ER3D: An Efficient Real-time 3D Object Detection Framework for Autonomous Driving

Haitao Meng, Changcai Li, Gang Chen, Zonghua Gu and Alois Knoll

***Abstract*—3D object detection is a vital computer vision task in mobile robotics and autonomous driving. However, most existing methods have exclusively focused on achieving high accuracy, leading to complex and bulky systems that can not be deployed in a real-time manner. In this paper, we propose the *ER3D (Efficient and Real-time 3D)* object detection framework, which takes stereo images as input and predicts 3D bounding boxes. Instead of using the complex network architecture, we leverage a fast-but-inaccurate method semi-global matching (SGM) for depth estimation. To eliminate the accuracy degradation in 3D detection caused by inaccurate depth estimation, we introduce decoupled regression head and 3D distance-consistency IoU loss to boost the accuracy performance of the 3D detector with a small computing overhead. ER3D achieves both high-precision and real-time performance to enable practical applications of 3D object detection systems on robotic systems. Extensive experiments with the comparison of the state of the arts demonstrate the superior practicability of ER3D, which achieves comparable detection accuracy with significant leadership on inference efficiency.**

I. Introduction

3D object detection is an essential vision task in scene perception and motion prediction of autonomous driving. LiDAR-based detection can achieve highly accurate and reliable environment perception [10], [29], but the high cost of LiDAR hardware hinders its mass deployment. Vision-based (Stereo or monocular) 3D object detection is considered as an attractive alternative solution to LiDAR due to the low cost of vision sensors (cameras). We focus on stereo vision-based 3D object detection in this paper. Although state-of-the-art approaches may achieve success in improving accuracy or accelerating models, none of them could fulfill the demand of obtaining accurate and real-time 3D object detection at the same time. We aim to bridge this gap in this paper, and present the object detection framework ER3D (Efficient and Real-time 3D) that achieves both high precision and real-time performance.

Currently, there have been significant advancements [2], [6], [11], [17] in the field of vision-based 3D object detection due to the appealing advantages of low cost, compact size, and high energy efficiency. Generally, these methods can be broadly categorized into accuracy-first approaches and efficiency-first approaches according to the emphasis aspects. The advancements that prioritize detection accuracy [2], [4], [17] typically propose a diverse array of novel architectures and system pipelines with the aim of elaborating the undiscovered optimization opportunities. Although these proposals tend to be accurate and reliable in producing 3D detection results, the bulky and high computational complexity architecture hinders the deployment of resource-constrained embedded systems (e.g., drones). For example, DSGN [2],

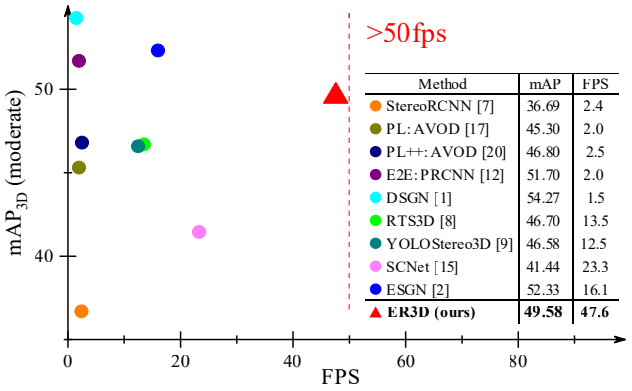


Fig. 1: Comparison of ER3D with state-of-the-art approaches. We illustrate the Accuracy (mAP_{3D} of $IoU = 0.7$ on moderate set of KITTI validation dataset) on the y-axis and FPS (Frames Per Second) on the x-axis. We collect the results of accuracy and runtime from the published papers. For more details please check Tab. II.

which achieves impressive 54.27% mAP_{3D} detection accuracy on the moderate set, takes 682 ms on the reasonably powerful Tesla V100 GPU platform, as shown in Fig. 1 and Tab. II. It is far too slow for mobile robotics systems. The second category which focuses on system latency performance offers solutions for improving computation efficiency and timeliness. They optimize their system through reduction and lightweight techniques to mitigate the computational burden. However, these techniques can only be seen as a rough and simple approach toward real-time responsiveness since they failed to optimize their system in a systematic view. As a result, they have to compromise their accuracy to achieve improvement in real-time performance.

To tackle the aforementioned problem, we propose a real-time 3D object detection framework *ER3D* that produces state-of-the-art detection accuracy, as shown in Fig. 1. ER3D leverages an efficient but less accurate stereo matching algorithm, Semi-Global Matching (SGM) [7], as the depth estimator to reduce the computational burden, attaining a significant boost on the real-time responsiveness. To eliminate the accuracy degradation produced by the biased depth estimation of SGM, we first perform an ablation study to identify the most important factors that impact detection accuracy. Then, based on this knowledge, we adopt decoupled regression head to place emphasis on important branches of the IoU metric to better understand the inner pattern of the depth map. Moreover, we define a custom loss function to further

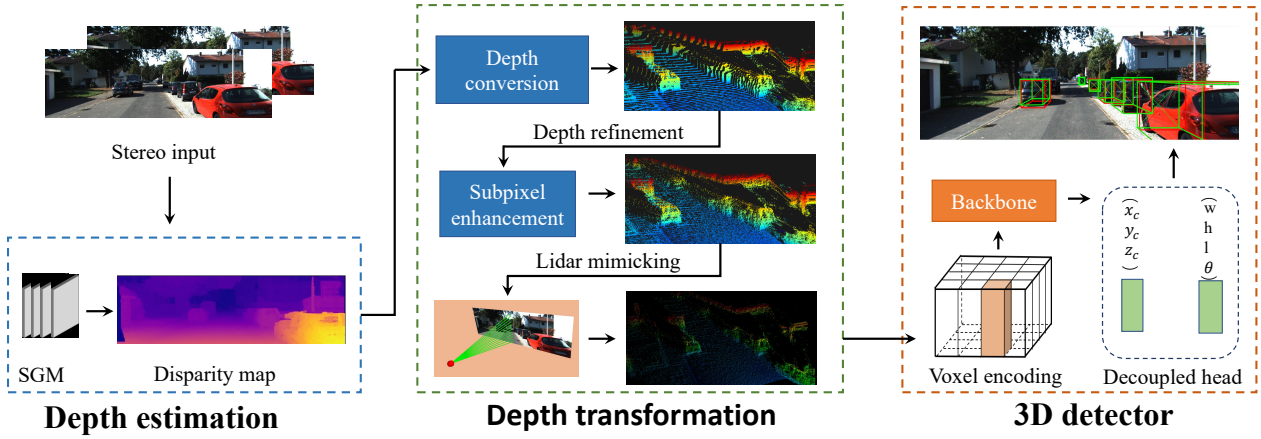


Fig. 2: Overview of proposed ER3D system.

sense the localization errors and amend it with corresponding loss terms. Performance evaluation on the KITTI dataset indicates that ER3D achieves a good tradeoff between the computational cost and the detection accuracy. To the best of our knowledge, ER3D is the fastest stereo-based 3D detection system (47 fps on TITAN RTX GPU) with state-of-the-art detection accuracy. It achieves 2× inference speed while still achieving similar accuracy as the state-of-the-art algorithms. (A release of code for reproducing our results, as well as a video demo, is available at an anonymous GitHub link https://github.com/weiyangdaren/ER3D.)

The rest of this article is organized as follows: In Section II, we elaborate on the motivation of this article through an ablation experiment. Then we present our real-time 3D object detection framework in Section III. The comparisons of our system with state-of-the-art are illustrated in Section IV. Section V briefly introduces stereo-matching algorithms and stereo-based 3D object detection approaches. Finally, we conclude this article in Section VI.

TABLE I: Results by replacing the specific item of the predicted labels and the ground truth labels on the KITTI validation set. We report the metric AP_{3D} with $IoU = 0.7$ of the car category. The ground truth values are denoted as gt and predicted values are denoted as $pred$. Left: results of PL:AVOD [25] with partial replacement with ground truth labels. Right: evaluation of ground truth labels partially covered by predicted results. Red indicates the delta of improvement and Green indicates the degradation. Note that x-axis denotes the depth dimension in the point cloud coordinate.

Configuration	mAP	Configuration	mAP
PL:AVOD [25]	38.97	Ground Truth [5]	100
w/ gt x_c	50.39(+11.42)	w/ pred x_c	64.36 (-35.64)
w/ gt y_c	45.86(+6.89)	w/ pred y_c	98.95(-1.05)
w/ gt z_c	41.33(+2.36)	w/ pred z_c	81.54(-18.46)
w/ gt (x_c, y_c, z_c)	68.18(+29.21)	w/ pred $((x_c, y_c, z_c)$)	40.56 (-59.44)
w/ gt w	39.51(+0.6)	w/ pred w	98.89(-1.11)
w/ gt h	39.01(+0.04)	w/ pred h	99.93(-0.07)
w/ gt l	39.84(+0.9)	w/ pred l	98.23 (-1.77)
w/ gt (w, h, l)	40.31(+1.34)	w/ pred (w, h, l)	85.58(-14.42)
w/ gt θ	39.43(+0.46)	w/ pred θ	81.06(-18.94)

II. Motivation

Estimating the bounding boxes (BBs) of the interest objects is the fundamental task of 3D object detection. For a stereo pair (I_L, I_R), the crucial goal of the models is to locate the center of each object C (x_c, y_c, z_c) and estimate the 3D size S (w, h, l) as well as the horizontal orientation θ in the 3D space. LiDAR-based methods [10], [29] outperform pseudo-LiDAR based methods [17], [25] in terms of detection accuracy, even with the same 3D detector model. Since the only difference between the two of them is the precision of depth perception (i.e., LiDAR point cloud measurement vs stereo-image-based depth estimation). We can easily attribute this accuracy degradation to imprecise depth perception. To verify this assumption, we follow the same idea of [14] to conduct an ablation experiment.

Setup. The most used evaluation metric, IoU [31], reveals the overlap proportion of the predicted BB with ground truth. Since it can be seen as a combination of seven indicators $(x_c, y_c, z_c, w, h, l, \theta)$, we can replace one of the indicators with ground truth while remaining others to separately test the effect of a specific item on the overall. Driven by this idea, we collect the ablation experiment results from the BB predictions of PL:AVOD [25] approach and summarize them in Tab. I.

Observations. Among seven different branches of replacement, a prominent accuracy boost appears when we replace the depth-related values (x_c, y_c, z_c) with the ground truth, implying that the most accuracy drop can be explained by inaccurate depth of the object center (x_c indicates the distance of the object to the optical center of the camera, y_c and z_c are derived with the participation of depth). Similar fact also arise when we cover the ground truth with predicted results (right side replacement of Tab. I). Therefore, we conclude that the precision of depth estimation is the key factor that affects the accuracy of 3D object detection.

III. Method

In this section, we first outline the flow of the proposed efficient and real-time 3D detection system. Then, we elaborate on two accuracy promotion schemes that we utilize to

enhance detection accuracy. Toward the practical deployment, we also provide parallel optimizations for GPU devices of our system, ensuring efficient utilization of hardware resources and enabling faster inference speeds.

A. ER3D Overview

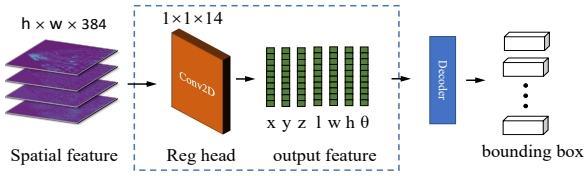
Driven by the observations we learned aforementioned, we propose our efficient and real-time 3D object detection framework. Fig. 2 illustrates the overview of the proposed ER3D framework in which depth estimation, depth transformation, and 3D detection are contained. For the depth estimation module, we abandon the inherent routine of employing deep and complex depth estimation network [17], [25], [29]. Instead, the fast-but-inaccurate method SGM [7] is utilized for depth estimation from the disparity map. The biggest challenge of doing so is the shaky depth map that significantly distracts the regression of the 3D detector, leading to poor detection performance. To eliminate this drawback, we first adopt a sub-pixel enhancement scheme [30] to transfer integer disparities of SGM [7] into the sub-pixel values, initiating continuously distribute point cloud representation. Then, we further propose decoupled regression head to place key emphasis on valuable regression branches (x_c, y_c, z_c) to gain the accuracy promotion with little overhead. In addition, with the consideration of the training process also exist unconsecrated error measurement and gradients, we also introduce our distance-consistency IoU loss to sense the localization errors and compensate it with corresponding loss terms.

To reduce computation load, we down-sampling the dense point cloud data transformed from the depth map by mimicking real LiDAR representation. This will considerably reduce the redundant information that is not in the interest regions (i.e., road and wall) and save tons of computing resources in the inference phase. Then, the sparse point cloud data is fed to the 3D detector to produce the final predictions. We adopt PointPillar [10] as the 3D detector for its highly efficient inference capability.

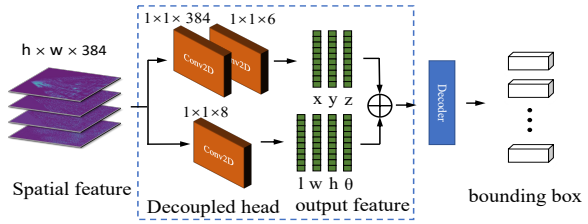
B. Decoupled Regression Head

Based on Tab. I, one of the major concerns about the indicators prediction is their imbalance contributions to the IoU metric. However, the regression architecture and training strategy of prior stereo visual-based approaches still stuck in the domain tailored for LiDAR-based methods [14], [25], [29], failed to fully exploit their underlying detection power.

Fig. 3(a) illustrates the architecture of a typical regression head. It first takes the spatial features from the previous network as input. Then, it utilizes a 1×1 convolutional layer to interpret these features and decode them as desired 3D information. However, due to the inaccurate depth issues of the visual algorithms, merely relying on one simple convolution layer can hardly distinguish the difference among various spatial dimensions (i.e., x_c, h, θ). To this end, we introduce decoupled regression head to help the network learns data patterns of different branches in the regression process. As



(a) Original regression head architecture



(b) Decoupled regression head architecture

Fig. 3: Illustration of different regression head. The convolutional layers adopt 1×1 kernel with 6, 8, 14 or 384 channels output.

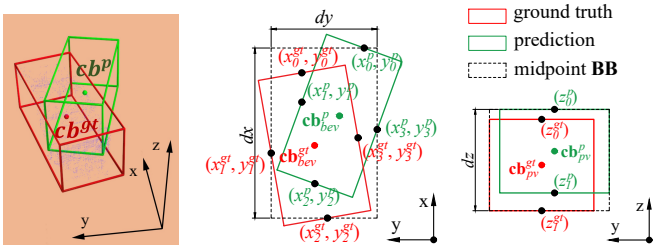
shown in Fig. 3(b), we divide the prediction of the indicators into two groups in which location-related items (x_c, y_c, z_c) are separately predicted with two 1×1 convolutional layers. By doing this, indicators corresponding to the inaccurate depth are pulled out and learned with dedicated layers, deducing a better prediction for the offset of the predicted depth.

We also measured the overhead caused by the proposed regression head. Without any modification, the original architecture of our 3D detector possesses 4.8M parameters. Then, if we adopt the proposed decoupled head, we could obtain a 3D detector with 4.9M parameters, which is only 0.1M more than the original one. According to the ablation study of Sec. IV-B, the introduced decoupled regression head can improve the overall accuracy by 2.58% with only 1.2 ms extra computing. Therefore, we believe this is a beneficial trade-off for the practical usage.

C. Loss Function Definition

When taking the LiDAR signal as input, supervising the training with smooth L1 Loss function could be effective [31]. However, for stereo visual-based 3D detection approaches, due to the objective fact that the depth prediction of distant objects is not accurate enough, there are inevitable offsets in different axis when transforming the 2D depth map into the 3D point cloud. In the 3D space, these offsets can lead to different contributions to the IoU metric and confuse the gradient direction, yielding an ill-posed convergence, as we clearly demonstrated in Tab. I. A similar observation was also reported in [14] where labels in the large distance are abandoned in the training phase to reduce the effect of biased depth input. However, this can only be seen as a rough strategy since inaccurate depth with a small offset will remain degrading the training of the model.

As shown in Tab. I, inaccurate depth is the main reason that affects the accuracy of the 3D detection. When we transform



(a) 3D view (b) BEV (c) PV

Fig. 4: An illustrative example. The 3D view in (a) is projected into the Bird's Eye View (BEV) in (b) and the Perspective View (PV) in (c). The ground truth 3D BB and its projections in BEV and PV are denoted by red color; the predicted 3D BB and its projections are denoted by green color.

the depth map into the 3D space coordinate (point cloud representation), this inaccurate depth will be represented as an inaccurate positioning problem (x_c, y_c, z_c). Therefore, to guide the 3D detector sensing this localization error, we define our loss function in (1):

$$\mathcal{L} = \mathcal{L}_{IoU} + \mathcal{L}_{smooth} + \mathcal{L}_d + \mathcal{L}_c \tag{1}$$

where the IoU loss $\mathcal{L}_{IoU} = 1 - IoU_{3D}$ and smooth L1 loss $\mathcal{L}_{smooth}$ are well-known terms from related works [10], [20] (details omitted); we will discuss the 3D distance loss $\mathcal{L}_d$ and the consistency loss $\mathcal{L}_c$ in detail next.

Fig. 4 shows an example to illustrate our loss function definition. We use $\mathbf{BB}^p$ and $\mathbf{BB}^{gt}$ to denote the predicted 3D BB and the ground truth 3D BB, respectively; $\mathbf{BB}^p_{bev}$ and $\mathbf{BB}^{gt}_{bev}$ to denote the predicted 2D BB and the ground truth 2D BB in BEV, respectively; $\mathbf{BB}^p_{pv}$ and $\mathbf{BB}^{gt}_{pv}$ to denote the predicted 2D BB and the ground truth 2D BB in perspective view(PV), respectively. We use $\mathbf{cb}^p$, $\mathbf{cb}^{gt}$, $\mathbf{cb}^p_{bev}$, $\mathbf{cb}^{gt}_{bev}$, $\mathbf{cb}^p_{pv}$ and $\mathbf{cb}^{gt}_{pv}$ to denote the center points of the corresponding BBs.

3D Distance Loss $\mathcal{L}_d$. We define the 3D Distance Loss $\mathcal{L}_d$ to minimize the normalized distance between the center points of the ground truth 3D BB and predicted 3D BB, as an extension to the corresponding loss function in [31] from 2D to 3D:

$$\mathcal{L}_d = \frac{\delta^2(\mathbf{cb}^p, \mathbf{cb}^{gt})}{dd^2} \tag{2}$$

where $\delta(\cdot)$ denotes the Euclidean distance function. The numerator, the squared Euclidean distance between two center points $\mathbf{cb}^p$ and $\mathbf{cb}^{gt}$, is equal to the squared diagonal length of the cube formed by the rectangle with diagonal line defined by two center points $\mathbf{cb}^p_{bev}$ and $\mathbf{cb}^{gt}_{bev}$ on the (x, y) plane (BEV), and the height equal to the z-axis vertical distance between two center points $\mathbf{cb}^p_{pv}$ and $\mathbf{cb}^{gt}_{pv}$ (PV). It can be computed with Pythagorean Theorem:

$$\delta^2(\mathbf{cb}^p, \mathbf{cb}^{gt}) = \delta^2(\mathbf{cb}^p_{bev}, \mathbf{cb}^{gt}_{bev}) + (\mathbf{cb}^p_{pv}(z) - \mathbf{cb}^{gt}_{pv}(z))^2 \tag{3}$$

where the first term denotes the squared Euclidean distance between the center points of 2D BBs $\mathbf{cb}^p_{bev}$ and $\mathbf{cb}^{gt}_{bev}$ on the (x, y) plane in BEV, and the second term denotes the squared height of the cube.

The denominator of (2), dd^2, is a normalization factor used to limit $\mathcal{L}_d$ to be in the range of $[0, 1]$. First, we consider the BEV, and define the *midpoint BB* as the minimum 2D BB enclosing the 8 BB edge midpoints, including $(x^p_0, y^p_0), \ldots, (x^p_3, y^p_3)$ of the predicted BB, and $(x^{gt}_0, y^{gt}_0), \ldots, (x^{gt}_3, y^{gt}_3)$ of the ground truth BB, shown as the dotted rectangle in Fig. 4(b). (We utilize the BB edge midpoints to define a relatively small midpoint BB, in order to produce a bigger loss term than using the BB vertex points to define a larger enclosing rectangle.) We use dx and dy to denote the dimensions of the midpoint BB on the x-axis and y-axis, respectively. It can be easily shown that the diagonal length of the midpoint BB is an upper bound on the Euclidean distance between $\mathbf{cb}^p_{bev}$ and $\mathbf{cb}^{gt}_{bev}$. Next, we consider the PV, and construct the smallest enclosing BB that covers both $\mathbf{BB}^p_{pv}$ and $\mathbf{BB}^{gt}_{pv}$, shown as the dotted rectangle in Fig. 4(c). We use z^p_0 and z^p_1 to denote height of the upper edge and lower edge of $\mathbf{BB}^p_{pv}$, respectively; z^{gt}_0 and z^{gt}_1 to denote height of the upper edge and lower edge of $\mathbf{BB}^{gt}_{pv}$, respectively; and dz to denote its height. It is obvious that dz is an upper bound on the vertical distance between $\mathbf{cb}^p_{pv}$ and $\mathbf{cb}^{gt}_{pv}$. The normalization factor dd^2 is defined as the squared diagonal length of the cube formed by the midpoint BB on the (x, y) plane (BEV) and the height equal to dz (PV). It can be computed with Pythagorean Theorem:

$$\begin{aligned} dx &= \max(x^p_0, \ldots, x^p_3, x^{gt}_0, \ldots, x^{gt}_3) \\ &\quad - \min(x^p_0, \ldots, x^p_3, x^{gt}_0, \ldots, x^{gt}_3) \\ dy &= \max(y^p_0, \ldots, y^p_3, y^{gt}_0, \ldots, y^{gt}_3) \\ &\quad - \min(y^p_0, \ldots, y^p_3, y^{gt}_0, \ldots, y^{gt}_3) \\ dz &= \max(z^p_0, z^p_1, z^{gt}_0, z^{gt}_1) - \min(z^p_0, z^p_1, z^{gt}_0, z^{gt}_1) \\ dd^2 &= dx^2 + dy^2 + dz^2 \end{aligned} \tag{4}$$

From our above discussions, we conclude that (2) returns a normalized value in the range of $[0, 1]$.

Consistency loss $\mathcal{L}_c$. We define $\mathcal{L}_c$ to measure the consistency of aspect ratios of the ground truth 3D BB and predicted 3D BB, as an extension to the corresponding loss function in [31] from 2D to 3D:

$$\mathcal{L}_c = \alpha v \tag{5}$$

where v measures the consistency of aspect ratio:

$$\begin{aligned} v = \frac{4}{3\pi^2}[&(\arctan\frac{h^p}{w^p} - \arctan\frac{h^{gt}}{w^{gt}})^2 \\ &+ (\arctan\frac{h^p}{l^p} - \arctan\frac{h^{gt}}{l^{gt}})^2 \\ &+ (\arctan\frac{w^p}{l^p} - \arctan\frac{w^{gt}}{l^{gt}})^2] \end{aligned} \tag{6}$$

And α is a trade-off weighting parameter defined as:

$$\alpha = \frac{v}{1 - IoU_{3D} + v} \tag{7}$$

by which the overlap area factor is given higher priority for regression, especially for non-overlapping cases [31].

D. Parallel Optimizations

Building on top of autonomous driving and mobile robotics, our system has been designed to avoid intensive calculation and complex structure to be aligned with the objective of real-time inference. We identify the critical path of each module and exploit the acceleration with CUDA programming to achieve the speed-up of the whole system. With these optimizations, our system can perform an accurate 3D object detection with 35 ms on the embedded device and 21 ms on the powful TITAN RTX GPU platform, enabling its practicability on deploying into the real applications.

For the depth estimator, we optimize a sub-pixel enhancement implementation for our SGM [7] algorithm. Since there is no data dependency along the disparity dimension, our proposal simply exploits the acceleration with the shared memory which preloads the whole slice of matching cost along the disparity dimension for block threads to achieve better memory bandwidth utilization. We launch massive of parallel threads with intensive register data reuse and warp-level data transformation to avoid fully depending on expensive global memory access. Meanwhile, since obtaining the pseudo-Lidar representation from dense depth map is also a main bottle neck of our system, which can easily consumes dozens or even hundreds of milliseconds [25]. We hence exploit a parallelism on GPUs to perform it timely. In the depth transformation module, we further utilize the efficient $cublasSgemm()$ function provided by the NVIDIA cuBLAS library [16] to obtain the 3D locations in the reference coordinate and the LiDAR coordinate. The $thrust$ [16] API is also adopted during the transformation to compact the data size by throwing outliers and invalid points.

IV. Experiment

A. Experiment Setup

Dataset. We evaluate ER3D on two 3D object detection benchmarks: KITTI [5] and ApolloScape [8]. The KITTI dataset contains 7,481 training samples and 7,518 (online) test samples, and the training samples are further divided into the train split (3,712 samples) and the validation split (3,769 samples). To evaluate the generalization ability, we also utilize the ApolloScape dataset [8] to conduct the qualitative evaluation for the proposed ER3D.

Performance Metrics. In the KITTI benchmark, objects are divided into three difficulty categories: $easy$, $moderate$, and $hard$ according to the situation of occlusion and the size of the interest object in the 3D image. In each category, approaches are evaluated by different IoU thresholds (0.5 or 0.7) in the BEV perspective and 3D perspective. On the test set mAP is calculated with 40 recall positions by the official server. The results on the validation set are calculated with 11 recall positions for fair comparison with other approaches.

Training Details. Our system contains the learnable model which takes LiDAR-like data as input to predict the 3D information. To obtain its training set, we utilize SGM with sub-pixel enhancement module to produce disparity maps. Then, we transform them into the point cloud representation to form the training set. Note that we only need to train the 3D detector module instead of the whole system. We train the 3D detector on NVIDIA TITAN RTX GPU with batch size of 4 for 80 epochs. We adopt the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.99$ and 0.01 weight decay.

B. Results

Comparison with State-of-the-art Methods. In this evaluation, we compare ER3D with state-of-the-art approaches that can be divided into accuracy-first approaches [2], [6], [11], [17], [18], [25], [29] and efficiency-first approaches [4], [12], [13], [23], [26]. Tab. II summarizes the mAP performance, runtime, and test platform of each method.

When we compare ER3D with accuracy-first approaches, ER3D suppresses TLNet [18], Stereo RCNN [11], PL:AVOD [25] and PL++:AVOD [29] by a large margin. It even achieves comparable accuracy to the E2E:PRCNN [17] and DSGN [2], and is only lag behind the most accurate 3D detector LIGA [6]. Moreover, we have to point out that all of these accuracy-first approaches require at least 350ms of computing. In contrast, the computing latency of ER3D is 21 ms, which is an order of magnitude faster than all of these approaches. Clearly, the large margin lead in running speed and the outstanding results in accuracy is the strongest proof of ER3D for its practicability and effectiveness.

Efficiency-first 3D detection models are also listed in Tab. II, whose runtime are less than 100 ms. Among these competitors, ER3D is no doubt the fastest 3D detection system with at least 2 times speed ahead among all compared state-of-the-art approaches. Besides this leadership in inference time, ER3D also outperformed RTS3D [12], YOLOStereo3D [13] and SCNetc [23] with notable leadership in accuracy. Although ESGN [4] achieves slightly higher accuracy than ER3D, it is worth noting that ER3D achieves 3 $\times$ faster inference speed than ESGN [4] using the GPU with half the computing power than ESGN [4].

Tab. III further illustrates the results of detection accuracy and the runtime on the KITTI online evaluation benchmark. ER3D retains the same trend as the validation set. It still achieves comparable results to most accurate approaches with a significant advantage on the runtime. In addition, there is interesting fact that ER3D always achieves relatively higher accuracy of AP_{3D} than it is of AP_{BEV} when compared to others. For example, the AP_{BEV} of ER3D is lower than E2E:PRCNN [17] by more than 3%, while this gap is narrowed to 0.82% in the AP_{3D}. Moreover, ER3D even achieves a leadership by 2.57% higher mAP on the $easy$ set of AP_{3D}

Fig. 5: Qualitative results of ER3D on the two datasets. The top two rows illustrate the prediction and the ground truth of the KITTI dataset and the bottom two rows show the 3D detection results on the ApolloScape dataset. The predictions of ER3D are illustrated as green and the ground truth 3D labels are illustrated as red. The depth of SGM is utilized to generate the point cloud in the first and the third row.

TABLE II: Results evaluated by metric AP_{BEV}/ AP_{3D} on validation set of the car category. GPU performance of FP32 (float) in terms of TFLOPS is obtained from NVIDIA documentation. RT stands for runtime in ms.

Method	Reference	Platform	RT(ms)	IoU = 0.5			IoU = 0.7		
				Easy	Moderate	Hard	Easy	Moderate	Hard
TLNet [18]	2019 CVPR	-	-	62.46/59.5	45.99/43.71	41.92/37.99	29.22/18.15	21.88/14.26	18.83/13.72
Stereo RCNN [11]	2019 CVPR	-	417	87.13/85.84	74.11/66.28	58.93/ 57.24	68.50/54.11	48.30/36.69	41.47/31.07
PL:AVOD [25]	2019 CVPR	TITAN RTX(16.3 TFLOPS)	514	89.0/88.5	77.5/76.4	68.7/61.2	74.9/61.9	56.8/45.3	49.0/39.0
PL++:AVOD [29]	2020 ICLR	TITAN RTX(16.3 TFLOPS)	399	89.4/89.0	79.0/77.8	70.1/69.1	77.0/63.2	63.7/46.8	56.0/39.8
E2E:PRCNN [17]	2020 CVPR	-	490	90.5/90.4	84.4/79.2	78.4/75.9	82.7/71.1	65.7/51.7	58.4/46.7
DSGN [2]	2020 CVPR	Tesla V100(14.1 TFLOPS)	682	- / -	- / -	- / -	83.24/72.31	63.91/54.27	57.83/47.71
LIGA [6]	2021 ICCV	TITAN Xp(12.1 TFLOPS)	350	97.22/97.06	90.27/89.97	88.36/87.94	89.35/84.92	77.26/67.06	69.05/63.80
RTS3D [12]	2021 AAAI	TITAN RTX(16.3 TFLOPS)	74	90.58/90.34	80.72/79.67	71.41/70.29	77.50/64.76	58.65/46.70	50.14/39.27
YOLOStereo3D [13]	2021 ICRA	GTX 1080Ti(11.3 TFLOPS)	80	- / -	- / -	- / -	- /72.06	- /46.58	- /35.53
PLUMENet [26]	2021 IROS	Tesla V100(14.1 TFLOPS)	80	87.8/ -	80.7/ -	75.2/ -	74.4/ -	61.7/ -	55.8/ -
SCNet [23]	2022 NC	Tesla V100(14.1 TFLOPS)	43	- / -	- / -	- / -	71.26/ 55.25	53.27/ 41.44	45.53/ 35.13
ESGN [4]	2022 TCSVT	RTX 3090(35.6 TFLOPS)	62	93.05/ -	82.22/ -	72.25/ -	82.24/72.44	63.86/52.33	64.63/43.74
ER3D	-	TITAN RTX(16.3 TFLOPS)	21	90.40/90.27	79.61/78.42	75.32/72.94	81.25/70.10	59.48/49.58	54.47/46.00

when compared to the E2E:PRCNN [17]. We believe this is due to the adoption of the decoupled regression head where depth-related items are predicted with independent branches to produce more accurate localization estimation.

Ablation Experiments. In order to demonstrate the effectiveness of the proposed promotion schemes, we provide this ablation experiment by applying different techniques and reporting their performance. Tab. IV summarize the accuracy of AP_{3D} and the runtime performance. It is clear that adopting the DC-IoU (dc.) and the decoupled head (dh.) achieve improvement over the baseline among every evaluation metric. On the $easy$ subset, approximately 3% of accuracy leap can be observed, which is the biggest benefit ER3D can attain by separately employing the DC-IoU (dc.) or decoupled head (dh.). If we enable these two promotion schemes simultaneously, ER3D can further boost its performance by up to 2%. It is also worth noting that, compared to the baselines, the adoption of the decoupled head (dh.) only increases 0.12 ms of extra computing overhead. Considering their solid accuracy improvement, we believe this ablation experiment is convincing enough to demonstrate the superior performance of our proposed approaches.

Qualitative Results. To demonstrate the generalization ability of ER3D, we visualize the 3D prediction results of two different dataset: KITTI dataset [5] and ApolloScape dataset [8]. The snapshot of the examples on two datasets is presented in Fig. 5 Note that we do not perform any fine-tuning or re-training on the ApolloScape dataset. Instead, we use the model

TABLE III: Results of the car category on KITTI test set. The AP_{BEV}/AP_{3D} in percentage at $IoU = 0.7$ are reported. All results are collected from the official leaderboard of KITTI website and corresponding published papers.

Method	RT(ms)	Easy	Moderate	Hard
TLNet [18]	-	29.22/18.15	21.88/14.26	18.83/13.72
PL:AVOD [25]	514	- /55.4	- /37.1	- / 31.37
PL++:AVOD [29]	399	66.8/ -	47.2/ -	40.30/ -
E2E:PRCNN [17]	490	79.6/64.8	58.8/43.9	52.1/38.1
LIGA [6]	350	88.15/81.39	76.78/64.66	67.40/57.22
YOLOStereo3D [13]	80	- /65.68	- /41.25	- / 30.42
SCNet [23]	43	62.97/49.94	42.12/31.3	35.37/25.62
RTS3D [12]	74	72.1/58.5	51.7/37.3	43.1/ 31.1
ESGN [4]	62	- /65.80	- /46.39	- /38.42
ER3D	21	76.19/67.37	54.68 /43.59	47.81/37.28

TABLE IV: Results on accumulating the proposed approaches of ER3D on KITTI validation set. RT is an abbreviation for runtime. We report AP_{3D} metric of the car category with $IoU = 0.7$. dc. is the DC-IoU loss. dh. represents the decoupled head. The baseline is defined as ER3D without dc. and dh. All reported results are collected from TITAN RTX GPU.

Configuration	Easy	Moderate	Hard	RT(ms)
baseline	66.55	48.15	42.31	19.9
+ dc.	69.13	48.72	43.77	19.9
+ dh.	69.23	49.81	43.85	21.1
+ dc. + dh.	70.10	49.58	46.00	21.1

trained on the KITTI dataset directly to obtain these qualitative results. As indicated in the figures, even though ApolloScape dataset [8] is a totally different dataset with different camera sensors and road scenarios to the KITTI dataset, ER3D still satisfactorily detects cars at different distances. It means that ER3D can be well generalized to the unseen dataset and hence is practical for real-world applications.

V. Related Work

In this section, we begin by introducing stereo matching algorithms, the preliminary techniques that we used to obtain the depth information of the surroundings. Then, we place our effort in the context of stereo-based 3D object detection approaches, providing a detailed overview of existing approaches and discussing their limitations.

A. Stereo Depth Prediction.

By exploiting the relationship of the same object between the different perspectives of the stereo camera, stereo matching algorithms estimate the dense depth map of the environment [21]. Following semi-global matching (SGM) [7] algorithm which proposes a smooth constraint to the matching cost volume establishes a classical paradigm for most inheriting stereo-matching models. However, SGM [7] can not achieve satisfactory accuracy due to the unreliable census transformation. Recent studies utilize deep neural networks (DNNs) to boost the matching cost generation and cost aggregation [1], [3], [27], [30]. Although a significant lead in accuracy is achieved, their intensive computation demand also leads to an explosion of the inference time.

B. Stereo-based 3D Object Detection.

Benefiting from their low-cost and small-size properties, stereo camera-based 3D object detection algorithms have drawn considerable attention and are being intensively explored. With the consideration of performance orientation, we divide established algorithms into accuracy-first and efficiency-first approaches.

Accuracy-First Approaches. Within this line, Pseudo-LiDAR-like stereo 3D detection approaches are popular and achieve solid performance. Typically, these approaches employ sophisticated stereo-matching models [1], [15] to acquire 3D point cloud data and leverage advancements in LiDAR-based 3D detection [9], [22] to generate 3D bounding boxes. Extensive research that fuses sparse LiDAR information [29] and conducts end-to-end training strategy [17] overcomes the shortages associated with depth displacement and suboptimal training paradigm, leading to the achievement of state-of-the-art detection accuracy. However, existing Pseudo-LiDAR approaches heavily rely on leveraging complex stereo-matching algorithms to facilitate the input of the 3D detection module, which consumes considerable computing resources. As a result, they cannot be executed in real-time. There are also additional clusters that directly regress the 3D bounding boxes by exploiting the power of stereo pairs. *Li et al.* [11] extends Faster R-CNN [19] for stereo input to detect and correlate objects in left and right images simultaneously. However, the main concern of this work is the vulnerability to the objects occlusion in the image where the dense alignment was directly operated. Disp-RCNN [24] and ZoomNet [28] utilize extra instance segmentation masks to obtain the object of interest with decent improvements. However, relying on external labels also increases the risk of detection failure when encountering unseen environments.

Efficiency-First Approaches. Approaches that focus on optimizing the system inference latency generally through various lightweight computation architectures and extra 2D/3D detection priors. For example, RTS3D [12] claims to achieve real-time 3D detection with state-of-the-art accuracy. However, the requirement of prior labels from monocular detection makes the utilization of RTS3D still time-consuming. Similarly, *Liu et al.* [13] incorporates the knowledge from real-time monocular 3D detection prior and incrementally detects the 3D bounding boxes with point-wise correlation module. SCNet [23] presents the framework that mainly depends on the predicted semantic key points to restore the 3D bounding boxes. Differently from the above approaches. *Gao et al.* [4] down-scales the feature generalization to speed up the algorithmic latency. *Wang et al.* [26] conduct the replacement of heavy cost volume with lightweight Pseudl-LiDAR feature volume to gain the advance in computing latency.

However, we argue that these existing efficiency-first approaches can only be regarded as rudimentary and incomplete proposals for practical real-time solutions, as none of them have made efforts to optimize their systems from a holistic perspective. A satisfactory solution that meets the requirements of both real-time processing and accurate 3D detection is still lacking. To tackle this challenge, the present work concentrates on a fast-but-inaccurate backbone system and incorporates efficient enhancement techniques to improve detection accuracy. This approach achieves an impressive detection performance that strikes a balance between speed and accuracy, addressing the limitations of previous approaches.

VI. Conclusion

In this paper, we present ER3D, an efficient and real-time 3D object detection system tailored for autonomous driving. To achieve low computing latency, we exploit SGM, an inaccurate-but-efficient depth estimator to produce the dense depth. We further provide an error-oriented decoupled regression head and 3D distance consistency IoU loss to improve detection accuracy caused by an inaccurate depth map. Evaluations on the KITTI dataset indicate ER3D is on par with state-of-the-art approaches on detection accuracy, while is several times faster than its counterparts. The qualitative result on the unseen dataset of ApolloScape also demonstrates the good generalization ability of ER3D.

VII. Acknowledgment

The Kempe Foundation, Sweden.

References

[1] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5410–5418, 2018.

[2] Y. Chen, S. Liu, X. Shen, and J. Jia. Dsgn: Deep stereo geometry network for 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12536–12545, 2020.

[3] X. Cheng, Y. Zhong, M. Harandi, Y. Dai, X. Chang, H. Li, T. Drummond, and Z. Ge. Hierarchical neural architecture search for deep stereo matching. *Advances in Neural Information Processing Systems*, 33:22158–22169, 2020.

[4] A. Gao, Y. Pang, J. Nie, Z. Shao, J. Cao, Y. Guo, and X. Li. Esgn: Efficient stereo geometry network for fast 3d object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 2022.

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

[6] X. Guo, S. Shi, X. Wang, and H. Li. Liga-stereo: Learning lidar geometry aware representations for stereo-based 3d detector. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[7] D. Hernandez-Juarez, A. Chacón, A. Espinosa, D. Vázquez, J. C. Moure, and A. M. López. Embedded real-time stereo estimation via semi-global matching on the gpu. *Procedia Computer Science*, 2016.

[8] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape open dataset for autonomous driving and its application. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[9] J. Ku et al. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8, 2018.

[10] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[11] P. Li, X. Chen, and S. Shen. Stereo r-cnn based 3d object detection for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7644–7652, 2019.

[12] P. Li, S. Su, and H. Zhao. Rts3d: Real-time stereo 3d detection from 4d feature-consistency embedding space for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1930–1939, 2021.

[13] Y. Liu, L. Wang, and M. Liu. Yolostereo3d: A step back to 2d for efficient stereo 3d detection. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[14] X. Ma et al. Delving into localization errors for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[15] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016.

[16] NVIDIA. https://docs.nvidia.com/cuda/cublas/index.html. Accessed Nov. 19, 2022.

[17] R. Qian et al. End-to-end pseudo-lidar for image-based 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[18] Z. Qin, J. Wang, and Y. Lu. Triangulation learning network: from monocular to stereo 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7615–7623, 2019.

[19] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[20] H. Rezatofighi et al. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019.

[21] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47:7–42, 2002.

[22] S. Shi et al. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019.

[23] Y. Shi et al. Stereo centernet-based 3d object detection for autonomous driving. *Neurocomputing*, 2022.

[24] J. Sun, L. Chen, Y. Xie, S. Zhang, Q. Jiang, X. Zhou, and H. Bao. Disp r-cnn: Stereo 3d object detection via shape prior guided instance disparity estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10548–10557, 2020.

[25] Y. Wang et al. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[26] Y. Wang, B. Yang, R. Hu, M. Liang, and R. Urtasun. Plumenet: Efficient 3d object detection from stereo images. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3383–3390. IEEE, 2021.

[27] H. Xu and J. Zhang. Aanet: Adaptive aggregation network for efficient stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1959–1968, 2020.

[28] Z. Xu et al. Zoomnet: Part-aware adaptive zooming neural network for 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[29] Y. You, Y. Wang, W.-L. Chao, D. Garg, G. Pleiss, B. Hariharan, M. Campbell, and K. Q. Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. In *International Conference on Learning Representations (ICLR)*, 2020.

[30] J. Zbontar, Y. LeCun, et al. Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.*, 17(1):2287–2318, 2016.

[31] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *Proceedings of the AAAI conference on artificial intelligence*, 2020.