RGBD Apple detection in mock orchards dataset using YOLOv5

NHL Stenden Lectoraat in Computer Vision & Data Science

Rick Lei

Supervisors: Maya Aghaei Gavari, Klaas Dijkstra

Abstract—Attaching a robotic arm on a drone and make it inspect apples in an orchard is proposed to be done autonomous. This approach uses simulated data to train and test the YOLOv5 object detect used for this application. This approach put together the following research question: Can simulated data be used to train the detection model and test the localization of apples in mock orchards? The following steps were taken to investigate whether this approach is a step in the right direction: collecting data, training the YOLOv5s model on this data, compare different approaches on YOLOv5s and localise the objects in this data with the best model for this purpose. The different approaches on using YOLOv5 were able to achieve equal mAP values. By using tiled YOLOv5s an accurate localization of the detected objects was achieved with a mAP of 90.9% on object detection. This concludes that simulated data can be used for this purpose. The next step is to examine how the simulated data performs on a real situation. And eventually have a drone fly towards the apples it is detecting.

Index Terms—Stereo-camera, YOLOv5, RGBD, Localization

1 Introduction

Drones are increasingly used to reach poorly accessible locations. They are able to arrive at these locations faster than humans can do [1]. Due to its agility drones can get to locations where human safety cannot be guaranteed. For example, emergency services using a drone to search for people that lay under the rubble of a collapsed building. This method of using drones has saved a significant number of lives already. At this time of writing, DJI states that there are more then 940 people saved by using drones to find them [2]. Besides the importance of drones for emergency services. Drones are also used for commercial purposes for example: a farmer that checks his crops by flying over it with a drone. Using a drone instead of a helicopter can save a lot of costs.

So currently, drones are being very effective for observing purposes. The drones do this through the cameras for the visual part and an e-nose for detecting gases in the air. This observing can already make a lot of difference in some situations. In 2020 a surfer was saved from a potential shark attack by an observing drone. The team at Surf Life Saving NSW detected the shark right before tragedy could struck and were able to save this mans life [3]. But is it also possible for a drone to do more than just observe? Innovation starts with an idea, MARS4Earth has this idea about a possible next step in using drones. This idea consists of attaching a robotic arm on a drone and make it interact with objects in its environment. To make this idea even easier, this is proposed to be done autonomously. Manually steering an arm to an object takes much effort, a person must guide the arm to the object by controlling the arm and the drone and at the same time avoid any obstacle. Making this autonomous will take the following steps: First an object must be detected in the 2D plane by using some sort of algorithm. With just this XY data in the frame of the camera an arm is only able to get the position in the 2D plane. To get to touching or grabbing the object the distance between arm and object is needed.

The main goal of this paper is to go through the smart vision aspects of this autonomous interaction between a drone and objects in its surroundings using simulated data. This will be done by taking the following steps: collecting data, training the detection model on this data and localise the objects in this data. After these steps, the object detector in combination with the distance will be able to give the correct XYZ-information of a detected object relative to the centre of the image it is detected in. This goal is converted to the following research question: Can simulated data be used to train the detection model and test the localization of apples in mock orchards?.

Rick Lei is a Mechanical Engineering student, studying a minor in Computer Vision & Data Science at the NHL Stenden University of Applied Sciences, E-mail: rick.lei@nhlstenden.nl.

Maya Aghaei Gavari is a researcher at the NHL Stenden Lectoraat in Computer Vision & Data Science, E-mail: maya.aghaei.gavari@nhlstenden.com.

Klaas Dijkstra is a lector at the NHL Stenden Lectoraat in Computer Vision & Data Science, E-mail: klaas.dijkstra@nhlstenden.com.

2 State of the art

2.1 Collecting RGBD data

The first way of collecting RGBD data is a light detection and ranging sensors or LiDAR for short is in combination with vision able to capture RGBD images. This is also used, in vehicles that can drive autonomously while analysing and measuring their environment a 3D LiDAR is used for this purpose. This camera is able to detect distances accurate to a few millimeters [4] in multiple planes instead of only one plane with 2D LiDAR. LiDARs in general are resistant to different types of light incidence [5]. There is one significant limitation for using a 3D LiDAR for this project. Cost, a 3D LiDAR is out of scope for this project. The second way of capturing RGBD images is by using a stereo camera, these are relatively cheap in comparison to a 3D LiDAR and are still able to calculate depth to a few millimetres significant [6]. These cameras consist of two separate cameras like humans sense depth with two eyes. The estimated depth is based on the triangulation of rays from multiple viewpoints. The performance of modern stereo cameras is almost equal to LiDAR lasers [7]. And therefore, an interesting way of measuring depth.

2.2 XYZ-position out of image

The distance of an object relative to the camera is something that is being researched by Chen Z [8] while this paper is being written. By using a stereo camera to detect and localize the objects a UAV can interact with objects it is detecting. In the case of Chen Z they also state that they improved standard YOLOv5 [9] to their YOLOv5_Tel by changing the feature pyramid network to a bidirectional pyramid network. They state that: The mean average precision value of their YOLOv5 version increased by values between 0.8 and 1.0 %, their model size and inference time also decreased. Their research

resembles this paper, both a stereo camera and using YOLOv5 for object detection. This paper will focus on real data with a real application instead of the blocks used in the paper of Chen Z.

2.3 Detection in orchards using RGBD

Fruit detection in orchards using RGBD images has been studied in the past. Tu [10] researched passion fruit orchards. By looking at the detection on RGB(Red, Green and Blue), Depth layer (Layer that contains the distance value of each pixel) and RGBD images Tu concluded that the performance of the faster RCNN model improved most by using RGBD images. The same result was concluded by a previous student that worked on the MARS4earth project as well.

3 Materials and Methods

This section describes the materials and methods used for this paper. For this project the object detector, YOLOv5 will be trained with a custom RGBD-dataset.

3.1 Mini-Orchard Dataset

The dataset used for this paper should meet the following requirements: Consist of RGBD images, all apples must be annotated, and the scenery must be similar to an apple orchard. Currently, there is no dataset that matches these exact requirements. Therefore, a custom dataset was created for this paper, the Mini-Orchard Dataset. This dataset consists of 500 RGBD images made in a controlled setup. The full process of making this dataset will be further explained in the next section.

3.1.1 Physical frame design

Generating a diverse dataset requires many different images. With images that contain multiple apples at different positions and different scenes in every image. This requirement was leading for the design of this setup. This design as shown in Figure 1 consists of the following parts: Grid and Grid Support. The materials used for this frame are 3D printed coupling pieces made from PLA and aluminium profiles. This setup does not require any fasteners or welding to assemble it. This makes it easy to assemble and disassemble and it gives the opportunity to use the parts for future projects as well. The Grid is designed in a way that the apples can have multiple XY-locations and Z-depth in one image. Therefore, it can generate more than 500 different images. The grid support is constructed so a drone can fly around the hanging objects without hitting the frame itself. This makes the setup usable for static and dynamic camera positions. For this paper only static camera positions will be used. To create an orchard-like data set, a background that resembles an orchard is added. A real image of an apple orchard was used for the background. This image was taken from the Apple_MOTS dataset [11]. This image consists of apple trees without apples on them.

3.1.2 Static camera position

This camera, a ZED Stereolabs stereo camera will be placed at a fixed distance from the frame as shown in Figure 2. This distance needs to be greater than 0.75 m because the camera detects depth from 0.75 m to 20 m.

3.1.3 Capture RGBD image

The ZED stereo camera was delivered with an API. This API and some additional code were used to capture one RGBA image and one depth(D) image. The desired output format is RGBD. This RGBD image is a combination of the two images, the RGBA layer captured by the left camera and the Depth layer captured with both cameras. To merge the RGBA and D layer, the RGBA image first had to be converted to RGB. The A in RGBA stands for alpha and indicates how opaque each pixel is. This alpha channel is removed by slicing without any consequences. The Depth layer is captured with both cameras. This results in occlusion that is displayed as a shadow on the Depth layer as shown in figure 3. This occlusion is used to calculate the depth values. After this conversion the RGB and D as shown in figure 4 can be stacked so an RGBD image is created.

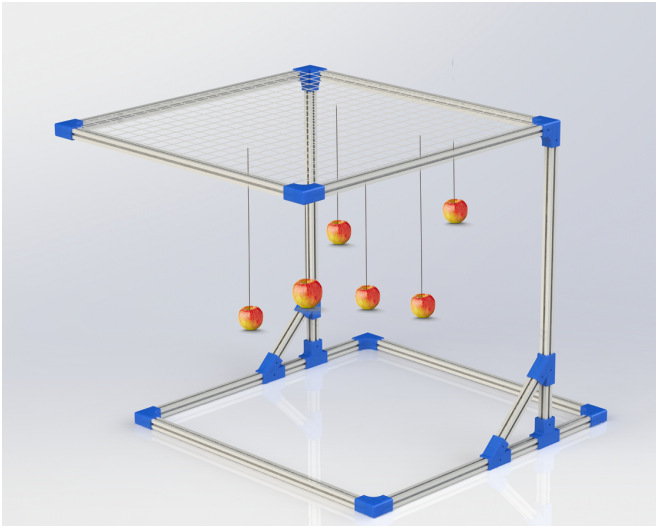


Fig. 1: 3D-view: Frame

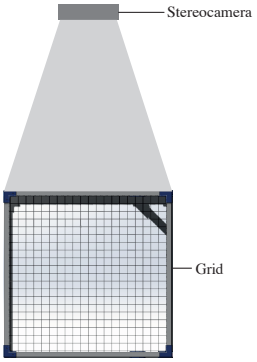


Fig. 2: Top-view: Setup

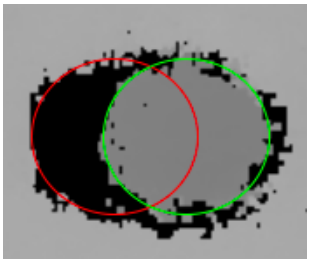
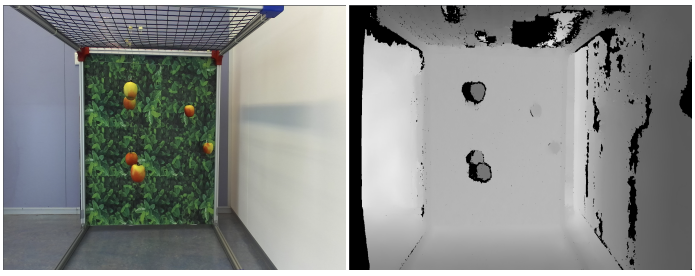


Fig. 3: Shadow



(a) RGB Image (b) Depth Image

Fig. 4: RGB and Depth image

3.1.4 Scenery

Only adding an orchard background to the frame would not be sufficient to resemble an apple orchard enough. The distribution of apples in the image is just as important. By studying real apple trees the apples are distributed across the images in such a way that the images look realistic.

3.1.5 Annotations

The YOLOv5 algorithm needs to be trained with annotated objects. All apples in this Mini-Orchard Dataset are annotated with Labelimg as shown in figure 5. These annotations were saved to XML to be fed to the YOLOv5 algorithm.

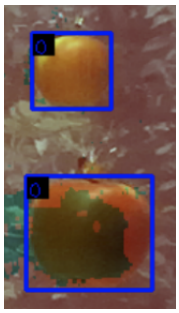


Fig. 5: Ground truth annotation

3.2 YOLOv5 Algorithm

For object detection and classification, the YOLOv5 algorithm [9] will be used. This algorithm will be trained with RGBD-images and its annotations. By detecting the apples in RGBD images makes that getting the 3D positions of these detected objects can be done very easily. The input data for this algorithm will therefore consist of RGBD images and their annotations. This data will be fed into the backbone of the algorithm: CSPDarknet. This backbone extracts the features out of the data. These features are then fused together by using feature pyramids as the neck: PANet. The Head: YOLOv3 layer, is putting out the final detection results by looking at the features. This results in class probabilities and bounding boxes. To be able to fit multiple applications the YOLOv5 has different sizes: small (YOLOv5s), medium (YOLOv5m), large (YOLOv5l) and extra-large (YOLOv5x). Where YOLOv5s uses the least amount of computational power and YOLOv5x uses the most. Because this model needs to be used remotely on a drone and this object detection process is only one of many processes that the drone needs to compute the smallest YOLOv5 model will be used for this paper.

3.2.1 Metrics

The trained models make predictions that can be True or False. If a model predicts an object rightfully it is considered a True (Right class) Positive (Detection). When the model makes a prediction that does not contain the desired object it is considered a False (Wrong class) Positive (Detection). The opposite of detecting is missing these objects. When an object is not detected it is called a False (Wrong class) Negative (No detection), if the model rightfully predicts there is nothing in the image or data it is called a True (Right Class) Negative (No Detection). True negatives do not occur in object detection. An object cannot be detected or missed when it is not in the image. Based on these outputs the following metrics will be calculated: Precision (1), Recall (2), F1-score (3) and mAP.

Where TP = True positive, FP = False Positive, FN = False Negative

$$Precision = \frac{TP}{TP+FP} \tag{1}$$

$$Recall = \frac{TP}{TP+FN} \tag{2}$$

$$F1\ score = \frac{2}{(\frac{1}{Precision} + \frac{1}{Recall})} \tag{3}$$

Mean Average Precision or short mAP is a metric that describes the average precision of the model calculated on all probability thresholds from 0 to 1. Average precision (4) is the area underneath the precision-recall curve. mAP is calculated by taking the mean of the average precision over all classes. This mAP gives a clear result of how the model performs over all classes.

$$Average\ Precision = \int_{r=0}^{1} Precision(recall) \tag{4}$$

3.2.2 Training Hardware

The training of the YOLOv5s model will be done on an Intel(R) Core(TM) i9-7960X CPU with 14 GB of memory and an NVIDIA GeForce RTX 2070 with 8 GB of VRAM, with Ubuntu 22.04.1 LTS as its operating system.

3.3 Preprocess data

Pre-processing the data before feeding it to the YOLOv5 algorithm can give some extra features such as reducing processing time or quality enhancement of the data. In this paper there are two types of pre-processing being used.

3.3.1 Resize

The first type resizes the full-size images (1920 x 1080 px) to a size YOLOv5s accepts. This way of pre-processing will be referenced as: R-YOLOv5s or simply Resized as shown in figure 6. By resizing the images some objects can become less visible, small objects become even smaller and can disappear. This is where the second way of pre-processing data can be useful, so called ”Tiling”.

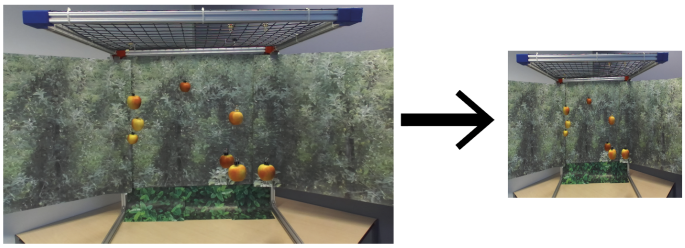


Fig. 6: Resize

3.3.2 Tiling

Tiling splits the full size image in a grid of smaller tiles that are able to be processed by the algorithm without resizing and losing small objects this is visualized in figure 7. This way of pre-processing will be referenced as: T-YOLOv5s or simply Tiled. This type can result in tiles with objects of interest on them and tiles without these objects. Tiling comes with two features that can alter what data the model will be trained on: Random Positive Tiling, this always feeds a tile with objects on them to the algorithm. The second feature, Random Tiling, feeds the algorithm tiles with or without objects.

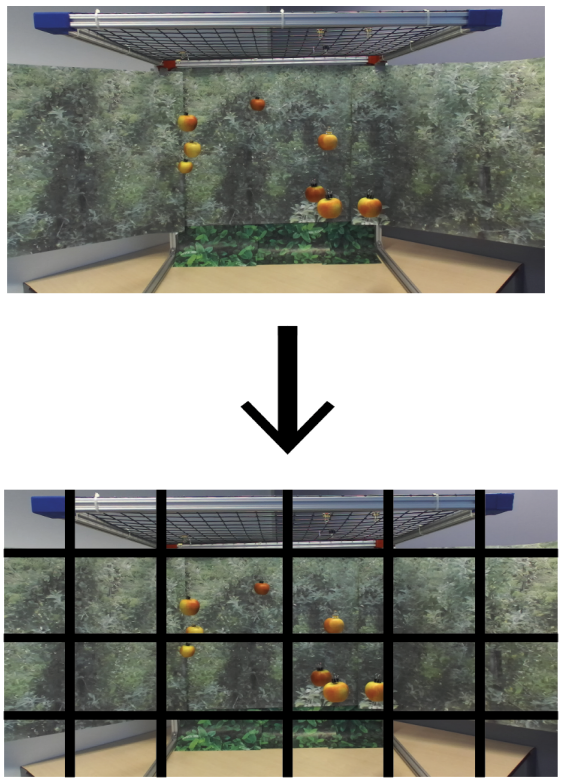


Fig. 7: Tiling

4 Experiments & Results

This paragraph describes what experiments were done for this research.

4.1 Experiment 1, Comparison

The first experiment will be about the comparison between R-YOLOv5 and T-YOLOv5 and the use of RGB and RGBD. This comparison will be done on the Mini-Orchard dataset. The results are expected to meet the studies mentioned in the state of the art section [10] . This experiment consists of a comparison of the following metrics: Precision, Recall, F1-score and mAP where mAP will be the leading metric. This experiment is executed under the following parameters as shown in table 1.

Table 1: Train settings

Epochs	30
Learning rate	0.001
Patience	20
Batch size	1
IoU threshold	0.5
Tile size	512 x 512 pixels
Number of tiles	4
Tile feature	RandomPositiveTiling

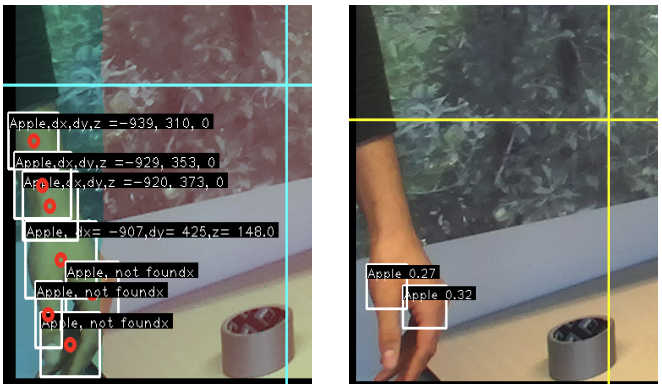
4.2 Results 1, Comparison

The results from this experiment are shown in table **??**. The models were trained with a max of 30 epochs. The model with a lower loss than the validation loss was saved as the best model. When looking at the mAP values of all the models they perform well. There are 642 apples in the test set, the trained models are able to predict at least 628 apples correctly. Tiled RGBD is the only outlier in these results. With a mAP of 90.8% it performs worse than the other models. This difference in mAP can be visualized with the comparison of Tiled RGBD and Tiled RGB displayed in figure 8. This figure shows that it has a problem with ignoring an arm that appears in some images. It is possible that this arm has never been seen before because of the RandomPositiveTiling feature. The arm is probably detected as an apple because it has a different colour and depth information than the background. This could be fixed by training with Random Tiling, so this arm has been seen before in training. To put this to the test the RGBD/Tiled has been retrained with the RandomTiling feature instead of the RandomPositiveTiling feature this result is shown in table **??**. This shows that by using this RandomTiling feature a mAP equal to the other models can be achieved.

Mini-Orchard Dataset

Method/ Datatype	Precision	Recall	F1 score	mAP
Resized/RGBD	**99.1%**	98.3%	98.7%	**90.9%**
Tiled/RGBD	97.9%	98.8%	98.4%	90.8%
Tiled/RGBD*	98.4%	97.8%	98.1%	**90.9%**
Resized/RGB	98.8%	98.6%	98.7%	**90.9%**
Tiled/RGB	98.3%	**99.1%**	98.7%	**90.9%**

Table 2: * Random Tiling instead of Random Positive Tiling



(a) RGBD Tiled (b) RGB Tiled

Fig. 8: Difference RGB/RGBD

4.3 Experiment 2, XYZ-locations

The aim of second experiment is to obtain the 3D position relative to the camera. This will be about the location of the detected apples in the image with the use of YOLOv5s. By using the RGBD-Dataset as input for the T-YOLOv5 the XYZ distance relative to the centre of the image can be shown in Tensorboard and saved to a TXT-file. With the results of experiment 1 the use of T-YOLOv5 on RGBD data is inaccurate. Because of the features that come with using T-YOLOv5 and RGBD this will still be used for this experiment.

This experiment is done under the following parameters:

- T-YOLOv5
- Mini-orchard dataset
- Image dimensions(XYZ): 1920:1080:255

4.3.1 Convert depth value to 8 bit

The depth value captured by the stereo camera is in mm. This is converted in values between 0 and 255 to achieve a similar range of values over all 4 channels. This was done by the following equations:

$$new\ value = \frac{value}{maximum\ distance/255} \tag{5}$$

The maximum distance is variable, this will result in a more uniform distribution of the 255. In this experiment the maximum distance is set at 3000 mm. The real maximum distance measured from camera to the background of this setup has a value of 2020 mm. Setting the maximum distance at 3000 mm gives a margin. It can occur that the stereo camera detects false depth values. All distances greater then 3000 mm are therefore set to 0. So, these errors will not interfere with the resulting distance.

4.3.2 Retrieve Depth detected object

An object is detected in the RGBD image. To retrieve the depth value of this detected object the depth layer is sliced to the dimensions of the detected bounding box. After the slicing is done the median value of this sliced array [10] is returned as the depth value of this object.

Fig. 9: Horizontal array slicing

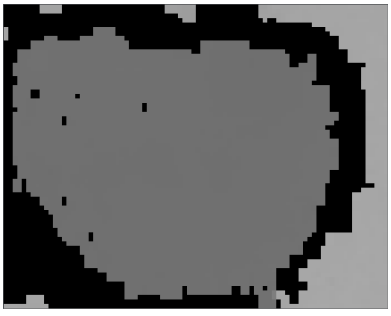


Fig. 10: Vertical array slicing

4.3.3 XY-pixel values to mm

The following formula is used for calculating the xy distance in mm:

$$Dimension(mm) = \frac{Distance\ (mm) * (Dimension\ (px) * Sensor\ dimension\ (mm)}{Image\ dimension\ (mm) * Focal\ length\ (mm)} \tag{6}$$

The specifications of the Zed camera [6] and the distance calculated by the camera are used to determine the xy distance between objects.

4.4 Results 2, XYZ-locations

This experiment results in the XYZ locations of the detected objects. By using the stereo camera to capture the RGBD images the detected objects can have a 3D location measured from the center of the image. This is added to the visualisation in Tensorboard and shown in figure 11 a full size image is available in the appendix. A TXT file is also saved with the XYZ positions relative to the center of all the detected objects per image.

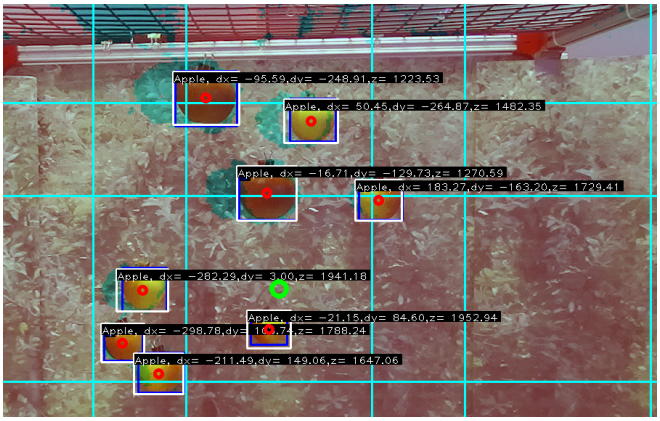


Fig. 11: Results experiment 2

4.4.1 Wrong depth values

It can occur that the wrong value is displayed instead of the actual depth value. This was discovered because some apples were having a distance of 0 in the images. This zero can only occur when the detection would be inside the black border around the image as shown in figure 13. Or if a value can not be calculated by the camera. The reason behind this wrong value has to do with how the dataset was created in the first place. Some apples were still swinging between the capture moment of the RGB and the Depth image. This resulted in the shifted values as shown in figure 12, The red circle is the apple location in RGB, the green circle contains only some depth points because of the swinging but not enough to make a good depth estimate. This problem will also occur when a drone is used to capture this data. Drones are not able to hover perfectly still in the air.

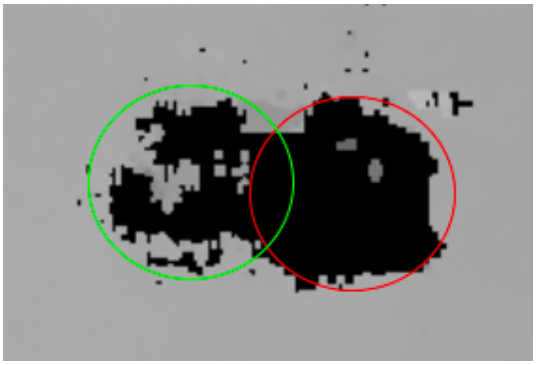


Fig. 12: Shifted depth

5 Discussion, Conclusion & Future Work

This section describes the discussion, conclusion, and future work of this research.

5.1 Discussion

To be able to localise apples in simulated data the best model was needed to make the best detections. The comparison between the different models gave a clear result, all models/features are able to perform equally. The only outlier is RGBD Tiled using Random Positive Tiling. This model detected more apples in an unseen object than other models. For the second experiment RGBD Tiled with Random Tiling was used because of the advantages it comes with such as: being able to keep small details more visible and the additional Depth data that is included in the RGBD images. This experiment proved that by using simulated data the detection model is capable of localising apples in RGBD data.

The first experiment, the comparison, also showed that the difference in depth is used to determine if an object is an apple or not and therefore detect an unseen object with a different depth value as an apple. These false detections lower the overall result of the model. But is detecting an unseen object worse than missing it? In a project where the detection is done by cameras on autonomous drones it could be good thing to detect more objects than only apples. But this is more interesting for applications like object avoidance than the detection of apples. And therefore out of this scope. The comparison turned out different than expected. The expected outcome was that resized RGBD would outperform the other approaches. The results of experiment one show a different result. The low difficulty of the test set could be one of the reasons why all the models perform equally. This also shows where the limit of this research lies. The real application is not tested in this research. This could perhaps give a more clear result of what model is best for the real situation. Although the models are not tested on real data the results are still contributing to answer the research question. The second experiment still shows that the RGBD data captured with the stereo camera can be used to localise the detected objects. This research also gives a good guide of what problems can occur with capturing RGBD data and the usage of this type of data. It showed that when apples will not always hang perfectly still. This moving 3D position is hard to validate with just one image. Would continuous footage(video) or just multiple pictures of one scene help with making the depth value more valid? The chance of getting a right calculation should be higher with multiple images than with one image.

5.2 Conclusion

This paper researched if simulated data can be used to train and test the localisation of apples in mock orchards. Based on the results of this paper it can be concluded that using the Tiled YOLOv5s detection model on the Mini-Orchard dataset results in an accurate localisation of the apples. The RGBD data captured by the stereo camera had the most impact on this result. The additional depth was used to make the 3D position of the detected objects complete. The use of tiling is still questionable. It preserves small objects instead of making them smaller. But are the benefits bigger than the drawbacks?

5.3 Future Work

This research has a clear conclusion but also stated that there is more future work to be done. The YOLOv5s model is trained on simulated data but not tested on the real application. The simulated dataset can be improved to fit the real application. This will result in more data and give the opportunity to test if the depth calculation improves by using more data of the same scene. The steps in this study are also applicable to other objects, this also ensures that future work does not have to be in combination with a drone and an apple orchard. The applications are endless for this technology.

Acknowledgements

- This project is financially supported by Regieorgaan SIA (part of NWO) and performed within the RAAK PRO project Mars4Earth

- Henk Lei Groente & Fruit for sponsoring the apples used to create this Mini-Orchard dataset.

References

[1] Andreas Claesson, Anders Bäckman, Mattias Ringh, Leif Svensson, Per Nordberg, Therese Djärv, and Jacob Hollenberg. Time to Delivery of an Automated External Defibrillator Using a Drone for Simulated Out-of-Hospital Cardiac Arrests vs Emergency Medical Services. volume 317, pages 2332–2334, 06 2017.

[2] DJI. Dji drone rescue map. site=https://enterprise.dji.com/drone-rescue-map/, organization=DJI, 2022.

[3] Jack Guy. This surfer had no idea how close he came to a shark – until he saw the drone footage. *CNN*.

[4] Sick AG. Mrs1104c-011010, 2023.

[5] Lidar-based semantic perception for autonomous vehicles.

[6] Stereolabs. How does zed work?, 2023.

[7] Alberto Broggi. A closer look at lidar and sterovision. site=https://www.ambarella.com/blog/a-closer-look-at-lidar-and-stereovision/, organization=Ambrella International LP, 2020.

[8] Zhangyi Chen, Xiaoling Li, Long Wang, Yueyang Shi, Zhipeng Sun, and Wei Sun. An object detection and localization method based on improved yolov5 for the teleoperated robot. volume 12, 2022.

[9] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, Zeng Yifu, Colin Wong, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, Victor Sonck, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Dhruv Nair, Max Strobel, and Mrinal Jain. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation, November 2022.

[10] Liu Zhuang Chen Zheng Wan Xue Tu, Pang. Passion fruit detection and counting based on multiple scale faster r-cnn using rgb-d images. 2020.

[11] Stefan de Jong, Hilmy Baja, Karsjen Tamminga, and João Valente. Apple mots: Detection, segmentation and tracking of homogeneous objects using mots. *IEEE Robotics and Automation Letters*, 7(4):11418–11425, 2022.

A Appendix

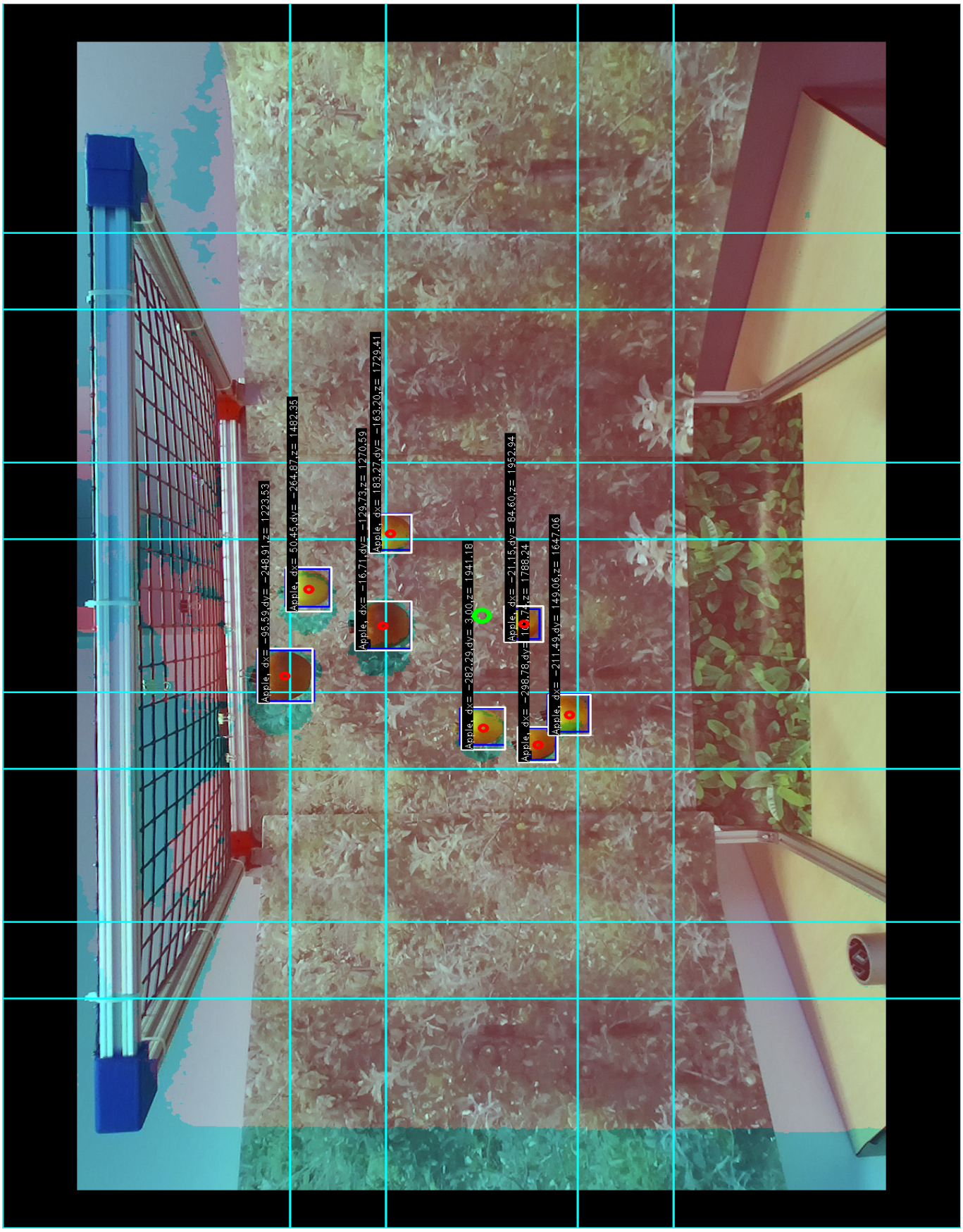


Fig. 13: Full size image, result experiment 2