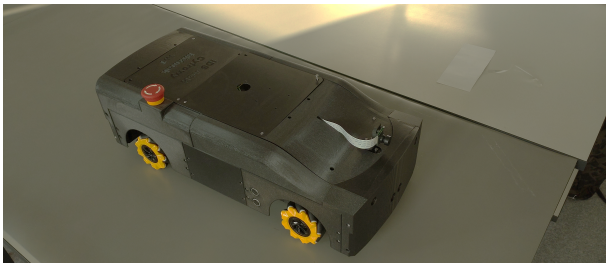

Fig. 2. 3D Printed Prototype of the Inspection Robot

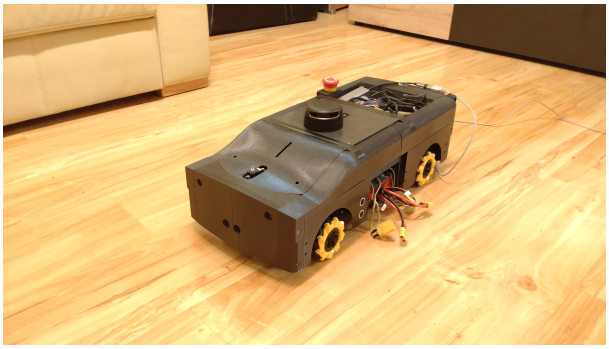

Fig. 3. Robot during LIDAR Testing in Lab Conditions

A. Data Collection and Exploration

Initial data collection began with a publicly available dataset from Kaggle [1] with . We used the FiftyOne [5] framework for extensive dataset exploration and analysis. FiftyOne provides valuable tools for:

- Visualizing dataset samples and their distributions
- Identifying potential biases in the data
- Detecting and removing duplicate or near-duplicate images
- Verifying label quality and consistency

Upon inspection using this framework, we found several limitations of gathered photos:

- Poor image quality in numerous samples.
- Limited variety in package types and damage patterns.
- Presence of unrealistic or staged damage scenarios

To address these limitations, we supplemented the initial dataset with custom image captured in realistic environments in different situations and scenes. This approach ensured that our training data better represented real-world scenarios encountered in logistics operations, resulting in improved classification.

This exploration phase was crucial for ensuring dataset quality prior to model training and helped identify specific areas that required additional data collection.

B. Augmentation Strategy

Data enhancement plays a critical role in improving the robustness of the model. Rather than applying generic augmentation techniques, we developed a domain-specific augmentation pipeline using the Albumentations library. Our

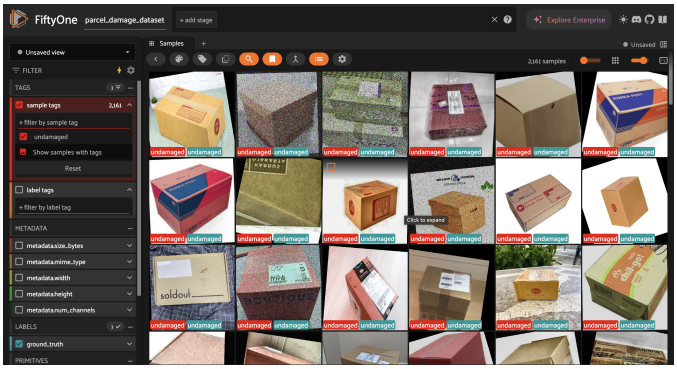

Fig. 4. Shows representative samples from our dataset, highlighting the variety of parcel types, lighting conditions, labeling, and damage patterns.

pipeline was carefully designed to simulate realistic variations that parcels might encounter in logistics environments while avoiding unrealistic transformations that could harm model performance. We applied:

- Conservative spatial transformations to simulate realistic camera angle variations:
  - Slight rotations (±10°) with a 50% probability
  - Small affine transforms (scale factor 0.95-1.05, translation ±3%)
  - Horizontal flips with a 30% probability
  - No-op spatial transformations with a 20% probability
- To simulate different indoor lighting conditions commonly found in warehouses and sorting facilities:
  - Random brightness/contrast adjustments (±10%) with a 50% probability
  - Color temperature shifts (brightness ±5%, contrast ±5%, saturation ±5%, hue ±5%) with a 30% probability
  - No lighting modifications with a 20% probability
- Shadow Effects To simulate natural lighting conditions:
  - Very mild random shadows with a 20% probability
  - No shadow effect with a 80% probability

The final pipeline included consistent resizing to 640×640 pixels while maintaining aspect ratio, implemented through a combination of LongestMaxSize and PadIfNeeded transformations with constant border padding. This pipeline was applied 10 times to each of the original photos.

The final dataset was organized into a structured format with two primary categories:

- Damaged: 2,372 training images and 903 validation images
- Undamaged: 2,600 training images and 1,000 validation images

IV. Training Classifier

For our parcel damage detection task, we selected the YOLOv11n-cls model [2], a classification variant of the YOLO (You Only Look Once) family. This approach was chosen for several reasons:

Fig. 5. Illustrates examples of our augmentation pipeline applied to sample images, demonstrating the realistic nature of the transformations.

- Efficient inference speed, critical for real-time applications in logistics
- Strong feature extraction capabilities from the backbone network
- Successful track record in related computer vision tasks
- Flexibility to be deployed on various hardware platforms

A. Training Methodology

We trained the model using the Ultralytics framework [2], which provides an efficient implementation of YOLO models with several optimizations for training and inference. The training configuration used the following parameters:

- Input image size: 640×640 pixels
- Training epochs: 100
- Model architecture: YOLOv11n-cls.pt

Other hyperparameters such as batch size, optimizer settings, and learning rate schedule were automatically determined by the Ultralytics framework based on dataset characteristics and hardware capabilities. This approach leverages the expertise embedded in the framework to achieve optimal training results without manual hyperparameter tuning saving a lot of time.

Our model development followed an iterative improvement process with following steps:

1) Initial training with the training part of created dataset resulted in our baseline model (best0)
2) Performance analysis of best0. That revealed a specific weakness: opened boxes were frequently misclassified as undamaged
3) Additional data collection focused specifically on opened boxes, labeled as damaged
4) Dataset augmentation with the new samples
5) Re-training with the enhanced dataset resulted in our improved model (best1)
6) Performance analysis of best1. That revealed no major weaknesses in classification

This iterative approach allowed us to improve the model performance. After the 72 epoch the model (best1) reached the highest accuracy and later this parameter remained at the same level. That shows that choice of maximum 100 epochs long training was enough for model to reach its best performance.

B. Model Optimization

During testing with real-time inference the PyTorch model's frames per second (FPS) was around 30, meaning that it can process that number of sample pictures in one second. After converting to CoreML format the model reached 251+ FPS. We converted our models to ONNX format to address two primary objectives: first, to enable robust metric extraction and validation within our performance monitoring dashboard, as the standardized ONNX format provides more predictable parameter access patterns compared to native PyTorch. Secondly, changing model's format allowed us to establish a framework-agnostic model representation that enhances deployment flexibility and ensures consistent inference behavior across different runtime environments. The training process was monitored using metrics that included training loss, validation loss, and top-1 accuracy, which were recorded every epoch.

We achieved this using the following process:

1) Export of the trained PyTorch model to ONNX format
2) Conversion from ONNX to CoreML using the coremltools library
3) Optimization of the CoreML model for the Apple Neural Engine

The high inference speed makes the model suitable for integration into high-throughput logistics environments where real-time processing is critical.

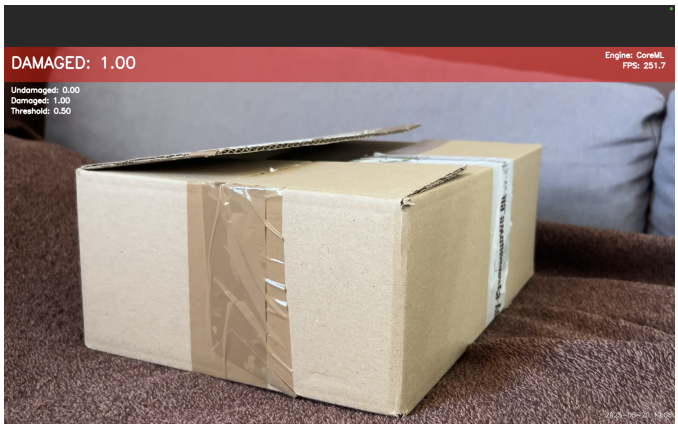

Fig. 6. Optimization process maintained model accuracy while significantly improving inference speed, achieving over 251 FPS on Apple M1 hardware during real-time testing.

V. Results and Evaluation

A. Performance Metrics

We evaluated our models using classification metrics including accuracy, precision, recall, and F1 score.

The best0 and best1 models achieved results as presented in Table I. The best1 model demonstrated significant improvements across all metrics. The enhancements in precision are particularly notable, indicating that our approach successfully reduced false positive classifications.

Also we tried our models under real-life conditions such as too dark or too light environment. We believe that our models

TABLE I
SHOWS THE TRAINING STATISTICS FOR BOTH MODELS, ILLUSTRATING THE IMPROVEMENT IN FINAL PERFORMANCE ACHIEVED BY THE BEST1 MODEL AFTER ADDRESSING THE IDENTIFIED WEAKNESSES IN THE DATASET.

Metric	best0 model	best1 model	Improvement
Accuracy	97.00%	98.50%	+1.50pp
Precision	94.80%	97.04%	+2.24pp
Recall	98.81%	99.74%	+0.93pp
F1 Score	96.76%	98.37%	+1.61pp

performance can be increased by hyper-tuning or changing the augmentation parameters for images. Scores achieved by both models in this scenario are presented in Table II.

TABLE II
SHOWS STATISTICS FOR BOTH MODELS IN BOTH DARKER AND LIGHTER ENVIROMENTS USING REAL-LIFE IMAGES AND AUGMENTATIONS

Metric	best0 model	best1 model	Improvement
Accuracy	92.06%	94.44%	+2.38pp
Precision	89.74%	91.14%	+1.4pp
Recall	97.22%	100%	+2.78pp
F1 Score	93.33%	95.36%	+2.03pp

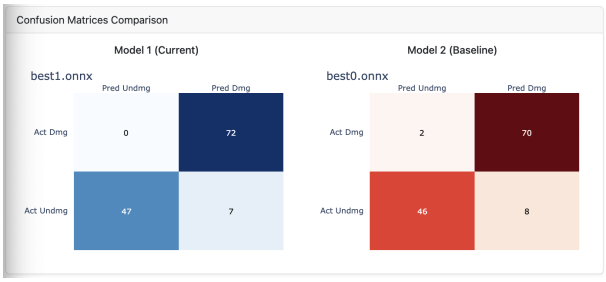

Fig. 7. Confusion matrices after overexposure the images

B. Confusion Matrix Analysis

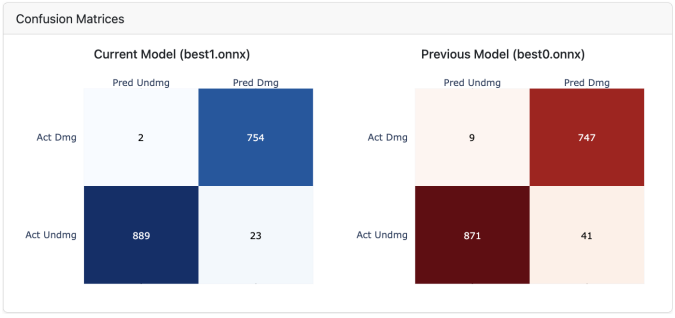

Fig. 8. Displays the confusion matrices for both models

Detailed analysis of these matrices reveals:

- True Positives (Damaged correctly classified): Increased from 747 to 754
- False Negatives (Damaged classified as Undamaged): Decreased from 9 to 2, representing a 77.8% reduction
- True Negatives (Undamaged correctly classified): Increased from 871 to 889
- False Positives (Undamaged classified as Damaged): Decreased from 41 to 23, representing a 43.9% reduction

The substantial reduction in false negatives is particularly important for our application context. In parcel damage detection, false negatives (missed damages) typically have a greater business impact than false positives, as they can lead to damaged items being delivered to customers, resulting in customer dissatisfaction and potential financial losses.

C. Precision-Recall Analysis

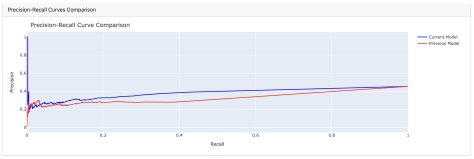

Fig. 9. Shows the precision-recall curves, where the best1 model maintains higher precision across all recall values. These analyses provide a more comprehensive view of model performance beyond the single operating point represented by the confusion matrix, demonstrating that the best1 model achieves superior performance across various classification thresholds.

D. Visual Inspection of Classification Results

Beyond quantitative metrics, we conducted a qualitative analysis of classification results to identify patterns in the remaining misclassifications. For the best1 model, we observed the following:

- False negatives (2 instances): Both cases involved parcels with minor damage that was partially obscured or at the edge of the package.
- False positives (23 instances): Primarily involved parcels with heavy tape reinforcement or unusual packaging materials that resembled damage patterns.

These observations provide valuable insights for future improvements, suggesting that focusing on capturing more examples of minor/obscured damage and diverse packaging materials could further enhance model performance. Fig. 6. 11

VI. DASHBOARD FOR VISUALIZATION AND ANALYSIS

The dashboard specially developed for our project provides a comprehensive interactive platform for model evaluation and comparison. Built using Dash, Plotly, and Bootstrap, it offers a clean and responsive interface that effectively visualizes model performance metrics while enabling real-time testing capabilities. Our dashboard has a feature of testing model performance or comparing models on loaded by user-specific folder or image via our locally hosted dashboard. Fig. 7.

A. Key Components and Features

1) Model Selection and Comparison
   - Flexible switching between current model (best1.onnx) and previous model (best0.onnx)

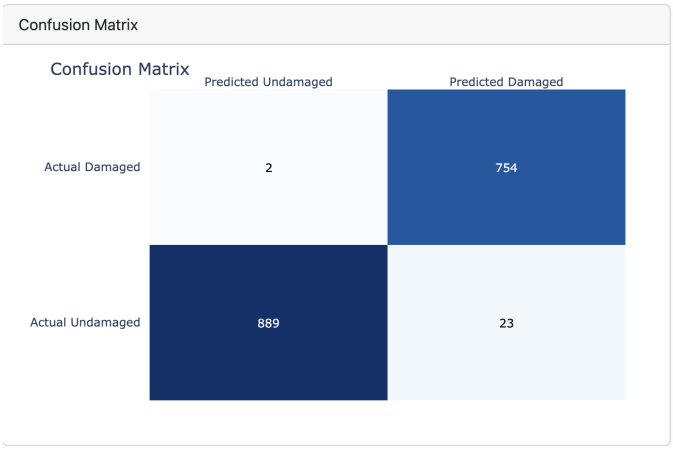

Fig. 10. Confusion matrix for best1

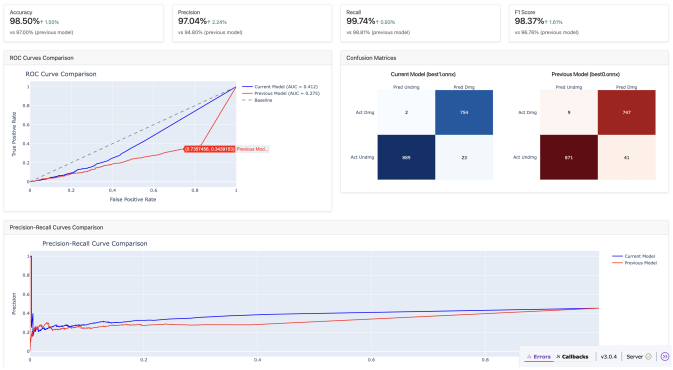

Fig. 11. Dashboard

- Side-by-side comparison view for direct performance assessment
- Custom test folder specification option for evaluating performance on different datasets

2) Comprehensive Visualization
   - Interactive confusion matrix heatmaps with annotation overlays
   - ROC curves with AUC metrics calculation
   - Precision-Recall curves for threshold-independent performance assessment
   - Dynamic graph generation using Plotly's interactive visualization capabilities
3) Real-Time Image Testing
   - Drag-and-drop or file selection interface for custom image upload
   - Simultaneous prediction using both model versions
   - Visual confidence display using progress bars and percentage indicators
   - Side-by-side comparison of model predictions on the same image
4) Performance Metrics Display
   - Clean card-based presentation of key metrics (Accuracy, Precision, Recall, F1 Score)
   - Percentage formatting with color-coding for easy interpretation

B. Technical Implementation Analysis

1) Model Integration and Inference: The model loading and prediction pipeline includes:

1) Image preprocessing pipeline:
   - Resizing to the expected 640×640 dimensions
   - RGB conversion and normalization (scaling pixel values to 0-1)
   - Tensor transformation to channel-first format (CHW) required by the model
   - Batch dimension addition for inference compatibility
2) Efficient prediction handling:
   - Proper input tensor naming and shaping for ONNX runtime compatibility
   - Output processing logic that accounts for the binary classification nature of the task
   - Confidence score calculation and class determination
3) Optimization techniques:
   - Results caching mechanism through JSON files to avoid redundant computation
   - Temporary file management for uploaded images with UUID-based naming
   - Progress tracking for batch evaluation processes

VII. Future works

While the current system achieves excellent performance, several promising directions for future development include:

1) Damage Localization: Extending the system to not only classify parcels as damaged but also localize and highlight specific damaged areas.
2) Damage Type Classification: Developing a multi-class model that can distinguish between different types of damage (e.g., tears, water damage, crushing) to provide more detailed information.
3) Edge Deployment: Further optimization for deployment on edge devices directly on conveyor systems using techniques like model quantization.
4) Multi-View Integration: Combining predictions from multiple camera angles to improve detection accuracy for damages that might only be visible from certain perspectives.
5) Continuous Learning Framework: Implementing a system for ongoing model improvement based on operational feedback and new data collection.

VIII. Conclusion

This article demonstrates the successful application of machine learning to a practical industrial problem, achieving performance levels that make it suitable for real-world deployment. The comprehensive approach - from data preparation through model development to deployment optimization and

evaluation - resulted in a system that balances accuracy, speed, and usability.

The very high recall rate of 99.74% is particularly noteworthy, as it addresses the most critical requirement for a damage detection system: minimizing missed detections. Combined with deployment-ready inference speeds and a user-friendly evaluation interface, this project represents a complete solution ready for practical implementation in logistics operations.

In summary, this parcel damage detection system showcases how thoughtful application of computer vision techniques, with particular attention to data quality and domain-specific considerations, can yield highly effective solutions to real-world logistics challenges.

APPENDIX

Appendixes, if needed, appear before the acknowledgment.

ACKNOWLEDGMENT

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g." Try to avoid the stilted expression, "One of us (R. B. G.) thanks ..." Instead, try "R. B. G. thanks..." Put sponsor acknowledgments in the unnumbered footnote on the first page.

REFERENCES

[1] Damaged and intact packages. http://web.archive.org/web/20080207010024/http://www.808multimedia.com/winnt/kernel.htm, accessed: 2025-05-23

[2] Jocher, G., Qiu, J.: Ultralytics yolo11 (2024), https://github.com/ultralytics/ultralytics

[3] Keith, R., La, H.M.: Review of autonomous mobile robots for the warehouse environment. arXiv preprint arXiv:2406.08333 (2024)

[4] Martinez-Carranza, J., Rojas-Perez, L.O.: Warehouse inspection with an autonomous micro air vehicle. Unmanned Systems **10**(04), 329–342 (2022). https://doi.org/10.1142/S2301385022410011, https://doi.org/10.1142/S2301385022410011

[5] Moore, B.E., Corso, J.J.: Fiftyone. GitHub. Note: https://github.com/voxel51/fiftyone (2020)

[6] Parikh, H., Saijwal, I., Panchal, N., Sharma, A.: Autonomous mobile robot for inventory management in retail industry. In: Singh, P.K., Wierzchoń, S.T., Chhabra, J.K., Tanwar, S. (eds.) Futuristic Trends in Networks and Computing Technologies. pp. 93–103. Springer Nature Singapore, Singapore (2022)

[7] Sanchez-Cubillo, J., Del Ser, J., Martin, J.L.: Toward fully automated inspection of critical assets supported by autonomous mobile robots, vision sensors, and artificial intelligence. Sensors **24**(12) (2024). https://doi.org/10.3390/s24123721, https://www.mdpi.com/1424-8220/24/12/3721

[8] Sodiya, E.O., Umoga, U.J., Amoo, O.O., Atadoga, A.: Ai-driven warehouse automation: A comprehensive review of systems. GSC Advanced Research and Reviews **18**(2), 272–282 (2024)

[9] Xuan, O.W., Selamat, H., Muslim, M.T.: Autonomous mobile robot for transporting goods in warehouse and production. In: Tan, A., Zhu, F., Jiang, H., Mostafa, K., Yap, E.H., Chen, L., Olule, L.J.A., Myung, H. (eds.) Advances in Intelligent Manufacturing and Robotics. pp. 555–565. Springer Nature Singapore, Singapore (2024)