Autonomous Vehicles: Using Stereo Vision for Object Distance Ranging

Rowan Ho

Object Detection

To detect objects in the provided images, I used the provided YOLO[1] detection system.

Images in the dataset contain areas of high and low brightness, often within the same image (figure 1a). This can reduce YOLO's effectiveness. To deal with both areas, a tiled histogram equalization can be used. Specifically, I used OpenCV's implementation of Contrast Limited Adaptive Histogram Equalization(CLAHE).[2]

Figure 1b shows CLAHE applied on figure 1a using a tile size of 8x8 pixels, and a clip limit of 1.0, a parameter which moderates the introduction of noise. The localised contrast correction leads to a brightening effect for the dark part of the image, whilst the brightly lit part benefits from less washed out colours.

Figure 1a: Dataset image with both bright and dark areas

Figure 1b: Image with CLAHE contrast equalisation applied

Stereo Ranging

I explored both a sparse and dense stereo ranging approach to obtain estimated depth of detected objects.

Dense Stereo Ranging

Dense stereo ranging calculates a disparity map for all pixels, and a resulting depth map $Z = f\frac{B}{d}$. Noisy disparity can be reduced by using a weighted least squares filter (wls filter)[3]. The wls filter requires output from both a left matcher (figure 2b) which computes disparity from the left image to the right image, and a right matcher which does the inverse. The two matchers' output is input into the filter to produce a much smoother disparity, and therefore depth map (figure 2c). I also used contrast equalisation before the image was input into the disparity matchers, which improved output.

Figure 2a: Image to obtain depth map for

Figure 2b: Disparity map using disparity from single left matcher

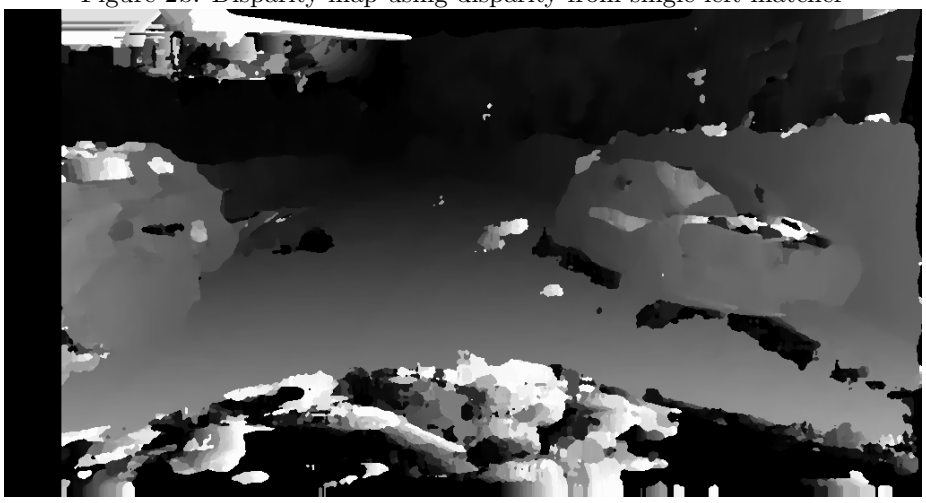
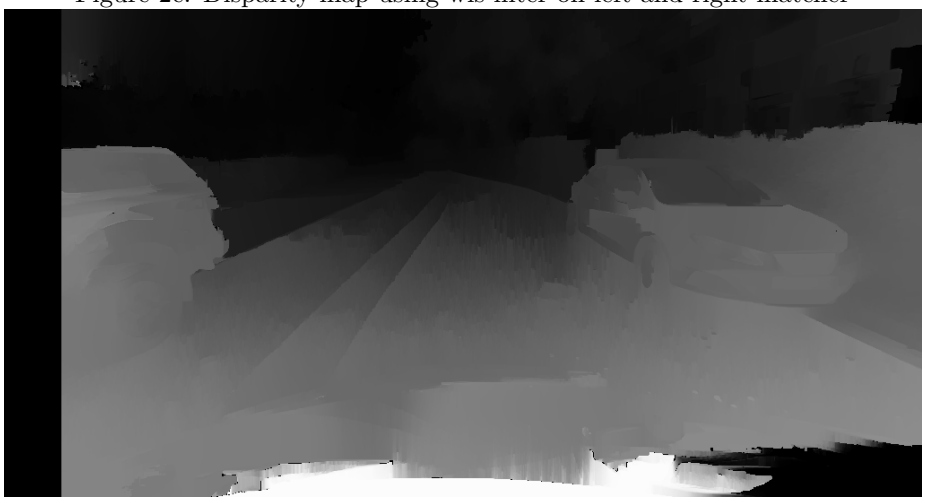


Figure 2c: Disparity map using wls filter on left and right matcher



The next step is to estimate depth for each object inside a bounding box. Bounding boxes incorporate both areas of the detected object and background areas around it. Figure 3a shows one such box, and 3b the histogram of the depth map inside the box. As the bounds include areas of background, a solution is to try a thresholding algorithm to find one lower cluster of depth values representing the foreground object and one cluster for higher values, which are likely to come from background. One such algorithm is Kmeans. Figure 4 shows the comparison of the mean, median and mode vs taking the mean of the lower kmeans cluster, for each box. The plot shows the relative distributions of these estimated depths for the whole dataset.

The mean tends to overestimate depth, but for further away objects, other methods have large spikes of frequency at certain values. This may be due to the depth map having less precision at larger depths, as disparity approaches 0. However, most detected objects in frames occured < 20m away, and evaluating close distances is important to an automated vehicle. Therefore, I chose the kmeans method over the others, as this tended to be the least variable/unreliable for these close objects, based on manuable observation.

Figure 3a: Bounding box drawn by YOLO

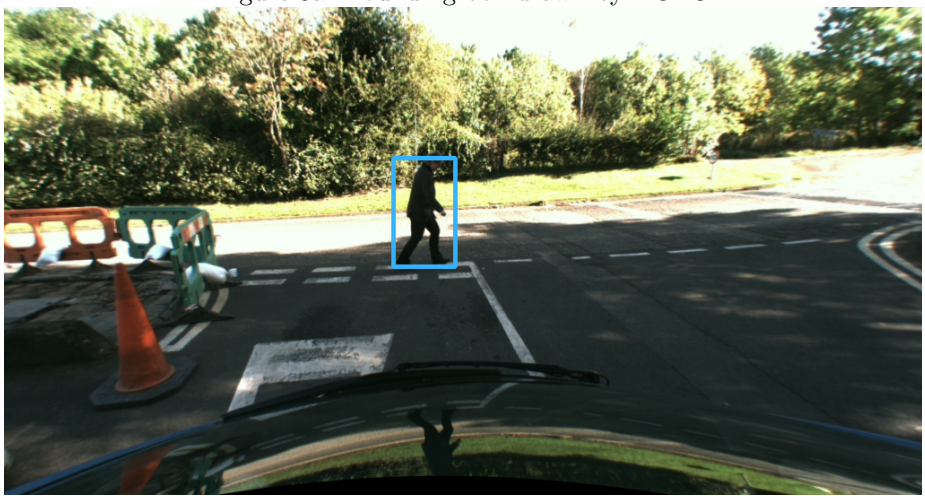


Figure 3b: Histogram plot of the box's depth map.

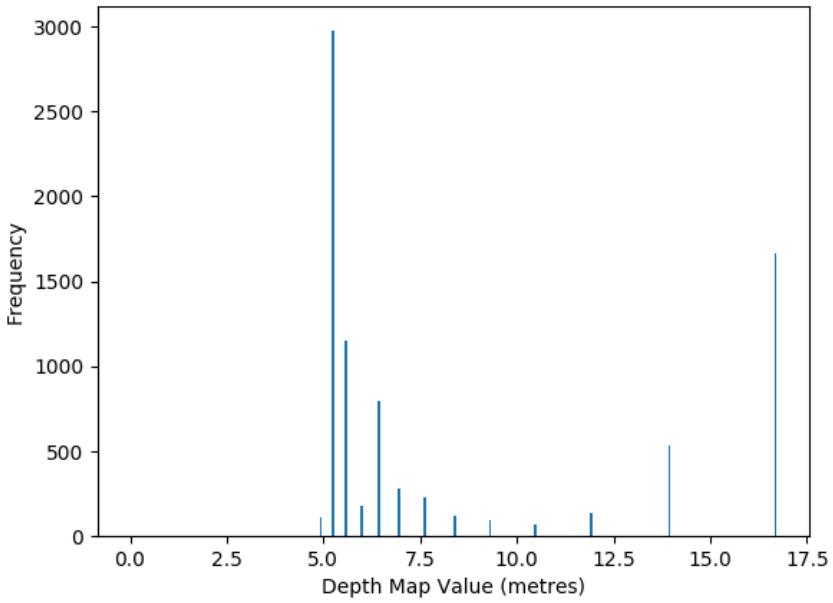
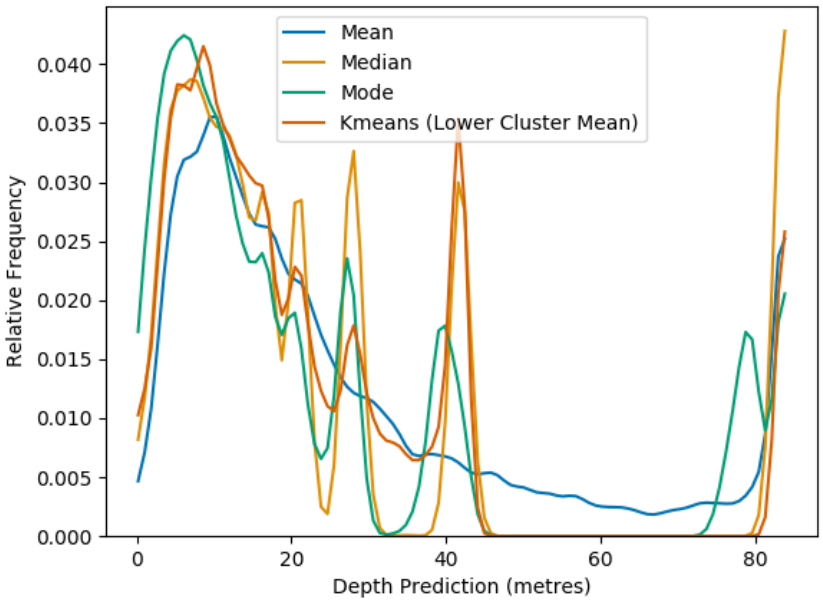


Figure 4: Distribution of Depth Estimation Methods



I also attempted using OpenCV's MOG2 background subtractor to remove areas of background from the disparity map. Figure 5b shows the resulting foreground mask applied to 5a. 5c shows the result with post processing on the mask. This processing consisted of identification of connected components (only keeping components over a threshold size), then a 'closing' operation, which applies dilation to close up small gaps in the foreground mask, followed by erosion.

However improved by post processing, the quality of the background subtraction was still poor. Due to the moving scene, the histories of each pixel are dramatically different as the car moves, which limits how well MOG2 can perform.

Figure 5a: Disparity map without foreground mask

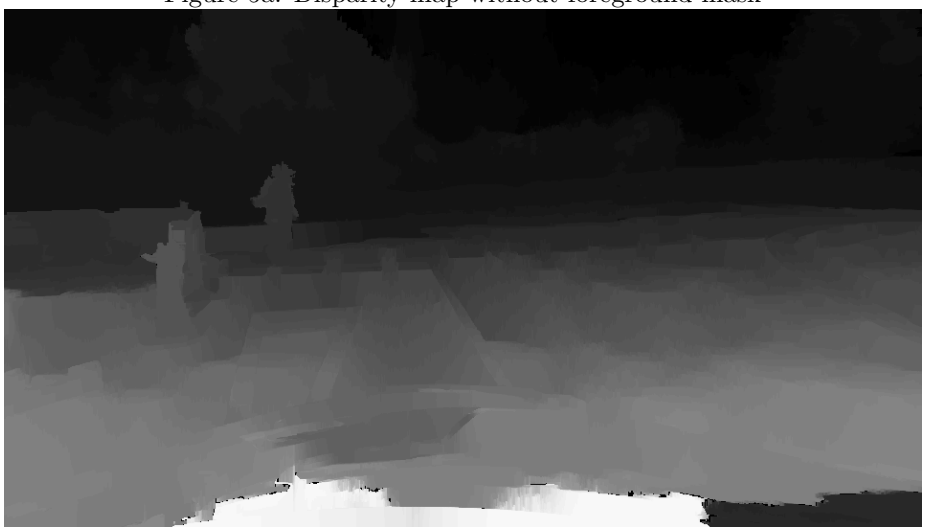


Figure 5b: Disparity map with foreground mask applied (no post processing)

Figure 5c: Disparity map with foreground mask applied (with post processing)

Sparse Stereo Ranging

For sparse stereo ranging, I used the ORB[4] feature detection algorithm to pick out feature points on both the left and right images. These can then be matched together (figure 6a).

Disparity between each matched feature point (figure 6b) can be used to obtain a corresponding depth point using $Z = f\frac{B}{d}$. Occasionally, the method detected few or no output results inside a bounding box. To reduce the likelihood of this, I increased the sensitivity of ORB's feature detection by tuning parameters.

Figure 7 shows the distribution of results comparing sparse and dense stereo disparity methods (using the clustering method for both). It is noticeable that dense stereo is more likely to estimate objects as being very far away (>80m). Sparse stereo is more resilient to this problem, maybe as using only feature points in the calculation means we are less likely to incorporate background areas. From manual observation, I thought the sparse implementation gave a more stable estimate of depth.

Figure 6a: Matching of ORB feature points

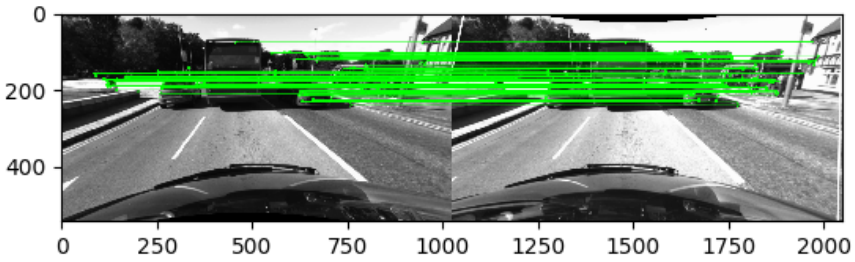


Figure 6b: Corresponding Sparse Disparity Map (dilated for visibility)

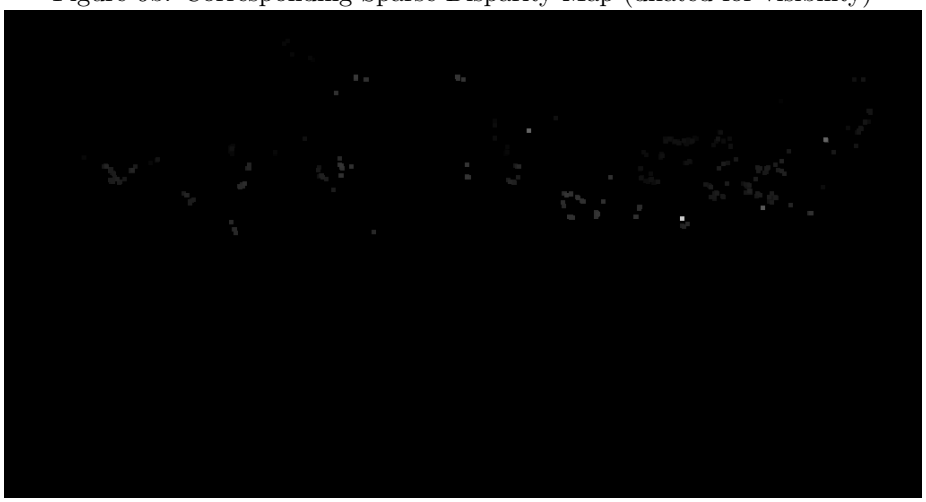
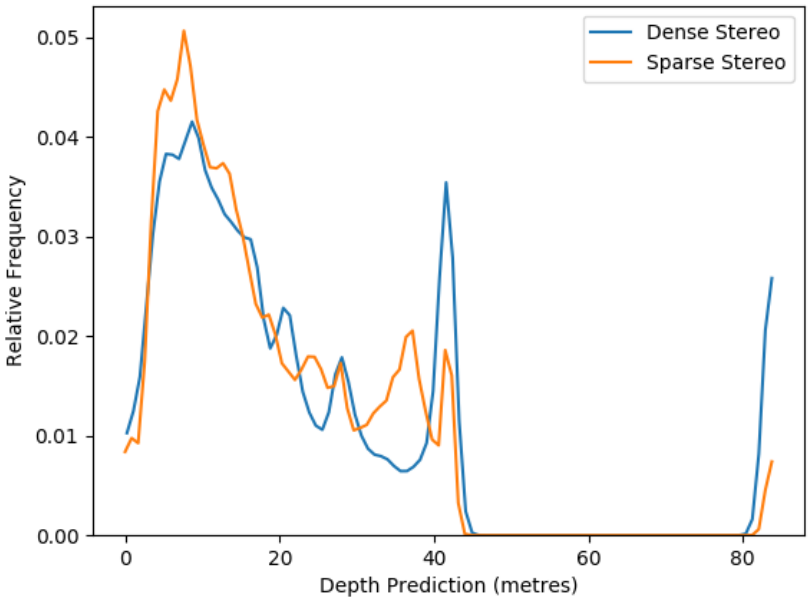


Figure 7: Dense vs Sparse Stereo Disparity Comparison



References

1. Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779-788.

2. Pizer, S.M., Amburn, E.P., Austin, J.D., Cromartie, R., Geselowitz, A., Greer, T., ter Haar Romeny, B., Zimmerman, J.B. and Zuiderveld, K., 1987. Adaptive histogram equalization and its variations. Computer vision, graphics, and image processing, 39(3), pp.355-368.

3. Min, D., Choi, S., Lu, J., Ham, B., Sohn, K. and Do, M.N., 2014. Fast global image smoothing based on weighted least squares. IEEE Transactions on Image Processing, 23(12), pp.5638-5653.

4. Rublee, E., Rabaud, V., Konolige, K. and Bradski, G.R., 2011, November. ORB: An efficient alternative to SIFT or SURF. In ICCV (Vol. 11, No. 1, p. 2).