MonoEdge: Monocular 3D Object Detection Using Local Perspectives

Minghan Zhu[1*], Lingting Ge[2], Panqu Wang[2], Huei Peng[1]
[1]University of Michigan
[2]TuSimple Inc
minghanz@umich.edu, lingting.ge@tusimple.ai, panqu.wang@tusimple.ai, hpeng@umich.edu

Abstract

We propose a novel approach for monocular 3D object detection by leveraging local perspective effects of each object. While the global perspective effect shown as size and position variations has been exploited for monocular 3D detection extensively, the local perspectives has long been overlooked. We design a local perspective module to regress a newly defined variable named keyedge-ratios as the parameterization of the local shape distortion to account for the local perspective, and derive the object depth and yaw angle from it. Theoretically, this module does not rely on the pixel-wise size or position in the image of the objects, therefore independent of the camera intrinsic parameters. By plugging this module in existing monocular 3D object detection frameworks, we incorporate the local perspective distortion with global perspective effect for monocular 3D reasoning, and we demonstrate the effectiveness and superior performance over strong baseline methods in multiple datasets.

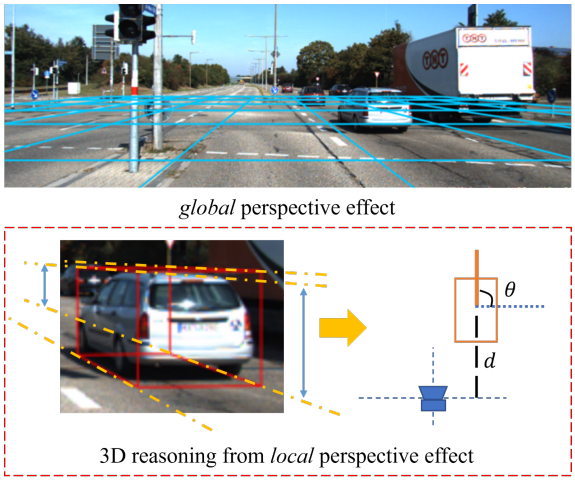

Figure 1. Up: the *global* perspective effect of appearance and size change along the depth direction of an image. Down: the *local* perspective effect is embedded in the shape distortion *inside* an object, as highlighted by the dashed lines. We propose to infer the depth d and yaw angle θ from the local perspectives.

1. Introduction

3D object detection is an important perception task for autonomous driving and other robotic applications. Monocular approaches are considered challenging mainly due to the lost of depth information in a single image. However, monocular 3D object detection still attracts a lot of interest, partly because of the low cost compared with LiDAR and the simple sensor setup compared with stereo cameras. Despite the absence of depth in a single image, monocular 3D inference is possible in practice through learning, due to the implicit prior knowledge of the physical scene layout and size of objects embedded in the data. Given such prior knowledge, the 3D localization of objects is connected with their size and position in the image plane, through the camera geometry.

Position and size are two major clues leveraged by existing monocular 3D object detection methods, and they are the result of global perspective effects on the whole image. For a camera mounted on a vehicle with viewing angle similar to the sight of human and common scenes where most objects are on the approximately flat ground, closer objects appear near the bottom of image, and the position of further objects are higher near the horizon. It is used as a prior knowledge for depth estimation in several works [18, 1]. In addition, objects appears larger when they are close and smaller otherwise. This relation is also often applied in the inference of depth [33, 34]. Meanwhile, some methods rely on pixel-wise depth image as an input or middle output [23]. As discussed in [6], a dense depth prediction network heavily relies on the pixel position in the image for depth prediction as well.

In this work, we propose to explore another type of clues for 3D reasoning: *local* perspectives. Here we aim to exploit the local distortion *inside* an object, rather than among different objects. In other words, the closer part of an object

*Work done during internship at TuSimple Inc.

appears larger than the remote part, and the magnitude of such distortion is directly connected with its distance. See Fig. 1 for a comparison between global perspectives and local perspectives. Capturing the perspective distortion of an object is a common skill in drawing, and it largely affects how real a painting looks, as it is tightly connected with the 3D perception of human beings. Inspired by this observation, we want to exploit this factor in improving the 3D inference of monocular computer vision.

To achieve this goal, we first propose a method to parameterize the magnitude of local perspective distortion, by defining the *keyedges* and the *keyedge-ratios*. Then we show that both the depth and yaw angle of an object can be determined given the keyedge-ratios and the physical size. Our orientation estimation is also novel in that the global yaw angle is directly obtained, rather than following most existing approaches that estimate the allocentric observation angle first [24]. Our approach, named MonoEdge, can be formed as a lightweighted module in the neural networks, and can be plugged in to existing monocular 3D detection frameworks. Our experiments demonstrate effectiveness and superior performance over strong baseline methods on the KITTI 3D detection benchmark [9], and the improvement is also validated on the nuScenes dataset [2].

In summary, our work has the following contributions:

- We propose to leverage local perspective distortion for monocular 3D object detection.
- Through our approach, the depth and global yaw angle can be estimated from the local appearance of an object in an image, without knowing the camera intrinsic parameters.
- We show the general applicability of our local-perspective-based method by incorporating it with strong baseline methods on the KITTI dataset [9] and on the nuScenes dataset [2], and improving them respectively.

2. Related work

2.1. Object and pixel depth estimation

As mentioned in several previous work [22, 32], inaccurate depth is a dominant single factor limiting the accuracy of monocular 3D object detection. Much research effort are dedicated to improving the object based depth estimation. A simplest strategy is to directly regress the depth of objects[19, 31], and it actually works quite well in benchmarks like KITTI [9] and nuScenes [2]. However, the directly regressed depth is generally overfitted to the camera intrinsic and extrinsic parameters of the training data, and performs poorly when the camera parameters are changed. [35] partially addressed this issue by estimating and compensating the extrinsic perturbation. Pixel-wise depth regression is a similar task, which is also used as an input or middle-output in some monocular 3D detection work [23, 7]. It suffers from similar difficulty in generalizing to different camera parameters [8]. As revealed in [6], the depth regression relies heavily on the height of the pixel in the image, implying the dependency on pixel positions and camera parameters.

The qualitative relation that pixels higher and nearer to the horizon are generally further away can serve as a prior for depth estimation. [1, 18] incorporated such prior into the network design by treating each row of the feature map differently in the convolution. Visual size is another important clue for 3D inference, and [33, 34] estimate the height of objects and then calculate the depth from it. [27, 21] focused on the uncertainty analysis of depth estimated from the appearance and physical height of objects. Overall, the position and size priors are governed by the global perspective effect on the whole image. [32, 5] model the global perspective relation among objects informed by their positions in the image. [17] explored data augmentation with size-position relation that is compliant to the global perspectives.

2.2. Object orientation estimation

Orientation estimation is another important topic for monocular 3D object detection. Deep3DBox [24] shows that the visible part of an object in an image is mainly determined by the local observation angle (also called the allocentric angle), instead of the global yaw angle (also called the egocentric angle, see Sec. 3.1). Since the visible part largely determines the appearance of an object in an image, they first regress the allocentric angle from the network, and then convert it to the egocentric yaw angle with the pixel position and camera intrinsics. This strategy is widely adapted in later works [12, 23]. Recently, Ego-Net [16] directly regresses the egocentric yaw angle from a network, which proposed a progressive lifting strategy with intermediate geometrical representations.

2.3. Perspective effect in 3D object detection

Some previous work contains the idea of using local shape distortion in 3D inference. GS3D [13] extracts features from the visible faces of a coarse 3D bounding box through perspective transform. KM3D-Net [14] designs a Vanishing-Point-to-Orientation network module to regress orientation from the projected 3D box corner points in the image plane. Some previous work follows a render-and-compare strategy [12, 11, 3, 20], which optimize the 3D properties by aligning the rendered shape model with the 2D observation. However, they do not explicitly parameterize the local perspective distortions or derive 3D properties from this information, as we will show in this paper.

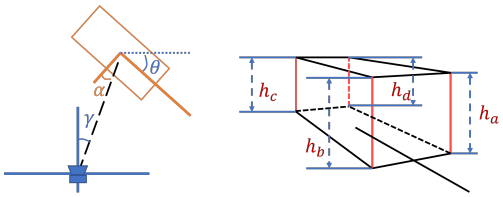

Figure 2. Left: the geometry and angle definition in 3D detection. All the angles denoted are shown in their positive direction according to KITTI's definition. Right: The projection of a 3D bounding box in an image, and the *keyedges* are highlighted in red.

3. Methodology

We first introduce the basic geometry and the definitions of variables in our model (Sec. 3.1). We then derive the depth and global yaw angle using local perspectives (Sec. 3.2). The network design for keyedge-ratio regression is presented in Sec. 3.3.

3.1. Preliminaries

First we explain the definition of parameters used in our model. As shown on the left of Fig. 2, the global yaw angle (or the *egocentric angle*) of an object is the angle between the camera's right axis and the object's front axis, denoted as θ. The local observation angle (or the *allocentric angle*) is the angle of the camera's position w.r.t. the object's right axis, denoted as α. The angle of the object's position w.r.t. the camera's front axis is also shown as γ, and we call it the *viewing angle*. They satisfy the equation $\theta = \alpha + \gamma$. Among these angles, γ could be calculated from the pixel's horizontal position as long as the camera intrinsic parameters are given.

The local perspective effect is shown as distortion inside an object where the further part appears smaller. Since most objects are aligned to the vertical axis (zero pitch and roll), the size distortion can be measured conveniently through the height change. Therefore, we define the four vertical edges in a 3D bounding box as *keyedges*, and use the ratio between the visual height of the keyedges to parameterize the local perspective distortion. We call them *keyedge-ratios*:

$$r_{ij} \triangleq \frac{h_i}{h_j} \tag{1}$$

where h_i and h_j are the visual height of the keyedges i and j. The keyedges are indexed in an object-centric way: indices a, b, c, d are assigned clockwisely starting from the front-left corner of the object, as shown on the right of Fig. 2.

3.2. Intrinsics-free depth and yaw angle derivation

In this part we show that the depth and global yaw angle can be calculated from the keyedge-ratios and the physical size of the object. We show the math in the minimal setup, where two keyedge-ratios r_{ba} and r_{bc} are given, and the real length, width, and height of the object are denoted l, w, h, respectively. We also denote the focal length as f (see Fig. 3).

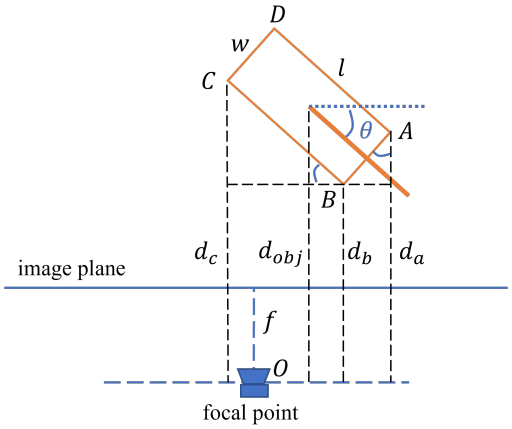

Figure 3. The geometry for calculating depth d_{obj} and yaw angle θ from keyedge-ratios, shown in top-down view. The three angles denoted with blue arcs are identical.

For any keyedge i,

$$\frac{f}{d_i} = \frac{h_i}{h} \Rightarrow d_a h_a = d_b h_b = d_c h_c = fh \tag{2}$$

Denote the physical depth of the three corners A, B, C (corresponding to the keyedges a, b, c) as d_a, d_b, d_c, we also have:

$$d_a = d_b + w\cos\theta, d_c = d_b + l\sin\theta \tag{3}$$

$$\Rightarrow r_{ba} = \frac{h_b}{h_a} = \frac{d_a}{d_b} = 1 + \frac{w\cos\theta}{d_b}, r_{bc} = 1 + \frac{l\sin\theta}{d_b} \tag{4}$$

From Eq. (4), we get d_b by cancelling θ and get θ by cancelling d_b:

$$\Rightarrow d_b = \frac{1}{\sqrt{\frac{(r_{ba}-1)^2}{w^2} + \frac{(r_{bc}-1)^2}{l^2}}} \tag{5}$$

$$\theta = \arctan2\ (w(r_{bc}-1), l(r_{ba}-1)) \tag{6}$$

Denote the depth of the object center as d_{obj}, we have

$$d_{obj} = d_b + \frac{1}{2}(l\sin\theta + w\cos\theta) \tag{7}$$

In this way, the object depth d_{obj} and yaw angle θ are obtained. Notice that the focal length is eliminated, and the final results only depend on r_{ba}, r_{bc}, l, and w. Therefore this method is *camera-intrinsics-free*. For other keyedges, the angles and signs in the equations will change accordingly, which we show in more detail in the appendix.

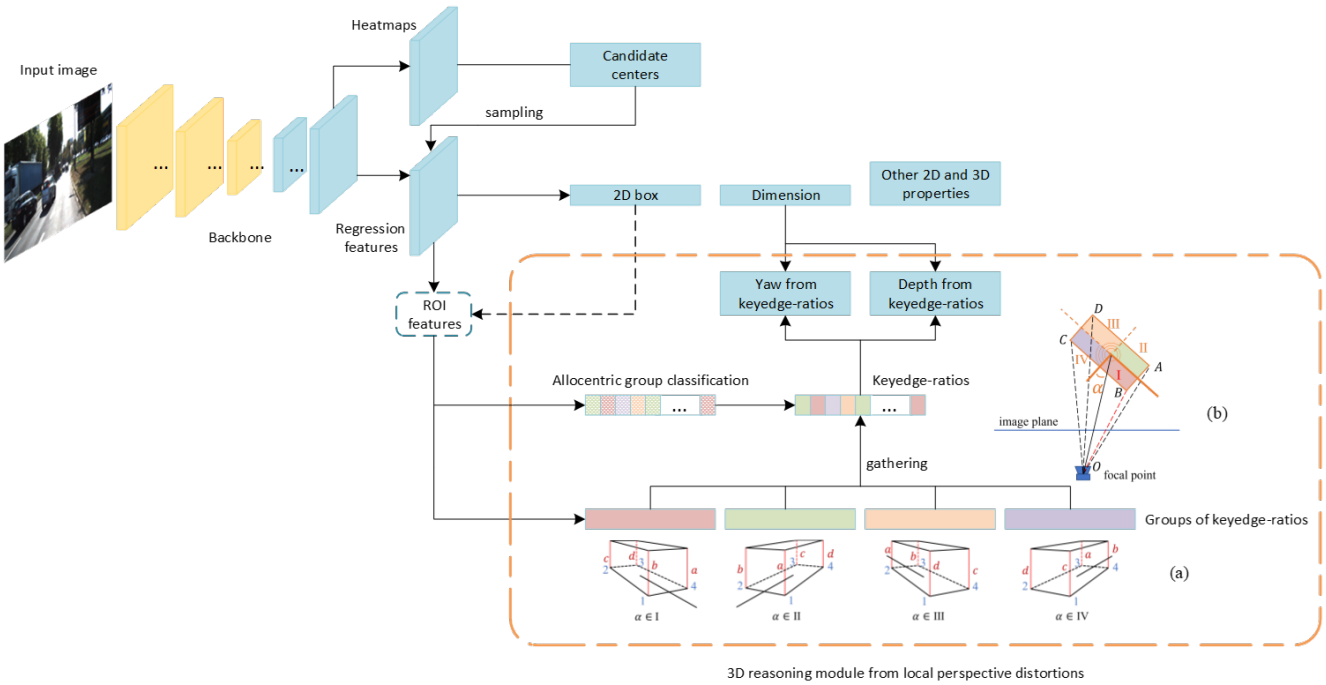

Figure 4. The overall structure of our network design. The orange dashed box highlights our local-perspective-based 3D inference module. Outside of the orange box is a typical one-stage or two-stage CNN architecture for Monocular 3D estimation, and we show it for the sake of completeness. The subfigure (a) illustrates the object-centric keyedge indices in red letters and the camera-centric keyedge indices in blue digits. The allocentric groups are visualized in subfigure (b). The dashed arrow and ROI-features are for two-stage networks only.

3.3. Network design

Overall, there are two main components in our module: keyedge-ratio regression with allocentric group classification and uncertainty-based depth fusion, as explained below.

3.3.1 Keyedge-ratio regression

The 3D inference based on local perspectives is independent of the absolute height of keyedges in an image. Therefore, we design a specific module to directly estimate keyedge-ratios.

First, we introduce a *camera-centric indexing* strategy to regress keyedge-raitos. This strategy is illustrated in Fig. 4 (a) as the blue numeric indices, in which index 1 is always assigned to the keyedge with shortest distance to the camera center. By using camera-centric indexing, all four keyedge-ratios $[r_{21}, r_{41}, r_{32}, r_{34}]$ are equal or smaller than 1 regardless of rotations (in most cases, see the appendix for further discussion), thus facilitating learning and convergence. The conversion between the camera-centric indexing $\{1, 2, 3, 4\}$ and the object-centric indexing $\{a, b, c, d\}$ is through the *allocentric group*, which is defined by the quarter that the allocentric angle α falls in, as visualized in Fig. 4 (b). Each allocentric group corresponds to a conversion between the camera-centric indices and object-centric indices as shown in Fig. 4 (b).

Second, we propose to use *grouped heads* for the keyedge-ratio regression. In other words, different channels are used to regress keyedge-ratios for each allocentric group. In the network, there are 16 channels in total for the regression of 4 keyedge-ratios of all possible 4 allocentric groups. For each object, only the 4 channels corresponding to its allocentric group are used. The motivation is that the appearance of an object is very different when they are presented in different allocentric groups, and we decouple the learning for each of them to generate more discriminative features. As the keyedge-ratio regression relies on the allocentric group of an object, we also add the the 4-way allocentric group classification task to the network, supervised with cross-entropy loss.

3.3.2 Uncertainty-based depth fusion

After the conversion from camera-centric indexing to object-centric indexing, the keyedge-ratios are reorganized to 4 tuples: $(r_{ad}, r_{ab}), (r_{ba}, r_{bc}), (r_{cb}, r_{cd}), (r_{dc}, r_{da})$. Each tuple generates an estimation of depth and yaw angle as shown in Sec. 3.2, and we design a dedicated uncertainty based fusion module to generate the final result.

Specifically, each keyedge-ratio prediction head r_{ij} is accompanied with a regressed uncertainty estimation value

Methods	Val, $AP_{3D}, IoU \geq 0.7$			Val, $AP_{BEV}, IoU \geq 0.7$			Test, $AP_{3D}, IoU \geq 0.7$			Test, $AP_{BEV}, IoU \geq 0.7$		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
MonoDIS [28]	11.06	7.60	6.37	18.45	12.58	10.66	10.37	7.94	6.40	17.23	13.19	11.12
M3D-RPN [1]	14.53	11.07	8.65	20.85	15.62	11.88	14.76	9.71	7.42	21.02	13.67	10.23
SMOKE [19]	-	-	-	-	-	-	14.03	9.76	7.84	20.83	14.49	12.75
MonoPair [5]	16.28	12.30	10.42	24.12	18.17	15.76	13.04	9.99	8.65	19.28	14.83	12.89
RTM3D [15]	-	-	-	-	-	-	14.41	10.34	8.77	19.17	14.20	11.99
Movi3D [29]	14.28	11.13	9.68	22.36	17.87	15.73	15.19	10.90	9.26	22.76	17.03	14.85
D4LCN [7]	22.32	16.20	12.30	31.53	22.58	17.87	16.65	11.72	9.51	22.51	16.02	12.55
PGD [32]	<u>24.35</u>	<u>18.34</u>	**16.90**	30.56	<u>23.67</u>	<u>20.84</u>	19.05	11.76	9.39	26.89	16.51	13.49
DDMP-3D [30]	-	-		-	-	-	19.71	12.78	9.80	28.08	17.89	13.44
Ground-Aware [18]	23.63	16.16	12.06	-	-	-	<u>21.65</u>	13.25	9.91	<u>29.81</u>	17.98	13.08
CaDDN [26]	23.57	16.31	13.84	-	-	-	19.17	13.41	11.46	27.94	18.91	17.19
MonoGeo [34]	18.45	14.48	12.87	27.15	21.17	18.35	18.85	13.81	11.52	25.86	18.99	16.19
MonoEF [35]	-	-	-	-	-	-	21.29	13.87	11.71	29.03	19.70	<u>17.26</u>
GUP Net [21]	22.76	16.46	13.72	31.07	22.94	19.75	20.11	14.20	11.77	-	-	-
AutoShape [20]	20.09	14.65	12.07	-	-	-	**22.47**	14.17	11.36	**30.66**	<u>20.08</u>	15.59
MonoRCNN [27]	16.61	13.19	10.65	25.29	19.22	15.30	18.36	12.65	10.03	25.48	18.11	14.10
Ours (based on MonoRCNN)	18.44	14.60	12.57	26.19	20.67	17.30	19.74	<u>14.35</u>	11.94	27.52	20.07	16.34
Improvement	*+1.83*	*+1.41*	*+1.92*	*+0.90*	*+1.45*	*+2.00*	*+1.38*	*+1.70*	*+1.91*	*+2.04*	*+1.96*	*+2.24*
MonoFlex [33]	23.64	17.51	14.83	<u>31.65</u>	23.29	20.02	19.94	13.89	<u>12.07</u>	28.23	19.75	16.89
Ours (based on MonoFlex)	**25.66**	**18.89**	<u>16.10</u>	**33.71**	**25.35**	**22.18**	21.08	**14.47**	**12.73**	28.80	**20.35**	**17.57**
Improvement	*+2.02*	*+1.38*	*+1.27*	*+2.06*	*+2.06*	*+2.16*	*+1.14*	*+0.58*	*+0.66*	*+0.57*	*+0.60*	*+0.68*

Table 1. Quantitative result on the KITTI validation set and test set. The best are highlighted in **bold** font. The second best are <u>underlined</u>. The improvement over baselines are in *italic* font.

σ_{ij}. Denote the depth estimated from keyedge ratios as d_{kr}, we approximate the uncertainty of d_{kr} using the first order derivative:

$$\sigma_{d_{kr}} = \sum_{(i,j)} \frac{\partial d_{kr}}{\partial r_{ij}} \sigma_{ij} \tag{8}$$

The partial derivatives $\frac{\partial d_{kr}}{\partial r_{ij}}$ are calculated using PyTorch's automatic differentiation engine [25].

The depth estimated from the keyedge-ratio tuples are averaged using the inverse uncertainty as the weight, $w_{d_t} = 1/\sigma_{d_t}, d_{fusion} = \sum_i d_t w_{d_t}$, where t indexes each depth prediction. The uncertainty-based weighted sum is also used when combining this method with other depth predictions in an existing network, as in Sec. 4. The keyedge-ratios and their uncertainties are together supervised by an uncertainty-aware loss function [10]:

$$L(r_{ij}, \sigma_{ij}; r^*_{ij}) = \frac{|r_{ij} - r^*_{ij}|}{\sigma_{ij}} + \log \sigma_{ij} \tag{9}$$

where r^*_{ij} is the ground truth keyedge ratio.

3.3.3 Components outside of the local perspective module

Aside from the proposed local perspective module estimating the keyedge-ratios as introduced above, there are other variables to be estimated to complete the monocular 3D object detection task. As our local perspective module can be plugged with various network structures, the list of variables estimated outside of the module slightly varies depending on the overall network structure used. Specifically, we incorporate our local perspective module with three networks: MonoFlex [33], MonoRCNN [27], and PGD [32], as they are representative one-stage and two-stage networks on KITTI [9] and nuScenes [2] benchmarks.

Commonly estimated variables across the networks include object classification score, 2D projected center, and physical size of 3D bounding box. The estimated physical length l and width w are used in Eqs. (5) to (7). The 2D bounding box and keypoints (defined by the 2D projection of 3D bounding box) are regressed in MonoFlex [33] and MonoRCNN [27], but are optional in PGD [32]. The list of variables and their corresponding regression heads and loss functions follow the original networks.

Fig. 4 is an overall illustration of the common architecture of the networks we incorporated with in this work. The dashed orange rectangle is the proposed local-perspective module, and outside of it are the backbone and regression heads of the plugged-in networks. The dashed arrow and the ROI-feature block are for MonoRCNN [27] as a two-stage network in which the regression heads are after extracting ROI features, while the one-stage networks (MonoFlex [33] and PGD [32]) follow the solid arrows.

4. Experiments

The experiments are conducted on the KITTI dataset [9] and nuScenes dataset[2]. KITTI has 7,481 images for training and 7,518 images for testing. We further split the official training set into training and validation set following [4] for fair comparison with other baselines. The dataset evaluates three object classes: Car, Pedestrian, and Cyclist, and we

mainly focus on the Car category when reporting the results following previous works. The evaluation metric is Average Precision of the 3D bounding boxes (AP_{3D}) and of the bird's eye-view 2D rotated bounding boxes (AP_{BEV}). We use 40 recall positions, which is the more meaningful metric compared with the 11-recall-position version, according to [28]. We also extend our experiments to the nuScenes dataset [2] to show that our method works under various environments.

4.1. Experiments on KITTI dataset

4.1.1 Implementation details

For the experiment on KITTI dataset, we choose two baselines, MonoFlex [33] and MonoRCNN [27], and implement our method based on these networks. They are representative one-stage and two-stage networks for monocular 3D object detection respectively. We fuse the local-perspective-(LP-) based estimation with the baselines' original estimation to show that our method brings value beyond existing approaches and can be incorporated to improve them. Both baselines have uncertainty estimation for their output, facilitating the uncertainty-based fusion as explained in Sec. 3.3.2. The training setups (batch size, training epochs, optimizer settings, data augmentations) are consistent with the baselines.

4.1.2 Quantitative and qualitative results

The quantitative results are shown in Tab. 1. Our method achieves consistent improvements on all evaluated metrics in the validation set and the test set over both baselines. The number of parameters in the network without and with our LP module is 21.47M v.s. 22.07M (for MonoFlex) and 69.71M v.s. 70.80M (for MonoRCNN), accounting for only 2.80% and 1.56% increase respectively. Therefore we account the improvement to the proposed methodology instead of the added parameters. The inference time overhead of our LP module is also limited, accounting for less than 10% increase (0.034s v.s. 0.037s for MonoFlex and 0.06s v.s. 0.065s for MonoRCNN). Qualitative examples of our results (with MonoFlex baseline) are shown in Fig. 7. Our method delivers more accurate localization performance.

4.1.3 Effect of local perspectives in estimation

In Tab. 2 we further decouple the effect of local perspective estimation. As shown in the second row, using keyedge-ratio regression alone for 3D detection yields suboptimal results. It is not surprising since the local perspective distortion of objects are more subtle to estimate than their size and position in an image. It is not a problem since we do not need to discard the stronger signal and restrict ourselves to local perspectives alone in practice. Nonetheless, the local-perspective-based approach shows its value when combined with the existing approaches. The universal improvements in all metrics for both baselines indicate that the local perspective distortion is a missing piece in previous work, and we may incorporate it with various existing approaches for general improvements.

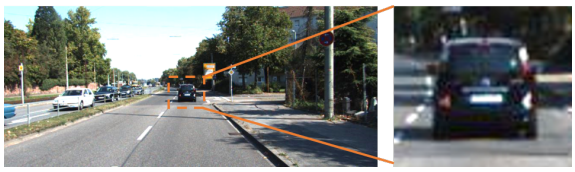

Figure 5. Example of a vehicle in the up front showing little local perspective distortion. In Sec. 4.1.5 we show that the performance of LP module does not deteriorate on such objects.

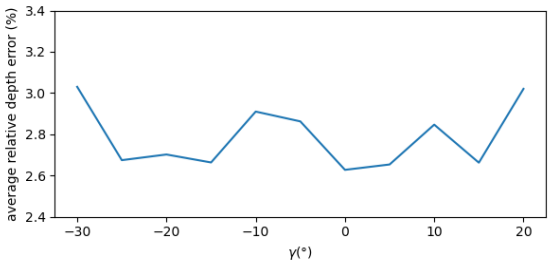

Figure 6. Analysis of the error of local perspective estimation for objects at different viewpoints, evaluated on car category of KITTI validation set.

4.1.4 Ablation study on Keyedge-ratio regression

Here we validate that the specific design of the keyedge-ratio regression head is beneficial. We experimented on the MonoFlex-based network, removing the grouped heads by using the same four channels to regress the four keyedge-ratios regardless of the allocentric group. We also experimented using the object-centric indexing instead of the camera-centric indexing.

The experimental results are shown in Tab. 3. The removal of either the grouped heads or the camera-centric indexing causes severe deterioration (up to 3 points), and the results are the worst when both are absent. It demonstrates that the proposed regression head design is crucial to the accurate estimation of keyedge-ratios.

4.1.5 Observability of local perspectives

One may suspect that the local perspective distortion is only observable for objects with certain viewpoints, which may limit the general applicability of the proposed approach. For example, a vehicle in front of the camera with only the rear face visible shows very little clue about its local perspective

Baselines		MonoFlex						MonoRCNN					
Methods		Val, $AP_{3D}, IoU \geq 0.7$			Val, $AP_{BEV}, IoU \geq 0.7$			Val, $AP_{3D}, IoU \geq 0.7$			Val, $AP_{BEV}, IoU \geq 0.7$		
B	LP	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
✓		23.64	17.51	14.83	31.65	23.29	20.02	16.61	13.19	10.65	25.29	19.22	15.30
	✓	21.13	16.08	14.07	28.91	21.29	18.91	11.98	9.81	8.41	17.61	14.14	12.49
✓	✓	**25.66**	**18.89**	**16.10**	**33.71**	**25.35**	**22.18**	**18.44**	**14.60**	**12.57**	**26.19**	**20.67**	**17.30**

Table 2. Comparison of different settings on KITTI validation set. **B** represents the estimation method in the baseline. **LP** represents our local-perspective estimation with keyedge-ratio regression.

Methods	Val, $AP_{3D}, IoU \geq 0.7$						Val, $AP_{BEV}, IoU \geq 0.7$					
	Easy		Moderate		Hard		Easy		Moderate		Hard	
Ours (based on MonoFlex)	25.66		18.89		16.10		33.71		25.35		22.18	
no grouped heads	23.01	*-2.65*	17.6	*-1.29*	14.82	*-1.28*	31.26	*-2.45*	23.48	*-1.87*	20.02	*-2.16*
no camera-centric indexing	22.81	*-2.85*	16.31	*-2.58*	13.71	*-2.39*	30.44	*-3.27*	22.35	*-3.00*	19.45	*-2.73*
no grouped heads & camera-centric indexing	21.21	*-4.45*	15.44	*-3.45*	13.20	*-2.90*	29.66	*-4.05*	21.30	*-4.05*	18.37	*-3.81*

Table 3. Ablation study on the KITTI validation set. Performance change compared with our default setup is highlighted in *italic* font.

distortion. An example is shown in Fig. 5. Does our method fail on such objects? The answer is no, as shown in the following.

To inspect this issue, we analyze the depth estimation accuracy for objects at different viewpoints, parameterized by the viewing angle γ as defined in Sec. 3.1. The underlying assumption is that the majority of vehicles in the up front are also axis-aligned, showing only a single face to the camera. It is generally true for the KITTI dataset. If the local perspective does not work for objects up-front, the depth estimation would deteriorate near $\gamma = 0$. Since objects in the center of view (with γ close to zero) tend to have larger depth, the error of estimated depth also tends to be larger. We rule out the effect of different distribution of ground truth depth at different γ by evaluating the relative depth error, i.e., $d_r = |d_{est} - d_{gt}|/d_{gt}$. However, it is not straight-forward to evaluate the (relative) depth error in a 3D detection task, because there is no clear definition on pairs of corresponding ground truth objects and estimations, which depends on detection association and confidence thresholds. Here we mimic *AOS* (average orientation similarity) in KITTI [9] and design a new metric called average relative depth error (*ARDE*):

$$ARDE = \frac{1}{40} \sum_{r \in [r_{40}]} \max_{\tilde{r} \geq r} s(\tilde{r}) \tag{10}$$

where

$$s(r) = \frac{1}{|\mathcal{D}(r)|} \sum_{i \in \mathcal{D}(r)} d_r(i) \tag{11}$$

in which $[r_{40}]$ is the set of 40 recall-rate points used in calculating average precision (*AP*), $\mathcal{D}(r)$ is the set of true positive detections at recall rate r, and $d_r(i)$ is the relative depth error of detection i. Here the *recalls* are w.r.t. the 2D bounding box detections, consistent with *AOS* in KITTI. This metric reflects the relative depth error for objects with various detection confidence. In Fig. 6, we show the result of depth error from local perspectives (LP) only based on the MonoFlex baseline. The range of γ is set to $[-30°, 20°]$ because objects outside of the range are of far less frequency. It turns out that there is no obvious pattern in the *ARDE* w.r.t. γ the viewing angle, indicating that the LP module does not suffer from up-front objects. The reason might be that the network is able to infer keyedge-ratios from the global information, exploiting the connection between global perspective information (position, size, etc.) and the local perspective distortion implicitly.

4.2. Experiments on nuScenes dataset

We conduct further experiments on the nuScenes [2] dataset to show that the value of our method is generally valid across different environments.

First we retrain the MonoFlex[33]-based networks without finetuning hyperparameters for nuScenes. Since MonoFlex only conducted experiments on KITTI, the baseline results are generated by us, and we only work on the images from the FRONT camera which have similar viewpoints as KITTI, resulting in 28,130/6,019 images in the training/validation set. We train 100 epochs with batch size 16 on 8 GPUs.

Then we also incorporate the local perspective module with PGD [32], which reported results on the nuScenes dataset. The official open-sourced implementation of PGD does not include the geometric graph part, thus the baseline used in the following comparison and the network fused with our local perspective module both do not have the geometric graph part. We follow the official split of nuScenes, resulting in 106,416 /36,114 images in the training and validation set. We follow the "1x" schedule of PGD, i.e., training 12 epochs with batch size 16 on 8 GPUs. As only the

Figure 7. Qualitative results on the KITTI validation set. In each column, the image is on the left, and 3D detection results in bird's-eye-view are shown on the right. The ground truths are in red, and the detections are in green. We achieve general improvements over the baseline.

Methods	mAP	car	truck	bus	trailer	construction _vehicle	pedestrian	motorcycle	bicycle	traffic _cone	barrier
MonoFlex*	0.167	0.336	0.080	0.177	0.023	0.000	0.238	0.104	0.057	0.330	0.325
Ours (based on MonoFlex)*	0.182	0.353	0.099	0.175	0.016	0.004	0.255	0.152	0.081	0.347	0.337
PGD	0.317	0.502	0.226	0.291	0.074	0.043	0.425	0.312	0.287	0.551	0.462
Ours (based on PGD)	0.321	0.512	0.233	0.294	0.070	0.044	0.427	0.317	0.295	0.555	0.460

Table 4. Experiment results on nuScenes validation set. The AP definition is different from that in KITTI. * means that the results are only on images from the FRONT camera.

learning schedules for evaluation on the validation set are released by PGD, our result is also on the validation set.

The experiment result is in Tab. 4. Here we report the Average Prevision (AP) metric defined by the official nuScenes benchmark, measured with the 2D center distance on the ground plane and averaged over thresholds of 0.5, 1, 2, 4 meters. mAP is calculated by taking the mean over all classes. We achieved higher mAP and higher AP in most categories, compared with both baselines.

5. Conclusion

We propose a method to leverage the local perspective effect for monocular 3D object detection. We design a new regression target called keyedge ratios to parameterize the local perspective distortion. Given a pair of keyedge ratios and the physical size of an object, the proposed method estimates the depth and the yaw angle of objects without camera intrinsic or extrinsic parameters.

The task of keyedge ratio regression is not restrictive to a specific network architecture, and can be appended to arbitrary backbones and middle layers. Therefore this work can be regarded as a generic plug-in module which may be used to augment a wide series of existing work. We incorporate the local-perspective-based module with three recent representative monocular 3D object detection networks and achieve consistent improvements on the KITTI dataset and the nuScenes dataset.

Our work also has some limitations. While the local-perspective-based method explores a new way for 3D reasoning, it needs to be combined with existing methods (e.g., based on visual size and position) to bring improvements. The method is not designed for objects that are very far away, in which case the local perspective distortion diminishes. As some works that exploit the global perspectives already emerge, it is an interesting direction to incorporate the global and local perspective effects together to improve the estimation of both of them. We believe our approach provides a novel view and additional opportunities for future research on monocular 3D object detection.

References

[1] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9287–9296, 2019.

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan,

Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019.

[3] Hansheng Chen, Yuyao Huang, Wei Tian, Zhong Gao, and Lu Xiong. Monorun: Monocular 3d object detection by reconstruction and uncertainty propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10379–10388, 2021.

[4] 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 *Advances in Neural Information Processing Systems*, pages 424–432. Citeseer, 2015.

[5] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12093–12102, 2020.

[6] Tom van Dijk and Guido de Croon. How do neural networks see depth in single images? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2183–2191, 2019.

[7] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and Ping Luo. Learning depth-guided convolutions for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1000–1001, 2020.

[8] Jose M Facil, Benjamin Ummenhofer, Huizhong Zhou, Luis Montesano, Thomas Brox, and Javier Civera. Camconvs: Camera-aware multi-scale convolutions for single-view depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11826–11835, 2019.

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

[10] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 5580–5590, Red Hook, NY, USA, 2017. Curran Associates Inc.

[11] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11867–11876, 2019.

[12] Abhijit Kundu, Yin Li, and James M Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3559–3568, 2018.

[13] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. Gs3d: An efficient 3d object detection framework for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1019–1028, 2019.

[14] Peixuan Li and Huaici Zhao. Monocular 3d detection with geometric constraint embedding and semi-supervised training. *IEEE Robotics and Automation Letters*, 6(3):5565–5572, 2021.

[15] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 644–660. Springer, 2020.

[16] Shichao Li, Zengqiang Yan, Hongyang Li, and Kwang-Ting Cheng. Exploring intermediate representation for monocular vehicle pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1873–1883, 2021.

[17] Qing Lian, Botao Ye, Ruijia Xu, Weilong Yao, and Tong Zhang. Geometry-aware data augmentation for monocular 3d object detection. *arXiv preprint arXiv:2104.05858*, 2021.

[18] Yuxuan Liu, Yuan Yixuan, and Ming Liu. Ground-aware monocular 3d object detection for autonomous driving. *IEEE Robotics and Automation Letters*, 6(2):919–926, 2021.

[19] Zechen Liu, Zizhang Wu, and Roland Tóth. Smoke: Single-stage monocular 3d object detection via keypoint estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 996–997, 2020.

[20] Zongdai Liu, Dingfu Zhou, Feixiang Lu, Jin Fang, and Liangjun Zhang. Autoshape: Real-time shape-aware monocular 3d object detection. In *ICCV*, pages 15641–15650, 2021.

[21] 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*, pages 3111–3121, 2021.

[22] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4721–4730, 2021.

[23] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2069–2078, 2019.

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

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, 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 H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Informa-*

tion Processing Systems 32, pages 8024–8035. Curran Associates, Inc., 2019.

[26] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3d object detection. In *CVPR*, pages 8555–8564, 2021.

[27] Xuepeng Shi, Qi Ye, Xiaozhi Chen, Chuangrong Chen, Zhixiang Chen, and Tae-Kyun Kim. Geometry-based distance decomposition for monocular 3d object detection. *arXiv preprint arXiv:2104.03775*, 2021.

[28] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1991–1999, 2019.

[29] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Elisa Ricci, and Peter Kontschieder. Towards generalization across depth for monocular 3d object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 767–782. Springer, 2020.

[30] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 454–463, 2021.

[31] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. *arXiv preprint arXiv:2104.10956*, 2021.

[32] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Probabilistic and geometric depth: Detecting objects in perspective. *arXiv preprint arXiv:2107.14160*, 2021.

[33] Yunpeng Zhang, Jiwen Lu, and Jie Zhou. Objects are different: Flexible monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3289–3298, 2021.

[34] Yinmin Zhang, Xinzhu Ma, Shuai Yi, Jun Hou, Zhihui Wang, Wanli Ouyang, and Dan Xu. Learning geometry-guided depth via projective modeling for monocular 3d object detection. *arXiv preprint arXiv:2107.13931*, 2021.

[35] Yunsong Zhou, Yuan He, Hongzi Zhu, Cheng Wang, Hongyang Li, and Qinhong Jiang. Monocular 3d object detection: An extrinsic parameter free approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7556–7566, 2021.