System of counting green oranges directly from trees using artificial intelligence

Matheus Felipe Gremes
State University of Maringa
Igor Rossi Fermo
State University of Maringa
Rafael Krummenauer
State University of Maringa
Franklin Cesar Flores
State University of Maringa
Cid Marcos Gonçalves Andrade (cid@deq.uem.br)
State University of Maringa
Oswaldo Curti da Mota Lima
State University of Maringa

Research Article

Keywords: Fruits account, Deep Learning, Yolo, Object Tracker

Posted Date: May 17th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2935161/v1

Additional Declarations: No competing interests reported.

System of counting green oranges directly from trees using artificial intelligence

Matheus Felipe Gremes (0000-0002-3013-323X)[a], Igor Rossi Fermo (0000-0002-8527-251X)[a], Rafael Krummenauer (0000-0002-2901-6001)[a], Franklin César Flores (0000-0002-7818-1564)[b], Cid Marcos Gonçalves Andrade (0000-0003-3101-1709)*[a], and Oswaldo Curty da Motta Lima (0000-0001-7716-2604)[a]

[a]State University of Maringá (UEM), Department of Chemical Engineering

[b]State University of Maringá (UEM), IT Department

May 14, 2023

Acknowledgements

For their support National Council for Scientific and Technological Development (CNPq) and National Council for the Improvement of Higher Education (CAPES).

*Corresponding author. Email: cid@deq.uem

Abstract

Agriculture is one of the most essential activities for humanity. Systems capable of automatically harvesting a crop using robots or performing a reasonable production estimate can reduce costs and increase production efficiency. With the advancement of computer vision, image processing methods are becoming increasingly viable in solving agricultural problems. Thus, this work aims to count green oranges directly from the trees through video footage filmed in line along a row of orange trees on the plantation. For the video image processing flow, a solution was proposed integrating the YOLOv4 network with object tracking algorithms. In order to compare the performance of the counting algorithm using the YOLOv4 network, an optimal object detector was simulated in which frame-by-frame corrected detections were used in which all oranges in all video frames were detected, and there were no erroneous detections. The use of YOLOv4 together with object detectors managed to reduce the number of double counting error and obtained a count close to the actual number of oranges visible in the video. The study also resulted in a database with an amount of 644 images with 43109 annotated oranges that can be used in future works.

Keywords: Fruits account, Deep Learning, Yolo, Object Tracker

1 Introduction

Agriculture is one of the most essential activities for humanity. Fruits are a great source of nutrients in most people's diets. Thus, continuous production is necessary to meet global demands [1]. One of the ways to increase production quality and reduce costs is through technological innovations such as computer vision, which has been showing significant advances in pre-harvest fruit image processing. The two computer vision applications that have recently developed the most in this area were in the fruit production estimation and robotic harvesting sectors as shown by [2].

Pre-harvest fruit crop production estimation is an essential task in precision agriculture. The procedure offers several benefits to producers. For example, performing site-specific management based on production estimation optimizes the number of materials needed, such as agricultural chemicals and fertilizers. It reduces the cost of labor in growing and harvesting production [3]. Production estimation has other benefits, such as estimating the storage capacity needed to store the fruits after harvest [4].

Recent advances have been made in this field [2, 5]. Nevertheless, there are still many challenges, such as variation in lighting conditions, object occlusion or overlap, low contrast between fruits and foliage, and variation in the shape and pose of the fruits [6, 7]. However, computer vision systems based on Deep Learning [8] can deal with these challenges, essential and necessary resources for the recognition of complex objects in external environments [9].

Algorithms based on Deep Learning have proven to be one of the most robust ways to detect objects [10, 11]. In particular, the object detection algorithms of the YOLO family [12] proved to be effective for counting fruits directly from trees [13, 14, 15]. Object detection results can be integrated by associating the data using object trackers to perform in-line fruit counts in the field [16].

Some works found in the literature present similar proposals to this one. However, most

perform the counting of the fruits using images or with ripe oranges [17, 18, 19, 20], unlike this work that performs the count using video and with green oranges.

This work proposes a solution for evaluating the production of oranges in the rows of the orange grove through detecting and tracking the fruits by image processing along the video frames. To accomplish that task, a new dataset of images of green oranges (pre-harvest) was created. Being the scientific and technological innovation the possibility of distinguishing the green color of the fruits from the green color of the leaves. This set was used to train a green-orange detector and the respective detected objects, marked by bounding boxes, go through a tracking process to compute the count.

2 Problem Statement

This work aims to count green oranges directly from the trees through video footage filmed in line along a row of orange trees on the plantation. As the images acquired for the detection and counting of fruits in this type of environment are dense scenes in quantities (that is, images or videos with a large number of objects, often accompanied by a large number of occlusions or overlaps [21]) there are some challenges in the task of detecting and counting pre-harvest fruits, and among the complicating aspects they highlight: i) occlusion, ii) the size of objects to be detected in the images and iii) the complex background [22], as can be seen in Figure 1.

Figure 1: Oranges with and without bounding boxes.

Long and Darrell [23] pointed out that occlusion is one of the biggest challenges in dense

scene quantities. There are two main types of occlusion: scene occlusion and inter-object occlusion [21]. Scene occlusion occurs when other non-object elements in the scene occlude the object; inter-object occlusion occurs when objects detected in the environment overlap other objects.

The reduced size of the objects, in this case, the oranges, in the image is another challenge in dense scenes in quantities. This problem causes objects to have a low resolution making them subject to noise and lighting changes that can lead to inaccurate detections.

The complex background includes two situations; the primary and most relevant for this work is that the background is similar to the objects, which makes it difficult to distinguish between the object and the background. Green oranges are easily confused with tree foliage, which makes detection and counting more complex when compared to ripe oranges.

Promising approaches to solving the fruit detection task are techniques based on Deep Learning. They are robust to the challenges discussed above and are more flexible and straightforward to shape compared to more traditional image processing techniques [10]. One of the main limitations of using techniques based on Deep Learning is the image database used to train the system [24]. The database must be large enough, with representative samples of the scenario to be treated and well labeled to perform the detection task efficiently. Therefore, preparing the database is one of the tasks that demand more effort and work in applying techniques based on Deep Learning. In addition, not all databases proposed in other works are available to be used, which makes the reproducibility of experiments complex [24], as is the case of the pre-harvest green oranges database.

Therefore, one of the contributions of this work was to create a database with labeled images of pre-harvest green oranges and make them available to the scientific community. The green-orange image database is a challenging dataset due to the arrangement of the oranges in the trees, the daytime lighting and the similarity in color between the foliage and the oranges. Thus, there are high levels of occlusion due to the foliage and the fruits

themselves, uncontrolled lighting due to daytime lighting, and significant similarity of color between the fruits and the background due to the maturation stage of the oranges. All characteristics of images of the fruit in nature [25].

The fruit count can be divided into two steps, detection of fruits, already discussed in the previous paragraphs, and tracking fruits through a sequence of frames of the footage of the trees. To count the fruits in the video is necessary to carry out the correspondence of each fruit in the adjacent frames. A given orange in several frames must be identified, and other oranges must be distinguished from it simultaneously. However, tracking the same oranges for identification is challenging, as their location and appearance vary due to environmental factors, such as lighting conditions and camera movement in the in-line footage of the rows of orange trees [26]. Many works in the literature deal with fruit detection; however, few use detection and tracking to perform fruit counting, especially for greenish fruits and under unstable lighting conditions [26]. Even if foliage, branches, or other oranges occlude orange, it is still necessary to track it. Otherwise, the orange will be counted twice or more each time it is detected again in successive video frames.

3 Materials and methods

Algorithms based on methods that use Deep Learning are the most efficient way to perform object detection [10, 11]. Considering speed and accuracy, YOLOv4 (You Only Look Once) has presented good performance among object detection models recently [27]. Furthermore, YOLOv4 was designed to allow training on a single conventional GPU, unlike other models. Recent studies have used YOLO-based models for fruit detection. They have shown that these models have enormous potential in accurately detecting fruit directly from trees [28, 29, 30, 31]. Thus, YOLOv4 [32] was chosen to carry out the detection of oranges in this work.

To carry out the YOLOv4 training to detect pre-harvest green oranges, a database containing images of oranges directly from the trees and duly annotated is necessary. It is one of the contributions of the present study, the creation of a pre-harvest and duly annotated green oranges dataset. Among the types of oranges available in the place where the images for the database were acquired, the type chosen for this work was the variety called "folha murcha", which is a Valencia-type orange tree [33], with data collection occurring between 7 and 6 months before harvest.

Data were obtained in the field using a Xiaomi Redmi Note 9PRO Smartphone filming in both portrait and landscape orientations. At the same time, the camera operator walked in a straight line parallel to the row of orange trees being filmed. Data collection took place on 03/15/2021 and 04/18/2021. The orange trees were filmed with 1920x1080p resolution at 60 frames per second. The images for the database were taken from the video frames at 3-second intervals.

The oranges were divided into three categories: green oranges, ripe oranges and spoiled oranges. Furthermore, they were annotated using the online tool CVAT (COMPUTER VISION ANNOTATION TOOL) [34] totaling 644 images with 43109 annotated oranges, of

which 532 images were separated for training and 112 images for tests. In Figure 2 and Figure 3, the oranges properly annotated using the CVAT tool are shown, where green oranges are annotated with blue bounding boxes, ripe oranges with yellow bounding boxes and spoiled oranges with purple bounding boxes.

Figure 2: Labeled oranges images.

The YOLOv4 training was done using the Darknet framework on the Google Colab platform with the Tesla P100-PCIE-16GB GPU. We configured and tuned the YOLOv4 architecture for our custom database. The main source code of the Darknet framework was prepared by [32]. We modified the last three layers of YOLOv4 to be compatible with the number of classes in our database, following the authors' guidelines [32]. The original YOLOv4 was trained in 80 classes; therefore, we have changed the number of classes to three: "green orange", "ripe orange", and "spoiled orange". We set the width x height of the network input image to 1056 x 1056. This input value was chosen to consider the size of the oranges in the image, if the oranges become smaller than 11 x 11 pixels after resizing the image in the input, this can compromise the quality of network detections. Data augmentation techniques and network hyperparameters were kept with the framework's default values. In addition, the maximum number of training epochs was set at 6000 following the formula provided by the authors (number of classes x 2000) [32].

Figure 3: Labeled oranges images.

Tensorflow [35] is a machine learning framework that works in data processing and has a wide variety of libraries and resources that allow the use of the latest Deep Learning algorithms and models in a flexible way. After The YOLOv4 model had been trained, the model was converted to a Tensorflow model, using the following source code as a basis to perform the conversion [36].

It is possible to use only detection to count objects. However, as discussed earlier, detection systems often fail in some occlusion and lighting situations. Thus, relying solely on the number of detections in an image to perform the orange count would be a wrong decision, especially in a pre-harvest scenario where the abovementioned situations are pretty standard. For this reason, a counting system must cover these limitations to ensure counting accuracy. One of the ways to do this is by assigning a unique ID to each orange detected and tracking it through video frames. In this way, obtaining more reliable results in counting objects in case of detection system failures is possible.

The system used in this work uses two methods of tracking the oranges using a unique ID across the frames. The first uses the Euclidean distance between the centroids of the objects detected by YOLOv4 in subsequent frames to relate the IDs of the oranges of the previous frame with the detected oranges by YOLOv4 in the current frame. The second uses object tracking algorithms to match the IDs of the oranges of the previous frame with the oranges of the current frame that YOLOv4 has not detected.

The first method, the Euclidean distance between centroids, is divided into four steps. Step 1: Get the coordinates of the bounding boxes and calculate their centroids. Step 2: Calculate the Euclidean distance between objects in the previous frame and objects in the current frame. Step 3: Update object coordinates. Step 4: Register objects with new IDs.

In step 1, the algorithm receives the coordinates of the bounding boxes and calculates their respective centroids. Assuming this is the first received set of bounding boxes. Each centroid, or object, is assigned a unique ID, as shown in the left image of Figure 4. The object's centroids are calculated at each video frame using the bounding boxes. However, instead of assigning new unique IDs to objects detected in the current frame, it is first necessary to determine if it is possible to associate the centroids of the new objects with the centroids of the objects of the previous frame. It is done through step 2.

In step 2, the Euclidean distance between each pair of object centroids of the previous frame and centroids of objects of the current frame is calculated. In the right image of Figure 4, the centroids of the objects of the previous frame are represented by the two blue dots, and the yellow dots represent the centroids of the objects of the current frame. Arrows represent Euclidean distances.

The central assumption made in the first method is that objects in consecutive frames tend to move little. In this way, the distance between the centroids of the same object in 2 consecutive frames will be smaller than the distances between all other centroids.

In step 3, the centroids of objects with the smallest Euclidean distance between them

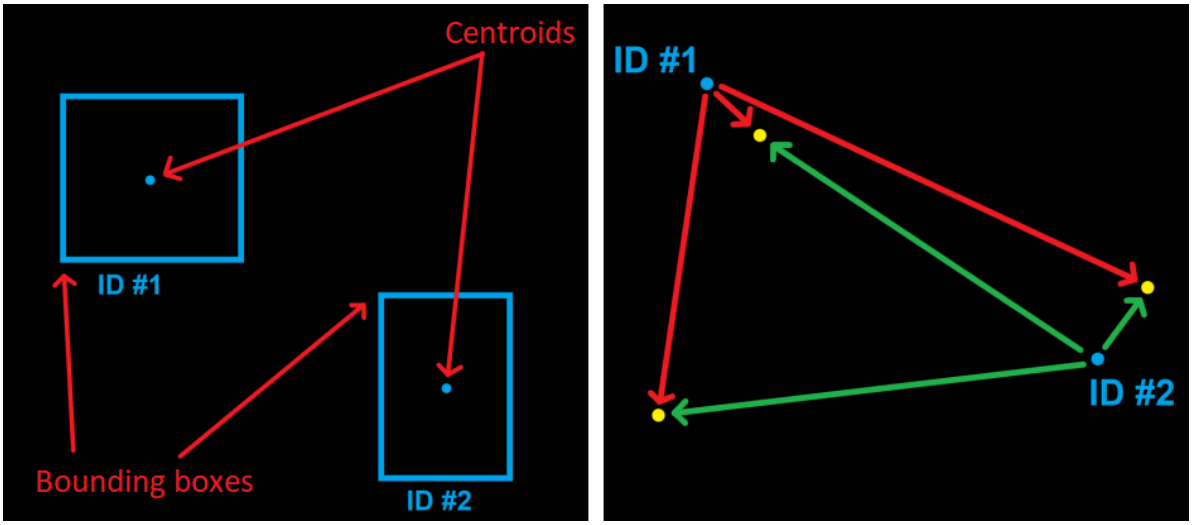

Figure 4: ID assignment to nearby centroids - step 1 and 2.

will have their IDs related, as shown in the left image of Figure 5. However, if the number of new objects is greater than the number of existing objects in the previous frame, or if these objects are at a distance greater than 70 pixels from all objects that have not been associated with an ID, then it will not be possible to associate the IDs of objects from the previous frame with all objects in the current frame, such as is shown in the left image of Figure 5 with the isolated yellow dot. Therefore, it is necessary to associate these new objects with new IDs in step 4.

In step 4, objects that have not been associated with objects with already existing IDs have new IDs assigned to them if the detection confidence of this object is 85% or more, as shown in the right image of Figure 5. This situation happens when an object, which was not part of the previous frames, is detected.

In Figure 6, we can see the first method for ten consecutive frames. In frame number 163, there are three oranges with IDs numbered 81, 80 and 77, each with a circle of different colors inside it to facilitate the identification of oranges of identical IDs in different frames. The smaller black circle inside the orange represents YOLOv4 detected orange in that frame. It is possible to notice that the oranges move very little between consecutive frames, which

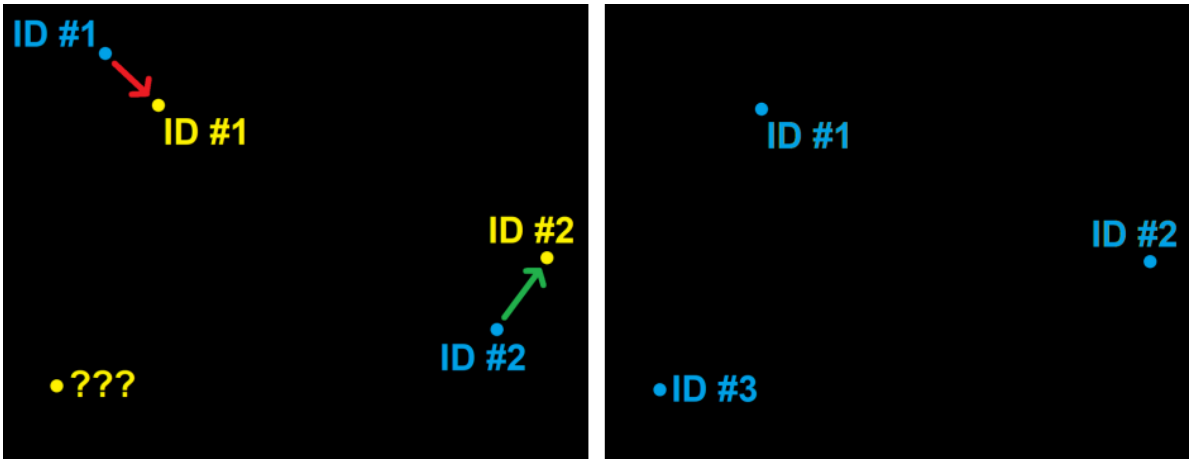

Figure 5: ID assignment to nearby centroids - step 3 and 4.

facilitates the attribution of the IDs between the oranges frame by frame.

The second method of tracking oranges through video frames with a unique ID is used when it is impossible to relate an object's ID from the previous frame to an object that YOLOv4 has detected in the current frame. In this case, we use object tracking algorithms to make this relationship.

In this process, the objective of object tracking algorithms is to estimate the object's state (position) over time [37]. When there is no change in the environment, tracking objects is not overly complex, but this is usually not the case. Various disturbances in the real world can disrupt tracking, including occlusion, variations in lighting, change of viewpoint, rotation and blurring due to motion [38]. The steps used to track the object in the video involve:

- selecting the object (target) in the initial frame with a bounding box
- initializing the tracker with information about the frame and the bounding box of the object
- using subsequent frames to find the new bounding box of the object in these frames

In this work, the following object trackers are used for comparison: Dlib correlation tracker (Dlib tracker) [39], Boosting [40], Multiple Instance Learning (MIL) [41], MedianFIow

Figure 6: ID assignment for nearby oranges.
Source: Research data

[42], Kernelized Correlation Filter (KCF) [38] and Channel and Spatial Reliability Tracker (CSRT) [43]. Except for the Dlib tracker, which is implemented using the Dlib-ml library [44], all other object trackers are implemented using the OpenCV (The Open Source Computer Vision) library [45] [38].

Object trackers are used to estimating the object's position when YOLOv4 cannot detect the object in the image in the current frame. It is possible because the trackers only need the previous frame and the object's bounding box. After that, object trackers can track the object frame by frame without the help of YOLOv4. The entire process is shown in the flowchart in Figure 7.

Figure 8 shows an example of this situation using the Dlib tracker. In frame 196, the orange of ID 85 was detected by YOLOv4; this is demonstrated by the black circle inside. In frame 197, the orange of ID 85 was not detected by YOLOv4; the white circle inside it visually demonstrates this, which indicates that the object tracking algorithm is tracking

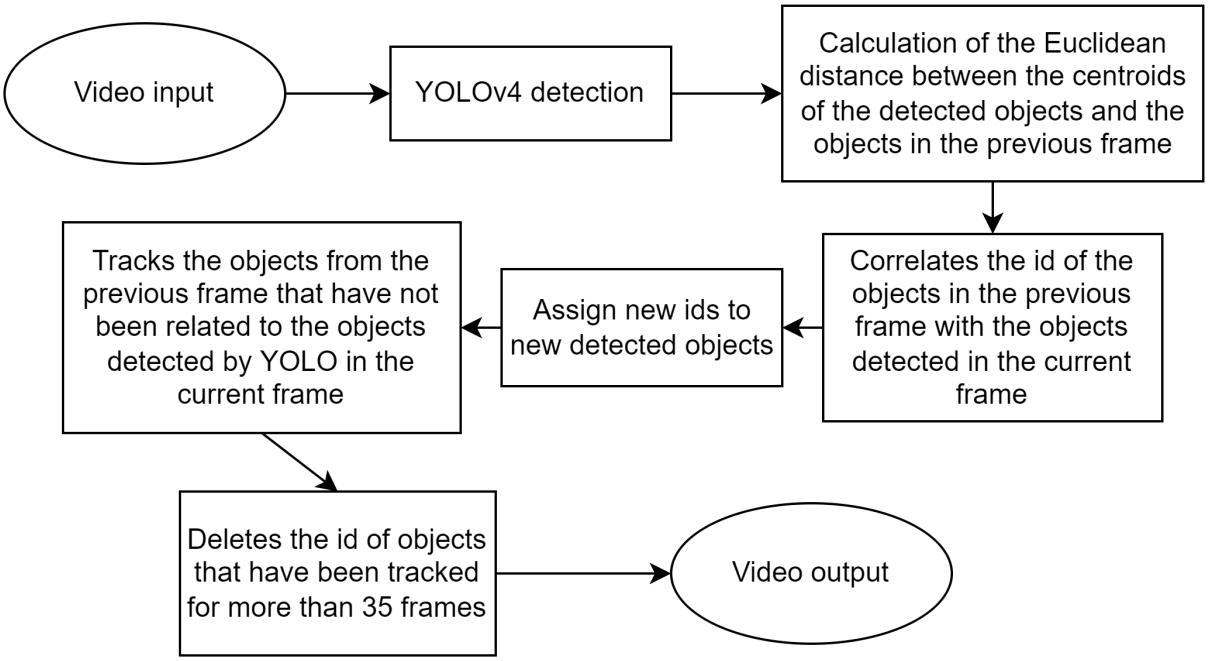

Figure 7: Counting algorithm flowchart.

the orange. The orange is then tracked through the frames even if YOLOv4 fails to detect it again. This tracking process takes place for 35 frames, if YOLOv4 cannot detect the orange again within those 35 frames, then the ID of that orange is deleted, and the object tracking algorithm stops tracking it.

Ten seconds of a video were chosen whose images were not used to compose the database to test the counting algorithm. So, in order to compare the performance of the algorithm using YOLOv4, a situation was simulated in which the detection algorithm used would detect all oranges in all video frames without committing any detection error. It was done by manually annotating all oranges in all the 600 frames of the 10-second video with bounding boxes. This information was used frame by frame to simulate an optimal detection. The actual number of 208 oranges possible to see in the video was also counted. The counting algorithm was then tested using YOLOv4 and the simulated optimal object detector together with each of the mentioned above tracking object algorithms.

Figure 8: ID 85 orange tracking.
Source: Research data

4 Results and discussions

After training YOLOv4, the network obtained the parameters shown in Table 1.

Table 1: **YOLOv4 parameters**

Model	**mAP50**	**Precision**	**Recall**	**F1-Score**	**Average IoU**
YOLOv4	80,16%	0,92	0,93	0,93	82,08%

The counting algorithm using the frame-by-frame corrected detections obtained the results shown in Table 2. As the detections were corrected frame by frame, to compare with the results obtained using the YOLOv4 network, there is no detection omission (oranges that were not detected in frames) and no false detections (objects that were detected as being orange, but which are not). The number of oranges counted by the algorithm remained close to the correct number of oranges, no matter which object tracker is used or even if none is used.

It is because whenever an orange is visible in the frame, it will be detected as the detections were manually corrected frame by frame. That way, when the object trackers are used, they are not tracking the oranges because if the orange was not detected, it is because it is no longer visible in the frame and not because the object detector failed. Thus, even if no object tracker is used, the number of oranges counted is still very close, even if the double counting number and the ID repetition number (situation in which the same ID is used in two oranges, generally when the first orange is occluded and soon after a new orange is detected nearby and the same ID of the occluded orange is assigned to the new detection) are more significant.

When the counting algorithm is used with YOLOv4 to perform the detection of oranges, we have a more significant discrepancy in the number of oranges counted, as shown in Table 3. The number of omission of detections was 15 oranges, meaning that during all frames, there were 15 oranges that YOLOv4 did not detect even once in all frames. It does not mean that

Table 2: **Orange count with the simulated optimal object detector**

Tracker	**Omission of detection** -	**Wrong detection** +	**Double count** +	**Repeated ID** -	**Number of oranges counted**	**Correct number of oranges counted**
No tracker	0	0	10	2	216	208
Dlib	0	0	8	0	216	208
Boosting	0	0	7	0	215	208
Csrt	0	0	8	1	215	208
Kcf	0	0	10	0	218	208
Medianflow	0	0	8	0	216	208
Mil	0	0	9	0	217	208

YOLOv4 failed to detect oranges 15 times in all frames, but rather that it missed 15 oranges in all frames at least once, so if the same orange was missed in multiple frames, it counts as an omission of detection only. If an orange appeared in 100 frames but was detected by YOLOv4 in 20 frames, it does not count as an omission of detection, as it is possible to track the orange in frames where YOLOv4 was unable to detect it using the object trackers.

The number of erroneous detections was 2, meaning that the network detected two objects as being orange and they were not. In the same way, even though YOLOv4 has detected the same object erroneously in several frames, this counts only as one false detection. The number of double counting of oranges and repetition of the same ID for different oranges was also higher when no tracker was used than when any of the proposed object trackers were used.

It was already expected because now, unlike the detections corrected frame by frame, there were times when the orange was visible in the frame, but YOLOv4 did not detect it. In this way, having an object tracker to be able to track the oranges in the frames in which YOLOv4 fails helps to reduce the number of double counting. Because if YOLOv4 detects the orange again in future frames, reassign the ID of the orange being tracked by the object tracker instead of assigning a new one ID.

Table 3: **Orange count with YOLOv4**

Tracker	**Omission of detection** -	**Wrong detection** +	**Double count** +	**Repeated ID** -	**Number of oranges counted**	**Correct number of oranges counted**
No tracker	15	2	25	5	215	208
Dlib	15	2	10	1	204	208
Boosting	15	2	10	1	204	208
Csrt	15	2	10	1	204	208
Kcf	15	2	16	1	210	208
Medianflow	15	2	10	1	204	208
Mil	15	2	9	1	203	208

Although frame-by-frame corrected detections are more accurate than YOLOv4-produced detections, the orange counting algorithm came closer to the correct number of oranges when using YOLOv4 detections than when using frame-by-frame corrected detections. It is because when using the corrected detections, there is no omission of detections or wrong detections, so the only errors present are double counting errors and ID repetition errors. ID repetition errors are much rarer than double counting errors, especially when using an object tracker. Thus, double counting errors are the majority of errors when using corrected detections. By their nature, double counting errors tend to overestimate the number of oranges counted. That is, they tend to introduce a positive error in the final count of the number of oranges.

When using the detections made by YOLOv4, there are some detection failures. Some oranges are not detected at all in every frame, either because of occlusion or lighting, and some objects are detected as being orange when they are not. We have a total of 15 oranges that were not detected and two objects that were erroneously detected as being oranges. In this way, we have a more significant error due to the oranges that were not detected than for the objects detected wrongly. Unlike double counting, the error introduced by the omission of detection tends to underestimate the correct number of oranges. That is, it introduces a negative error in the final count. Thus, when the detections made by YOLOv4 are used, the

error introduced by the omission of detection tends to balance the error introduced by the double counting. That is why the counting algorithm arrived at a count closer to the actual value when using YOLOv4 detections than frame-by-frame corrected detections.

It is also possible to notice that there was no significant difference in the count when different object trackers were used. It is because the object tracker only tracks the object for a maximum of 35 frames after the object detector can no longer detect it. If, after these 35 frames, the object detector has not detected a nearby orange that can be assigned to the orange tracked by the object tracker, the ID of that orange is deleted. The video acquisition was performed at 60 frames per second, there are no significant variations from one frame to its subsequent frame, as seen in Figure 8. Therefore, at 35 frames, there are no significant variations in the orange image between the first frame and the last one, which makes it very difficult to track, making the object trackers have similar performances.

In Figure 9 and Figure 10, 20 frames are shown spaced from the first frame to the last of the 10-second video, where it is possible to see the counting algorithm with YOLOv4 for detection and the Dlib tracker for tracking objects. In the upper left corner of each frame is indicated respectively from top to bottom:

- the number of that frame
- the total number of oranges already counted by the algorithm
- the total number of oranges present in that frame
- the type of object tracker used
- the number of oranges being tracked by the object tracker

In the upper left corner of each orange's bounding box, we have a legend indicating the ID number assigned to that orange. The larger colored circles are used to distinguish oranges of different IDs visually. The smaller black and white circles within the larger colored circles

are used to indicate whether the object detector detected that orange in that frame or not. Black if the object detector was able to detect and white if the object detector was not able, and therefore the object tracker is being used to track orange through the respective frames.

Figure 9: Frames 1, 32, 64, 96 and 128 of the counting algorithm

In Figure 11, we have a situation of occlusion by a branch in which the algorithm can reassign the same ID to the respective oranges, thus avoiding double counting. In frame 208, the algorithm is tracking the oranges of IDs 13 and 91, and we can see from the black circle marking that the object detector detected both in that frame. In frame 228, both oranges are no longer detected by the object detector, as it is possible to see by the marking of the white circle. Currently, the oranges are being tracked using the Dlib object tracker. In frame 248, both oranges are detected again by the object detector, and their IDs are reassigned

Figure 10: Frames 160, 192, 224, 256 and 288 of the counting algorithm

without double counting.

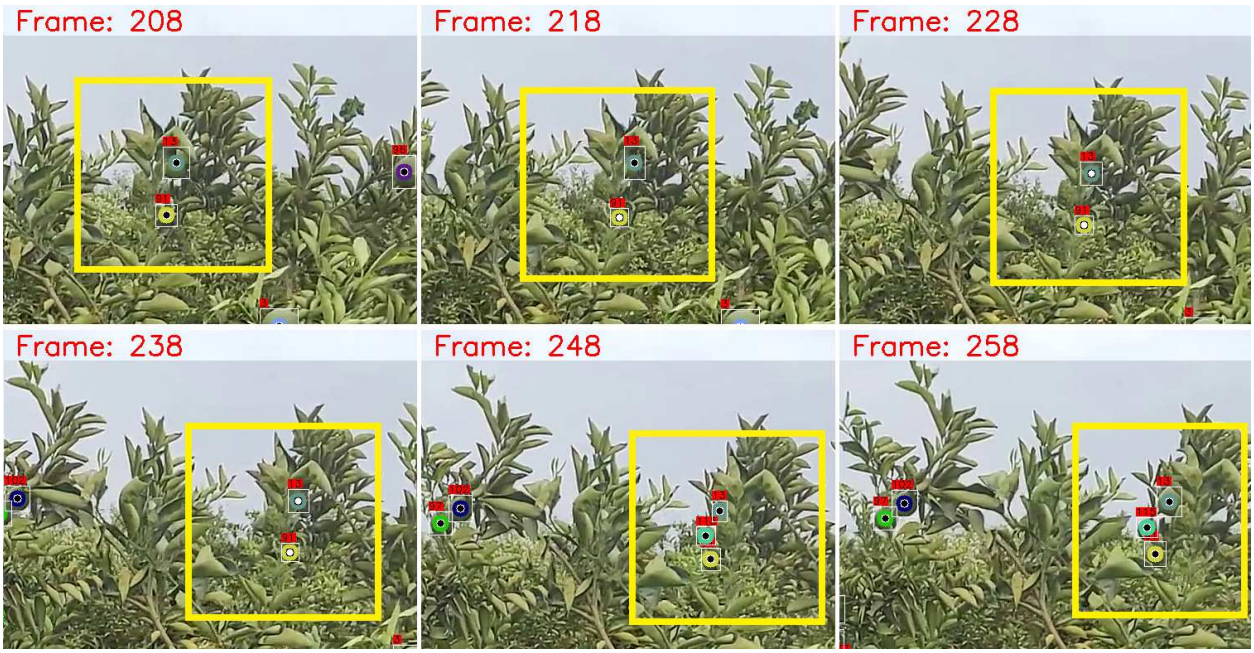

Figure 11: Occlusion situation without double counting - frame 208 to frame 258.

Unlike the situation shown in Figure 11, in which there was an occlusion situation, but the counting algorithm reassigned the orange IDs, thus avoiding double counting, in Figure 12, we have an occlusion situation in which the algorithm was not able to avoid double counting. The orange of ID 84 was being tracked until the moment it was occluded by foliage and is no longer detected by the object detector in frame 156. After not being detected again by the object detector in the following frames, the algorithm stops tracking this orange in frame 186. However, in frame 280, that same orange is again detected by the object detector after the foliage stops obstructing her line of sight. However, in this case, we do not have the position of the ID 84 bounding box next to it to reassign the same ID since the algorithm stopped tracking it at frame 186, so a new ID number 124 is assigned to that orange, so the same orange was counted twice.

This same occlusion and double counting situation shown in Figure 12 occurs again in Figure 13. The orange in frame 126 is being tracked by the algorithm and has ID 11. In

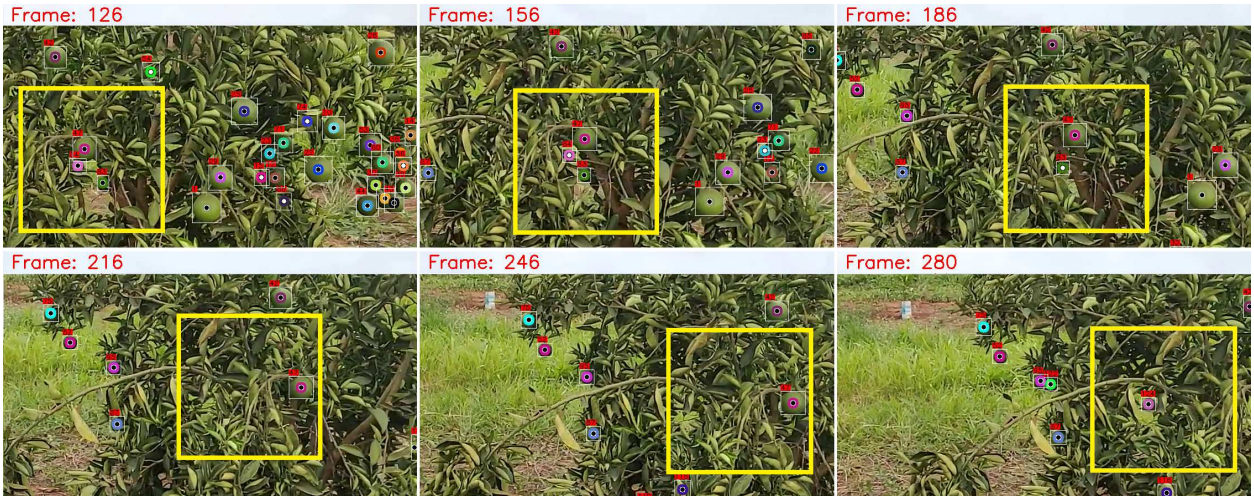

Figure 12: Foliage occlusion situation with double count - frame 126 to frame 280.

frame 144, this orange is no longer detected by the object detector and becomes tracked by the Dlib object tracker. In frame 162, as the object detector could not detect the orange again due to the occlusion caused by the foliage, the algorithm stopped tracking the orange and stopped showing its ID and its bounding box in subsequent frames. However, in frame 218, that same orange is again detected by the object detector, now receiving ID number 102, so the same orange was counted twice.

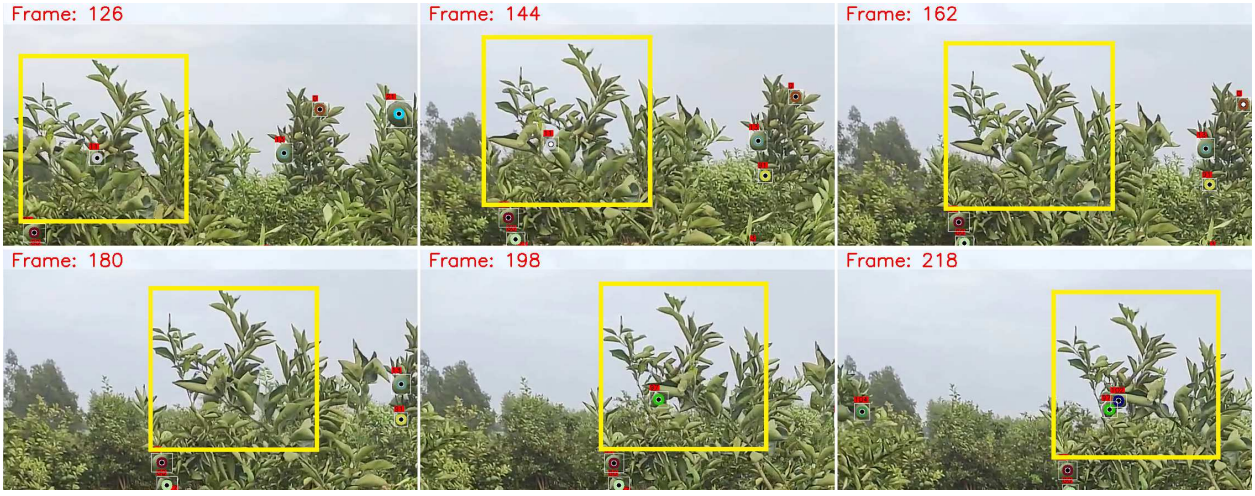

Figure 13: Double count situation - frame 126 to frame 218.

In Figure 14, we have another double counting situation, but this time instead of occlusion

due to foliage, occlusion occurs due to another orange. The orange of ID 99 was being tracked by the algorithm, as it is possible to see in frames 256 and 268. However, in frame 280, the orange of ID 99 is obstructed by the orange of ID 103 and is no longer detected by the object detector. At that moment, the Dlib object tracker now does the orange tracking. As the object detector cannot detect the orange of ID 99 again in subsequent frames, the algorithm stops tracking this orange and stops showing its bounding box and its ID in the frames, as seen in frame 301. However, the orange that had been obstructed by the orange ID 103 becomes visible again in frame 316, as it is impossible to reassign ID 99, as it was discarded in previous frames, the algorithm assigns the new ID number 133. Thus performing the double counting of the same orange.

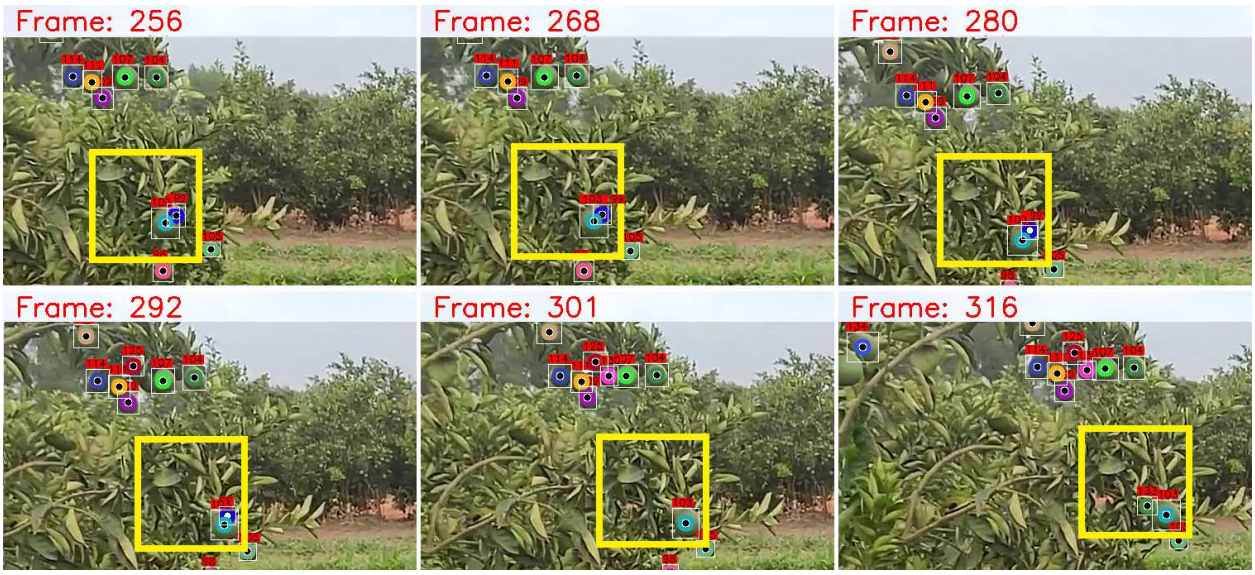

Figure 14: Double count situation - frame 256 to frame 316

5 Conclusion

This study addressed the problems of detecting, tracking and counting oranges from video footage of the plantation. For the video image processing flow, a solution was proposed to integrate the YOLOv4 network with object tracking algorithms, providing promising results.

The YOLOv4 network demonstrated competence in detecting small oranges compared to the image size and greens with a color closely resembling the surrounding foliage. At the same time, the object tracking algorithms benefited from the information provided by YOLOv4 to position the images. Bounding boxes in oranges and performing tracking at times when YOLOv4 could not detect them, thus reducing the double counting error.

Frame-by-frame corrected detections were used in which all oranges in all video frames were detected, and there were no erroneous detections to compare the performance of the counting algorithm using the YOLOv4 network. The results were promising. The use of YOLOv4 and object detectors reduced the number of double counting errors and obtained a count close to the actual number of oranges visible in the video.

The study also resulted in a database with an amount of 644 images with 43109 annotated oranges that can be used in future works.

6 Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

7 Data Availability

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

References

[1] Halimatu Sadiyah Abdullahi, R Sheriff, and Fatima Mahieddine. "Convolution neural network in precision agriculture for plant image recognition and classification". In: *2017 Seventh International Conference on Innovative Computing Technology (INTECH)*. Vol. 10. Ieee. 2017, pp. 256–272.

[2] Matheus Felipe Gremes, Rafael Krummenauer, Cid Marcos Gonçalves Andrade, and Oswaldo Curty da Motta Lima. "Pre-Harvest Fruit Image Processing: A Brief Review". In: *Brazilian Journal of Experimental Design, Data Analysis and Inferential Statistics* 1.2 (2021), pp. 107–121.

[3] Kyosuke Yamamoto, Wei Guo, Yosuke Yoshioka, and Seishi Ninomiya. "On plant detection of intact tomato fruits using image analysis and machine learning methods". In: *Sensors* 14.7 (2014), pp. 12191–12206.

[4] Qi Wang, Stephen Nuske, Marcel Bergerman, and Sanjiv Singh. "Automated crop yield estimation for apple orchards". In: *Experimental robotics*. Springer. 2013, pp. 745–758.

[5] Igor R Fermo, Thiago S Cavali, Lucas Bonfim-Rocha, Caio L Srutkoske, Franklin C Flores, and Cid MG Andrade. "Development of a low-cost digital image processing system for oranges selection using hopfield networks". In: *Food and bioproducts processing* 125 (2021), pp. 181–192.

[6] Songtao Wu, Shenghua Zhong, and Yan Liu. "Deep residual learning for image steganalysis". In: *Multimedia tools and applications* 77.9 (2018), pp. 10437–10453.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012).

[8] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), pp. 436–444.

[9] Thiago T Santos, Leonardo L de Souza, Andreza A dos Santos, and Sandra Avila. "Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association". In: *Computers and Electronics in Agriculture* 170 (2020), p. 105247.

[10] Andreas Kamilaris and Francesc X Prenafeta-Boldú. "Deep learning in agriculture: A survey". In: *Computers and electronics in agriculture* 147 (2018), pp. 70–90.

[11] Andreas Kamilaris and Francesc X Prenafeta-Boldú. "A review of the use of convolutional neural networks in agriculture". In: *The Journal of Agricultural Science* 156.3 (2018), pp. 312–322.

[12] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[13] Anand Koirala, Kerry B Walsh, Zhenglin Wang, and Cheryl McCarthy. "Deep learning–Method overview and review of use for fruit detection and yield estimation". In: *Computers and electronics in agriculture* 162 (2019), pp. 219–234.

[14] Kushtrim Bresilla, Giulio Demetrio Perulli, Alexandra Boini, Brunella Morandi, Luca Corelli Grappadelli, and Luigi Manfrini. "Single-shot convolution neural networks for real-time fruit detection within the tree". In: *Frontiers in plant science* 10 (2019), p. 611.

[15] Yuanyue Ge, Ya Xiong, Gabriel Lins Tenorio, and Paal Johan From. "Fruit localization and environment perception for strawberry harvesting robots". In: *IEEE Access* 7 (2019), pp. 147642–147652.

[16] Xu Liu, Steven W Chen, Chenhao Liu, Shreyas S Shivakumar, Jnaneshwar Das, Camillo J Taylor, James Underwood, and Vijay Kumar. "Monocular camera based fruit counting and mapping with semantic data association". In: *IEEE Robotics and Automation Letters* 4.3 (2019), pp. 2296–2303.

[17] Xiaohua Zhang, Arash Toudeshki, Reza Ehsani, Haoling Li, Wenfeng Zhang, and Ruijun Ma. "Yield estimation of citrus fruit using rapid image processing in natural background". In: *Smart Agricultural Technology* 2 (2022), p. 100027.

[18] Ulzii-Orshikh Dorj, Malrey Lee, and Sang-seok Yun. "An yield estimation in citrus orchards via fruit detection and counting using image processing". In: *Computers and Electronics in Agriculture* 140 (2017), pp. 103–112.

[19] Wenli Zhang, Jiaqi Wang, Yuxin Liu, Kaizhen Chen, Huibin Li, Yulin Duan, Wenbin Wu, Yun Shi, and Wei Guo. "Deep-learning-based in-field citrus fruit detection and tracking". In: *Horticulture Research* 9 (2022).

[20] Walter Maldonado Jr and José Carlos Barbosa. "Automatic green fruit counting in orange trees using digital images". In: *Computers and Electronics in Agriculture* 127 (2016), pp. 572–581.

[21] Qian Zhang, Yeqi Liu, Chuanyang Gong, Yingyi Chen, and Huihui Yu. "Applications of deep learning for dense scenes analysis in agriculture: A review". In: *Sensors* 20.5 (2020), p. 1520.

[22] Longsheng Fu, Fangfang Gao, Jingzhu Wu, Rui Li, Manoj Karkee, and Qin Zhang. "Application of consumer RGB-D cameras for fruit detection and localization in field: A critical review". In: *Computers and Electronics in Agriculture* 177 (2020), p. 105687.

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[24] José Naranjo-Torres, Marco Mora, Ruber Hernández-Garcı´a, Ricardo J Barrientos, Claudio Fredes, and Andres Valenzuela. "A review of convolutional neural network applied to fruit image processing". In: *Applied Sciences* 10.10 (2020), p. 3443.

[25] Steven W Chen, Shreyas S Shivakumar, Sandeep Dcunha, Jnaneshwar Das, Edidiong Okon, Chao Qu, Camillo J Taylor, and Vijay Kumar. "Counting apples and oranges with deep learning: A data-driven approach". In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 781–788.

[26] Kenta Itakura, Yuma Narita, Shuhei Noaki, and Fumiki Hosoi. "Automatic pear and apple detection by videos using deep learning and a Kalman filter". In: *OSA Continuum* 4.5 (2021), pp. 1688–1695.

[27] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. "Scaled-yolov4: Scaling cross stage partial network". In: *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*. 2021, pp. 13029–13038.

[28] Anand Koirala, KB Walsh, Zhenglin Wang, and C McCarthy. "Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of 'MangoYOLO'". In: *Precision Agriculture* 20.6 (2019), pp. 1107–1135.

[29] Guoxu Liu, Joseph Christian Nouaze, Philippe Lyonel Touko Mbouembe, and Jae Ho Kim. "YOLO-tomato: A robust algorithm for tomato detection based on YOLOv3". In: *Sensors* 20.7 (2020), p. 2145.

[30] Lin Wu, Jie Ma, Yuehua Zhao, and Hong Liu. "Apple detection in complex scene using the improved YOLOv4 model". In: *Agronomy* 11.3 (2021), p. 476.

[31] Bin Yan, Pan Fan, Xiaoyan Lei, Zhijie Liu, and Fuzeng Yang. "A real-time apple targets detection method for picking robot based on improved YOLOv5". In: *Remote Sensing* 13.9 (2021), p. 1619.

[32] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. *Yolov4: Optimal speed and accuracy of object detection*. 2020.

[33] RP Leite Junior. "A citricultura no Paraná". In: *Circular, Londrina: IAPAR* 72 (1992).

[34] D Roberts, M Wang, W Torres Calderon, and M Golparvar-Fard. "An annotation tool for benchmarking methods for automated construction worker pose estimation and activity analysis". In: (2019), pp. 307–313.

[35] Yuan Tang. "TF. Learn: TensorFlow's high-level module for distributed machine learning". In: *arXiv preprint arXiv:1612.04251* (2016).

[36] Jack Wotherspoon. *GitHub - theAIGuysCode/tensorflow-yolov4-tflite: YOLOv4, YOLOv4-tiny, YOLOv3, YOLOv3-tiny Implemented in Tensorflow 2.0, Android. Convert YOLO v4 .weights tensorflow, tensorrt and tflite*. 2021. URL: `https://github.com/theAIGuysCode/tensorflow-yolov4-tflite` (visited on 12/20/2021).

[37] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. “Online object tracking: A benchmark”. In: (2013), pp. 2411–2418.

[38] Adnan Brdjanin, Nadja Dardagan, Dzemil Dzigal, and Amila Akagic. “Single object trackers in opencv: A benchmark”. In: *2020 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*. IEEE. 2020, pp. 1–6.

[39] Martin Danelljan, Gustav Häger, Fahad Khan, and Michael Felsberg. “Accurate scale estimation for robust visual tracking”. In: *British Machine Vision Conference, Nottingham, September 1-5, 2014*. Bmva Press. 2014.

[40] Helmut Grabner, Michael Grabner, and Horst Bischof. “Real-time tracking via on-line boosting.” In: *Bmvc*. Vol. 1. 5. Citeseer. 2006, p. 6.

[41] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. “Visual tracking with online multiple instance learning”. In: *2009 IEEE Conference on computer vision and Pattern Recognition*. IEEE. 2009, pp. 983–990.

[42] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. “Forward-backward error: Automatic detection of tracking failures”. In: *2010 20th international conference on pattern recognition*. IEEE. 2010, pp. 2756–2759.

[43] Alan Lukezic, Tomas Vojir, Luka ˇCehovin Zajc, Jiri Matas, and Matej Kristan. “Discriminative correlation filter with channel and spatial reliability”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6309–6318.

[44] Davis E King. “Dlib-ml: A machine learning toolkit”. In: *The Journal of Machine Learning Research* 10 (2009), pp. 1755–1758.

[45] Ivan Culjak, David Abram, Tomislav Pribanic, Hrvoje Dzapo, and Mario Cifrek. “A brief introduction to OpenCV”. In: *2012 proceedings of the 35th international convention MIPRO*. IEEE. 2012, pp. 1725–1730.