Investigation of Alternative Distance Detection Using a Single Camera Warped Lens

Teague Sangster Simmon Chura Elijah Pearce Ryan Renales Jwann Sy
San Diego State University
tsangster3359@sdsu.edu squan3671@sdsu.edu epearce4653@sdsu.edu rrenales1971@sdsu.edu jsy@sdsu.edu

Abstract

This project explores using a modified camera and an AI model to measure object distance using distorted images, offering a cheaper alternative to systems like LIDAR. We applied a blur to an existing dataset to simulate distortion and fine-tuned a YOLO V3 model to detect objects under these conditions. While YOLO V3 has shown strong performance on clean datasets, our results showed a noticeable drop in accuracy with distortion, even after fine-tuning. Although the method was unsuccessful, it highlighted key limitations in using older AI models with imperfect visual input and pointed to the need for more advanced or specialized solutions.

1. Introduction

Accurate measurement of distance between a camera lens and surrounding objects is a necessary detail for various modern computer vision applications. Distance estimation can be found in prevalent systems such as self-driving vehicles, collision avoidance systems, and surveillance systems. The traditional approach for distance estimation relies on LiDAR systems or stereoscopic cameras, which gather 3D spatial information. While these systems are typical and practical in today's technological use, they present significantly expensive costs. This issue exposes accessibility challenges to many users. Since these conventional systems require external hardware such as laser emitters, scanners, or multiple cameras, they also may present a portability problem.

This project examines how real-time distance estimation and object detection can be performed using only a single monocular camera. This task is challenging since monocular camera feeds do not inherently provide in-depth information. The goal is to test whether object detection models can still function under visual distortion to simulate depth without relying on external hardware. Success in this would present a simple and affordable alternative to traditional depth perception systems.

As technology continues to move toward smaller, more cost-efficient, and real-time systems, there is growing interest in camera-based setups with constrained resources and components. Single cameras are lightweight and an accessible alternative. They are especially valuable for mobile devices like drones or smartphones. This project evaluates whether combining a monocular camera with the right AI model could provide a functional substitute for more complex depth perception systems. This approach may expand to low-cost, portable computer vision systems for a broader range of users and applications.

1.1. Related Work

Single-camera distance estimation and object detection are two relevant research areas in computer vision. The paper titled "Real-Time Distance Measurement Using a Modified Camera" by Liu et al. [1] presents that accurate distance estimation can be calculated from a single camera without assistance from other stereo vision or LiDAR systems. Their approach uses a still camera with an inclined image sensor, causing the image to defocus and blur in some regions of the image. Alternatively, the image is sharp at a specific point, which varies with distance. The in-focus area is the maximum sharpness value, which can determine the image plane. The image distance is then known, and object distance can be calculated with the lens formula. This method demonstrates a cheaper solution using a single camera feed through careful calibration and geometrical calculations.

A more recent paper titled "Dist-YOLO: Fast Object Detection with Distance Estimation" by Vajgl et al. [2] integrates a YOLO object detection model with distance estimation techniques based on known camera parameters. Their method demonstrates real-time object detection and distance estimation, making this application more flexible for dynamic environments and everyday detection tasks. Dist-YOLO does not rely on image focus like the previous method but obtains distance estimation from the object detected size in an image.

This project builds on the strengths of both methods but ultimately highlights the difficulty in combining them. While we could get each component working individually, a modified monocular camera for theoretical distance estimation based on blur [1] and object detection with distance estimation using Dist-YOLO [2], integrating the two proved unsuccessful. The modified camera's image warping interfered with Dist-YOLO's detection, and our camera setup could not reliably produce usable depth data, requiring us to generate synthetic data instead. Our findings suggest that while each method shows potential independently, their combination is not a practical solution.

2. Dist-Yolo Modified Approach

Previously, monocular distance detection utilized unaltered visual data to train visual detection models for distance estimation. We wish to utilize a modified camera version using the characteristics of the modified camera to allow the model another layer of information it can use to gauge distance of a recognised object. For training we will use a synthetically modified image dataset of cars, cyclists, and pedestrians with LiDAR distance measuring. With the goal to compare the distance and object recognition accuracy using a modified version of previously tested method of visual detection Dist-Yolo[2]. For object detection we will be using a Yolo-v8 model trained to look for cars, cyclists, and pedestrians as classes, with their distance as an additional parameter of training, on a modified dataset that has been previously tested. We will compare the results and examine if this is a viable method of distance recognition.

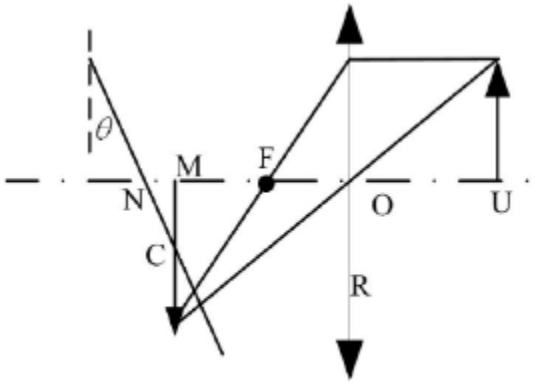


Figure 1. Skew sensor

2.1. Camera Modification

To compute the distance of an object utilizing a single modified camera takes advantage of geometrical truths. Given a lens at point O, an object at point U, an image sensor at point N inclined at an angle of σ, the object can project an image at plane M. The intersection of plane M and the image sensor is point C. A horizontal line that passes through point C is the only place where the image created is in focus, all other places the image projected is blurred. Figure (1)[1] is a visual representation of the described geometric example. The image distance formula can be expressed:

$$\overline{OM} = \overline{ON} - \overline{NC}sin\theta \quad (1)$$

this will allow us to find the object distance by utilizing the lens formula:

$$\frac{1}{\overline{OM}} + \frac{1}{\overline{OU}} = \frac{1}{f} \quad (2)$$

Where f is the focal length of the lens. By substituting the image distance formula into the object distance formula we can find the object distance as:

$$\overline{OU} = \frac{f(\overline{OU} - \overline{NC}\,sin\theta)}{\overline{ON} - \overline{NC}sin\theta - f} \quad (3)$$

The final unknown parameter of this equation is the length of $\overline{NC}$, which can be found using a definition function. Suppose that point N is the middle of the image sensor and point C is the clearest horizontal line of the image projected onto the sensor. This means that the farther from point C a pixel is, the less definition or degree of focus it is. There are several definition functions that can be used, Signal Noise Ratio, Frequency Selective Weight Median filter function, or Tenegrade function, but the simplest would be to use a Sum Modulus Differential function (SMD). SMD simply calculates the sum of the first order grey difference of two adjacent pixels on horizontal or vertical direction.

$$SMD_x = \sum_x \sum_y |f(x,y) - f(x,y-1)|$$

$$SMD_y = \sum_x \sum_y |f(x,y) - f(x+1,y)|$$

where $f(x,y)$ are the grey value of the image pixel at point (x,y). The maximum value of D(m) = $SMD_x + SMD_y$ is the sensor-row number that the definition function "climaxes" and is the point of most definition or point C, where x = mth row of the image and 1 < y < max width of the sensor resolution. Then NC can be found:

$$\overline{NC} = \frac{(m_{max} - h_{resolution})}{w_{resolution}} * h_{camera} \quad (4)$$

By plugging the length of $\overline{NC}$ equation into the object distance formula, we are theoretically capable of finding the distance of an object using only a single modified camera.

2.2. Synthetic Blur Filter

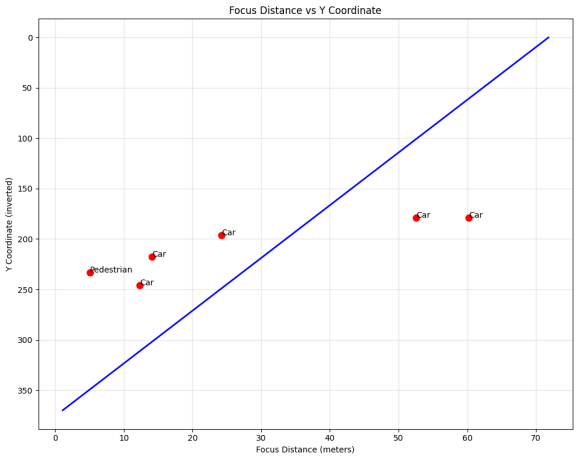
In order to implement the modified camera approach to previously trained data of the original Dist-Yolo model, we will add a synthetic blur to our image data, utilizing a two pass blurring and object segmentation system using a distance map.The two pass system begins with generating the sigma map based on the distance map of the data image. The Distance map is a data structure that contains the pixel distance from the camera created during data collection. This distance should correlate with the definition of that pixel, allowing the creation of a sigma map that will be used to blur each pixel by the correlated definition using a spatially varying gaussian blur over the sigma map. Each sigma value is defined as:

$$\sigma(d) = \{0, \quad d \leq d_{start} \quad (5)$$
$$\{(d - d_{start}) * g_{rate}, \; d > d_{start}$$

with d_{start} is the distance where there should start a blur and d_rate is the rate of blur per distance. Once the sigma map is generated we can pre-blur the background layers using a gaussian blurring algorithm. Inverting the foreground mask keeps the foreground objects sharp for Pass two. Pass two generates a foreground sigma map, with

$$d_{foreground\ start} = d_{start} * h_{depth\ of\ field} \quad (6)$$

allowing for images to be too close to the camera and thus also out of focus. Then the foreground is blurred using the foreground sigma map, creating two images, the blurred foreground and the blurred background. The two images are then combined because the sigma mapped pixels for the foreground and background should be exclusive from each other, allowing them to be merged into a single image.



^^^^^ This is an actual example of the synthetic data being created

3. Experimental Results

3.1. Phase 1 Results

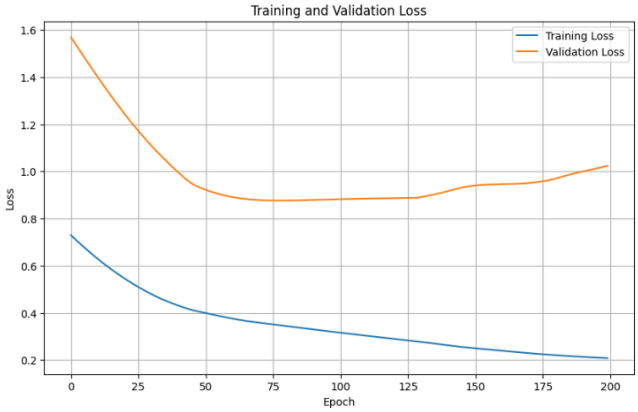


Table 1: Training and Validation Loss Graph. Note that there is a sizable difference between training loss and validation loss. Also that validation loss is increasing at the end and training loss is decreasing at the end. This suggests that the model is potentially overfitting to the training data.

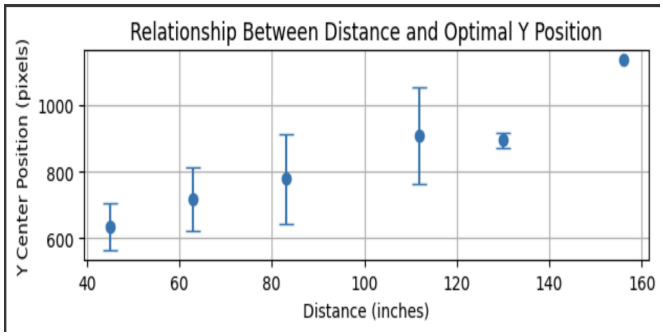


Table 2: Line Chart with STD between predicted distance and Y position of the object on the image. Note that there is a high amount of variance in estimating an image predicted to be in the midrange of distances. This suggests there is bad performance on estimating midrange distance.

Image 1a: An example of the camera focusing on the object(ArUco marker).

Image 1b: Canny edge detection applied to Image 1a. Note that the edges of the object(ArUco marker) are captured as well as some edges of the handrail. The inclusion of the handrail edges suggests that the camera is focused on an area rather than the intended target.

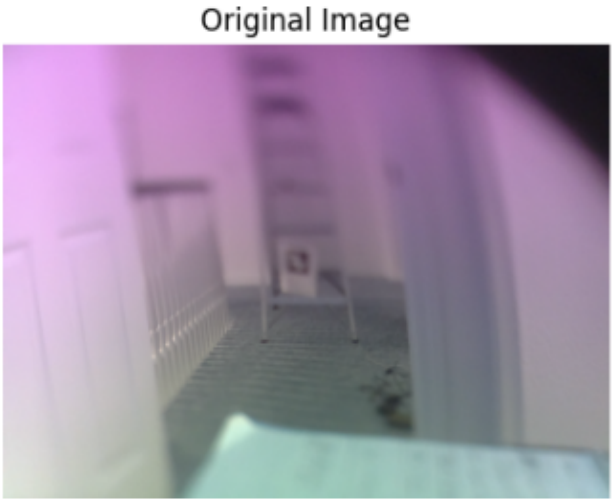


Image 2a: An example of the camera not focusing as intended.

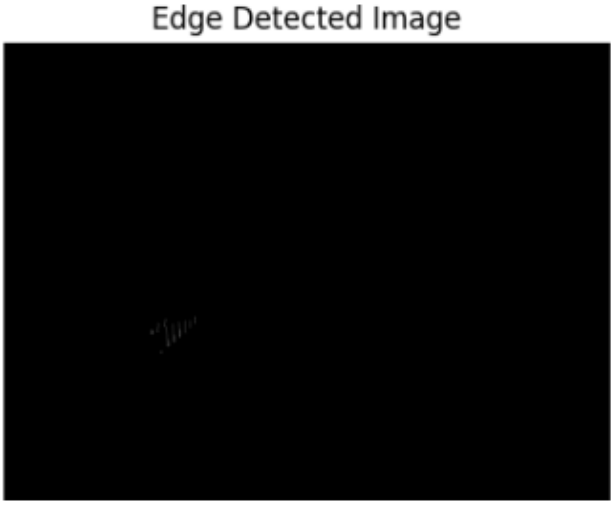


Image 2b: Canny edge detection applied to Image 1a. Note that the object's(ArUco marker) edges are not captured and only the edges of the handrail are being captured. It is suggested to use synthetic data as the camera is not working as intended for an upscaled implementation.

3.2 Phase 2 Results

Image 3: An example of the model labeling the distances of the object in focus. The green bounding boxes represent the ground truth label. This is the actual box of the object and what should be detected. This is found pre-blur and pre-training and validated by a person for each picture. The red bounding boxes represent the YOLO model's predicted distance of that object. Note that the model is mistaking two cars as one object but does accurately label the distance of the further car(a predicted 25.1 m vs the actual 24.4m).

Image 4: An example of the Blurred YOLO model failing. Here the green boundary boxes represent the ground truth labels (The actual objects). The red boundary boxes are where the blurred model detect objects to be. This photo emphasizes the complete failure of the Blurred model as the object it was supposed to find was correctly unfocused and it should have had no issue finding it, however this was not the case.

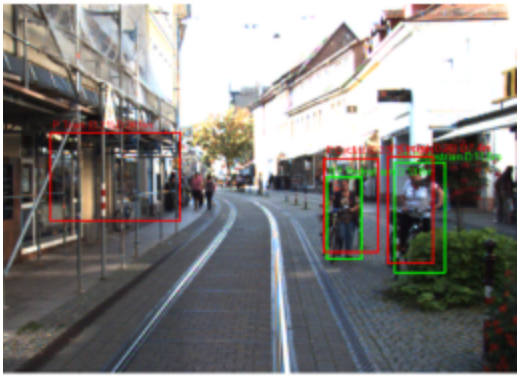


Image 5: An example of Tuned Dist-YOLO correctly finding two people and their distance but also finding an object that doesn't exist. Overall this is a good performance by the Tuned Dist-Yolo model.

Image 5: Key for the following three graphs

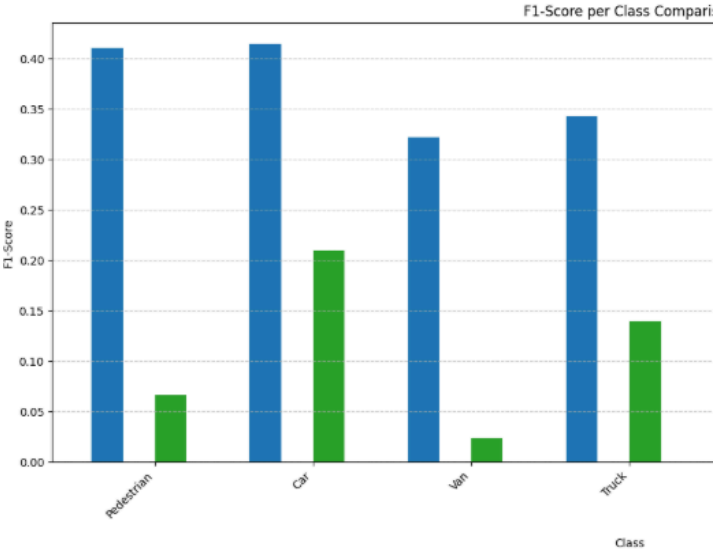


Table 3a: F1-score for all the objects tested. The YOLO model was also tested versus people sitting, cyclists, and trams which was not included here to fit the graphs. It is worth noting the people sitting were the worst performing for all three graphs. As expected, there are no scores for the blur only images as objects would not be able to be detected. The model performs worse in the tuned blurred images compared to normal images, with an exception regarding trams(not depicted here).

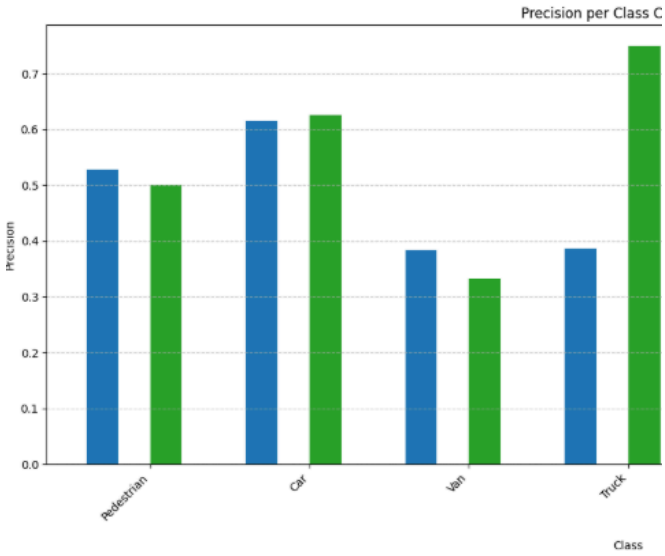


Table 3b: Precision for each object tested. The YOLO model performs the same and in some cases better in avoiding false positives between non-blurred and tuned blurred images.

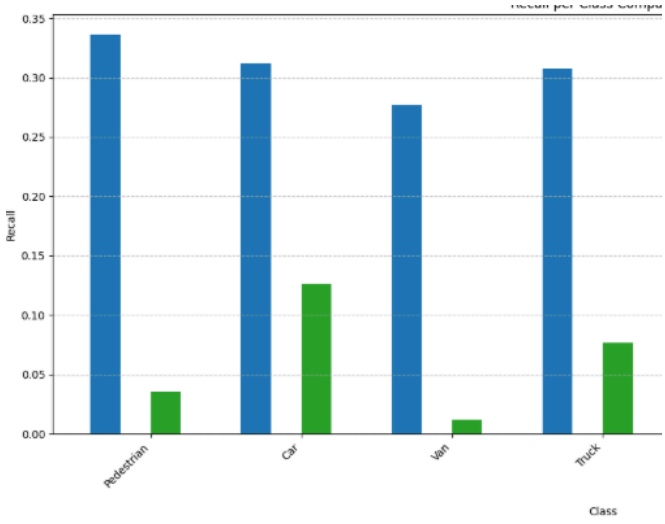


Table 3c: Recall for each object tested. The YOLO model shows that it had terrible performance in tuned blurred images in attempting to avoid false negatives, which explains the low F1-scores from Table 3a.

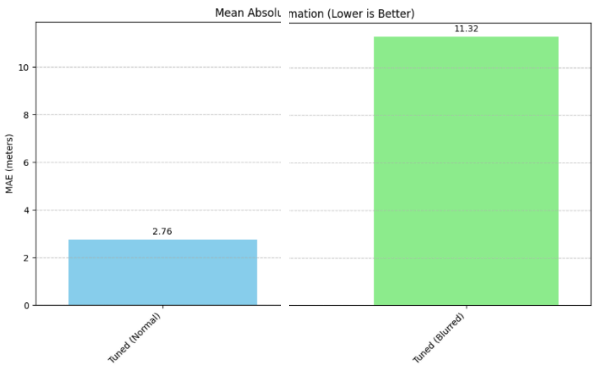


Table 4: Mean Absolute Error for Distance Estimation. YOLO makes negligible errors on normal images but makes noticeable errors on tuned blurred images.

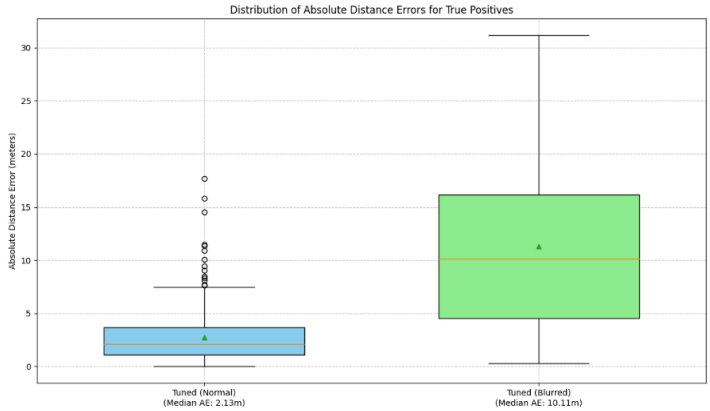


Table 5: Distribution Chart of Absolute Distance Errors for True Positives. The YOLO model keeps a tight distribution with a number of notable outliers for normal images but has a wider distribution for the tuned blurred images. The latter is likely due to the blur causing the outliers seen on the normal images to be part of the more general target objects rather than actual outliers.

4. Discussion and Conclusions

4.1. Findings

This project's goal is to see if camera modification could be used to replace current distance and object recognition capabilities that other systems offer. We wanted to see if we could provide those outcomes without compromising the functionality of the camera for other computer vision applications. This project's goal was to test if we could modify a YOLO V3 distance estimation model to accurately predict distance from a distorted image captured by a camera lens offset by a 45 degree angle. Although a distance detection model using YOLO V3 was already proven to work on a dataset of 7000 undistorted images with 89 percent accuracy from the [2] paper. We wanted to push the model to its limits by fine tuning it and testing its distance detection capabilities

when trained off intentionally blurred images using the camera offset technique from the [1] paper.

We started this project by first attempting to secure a dataset to train the models off of. We originally modified a telescopic camera by offsetting the lens by a 45 degree angle to introduce blurring. Since this did not produce usable data, we created an algorithm that applied a gradient blur to all 7000 images of [4]'s dataset to simulate what the modified camera should have captured for a synthetic dataset. We then modified a YOLO V3 distance estimation model by changing the model to account for the distance labels from the KITTI dataset it will train off, updated the preprocessing function to handle the extra distance dimension and changed the confidence_loss weighting to prevent overconfidence/underconfidence from occurring. We then trained the modified model using our synthetic dataset of 7000 images and trained the unmodified model using the unedited warped images from that same dataset. Because both datasets were the same just with visual distortions applied to one, we were able to use the same ground truth labels for both.

We trained 2 models on the same dataset, as it would allow us to determine with certainty if detection changes were from camera modifications. It is important to state that both models used started with the same weights at the start of fine-tuning. For clarification, these 2 models are firstly, the fine-tuned version of the original weights from the first paper on our dataset, and secondly, the same weights but fine-tuned on our blurred dataset.
After comparing both models, we have come to the conclusion that the modified model trained on the synthetic data was no longer functional. We believe this is due to the loss of confidence (how it determines what something is) because of image distortions. In comparison, the unmodified model had a success rate of 31%. The modified model was unsuccessful at object detection and distance estimation to the point where it did not have any correct detections to show on the graph. (For control purposes, confidence levels required to claim a detection were kept the same for both models) Although the blurred model was trained to detect on blurred images, it failed regardless of if it was provided a blurred or unblurred image for testing.

The true incompatibility of our modified camera with object detection workloads is highlighted by how our blurred model failed to be confident enough to detect regardless of what image set it was given. Regardless of if it was asked to predict on the modified or unmodified version of an image, it failed every single time. In comparison, the model tuned on non-blurred data was able to detect for both, albeit much, much worse on the blurred/modified images. This is because objects out of the *correct* focus distance in an image would become warped and sometimes lose so much quality that tuned YOLO just couldn't "see"/detect them anymore.

4.2. Successes and Failures

Our project's first failure occurred when we attempted to gather our own dataset. The original plan for this project was to gather real data using the approach of offsetting a camera lens by a 45 degree angle as described in Liu Xiaoming, Qin Tian, Chen Wanchun, and Yin Xingliang's research "Real-time distance measurement using a modified camera". We chose to recreate their research to not only validate their attempts but try to have a real-time system based on theirs.

This is where our first failure occurred. We originally believed it was the characteristic of our camera and how it differed from their study. However, looking at their sample results, even their most in focus region was blurry and low resolution. This issue only paved the way for camera usability concerns. In our version which had non-lab-perfect conditions, warping was intense. There was the expected spatial information warping but also unexpected color distortion caused by the lens. Any color distortion meant the ability of our camera to be a functional *camera* was instantly limited.

This initial setback turned out to be a major failure. Even while using Aruco tags and heavily preprocessing the image using grayscale and sharpening techniques, most images were unusable for distance detection. The reason why our implementation of the 45-degree camera failed can be directly linked to hardware limitations.

The research we followed used a camera with a very shallow depth of field and performed experiments within a foot of the camera lens. This is satisfactory for their test case but because we wanted to apply this research to computer vision, we had to use a different lens. Our application tried recreating the phenomena using a telephoto-type lens at much larger distances, up to multiple meters away. The combined initial limitations of sharpness in the first paper, combined with color distortion from imperfect conditions and warping from using a lens with any general application, combined to lead us to a dead-end. As such, it became time to pivot. By creating synthetic data to simulate a "perfect" modified camera, we were able to test AI based integrations.

Once we created our synthetic data and modified YOLO v3, we found ourselves with exploding loss values. Since our main goal was to fine-tune the YOLO V3 model, we only intended to change approximately three layers of the network so it could learn distance detection for our synthetic data. While training the model we forgot to freeze the rest of the layers in that model, so just as quickly as it started to show potential, it started failing. In our second run, we found that our confidence levels were being emphasized too much in our loss and as

such, our model found itself with another failed training run. We believe this confidence loss was due to the restructuring of our YOLO-dist model, which was originally intended to detect 8+ possible objects.

By collapsing the possible classes down to 3, it helped us simplify training but meant that confidence really wanted to update its weights. (Folding was done like this: truck was folded into the car, and the seated bicyclist into bicyclist. More details are beyond the scope of this section and are well documented in the Github) We overcame this final hurdle by de-emphasizing the importance of updating confidence in calculating loss. A dramatic reweighting of 1 * .1 forced the last 3 layers to focus their training on distance rather than confidence. These changes turned out to be exactly what we needed, so our third round of training for both models was successful and used in our predictions.

Ultimately our largest failure was that blurred images just aren't good to train on. As mentioned above, the model trained on blurred images performed horribly. Blurred YOLO V3 trained off of synthetic data lost all detection ability to the point where it could not detect distance even from an unblurred image. (Proven by the image below)

Our project's singular success was that our fine-tuned distance estimation model was functional as long as it did not try to incorporate the modified camera. (Proven by Image 5)

4.3. Future Work

In the future, we will try to adapt the model to a newer version of YOLO, such as YOLO V11. YOLO V3 was chosen due to its simplicity and the amount of groundwork already publicized in tuning it. This public information about YOLO V3 is what made this possible at all, but now that we've done it on a simpler model, we have concluded that it would absolutely be possible on a more advanced and modern one like YOLO V11. After updating the model to a newer version, we will train the modified V11 model using the synthetically blurred dataset and train the unmodified model with the unblurred images to establish a control case. We will then be able to compare the distance detection abilities and success rate between the YOLO V3 and YOLO V11 models trained on unblurred data and the YOLO V3 and YOLO V11 models trained on synthetically blurred data that simulates the 45 degree offset camera. We hope that YOLO V11 can overcome the loss of confidence issues we experienced with the YOLO V3 blurred model.

An open ended problem we have come across during our research is how we can get an AI model to accurately detect distance from distorted images. In real world applications such as self-driving cars, LIDAR and stereoscopic camera systems are used for distance detection due to their high reliability. However, the major hurdle behind replacing these expensive technologies for a single camera is the possibility of the lens being disturbed by rain or other environmental factors which will lead to image distortions. If we can create an AI model that successfully detects distance even when given a distorted and blurred image, we could replace current applications of LIDAR and stereoscopic systems with a single camera. This open ended problem is what our project was intending to but unable to solve using YOLO V3. Through our experimentation and research, we ultimately learned that distance detection of blurry images is currently impossible with the current capabilities of YOLO V3.

References

[1] L. Xiaoming, Qin Tian, Chen Wanchun, and Y. Xingliang, "Real-time distance measurement using a modified camera," *2010 IEEE Sensors Applications Symposium (SAS)*, Feb. 2010, doi: https://doi.org/10.1109/sas.2010.5439423

[2] M. Vajgl, P. Hurtik, and T. Nejezchleba, "Dist-YOLO: Fast Object Detection with Distance Estimation," *Applied Sciences*, vol. 12, no. 3, p. 1354, Jan. 2022, doi: https://doi.org/10.3390/app12031354.

[3] "Marek Vajgl / Yolo With Distance · GitLab," *GitLab*, 2024. https://gitlab.com/EnginCZ/yolo-with-distance

[4] "The KITTI Vision Benchmark Suite," *www.cvlibs.net*. https://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d

Description of Individual Effort: The project proposal, presentation, and final report had contributions from every member of the group. The gradient blurring algorithm was written by Simmon and Elijah with final modifications and comments from Teague. The model was written by Teague and Simmon. The data collection portion of the project took more time than expected due to the amount of challenges we encountered. The model implementation also took a majority of the time but was expected to be a long process as we had to fine-tune the weights.