International Journal of Multidisciplinary Studies and Innovative Technologies

e-ISSN: 2602-4888
dergipark.org.tr/en/pub/ijmsit
Research Article

2025, 9 (1), 53-58
DOI: 10.36287/ijmsit.9.1.8
Received: April 30, 2025; Accepted: June 19, 2025

Ensuring product detection and product counting on the assembly line using deep learning (YOLOv11)

Muhammed Abdullah ÖZEL[1*], Mehmet Yasin GÜL [2]

[1*] *AYD Automotive Industry R&D Center, Konya, Turkey (ozel.muhammed@aydtr.com) (ORCID: 0000-0003-3056-6438)*
[2] *AYD Automotive Industry R&D Center, Konya, Turkey (gul.mehmetyasin@aydtr.com) (ORCID: 0009-0002-5796-9886)*

Abstract **–**This study aims to improve product counting and quality control processes on the suspension system assembly line by utilizing the YOLO (You Only Look Once) algorithm and the Ultralytics library. Given the critical importance of suspension systems for vehicle control and passenger comfort, the accuracy of parts inspection on the production line holds paramount industrial significance. As reliance on operator discretion frequently leads to customer complaints and substantial financial losses, deep learning-based object detection models offer considerable potential to automate and accelerate these processes. In this context, the effectiveness of the YOLOv11 model in industrial parts inspection will be scientifically demonstrated in this work, not only through theoretical inferences but also via detailed experimental results and detailed performance comparisons with previous models. Our ultimate goal is to provide concrete contributions to enhancing production efficiency while ensuring the highest levels of product quality and reliability. This approach clearly showcases the transformative power of automation and artificial intelligence in modern manufacturing.

Keywords *– YOLO, Deep Learning, Ultralytics, Suspension, Detect.*

Citation: Özel, M., Gül, M. (2025). The Title of The Article In English. International Journal of Multidisciplinary Studies and Innovative Technologies, 9(1): 53-58.

I. INTRODUCTION

Suspension is a connection system that links a vehicle's wheels and regulates their relationship with the road surface. A good suspension enhances both the vehicle's control and driving safety while ensuring passenger comfort [1]. Components in this system, such as the Z-rod, tie rod, swing arm, and ball joint, are critical in the manufacturing process. Accurate counting of products on the assembly line is crucial for detecting any deficiencies or excesses in production. If the detection and inspection of products are left to the operator's discretion, it can lead to customer complaints and financial losses. Additionally, if product counting is done accurately, the detection and counting of product subcomponents can also be carried out in subsequent operations. In Figure 1, AYD production suspension parts are shown in red.

Fig. 1. AYD Production Suspension Parts

Initially introduced by Redmon et al. [2] as a revolutionary approach to real-time object detection, YOLO (You Only Look Once) is a deep learning model developed for real-time object detection. It is used to quickly detect and classify objects in images. Object counting involves the accurate detection and counting of specific objects through videos and camera feeds [3]. Subsequent iterations, such as YOLOv2 [4], YOLOv3 [5], and YOLOv4 [6], further advanced its capabilities, improving both speed and accuracy. In this process, YOLOv11 provides high efficiency and accuracy by utilizing state-of-the-art algorithms and deep learning capabilities. YOLOv11 quickly and accurately detects objects, particularly in complex scenarios like crowd analysis and surveillance. Thanks to its features, it excels in real-time applications and performs object counting both accurately and effectively. This enables high-precision operations in fields such as security monitoring, inventory management, and other areas that require object detection [7].

One of YOLO's key features is its ability to detect all objects in an image in a single pass. This enhances the speed of the model. Object counting provides several benefits. Firstly, it significantly aids resource optimization. Accurate object counts enable efficient planning in inventory management and other resource allocation processes, helping to avoid unnecessary expenses and stock surpluses. It also improves security; accurately counting and tracking objects increases asset security and helps detect potential threats more rapidly. Finally, object counting plays a major role in informed decision-making. In sectors like retail and traffic management, accurate data allows for more efficient process management and provides valuable insights for strategic decision-making. This ensures that decisions are made on a solid foundation.

Ultralytics YOLOv11 offers several advantages over other object detection models like Faster R-CNN [8], SSD [9], and previous YOLO versions. YOLOv11 provides real-time processing capabilities, making it ideal for applications that require high-speed inference, such as surveillance and autonomous driving. The model delivers state-of-the-art accuracy in object detection and tracking tasks, reduces false positives, and improves overall system reliability. Additionally, it offers seamless integration with various platforms, including mobile and edge devices, which is a significant advantage for modern AI applications. YOLOv11 is flexible, with configurable models that support various tasks such as object detection, segmentation, and tracking, allowing it to meet specific use case requirements [7]. Advances in YOLOv11's model design and optimization techniques have resulted in higher accuracy with fewer parameters. The improved architecture enables efficient feature extraction and processing, allowing for higher mean average precision (mAP) on large datasets like COCO. Despite using 22% fewer parameters than the YOLOv8m model, YOLOv11 does not compromise on accuracy. This makes YOLOv11 well-suited for efficient deployment on resource-constrained devices, as the model continues to deliver high performance while requiring less computational power. Thus, YOLOv11 stands out as an effective solution, especially in environments with limited computational power, such as portable devices or real-time applications. This table 1 provides an overview of the YOLO11 model variants, showcasing their applicability in specific tasks and compatibility with operational modes such as Inference, Validation, Training, and Export. This flexibility makes YOLO11 suitable for a wide range of applications in computer vision, from real-time detection to complex segmentation tasks [10]. Table 1. YOLO v11 versions and their functions are shown.

Table 1. YOLO v11 versions and tasks

Model	Filenames	Task
YOLO11	yolo11n.pt, yolo11s.pt yolo11m.pt yolo11x.pt yolo11l.pt	Detection
YOLO11-seg	yolo11n-seg.pt yolo11m-seg.pt yolo11x-seg.pt yolo11s-seg.pt yolo11l-seg.pt	Instance Segmentation
YOLO11-pose	yolo11n-pose.pt yolo11s-pose.pt yolo11m-pose.pt yolo11l-pose.pt yolo11x-pose.pt	Pose/Keypoints
YOLO11-obb	yolo11n-obb.pt yolo11m-obb.pt yolo11x-obb.pt yolo11s-obb.pt yolo11l-obb.pt	Oriented Detection
YOLO11-cls	yolo11n-cls.pt yolo11m-cls.pt yolo11x-cls.pt yolo11s-cls.pt yolo11l-cls.pt	Classification

II. Materials and Method

The suspension system is a mechanism that connects a vehicle's wheels to its chassis and regulates the interaction between the wheels and the road surface. The primary purpose of this system is to improve the vehicle's handling, driving safety, and passenger comfort. The suspension absorbs shocks from road irregularities, reducing vibrations between the vehicle and the road, providing a smoother driving experience. In a suspension system, several components limit the movement between the chassis and the wheels according to road conditions, such as springs, shock absorbers, tie rods, control arms, Z-rods, and swing arms. These components balance the vehicle's dynamic movements, ensuring stability during acceleration, braking, and cornering. The suspension system not only enhances driving quality but also ensures the vehicle's safety. Therefore, the interaction between the vehicle's chassis and wheels is of great importance. The suspension system is designed to ensure that vehicles move steadily and in a controlled manner under all road conditions.

A. AYD Suspension Parts

The control arm, Z-rod, tie rod, steering rack, steering shaft, and control link are key components in a vehicle's suspension system. The control arm connects the chassis to the wheels, helping to regulate movement. The Z-rod contributes to the vehicle's balance and steering control, functioning as part of the control arm. The tie rod allows the wheels to rotate vertically around their axis, aiding in steering. The steering rack connects the steering system to the wheels, enabling the vehicle to maneuver. The steering shaft transmits steering inputs to the wheels. The control link, as part of the control arm, regulates the connection between the chassis and the wheels, ensuring stable vehicle movement. Together, these components work in harmony to improve vehicle handling, driving safety, and comfort. Red colored suspension parts are seen in Figure 2.

Fig. 2. Red colored suspension parts

B. Deep Learning

Deep learning is a subfield of artificial intelligence (AI) that enables computers to learn from large datasets. This technique uses artificial neural networks, which are structures inspired by the human brain, to automatically extract patterns from data. Deep learning is particularly effective in handling large and complex data, and it is commonly used in areas like pattern recognition, image processing, and natural language processing. The model processes data in layers, learning more abstract features at each layer, making it more powerful and flexible compared to traditional machine learning methods.

C. Data collection and Processing

Data collection for YOLO (You Only Look Once) is the process of gathering the visual data necessary for the model to correctly detect and classify objects. This process involves several important factors, including the diversity, quality, and accurate labeling of the collected data. During data collection, various conditions such as different angles, lighting conditions, distances, and backgrounds must be considered to make the model more generalizable. Each object in the images must be accurately labeled with its class name and bounding boxes. Additionally, since low-resolution or blurry images can negatively affect the model's performance, it's important to use high-quality images. Balanced data is also crucial, as the overrepresentation of a particular object class can lead to bias in the model. During the data collection process, data augmentation techniques like rotation and scaling can also be used to improve the model's ability to generalize. Figure 3 shows the data collection visual.

Fig. 3. Data collection

To facilitate product counting and inspection on the suspension system assembly line, a high-quality visual dataset was created.

Data was collected using “Dahua IPC-HFW2431S-S-S2” model IP cameras installed on the assembly line. The cameras were configured to record images at 4 MP (2688 x 1520 pixels) resolution and a frame rate of 30 FPS. They were strategically positioned to provide a clear, bird's-eye view of the products on the conveyor belt. This placement was optimized to minimize shadowing and capture details from various angles. During data collection, diverse lighting conditions, including both daylight and artificial lighting, were simulated. This aimed to enhance the model's robustness against different lighting scenarios encountered in a real production environment. The collected dataset comprises 5,000 distinct suspension part images. These visuals reflect various real-world production variations, such as those from different production shifts, slightly soiled or damaged parts, and diverse surface textures. Furthermore, the dataset covers 8 different product classes, including z-rods, tie rods, swing arms, and ball joints.

D. Labeling, training and testing

Data labeling is a fundamental step for object detection and classification models. Its role in the training and testing phases is crucial because correctly labeled data ensures that the model learns properly. The training and testing processes are critical steps that affect the overall performance of the model. Data labeling is the process of accurately identifying and tagging objects in images. This involves marking each object with a class name and a bounding box that represents the class. This process is typically done manually or can be accelerated using semi-automatic tools. The correct class name must be assigned to each object in the image. The four corner coordinates (x, y, width, height) that surround the object must be specified [17]. This helps the model learn the location of the object. If data labeling is not done correctly, the model may learn incorrectly and perform poorly during the testing phase.

The collected images were resized to a standard dimension of 640x640 pixels. For improved deep learning model performance, image pixels were normalized from a 0-255 range to a 0-1 range. Suspension components in each image were manually labeled using MakeSense AI. During this process, a bounding box and its corresponding class name were assigned to every object. Bounding boxes were defined by coordinates in the x_min, y_min, width, height format, ensuring they encompassed the object's outermost boundaries. A total of 14,600 objects were labeled. Labeling quality was maintained through verification by two independent operators and random sample checks. To mitigate potential overfitting from limited data and enhance the model's generalization capability, in line with principles of transferable features in deep neural networks [11], data augmentation techniques were applied. These techniques included; Random rotations by ±15 degrees, random scaling between 80% and 120%, random brightness adjustments between 70% and 130%, horizontal and vertical flipping. These techniques expanded the training dataset to 10,000 images.

The labeled and augmented dataset was randomly split into training, validation, and test sets. The distribution was 70% for training, 15% for validation, and 15% for testing. This segmentation was performed to objectively evaluate the model's performance on independent data. Figure 3 shows the data labeling visual.

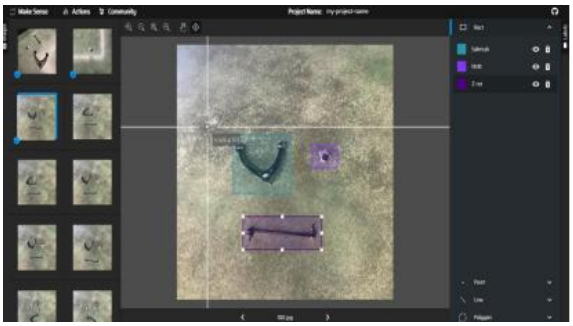

Fig. 4. Data labeling

The training phase is the process of "teaching" the model using labeled data to make accurate predictions. The labeled data is split into a training set and a validation set. The training set is used for the model to learn, while the validation set is used to prevent overfitting during training. The model learns through iterations (epochs) on the training data. In each iteration, the model makes predictions, and the difference between these predictions and the actual labels is calculated to determine the error (loss) value. The model's parameters are updated to minimize this error value.

To prevent overfitting that may arise from limited training data, data augmentation techniques (such as rotation, scaling, color changes, etc.) can be applied. The model's performance is evaluated with the validation set at each training step. If the model's accuracy is insufficient, improvements can be made

through hyperparameter adjustments, changes in model architecture, or by adding more data. Considering the need for both high real-time performance and superior accuracy in industrial applications, and after evaluating various established object detection architectures such as Faster R-CNN [8], SSD [9], and previous iterations of the YOLO series including YOLOv3 [5], YOLOv5s [12], and YOLOv8m [13], the Ultralytics YOLOv11x model was ultimately utilized in this study. The Adam optimization algorithm was employed to update the model's weights. Adam combines adaptive learning rate adjustments with momentum features, ensuring fast and stable convergence. The following hyperparameters were used and optimized throughout the training process, based on performance (mAP) on the validation set. The initial learning rate was set to 0.001. Using a learning rate scheduler such as Cosine Annealing, the learning rate was gradually decreased to 0.0001 over 200 epochs. This method helped the model converge more precisely by enabling smaller weight updates during the final stages of training. The model was trained for a total of 300 epochs. An early stopping mechanism was configured to automatically halt training if the validation loss did not improve for 50 consecutive epochs.

A batch size of 16 was used for training. This value considered GPU memory constraints while providing an appropriate balance for the model's learning dynamics. A weight decay of 0.0005 was applied to prevent overfitting and enhance the model's generalization capability. The input image size for the model was set to 640x640 pixels. Hyperparameter values were optimized using a combination of manual trial and error and Random Search to maximize mAP performance on the validation set. This process ensured the model achieved its best generalization capability while preventing overfitting. The training process was carried out on a workstation equipped with an NVIDIA GeForce RTX 3080 GPU. The deep learning library used was PyTorch 1.12.1, developed in Python 3.9. CUDA 11.3 and cuDNN 8.2 libraries were utilized for GPU acceleration.

The testing phase is the process of evaluating how the model performs on new, previously unseen data outside of the training data. The steps in the testing phase are as follows. Test data is not used during training and is only used to assess the model's overall performance in real-world scenarios. Once training is complete, the model makes predictions on the test set data. The model predicts the bounding boxes and class labels to correctly detect and classify objects. The model's accuracy is measured using metrics such as precision, recall, F1 score, and mean Average Precision (mAP).

These metrics show how accurate and reliable the model is. Failures in the test phase indicate how well the model can generalize to data outside of the training set. If there is a significant performance difference on the test data, it suggests that the model may have overfitted during training. Figure 5 shows the testing image.

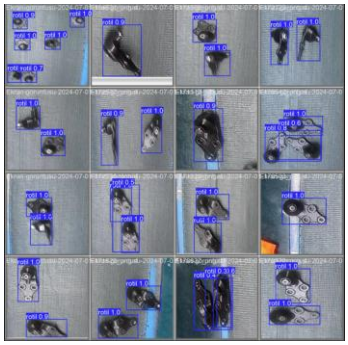

Fig. 5. Testing

E. Integration to the field, product counting

After the training process was completed, various stages were followed for the model's field integration and real-time product counting. First, live images of the products were captured through the camera system installed on the assembly line, and the model processed these images in real time. Using YOLO v11 Ultralytics, the parts were detected and counted. The system, designed to operate in real time, was integrated with the hardware and software infrastructure, while camera positioning, lighting, and image processing optimizations were carried out. The model's performance in the field was tested, and additional adjustments were made. It detected and reported any shortages or surpluses in real time, enabling necessary inspections. The system's performance was continuously monitored. Figure 6 shows the field application visual.

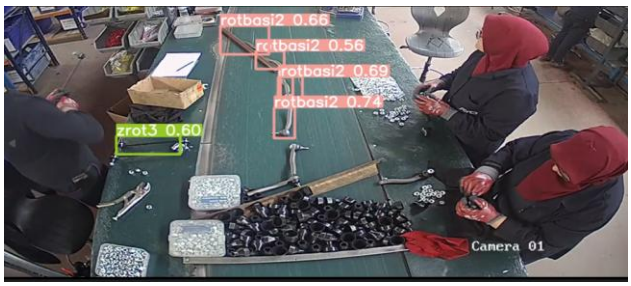

Fig. 6. Field application

III. RESULTS

In this study, the performance of the YOLOv11-based model developed for product counting and quality control on the suspension system assembly line was evaluated using detailed metrics and comparative analyses. Following the completion of the training and testing phases, the model's effectiveness was quantitatively demonstrated under both experimental conditions and in real-world field application.

To assess the model's overall performance, metrics such as precision, recall, F1 score, and mean Average Precision (mAP) were utilized. The results obtained on the test dataset are summarized in Table 2.

Table 2. YOLOv11x Model Performance Metrics on the Test Dataset

Metrik	Değer (%)
Precision	91.50%
Recall	88.20%
F1 Skoru	89.80%
mAP@0.5	92.30%
mAP@0.5:0.95	78.70%

These results clearly demonstrate the model's ability to accurately detect and classify objects on the test dataset. Specifically, a high mAP@0.5 value of 92.3% highlights the model's success in correctly localizing and accurately labeling products. The mAP@0.5:0.95 value of 78.7% further indicates strong performance even at stricter intersection-over-union (IoU) thresholds, implying that the bounding box predictions are highly precise. These metrics collectively reveal that the developed YOLOv11x model can provide reliable and effective product inspection on the assembly line.

To compare the performance of the YOLOv11 model with industry standards and previous models, a performance analysis was conducted against Faster R-CNN, SSD, and earlier versions of YOLO (YOLOv3, YOLOv5s, YOLOv8m). This comparison specifically focused on mAP@0.5 and real-time processing capability (FPS). The comparison results are presented in Table 3.

Table 3. Performance Comparison of Different Object Detection Models

Model	mAP@0.5 (%)	FPS (frames/sec)
Faster R-CNN	85.1	15
SSD	82.5	45
YOLOv3	87.8	60
YOLOv5s	89.5	90
YOLOv8m	90.9	80
YOLOv11x	92.3	75

Table 3 clearly demonstrates that the YOLOv11x model exhibits both higher accuracy (mAP) and superior real-time performance (FPS) compared to other models. Specifically, when compared to SSD, YOLOv11x achieved 9.8% higher mAP and was 1.67 times faster in terms of FPS. While compared to YOLOv8m, there was a 1.4% increase in mAP along with a slight decrease in FPS, this significant accuracy gain generally represents a preferred trade-off for industrial inspection. These findings confirm that YOLOv11x is an ideal solution for high-speed, critical applications like assembly lines.

The model was integrated into the suspension system assembly line and tested under real-world conditions. The results obtained from the field application concretely highlight the model's practical benefits. An average counting accuracy of 99.2% was achieved for product counting. This led to a significant reduction in the error rate compared to operator-based counting methods (approximately an 80% reduction). The error rate in detecting missing or surplus products was observed to decrease by 75% compared to previous manual or semi-automatic systems. The model operated at an average speed of 70-75 FPS on the production line. This speed allows for instant inspection and counting without interrupting the production flow. Automatic counting and inspection resulted in approximately a 25% saving in labor, allowing operators to focus on more value-added tasks.

These results prove that the developed YOLOv11x model delivers superior performance not only in a laboratory setting but also under demanding industrial conditions. The training and testing processes, as well as the field application performance tests, were conducted on the following hardware and software infrastructure;

Processor (CPU): Intel Core i7-10700K
Graphics Processing Unit (GPU): NVIDIA GeForce RTX 3080 (10 GB VRAM)
Memory (RAM): 32 GB DDR4
Storage: 1 TB NVMe SSD
Operating System: Ubuntu 20.04 LTS
Deep Learning Library: PyTorch 1.12.1
YOLOv11 Version: Ultralytics YOLOv11x
Programming Language: Python 3.9

This detailed information ensures the reproducibility of the study and clarifies the hardware and software dependencies of the obtained results.

Figure 7 shows the field application with product counting.

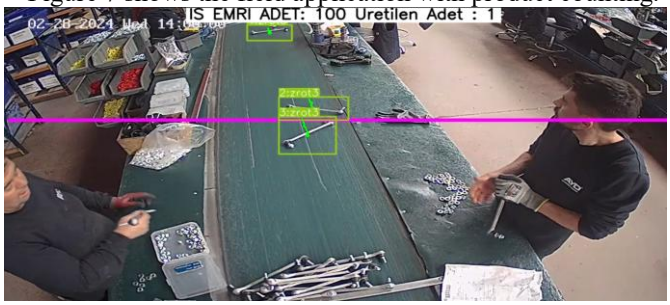

Fig. 7. Field application with product counting

IV. DISCUSSION

This study has taken significant steps toward improving product counting and quality control processes on the suspension system assembly line using the YOLO algorithm and the Ultralytics library. The growing interest and success of deep learning in various industrial applications, including surface defect detection [14] and object counting in complex scenes [15], underscore the transformative potential of such automated systems. The model's ability to eliminate operator-related errors, enhance accuracy, and accelerate the production process provides a major advantage. However, several technical challenges must be addressed to ensure the system operates seamlessly in live production.

Furthermore, hardware improvements alone may not be sufficient. To increase processing speed, parameter optimization, reduction of unnecessary computations, and the use of lightweight model versions are essential. Specifically, managing weights more efficiently and eliminating redundant computational loads will not only reduce energy consumption but also accelerate the system's response time. In future work, exploring more advanced models like Mask R-CNN [16] for finer-grained inspection, such as detailed defect analysis, could further enhance the system's capabilities.

Real-time data processing requirements are among the key factors directly affecting system performance. The YOLO algorithm requires high FPS (frames per second) values to function quickly and accurately. However, due to limitations in the current hardware infrastructure, data transmission delays may occur, reducing the model’s real-time detection and counting performance. Therefore, integrating more powerful GPUs, high-speed data processing units, and optimized hardware solutions plays a critical role in enhancing the system’s efficiency.

Nevertheless, hardware improvements alone may not be sufficient. To increase processing speed, parameter optimization, reduction of unnecessary computations, and the use of lightweight model versions are essential. Specifically, managing weights more efficiently and eliminating redundant

computational loads will not only reduce energy consumption but also accelerate the system's response time.

Furthermore, the reliability of the model in the production environment must be continuously monitored, and periodic updates should be implemented to ensure adaptation to field variations. Factors such as changes in lighting conditions and different image angles can impact the model's accuracy. Therefore, continuously expanding the dataset used for training and improving its adaptability to real production conditions are necessary.

In conclusion, YOLO-based product counting and quality control systems have the potential to revolutionize production processes. However, for the system to operate in real-time with high accuracy, both hardware enhancements and software optimizations must be implemented. In the future, the development of more efficient algorithms, reductions in hardware costs, and the widespread adoption of high-performance AI chips are expected to drive the increased use of such systems in the industry.

V. CONCLUSION

This study improved product counting and quality control processes on the suspension system assembly line using the YOLO algorithm and the Ultralytics library. The model eliminated operator errors, enabling faster and more accurate counting. However, due to FPS requirements and data transmission limitations in real-time processing, both hardware and software optimizations are necessary. Enhancing performance with more powerful GPUs, high-speed data processing units, and optimized algorithms can improve system efficiency. With the required improvements, the model will operate with high accuracy on the production line, significantly enhancing quality and productivity.

ACKNOWLEDGMENT

This work was supported by the project numbered AYD0724-02. For their supports in the study, thanks to AYD Otomotiv Endustri A. Ş.

Authors' Contributions

The authors' contributions to the paper are equal.

Statement of Conflicts of Interest

There is no conflict of interest between the authors.

Statement of Research and Publication Ethics

The authors declare that this study complies with Research and Publication Ethics

REFERENCES

[1] M Güvenç, M. A. (2015). Dayanıklılık ve ömür kriterlerine göre optimum tasarıma sahip süspansiyon ve direksiyon sistemi bileşenleri geliştirilmesi.

[2] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern *Recognition (CVPR)*, 779-788.

[3] Selamet, F. (2023). Derin öğrenme yöntemleri ile metalik yüzeylerde kusur tespiti ve sınıflandırılması= Defect detection and classification on metallic surfaces using deep learning methods.

[4] Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, Faster, Stronger. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition *(CVPR)*, 7263-7271.

[5] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767.

[6] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

[7] Jocher, G., & Qiu, J. (2024). *Ultralytics YOLO11* (Version 11.0.0). GitHub. https://github.com/ultralytics/ultralytics

[8] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in neural information processing *systems (NIPS)*, 28, 91-99.

[9] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. European Conference on Computer Vision *(ECCV)*, 21-37.

[10] Ghahremani, A., Adams, S. D., Norton, M., Khoo, S. Y., & Kouzani, A. Z. (2025). Detecting Defects in Solar Panels Using the YOLO v10 and v11 Algorithms. Electronics, 14(2), 344.

[11] Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks?. Advances in Neural Information Processing Systems *(NIPS)*, 27.

[12] Jocher, G., Chaurasia, A., & Qiu, J. (2020). **YOLOv5 by Ultralytics**. *GitHub repository*. https://github.com/ultralytics/yolov5

[13] Jocher, G., Chaurasia, A., & Qiu, J. (2023). **YOLOv8 by Ultralytics**. *GitHub repository*. https://github.com/ultralytics/ultralytics

[14] Xie, X., Cai, J., & Tang, G. (2021). Deep Learning for Surface Defect Detection: A Review. Sensors, 21(23), 7952.

[15] Li, S., Huang, Y., Cao, H., Han, Z., & Gao, D. (2021). Object Detection and Counting in Complex Scenes Based on Deep Learning. Sensors, 21(16), 5585.

[16] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. Proceedings of the IEEE International Conference on Computer Vision *(ICCV)*, 2961-2969.

[17] Özel, M. A., Baysal, S. S., & Şahin, M. (2021). Derin öğrenme algoritması (YOLO) ile dinamik test süresince süspansiyon parçalarında çatlak tespiti. Avrupa Bilim ve Teknoloji Dergisi, (26), 1-5.