Probabilistic Refinement for RoI-based Monocular 3D Object Detection

Tingyu Zhang, Xinyu Yang, Zhigang Liang, Yanzhao Yang, Jian Wang

Abstract—**Monocular 3D object detection has garnered significant attention due to its cost-effectiveness and simplified setup. In this study, we delve into Region of Interest (RoI)-based monocular detectors. Previous approaches treat all parts of the RoI equally. However, different regions within the RoI hold varying importance, and accurate RoI estimation may be hindered by occlusions or long distances. Thus, we introduce the Multi-Scale Grid Attention (MSGA) mechanism to investigate multi-scale RoI exploration and the significance of RoI regions. Moreover, while existing methods treat depth estimation as a probability estimation during training, they do not effectively utilize probabilistic properties during inference. To tackle these issues, we propose a novel probabilistic post-processing method to enhance detection robustness. Experimental evaluations are conducted on the KITTI and Waymo datasets, achieving state-of-the-art performance.**

Index Terms—**Camera, Intelligent vehicles, Monocular 3D object detection.**

I. Introduction

MONOCULAR camera-based 3D object detection has become a significant research area in computer vision, with applications spanning autonomous driving, robotics, and augmented reality. Unlike conventional approaches dependent on depth sensors or multi-camera setups, monocular 3D object detection leverages the capabilities of a single camera to estimate the three-dimensional spatial information of objects within its field of view.

The significance of monocular 3D object detection lies in its potential to provide depth perception using only the information captured by a single camera. This is particularly advantageous for real-world applications where cost, simplicity, and computational efficiency are essential considerations. Achieving accurate and robust 3D object detection from monocular images involves addressing complex challenges such as scale ambiguity, occlusions, and perspective distortions inherent to a single viewpoint.

The absence of explicit depth information poses a significant challenge for monocular 3D detectors, leading to a considerable disparity compared to point cloud-based methods. To bridge this gap, certain approaches [8], [11], [18] initially forecast multiple RoI candidates and subsequently generate predictions for each. 3D RoI-based methods, such as those discussed in [18], often depend on pre-trained 3D monocular detectors. In contrast, 2D RoI-based methods solely predict 2D bounding boxes, offering a lighter and more efficient alternative. Furthermore, given the real-time demands of 3D object detection, efficiency is paramount. Therefore, this paper exclusively focuses on 2D RoI-based methods.

2D RoI-based methods [8], [11] typically involve three key steps. Firstly, the RoI is partitioned into distinct segments, followed by the application of RoI feature extraction techniques (such as RoI Pooling [27], RoI Warp [28], and RoI Align [23]) to extract pertinent features. Secondly, the RoI is fed into a neural network to further refine these features and generate predictions for each region. Finally, the individual predictions are integrated to yield the final results.

However, the second and third steps present certain challenges. In the second step, each partition of the RoI is treated equally in previous works, disregarding the varying importance of different RoI regions. For instance, empirically, information pertaining to the tire or license plate number holds more significance than that concerning the car body, and background regions should have less influence than foreground areas. Taking DID-M3D [11] as an example, it predicts the depth to the object surface, referred to as visual depth, to decouple the depth. In the KITTI [29] dataset, the training data for the depth completion task and 3D object detection task do not entirely overlap, resulting in some images in KITTI 3D lacking ground truth visual depth. Therefore, DID-M3D introduces a depth completion network to generate a dense visual depth map. We select images with depth labels and compare the ground truth with the depth map generated by the depth completion network. As depicted in Fig. 1, visual depth near the center of an object is more precise than at the edges. Taking the central visual depth of the RoI as the baseline, the distribution of visual depth in the RoI is imbalanced, as shown in Fig. 2. Additionally, 2D RoIs are generated by the network, potentially resulting in incomplete object warping. To address these issues, we propose the MSGA module. Specifically, we enlarge the RoI by adding various pixels to its size, extract multi-scale features using a network, and apply a Grid Attention (GA) module to assess the importance of different regions and further adjust the RoI features. In the third step, the results are integrated to obtain the final outcomes. Previous approaches commonly employed simple averaging or weighted summation. While adequate for properties like dimensions and position, where each grid shares the same regression target, this method encounters challenges in depth estimation. In particular, depth is decomposed into visual depth and attribute depth [11], as illustrated in Fig. 3. Each grid possesses distinct visual and attribute depths, leading to an imbalanced distribution that renders simple averaging subopti-

Tingyu Zhang, Zhigang Liang, Yanzhao Yang and Jian Wang are with College of Computer Science and Technology, Jilin University, Changchun, China

Xinyu Yang is with Automotive safety and Simulation Test Department, China Automotive Innovation Corporation, Nanjing, China

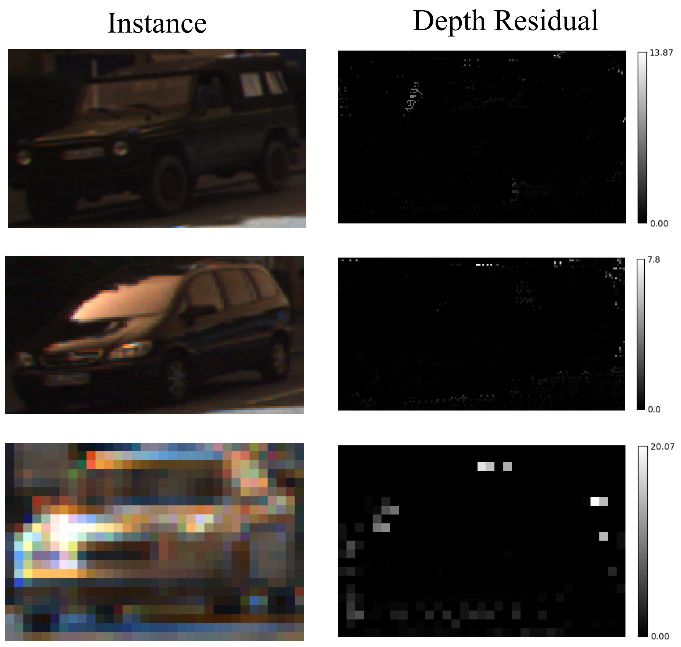


Fig. 1. Instance and its depth residual. The depth residual of an instance is computed by taking the absolute difference between the predicted depth generated by the depth completion network and the corresponding ground truth. The left column shows the raw image of the instance, while the right column illustrates the depth residual. Lighter colors in the latter column indicate higher depth residuals.

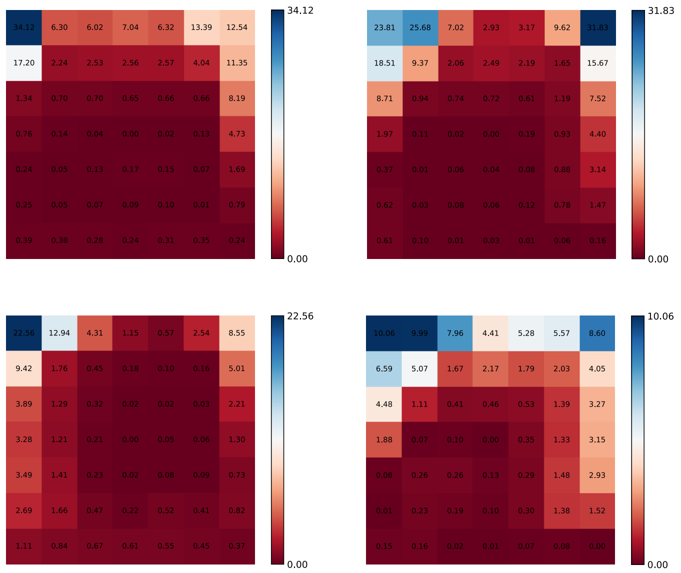


Fig. 2. Visual depth offset. The visual depth of the central grid within the RoI is considered the baseline, from which the offset to other grids is computed. This absolute offset is subsequently labeled for each grid.

mal. Moreover, deriving the weights for weighted summation proves challenging. In our study, we adopt the assumption from previous research [8], [11] that depth follows a certain distribution, leveraging this probability distribution in the post-processing phase. This approach enhances the integration process's rationality.

In summary, our contributions can be outlined as follows:

- We meticulously explore the limitations of 2D RoI-based monocular 3D object detection, highlighting an inconsistency in the significance of RoI grids. We introduce the notion of varying importance among RoI grids and propose the utilization of Multi-Scale Grid Attention to address this issue.
- During the post-processing phase, we meticulously consider the assumed depth distribution, employing probabilistic methods for refinement.
- The experimental results underscore the superiority of our method compared to existing approaches on the KITTI and Waymo datasets.

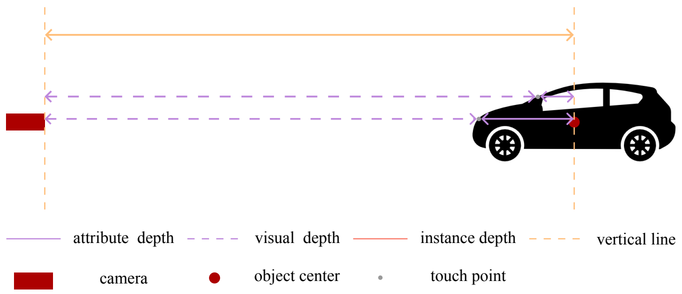


Fig. 3. Decoupled depth. The instance depth is defined as the distance between the object center to the camera plane. And the visual depth is distance of object surface to the camera plane. The attribute depth is obtained by the minus of instance depth and visual depth.

II. Related Work

A. Monocular 3D Object Detection without RoI

Given the inherent challenge of directly estimating instance depth with a monocular camera due to its ill-posed nature, previous studies have endeavored to leverage these constraints effectively. Mousavian et al. [1] introduced constraints between 2D and 3D bounding boxes to formulate an equation group, integrating specific priors like driving direction to constrain the equation's degrees of freedom. PGD [13] and MonoPair [3] explore relationships between different instances, while DDMP-3D [39] considers the relationships between neighboring pixels, utilizing these constraints to refine box estimations.

Some alternative methods [14], [30], [31] employ numerous anchors placed on the 3D plane and extract anchor features through projection. These approaches encounter common issues associated with anchor-based methods, including the proliferation of anchors and the nontrivial configuration of anchor hyperparameters.

Due to the scale variance in the image plane and the notable success of LiDAR detectors, several methods endeavor to convert the 2D image plane into the 3D plane. OFT [32] and ImVoxelNet [12] utilize the projection of predefined 3D voxels onto the image plane to populate the features of each voxel. CaDDN [7] partitions the depth range into segments and predicts the probability for each segment, followed by the expansion of pixel features into frustum features. MonoNeRD [19] leverages 2D image features to generate NeRF-like 3D representations. Additionally, various pseudo-LiDAR methods [24], [33], [34] integrate an independent depth completion network to generate a dense depth map, using the calibration matrix to derive the pseudo LiDAR. Subsequently, a LiDAR-based detector is employed to obtain the final results. Despite its success, DD3D [9] argues that pseudo LiDAR is unnecessary for detection. Instead, it employs a single model to predict depth and 3D bounding box simultaneously, obviating the need for an independent depth completion network.

Monocular 3D object detection heavily relies on accurate depth estimation [37]. Even slight errors in depth estimation can lead to objects being significantly offset from ground truth positions. To address this challenge, many methods aim to introduce additional supervision to constrain depth offsets. Key point estimation, as demonstrated in works such as Polygon [5], is commonly employed because it imposes eight additional constraints on the model. Representative methods utilizing this approach include RTM3D [35], SMOKE [4], MonoDDE [10], and Monocon [15].

There are several methods that employ other different strategies to improve performance. For instance, M3DSSD [38], MonoPGC [20], CIE [16], and SSD-MonoDETR [22] incorporate attention mechanisms to enhance performance. MonoFlex [6] specifically addresses truncated objects by predicting edge heatmaps for them.

B. Monocular 3D Object Detection with RoI

RoI-based object detection can be broadly categorized into 3D RoI-based and 2D RoI-based methods. 3D RoI-based approaches [17], [18] typically depend on a separately pretrained monocular 3D object detector to produce 3D proposals, a process that does not meet the real-time demands of autonomous driving. In contrast, 2D RoI-based methods [2], [8], [11], [21] share most of the network in both RoI generation and object detection, rendering them more efficient.

In this study, our emphasis is on 2D RoI-based methods. Recognizing the challenges posed by RoI generation noise and the varying significance of RoI components, we introduce the MSGA module to effectively address these issues. Additionally, we introduce a novel probabilistic post-processing strategy to leverage the probabilistic assumptions inherent in depth estimation.

III. Method

A. Problem Definition

In the domain of monocular 3D object detection, the primary input consists of the RGB image I. The main goal is to determine essential properties of the 3D bounding boxes, including the 3D center coordinates x_c, y_c, z_c, the 3D dimensions l, w, h and the observation angle θ, which is more related to image appearance than orientation angle [1]. A crucial element in this procedure is the projection matrix P, as described in Eq. (1).

$$P = \begin{pmatrix} f & 0 & c_u & -fb_x \\ 0 & f & c_v & -fb_y \\ 0 & 0 & 1 & -fb_z \end{pmatrix} \tag{1}$$

where f denotes the focal length, while c_u and c_v denote the vertical and horizontal positions of the camera in the image, respectively. Furthermore, b_x, b_y and b_z indicate the baseline relative to the reference camera. Notably, these values are non-zero in the KITTI dataset and zero in the Waymo dataset.

In a monocular setting, directly determining the 3D center position poses a significant challenge, mainly due to the considerable variability in 3D center scale. As a result, many studies opt to predict the projected 3D center on the image plane, denoted as x_{ic}, y_{ic}, along with the corresponding depth d. The recovery of the 3D bounding box center is then accomplished using Eq. (2).

$$dx_{2d} = Px_{3d} \tag{2}$$

where P denotes the projection matrix, x_{3d} represents the homogeneous 3D bounding box center $(x_c, y_c, z_c, 1)^T$, x_{2d} signifies the homogeneous projected 3D center on the image plane $(x_{ic}, y_{ic}, 1)^T$, and d denotes the depth of x_{3d}.

B. Overview

The schematic overview of our methodology is depicted in Fig. 4. Initially, the provided image I is processed through the image backbone, as discussed in Section III-B1. Subsequently, a 2D detection head is employed to extract 2D properties, including the width and height of the bounding box, and to generate the heatmap for the projected 3D object center, as elaborated in Section III-B2. Key point estimation, following the methodology of Monocon [15], is incorporated into our approach. Utilizing the heatmap in conjunction with the predicted width and height, the 2D RoIs are determined. The RoIs are then input into the MSGA module to refine the features, as explained in Section III-B3. Employing the DID-M3D [11] approach for each RoI grid, we predict both the visual depth and attribute depth, enabling subsequent prediction of 3D properties such as dimensions and observation angle. In the post-processing phase, we apply a novel depth integration strategy to consolidate RoI depths, taking into account the practical significance of probability distribution. This strategy is outlined in Section III-C.

1) Image Backbone: Given an input RGB image I with dimensions $3 \times H \times W$, we utilize a feature backbone $f(\cdot; \Theta)$ to compute the feature map F with dimensions $D \times h \times w$ as Eq. (3):

$$F = f(I; \Theta) \tag{3}$$

where Θ represents all learnable parameters, D denotes the output feature map dimension (e.g., $D = 512$), and h and w are determined by the overall subsampling rate s in the backbone (e.g., $s = 4$). We employ the DLA-34 [40] network as our chosen backbone.

2) 2D Detection Head: Utilizing the output feature map F from the backbone as input, we route it through three detection heads. Each detection head comprises a series of operations: a 2D convolution, Rectified Linear Unit (ReLU) activation function, followed by another 2D convolution. Specifically, the first detection head is responsible for predicting the heatmap H indicating the projected 3D object center. The process of heatmap generation follows the methodology outlined in CenterNet [41]. The second detection head focuses on predicting the offsets Δx and Δy between the projected 3D object center and the center of the 2D bounding box. Finally, the third

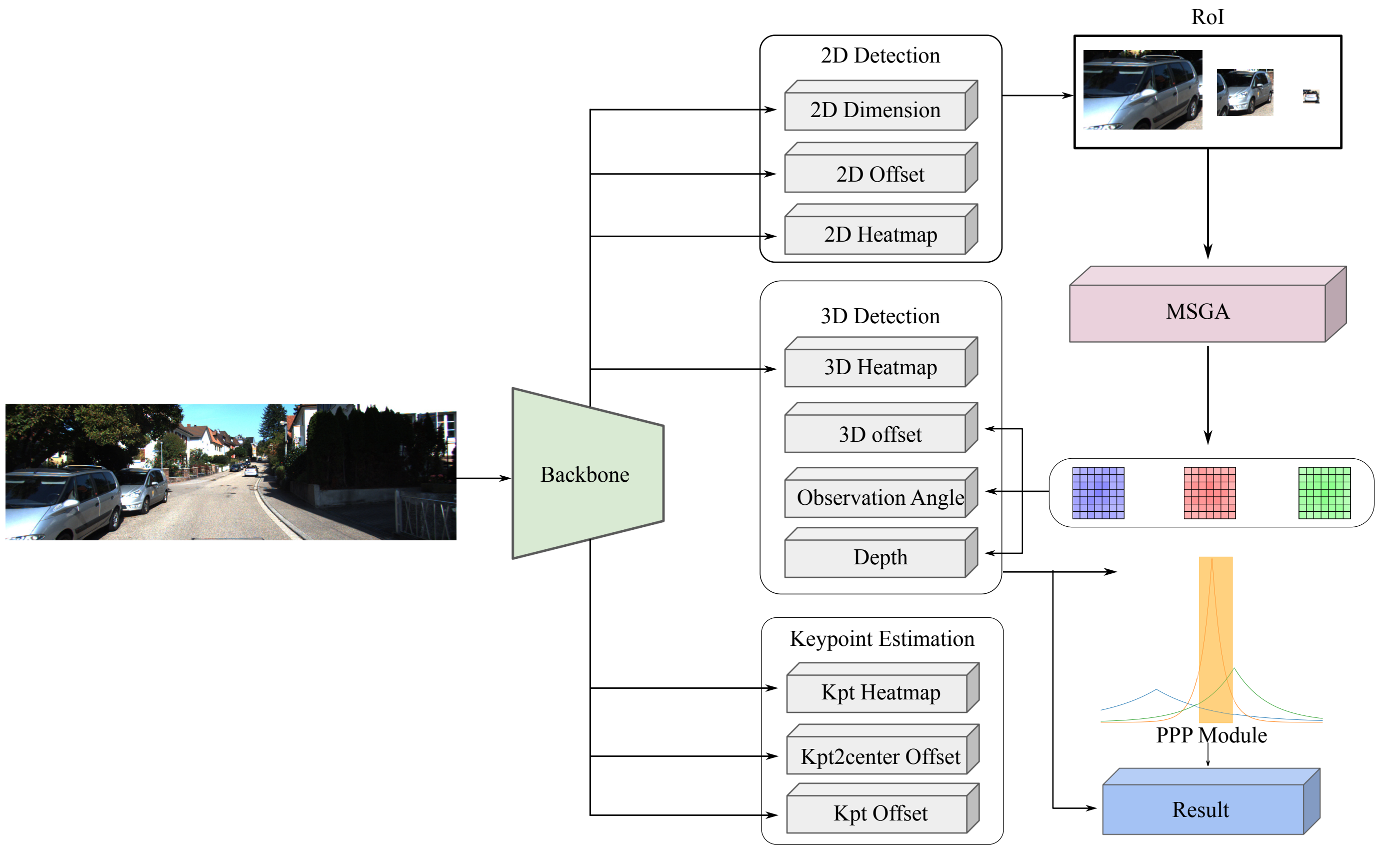


Fig. 4. Overview of our model. Initially, the input image undergoes processing through the backbone network to extract image features. These features are subsequently utilized for 2D detection, keypoint estimation, and 3D heatmap generation. Leveraging the results of 2D detection, the Region of Interest (RoI) features are directed to the Multi-Scale Grid Attention (MSGA) module for refinement. The enhanced RoI features are then employed to predict the 3D offset, observation angle, and depth. The depth estimation undergoes optimization via the Probabilistic Post-Processing (PPP) module. Finally, the results are decoded by both 3D properties and optimized depth.

detection head is tasked with predicting the width w_{2d} and height h_{2d} of the 2D bounding box.

3) Multi-Scale Grid Attention Module: During the training phase, we utilize the ground truth projected 3D object center $(x_{c_{gt}}, y_{c_{gt}})$, predicted offsets Δx and Δy, as well as the predicted width w_{2d} and height h_{2d} of the 2D bounding box to compute the RoI using Eq. (4). Each RoI is defined by its top-left and bottom-right boundary points.

$$\begin{aligned} RoI = &(x_{c_{gt}} - w_{2d}/2, y_{c_{gt}} - h_{2d}/2, \\ &x_{c_{gt}} + w_{2d}/2, y_{c_{gt}} + h_{2d}/2) \end{aligned} \tag{4}$$

During the inference phase, we select the top 50 positions from the heatmap to represent the predicted projected 3D object center. Subsequently, we combine the predicted offset and 2D dimensions to generate the RoI.

Adjacent background information plays a crucial role in distinguishing foreground instances [42], particularly when the predicted RoI may not accurately encapsulate the object. Similarly, in point cloud-based 3D object detection, such as Pyramid-RCNN [42], 3D RoIs encounter analogous challenges. To address this issue, enlarging the RoI is a common strategy. In Pyramid RCNN, the RoI is expanded proportionally based on the original RoI. In 3D dimensions, scale remains constant, implying consistent object dimensions irrespective of distance. In contrast, in the image plane, scale varies with distance. Proportional enlargement may not be optimal, potentially introducing excessive background pixels for nearby objects. Thus, we adopt a fixed number of pixels enlargement strategy for the RoI. Determining the required pixels for RoIs with different sizes and varying accuracies of 2D bounding box prediction is challenging. To overcome this challenge, we propose a Multi-Scale configuration, illustrated in Fig 5. Initially, the original RoI is cropped by the predicted 2D bounding box. We uniformly increase the width and height of the 2D bounding box by 5 and 15 pixels, respectively, to create the enlarged RoI. Subsequently, RoI Align [23] is applied to generate 7×7 RoI features. A 1×1 convolution and sigmoid function are then applied to generate the attention map for each grid. The RoI feature is multiplied by the attention map, and the result is added to the RoI feature to obtain the merged feature. Finally, the multi-scale merged features are concatenated to form the final feature.

The MSGA module consists of two key components: the multi-scale RoI feature and the Grid Attention mechanism. The advantages of employing the multi-scale RoI feature are elucidated as follows:

- Enhanced inclusion of foreground pixels: multi-scale features mitigate errors in 2D bounding box predictions, improving the likelihood of encompassing whole foreground pixels of the predicted object.

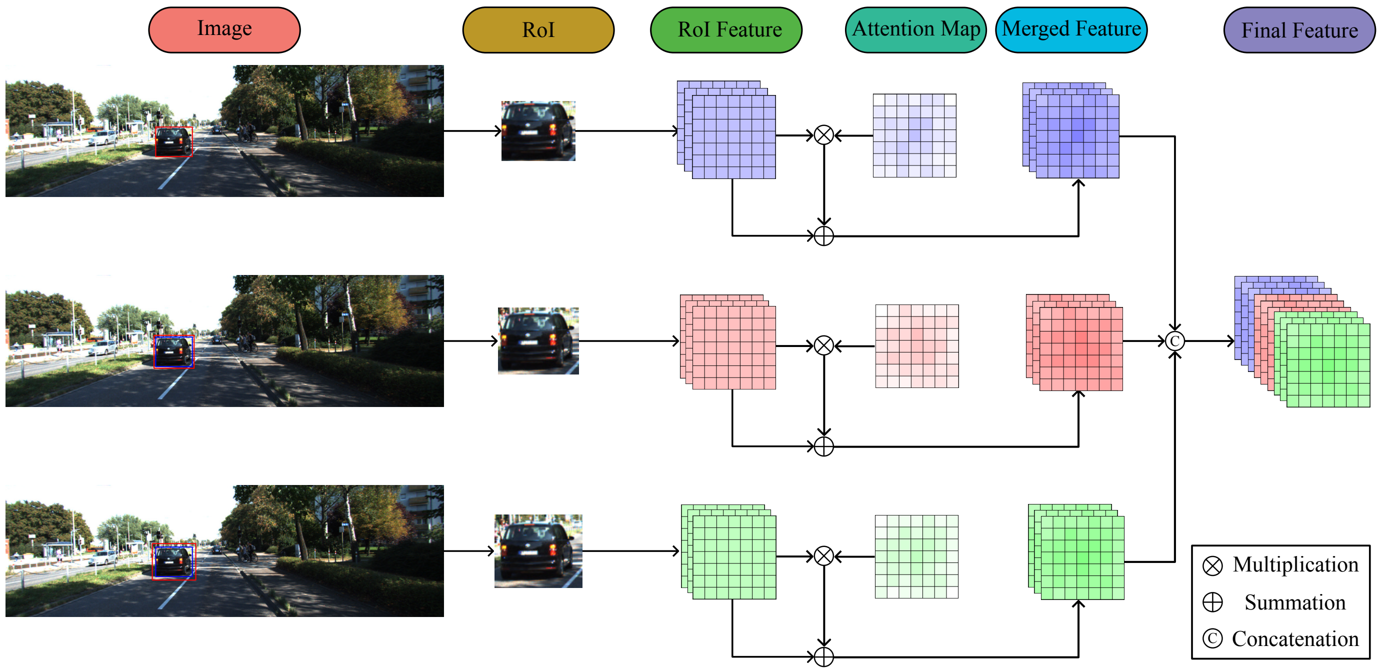


Fig. 5. Multi-Scale Grid Attention Module. We use three scale to generate the RoI, the RoI feature is obtained by RoI Align. We use 1D convolutional network to generate the attention map of RoI grid. Then we multiply the attention map and the RoI feature, fed to residual summation for the merged feature. The final feature is integrated by the multi-scale merged feature.

- Improved background information utilization: by incorporating adjacent background pixels, multi-scale features enable better extraction of information from the background region.

The advantages of the Grid Attention mechanism are summarized as follows:

- Discriminative pixel attention: grid attention enables the model to discern between background and foreground pixels, prioritizing for more accurate training. This discrimination enhances the model's ability to focus on relevant visual elements.
- Importance discrimination among foreground pixels: grid attention further discriminates among foreground pixels, assigning higher importance to specific elements. For instance, identifying features like a vehicle's license plate, which serves as a distinctive identifier, is prioritized.

4) 3D Detection Head: For each set of RoI features, we employ seven distinct detection heads to predict various properties. These include the 3D dimensions offset, denoted as dim_{3d}, in comparison to the mean size of each class; the offset $offset_{3d}$ representing the quantization error between the projected 3D object center and the corresponding pixel; the visual depth d_{vis} and its associated uncertainty σ_{vis}; the attribute depth d_{att} and its corresponding uncertainty σ_{att}; and the observation angle θ.

The detection heads responsible for predicting dim_{3d}, $offset_{3d}$, and θ consist of a sequence comprising a 2D convolution layer followed by batch normalization, Rectified Linear Unit, and another 2D convolution layer. Conversely, the detection heads tasked with predicting d_{vis}, σ_{vis}, d_{att}, and σ_{att} are composed of a 2D convolution layer, LeakyReLU activation function [43], and another 2D convolution layer.

5) Keypoint Estimation: For keypoint estimation, we adopt the approach proposed by Monocon [15]. Specifically, we estimate the keypoint heatmap $H_k \in \mathbb{R}^{9\times w\times h}$ to predict the projected positions of the 8 corners and the center of the 2D bounding box. Estimating H_k involves a series of operations: a 2D convolution, ReLU activation, and another 2D convolution. Additionally, we predict the offsets $(\Delta k2c_x, \Delta k2c_y)$between each keypoint and the projected 3D object's center pixels, as well as the quantization error $(\Delta kx, \Delta ky)$ resulting from rounding off the keypoint positions. These components are determined through a sequence of operations: a 2D convolution, batch normalization, ReLU activation, and another 2D convolution. It's important to note that keypoint estimation is employed solely during training to provide supplementary supervision and is not utilized during inference.

6) Loss Functions:

a) 2D Heatmap: Our loss function follows the methodology outlined in CenterNet [41]. To guarantee a bounding box overlap of at least 0.7 IoU with the ground truth, we calculate the radius accordingly. Specifically for H, we utilize a modified focal loss, defined as Eq. (5):

$$L_{heat} = \frac{-1}{N}\sum_{H}\begin{cases}(1-H)^{\alpha}\log(H) & \text{if } \hat{H}=1\\ (1-\hat{H})^{\beta}(H)^{\alpha} & \\ \log(1-H) & \text{otherwise}\end{cases} \tag{5}$$

where $\hat{H}$ denotes the target heatmap, with N representing the number of keypoints in the image. For our experiments, we set the hyper-parameters α and β for the focal loss to 2 and 4, respectively.

b) 2D Box: For the 2D bounding boxes, we predict the offsets between the peak in the 2D heatmap H and the center of the 2D bounding box, denoted as Δx_{2d} and Δy_{2d}, along with the sizes of the bounding box, represented by h_{2d} and w_{2d}. These values are determined using the L1 loss, as formulated in Eq. (6).

$$L_{box_{2d}} = \sum_{o \in \{\Delta x_{2d}, \Delta y_{2d}, w_{2d}, h_{2d}\}} |o - \hat{o}| \tag{6}$$

where o denotes the predicted value and $\hat{o}$ denotes the regression target.

c) 3D Box: For 3D box, we need to regress the offset between the peak in 2D heatmap and the projected 3D object center $\Delta x_{3d}, \Delta y_{3d}$, along with the offset of ground truth dimensions and the averaged dimensions $\Delta l_{3d}, \Delta w_{3d}, \Delta h_{3d}$. They are all calculated by the L1 loss, which is formulated as Eq. (7).

$$L_{box_{3d}} = \sum_{o \in \{\Delta x_{3d}, \Delta y_{3d}, \Delta l_{3d}, \Delta w_{3d}, \Delta h_{3d}\}} |o - \hat{o}| \tag{7}$$

d) Depth: For depth estimation, we make the assumption that the depth follows the Laplace distribution [11], and use Eq. (8) to calculate different depth and its uncertainty.

$$L_{depth} = \sum_{* \in \{vis, att, ins\}} \frac{\sqrt{2}}{e^{\frac{u_*}{2}}} \times |d_* - \hat{d}_*| + \frac{u_*}{2} \tag{8}$$

where d_{vis}, d_{att}, and d_{ins} represent the visual depth, attribute depth, and instance depth, respectively, while u_{vis}, u_{att}, and u_{ins} denote the uncertainty associated with each depth. The symbol $\hat{d}_*$ denotes the regression target. It's essential to clarify that our network predicts only the visual depth and attribute depth, with the instance depth computed as the sum of these two. Additionally, we calculate the instance depth uncertainty using Eq. (9).

$$u_{ins} = log(e^{u_{vis}} + e^{u_{att}}) \tag{9}$$

e) Orientation Angle: We employ the Multi-bin loss [1] for the observation angle. Specifically, we partition 2π into 12 bins, and the network generates the classification vector θ_{cls} to identify the bin in which the angle lies, along with the offset θ_{reg} between the bin center and the ground truth angle. The cross-entropy loss is utilized for θ_{cls}, as formulated in Eq. (10).

$$L_{angle_{cls}} = -\sum_{i=1}^{12} y_i \log(p_i) \tag{10}$$

where y_i as the indicator for the ith bin, where it equals 1 if the angle falls within the bin, and 0 otherwise. p_i represents the predicted probability of the ith bin. As for θ_{reg}, we employ the L1 loss, as formulated in Eq. (11).

$$L_{angle_{reg}} = |\theta_{reg} - \hat{\theta_{reg}}| \tag{11}$$

where $\hat{\theta_{reg}}$ is the regression target.

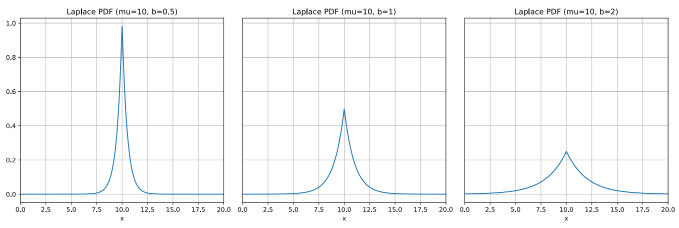


Fig. 6. Laplace Probability Density Function. Left distribution means $\mu = 10, b = 0.5$. Middle distribution means $\mu = 10, b = 1$. Right distribution means $\mu = 10, b = 2$.

f) Keypoints: We incorporate an auxiliary keypoint loss inspired by Monocon [15]. To generate the keypoint heatmap $H_k \in \mathbb{R}^{9 \times w \times h}$, we project the eight corners onto the image plane and use the center of 2D bounding box, thus representing nine keypoints. The loss function mirrors that of the 2D heatmap, albeit with a looser IoU threshold of 0.3, as adopted in Monocon [15]. This loss function is detailed in Eq. (12).

$$L_{H_k} = \frac{-1}{N} \sum_{H_k} \begin{cases} (1 - H_k)^{\alpha} \log(H_k) & \text{if } \hat{H}_k = 1 \\ (1 - \hat{H}_k)^{\beta} (H_k)^{\alpha} & \\ \log(1 - H_k) & \text{otherwise} \end{cases} \tag{12}$$

In addition to the keypoint heatmap, we also predict the offset between each keypoint and the projected 3D object center pixel, denoted as $\Delta k2c_x$ and $\Delta k2c_y$, along with the offset between the peak in the keypoint heatmap and the keypoint position, represented as Δkx and Δky. These offsets are all calculated using the L1 loss, as formulated in Eq. (13).

$$L_{keypoint} = \sum_{i=1}^{9} \sum_{o \in \{\Delta k2c_x, \Delta k2c_y, \Delta kx, \Delta ky\}} |o^i - \hat{o^i}| \tag{13}$$

where o^i denotes the prediction of the i-th keypoint, while $\hat{o^i}$ signifies the target corresponding to the i-th keypoint.

The total loss is computed as the summation of the aforementioned losses, as defined in Eq. (14).

$$\begin{aligned} L_{total} =& \lambda_{heat} L_{heat} + \lambda_{box_{2d}} L_{box_{2d}} + \lambda_{box_{3d}} L_{box_{3d}} \\ &+ L_{depth} + \lambda_{angle}(L_{angle_{cls}} + L_{angle_{reg}}) \\ &+ \lambda_{H_k} L_{H_k} + \lambda_{keypoint} L_{keypoint} \end{aligned} \tag{14}$$

where λ_* is the weight of each loss, is dynamically adjusted through hierarchical task learning, as proposed by GUPNet [8]. Details of this adaptation process will be elucidated in Section IV-B.

C. Probabilistic Post Processing

Previous studies have assumed that depth adheres to the Laplace distribution. The Probability Density Function (PDF) of the Laplace distribution is defined as shown in Eq. (15).

$$f(x|\mu, b) = \frac{1}{2b} \exp\left(-\frac{|x - \mu|}{b}\right) \tag{15}$$

where μ represents the positional parameter, and b represents the scale parameter. As illustrated in Fig. 6, decreasing b sharpens the PDF curve, while increasing it flattens the curve. In our work, the predicted uncertainty is the standard

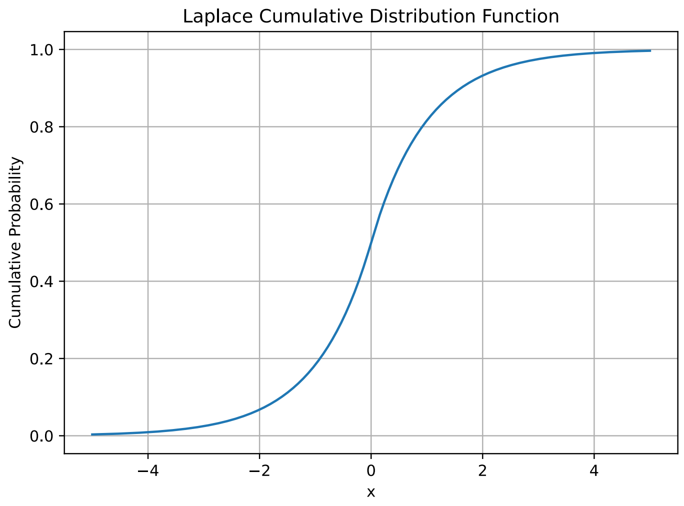


Fig. 7. Laplace Cumulative Density Function.

derivation of the Laplace distribution which is directly proportional to b. When uncertainty is smaller, the depth distribution becomes more concentrated, whereas larger uncertainty results in a more dispersed depth distribution.

Previous methods [8], [11] have utilized the standard deviation σ which is adhere to the predicted uncertainty to assess the accuracy of depth estimation, employing $e^{-\sigma}$ to gauge the confidence level of the estimation. However, the rationale behind importance estimation lacks a solid theoretical foundation. In GUPNet [8], it is stated that their objective is merely to normalize the value between 0 and 1. In this study, we introduced the Probabilistic Post-Processing (PPP) module to incorporate probabilistic distributions into our method.

Based on fundamental probabilistic principles, the probability of a variable lying within the narrow range $[x-\delta, x+\delta]$ can be computed by calculating the area bounded by the PDF curve and the axis within that range. This computation can be performed using a definite integral or by subtracting the Cumulative Density Function (CDF). The CDF for the Laplace distribution is defined as shown in Eq. (16).

$$\begin{aligned} F(x) &= \int_{-\infty}^{x} f(u)\mathrm{d}u \\ &= \begin{cases} \frac{1}{2}\exp\left(-\frac{\mu-x}{b}\right) & \text{if } x<\mu \\ 1-\frac{1}{2}\exp\left(-\frac{x-\mu}{b}\right) & \text{if } x\geq\mu \end{cases} \\ &= 0.5\left[1+\operatorname{sgn}(x-\mu)\left(1-\exp(-|x-\mu|/b)\right)\right] \end{aligned} \tag{16}$$

where $\operatorname{sgn}(\cdot)$ denotes the sign function, while $F(x)$ represents the probability of a sample lying within the range $(-\infty, x]$. The CDF curve is illustrated in Fig. 7.

In this work, we use the idea of maximum likelihood function. As above said, we can calculate the probability of the variable locating in the range $[x-\delta, x+\delta]$ by $F(x+\delta)-F(x-\delta)$. And we define the likelihood function as Eq. (17).

$$L(x) = \sum_{1}^{49} F(x+\delta) - F(x-\delta) \tag{17}$$

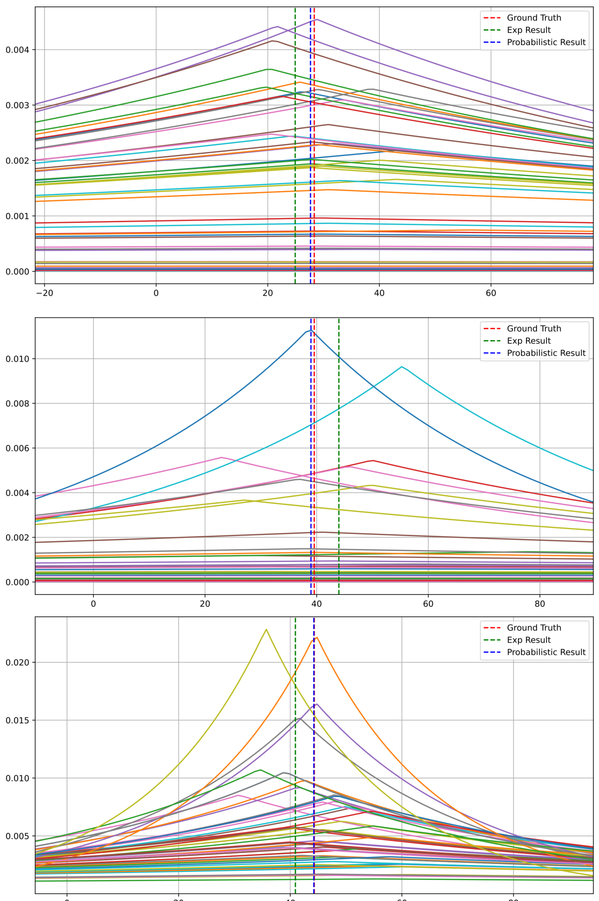


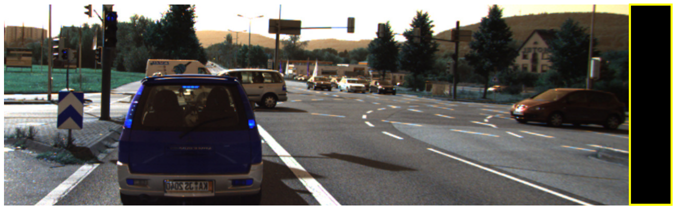
Fig. 8. Examples of PPP modules. The red dotted line denotes the ground truth instance depth, while the green dotted line represents the predicted instance depth computed through exponential weighted summation. Our method's predicted depth is indicated by the blue dotted line.

We aim to find the optimal depth value that maximizes the likelihood of observing the predictions within a narrow range centered around it. We said we use the idea of maximum likelihood function but not direct this method, because maximum likelihood function is that we know about which distribution the real data follows, and we use the observed data to predict the distribution parameters. However, in our method, we use the predicted distribution to determine the observed position with maximum likelihood. We think our method is more like a multi-model fusion method. In Fig. 8, we give some examples of the application of our method.

IV. Experiments

A. Dataset

1) KITTI Dataset: The KITTI dataset [29] is widely acknowledged as a standard benchmark in the field of 3D object detection. It comprises 7,481 images paired with finely calibrated 3D bounding boxes for training, as well as 7,518 samples designated for testing. We split the training set into two subsets: a training with 3,712 samples and a validation set with 3,769 samples. This division was employed for fine-tuning and optimizing hyperparameters during model development. For the final submissions to the KITTI test server, we train the model on the whole dataset and use the weight of the last epoch.


crop size < image size

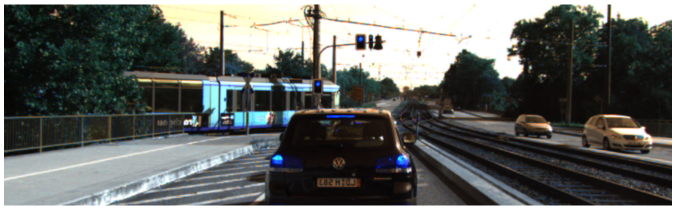

crop size > image size

Fig. 9. Different circumstance of random cropping. When the crop size is smaller than the image size, the image is scaled to fit the crop size, resulting in black edges around the image. Conversely, if the crop size exceeds the image dimensions, the image is scaled to match the crop size, retaining only the portion within the image boundary.

2) Waymo Dataset: We perform experiments on Waymo dataset [46], which is a large-scale modern dataset for self-driving. It contains 798 sequences for training and 202 sequences for validation. We sample every 3^{rd} frame from the training sequence to form a small training set like CaDDN. The processed training sets have approximately 50k frames.

B. Implementation Details

In MSGA module, we use 1×1 convolution and leaky ReLU and 1×1 convolution along with a Sigmoid function to get the attention map. For enlarged RoI, we add no pixels, 5 pixels and 15 pixels, to each side of the RoI. We have test different configuration in the Sec IV. We set $\sigma = 0.1$ to refine our post processing process. The value of σ is choosen based on the experiments. We use Adam as our optimizer, and set weight decay as 0.00001. We train our model for 200 epochs. At the first 5 epochs, we use the cosine function to transform the learning rate from 0.00001 to 0.001. The learning rate is set to 0.001 in the remaining epochs. And will decay by 0.1 at the 90 and 120 epoch. In the training phase, we remove the frame with no ground truth label for more stable and robust training. For visual depth labeling, we utilize the LRRU method [45] to execute depth completion, thereby generating labeled visual depth data for network training.

For data augmentation, we use random flip and random crop. After flipping and cropping, affine transformation will be applied on the cropped image to resize the image to the size of $(1280, 384)$ for batch processing. It should be noted that, after cropping and affine transformation, it may cause the processed image having black edge or remove some objects outside the picture, as depicted in Fig. 9. For the first condition, for the keypoint of centerpoint that locates in the black space, we calculate their heatmap. For that outside the whole image, we do not calculate their heatmap. But in training phase, the heatmap of black edge do not participate into the loss calculation. For the second condition, if centers of all objects are outside the image, we random sample another item in the dataset.

In the PPP module, optimizing depth determination is framed as an optimization problem, for which we employ the Brent algorithm [48] to find the solution. During the training phase, we adopt the hierarchical task learning strategy introduced by GUPNet [8]. This approach involves training different tasks sequentially. Initially, we focus on training the 2D object detection task, which includes generating 2D heatmaps and bounding boxes. Throughout this phase, λ_{heat} and $\lambda_{box_{2d}}$ are both set to 1. Subsequently, the weights $\lambda_{box_{3d}}$ and λ_{angle} are adjusted based on the convergence of 2D bounding box predictions. The weights λ_{depth}, λ_{H_k}, and $\lambda_{keypoint}$ are determined by the convergence of both 2D bounding box predictions and 3D size predictions. Further details regarding these adjustments can be found in GUPNet [8].

TABLE I
CAR RESULTS ON KITTI TESTING DATASET

Method	Car(IoU=0.7)			3D mAP
	Easy	Mod.	Hard	
MonoDLE [37]	17.23	12.26	10.29	13.29
MonoFlex [6]	19.94	13.89	12.07	15.30
GUPNet [8]	20.11	13.20	11.77	15.03
MonoCon [15]	22.50	16.46	13.95	17.64
DD3D [9]	23.19	16.87	14.36	18.14
MonoDTR [14]	21.99	15.39	12.73	16.70
MonoPGC [20]	24.68	17.17	14.14	18.66
MonoDETR [47]	25.00	16.47	13.58	18.35
GUPNet++ [21]	24.99	16.48	14.58	18.68
Ours	**25.43**	**17.89**	**15.01**	**19.44**

C. Experiment Results

1) KITTI Test Dataset: We perform experiments on the KITTI test dataset and present the comparison between our method and state-of-the-art approaches in Table I, with the best results highlighted in bold. The results demonstrate that our method surpasses others.

2) KITTI Validation Dataset: Experiments were conducted on the KITTI validation dataset, and the results are presented in Table II, with superior outcomes highlighted in bold. Our method demonstrates superior performance across most metrics, except for the Easy subset with 0.7 IoU threshold. However, on the test dataset, our model outperforms GUPNet++ across all metrics. This suggests that GUPNet++ may suffer from slight overfitting, resulting in a smaller performance gap between our method and theirs on the validation dataset.

3) Waymo Validation Dataset: Experiments are conducted on the Waymo validation dataset, and the comparison between our method and other approaches is presented in Table III, with the best results highlighted in bold. Since many other methods only conduct experiments on a subset of the Waymo Dataset, we adopt the same strategy for fair comparison. The results shows that our methods outperform other methods except CaDDN in 0-30m. We think that this is because CaDDN use the depth classification for depth estimation, and this type of

TABLE II
CAR RESULTS ON KITTI VALIDATION DATASET

Method	3D@IoU=0.7			BEV@IoU=0.7			3D@IoU=0.5			BEV@IoU=0.5		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
MonoDLE [37]	17.45	13.66	11.68	24.97	19.33	17.01	55.41	43.42	37.81	60.73	46.87	41.89
MonoFlex [6]	23.64	17.51	14.83	-	-	-	-	-	-	-	-	-
GUPNet [8]	22.76	16.46	13.72	31.07	22.94	19.75	57.62	42.33	37.59	61.78	47.06	40.88
MonoCon [15]	26.33	19.01	15.98	-	-	-	-	-	-	-	-	-
MonoDTR [14]	24.52	18.57	15.51	33.33	25.35	21.68	64.03	47.32	42.20	69.04	52.47	45.90
MonoPGC [20]	25.67	18.63	15.65	34.06	24.26	20.78	-	-	-	-	-	-
MonoDETR [47]	28.84	20.61	16.38	-	-	-	-	-	-	-	-	-
GUPNet++ [21]	**29.03**	20.45	17.89	38.82	27.95	24.96	66.66	49.65	45.23	71.55	54.00	49.34
Ours	28.93	**21.07**	**18.13**	**39.34**	**28.58**	**25.89**	**67.44**	**51.37**	**46.88**	**72.34**	**55.13**	**50.23**

TABLE III
RESULTS OF WAYMO VALIDATION DATASET

Methods	3D mAP/mAPH			
	Overall	0-30m	30-50m	50m-inf
Under Level 1(IoU=0.5)				
PatchNet [24]	2.92/2.74	10.01/9.75	1.09/0.96	0.23/0.18
CaDDN [7]	17.54/17.31	**45.00/44.46**	9.24/9.11	0.64/0.62
PCT [25]	4.20/4.15	14.70/14.54	1.78/1.75	0.39/0.39
MonoJSG [26]	5.65/5.47	20.86/20.26	3.91/3.79	0.97/0.92
SSD-MonoDETR [22]	11.83/-	27.69/-	5.33/-	0.85/-
DID-M3D [11]	20.66/20.47	40.92/40.60	15.63/15.48	5.35/5.24
Ours	**22.69/22.33**	43.27/42.89	**16.94/16.72**	**6.38/6.12**
Under Level 2(IoU=0.5)				
PatchNet [24]	2.42/2.28	10.01/9.73	1.07/0.94	0.22/0.16
CaDDN [7]	16.51/16.28	**44.87/44.33**	8.99/8.86	0.58/0.55
PCT [25]	4.03/3.99	14.67/14.51	1.74/1.71	0.36/0.35
MonoJSG [26]	5.34/5.17	20.79/20.19	3.79/3.67	0.85/0.82
SSD-MonoDETR [22]	11.34/-	27.62/-	5.21/-	0.76/-
DID-M3D [11]	19.37/19.19	40.77/40.46	15.18/15.04	4.69/4.59
Ours	**21.47/21.33**	42.39/42.14	**16.53/16.37**	**5.12/5.01**

TABLE IV
EFFECTS OF DIFFERENT COMPONENTS

No.	MS	GA	PPP	Car (IoU=0.7)			3D mAP
				Easy	Mod.	Hard	
(a)				26.46	18.43	15.32	20.07
(b)	✓			27.56	19.32	15.98	20.95
(c)		✓		27.23	19.01	16.47	20.90
(d)			✓	27.34	19.45	16.35	21.05
(e)	✓	✓		28.12	20.31	17.12	21.85
(f)		✓	✓	28.33	20.45	17.56	22.11
(g)	✓		✓	28.42	20.44	17.43	22.10
(h)	✓	✓	✓	**28.93**	**21.07**	**18.13**	**22.71**

method will may more attention to the near objects, and our methods treat the objects in different distances more balanced.

D. Ablation Studies

1) Influence of different components: We evaluated the MSGA and PPP modules as presented in Table IV. Our baseline model, without these modules, serves as a reference point. Results from experiments (b), (c) and (d) demonstrate that incorporating either module individually enhances performance. Moreover, for combinations involving both modules, such as (e), (f) and (g), those with PPP exhibit superior performance compared to those without. This suggests that

TABLE V
EFFECTS OF MULTI-SCALE METHODS

FT	CD	Scale	Car (IoU=0.7)			3D mAP
			Easy	Mod.	Hard	
5	5	-	28.67	20.88	**18.23**	22.59
5	10	-	28.93	**21.07**	18.13	**22.71**
5	15	-	28.88	20.78	17.69	22.45
10	5	-	28.91	20.83	17.54	22.43
10	10	-	**29.01**	20.75	17.49	22.42
10	15	-	28.74	20.69	16.58	20.00
15	5	-	28.63	20.83	16.79	22.08
15	10	-	28.66	20.66	16.54	21.95
15	15	-	28.74	20.88	16.32	21.98
-	-	(1.0, 1.05, 1.1)	25.42	17.56	14.49	19.16
-	-	(1.1, 1.2, 1.3)	24.95	16.69	13.78	18.47

TABLE VI
EFFECTS OF GRID ATTENTION ACTIVATION FUNCTION

Method	Car (IoU=0.7)			3D mAP
	Easy	Mod.	Hard	
Sigmoid	28.93	**21.07**	**18.13**	**22.71**
Hard Sigmoid	28.74	21.05	17.74	22.51
Tanh	**29.03**	20.94	17.99	22.65
Softmax	27.93	20.66	17.32	21.97

TABLE VII
EFFECTS OF PROBABILISTIC POST PROCESSING

Method	Car (IoU=0.7)			3D mAP
	Easy	Mod.	Hard	
δ=0.05	27.95	20.13	16.54	21.54
δ=0.1	**28.93**	**21.07**	**18.13**	**22.71**
δ=0.2	28.66	20.73	17.63	22.34
δ=0.3	28.65	20.76	17.79	22.40
δ=0.4	28.03	20.24	17.02	21.76

probabilistic information plays a crucial role in network learning, particularly for models with probabilistic assumptions. Experiment (h) indicates that the model incorporating all modules achieves the highest performance, highlighting the effectiveness of our proposed modules in directing attention to RoI grids and leveraging probabilistic cues more effectively.

2) Influence of Multi-Scale Methods: We tested the impact of enlarging the RoI on performance. Initially, we uniformly increased both the width and height by the same number of pixels, and also explored proportionally enlarging the RoI. Our findings revealed a significant performance drop when enlarging the bounding box by factors of 1.0, 1.05, and 1.1. Further proportional increases exacerbated this decline due to scale variance in the image plane. Conversely, when adopting a strategy of enlarging the RoI by a fixed number of pixels, we devised a scheme to generate additional pixels in a multi-scale fashion, forming an arithmetic series based on the First Term (FT) and the Common Difference (CD). The experiment is depicted in Table V. Increasing the FT and CD led to deteriorating performance in hard partitions. This is likely because objects in hard partitions typically have small 2D bounding boxes, causing background pixels to dominate the learning process when more pixels are added. Conversely, changes in FT and CD had less impact on easy and moderate partitions, consistent with our assumption that objects in these partitions have larger 2D bounding boxes, making minor pixel adjustments less influential. Proportionally enlarging the bounding box resulted in notably worse performance compared to our methods, attributed to scale variance in the image plane. Even in hard partitions where bounding boxes are small, performance was inferior to simply adding fixed pixels. This discrepancy may stem from the network's confusion regarding background and foreground pixels due to varying bounding box sizes and the corresponding distribution of background and foreground pixels within the enlarged bounding box. The prevalence of large objects outweighs that of small objects, indicating an imbalance in object sizes. Proportionally enlarging RoIs exacerbates this issue, resulting in a significant increase in background pixels. Consequently, the network might inadvertently learn the relative proportion of background to foreground pixels. This can lead to confusion during the processing of small objects, hindering the model's learning process.

3) Influence of Grid Attention Activation Function: To scale the output of the attention map to the range $[0, 1]$, we evaluate several activation functions for comparison, as detailed in Table VI. Sigmoid, Hard Sigmoid [44], and Softmax are directly applied to the attention map. However, since the Tanh function maps values to the range $[-1, 1]$, we employ the transformation outlined in Eq. (18) to map these values to $[0, 1]$.

$$att = (1 + tanh(x))/2 \tag{18}$$

where att is the attention map, x is the output of the network.

The experimental results are presented in Table VI. Sigmoid, hard sigmoid, and tanh functions exhibit comparable performance, while the softmax function performs notably worse. This discrepancy can be attributed to the softmax function's tendency to assign high attention to one grid and low attention to others. However, since the RoI is not determined solely by one grid, this approach proves less effective.

4) Influence of Probabilistic Posting Processing: We selected the hyper-parameters δ in the Probabilistic Post-Processing (PPP) module to optimize its performance. If δ is too small, the model tends to focus solely on predictions with the lowest uncertainty. Conversely, if δ is too large, the differences between positions become less discernible. The experimental findings are detailed in Table VII. Setting δ to values such as 0.05 or 0.4 leads to a drop in performance. Consequently, we determined that $\delta = 0.1$ yields the best results based on our experiments.

V. CONCLUSION

In this paper, we delve into the challenges of RoI-based monocular 3D object detection. We identify two main issues: errors in 2D bounding box prediction and overlooked variations in RoI grid weights, hindering effective network learning. To address these issues, we introduce the MSGA module, which allocates grid attention intelligently and integrates multi-scale features to mitigate prediction errors. Additionally, acknowledging the probabilistic assumptions prevalent in existing methods, we propose the PPP module to better leverage probabilistic cues. Both modules significantly enhance baseline performance. However, our method employs an independent depth completion network to generate ground truth visual depth, which, although not entirely accurate, may introduce noise into the training data. Hence, denoising is a future avenue of research to enhance the robustness of our network.

VI. ACKNOWLEDGEMENTS

This study is supported in part by the National Natural Science Foundation of China (62272194, 62172186), and in part by the Science and Technology Development Plan Project of Jilin Province (20220101101JC).

REFERENCES

[1] A. Mousavian, D. Anguelov, J. Flynn and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, 2017, pp. 7074-7082.

[2] F. Manhardt, W. Kehl and A. Gaidon, "Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA. 2019, pp. 2069-2078.

[3] Y. Chen, L. Tai, K. Sun and M. Li, "Monopair: Monocular 3d object detection using pairwise spatial relationships," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, USA, 2020, pp. 12093-12102.

[4] Z. Liu, Z. Wu and R. Tóth, "Smoke: Single-stage monocular 3d object detection via keypoint estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, USA, 2020, pp. 996-997.

[5] Y. Cai, B. Li, Z. Jiao, H. Li, X. Zeng and X. Wang, "Monocular 3d object detection with decoupled structured polygon estimation and height-guided depth estimation," in *Proc. AAAI Conf. Artif. Intell.*, New York, USA, 2020, pp. 10478-10485.

[6] Y. Zhang, J. Lu and J. Zhou, "Objects are different: Flexible monocular 3d object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Nashville, TN, USA, 2021, pp. 3289-3298.

[7] C. Reading, A. Harakeh, J. Chae and SL. Waslander, "Categorical depth distribution network for monocular 3d object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Nashville, TN, USA, 2021, pp. 8555-8564.

[8] Y. Lu et al., "Geometry uncertainty projection network for monocular 3d object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Montreal, Canada. 2021, pp. 3111-3121.

[9] D. Park, R. Ambrus, V. Guizilini, J. Li and A. Gaidon, "Is pseudo-lidar needed for monocular 3d object detection?," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Montreal, Canada. 2021, pp. 3142-3152.

[10] Z. Li, Z. Qu, Y. Zhou, J. Liu, H. Wang and L. Jiang, "Diversity matters: Fully exploiting depth clues for reliable monocular 3d object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, New Orleans, LA, USA, 2022, pp. 2791-2800.

[11] L. Peng, X. Wu, Z. Yang, H. Liu and D. Cai, "Did-m3d: Decoupling instance depth for monocular 3d object detection," in *Eur. Conf. Comput. Vis.*, Tel-Aviv, Israel, 2022, pp. 71–88.

[12] D. Rukhovich, A. Vorontsova and A. Konushin, "Imvoxelnet: Image to voxels projection for monocular and multi-view general-purpose 3d object detection," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, Waikoloa, HI, USA. 2022, pp. 2397-2406.

[13] T. Wang, ZHU. Xinge, J. Pang and D. Lin, "Probabilistic and geometric depth: Detecting objects in perspective," in *Conf. on Robot. Learn.*, London, UK, 2022, pp. 1475-1485.

[14] KC. Huang, TH. Wu, HT. Su and WH. Hsu, "Monodtr: Monocular 3d object detection with depth-aware transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, New Orleans, LA, USA, 2022, pp. 4012-4021.

[15] X. Liu, N. Xue and T. Wu, "Learning auxiliary monocular contexts helps monocular 3d object detection," in *Proc. AAAI Conf. Artif. Intell.*, Vol. 36, No. 2, Virtual, 2022, pp. 1810-1818.

[16] Q. Ye, L. Jiang, W. Zhen and Y. Du, "Consistency of implicit and explicit features matters for monocular 3d object detection," 2022, *arXiv:2207.07933*.

[17] X. Liu, C. Zheng, KB. Cheng, N. Xue, GJ. Qi and T. Wu, "Monocular 3d object detection with bounding box denoising in 3d by perceiver," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Paris, France. 2023, pp. 6436-6446.

[18] Z. Min, B. Zhuang, S. Schulter, B. Liu, E. Dunn and M. Chandraker, "Neurocs: Neural nocs supervision for monocular 3d object localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Vancouver, Canada. 2023, pp. 21404-21414.

[19] J. Xu et al., "Mononerd: Nerf-like representations for monocular 3d object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Paris, France. 2023, pp. 6814-6824.

[20] Z. Wu, Y. Gan, L. Wang, G. Chen and J. Pu, "Monopgc: Monocular 3d object detection with pixel geometry contexts," in *Int. Conf. Robot. Automat.*, London, UK. 2023, pp. 4842-4849.

[21] Y.Lu et al., "GUPNet++: Geometry Uncertainty Propagation Network for Monocular 3D Object Detection," in *Int. Conf. Robot. Automat.*, 2023, *arXiv:2310.15624*.

[22] X. He et al., "Ssd-monodetr: Supervised scale-aware deformable transformer for monocular 3d object detection," in *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, pp. 555-567, Ian. 2024, doi: 10.1109/TIV.2023.3311949.

[23] K. He, G. Gkioxari, P. Dollar and R. Girshick, "Mask r-cnn," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Venice, Italy. 2017, pp. 2961-2969.

[24] X. Ma, S. Liu, Z. Xia, H. Zhang and X. Zeng, "Rethinking pseudo-lidar representation," in *Eur. Conf. Comput. Vis.*, Virtual, 2020, pp. 311–327.

[25] L. Wang et al., "Progressive coordinate transforms for monocular 3d object detection," in *Adv. Neural Inf. Process. Syst.*, Vol. 34, pp. 13364–13377, 2021.

[26] Q. Lian, P. Li and X. Chen, "Monojsg: Joint semantic and geometric cost volume for monocular 3d object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, New Orleans, LA, USA, 2022, pp. 1070-1079.

[27] R. Girshick, "Fast r-cnn," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Santiago, Chile. 2015, pp. 1440-1448.

[28] J. Dai, K. He and J. Sun, "Instance-aware semantic segmentation via multi-task network cascades," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, 2016, pp. 3150-3158.

[29] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Providence, RI, 2012, pp. 3354–3361.

[30] G. Brazil and X. Liu, "M3d-rpn: Monocular 3d region proposal network for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Seoul, Korea. 2019, pp. 9287-9296.

[31] M. Ding, et.al., "Learning depth-guided convolutions for monocular 3d object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, USA, 2020, pp. 1000-1001.

[32] T. Roddick, A. Kendall and R. Cipolla, "Orthographic feature transform for monocular 3d object detection," 2018, *arXiv:1811.08188*.

[33] X. Ma, Z. Wang, H. Li, P. Zhang, W. Ouyang and X, Fan, "Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Seoul, Korea. 2019, pp. 6851-6860.

[34] X. Weng, K. Kitani, "Monocular 3d object detection with pseudo-lidar point cloud," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop*, Seoul, Korea. 2019.

[35] P. Li, H. Zhao, P. Liu and F. Cao, "Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving," in *Eur. Conf. Comput. Vis.*, Virtual, 2020, pp. 644-660.

[36] B. Mildenhall, PP. Srinivasan, M. Tancik, JT. Barron, R. Ramamoorthi and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *Commun. ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[37] X. Ma et.al., "Delving into localization errors for monocular 3d object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Nashville, TN, USA, 2021, pp. 4721-4730.

[38] S. Luo, H. Dai, L. Shao and Y. Ding, "M3dssd: Monocular 3d single stage object detector," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Nashville, TN, USA, 2021, pp. 6145-6154.

[39] L. Wang et.al., "Depth-conditioned dynamic message propagation for monocular 3d object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Nashville, TN, USA, 2021, pp. 454-463.

[40] F. Yu, D. Wang, E. Shelhamer, T. Darrell, "Deep layer aggregation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, 2018, pp. 2403-2412.

[41] X. Zhou, D. Wang, P. Krähenbühl, "Objects as points," 2019, *arXiv:1904.07850*.

[42] J. Mao, M. Niu, H. Bai, X. Liang, H. Xu, C. Xu, "Pyramid r-cnn: Towards better performance and adaptability for 3d object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Montreal, Canada. 2021, pp. 2723-2732.

[43] AL. Maas, AY. Hannun, AY. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, Atlanta, USA. 2013.

[44] M. Courbariaux, Y. Bengio, JP. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3123-3131.

[45] Y. Wang, B. Li, G. Zhang, Q. Liu, T. Gao, Y. Dai, "Lrru: Long-short range recurrent updating networks for depth completion," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Paris, France. 2023, pp. 9422-9432.

[46] P. Sun et al., "Scalability in perception for autonomous driving: Waymo open dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Seattle, WA, USA, 2020, pp. 2446-2454.

[47] R. Zhang et al., "MonoDETR: Depth-guided transformer for monocular 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Paris, France. 2023, pp. 9155-9166.

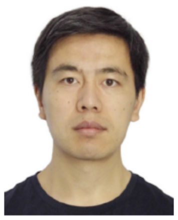
[48] R. P. Brent, "Algorithms for Minimization Without Derivatives." , Upper Saddle River, NJ, USA: Prentice-Hall, 1973.

Tingyu Zhang Tingyu Zhang received the B.Sc. degree in Mathematics and Statistics from the Nanjing University of Information Science and Technology, Nanjing, China, in 2019. He is currently pursuing the Ph.D. degree in computer science and technology with Jilin University, Changchun. His current research interests include intelligent vehicles, point cloud analysis and 3D object detection.

Xinyu Yang Xinyu Yang received her master's degree in computer technique from Jilin University of science and technology. Her research interests include vehicular networks and intelligent driving, especially for privacy protection. She currently works for China Automotive Innovation Corporation.

Zhigang Liang Zhigang Liang graduated from Jilin University with a bachelor's degree in Engineering mechanics in 2019. He is currently pursuing a PhD in Computer Science and Technology at Jilin University in Changchun. His current research interests include self-driving car digital twin testing and intelligent transportation.

Yanzhao Yang Yanzhao Yang received his master's degree from Jilin University in 2011, major in Computer Science and Technology. He is currently pursuing the Ph.D degree in computer science and Technology with Jilin University, Changchun. His current research interests include intelligent vehicles, Physical Model of ADAS sensor and machine learning.



Jian Wang Jian Wang received the B.Sc., M.Sc., and Ph.D. degrees in computer science from Jilin University, Changchun, China, in 2004, 2007, and 2011, respectively. He is currently a Professor with the College of Computer Science and Technology, Jilin University. He has published over 60 articles in international journals. His research interests include wireless communication and vehicular networks, especially for network security and privacy protection.