UAS Recruitment Task 2

After seeing the problem at hand, I thought of two approaches.

What I learnt about the model weight given: The fruit classes for the custom dataset trained model weight were Class 1 for Yellow Fruit, Class 2 for Purple Fruit and Class 3 for the plant itself. I found these out by testing the images and printing the class labels and confidence scores on the test images with my python code snippet file named: 03.py. Also while trying to do so I found that the model used as base for the custom training is yolov8 version as the pandas method for bounding box retrieval wasn’t working on it, which I read used to work till yolov5 and then when I tried the method for box retrieval for yolov8 one it worked.

My first approach thought was to just crop the fruit classes bounding box from the whole image. And then compare between the ones from the front and the back. But then I came to the realization that it seemed the fruit shapes were identical and that its of significance but the plant images were identical too. And I was going to have to get the coordinates of the bounding boxes from the front of the image and compare them to the coordinates from the back image anyway. So I changed my approach to the second one.

My second and prevailing approach was to:

* Get cropped images of the plants only and then proceed further such that the image sizes were now same and I can just use direct coordinates when I use yolo on the cropped images of plants to detect fruits. As earlier it would have been difficult as the front image and back image didn’t have the plant locations aligned to the same coordinates. To make the cropped images I just used yolo and conditional statement to get the bounding box coordinates if the class id was 3.
* After getting the cropped images I used a function to keep the cropped image object size same without scaling and just fill the else with a colour. This was a later addition while troubleshooting my code, for solving the error that it was not detecting the fruits, as the custom trained weight was trained to work only with images of size 640x480 and only with objects of the original size they are in the given example images. I used ai to help me with writing the simple and tedious image resizing and filling code.
* The resized cropped images are then appended to a list and I do the likewise with the back images too.
* Then I use a mirror function which uses open cv flip method to flip the image and then also create another list of back cropped images of plants flipped and in reverse order. This is done so that the list of cropped plant images from the front and the back contain the same plant images in the same order.
* I defined a function to check match from the cropped images of the front and back. In each plant image from front and back the function iterates through all the bounding boxes, while using a conditional statement to avoid the class id 3 bounding boxes. Then the retrieved coordinates of the bounding box are added to a set in the form of tuple from both perspective of image lists. Thus, avoiding duplicate addition of the same bounding box coordinates.

Advanced Version: In the final with threshold matching.py file I have used a Euclidian function that determines the Euclidian distance of two coordinate points and returns it. Then I have used a threshold variable and a conditional statement to allow the addition of a new bounding box coordinates into the set only if there isn’t already a bounding box coordinate in it which is lesser than the threshold distance from it (the new bounding box coordinate).

* Summary of the check matching function is to just not allow the addition of the same or near coordinates of bounding box. And as the images are of the same plant and flipped to align the coordinates too thus it is able to check for the duplication counting of the fruits. The final with a threshold.py is a little better theoretically then the final with direct matching.py as it counters for little error and the threshold can also be adjusted if need be.
* I have a function to just check the existence of 2nd or 1st class id in the class label list which is retrieved along all the bounding boxes in the two lists of plant images. I noticed that all the example images had only a single type of fruit on it so I didn’t make this function more complex and use it to determine the count of each color fruit on a plant. It could be done in the check matching function to retrieve class id too and append them into the list with the threshold conditional statement. So this function just simple checks what one kind fruit is available on the plants are.
* Then I simultaneously iterate through, each same plant from front cropped images list and back crop and flipped images list, and use the check matching function to get the set of unique fruit bounding boxes from both perspective images of the plant. The length of this set for each plant is used as a count and the count is appended into a list for each plant. Then the elements of the list is summed to find the total count and printed along with the presence of the colour of fruit.

Note: I am submitting all of my progress files and snippets and everything I used AI in has ai written in its name. THE APPROACH AND ALMOST ALL THE IMPLEMENTATION OF IT WAS WRITTEN WHOLELY BY MYSELF.

Note for respected UAS DTU Seniors: I sincerely applogize for my late submission and also due to being late I wasn’t able to refine my code further. I am to blame for this and I hope you can forgive my due diligence.