Multivariate Probabilistic Monocular 3D Object Detection

Xuepeng Shi[1] Zhixiang Chen[1] Tae-Kyun Kim[1,2]
[1]Imperial College London [2]KAIST

Abstract

In autonomous driving, monocular 3D object detection is an important but challenging task. Towards accurate monocular 3D object detection, some recent methods recover the distance of objects from the physical height and visual height of objects. Such decomposition framework can introduce explicit constraints on the distance prediction, thus improving its accuracy and robustness. However, the inaccurate physical height and visual height prediction still may exacerbate the inaccuracy of the distance prediction. In this paper, we improve the framework by multivariate probabilistic modeling. We explicitly model the joint probability distribution of the physical height and visual height. This is achieved by learning a full covariance matrix of the physical height and visual height during training, with the guide of a multivariate likelihood. Such explicit joint probability distribution modeling not only leads to robust distance prediction when both the predicted physical height and visual height are inaccurate, but also brings learned covariance matrices with expected behaviors. The experimental results on the challenging Waymo Open and KITTI datasets show the effectiveness of our framework [1].

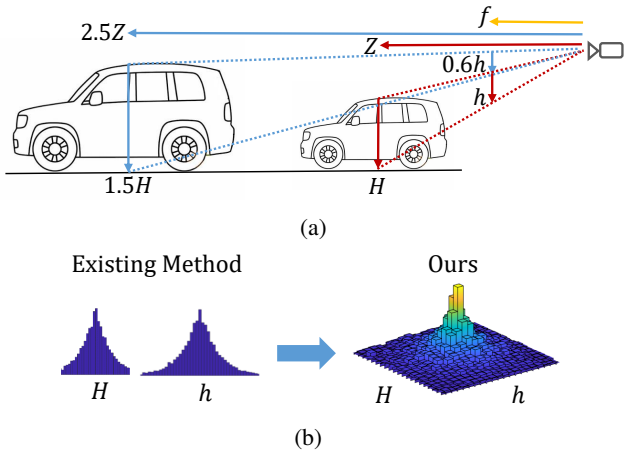


Figure 1: a) The inaccurate physical height and visual height prediction may exacerbate the inaccuracy of the distance prediction, if two predicted errors cannot be canceled by each other. For example, if the predicted physical height increase to 1.5 times and the predicted visual height decrease to $\frac{3}{5}$ times, the predicted distance will increase to 2.5 times. b) Existing method [43] models the physical height and visual height as two independent variables. In contrast, our method models the joint probability distribution of these two variables to explicitly learn the correlation.

1. Introduction

3D object detection aims to locate objects with 3D bounding boxes. It is widely used in and important to autonomous driving. LiDAR and RGB image sensors are commonly used for this task. Compared to LiDAR-based 3D object detection [54, 41, 56, 23], image-based monocular 3D object detection [46, 5, 40] is with low computation and energy cost as the 3D spatial locations of objects are inferred from monocular images. Despite the advantage of computation cost, monocular 3D object detection is challenging because it is essentially an ill-posed problem to infer the distance of objects from 2D images. To infer the spatial location of an object, the object visual appearance can be exploited when considering the inverse process of the imaging geometry [14]. Such that, the factors in this process, including the prior of object physical size, scene layout, and the camera's imaging process are of great importance to monocular 3D object detection, especially the distance prediction.

In monocular 3D object detection, different geometric priors have been utilized to recover the distance indirectly. Deep3Dbox [33] recovers the distance by the physical size and the 2D bounding boxes. Keypoint-based methods [18, 24] recover the distance by the physical size and the predicted eight projected corners. Shape-based methods [4, 28] recover the distance by the physical size and the predicted shape of objects. MonoRCNN [43], GUPNet [29], and DEVIANT [21] recover the distance by the physical height and the projected visual height, which improves the distance prediction.

[1]https://github.com/Rock-100/MonoDet

Although such decomposition framework [43, 29, 21] can introduce explicit constraints on the distance prediction, they suffer from inaccurate physical height and visual height prediction. As shown in Fig. 1a, the inaccurate physical height and visual height prediction may exacerbate the inaccuracy of the distance prediction. To improve the accuracy of the physical height and visual height prediction, uncertainty modeling [19] is used to predict the heights in [43, 29, 21]. Uncertainty modeling can make the physical height and visual height prediction more accurate, as it can relieve the negative effect of noisy training samples. GUPNet [29] further introduces a learnable depth bias to correct the distance prediction error. However, the existing works do not explicitly model the joint probability distribution of the physical height and visual height, which may hinder the model to capture the correlation between the two heights.

To resolve the above gap, we propose a multivariate probabilistic framework. As shown in Fig. 1b, we explicitly model the joint probability distribution of the physical height and visual height, instead of modeling these two variables independently as in [43]. This is achieved by learning a full covariance matrix of the physical height and visual height during training, with the guide of a multivariate likelihood. Such explicit modeling not only leads to accurate physical height and visual height prediction, but also makes the model explicitly learn the correlation between the two heights. Capturing the correlation can help the model achieve robust distance prediction when both the predicted physical height and visual height are inaccurate, as the predicted errors of the two heights can be canceled by each other. Besides, we model the uncertainties of the physical size, yaw angle, and projected center. This leads to better physical size, yaw angle, and projected center prediction and also improves the 3D object detection accuracy.

To better evaluate our method, we conduct experiments on both the widely used KITTI dataset [13] and the more recent Waymo Open dataset [47]. The Waymo Open dataset [47] is much more diverse and challenging than the KITTI dataset [13]. The experimental results show our method can predict covariances as expected effectively and support the superiority of our method.

The contribution of this paper is three-fold:

1. Originally explicitly modeling the joint probability distribution of the physical height and visual height to improve the 3D object detection accuracy, with the guide of a multivariate likelihood during training.

2. An accurate and robust monocular 3D object detection framework with probabilistic outputs for all 3D variables.

3. Achieving the state-of-the-art (SOTA) accuracy on the monocular 3D object detection task of the challenging Waymo Open dataset [47].

2. Related Work

2.1. Monocular 3D Object Detection

Monocular 3D object detection has drawn much attention. Learning-based methods [55, 7, 44, 32] directly regress the distance of objects by adding distance branches to 2D object detectors. These methods are simple and efficient but there is no explicit constraint in the distance prediction. Pseudo-LiDAR-based methods [52, 31, 49, 48, 58] first predict the depth map of an input image using an external monocular depth estimator, then predict the distance of objects with the aid of the estimated depth map. The accuracy of monocular 3D object detection is bounded by the accuracy of monocular depth estimation. 3D-anchor-based methods [2, 3, 22] predict the transformations from the 3D anchor boxes to the ground-truth 3D bounding boxes, which can ease the challenging distance learning. BEV-based methods [40, 38] first transform the feature maps from perspective view to orthographic view, then directly conduct 3D object detection in the 3D space. Equivariance-based method [21] designs depth equivariant backbones for monocular 3D object detection, which improves the generalization ability. Ensemble-based method [25] ensembles multiple distance predictions from different cues, which can improve the distance prediction accuracy. Video-based methods [3, 50] exploit the temporal information to improve the 3D object detection accuracy.

Many recent works in monocular 3D object detection decompose the distance of objects and recover it indirectly. These methods can improve the distance prediction accuracy as explicit constraints are introduced. Deep3Dbox [33] recovers the distance by minimizing the re-projection error between the four boundaries of projected 3D bounding boxes and 2D bounding boxes. Keypoint-based methods [18, 24] recover the distance by minimizing the re-projection error between the eight projected corners of 3D bounding boxes and the predicted eight projected corners. Shape-based methods [57, 35, 34, 4, 28] recover the distance by minimizing the re-projection error between the dense shape of objects and the predicted projected keypoints. MonoJSG [26] proposes the semantic and geometric cost volume to better recover the distance of objects. DID-M3D [37] decomposes the instance depth of objects into visual depth and attribute depth. MonoRCNN [43] and GUPNet [29] recover the distance by the physical height and the projected visual height. However, these existing methods do not explicitly model the joint probability distribution of multiple decomposed variables. In contrast, our method explicitly models the joint probability distribution of the physical height and visual height, which leads to accurate and interpretable distance prediction. We use MonoRCNN [43] as the baseline to illustrate the effectiveness of modeling the joint probability distribution.

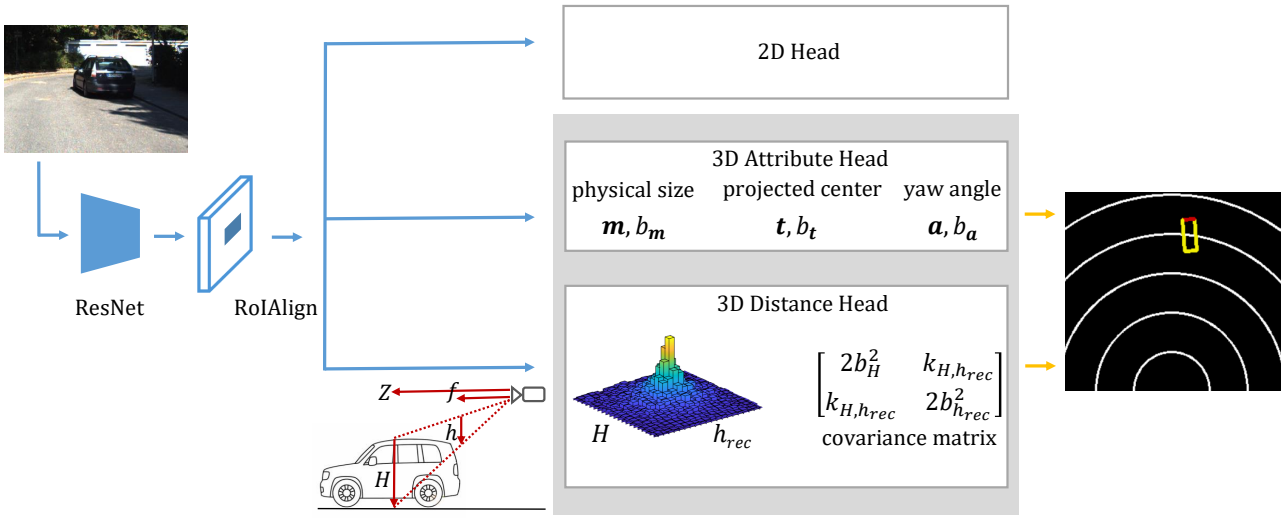


Figure 2: **Main architecture of MonoRCNN++**. Our MonoRCNN++ explicitly models the joint probability distribution of the physical height and visual height. Such explicit modeling not only leads to accurate physical height and visual height prediction, but also makes the model explicitly learn the correlation between the two heights.

2.2. Uncertainty and Covariance Estimation

The uncertainty-aware regression loss [19] has been utilized in many computer vision tasks. In 2D object detection, [17, 8] use the loss for bounding box regression. In 3D pedestrian localization, MonoLoco [1] uses the loss for 3D location regression. In LiDAR 3D object detection, [12, 11] introduce the loss to model the uncertainties of 3D variables. In monocular 3D object detection, [42, 32, 43, 29] use the loss for the distance-related variables to improve the accuracy of the distance prediction. However, when introducing the loss to multiple variables, these existing works simply apply the loss to these variables independently. In contrast, our method explicitly models the joint probability distribution and the covariance of different variables during training.

SUPN [10] is a seminal work studying the covariance estimation in computer vision. It extends a Variational Auto Encoder (VAE) [20] using a likelihood model with a full covariance matrix. By encoding a full covariance matrix, the samples obtained from such a model capture pixel-level correlations in the image domain and are free from salt-and-pepper (independent) noise. SUPN [10] is further adopted in [45] for monocular depth estimation to capture the pixel-level covariance. In contrast, our method focuses on monocular 3D object detection and considers the covariance in predicting the distance of objects.

3. Proposed MonoRCNN++

We first present the basic framework. Then we detail the probabilistic modeling in 3D detection heads. Finally, we show how learned covariances and uncertainties behave. We term our method MonoRCNN++ and the main architecture is illustrated in Fig. 2.

3.1. Basic Framework

Monocular 3D object detection aims to predict the 3D bounding boxes of objects from monocular images. Following MonoRCNN [43], MonoRCNN++ directly predicts the 3D bounding boxes of objects from RGB images based on the imaging geometry [14]. We build the basic framework upon Faster R-CNN [39], use a ResNet [16] with FPN [27] as the backbone, and use RoIAlign [15] to extract the crops of object features. We introduce two 3D detection heads, i.e., the 3D distance head and 3D attribute head, to adapt to monocular 3D object detection.

3D distance head recovers the distance of objects and is based on the geometry-based distance decomposition [43]. Specifically, the distance of an object Z is decomposed into the physical height H, and the reciprocal of the projected visual height $h_{rec} = \frac{1}{h}$, which is formulated as

$$Z = \frac{fH}{h} = fHh_{rec}, \tag{1}$$

where f denotes the focal length of the camera. 3D distance head regresses $\mathbf{d} = [H, h_{rec}]^{\mathrm{T}}$ and recovers Z by Eq. (1).

3D attribute head predicts the physical size, yaw angle, and projected center of objects. The physical size is denoted as $\mathbf{m} = [W, H, L]^{\mathrm{T}}$. The yaw angle is denoted as $\mathbf{a} = [\sin(\theta), \cos(\theta)]^{\mathrm{T}}$, where θ is the allocentric pose of 3D bounding boxes. Following [13, 47], only the yaw angle of the 3D bounding boxes is considered, and the roll and pitch

angles are assumed to be zero. The 2D projected center of a 3D bounding box is denoted as $\mathbf{p} = [p_x, p_y]^{\mathrm{T}}$.

MonoRCNN++ predicts the 3D center $[p_x, p_y, Z]^{\mathrm{T}}$ in pixel coordinates, and converts it to camera coordinates using a projection matrix $\mathbf{P}$ during inference, formulated as

$$\begin{bmatrix} p_x \cdot Z \\ p_y \cdot Z \\ Z \end{bmatrix}_{\mathrm{P}} = \mathbf{P} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}_{\mathrm{C}}. \tag{2}$$

Following [13, 47], per-image projection matrices are assumed to be available during both training and inference.

3.2. 3D Distance Head

To improve the prediction accuracy of the physical height and visual height, our MonoRCNN++ models $\mathbf{d} = [H, h_{rec}]^{\mathrm{T}}$ using a multivariate distribution with a full covariance matrix. Differently, MonoRCNN [43] simply applies the uncertainty-aware regression loss [19] to H and h_{rec} independently.

Let $\mathbf{d}$ be the prediction, $\hat{\mathbf{d}}$ be the groundtruth, and $\mathbf{\Sigma}$ be the predicted covariance matrix. Let $\mathbf{E}$ denote $(\mathbf{d} - \hat{\mathbf{d}})^{\mathrm{T}}\mathbf{\Sigma}^{-1}(\mathbf{d} - \hat{\mathbf{d}})$. For the regression of $\mathbf{d}$, the likelihood with a multivariate Laplace distribution is

$$p(\hat{\mathbf{d}} \,|\, \mathbf{d}, \mathbf{\Sigma}) = \frac{2}{(2\pi)^{\frac{N}{2}} |\mathbf{\Sigma}|^{\frac{1}{2}}} \frac{(\frac{\pi}{2\sqrt{2\mathbf{E}}})^{\frac{1}{2}} e^{-\sqrt{2\mathbf{E}}}}{\sqrt{\frac{\mathbf{E}}{2}}^{\frac{N}{2}-1}}, \tag{3}$$

where N is the length of $\mathbf{d}$. In our case, $N = 2$. The loss function of 3D distance head can then be formulated as

$$L_{dis} = -\log(p(\hat{\mathbf{d}} \,|\, \mathbf{d}, \mathbf{\Sigma})). \tag{4}$$

The covariance matrix $\mathbf{\Sigma}$ contains variances and covariances. $b_H = \sqrt{\frac{\mathbf{\Sigma}_{0,0}}{2}}$ and $b_{h_{rec}} = \sqrt{\frac{\mathbf{\Sigma}_{1,1}}{2}}$ are the scale parameters of the multivariate Laplace distribution, which can be interpreted as the predicted uncertainties of H and h_{rec}, respectively. $k_{H,h_{rec}} = \mathbf{\Sigma}_{0,1}$ is the predicted covariance of H and h_{rec}.

Covariance matrices are positive definite, thus it is difficult to directly predict $\mathbf{\Sigma}$ or $\mathbf{\Sigma}^{-1}$. Following [10], we represent the precision matrix $\mathbf{\Sigma}^{-1}$ via its Cholesky decomposition

$$\mathbf{\Sigma}^{-1} = \mathbf{L}\mathbf{L}^{\mathrm{T}}, \tag{5}$$

where $\mathbf{L}$ is a lower triangular matrix with positive diagonal elements. $\mathbf{L}$ can be formulated as

$$\mathbf{L} = \begin{bmatrix} e^{l_{0,0}} & 0 \\ l_{1,0} & e^{l_{1,1}} \end{bmatrix}. \tag{6}$$

Our model explicitly predicts $l_{0,0}, l_{1,0}, l_{1,1}$ to form $\mathbf{L}$, and then we can obtain $\mathbf{\Sigma}^{-1}$ by Eq. (5). We can further obtain the determinant of the covariance matrix $\mathbf{\Sigma}$ in Eq. (3) by

$$|\mathbf{\Sigma}| = \frac{1}{|\mathbf{\Sigma}^{-1}|} = \frac{1}{|\mathbf{L}\mathbf{L}^{\mathrm{T}}|} = e^{-2(l_{0,0}+l_{1,1})}. \tag{7}$$

Eq. (3) can then be computed using Eq. (5) and Eq. (7).

3.3. 3D Attribute Head

To improve the prediction accuracy of the physical size, yaw angle, and projected center, our MonoRCNN++ uses the uncertainty-aware regression loss [19] with the Laplace assumption. Differently, MonoRCNN [43] uses the L_1 regression loss for those variables.

The loss functions for the physical size $\mathbf{m}$ and yaw angle $\mathbf{a}$ can be formulated as

$$L_{size} = \frac{L_1(\hat{\mathbf{m}}, \mathbf{m})}{b_{\mathbf{m}}} + \log(b_{\mathbf{m}}), \tag{8}$$

$$L_{yaw} = \frac{L_1(\hat{\mathbf{a}}, \mathbf{a})}{b_{\mathbf{a}}} + \log(b_{\mathbf{a}}), \tag{9}$$

where $\hat{\mathbf{m}}$ and $\hat{\mathbf{a}}$ are the groundtruths, $\mathbf{m}$ and $\mathbf{a}$ are the predictions, and $b_{\mathbf{m}}$ and $b_{\mathbf{a}}$ are the learnable variables of uncertainties (the scale parameters of the Laplace distribution).

For the projected center prediction, the training target of a center is normalized by its proposal size. Let (x_1, y_1, x_2, y_2) denote the top-left and bottom-right corners of the proposal, and $\hat{\mathbf{p}} = [\hat{p}_x, \hat{p}_y]^{\mathrm{T}}$ and $\mathbf{p} = [p_x, p_y]^{\mathrm{T}}$ denote the groundtruth center and the predicted center, respectively. Let $\hat{\mathbf{t}}$ and $\mathbf{t}$ denote the normalized groundtruth center and the normalized predicted center, respectively, where $\hat{\mathbf{t}}$ is defined as

$$\hat{\mathbf{t}} = (\frac{\hat{p}_x - x_1}{x_2 - x_1}, \frac{\hat{p}_y - y_1}{y_2 - y_1}). \tag{10}$$

The projected center loss function can be formulated as

$$L_{kpt} = \frac{L_1(\hat{\mathbf{t}}, \mathbf{t})}{b_{\mathbf{t}}} + \log(b_{\mathbf{t}}), \tag{11}$$

where $b_{\mathbf{t}}$ is the learnable variable of uncertainties (the scale parameters of the Laplace distribution). During inference, the normalized predicted center $\mathbf{t}$ is transformed to the predicted center $\mathbf{p}$.

The overall training loss function for two 3D detection heads is

$$L_{3D} = L_{dis} + L_{size} + L_{yaw} + L_{kpt}. \tag{12}$$

3.4. How Learned Covariances Behave

For monocular 3D object detection, the larger the physical height of an object, the larger the average projected visual height of this object. Thus, H and h_{rec} are negatively correlated. We show predicted covariances in Fig. 4.

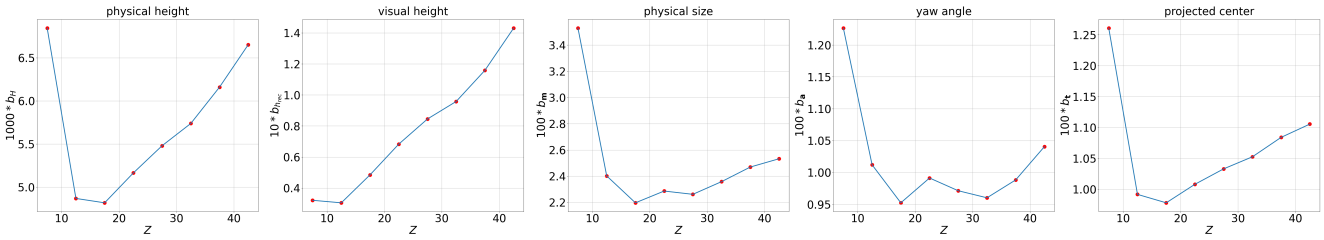


Figure 3: **Predicted uncertainties** of the car class on the val subset of the KITTI val split [6]. We uniformly divide the distance range into 8 intervals and show the average uncertainty of each interval. Predicted uncertainties are larger for nearby truncated objects and faraway small objects.

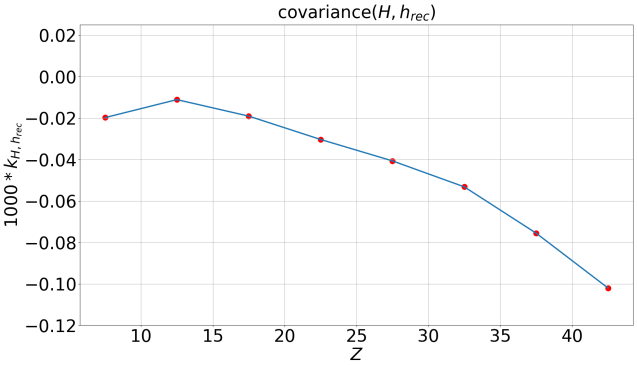


Figure 4: **Predicted covariances** of the car class on the val subset of the KITTI val split [6]. We uniformly divide the distance range into 8 intervals and show the average covariance of each interval. Predicted covariances are negative and their magnitudes increase with the increase of the distance Z.

We can see our model can predict covariances as expected effectively. Explicitly modeling the covariances can make the model achieve accurate prediction of $\mathbf{d} = [H, h_{rec}]^{\mathrm{T}}$ and explicitly learn the correlation between H and h_{rec}. In Tab. 1, we show some challenging cases such as faraway objects (first row), occluded objects (second row), and truncated objects (bottom two rows). We can see that with the negative covariances, the predicted errors of the two heights can be canceled by each other when recovering the distance during inference. We also show predicted uncertainties in Fig. 3. We can see for all variables, their uncertainties are larger for nearby truncated objects and distant small objects.

3.5. Implementation Details

The backbone of MonoRCNN++ is ResNet-50 [16] with FPN [27] and is pretrained on the ImageNet [9]. We extract ROI features (size: $256 \times 7 \times 7$) from P2, P3, P4 and P5 of the backbone, as defined in [27]. We use five scale anchors of $\{32, 64, 128, 126, 512\}$ with three ratios $\{0.5, 1, 2\}$. Each detection head consists of two hidden fully connected layers (size: 1024) and an output fully connected

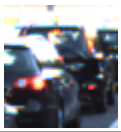
	H (meters) [P/G]	h (pixels) [P/G]	Z (meters) [P/G]	$k_{H,h_{rec}}$ [P]
	1.43/1.51	25.56/27.01	40.29/40.34	-6.67×10^{-5}
	1.52/1.65	38.86/43.28	28.11/27.40	-6.50×10^{-5}
	1.40/1.38	183.31/173.09	5.51/5.75	-4.24×10^{-5}
	1.59/1.63	203.22/209.54	5.65/5.61	-2.95×10^{-5}

Table 1: **Predicted covariances and two heights** on the val subset of the KITTI val split [6]. The predicted errors of the two heights can be canceled by each other when recovering the distance. 'P' means predictions and 'G' means groundtruths.

layer. Images are scaled to a fixed height of 512 pixels for the experiments on the KITTI dataset [13], and 640 pixels for the experiments on the Waymo Open dataset [47]. The training batch size is 8. The total iteration number is 6×10^4, 1.2×10^5 and 1.8×10^5 on the training subset of the KITTI val split [6], the training subset of the KITTI test split [13], and the training subset of the Waymo Open dataset [47], respectively. During training random mirroring and photometric distortion are used as augmentation, and during inference no augmentation is used. We implement our method with PyTorch [36] and Detectron2 [53]. All the experiments run on a server with 2.2 GHz CPU and GTX Titan X.

4. Experiments

We first describe the datasets we use, i.e., the KITTI dataset [13] and Waymo Open dataset [47]. Then we present ablation studies on the KITTI dataset [13]. Finally

we comprehensively benchmark our MonoRCNN++ on the Waymo Open dataset [47] and KITTI dataset [13]. We also visualize qualitative examples.

4.1. Datasets

KITTI dataset [13] provides multiple benchmarks for computer vision problems in autonomous driving. The 3D Object Detection task is used to evaluate the 3D object detection performance. This task provides 7481 training images with 2D and 3D bounding box annotations, and 7518 test images with no annotation. Each object is assigned a difficulty level, i.e., easy, moderate, or hard. We only use the images from the left cameras for training. We train and evaluate our model with the car, pedestrian, and cyclist classes.

Waymo Open dataset [47] is a large-scale, diverse, and challenging autonomous driving dataset. It provides 798 training sequences and 202 validation sequences from different scenes. Following [49], we only use the RGB images from the front camera, consider object labels in the front camera's field of view, and evaluate results on the validation sequences. Following [49], we form our training set ($52\,386$ images) by sampling one frame out of every three frames from the 798 training sequences, and form our validation set ($39\,848$ images) using all the frames from 202 validation sequences. We adopt the official evaluation [47] to calculate the average precision (AP). The evaluation is separated by difficulty level (LEVEL_1, LEVEL_2) and distance to the sensor. Following [49, 26, 21], we evaluate our model with the vehicle class.

4.2. Ablation Studies

We conduct ablation studies to show the effectiveness of modeling the joint probability distribution, as shown in Tab. 2. We show the results of the car class on the val subset of the KITTI val split [6]. We first set the baseline 'B' predicting a diagonal covariance matrix. From Tab. 2, we can see:

1) Modeling the covariance of the physical height and visual height in 3D distance head is effective. Comparing 'B+U+C' with 'B+U', we can see introducing the covariance modeling can improve the 3D object detection accuracy. Specifically, 'B+U+C' surpasses 'B+U' by $9.98\%/5.91\%/5.36\%$ in AP_{3D} and $6.58\%/8.45\%/3.23\%$ in AP_{BEV}. This supports that explicitly modeling the joint probability distribution with a full covariance matrix can achieve accurate prediction of physical height and visual height and explicitly learn the correlation, leading to accurate and robust monocular 3D object detection.

2) Modeling the uncertainties in 3D attribute head is beneficial. Comparing 'B+U' with 'B', we can see introducing the uncertainty modeling can slightly improve the 3D object detection accuracy. We assume that the uncertainty modeling can alleviate the negative influence of noisy training samples during training and makes the model focus on more achievable training samples, which leads to more accurate physical size, yaw angle, and projected center prediction.

	$\text{AP}\vert_{R_{40}}$ [Easy / Mod / Hard] ↑	
	AP_{3D}	AP_{BEV}
B	17.29 / 13.94 / 11.85	24.41 / 18.52 / 16.83
B+U	17.34 / 14.04 / 11.95	24.78 / 19.18 / 16.73
B+U+C	19.07 / 14.87 / 12.59	26.41 / 20.80 / 17.27

Table 2: **Ablation studies** on the val subset of the KITTI val split [6]. 'B' means the baseline. 'U' means using the uncertainty-aware regression loss [19] instead of L_1 regression loss in 3D attribute head. 'C' means modeling the joint probability distribution of the physical and visual height with a full covariance matrix, instead of a diagonal matrix.

4.3. Comparisons on the Waymo Open Dataset

Following [49, 26, 21], we comprehensively benchmark our MonoRCNN++ using the vehicle class on the val set of the Waymo Open dataset [47], shown in Tab. 3. Note that GUPNet [29] and DEVIANT [21] use the scale data augmentation during training to improve their accuracy. Although our MonoRCNN++ does not use this augmentation during training, we can see MonoRCNN++ still achieves the best accuracy. 1) When the IoU threshold is 0.7, our method achieves the best overall 3D AP and surpasses the second [21] by a large margin. Specifically, MonoRCNN++ surpasses DEVIANT [21] by 59.11% / 60.71% in LEVEL_1 / LEVEL_2, respectively. This shows our MonoRCNN++ is significantly better than GUPNet [29], DEVIANT [21], and MonoJSG [26] under the strict evaluation (IoU > 0.7). Our method also achieves the best accuracy for nearby objects within 30 meters, and the second best accuracy for objects beyond 30 meters. 2) When the IoU threshold is 0.5, our method achieves the best overall 3D AP. For nearby objects within 30 meters, our method also achieves the best accuracy. For faraway objects beyond 50 meters, our method achieves the second best accuracy. We also visualize some qualitative examples in Fig. 5.

4.4. Comparisons on the KITTI Dataset

We comprehensively benchmark MonoRCNN++ on the KITTI test dataset [13] in Tab. 4. We can see 1) Comparing MonoRCNN++ with MonoRCNN [43], we can see MonoRCNN++ is better. Firstly, MonoRCNN++ surpasses MonoRCNN [43] by $9.37\%/8.46\%/13.06\%$ in the AP_{3D} of the car class on the easy/moderate/hard subsets, respectively. Secondly, our MonoRCNN++ is a multi-class model while MonoRCNN [43] is a single-class model. 2) With-

Method	Input	LEVEL_1 (IoU >0.5) ↑				LEVEL_2 (IoU >0.5) ↑			
		Overall	0 - 30m	30 - 50m	50m - ∞	Overall	0 - 30m	30 - 50m	50m - ∞
PatchNet (ECCV 20) [30]	I+D	2.92	10.03	1.09	0.23	2.42	10.01	1.07	0.22
PCT (NeurIPS 21) [49]	I+D	4.20	14.70	1.78	0.39	4.03	14.67	1.74	0.36
GUPNet (ICCV 21) [29]	I	10.02	24.78	4.84	0.22	9.39	24.69	4.67	0.19
MonoJSG (CVPR 22) [26]	I	5.65	20.86	3.91	0.97	5.34	20.79	3.79	0.85
DEVIANT (ECCV 22) [21]	I	10.98	26.85	5.13	0.18	10.29	26.75	4.95	0.16
MonoRCNN++ (Ours)	I	11.37	27.95	4.07	0.42	10.79	27.88	3.98	0.39
Method	Input	LEVEL_1 (IoU >0.7) ↑				LEVEL_2 (IoU >0.7) ↑			
		Overall	0 - 30m	30 - 50m	50m - ∞	Overall	0 - 30m	30 - 50m	50m - ∞
PatchNet (ECCV 20) [30]	I+D	0.39	1.67	0.13	0.03	0.38	1.67	0.13	0.03
PCT (NeurIPS 21) [49]	I+D	0.89	3.18	0.27	0.07	0.66	3.18	0.27	0.07
GUPNet (ICCV 21) [29]	I	2.28	6.15	0.81	0.03	2.14	6.13	0.78	0.02
MonoJSG (CVPR 22) [26]	I	0.97	4.65	0.55	0.10	0.91	4.64	0.55	0.09
DEVIANT (ECCV 22) [21]	I	2.69	6.95	0.99	0.02	2.52	6.93	0.95	0.02
MonoRCNN++ (Ours)	I	4.28	9.84	0.91	0.09	4.05	9.81	0.89	0.08

Table 3: **Comparisons on the Waymo Open val set** [47]. We evaluate on the vehicle class and use 3D AP (IoU >0.5 and 0.7) as metric. 'Input' means the input data modality used during training and inference. 'I' denotes image and 'D' denotes depth. Red / blue indicate the best / second, respectively. The results of [30] and [29] are from [49] and [21], respectively.

Method	Input	AP_{3D} [Easy / Mod / Hard] ↑		
		Car	Pedestrian	Cyclist
AM3D (ICCV 19) [31]	I + D	16.50 / 10.74 / 9.52	-	-
PatchNet (ECCV 20) [30]	I + D	15.68 / 11.12 / 10.17	-	-
DDMP-3D (CVPR 21) [48]	I + D	19.71 / 12.78 / 9.80	4.93 / 3.55 / 3.01	4.18 / 2.50 / 2.32
PCT (NeurIPS 21) [49]	I + D	21.00 / 13.37 / 11.31	-	-
Kinematic3D (ECCV 20) [3]	I + V	19.07 / 12.72 / 9.17	-	-
M3D-RPN (ICCV 19) [2]	I	14.76 / 9.71 / 7.42	4.92 / 3.48 / 2.94	0.94 / 0.65 / 0.47
MonoPair (CVPR 20) [7]	I	13.04 / 9.99 / 8.65	10.02 / 6.68 / 5.53	3.79 / 2.12 / 1.83
RTM3D (ECCV 20) [24]	I	14.41 / 10.34 / 8.77	-	-
GrooMeD-NMS (CVPR 21) [22]	I	18.10 / 12.32 / 9.65	-	-
MonoDLE (CVPR 21) [32]	I	17.23 / 12.26 / 10.29	9.64 / 6.55 / 5.44	4.59 / 2.66 / 2.45
MonoRUn (CVPR 21) [4]	I	19.65 / 12.30 / 10.58	10.88 / 6.78 / 5.83	1.01 / 0.61 / 0.48
PGD (CoRL 21) [51]	I	19.05 / 11.76 / 9.39	2.28 / 1.49 / 1.38	2.81 / 1.38 / 1.20
GUPNet (ICCV 21) [29]	I	20.11 / 14.20 / 11.77	14.72 / 9.53 / 7.87	4.18 / 2.65 / 2.09
DEVIANT (ECCV 22) [21]	I	21.88 / 14.46 / 11.89	13.43 / 8.65 / 7.69	5.05 / 3.13 / 2.59
MonoRCNN (ICCV 21) [43]	I	18.36 / 12.65 / 10.03	-	-
MonoRCNN++ (Ours)	I	20.08 / 13.72 / 11.34	12.26 / 7.90 / 6.62	3.17 / 1.81 / 1.75

Table 4: **Comparisons on the KITTI test benchmark** [13]. 'Input' means the input data modality used during training and inference. 'I', 'D', and 'V' denote image, depth, and video, respectively. '-' denotes that results are not available for single-class models.

out using additional data modality, MonoRCNN++ outperforms PGD [51], MonoRUn [4], MonoDLE [32], PCT [49], DDMP-3D [48], and Kinematic3D [3]. 3) Although GUPNet [29] and DEVIANT [21] performs better than our MonoRCNN++ on the KITTI dataset [13], ours performs better on the much larger and more challenging Waymo Open dataset [47]. We argue this is due to the probabilistic learning nature of the covariance modeling and uncertainty modeling. Such a probabilistic learning framework requires ample training samples to discover the intrinsic distribution

Figure 5: **3D detection results of MonoRCNN++** on the test set of the KITTI test split [13] (top row) and val set of the Waymo Open dataset [47] (bottom three rows). MonoRCNN++ predicts accurate 3D bounding boxes for various challenging cases. The red boxes in the image planes represent the 2D projections of the predicted 3D bounding boxes. The yellow / green boxes in the bird's eye view results represent the predictions and groundtruths, respectively, and the red / blue lines indicate the yaw angle. The radius difference between two adjacent white circles is 5 meters.

of the target variables, as discussed in [19]. This coincides with our observation that the Waymo Open dataset [47] in our experiments is about 7 times larger than the KITTI dataset [13]. Another reason for the gap on KITTI [13] can be GUPNet [29] and DEVIANT [21] use the scale data augmentation to improve their accuracy (Tab.7 of [21]), while ours does not use. Finally, we visualize some qualitative examples in Fig. 5.

5. Conclusion

In this paper, we have proposed MonoRCNN++, a probabilistic monocular 3D object detection framework. MonoRCNN++ originally explicitly models the joint probability distribution of the physical height and visual height, which leads to accurate and interpretable monocular 3D object detection. MonoRCNN++ can predict the covariance matrices as expected effectively. The experimental results on the monocular 3D object detection tasks of the challenging Waymo Open [47] and KITTI [13] datasets show the effectiveness of our framework.

Acknowledgments This work is in part sponsored by KAIA grant (22CTAP-C163793-02, MOLIT), NST grant (CRC 21011, MSIT), KOCCA grant (R2022020028, MCST) and the Samsung Display corporation.

References

[1] Lorenzo Bertoni, Sven Kreiss, and Alexandre Alahi. Monoloco: Monocular 3d pedestrian localization and uncertainty estimation. In *ICCV*, 2019.

[2] Garrick Brazil and Xiaoming Liu. M3D-RPN: monocular 3d region proposal network for object detection. In *ICCV*, 2019.

[3] Garrick Brazil, Gerard Pons-Moll, Xiaoming Liu, and Bernt Schiele. Kinematic 3d object detection in monocular video. In *ECCV*, 2020.

[4] Hansheng Chen, Yuyao Huang, Wei Tian, Zhong Gao, and Lu Xiong. Monorun: Monocular 3d object detection by reconstruction and uncertainty propagation. In *CVPR*, 2021.

[5] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *CVPR*, 2016.

[6] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G. Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *NeurIPS*, 2015.

[7] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships. In *CVPR*, 2020.

[8] Jiwoong Choi, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving. In *ICCV*, 2019.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[10] Garoe Dorta, Sara Vicente, Lourdes Agapito, Neill D. F. Campbell, and Ivor Simpson. Structured uncertainty prediction networks. In *CVPR*, 2018.

[11] Di Feng, Lars Rosenbaum, and Klaus Dietmayer. Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection. In *ITSC*, 2018.

[12] Di Feng, Lars Rosenbaum, Fabian Timm, and Klaus Dietmayer. Leveraging heteroscedastic aleatoric uncertainties for robust real-time lidar 3d object detection. In *IV*, 2019.

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *CVPR*, 2012.

[14] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *ICCV*, 2017.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang. Bounding box regression with uncertainty for accurate object detection. In *CVPR*, 2019.

[18] Eskil Jörgensen, Christopher Zach, and Fredrik Kahl. Monocular 3d object detection and box fitting trained end-to-end using intersection-over-union loss. *CoRR*, 2019.

[19] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NeurIPS*, 2017.

[20] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[21] Abhinav Kumar, Garrick Brazil, Enrique Corona, Armin Parchami, and Xiaoming Liu. DEVIANT: Depth EquiVarIAnt NeTwork for Monocular 3D Object Detection. In *ECCV*, 2022.

[22] Abhinav Kumar, Garrick Brazil, and Xiaoming Liu. Groomed-nms: Grouped mathematically differentiable NMS for monocular 3d object detection. In *CVPR*, 2021.

[23] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019.

[24] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. RTM3D: real-time monocular 3d detection from object keypoints for autonomous driving. In *ECCV*, 2020.

[25] Zhuoling Li, Zhan Qu, Yang Zhou, Jianzhuang Liu, Haoqian Wang, and Lihui Jiang. Diversity matters: Fully exploiting depth clues for reliable monocular 3d object detection. In *CVPR*, 2022.

[26] Qing Lian, Peiliang Li, and Xiaozhi Chen. Monojsg: Joint semantic and geometric cost volume for monocular 3d object detection. In *CVPR*, 2022.

[27] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[28] Zongdai Liu, Dingfu Zhou, Feixiang Lu, Jin Fang, and Liangjun Zhang. Autoshape: Real-time shape-aware monocular 3d object detection. In *ICCV*, 2021.

[29] Yan Lu, Xinzhu Ma, Lei Yang, Tianzhu Zhang, Yating Liu, Qi Chu, Junjie Yan, and Wanli Ouyang. Geometry uncertainty projection network for monocular 3d object detection. In *ICCV*, 2021.

[30] Xinzhu Ma, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang. Rethinking pseudo-lidar representation. In *ECCV*, 2020.

[31] Xinzhu Ma, Zhihui Wang, Haojie Li, Pengbo Zhang, Wanli Ouyang, and Xin Fan. Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. In *ICCV*, 2019.

[32] Xinzhu Ma, Yinmin Zhang, Dan Xu, Dongzhan Zhou, Shuai Yi, Haojie Li, and Wanli Ouyang. Delving into localization errors for monocular 3d object detection. In *CVPR*, 2021.

[33] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *CVPR*, 2017.

[34] J. Krishna Murthy, G. V. Sai Krishna, Falak Chhaya, and K. Madhava Krishna. Reconstructing vehicles from a single image: Shape priors for road scene understanding. In *ICRA*, 2017.

[35] J. Krishna Murthy, Sarthak Sharma, and K. Madhava Krishna. Shape priors for real-time monocular object localization in dynamic environments. In *IROS*, 2017.

[36] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming

Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

[37] Liang Peng, Xiaopei Wu, Zheng Yang, Haifeng Liu, and Deng Cai. Did-m3d: Decoupling instance depth for monocular 3d object detection. In *ECCV*, 2022.

[38] Cody Reading, Ali Harakeh, Julia Chae, and Steven L. Waslander. Categorical depth distribution network for monocular 3d object detection. In *CVPR*, 2021.

[39] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[40] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. In *BMVC*, 2019.

[41] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019.

[42] Xuepeng Shi, Zhixiang Chen, and Tae-Kyun Kim. Distance-normalized unified representation for monocular 3d object detection. In *ECCV*, 2020.

[43] Xuepeng Shi, Qi Ye, Xiaozhi Chen, Chuangrong Chen, Zhixiang Chen, and Tae-Kyun Kim. Geometry-based distance decomposition for monocular 3d object detection. In *ICCV*, 2021.

[44] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Manuel Lopez-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *ICCV*, 2019.

[45] Ivor J. A. Simpson, Sara Vicente, and Neill D. F. Campbell. Learning structured gaussians to approximate deep ensembles. In *CVPR*, 2022.

[46] Shiyu Song and Manmohan Chandraker. Joint SFM and detection cues for monocular 3d localization in road scenes. In *CVPR*, 2015.

[47] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020.

[48] Li Wang, Liang Du, Xiaoqing Ye, Yanwei Fu, Guodong Guo, Xiangyang Xue, Jianfeng Feng, and Li Zhang. Depth-conditioned dynamic message propagation for monocular 3d object detection. In *CVPR*, 2021.

[49] Li Wang, Li Zhang, Yi Zhu, Zhi Zhang, Tong He, Mu Li, and Xiangyang Xue. Progressive coordinate transforms for monocular 3d object detection. In *NeurIPS*, 2021.

[50] Tai Wang, Jiangmiao Pang, and Dahua Lin. Monocular 3d object detection with depth from motion. In *ECCV*, 2022.

[51] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Probabilistic and geometric depth: Detecting objects in perspective. In *CoRL*, 2021.

[52] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark E. Campbell, and Kilian Q. Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*, 2019.

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

[54] Bin Yang, Wenjie Luo, and Raquel Urtasun. PIXOR: real-time 3d object detection from point clouds. In *CVPR*, 2018.

[55] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *CoRR*, 2019.

[56] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018.

[57] M. Zeeshan Zia, Michael Stark, and Konrad Schindler. Towards scene understanding with detailed 3d object representations. *IJCV*, 2015.

[58] Zhikang Zou, Xiaoqing Ye, Liang Du, Xianhui Cheng, Xiao Tan, Li Zhang, Jianfeng Feng, Xiangyang Xue, and Errui Ding. The devil is in the task: Exploiting reciprocal appearance-localization features for monocular 3d object detection. In *ICCV*, 2021.