sensors

MDPI

Article

Application of Machine Vision Techniques in Low-Cost Devices to Improve Efficiency in Precision Farming

Juan Felipe Jaramillo-Hernández [1,2,*], Vicente Julian [1,2], Cedric Marco-Detchart [1] and Jaime Andrés Rincón [3]

[1] Valencian Research Institute for Artificial Intelligence, Universitat Politècnica de València (UPV), Camí de Vera s/n, 46022 Valencia, Spain; vjulian@upv.es (V.J.); cedmarde@upv.es (C.M.-D.)
[2] Valencian Graduate School and Research Network of Artificial Intelligence (VALGRAI), Universitat Politècnica de València, Camí de Vera s/n, 46022 Valencia, Spain
[3] Departamento de Digitalización, Escuela Politécnica Superior, Universidad de Burgos, 09006 Miranda de Ebro, Spain; jarincon@ubu.es
* Correspondence: jfjarher@posgrado.upv.es

Abstract: In the context of recent technological advancements driven by distributed work and open-source resources, computer vision stands out as an innovative force, transforming how machines interact with and comprehend the visual world around us. This work conceives, designs, implements, and operates a computer vision and artificial intelligence method for object detection with integrated depth estimation. With applications ranging from autonomous fruit-harvesting systems to phenotyping tasks, the proposed Depth Object Detector (DOD) is trained and evaluated using the Microsoft Common Objects in Context dataset and the MinneApple dataset for object and fruit detection, respectively. The DOD is benchmarked against current state-of-the-art models. The results demonstrate the proposed method's efficiency for operation on embedded systems, with a favorable balance between accuracy and speed, making it well suited for real-time applications on edge devices in the context of the Internet of things.

Keywords: computer vision; object detection; depth estimation; precision agriculture

Citation: Jaramillo-Hernández, J.F.; Julian, V.; Marco-Detchart, C.; Rincón, J.A. Application of Machine Vision Techniques in Low-Cost Devices to Improve Efficiency in Precision Farming. *Sensors* **2024**, *24*, 937. https://doi.org/10.3390/s24030937

Academic Editors: José Manuel Cadenas, María Carmen Garrido and Raquel Martínez España

Received: 28 December 2023
Revised: 18 January 2024
Accepted: 27 January 2024
Published: 31 January 2024

1. Introduction

In recent years, technologies such as artificial intelligence (AI) [1–3], the Internet of things (IoT) [4,5], electronics [6–8], and edge computing [9] have become essential for enhancing energy efficiency, autonomy, and sustainability in global agriculture systems. This is especially critical due to the challenges posed by exponential population growth [10], which affects food security and leads to complex sociocultural problems and precarious working conditions in fields in Spain [11].

Harvesting is a fundamental process in the food production chain involving the collection of ripe fruits for consumption or commercial purposes. Technological innovation in this process is pivotal for improving agricultural productivity, soil management, climate resilience, and environmental remediation [12]. Integrating complex data acquisition systems, processing algorithms, and control systems to develop automated harvesting platforms can address these challenges [13–18]. Moreover, high-resolution sensors and high-end computers can lead to expensive hardware expenses. However, levering software capabilities such as optimized deep learning algorithms trained with high-resolution data can significantly reduce hardware costs while preserving high-quality data acquisition in new precision agriculture solutions [19–21].

One of the crucial tasks in the automation of fruit harvesting is the efficient spatial localization of the fruits. State-of-the-art fruit detection relies on fully convolutional neural networks (CNNs) for an optimal speed-precision balance [22]. Furthermore, integrating depth estimation can strain processing resources, requiring either stereo systems [23–25], LIDAR sensors [26], or dedicated monocular networks [27–29]. In this sense, this work

presents a novel Depth Object Detector (DOD) method: a deep-learning-based lightweight object detection algorithm with monocular depth estimation for cost-effective systems and real-time applications.

The novelty of our approach consists in leveraging the state-of-the-art model You Only Look Once Version 8 (YOLOv8), proposed by Jocher et al. [30]. This involves the integration of a novel regression head designed to model depth estimation as a representative value for each object while concurrently optimizing the network's computational efficiency. Moreover, due to the absence of a public dataset incorporating fruit detection and depth information, we propose an initial solution by modifying conventional object detection datasets. This requires incorporating representative depth labels for each object using the state-of-the-art monocular depth estimation model MiDaS [31,32]. In future work, we expect to construct an integrated dataset with physical metrics to calibrate and enhance the performance of depth estimation.

The Microsoft Common Objects in Context (COCO) dataset [33] is used to assess the computational performance to validate the proposed architecture size. Regarding fruit detection, the MinneApple dataset [34] is chosen for its uniform images capturing apple orchards at a consistent relative distance from the camera point-of-view. Finally, a quantized version of the proposed method is evaluated on an embedded system, demonstrating its capabilities in terms of size and speed.

The rest of the paper is structured as follows: Section 2 presents the related work; Section 3 describes the proposed DOD method; Section 4 presents and analyzes the different tests against the state of the art; finally, some conclusions are presented in Section 5.

2. Related Work

Object detection is a fundamental technique in computer vision that enables computer systems to identify and locate objects within images or videos using advanced algorithms to analyze visual patterns and distinguish objects from the background in a scene. On the other hand, depth estimation is a fundamental technique in computer vision that involves calculating the depth of each pixel in an image. Traditionally, this task has been addressed by using the disparity in stereo or multiview images to provide a basis in pixel coordinates for triangulating the distance of each point from the virtual camera [35,36].

In recent years, thanks to scientific progress and recent attention, neural networks have been able to overcome the depth estimation challenge using datasets composed of monocular images and their depth maps as ground truth, thus representing the multi-dimensional relationships within the semantic context of the foreground objects and the background of a given scene to infer the depth of each pixel in the image [37–39].

Even further, it is possible to optimize the depth estimation by focusing solely on key objects through object detection techniques [24,26–28]. This strategy maximizes computational efficiency and optimizes processing time by prioritizing the foreground of the scene, making it an attractive option for real-time applications such as autonomous driving [40,41], robotics [42], surveillance [43], or assisted surgery [44].

2.1. Depth Estimation

The current state of the art of monocular depth estimation is defined by Ranft et al. [31,32] (2021, 2022) by their architecture based on vision transformers (ViTs) [45] in place of CNNs as the backbone of dense prediction tasks. The transformer has a global receptive field at a constant and relatively high resolution, allowing for more detailed and globally consistent predictions than fully convolutional networks, especially when a large quantity of training data is available. Therefore, Zhao et al. [46] (2023) present a visual perception architecture that leverages ViTs by taking advantage of the semantic information of a pretrained text-to-image diffusion model in visual perception tasks such as depth estimation.

Further, Peluso et al. [47] (2022) propose an efficient monocular depth estimation method for microcontrollers based on a lightweight CNN with a shallow pyramidal architecture. By using optimization strategies to perform calculations on 8-bit data and

mapping the high-level description of the network to low-level layers optimized for the target microcontroller architecture, experimental results show that it is possible to obtain depth estimates sufficiently accurate for objects with large overlap areas.

2.2. Object Detection

Among the deep learning architectures that mark the current state of the art of object detection, the open-source algorithm You Only Look Once (YOLO) introduced by Redmon et al. [48] (2015) has stood out for its balance between speed and precision thanks to its evolution through successive iterations that improve previous versions to overcome limitations and improve performance [49]. The YOLOv8 model, proposed by Jocher et al. [30] (2023), establishes the current state of the art in object detection for fully convolutional architectures regarding speed and accuracy.

On the other hand, the revolution of cross-attention models, such as ChatGPT [50], has marked a breakthrough in generative AI for text-to-text, text-to-image, and image-to-image tasks. As an outcome, Meta proposes its open-source SAM (Segment Anything) model by Kirillov et al. [51] (2023) for object detection and semantic segmentation. This architecture consists of a ViT-based image encoder and a cue-guided mask decoder. Chaoning et al. [52] (2023) propose MobileSAM, a more optimal and faster version than SAM, with the same features but fewer parameters, ideal for mobile applications.

2.3. Object Detection for Precision Agriculture

Häni et al. [34,53,54] (2019) present MinneApple, a new dataset to advance state-of-the-art fruit detection, segmentation, and counting in orchard environments, providing a large variety of high-resolution images of different apple tree species collected at the University of Minnesota's Horticultural Research Center (HRC) between June 2015 and September 2016. Additionally, they present a benchmark performance analysis for the tasks using different object detection model architectures based on regions with CNN features (R-CNN) [55] with ResNet50 [56] as the feature extraction backbone, along with their proposed Tiled Faster R-CNN architecture.

Xiang et al. [57] (2021) proposed a system for the detection of loose oil palm fruits using the Faster R-CNN architecture [58] on NVIDIA Jetson TX2 hardware. In their study, 500 images of loose fruits were collected from an oil palm farm in Bukit Bangkong, Selangor, during harvesting. The model achieved an accuracy of approximately 94% for an intersection over union (IoU) threshold equal to 0.5, demonstrating that the developed system was capable of detecting loose oil palm fruits accurately and had the potential to contribute to the development of an automatic fruit-harvesting system.

Nagaraju et al. [59] (2022) proposed a fruit recognition technique based on the YOLOv5 [60] that detects custard apples, pomegranates, and wax berries. They collected images of fruits in a natural environment and preprocessed them to create a private dataset. With a mean average precision (mAP) of 89.4% at the 0.5 IoU threshold, they demonstrated that their system had significant implications for autonomous fruit-harvesting systems in orchards.

Wu et al. [61] (2023) present a Normal Detection Matched Fruit Counting System (NDMFCS), employing YOLOv4-tiny [62] for object detection, abnormal fruit detection thresholding, and trunk tracking with identity assignment. Results from 10 video sets show significant improvements, with fruit detection precision rising from 89.1% to 93.3%, enhancing overall counting accuracy to 95.0%. NDMFCS demonstrates promise as a technical solution for precise fruit yield estimation in modern apple orchards.

2.4. Object Detection with Depth Integration

Wang et al. [27] (2021) presented a real-time object detection and depth estimation approach based on CNNs. For depth estimation, they introduced binocular vision into a monocular-vision-based disparity estimation network and used the epipolar constraint to improve prediction accuracy. Finally, they integrated the 2D location of the detected

object with the depth information to achieve real-time depth detection and estimation. The results demonstrated that the proposed approach obtains better results than conventional methods. However, computing was complex and expensive in terms of processing.

Lee et al. [24] (2022) present a simplified approximation of depth in stereoscopic image objects by quantifying depth values into a small number of representative values. This allows for the avoidance of complexity in calculations by estimating only a representative depth value for each object instance and not having to estimate the values of all the pixels that contain the object. Their results on the KITTI dataset [63] demonstrate that despite the low complexity, their approximation method significantly improves object detection performance.

Fan et al. [28] (2022) sought to improve the real-time performance of 3D reconstruction by proposing a novel approach to reduce the consumption of computational resources by extracting significant regions from depth maps by fusing 2D object detection and self-supervised monocular depth estimation.

Usman et al. [26] (2022) introduce a point-pixel fusion system for object detection and classification with depth information for an autonomous driving system. Specifically, they combine the points of a LIDAR sensor with a 2D image, which is processed by an object detection model that extracts regions of interest to determine the depth in the highlighted objects, thus discarding the rest of the LIDAR points and preserving only the regions of interest.

Within the context of this work, the method proposed by Lee et al. [24] for depth estimation is the most relevant to this study, as we propose to address the depth estimation by using a single representative value for each detected object instance on monocular images.

3. Proposed Method

In precision agriculture, our method is intended to be implemented in autonomous systems for crop harvesting or phenotyping tasks. As described in Figure 1, by the integration with low-cost embedded systems equipped with a digital camera as a photoelectric transducer to obtain 2D RGB images of crops, the output of our proposed method serves as a control signal for various tasks, such as fruit harvesting, fruit counting, disease detection, or other phenotype characteristics, such as the size, width, color, and age of the fruits.

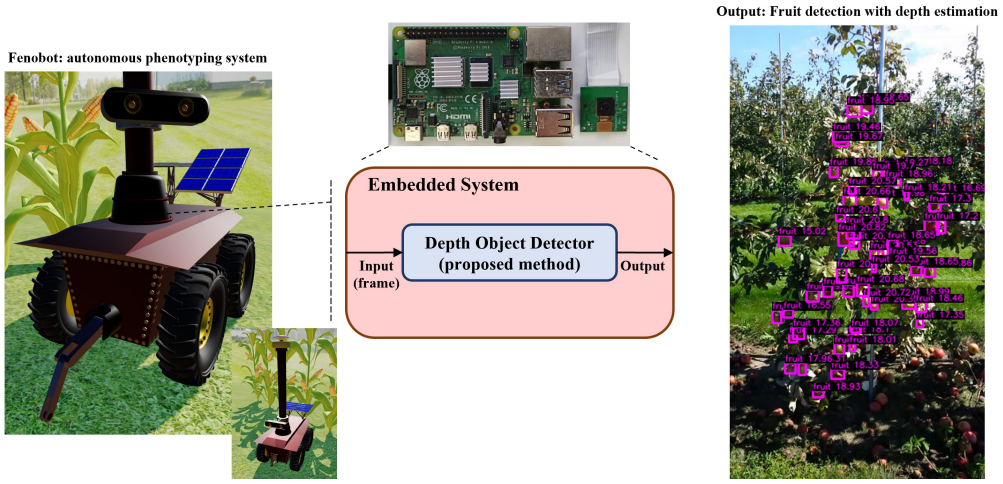


Figure 1. Integration diagram of the DOD method within an autonomous phenotyping robot. The Fenobot images are taken from our simulation. The orchard image is taken from MinneApple.

3.1. Architecture

The Depth Object Detector (DOD) architecture is a fully CNN inspired by the object detection architecture of YOLOv8 [30]. It is adapted to reduce the number of parameters for its application on low-cost or edge devices while integrating the depth estimation of

each bounding box (bbox) as an extra regression head on the output layers. As shown in Figure 2, the network architecture is mainly composed of the following sections:

Feature extraction: the feature extraction layers are primarily composed of the C2f (Conv-to-Features) block and the SPPF (fast spatial pyramid pooling) [64] block.

Neck: a neck designed as a feedback closed loop, inspired by Efficient Layer Aggregation Networks [65], which enhances the gradient distribution by the shortest and longest gradient path along the network.

Detection heads: the network comprises three output detection layers for different detection scales. Prediction heads represent each one of the following tasks: bbox regression, class classification, and depth value estimation, as our aggregation on the proposed method in this work.

The main blocks that compose the architecture are described as follows:

Conv: Convolution module composed of a 2D spatial convolution defined by a kernel size k, a stride size s, a padding size p, and input and output filter sizes c_{in} and c_{out}. The output of the convolution undergoes a 2D batch normalization [66] followed by the SiLU (sigmoid-weighted linear unit [67]) activation function:

$$\text{silu}(x) = x\left(\frac{1}{1+e^{-x}}\right) \tag{1}$$

Bottleneck: bottleneck block based on ResNet (2016) [56], consisting of two residual-connected convolutional feature extraction modules to mitigate the vanishing gradient problem [68].

C2f: Partial bottleneck block with two convolutional modules between depth-crossing stages n. Cross-Stage Partial Networks [69] inspired the C2f, which allows features to be partially preserved, communicated, and combined between different stages of the network. This produces better feature reuse and enables the network to capture more complex patterns and relationships, improving accuracy.

SPPF: A fast spatial pyramid pooling (SPP) [64] module that speeds up computation by pooling features into a fixed-size map. Sequential max-pooling operations aim to separate the most relevant features and significantly increase the receptive field in the context without decreasing the network's speed.

Detect: detection block composed of three decoupled heads composed of two convolutional modules and a final 2D convolution to predict for each prediction cell:

1. Bounding box regression: The output is an anchor-free [70] distribution of reg_max values for each distance left, right, top, bottom (l, r, t, b) relative to the center of the prediction cell. After linearly projecting the distributions into four-pixel coordinates in the inference process, the width and height of the bboxes are in the range:

$$x_1, y_1, x_2, y_2 = [0,\ 2(reg\_max - 1)max(stride)] \tag{2}$$

   given that:

$$x_2y_2 - x_1y_1 = rb + lt \tag{3}$$

   The distributional focal loss (DFL) function, proposed by Li et al. [71], introduces the hyperparameter reg_max to prevent the boxes from being too large or too small, ensuring the sensitivity of the predictions. For our proposed DOD method, $reg\_max = 4$ and $max(stride) = 32$.
2. Classification: The output is nc logits. For fruit detection, it is only considered one class $nc = 1$. In the case of the COCO dataset [33], the number of classes is $nc = 80$.
3. Depth: The output is one representative depth value as a dimensionless quantity; the closer the object from the foreground, the higher the depth value and vice versa. This quantity is described in detail in Section 3.4.

It is important to note that the training of the network adopts the task-aligned one-stage object detection [72] label assignment strategy to speed up the convergence by

selecting a top-*k* number of positive predictions for each ground truth based on a weighted classification and regression score.

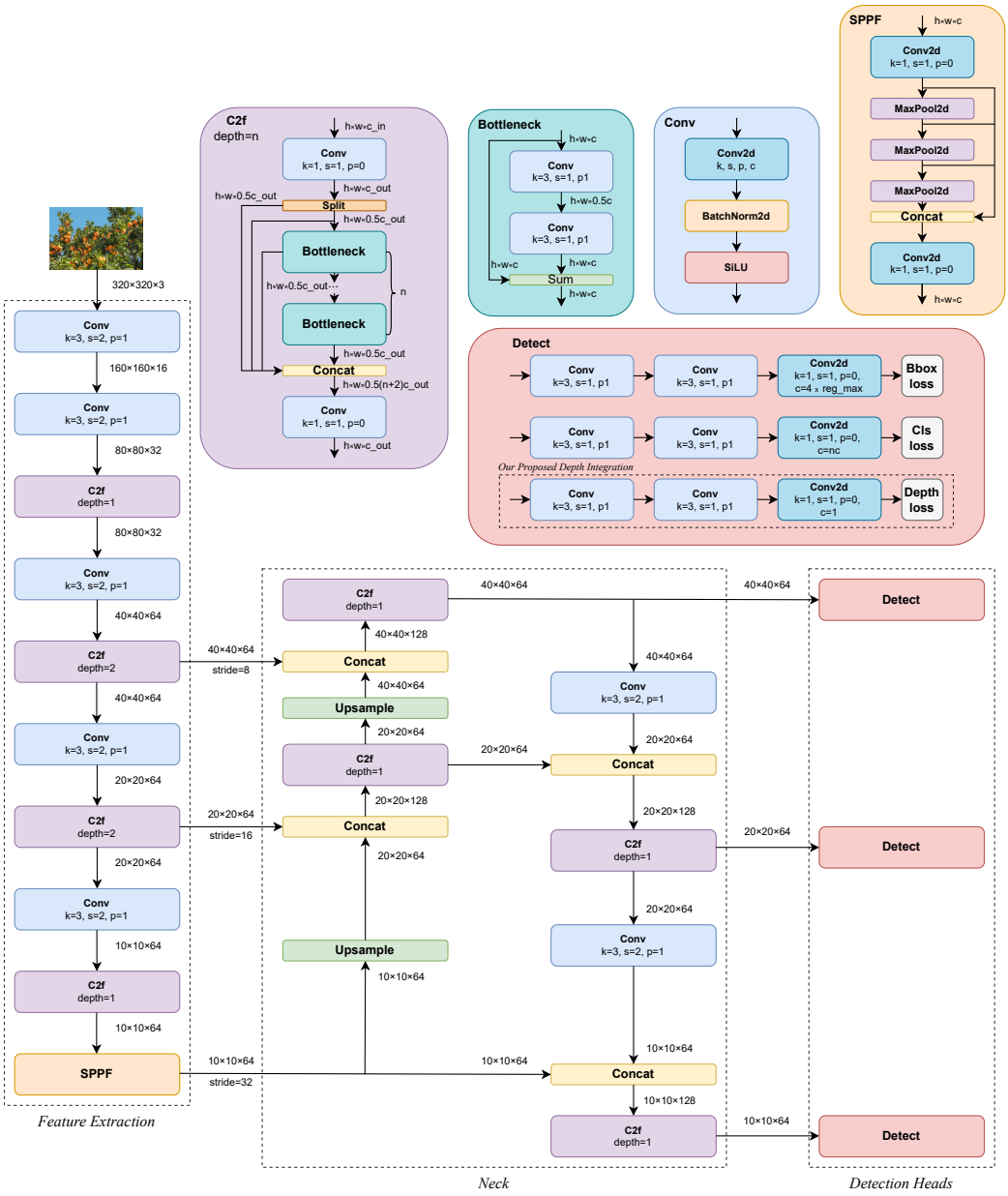


Figure 2. DOD architecture inspired by YOLOv8.

3.2. Inference Process

The detection heads generate a fixed number of predictions for each stride, regardless of the detected objects (see Figure 2). This tensor must be processed by an inference process (see Figure 3) to obtain a filtered result containing only the four-pixel coordinates, the highest confidence class, and the relative depth of each valid object detected in the input image.

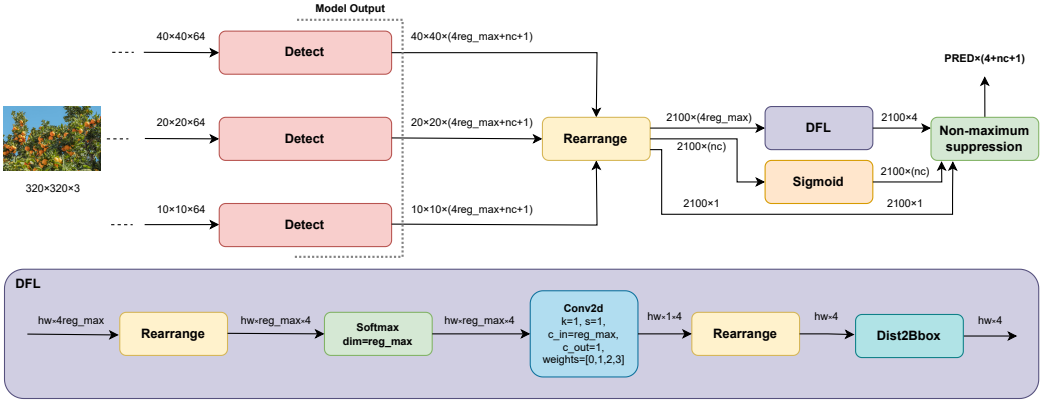


Figure 3. DOD's inference process.

The softmax function is applied to obtain a probability vector for each distance distribution. These vectors are then linearly transformed into the four distances through a 2D convolution without gradient with $c_{out} = 4$ filters and a kernel size $k = 1$, whose weights are preinitialized as $w = [0, 1, 2, 3]$.

Once the distances l, r, t, and b for each prediction are obtained, they are added to the pixel coordinates of the central point of each prediction cell and scaled according to the stride of the level where they were predicted, either 8, 16, or 32. This ultimately yields a bbox for each prediction with coordinates x, y relative to the dimensions of the input image.

Finally, a nonmaximum suppression (NMS) postprocessing technique is used to reduce the number of overlapping bboxes according to the Jaccard index, also known as IoU (see Equation (8)), which measures the degree of similarity between two boxes.

3.3. Loss Function

The weights of the network are adjusted by minimizing the mathematical formula described in Equation (4), which is the generalized loss function incorporating individual loss weights and a regularization term with weight decay ϕ. This is achieved using Equation (5) as the weight update rule with a learning rate η and an update velocity term with momentum β, as described in Equation (6). The specialized loss function is described in Equation (7) and is inspired and adapted from Reis et al.'s [73] description.

$$\mathcal{L}(\theta) = \frac{\lambda_{box}}{N_{pos}}\mathcal{L}_{box}(\theta) + \frac{\lambda_{cls}}{N_{pos}}\mathcal{L}_{cls}(\theta) + \frac{\lambda_{dfl}}{N_{pos}}\mathcal{L}_{dfl}(\theta) + \frac{\lambda_{depth}}{N_{pos}}\mathcal{L}_{depth}(\theta) + \phi\|\theta\|_2^2 \tag{4}$$

$$\theta^t = \theta^{t-1} - \eta V^t \tag{5}$$

$$V^t = \beta V^{t-1} + \nabla_\theta \mathcal{L}(\theta^{t-1}) \tag{6}$$

$$\begin{aligned}\mathcal{L} = {} & \frac{\lambda_{box}}{N_{pos}}\sum_{x,y}\mathbb{1}_{c^*_{x,y}}\left[1 - q_{x,y} + \frac{\left\|b_{x,y} - \hat{b}_{x,y}\right\|_2^2}{\rho^2} + \alpha_{x,y}v_{x,y}\right] + \frac{\lambda_{cls}}{N_{pos}}\sum_{x,y}\sum_{c\in clases} y_c log(\hat{y}_c) + (1-y_c)log(1-\hat{y}_c)) \\ & + \frac{\lambda_{dfl}}{N_{pos}}\sum_{x,y}\mathbb{1}_{c^*_{x,y}}\left[-(d_{(x,y)+1} - d_{x,y})log(\hat{d}_{x,y}) + (d_{x,y} - d_{(x,y)-1})log(\hat{d}_{(x,y)+1})\right] + \frac{\lambda_{depth}}{N_{pos}}\sum_{x,y}\mathbb{1}_{c^*_{x,y}}(z_{x,y} - \hat{z}_{x,y})^2\end{aligned} \tag{7}$$

$$q_{x,y} = IoU(x,y) = \frac{\left|\hat{\beta}_{x,y} \cap \beta_{x,y}\right|}{\left|\hat{\beta}_{x,y} \cup \beta_{x,y}\right|} \tag{8}$$

$$v_{x,y} = \frac{4}{\pi^2}\left(\arctan\left(\frac{w_{x,y}}{h_{x,y}}\right) - \arctan\left(\frac{\hat{w}_{x,y}}{\hat{h}_{x,y}}\right)\right)^2 \tag{9}$$

$$\alpha_{x,y} = \frac{v}{1 - q_{x,y}} \tag{10}$$

$$\hat{y}_c = \sigma(\cdot) \tag{11}$$

$$\hat{d}_{x,y} = softmax(\cdot) \tag{12}$$

where:

- N_{pos} is the total number of cells containing an object (positive predictions).
- $\mathbb{1}_{c^*_{x,y}}$ is the indicator function for cells with detected objects.
- $q_{x,y}$ is the IoU between predicted and ground-truth bboxes (Equation (8)).
- $\beta_{x,y}$ is a tuple $(x_{coord}, y_{coord}, width, height)$ representing a ground-truth bbox.
- $\hat{\beta}_{x,y}$ is a bbox predicted by a respective cell.
- $b_{x,y}$ is a tuple (x_{coord}, y_{coord}) representing the central point of a ground-truth bbox.
- $\hat{b}_{x,y}$ is the central point of a bbox predicted by a respective cell.
- ρ is the diagonal distance of the minimum bbox enclosing both a predicted and a ground-truth bbox.
- $v_{x,y}$ measures consistency in the aspect ratio between predicted and ground truth bboxes based on their width and height, respectively, $(w_{x,y}, h_{x,y})$, $(\hat{w}_{x,y}, \hat{h}_{x,y})$ (Equation (9)).
- $\alpha_{x,y}$ is a positive compensation where the overlap area factor has higher priority for regression, especially for nonoverlapping cases (Equation (10)).
- y_c is the ground-truth label for class c for each individual cell, regardless of whether an object is present.
- $\hat{y}_c$ is the predicted probability for class c for each individual cell, regardless of whether an object is present (Equation (11)).
- $d_{(x,y)+1}$ and $d_{(x,y)-1}$ are tuples (l, r, t, b) with values closest to the left and right of a ground-truth bbox whose tuple $(x_{coord}, y_{coord}, width, height)$ has been transformed to a relative distance from the center of a positive prediction cell.
- $\hat{d}_{x,y}$ are the probabilities of the predicted $4 \times reg\_max$ distribution by a cell containing an object.
- $z_{x,y}$ is the representative value of the relative depth to the background scene of the object in the ground-truth bbox.
- $\hat{z}_{x,y}$ is the representative value of the relative depth to the background scene of the detected object in the prediction cell.

The first term is the complete intersection over union (CIoU) loss proposed by Zheng et al. [74], which incorporates an improvement over the traditional Jaccard index Loss by considering three crucial geometric factors: the overlapped area, the distance between central points, and the aspect ratio between predicted and reference boxes. It penalizes inaccurate predictions severely.

The second term is the binary cross-entropy (BCE) as the classification loss, allowing each cell to predict one or more classes in the case of multilabel classification. This forces the model to learn the distribution of each class independently.

The third term is the distributional focal loss (DFL) proposed by Li et al. [71], which compels the network to quickly focus on values near the reference box by explicitly increasing the probabilities in the predicted $4 \times reg\_max$ distribution relative to the values closest (to the left and right) of the reference box.

The fourth term is the mean squared error (MSE), which is the loss for the depth estimation in our proposed method. This loss compels the depth integration as a regression problem for a representative depth value for each object detected.

3.4. Depth Integration

The used datasets for evaluation only contain labels for the object detection task. To address the lack of depth labels, we propose the usage of the state-of-the-art ViT model

MiDaS, proposed by Ranft et al. [31,32], to predict a representative depth of each label in datasets, as described in Algorithm 1.

Algorithm 1 Depth extraction.

Require: object detection dataset D, trained model for monocular depth estimation h
Ensure: dataset D' with representative depth for each reference object
for each dataset D **do**
    Predict depth map $z \leftarrow h(image)$
    for each object **do**
        Extract object bounding box in depth map $z_{obj} \leftarrow z \cap object$
        Calculate representative depth box value $z'_{obj} \leftarrow 0.5(\text{mean}(z_{obj}) + \max(z_{obj}))$
        Store label $obj_{depth} \leftarrow z'_{obj}$
    end for
end for

This algorithm aims to extract one representative depth value for each object as the mean between the mean and maximum depth intensity within the bounding-box pixels of the depth map synthesized by MiDaS. Moreover, this prediction output does not have a constant range or a physical metric estimation. A higher intensity value in the depth map means the object is closer to the foreground. In contrast, a lower intensity means the object is closer to the background, as seen in Figure 4.

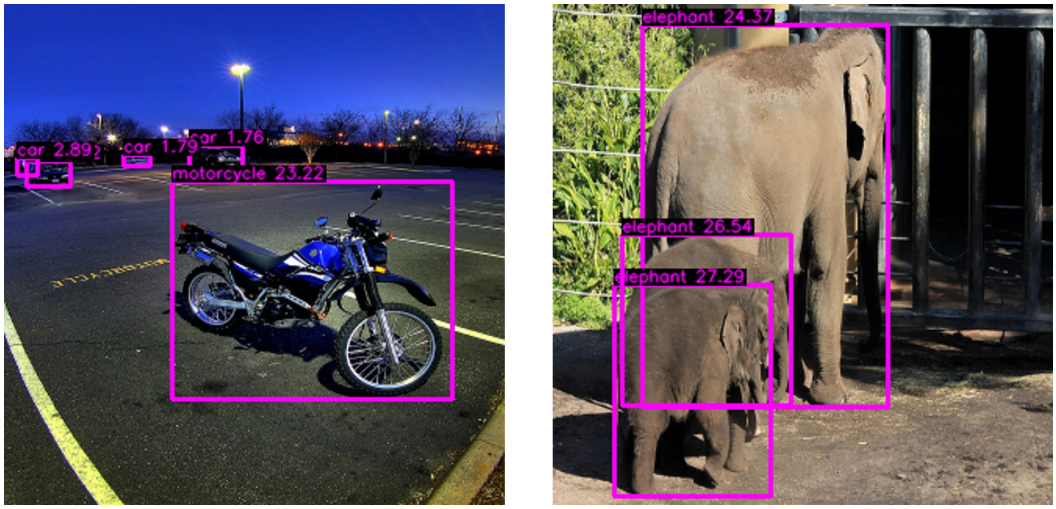


Figure 4. Depth integration into object detection. The higher the representative depth value, the closer the object is to the virtual camera. Images taken from COCO.

While this approach does not contain a feasible physical quantity for the depth values, it is intended to learn, as a first stage, the occlusion effects, disparity motion, and background segmentation from the detected objects considering its spatial context and semantic information, continuing the line of investigation on monocular depth estimation [37–39].

3.5. Data Augmentation

The data augmentation strategies used in training were Mosaic, as introduced in YOLOv8 training by merging four images into one to alleviate batch load and boost spatial context awareness; MixUp, proposed by Zhang et al. [75], which combines two training samples and labels to generate synthetic examples, thus facilitating regularization and improving generalization; ColorJitter, implemented with a 50% probability, which applies color transformations to enhance adaptability to diverse lighting conditions; And lastly, HorizontalFlip, with a 50% probability, which horizontally flips images and labels.

Collectively, these techniques aimed to improve model robustness and performance during training. Figure 5 describes some examples of the methods above.

Figure 5. Mosaic augmentation (first two from the left) and MixUp augmentation (last two from the left). Images taken from COCO.

4. Results

To evaluate the performance of our DOD method, we conducted the following experimental study:

1. We primarily benchmarked it against the state-of-the-art YOLOv8 model in a common object detection task using the COCO dataset. This direct comparison, conducted under identical conditions, provided insights into the performance disparities due to having nearly three times fewer parameters.
2. Then, we evaluated DOD's fruit detection performance on the MinneApple dataset and compared the results against the benchmark published by Häni et al. [34,53,54].
3. DOD trained only with MinneApple presented a generalization deficit for relatively large or medium-sized fruits. Therefore, we added some training samples from the Apples dataset [76] and compare the performance disparities on MinneApple.
4. Afterwards, we used MinneApple again to benchmark DOD's performance with a better generalization on the embedded system, a Raspberry Pi 4 board [77,78], using a 32-bit floating-point precision and a quantized 8-bit signed-integer precision. The quantization process aimed to balance model accuracy and reduced storage for computational requirements, making it suitable for deployment on microcontrollers or embedded systems with limited resources.
5. Finally, an ablation study was performed to analyze the behavior of the different main components of the DOD architecture.

The hardware and libraries used in these evaluations are listed as follows:

- CPU AMD Ryzen 7 5800H 3.20 GHz.
- GPU NVIDIA GeForce RTX 3070 Laptop GPU.
- CPU ARM Cortex-A72 1.5 GHz 64-bits Broadcom SoC BCM2711 on Raspberry Pi 4.
- NVIDIA driver version: 520.61.05.
- CUDA version: 11.8.89.
- PyTorch version: 2.0.1+cu117.
- Torchvision version: 0.15.2+cu117.
- OpenCV version: 4.7.0.
- Albumentation version: 1.3.0.

4.1. Common Objects Detection: COCO

Figure 6 illustrates the learning curve for each term of the loss function (see Equation (7)) during the training of DOD for object detection and depth estimation on COCO. The hyperparameters used in the training strategy were as follows:

- Adam optimizer [79] with its AMSGrad variant [80].
- Three warm-up epochs with a linear learning rate schedule from 0.0001 to 0.001.
- A cosine annealing factor from 0.001 to 0.0001.
- For object detection: 140 training epochs. MixUp: first 50 epochs. Mosaic: first 120 epochs.

- For depth estimation: 50 training epochs. MixUp: none. Mosaic: first 25 epochs.

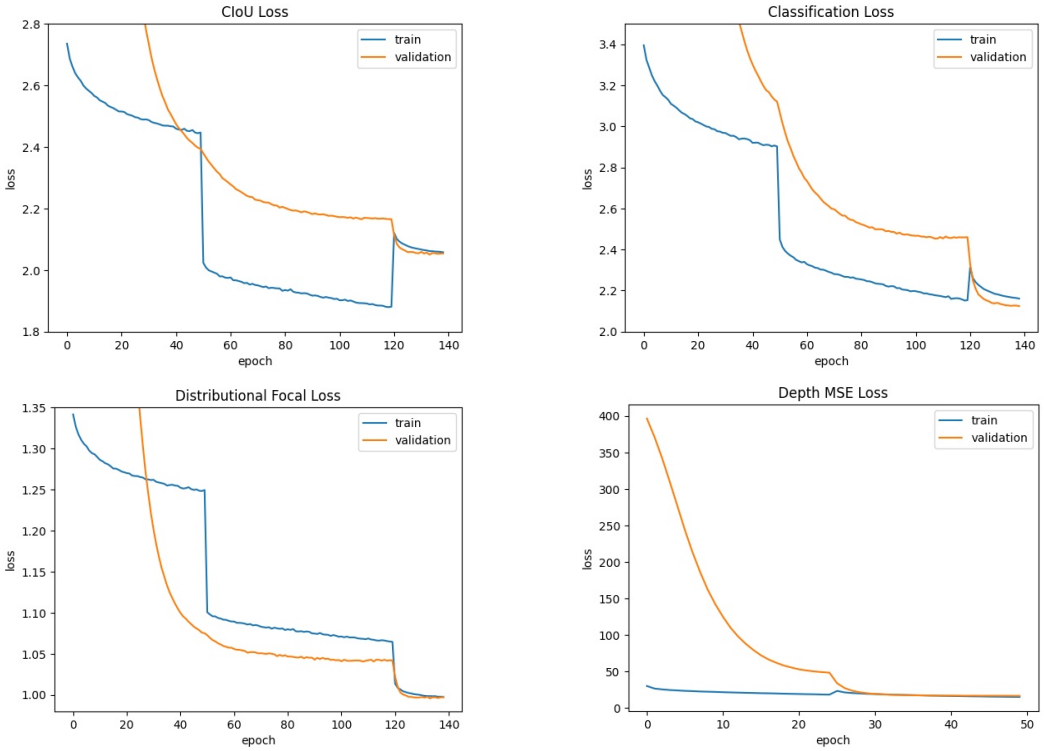


Figure 6. Learning curve for each term of the loss function (see Equation (7)) during the training of DOD on the COCO dataset [33]. Training duration: 55,440 s.

Table 1 presents the results obtained from the evaluation under the same conditions for our proposed DOD method with the lowest validation loss weights found in training and the pretrained YOLOv8n model on COCO. Figure 7 illustrates each detector's F1–confidence and precision–recall curves. Figure 8 contains some visual results on validation images.

Our proposed one-million-parameter network, trained on 80 classes, shows visually comparable performance to the baseline on the COCO dataset despite the anticipated low scores. While acknowledging a decrease in accuracy, this trade-off is justified by the higher frame rate capability achieved. In edge device scenarios, GPU utilization is crucial for efficient processing. Although the frame rate improvement might not seem significant in GPU-centric evaluations, its impact becomes pronounced in real-world edge device applications, where quicker inference enhances suitability for deployment, striking a balance between accuracy and responsiveness.

Table 1. Evaluation metrics obtained by our DOD proposed method and the state-of-the-art YOLOv8 trained and evaluated on COCO. P (%) : precision for the best confidence threshold. R (%): recall for the best confidence threshold. mAP 50 (%): mean average precision for $IoU = 0.5$. mAP 50–95 (%): mean average precision for $IoU \in [0.5, 0.95; 0.05]$. MSE depth: mean squared error of depth estimation. Vel. (fps): average inference time for four times the validation partition with a batch of unit size. Parameters (M): number of parameters. Size (MB): storage memory size.

Model	P (%)	R (%)	mAP 50 (%)	mAP 50–95 (%)	MSE Depth	Vel. CPU * (fps)	Vel. GPU * (fps)	Parameters (M)	Size (MB)
YOLOv8n	59.3	39.7	42.8	29.3	-	47.3	83.8	3.15	6.23
DOD	41.3	25.9	24.3	12.3	59.3	57.1	84.7	1.06	4.24

* CPU: AMD Ryzen 7 5800H 3.20 GHz. GPU: NVIDIA GeForce RTX 3070 Laptop GPU.

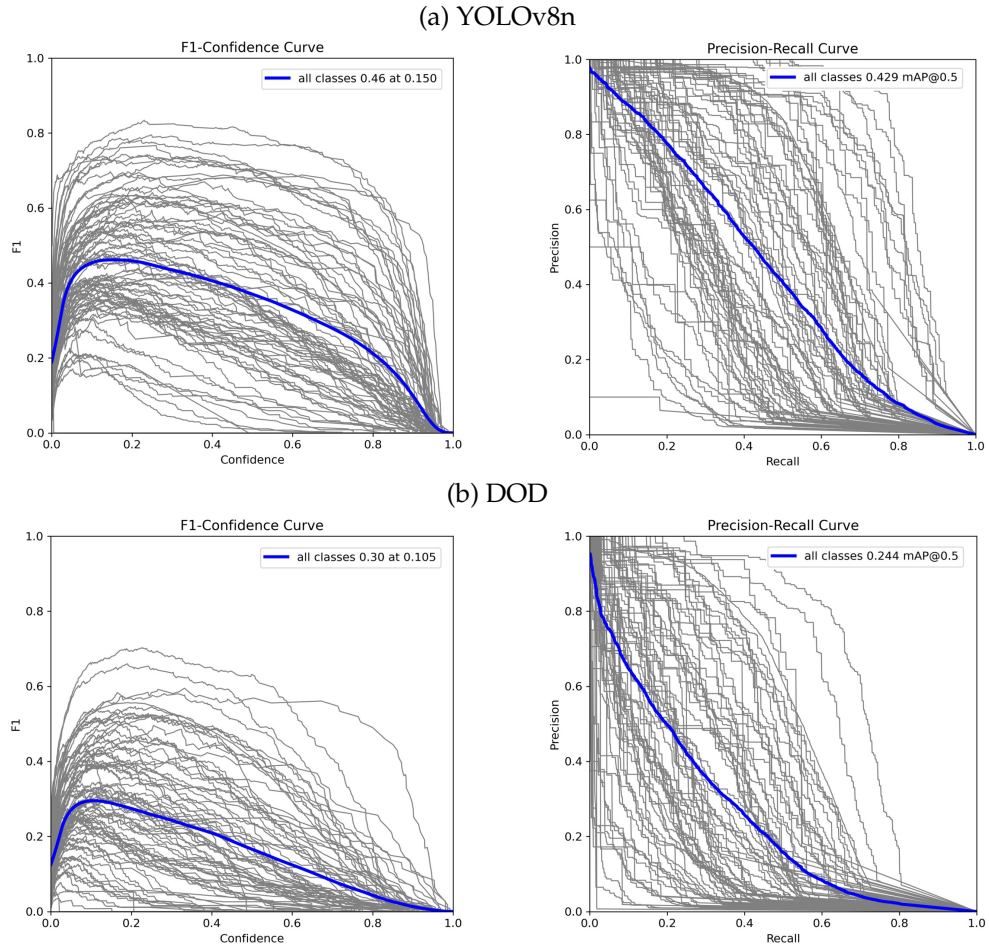


Figure 7. F1–confidence and precision–recall curves for YOLOv8n [30] and DOD evaluated on COCO. F1–confidence shows the harmonic mean of precision and recall for different confidence thresholds for $IoU = 0.5$. Precision–recall illustrates the trade-off between precision and recall for different confidence thresholds for $IoU = 0.5$.

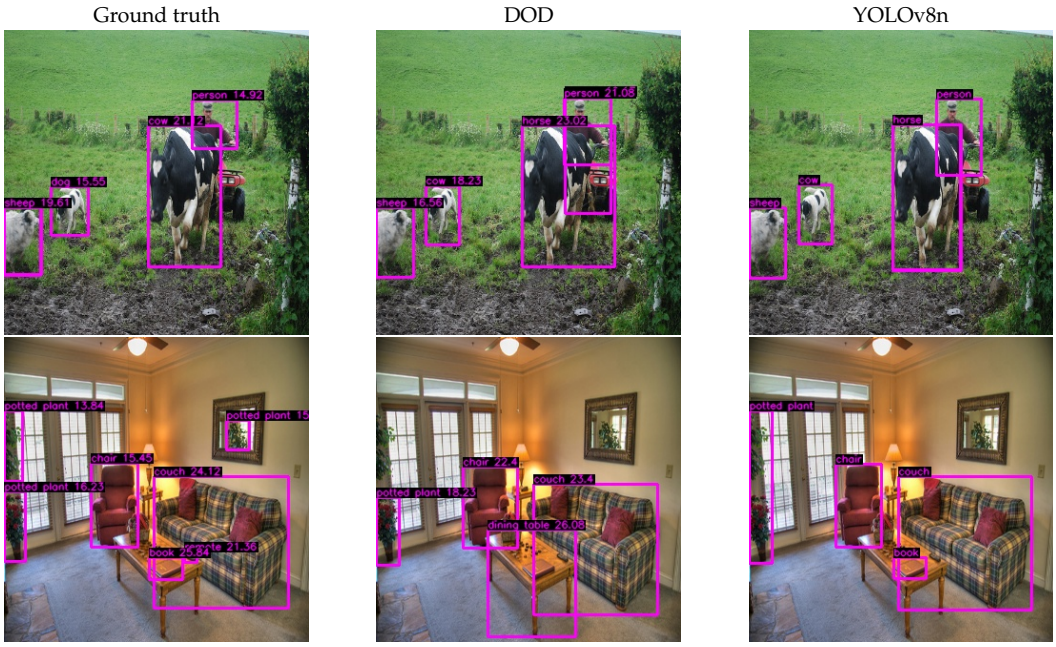


Figure 8. Inference results of DOD and YOLOv8n [30] models after applying nonmaximum suppression with a confidence threshold of 10% (best F1-score, see Figure 7) and an IoU threshold of

60%. The higher the representative depth value, the closer the object is to the virtual camera. Images taken from COCO.

4.2. Fruit Detection: MinneApple

Figure 9 illustrates the learning curve for each term of the loss function (see Equation (7)) during the training of DOD for object detection and depth estimation on MinneApple. The hyperparameters used in the training strategy were as follows:

- Adam optimizer [79] with its AMSGrad variant [80].
- Three warm-up epochs with a linear learning rate schedule from 0.0001 to 0.001.
- A cosine annealing factor from 0.001 to 0.0001.
- For object detection: 200 training epochs. MixUp: first 100 epochs. Mosaic: not applied.
- For depth estimation: 50 training epochs. MixUp: none. Mosaic: first 25 epochs.

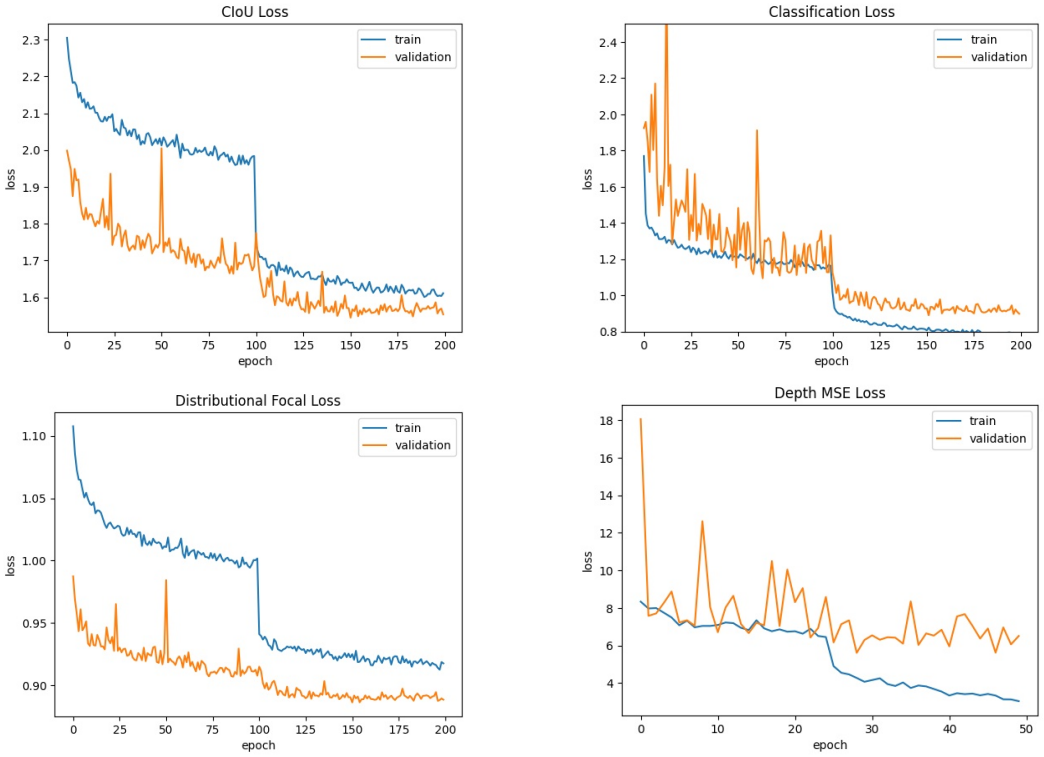


Figure 9. Learning curve for each term of the loss function (see Equation (7)) during the training of DOD on the MinneApple dataset. Training duration: 2426 s.

Table 2 presents the results obtained on MinneApple for DOD with the lowest validation loss weights found in training, along with the results published by Häni et al. [34,53,54] for the following R-CNN-based detection models: Tiled Faster R-CNN [34], Mask R-CNN [81] and Faster R-CNN [58]. Figure 10a illustrates our proposed method's F1–confidence and precision–recall curves. Figure 11 shows some inference results on validation images.

Furthermore, our proposed model performed well on the MinneApple dataset, achieving notable results with nearly forty times fewer parameters. This underscores the efficiency and effectiveness of incorporating state-of-the-art techniques in the network architecture to obtain remarkable results despite its lean parameter configuration.

Table 2. Evaluation metrics obtained on MinneApple by the proposed DOD method and the different architectures benchmark by Häni et al. [34,53,54] . P (%): precision for the best confidence threshold. R (%): recall for the best confidence threshold. mAP 50 (%): mean average precision for $IoU = 0.5$. mAP 50–95 (%): mean average precision for $IoU \in [0.5, 0.95; 0.05]$. MSE depth: mean squared error of depth estimation. Parameters (M): number of parameters.

Model	P (%)	R (%)	mAP 50 (%)	mAP 50–95 (%)	MSE Depth	Parameters (M)
DOD *	73.7	60.8	68.5	35.7	9.4	1.1
DOD **	68.6	58.1	62.4	31.9	8.5	1.1
TF-RCNN	-	-	63.9	34.1	-	≈41
F-RCNN	-	-	77.5	43.8	-	≈41
M-RCNN	-	-	76.3	43.4	-	≈63

* Trained only with MinneApple. ** Trained with MinneApple and Apples.

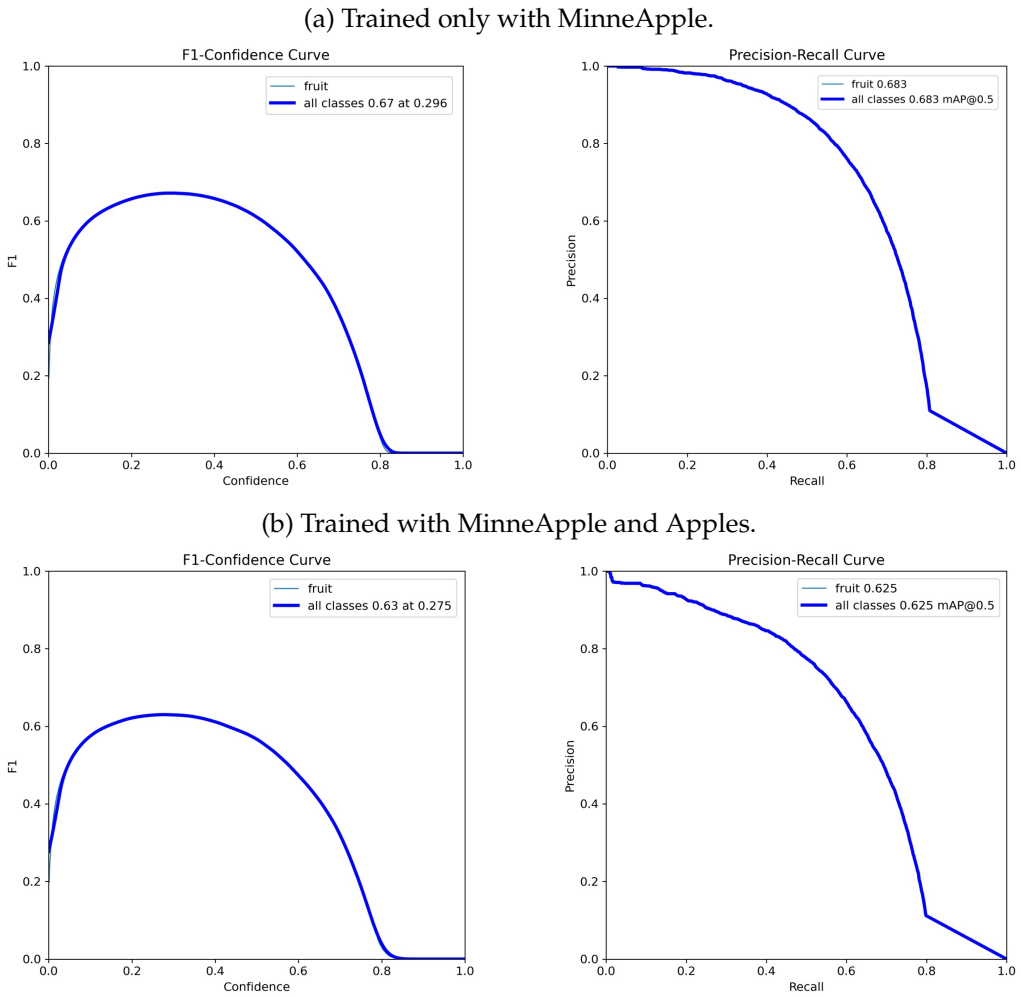


Figure 10. F1–confidence and precision–recall curves for DOD on MinneApple. F1–confidence shows the harmonic mean of precision and recall for different confidence thresholds for $IoU = 0.5$. Precision–recall illustrates the trade-off between precision and recall for different confidence thresholds for $IoU = 0.5$.

Figure 11. Inference results of DOD trained on MinneApple after applying nonmaximum suppression with a confidence threshold of 27% (best F1-score, see Figure 10) and an IoU threshold of 80%. Images taken from MinneApple.

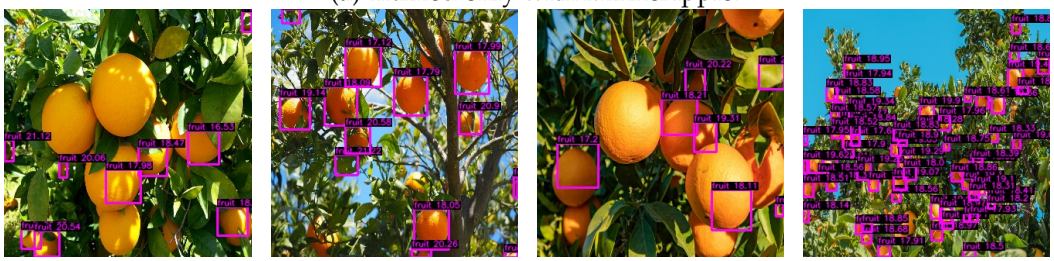
4.2.1. Improving Generalization

Figure 12a shows a generalization deficit for relatively large or medium-size fruits when training only with the MinneApple dataset. This is due to the dataset's homogeneity on its 670 training images, taken in orchard environments, and containing between 1 and 120 object instances per image.

Therefore, adding new training samples with different contexts and sizes can help alleviate this problem. For this reason, we used the Apples [76] dataset with 667 images containing between 1 and 29 apples from different context images taken from the web.

Table 2 presents the results obtained on MinneApple for the improved generalization version of DOD. Figure 10b illustrates its F1–confidence and precision–recall curves. Despite decreasing the detection scores, the better generalization version can now detect much larger fruits, as seen in Figure 12b. Furthermore, the new variety of sizes and spatial contexts in the training samples has improved the spatial awareness of the depth estimation heads, achieving a lower root-mean-square error than the DOD version trained with MinneApple alone.

(**a**) Trained only with MinneApple.



(**b**) Trained with MinneApple and Apples.

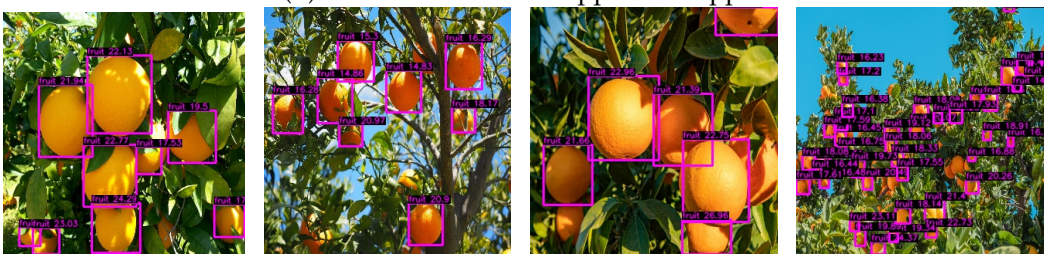


Figure 12. Inference results of DOD versions trained on MinneApple and Apples, respectively, after applying nonmaximum suppression with a confidence threshold of 20% and an IoU threshold of 80%. Images freely accessible from the web.

4.2.2. Quantization

The objective of the quantization process is to optimize the storage space required by the operations and weights of the DOD architecture to operate optimally on microcontrollers or embedded systems (see Figure 1). The methodology used in the quantization process is described as follows:

1. Initialize the DOD with the weights of the best-trained version.
2. Create a quantizable version of the DOD by specifying the operations to be quantized using Pytorch's Quant/DeQuant placement methods [82]. The library only supports quantizing the following operations: 2D convolution, batch normalization, linear layer, and rectified linear unit (ReLU) activation. The architecture proposed in this work (see Figure 2) uses the sigmoid linear unit (SiLU) activation function for its superior performance in the state of the art compared to ReLU. Therefore, the only quantizable operations in the proposed model are 2D convolutions and their batch normalization.
3. Copy the weights of all operations from the DOD in 32-bit floating-point precision to the quantizable model.
4. Apply the QNNPACK (Quantized Neural Networks PACKage) [83] quantization method integrated into Pytorch [82] to convert from 32-bit floating-point precision to 8-bit signed-integer precision.
5. Calibrate the quantized model using a small number of inference steps on the validation dataset. This is performed to identify the operating ranges of quantized operations and assign the most optimal variable type for storing each weight and operation.

Table 3 lists the results obtained on MinneApple for the improved generalization version of DOD, using 32-bit floating-point precision and 8-bit signed-integer precision. See Figure 13 for a visual comparison.

Although a few of the scores decrease, the int8 quantized version gains a slight frame rate increase on high-end devices. On the other hand, for low-end devices such as the Raspberry Pi 4, kernel implementations of the quantized operations have yet to be optimized to show a noticeable impact on frame rate capabilities [84]. Moreover, the memory footprint reduction is almost three times that of the fp32 version, thus impacting low-end devices.

Table 3. Evaluation metrics obtained by DOD evaluated on MinneApple. P (%): precision for the best confidence threshold. R (%): recall for the best confidence threshold. mAP 50 (%): mean average precision for $IoU = 0.5$. mAP 50–95 (%): mean average precision for $IoU \in [0.5, 0.95; 0.05]$. MSE depth: mean squared error of depth estimation. Vel. (fps): average inference time for four times the validation partition with a batch of unit size. Parameters (M): number of parameters. Size (MB): storage memory size.

Model	P (%)	R (%)	mAP 50 (%)	mAP 50–95 (%)	MSE Depth	Vel. AMD * (fps)	Vel. ARM ** (fps)	Parameters (M)	Size (MB)
DOD (fp32)	68.6	58.1	62.4	31.9	8.5	27.2	2.14	1.04	4.18
DOD (int8)	67.2	57.8	61.4	30.4	9.3	33.6	2.34	1.04	1.32

* CPU: AMD Ryzen 7 5800H 3.20 GHz. ** CPU: ARM Cortex-A72 1.5 GHz 64-bits Broadcom SoC BCM2711 on Raspberry Pi 4.

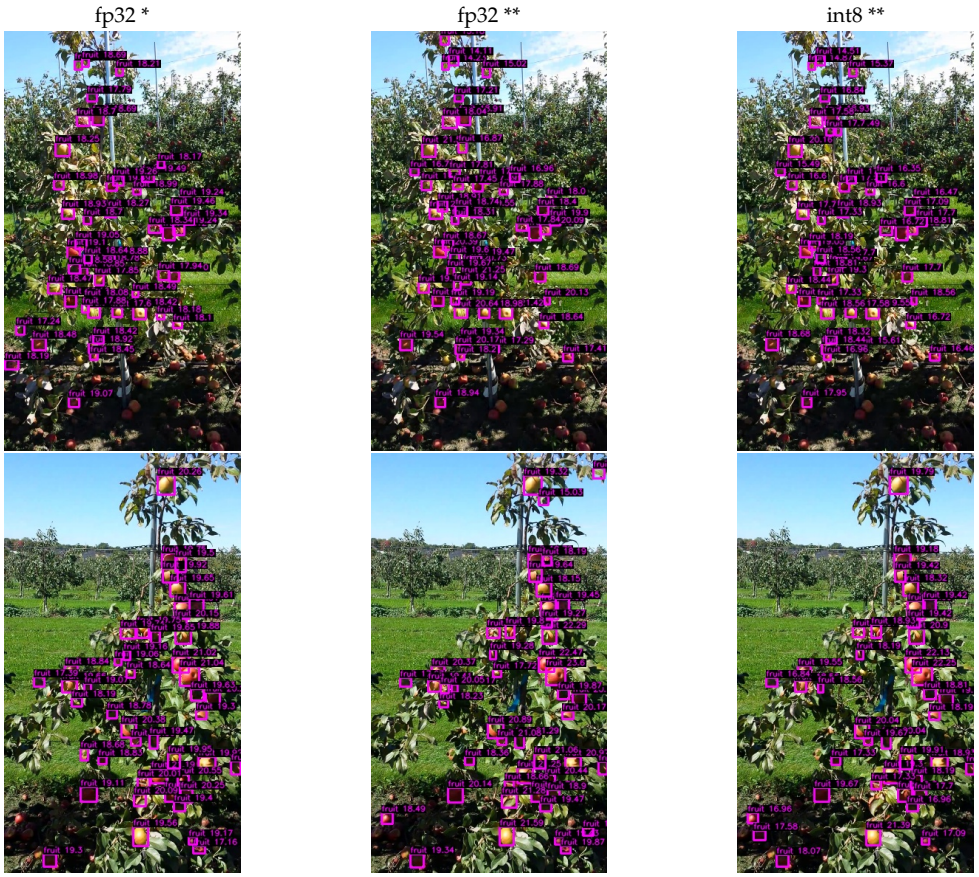


Figure 13. Inference results of different precision versions of the DOD method trained with MinneApple and Apples [76] datasets, respectively. Nonmaximum suppression with a confidence threshold of 20% and an IoU threshold of 80%. Images taken from MinneApple. * Trained only with MinneApple. ** Trained with MinneApple and Apples.

4.3. Ablation Study

Inspired by the ablation study of Meyes et al. [85] for VGG-19 [86], we conducted ablations of groups of similar filters with proportions of 10%, 25%, and 50% relative to the total number of filters in each Conv, C2f, and Detect blocks in the network (see Figure 2). Filter similarity within a group was based on the absolute Euclidean distance of the normalized filter weights. Ablations were performed by manually setting the weights and biases of all incoming connections for a filter to zero, effectively nullifying any activation

from that filter. The effect of ablations was evaluated by testing the detection and depth estimation performance of the network on MinneApple.

Figure 14 shows the results obtained in the ablation study of the DOD method with 32-bit floating-point precision and trained only in MinneApple. The results of the ablation with proportions of 10%, 25%, and 50% suggest that the initial blocks of the architecture (i.e., Conv1, Conv2, C2f1, Conv3, C2f2) play a crucial role in the information on which the network relies for fruit detection and depth estimation. This can be interpreted from the perspective that these blocks lay the foundations for the connections in the neck of the network, with Conv1 and C2f1 being particularly significant since their ablation represents the maximum performance loss. In the ablation with a proportion of 10%, it is possible to infer that Conv3 contains essential information for communication between the network's neck and the input image.

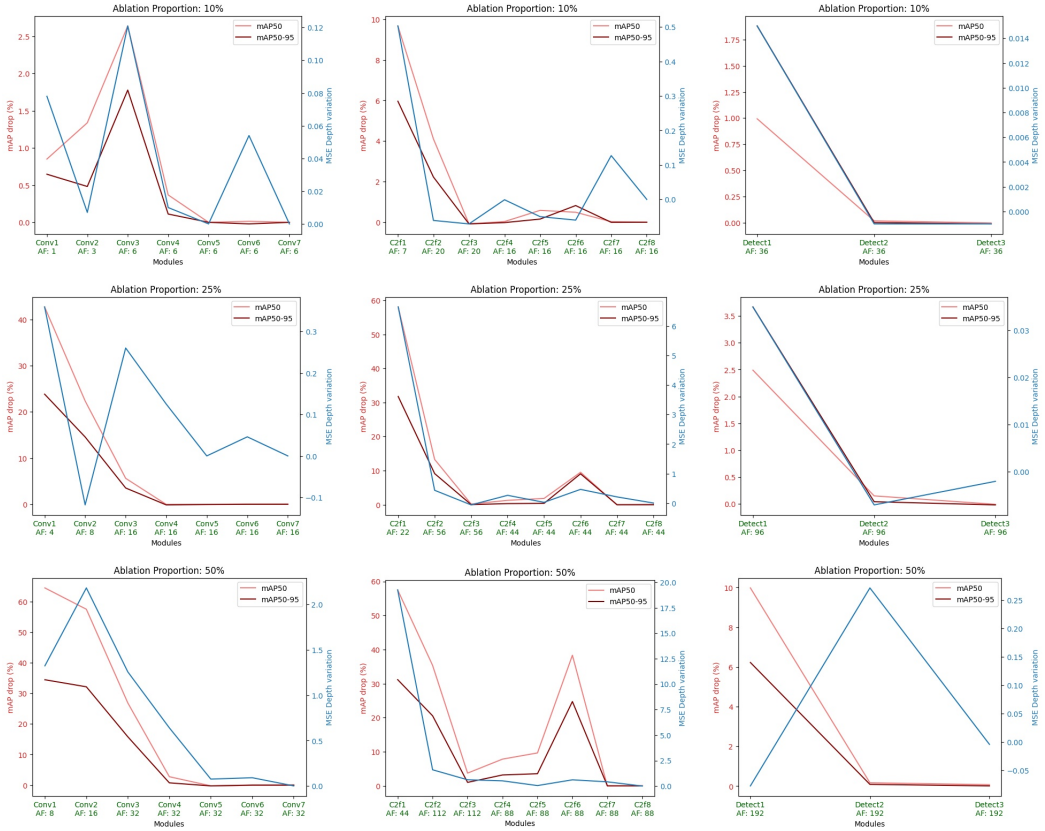


Figure 14. Effect on the evaluation metrics of ablations of different amounts (first row: 10% of layer filters; second row: 25% of layer filters; third row: 50% of layer filters) in all convolutional layers of the blocks (left: Conv block; center: C2f block; right: Detect block) that compose the DOD architecture (see Figure 3). AF is for "Ablated Filters".

On the other hand, from the overall ablation study, it can be deduced that the other blocks in the architecture (i.e., Conv4, C2f3, Conv5, C2f4, Conv6, C2f7, Conv7, C2f8), excluding C2f5 and C2f6, do not contribute any information to the network. However, this behavior is expected when considering the discussion in Section 4.2.1 regarding the nature of MinneApple images. When detecting such small fruits within an image, activation will only occur at the first prediction level (i.e., 40x40x64 features), responsible for detecting smaller objects in an input image due to its smaller stride. This argument is supported by the 50% proportion ablation, where C2f6, fed back by C2f5 and feeding Detect 1, contains crucial information for MinneApple fruit detection. Moreover, among the three blocks in the Detect module, Detect 1 is the only one impacting the network's performance, emphasizing the behavior described before and the determining role of the neck designed as a feedback

closed loop, such as the impact of Efficient Layer Aggregation Networks [65] on current state-of-the-art deep learning architectures.

5. Conclusions and Future Work

This work presents the Depth Object Detector (DOD) method as a novel computer vision method for object detection with depth estimation for real-time applications in low-cost embedded or microcontroller systems. The current state of the art in object detection inspired the proposed method's conception, design, implementation, and operation.

The detection capability of the proposed model was validated through an evaluation on the COCO dataset [33] and a comparison with the YOLOv8 model, which sets the current state of the art. Despite obtaining lower metrics, the proposed method achieved satisfactory visual results in this complex task with 80 classes, all with an architecture of approximately 1 million parameters.

On the other hand, performance in fruit detection was evaluated on the MinneApple dataset [34]. The results exceeded expectations by achieving higher metrics than the method proposed by Häni et al. [53,54], with at least 40 million parameters. The visual results and metrics validated the effectiveness and accuracy of DOD for this task.

Regarding depth estimation, the evaluation of the proposed method was limited to the MSE due to the lack of an analogous method to that evaluated in these tasks as well as datasets for detection with depth labels obtained through reliable physical measurements, at the time of the publication of this work.

In summary, the main contribution of our proposed method lies in integrating depth estimation as a regression head inside a lightweight object detection architecture. By using MiDaS to predict depth labels for conventional object detector datasets, the network can learn to identify occlusion effects and semantic background segmentation in parallel with object detection. This work marks a path for research into integrating these two techniques in monocular vision systems with low-cost hardware and efficient deep learning architectures.

In future work, it will be necessary to adjust the depth values obtained in detection by calibrating the model with measurements from physical instruments. This process will allow us to collect and analyze many experiments, which, in turn, can be used to retrain and adjust the accuracy in the depth estimation of the DOD.

Finally, the results indicate that the quantized DOD method is well suited for deployment on resource-constrained embedded systems, such as the Raspberry Pi. The reduced storage requirements and the efficient inference speed make it a viable solution for real-time applications with low-cost hardware for deployment in various applications that demand lightweight and efficient object detection with depth estimation. Moreover, the ablation study of the proposed method suggests that in future work, it will be possible to reduce large portions of the architecture network without compromising performance, as long as the nature and homogeneity of the target input images are considered.

The source code is available for reproducibility purposes at the GitHub repository: https://github.com/Jaramilloh/Depth-Object-Detector-DOD, accessed on 16 January 2024.

Author Contributions: Conceptualization, J.F.J.-H., V.J. and J.A.R.; methodology, J.F.J.-H., V.J. and J.A.R.; software, J.F.J.-H.; validation, J.F.J.-H.; formal analysis, J.F.J.-H.; investigation, J.F.J.-H.; resources, J.F.J.-H.; data curation, J.F.J.-H.; writing—original draft preparation, J.F.J.-H.; writing—review and editing, J.F.J.-H., V.J., J.A.R. and C.M.-D.; visualization, J.F.J.-H., V.J., J.A.R. and C.M.-D.; supervision, V.J. and J.A.R.; project administration, J.F.J.-H., V.J., J.A.R. and C.M.-D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported with grant PID2021-123673OB-C31, TED2021-131295B-C32 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe", PROMETEO grant CIPROM/2021/077 from the Conselleria de Innovación, Universidades, Ciencia y Sociedad Digital—Generalitat Valenciana and Early Research Project grant PAID-06-23 by the Vice Rectorate Office for Research from Universitat Politècnica de València (UPV).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Jahan, N.; Akilan, T.; Phalke, A.R. Machine Learning for Global Food Security: A Concise Overview. In Proceedings of the 2022 IEEE International Humanitarian Technology Conference (IHTC), Ottawa, ON, Canada, 2–4 December 2022; pp. 63–68. [CrossRef]
2. Kiruthiga, C.; Dharmarajan, K. Machine Learning in Soil Borne Diseases, Soil Data Analysis & Crop Yielding: A Review. In Proceedings of the 2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), Bengaluru, India, 27–28 January 2023; pp. 702–706. [CrossRef]
3. Kolhe, P.; Kalbande, K.; Deshmukh, A. Internet of Thing and Machine Learning Approach for Agricultural Application: A Review. In Proceedings of the 2022 10th International Conference on Emerging Trends in Engineering and Technology—Signal and Information Processing (ICETET-SIP-22), Nagpur, India, 29–30 April 2022; pp. 1–6. [CrossRef]
4. Basharat, A.; Mohamad, M.M.B. Security Challenges and Solutions for Internet of Things based Smart Agriculture: A Review. In Proceedings of the 2022 4th International Conference on Smart Sensors and Application (ICSSA), Kuala Lumpur, Malaysia, 26–28 July 2022; pp. 102–107. [CrossRef]
5. Ranganathan, V.; Kumar, P.; Kaur, U.; Li, S.H.; Chakraborty, T.; Chandra, R. Re-Inventing the Food Supply Chain with IoT: A Data-Driven Solution to Reduce Food Loss. *IEEE Internet Things Mag.* **2022**, *5*, 41–47. [CrossRef]
6. Bini, D.; Pamela, D.; Prince, S. Machine Vision and Machine Learning for Intelligent Agrobots: A review. In Proceedings of the 2020 5th International Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, India, 5–6 March 2020; pp. 12–16. [CrossRef]
7. Shahrooz, M.; Talaeizadeh, A.; Alasty, A. Agricultural Spraying Drones: Advantages and Disadvantages. In Proceedings of the 2020 Virtual Symposium in Plant Omics Sciences (OMICAS), Colombia, India, 23–27 November 2020; pp. 1–5. [CrossRef]
8. Sharma, M.; Hema, N. Comparison of Agricultural Drones and Challenges in Implementation: A Review. In Proceedings of the 2021 7th International Conference on Signal Processing and Communication (ICSC), Noida, India, 25–27 November 2021; pp. 26–30. [CrossRef]
9. Zhang, X.; Cao, Z.; Dong, W. Overview of Edge Computing in the Agricultural Internet of Things: Key Technologies, Applications, Challenges. *IEEE Access* **2020**, *8*, 141748–141761. [CrossRef]
10. United Nations Department of Economic and Social Affairs, Population Division. *World Population Prospects 2022: Summary of Results;* Technical Report; Population Division, Department of Economic and Social Affairs, United Nations: New York, NY, USA, 2022. Available online: https://www.un.org/development/desa/pd/sites/www.un.org.development.desa.pd/files/undesa_pd_2022_wpp_key-messages.pdf (accessed on 21 January 2024).
11. Briones-Vozmediano, E.; González-González, A. Explotación y precariedad sociolaboral, la realidad de las personas migrantes trabajadoras en agricultura en España. *Arch. Prevención Riesgos Laborales* **2022**, *25*, 18–24. [CrossRef] [PubMed]
12. FAO. "Digital Action" @ WSIS Forum 2023: FAO Takes Stock of Agrifood Systems Transformation for SDGs. 2023. Available online: https://www.fao.org/e-agriculture/news/digital-action (accessed on 21 January 2024).
13. Nanda, A.; Swain, K.K.; Reddy, K.S.; Agarwal, R. sTransporter: An Autonomous Robotics System for Collecting Fresh Fruit Crates for the betterment of the Post Harvest Handling Process. In Proceedings of the 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 March 2020; pp. 577–582. [CrossRef]
14. Arikapudi, R.; Vougioukas, S.G. Robotic Tree-Fruit Harvesting with Telescoping Arms: A Study of Linear Fruit Reachability Under Geometric Constraints. *IEEE Access* **2021**, *9*, 17114–17126. [CrossRef]
15. Elfferich, J.F.; Dodou, D.; Santina, C.D. Soft Robotic Grippers for Crop Handling or Harvesting: A Review. *IEEE Access* **2022**, *10*, 75428–75443. [CrossRef]
16. Qiu, A.; Young, C.; Gunderman, A.L.; Azizkhani, M.; Chen, Y.; Hu, A.P. Tendon-Driven Soft Robotic Gripper with Integrated Ripeness Sensing for Blackberry Harvesting. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 11831–11837. [CrossRef]
17. Mail, M.F.; Maja, J.M.; Marshall, M.; Cutulle, M.; Miller, G.; Barnes, E. Agricultural Harvesting Robot Concept Design and System Components: A Review. *AgriEngineering* **2023**, *5*, 777–800. [CrossRef]
18. Droukas, L.; Doulgeri, Z.; Tsakiridis, N.L.; Triantafyllou, D.; Kleitsiotis, I.; Mariolis, I.; Giakoumis, D.; Tzovaras, D.; Kateris, D.; Bochtis, D. A Survey of Robotic Harvesting Systems and Enabling Technologies. *J. Intell. Robot. Syst.* **2023**, *107*, 21. [CrossRef]
19. Dai, Q.; Cheng, X.; Qiao, Y.; Zhang, Y. Agricultural Pest Super-Resolution and Identification with Attention Enhanced Residual and Dense Fusion Generative and Adversarial Network. *IEEE Access* **2020**, *8*, 81943–81959. [CrossRef]
20. Yamamoto, K.; Togami, T.; Yamaguchi, N. Super-Resolution of Plant Disease Images for the Acceleration of Image-based Phenotyping and Vigor Diagnosis in Agriculture. *Sensors* **2017**, *17*, 2557. [CrossRef]
21. Liu, J.; Yu, S.; Liu, X.; Lu, G.; Xin, Z.; Yuan, J. Super-Resolution Semantic Segmentation of Droplet Deposition Image for Low-Cost Spraying Measurement. *Agriculture* **2024**, *14*, 106. [CrossRef]

22. Xiao, F.; Wang, H.; Xu, Y.; Zhang, R. Fruit Detection and Recognition Based on Deep Learning for Automatic Harvesting: An Overview and Review. *Agronomy* **2023**, *13*, 1625. [CrossRef]
23. Kang, H.; Zhou, H.; Chen, C. Visual Perception and Modeling for Autonomous Apple Harvesting. *IEEE Access* **2020**, *8*, 62151–62163. [CrossRef]
24. Lee, Y.; Lee, H.; Lee, E.; Kwon, H.; Bhattacharyya, S. Exploiting Simplified Depth Estimation for Stereo-based 2D Object Detection. In Proceedings of the 2022 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), Washington, DC, USA, 11–13 October 2022; pp. 1–6. [CrossRef]
25. Mirbod, O.; Choi, D.; Heinemann, P.H.; Marini, R.P.; He, L. On-tree apple fruit size estimation using stereo vision with deep learning-based occlusion handling. *Biosyst. Eng.* **2023**, *226*, 27–42. [CrossRef]
26. Usman, M.; Ling, Q. Point-pixel fusion for object detection and depth estimation. In Proceedings of the 2022 41st Chinese Control Conference (CCC), Heifei, China 25–27 July 2022; pp. 5458–5462. [CrossRef]
27. Wang, H.M.; Lin, H.Y.; Chang, C.C. Object Detection and Depth Estimation Approach Based on Deep Convolutional Neural Networks. *Sensors* **2021**, *21*, 4755. [CrossRef]
28. Fan, C.; Yin, Z.; Huang, X.; Li, M.; Wang, X.; Li, H. Faster 3D Reconstruction by Fusing 2D Object Detection and Self-Supervised Monocular Depth Estimation. In Proceedings of the 2022 11th International Conference of Information and Communication Technology (ICTech)), Wuhan, China, 4–6 February 2022; pp. 492–497. [CrossRef]
29. Coll-Ribes, G.; Torres-Rodríguez, I.J.; Grau, A.; Guerra, E.; Sanfeliu, A. Accurate detection and depth estimation of table grapes and peduncles for robot harvesting, combining monocular depth estimation and CNN methods. *Comput. Electron. Agric.* **2023**, *215*, 108362. [CrossRef]
30. Jocher, G.; Chaurasia, A.; Qiu, J. Ultralytics YOLOv8. 2023. Available online: https://docs.ultralytics.com (accessed on 21 January 2024).
31. Ranftl, R.; Bochkovskiy, A.; Koltun, V. Vision Transformers for Dense Prediction. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021.
32. Ranftl, R.; Lasinger, K.; Hafner, D.; Schindler, K.; Koltun, V. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 1623–1637. [CrossRef] [PubMed]
33. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In Proceedings of the Computer Vision—ECCV 2014, Zurich, Switzerland, 6–12 September 2014; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer: Cham, Switzerland, 2014; pp. 740–755.
34. Häni, N.; Roy, P.; Isler, V. MinneApple: A Benchmark Dataset for Apple Detection and Segmentation. *IEEE Robot. Autom. Lett.* **2020**, *5*, 852–858. [CrossRef]
35. Scharstein, D.; Szeliski, R.; Zabih, R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In Proceedings of the IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001), Kauai, HI, USA, 9–10 December 2001; pp. 131–140. [CrossRef]
36. Szeliski, R. *Computer Vision —Algorithms and Applications*, 2nd ed.; Texts in Computer Science; Springer: Berlin/Heidelberg, Germany, 2022. [CrossRef]
37. Bazrafkan, S.; Javidnia, H.; Lemley, J.; Corcoran, P. Semiparallel deep neural network hybrid architecture: First application on depth from monocular camera. *J. Electron. Imaging* **2018**, *27*, 043041. [CrossRef]
38. Kuznietsov, Y.; Stückler, J.; Leibe, B. Semi-Supervised Deep Learning for Monocular Depth Map Prediction. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2215–2223. [CrossRef]
39. Masoumian, A.; Rashwan, H.A.; Cristiano, J.; Asif, M.S.; Puig, D. Monocular Depth Estimation Using Deep Learning: A Review. *Sensors* **2022**, *22*, 5353. [CrossRef]
40. Park, C.; Kim, H.; Kim, M.; Sung, J.; Paik, J. Monocular 3D Object Detection of Moving Objects Using Random Sampling and Deep Layer Aggregation. In Proceedings of the 2023 IEEE International Conference on Consumer Electronics (ICCE), Berlin, Germany, 2–5 September 2023; pp. 1–2. [CrossRef]
41. Wang, H.M.; Lin, H.Y. A Real-Time Forward Collision Warning Technique Incorporating Detection and Depth Estimation Networks. In Proceedings of the 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Toronto, ON, Canada, 11–14 October 2020; pp. 1966–1971. [CrossRef]
42. Kato, H.; Nagata, F.; Murakami, Y.; Koya, K. Partial Depth Estimation with Single Image Using YOLO and CNN for Robot Arm Control. In Proceedings of the 2022 IEEE International Conference on Mechatronics and Automation (ICMA), Guilin, China, 7–9 August 2022; pp. 1727–1731. [CrossRef]
43. Pogaru, S.; Bose, A.; Elliott, D.; O'Keefe, J. Multiple Object Association Incorporating Object Tracking, Depth, and Velocity Analysis on 2D Videos. In Proceedings of the SoutheastCon 2021, Atlanta, GA, USA, 10–13 March 2021; pp. 1–6. [CrossRef]
44. Xu, C.; Huang, B.; Elson, D.S. Self-Supervised Monocular Depth Estimation with 3-D Displacement Module for Laparoscopic Images. *IEEE Trans. Med. Robot. Bionics* **2022**, *4*, 331–334. [CrossRef]
45. Liu, Y.; Zhang, Y.; Wang, Y.; Hou, F.; Yuan, J.; Tian, J.; Zhang, Y.; Shi, Z.; Fan, J.; He, Z. A Survey of Visual Transformers. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, 1–21. [CrossRef]
46. Zhao, W.; Rao, Y.; Liu, Z.; Liu, B.; Zhou, J.; Lu, J. Unleashing Text-to-Image Diffusion Models for Visual Perception. *arXiv* **2023**. http://arxiv.org/abs/2303.02153.

47. Peluso, V.; Cipolletta, A.; Calimera, A.; Poggi, M.; Tosi, F.; Aleotti, F.; Mattoccia, S. Monocular Depth Perception on Microcontrollers for Edge Applications. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 1524–1536. [CrossRef]
48. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788. [CrossRef]
49. Terven, J.; Cordova-Esparza, D. A Comprehensive Review of YOLO: From YOLOv1 and beyond. *arXiv* **2023**. http://arxiv.org/abs/2304.00501.
50. Zhang, C.; Zhang, C.; Li, C.; Qiao, Y.; Zheng, S.; Dam, S.K.; Zhang, M.; Kim, J.U.; Kim, S.T.; Choi, J.; et al. One Small Step for Generative AI, One Giant Leap for AGI: A Complete Survey on ChatGPT in AIGC Era. *arXiv* **2023**. http://arxiv.org/abs/2304.06488.
51. Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A.C.; Lo, W.Y.; et al. Segment Anything. *arXiv* **2023**. http://arxiv.org/abs/2304.02643.
52. Zhang, C.; Han, D.; Qiao, Y.; Kim, J.U.; Bae, S.H.; Lee, S.; Hong, C.S. Faster Segment Anything: Towards Lightweight SAM for Mobile Applications. *arXiv* **2023**. http://arxiv.org/abs/2306.14289.
53. Häni, N.; Roy, P.; Isler, V. A comparative study of fruit detection and counting methods for yield mapping in apple orchards. *J. Field Robot.* **2019**, *37*, 263–282. [CrossRef]
54. Häni, N.; Roy, P.; Isler, V. Apple Counting using Convolutional Neural Networks. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2559–2565. [CrossRef]
55. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587. [CrossRef]
56. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778. [CrossRef]
57. Xiang, A.J.; Huddin, A.B.; Ibrahim, M.F.; Hashim, F.H. An Oil Palm Loose Fruits Image Detection System using Faster R -CNN and Jetson TX2. In Proceedings of the 2021 International Conference on Electrical Engineering and Informatics (ICEEI), Kuala Terengganu, Malaysia, 12–13 October 2021; pp. 1–6. [CrossRef]
58. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
59. Nagaraju, Y.; Venkatesh; Venugopal, K.R. A Fruit Detection Method for Vague Environment High-Density Fruit Orchards. In Proceedings of the 2022 IEEE 3rd Global Conference for Advancement in Technology (GCAT), Bangalore, India, 7–9 October 2022; pp. 1–6. [CrossRef]
60. Jocher, G.; Stoken, A.; Chaurasia, A.; Borovec, J.; NanoCode012; Xie, T.; Kwon, Y.; Michael, K.; Changyu, L.; Fang, J.; et al. ultralytics/yolov5: v6.0—YOLOv5n 'Nano' Models, Roboflow Integration, TensorFlow Export, OpenCV DNN Support. 2021. Available online: https://doi.org/10.5281/zenodo.5563715 (accessed on 21 January 2024). [CrossRef]
61. Wu, Z.; Sun, X.; Jiang, H.; Mao, W.; Li, R.; Andriyanov, N.; Soloviev, V.; Fu, L. NDMFCS: An automatic fruit counting system in modern apple orchard using abatement of abnormal fruit detection. *Comput. Electron. Agric.* **2023**, *211*, 108036. [CrossRef]
62. Ning, M.; Lu, Y.; Hou, W.; Matskin, M. YOLOv4-object: An Efficient Model and Method for Object Discovery. In Proceedings of the 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, 12–16 July 2021; pp. 31–36. [CrossRef]
63. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets Robotics: The KITTI Dataset. *Int. J. Robot. Res. (IJRR)* **2013**, *32*, 1231–1237. [CrossRef]
64. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In Proceedings of the Computer Vision – ECCV 2014, Zurich, Switzerland, 6–12 September 2014; Springer International Publishing: Cham, Switzerland, 2014; pp. 346–361. [CrossRef]
65. Wang, C.Y.; Liao, H.Y.M.; Yeh, I.H. Designing Network Design Strategies Through Gradient Path Analysis. *arXiv* **2022**. http://arxiv.org/abs/2211.04800.
66. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on International Conference on Machine Learning, ICML'15, Lille, France, 6–11 July 2015; Volume 37, pp. 448–456.
67. Elfwing, S.; Uchibe, E.; Doya, K. Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning. *arXiv* **2017**. http://arxiv.org/abs/1702.03118. [CrossRef]
68. Pascanu, R.; Mikolov, T.; Bengio, Y. On the Difficulty of Training Recurrent Neural Networks. In Proceedings of the 30th International Conference on International Conference on Machine Learning, ICML'13, Atlanta, GE, USA, 16–21 June 2013; Volume 28, pp. III-1310–III-1318.
69. Wang, C.Y.; Mark Liao, H.Y.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1571–1580. [CrossRef]
70. Tian, Z.; Shen, C.; Chen, H.; He, T. FCOS: Fully Convolutional One-Stage Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019.

71. Li, X.; Lv, C.; Wang, W.; Li, G.; Yang, L.; Yang, J. Generalized Focal Loss: Towards Efficient Representation Learning for Dense Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 3139–3153. [CrossRef]
72. Feng, C.; Zhong, Y.; Gao, Y.; Scott, M.R.; Huang, W. TOOD: Task-aligned One-stage Object Detection. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021; pp. 3490–3499. [CrossRef]
73. Reis, D.; Kupec, J.; Hong, J.; Daoudi, A. Real-Time Flying Object Detection with YOLOv8. *arXiv* **2023**. http://arxiv.org/abs/2305.09972.
74. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 12993–13000. [CrossRef]
75. Zhang, H.; Cissé, M.; Dauphin, Y.N.; Lopez-Paz, D. mixup: Beyond Empirical Risk Minimization. In Proceedings of the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018.
76. Ciaglia, F.; Zuppichini, F.S.; Guerrie, P.; McQuade, M.; Solawetz, J. Roboflow 100: A Rich, Multi-Domain Object Detection Benchmark. *arXiv* **2022**. http://arxiv.org/abs/arXiv:2211.13523.
77. Raspberry Pi Foundation. Raspberry Pi. 2023. Available online: https://www.raspberrypi.org/ (accessed on 21 August 2023).
78. Gay, W. *Raspberry Pi Hardware Reference*, 1st ed.; Apress: New York City, NY, USA, 2014.
79. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.
80. Reddi, S.J.; Kale, S.; Kumar, S. On the Convergence of Adam and Beyond. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
81. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988. [CrossRef]
82. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic differentiation in PyTorch. In Proceedings of the NIPS-W, Long Beach, CA, USA, 4–9 December 2017.
83. Dukhan, M.; Wu, Y.; Lu, H.; Maher, B. QNNPACK: Quantized Neural Network PACKage. 2019. Available online: https://github.com/pytorch/QNNPACK (accessed on 22 August 2023).
84. Ahn, H.; Chen, T.; Alnaasan, N.; Shafi, A.; Abduljabbar, M.; Subramoni, H.; Panda, D. Performance Characterization of using Quantization for DNN Inference on Edge Devices: Extended Version. In Proceedings of the IEEE ICFEC 2023, Bengaluru, India, 30–31 January 2023.
85. Meyes, R.; Lu, M.; de Puiseau, C.W.; Meisen, T. Ablation Studies in Artificial Neural Networks. *arXiv* **2019**. http://arxiv.org/abs/1901.08644.
86. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA, USA, 7–9 May 2015; pp. 1–14.