Pixel Ratio: A Promising Approach for Accurate 3D Bounding Box Estimation of Occluded Vehicles

Paul Saunders
Department of Computing Science
University of Alberta
psaunder@ualberta.ca

Cameron Hildebrandt
Department of Computing Science
University of Alberta
childebr@ualberta.ca

Khalid Al-Mahrezi
Department of Computing Science
University of Alberta
almahrez@ualberta.ca

Curtis Kennedy
Department of Computing Science
University of Alberta
ckennedy@ualberta.ca

Domain Experts
Dr. Ehsan Hashemi, Dr. Arunava Banerjee, Marcelo Jafett Contreras Cabrera
ehashemi@ualberta.ca, arunava2@ualberta.ca, marceloj@ualberta.ca

Abstract

Accurately identifying 3D bounding boxes around occluded objects in images is a challenging task in computer vision. Most existing methods focus on building different model architectures, with few additional techniques used to improve the performance of these models outside train-time optimizations such as SGD and Adam. In this paper, we propose to incorporate a novel metric called the “Pixel Ratio” into the loss functions of existing models to improve the model’s performance in identifying 3D bounding boxes around cars in images without the use of depth data. Specifically, we aim to improve the model’s performance on heavily occluded vehicles. Our experimental results show that incorporating this metric into the model does not hurt its overall performance, and may even provide some improvements in cases where occlusion is present.

1 Introduction

Object detection and recognition in images has been an active research area in computer vision for many years [1]. One important task in this field is “3D Bounding Box Estimation”, in which the goal is to accurately identify the 3D bounding boxes around objects in images. An example of such a box can be seen in figure 1. Estimation of these 3D bounding boxes is quite useful in the areas of autonomous driving, robotics, and surveillance as they allow the agent to make informed decisions about where an object lies in 3D space. While significant progress has been made in this area [2], accurate 3D bounding box estimation *without depth data* for occluded objects remains a challenging task. Indeed, many of the top models in this field struggle to do well on the task, reaching around 15%-20% average precision [3]. Such methods of addressing this problem are primarily concerned with building different model architectures designed for this task [4] [5]. The loss functions for these models typically follow from the model architecture, with few additional techniques used outside train-time optimizations.

In this project, we propose to incorporate into existing models and loss functions a novel metric called the “Pixel Ratio”, with the intent to improve the performance of existing machine learning models in

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

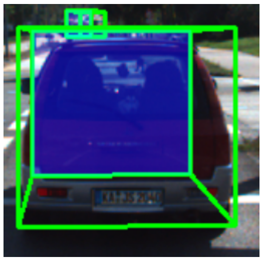


Figure 1: An example of a 3D bounding box for an SUV

identifying 3D bounding boxes around cars in images, without the use of depth data. The Pixel Ratio is defined as the ratio of the number of pixels that belong to a car to the number of pixels that belong to the car's surrounding 2D area. This metric has not been extensively studied in the context of 3D bounding box estimation for occluded objects, and we believe it may provide useful information to improve the performance of existing models.

Our main objective over the course of the project was to investigate the effectiveness of the Pixel Ratio metric in improving the performance of machine learning models designed to identify 3D bounding boxes around occluded cars in images. We hypothesized that including the Pixel Ratio metric in these models will improve their performance in estimating 3D bounding boxes for occluded vehicles. Our experimental results showed that incorporating this metric into the model does not hurt its overall performance, and may even provide some improvements in cases where occlusion is present.

2 Existing work

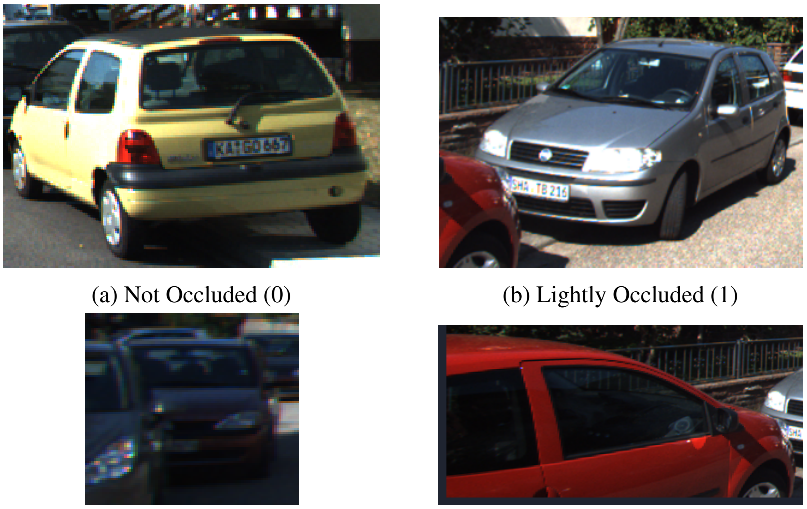
2.1 Dataset

The KITTI dataset [6] is a well-known standard dataset within the fields of object detection and bounding box estimation. Its collection of images for 3D bounding box estimation is several thousand images in size, all taken within the country of Germany. Images contain busy highways, urban streets, rural country, and many other scenarios. Each car in the dataset has an associated 2D bounding box, 3D bounding box, and occlusion value, among other values of which we did not make use. We should note that the occlusion value is an integer $\in \{0, 1, 2, 3\}$, which represents "not occluded", "slightly occluded", "heavily occluded", and "occlusion unknown", respectively. Examples of such occlusion values and corresponding cars can be found in Figure 2. For training, we made use of just over 20,000 of these cars, with the others held out for testing.

2.2 Background

3D object detection is a well studied problem, and remains at the forefront of autonomous vehicle research and development. The monocular (single-lens) case is attractive due to its low cost and ease of use, as opposed to an expensive LiDAR system. LiDAR is an optical sensing technology used to map points in the environment with their location in 3D space. The performance of monocular 3D object detection is slower with lower average precision than using LiDAR [3]. The increased challenge over 2D object detection is that there is no depth information in an RGB image. This poses the fundamental issue of determining 3D depth in a 2D image.

Subsequently, a solution methodology called 'pseudo-LiDAR' was developed, in which depth information is estimated using various methods that do not rely on LiDAR data. These methods focus on estimating the 3D point cloud, and then feed it as input to a proven state-of-the-art 3D detection method. Other non pseudo-LiDAR methods will implicitly predict depth as they attempt to directly predict 3D boxes. Across these two techniques lies a growing list of methods, including transfer learning and using geometric constraints. In addition, some methods utilize LiDAR data in training but not for inference. What follows is a discussion of some previously developed models in the field. We indicate the use of LiDAR in the heading titles below.



(a) Not Occluded (0)

(b) Lightly Occluded (1)

(c) Heavily Occluded (2)

(d) Unknown (image boundaries) (3)

Figure 2: Examples of the different occlusion labels in the KITTI dataset.

2.3 DD3D (requires LiDAR for training)

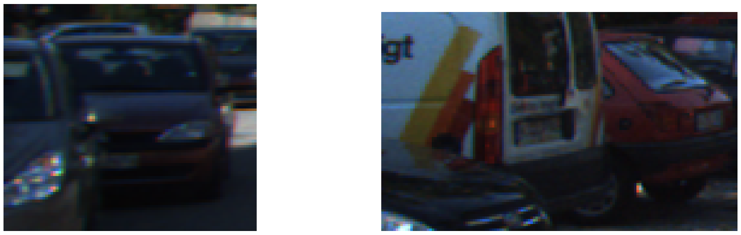
DD3D is an implementation of [7]. In order to bridge the gap between methods that require and do not require LiDAR data, the model is pre-trained using depth. For inference, just a single RGB image is needed. Park et al. describe this strategy as ‘the best of both worlds’, being an end-to-end, single stage model. However, the authors note that a limitation of this approach is that it suffers from limited generalizability. This means that the performance degrades when evaluated on data that is unlike what it was pre-trained on. The pre-training process also benefits from having many samples. [7] reported best performance when DD3D was pre-trained with 15 million images and accompanying LiDAR data. Our takeaways were that despite strong performance, DD3D required extensive pre-training on non-KITTI data, and did not employ any specific strategies to combat occlusion. Therefore, due to the pre-training required, we chose to not include the DD3D model in our experiments.

2.4 MonoRCNN++ (does not require LiDAR)

MonoRCNN++ [8] is a framework which uses multivariate probabilistic modelling to learn the joint probability distribution of the visual (in-picture) height with the physical (in real life) height. While existing models treat these two variables as independent, [8] demonstrates otherwise by learning the correlation between visual and physical height. To address the problem of occlusion, the authors use uncertainty modelling [9] for the physical size, yaw angle, and projected centre. Uncertainty-aware regression loss helps make the model focus more on less occluded samples when learning. This aims to prevent training samples with heavy occlusion from degrading the model’s predictions. Our takeaway was that we hope to expand on the idea of building an occlusion-aware loss function, but need a precise way to quantify occlusion compared to KITTI’s categorical labels.

2.5 YOLO3D (does not require LiDAR)

YOLO3D [10] is an implementation of [4]. It uses a YOLO (You Only Look Once) model to first identify objects in the image. The output of the YOLO model is a 2D bounding box, confidence score, and label for each detected object in the image. Having an accurate 2D bounding box is crucial for this framework to succeed. The YOLO3D framework then takes that output and, using a convolutional neural network, regresses the 3D object properties such as orientation and object dimensions. Finally, the 2D bounding box is used to apply geometric constraints on the object orientation and object dimensions to estimate the final 3D bounding box. We found the use of proven 2D methods to help predict the 3D bounding box appealing. Our takeaway was that YOLO3D would be a good baseline, with high potential for adding occlusion-awareness to either the 2D or 3D stages.



(a) Parallel-parked red car

(b) Red car behind van

Figure 3: Two cars labelled "heavily occluded" can have different true occlusion levels.

3 Methodology

3.1 Overview

Our methodology is focused around the addition of an occlusion-aware metric into the existing loss functions of several existing models. We then retrain the models with the new loss functions. Our aim is for such models to place more emphasis on reducing the loss on cars that are heavily occluded than on reducing the loss with no preference between cars (as would typically be the case). Such models will be placed after an existing model in a pipeline (we used a YOLO [11]model). The first model will identify labelled two-dimensional bounding boxes around cars which, along with the image, will be fed to the second model. The second model will then use the 2D bounding box-image pair to identify a 3D bounding box. Note that some images may have multiple bounding boxes, and in the second model, each bounding box and corresponding image is treated as a single data point. Furthermore, any transformations that occur to the image as a result of this bounding box are model specific. For instance, some models such as YOLO3D [10] crop the image to the 2D bounding box boundaries before feeding it through the 3D bounding box detector. Other models simply use the entire image and bounding box as input.

It's important to note that our use of the KITTI dataset was limited to the provided training images. Although the KITTI dataset does include images explicitly labelled for testing, the labels for this data are not public. This is to ensure that the models on KITTI's leaderboards are only trained on the intended training data. Therefore, we were forced to only use KITTI's training dataset for all facets of our experiments, as our results were not submitted to KITTI for evaluation.

3.2 Model architectures

Two model architectures were used in our research: an architecture designed explicitly for 3D bounding box detection called YOLO3D [10], and a more basic model architecture built off of ResNet-18 [12]. The purpose of using the ResNet-18 architecture is to establish a baseline, as it does not have any task-specific adaptations other than the format of the output. This is to ensure that any change in a model's performance from inclusion of the pixel ratio metric was due to the metric itself and not a quirk of the more task-specific model architectures.

3.3 An occlusion-aware metric

We define an "occlusion-aware metric" to be any metric whose value is strongly correlated with the occlusion of any particular object within an image. In our case, such objects were limited to those labelled as *cars* in the KITTI dataset [11]. As stated above, the default KITTI occlusion values are the integers $0, 1, 2, 3$. Such a metric is not precise enough to sufficiently describe all possible occlusion scenarios in which a car can exist (such as occlusion by other cars, objects, image boundaries, etc.). For instance, as shown in Figure 3, two cars labelled as "heavily occluded" might indeed have different levels of occlusion to a person. Therefore, to rectify this shortcoming in the KITTI dataset, we propose a new occlusion aware metric that we have dubbed *Pixel Ratio*.

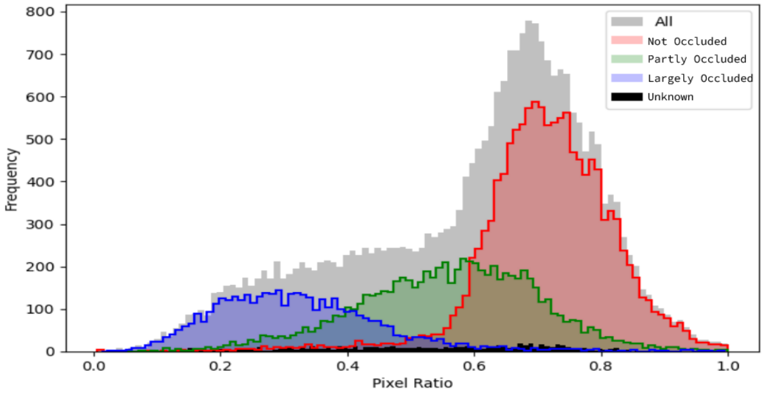


Figure 4: Histogram of pixel ratios, categorized by the KITTI occlusion label

	Depth	Pixel Ratio	Pixel Count
Depth	1	0.0052	−0.53
Pixel Ratio	0.0052	1	0.065
Pixel Count	−0.53	0.065	1

Table 1: Correlation Table between depth, pixel ratio, & pixel count

3.4 Pixel ratio

Any car in an image has a tight two-dimensional bounding box that surrounds it. Through the use of object segmentation models such as Detectron2 [13], we can discover all pixels that fall within both the two-dimensional bounding box and the car itself. Let P_c be the set of all pixels within the bounds of the car's outline, and let P_b be the set of all pixels within the car's 2D bounding box. Then,

$$\texttt{Pixel Ratio} = \frac{|P_c|}{|P_b|}$$

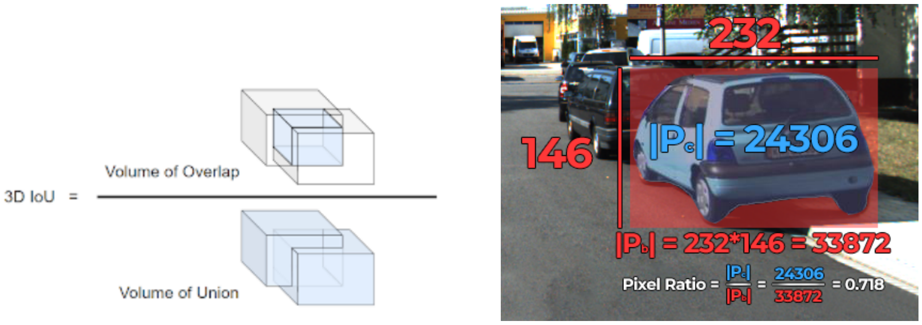
Since a higher Pixel Ratio corresponds to a less occluded car, we instead incorporate $1-\texttt{Pixel Ratio}$ into the models to emphasize occluded values. Observe that in principle, $P_c < P_b$. In our case, we use the two-dimensional box from KITTI since that's what each model uses as the ground truth, yet we use the pixel labels from the segmentation model to get the pixel counts. Since there can be disagreement between KITTI's and Detectron2's bounding boxes, there are several instances in which $P_c \geq P_b$. We chose to omit these instances rather than trim them to 1, as it left us with numerous samples in which we knew the value of the Pixel Ratio exactly. It also reduced the time in which the models could train, as they had fewer training examples to process.

3.5 Depth-independence of pixel ratio

A glaring issue with using pixel count as a measure of occlusion is the connection of pixel count to depth. An unoccluded car located far away from the camera will inherently have a lower pixel count than an unoccluded car closer to the camera. Therefore, as we only want to bias the model towards improving on more occluded cars (and not those farther from the camera), we needed to be sure that the pixel ratio metric was not influenced by depth. Indeed, as is visible in table 1, the pixel ratio metric has virtually zero correlation with depth. Yet, we can see quite a strong correlation between the pixel *count* and the depth. This implies that the normalization present in the pixel ratio metric through dividing by the 2D bbox's pixel count is useful in reducing correlation. Furthermore, this normalization still results in a metric that is strongly correlated with occlusion, which is our ultimate goal with the pixel ratio metric. This can be seen in Figure 4.

3.6 Training

The method in which each model was trained and the specific way in which the pixel ratio was integrated into the loss functions varied between the training scripts of different models, though it



(a) 3D Intersection over Union (IoU)

(b) Pixel counts for a car & bounding box

Figure 5

was consistent between the baseline and ratio versions of a given model. For the ResNet-18 models, the loss function comprised a linear combination of orientation, dimension, and centre losses between the predicted & ground truth 3D bounding box. Therefore, the final loss could simply be multiplied by each data point's pixel ratio value, before being averaged to produce a final loss value for the mini-batch. For YOLO3D, the loss function is a little more complex, as it involves estimating offsets from default orientations and dimensions, then taking the argmax of different confidence values of each offset [4]. In other words, this loss is more heavily related to predicted angle and the ground truth angle than with the ResNet-18 model. Therefore, for YOLO3D, the pixel ratio was only combined with the sub-loss for the orientation of the bounding box. The other sub-loss functions in YOLO3D did not get weighted by the pixel ratio.

As this project required the training of several model architectures, we wanted to be sure that the training procedure was as applicable as possible to each model. Perhaps one model could be trained with cross validation, while another lends itself to a train / validation split. Therefore, to standardize testing, we divided the training dataset of KITTI into a train & test set. 80% of data became training data, while the other 20% became testing data. It was important that we executed this as a stratified random split to ensure that the distribution of each occlusion class stayed approximately the same in both halves of the split.

To create a meaningful measure of the impact our pixel ratio had on the quality of predictions, we trained and evaluated two versions of each model. We built one version of the model that was trained on our training samples which did not use the pixel ratio metric, this was the "baseline" model. The second version of the model utilized our pixel ratio metric in the loss function for training, this became the "ratio" model.

3.7 Evaluation

The primary method used to evaluate a single predicted 3D bounding box is 3D Intersection over Union (IoU). For a ground truth bounding box and predicted bounding box, we divide the volume of the boxes' intersection by the volume of the boxes' union. This value will always be in $[0, 1]$, with higher scores indicating better performance and identical boxes receiving a value of 1. The KITTI dataset [6] includes bounding boxes that are oriented arbitrarily about the vertical axis, which necessitates the use of a non-trivial computation of 3D IoU to account for the misalignment of orientation of the bounding boxes. Following the concepts of the algorithm in [14], we developed our own 3D rotated IoU calculation for evaluations in this paper. An example of (axis-aligned) 3D IoU can be seen in figure 5a

Models were evaluated using the standard method of average precision (AP). Several IoU thresholds are set in the range of $[0, 1]$. For each of these thresholds, we get the fraction of predicted 3D bounding boxes which have an IoU less than that threshold. This fraction forms the precision at that IoU threshold. We then take the average of all such precisions to get the final AP score. Note that AP also falls within the range $[0, 1]$, with higher scores indicating better performance.

Metric	3D Intersection over Union				Average Precision			
Occlusion Level	0	1	2	3	0	1	2	3
YOLO 3D Baseline	0.0899	0.0784	0.085	0.0927	9.51	8.37	9.01	9.75
YOLO 3D Ratio	0.0936	0.0861	0.0908	0.0968	9.86	9.12	9.58	10.2
ResNet-18 Baseline	0.108	0.103	0.109	0.124	11.3	10.8	11.4	12.9
ResNet-18 Ratio	0.107	0.108	0.115	0.12	11.2	11.3	12	12.5

Table 2: 3D IoU and Average Precision scores for tested models (larger is better)

Metric	OE			
Occlusion Level	0	1	2	3
YOLO 3D Baseline	0.113	0.211	**0.565**	0.636
YOLO 3D Ratio	0.085	0.165	**0.369**	0.522
ResNet-18 Baseline	0.252	0.271	0.3	0.59
ResNet-18 Ratio	0.236	0.268	0.278	0.555

Table 3: Orientation Error for tested models (smaller is better)

To gain more insight into how the integration of our pixel ratio metric affected the performance of a model, and to better understand the results on AP, we leveraged several post-training evaluation metrics similar to those used in other papers. [4] These metrics are:

1. Coordinate Error (CoE) – the Euclidean distance between the 3D centres of the objects. Values are always $\in [0, \infty)$, with smaller indicating better.
2. Camera Error (CaE) – the error in Euclidean distance from the camera to the centre of the object, useful for driving scenarios where the ability to avoid collisions is paramount. Values are always $\in [0, \infty)$, with smaller indicating better.
3. Orientation Error (OE) – the error in our estimation of the orientation of the object. Values are always $\in [0, \pi)$, with smaller indicating better.

CoE, CaE, and OE are intended to quantify particular facets of the predictions and help us understand a model's strengths and weaknesses, while 3D IoU is particularly useful in quantifying the overall quality of predictions, as the score is penalized by all inaccuracies.

4 Results

4.1 Discussion

As is visible in table 2, including the pixel ratio metric in the loss function of a model is not an easy way to drastically improve the performance of the model. The change in performance between the baseline model and ratio model is quite small for both YOLO3D and ResNet-18, and certainly not statistically significant. A researcher looking for a one-size-fits-all solution for improving their model should likely look elsewhere. However, despite the pixel ratio metric not being a simple solution for improving a model's occlusion performance, it is promising to see that it does not hinder the performance of the underlying model. Even when including the pixel ratio metric in the loss function, the model is able to learn the training data just as well as if the pixel ratio was absent. Consequently, due to the simple manner in which the pixel ratio can be included in a loss function, there is little reason why an individual developing such a model would knowingly omit such a metric.

Moreover, there are some cases where the improvement yielded from the pixel ratio is statistically significant ($\alpha = 0.05$), as can be seen in table 3. In the bolded instance, we see that there exists a statistically significant drop in orientation error for the YOLO3D ratio model at high occlusion levels, as compared to the baseline. This is promising, as finding the correct orientation of a bounding box plays a significant role in reducing the IoU between the predicted and ground truth boxes.

4.2 What we learned

Despite pixel ratio being a promising metric given figure 4 and the lack of correlation between depth and pixel count in table 1, we learned that in its current state, naively injecting pixel count into the loss function as a scalar multiplier is not by itself enough to greatly improve a model's performance in detecting 3D bounding boxes for occluded objects. However, this metric shows some promise and may yield more interesting results with further experimentation. Overall, we would have loved to push the metric further and look at whether it could improve performance better on some of the other models we studied.

4.3 Why are these results important?

This is a starting point for further research. As we've shown, integrating this metric does not seem to hinder the performance of a model. In some cases, the pixel ratio metric can slightly improve performance. Therefore, we do feel that this work has the potential to be a building block that helps enable a larger improvement in occlusion-focused models.

In summary, although the pixel ratio metric is not as effective at improving AP as we had initially hoped, its ability to improve orientation error in some models does show that its effect can be felt within the training of the model. This fact offers promising potential for future research into similar metrics.

5 Further work

5.1 Improved Pixel Ratio

Despite its positive attributes, the Pixel Ratio metric is an *approximation* of a car's level of occlusion. The bounding box itself is not part of the car, and indeed is only a stand-in for the car's true pixel count were it to not be occluded. An improved version of this metric would see the denominator not be the pixel count of the bounding box, but the pixel count of the same car with the occluding object(s) removed. This would provide a better reflection of the car's true occlusion amount, as humans don't take use bounding boxes when recognizing occlusion; they use the fraction of the object itself that is blocked.

5.2 Analysis of more models

As discussed above, there has been plenty of work surrounding 3D bounding box detection of moving vehicles in recent years. Consequently, plenty of unique and interesting models have been developed to attempt to solve this task. Unfortunately, due to time constraints, we could only make use of a few of those models in our analysis. One could easily see how a more in-depth attempt to understand the properties of the pixel ratio metric could influence the performance of the loss function, through adding it to several more models and looking at whether any of the model's characteristics influence the performance of the new loss function.

6 Conclusion

Over the course of this project, we investigated the effectiveness of incorporating into existing machine learning models a novel metric dubbed "Pixel Ratio". Our hope is that such a metric would improve the performance of these models in identifying 3D bounding boxes around occluded cars in images. The experimental results showed that including the Pixel Ratio metric in the model does not hinder its overall performance. In some cases, it may even improve a model's results on objects where occlusion is present. Our hope is that this work provides a potential path for further research in the field of occlusion-focused 3D bounding box estimation.

References

[1] Stamatia Dasiopoulou, Vasileios Mezaris, Ioannis Kompatsiaris, V-K Papastathis, and Michael G Strintzis. Knowledge-assisted semantic video object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(10):1210–1224, 2005.

[2] Hai Wu, Chenglu Wen, Shaoshuai Shi, Xin Li, and Cheng Wang. Virtual sparse convolution for multimodal 3d object detection. *arXiv preprint arXiv:2303.02314*, 2023.

[3] Yu Hong, Hang Dai, and Yong Ding. Cross-modality knowledge distillation network for monocular 3d object detection. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part X*, pages 87–104. Springer, 2022.

[4] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7074–7082, 2017.

[5] He Liu, Huaping Liu, Yikai Wang, Fuchun Sun, and Wenbing Huang. Fine-grained multi-level fusion for anti-occlusion monocular 3d object detection. *IEEE Transactions on Image Processing*, 31:4050–4061, 2022.

[6] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[7] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. Is pseudo-lidar needed for monocular 3d object detection? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3142–3152, 2021.

[8] Xuepeng Shi, Zhixiang Chen, and Tae-Kyun Kim. Multivariate probabilistic monocular 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4281–4290, 2023.

[9] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[10] ruhyadi. Yolo3d. `https://github.com/ruhyadi/YOLO3D`, 2022.

[11] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[14] Lanxiao Li. rotated_iou: Differentiable iou of rotated bounding boxes using pytorch. `https://github.com/lilanxiao/Rotated_IoU`, 2021.

Acknowledgments and Disclosure of Funding

We would like to acknowledge the incredible help that we have received from the CMPUT 469 domain experts. Dr. Ehsan Hashemi, Dr. Arunava Banerjee, and Marcelo Jafett Contreras Cabrera were invaluable throughout this semester, and we could not have achieved our results without their advice.