SPAN: Spatial-Projection Alignment for Monocular 3D Object Detection

Supplementary Material

Content

This Appendix contains the following parts:

- **Detailed Discussion on Projection Constraint**. We present a detailed discussion from both geometric and statistical perspectives, elucidating why the projection constraint holds.
- **Detailed Loss Function**. We delineate the training loss functions in detail, including all components of the 2D and 3D regression losses.
- **Implementation of Hierarchical Task Learning**. We provide a detailed implementation of our Hierarchical Task Learning strategy.
- **Experiments on Waymo Open Dataset**. We demonstrate the superior performance of our proposed method on Waymo Open Dataset.
- **Compared with Prior Geometric Methods**. We compare SPAN with prior geometric approaches, highlighting the advantages of our differentiable soft constraints over hard algebraic solving.
- **Why MGIoU?** We provide ablation study to justify the choice of MGIoU and demonstrate its optimization dynamics.
- **Disentangling HTL & Geometric Alignment Gains**. We conduct experiments to isolate the contributions of HTL and geometric alignment losses.
- **Interplay between Projection Alignment & Depth Bias**. We analyze the interaction between projection alignment and depth bias, showing how they mutually regularize each other.
- **More Qualitative Results**. We present more qualitative results, illustrating that our proposed method achieves better localization for distant objects.

A. Detailed Discussion on Projection Constraint

In Sec. 3.3 we extend the projection constraint originally proposed in Deep3DBox [25], which stipulates that the projected vertices of a 3D bounding box must satisfy the condition that their extreme u and v coordinates coincide exactly with the boundaries of the associated 2D box. We now provide a formal justification for this requirement.

Let S denote the set of points enclosed by a 3D bounding box, and let $\mathbf{P}_n \in S$ be an arbitrary point therein. Under the assumption that the camera intrinsics remain constant, the projection of $\mathbf{P}_n$ onto the 2D image plane is given by $\mathbf{p}_n = (u_n, v_n)$.

$$\begin{aligned} u_n &= f_u \frac{x_n}{z_n} + c_u = f_u f\left(x_n, z_n\right) + c_u, \\ v_n &= f_v \frac{y_n}{z_n} + c_v = f_v f\left(y_n, z_n\right) + c_v, \end{aligned} \tag{13}$$

with (x_n, y_n, z_n) being the coordinates of $\mathbf{P}_n$ in the camera frame. We first aim to demonstrate that the extrema of u_n over the point set S can be attained at the vertices of the 3D bounding box, and the same holds for v_n.

A.1. Problem simplification.

The original claim reduces to demonstrating that the extremum of the function $f\left(x_n, z_n\right)$ over the rectangular domain $\mathcal{D}_{xz}\{(x, z) | x_{\min} \leq x_n \leq x_{\max}, 0 < z_{\min} \leq z_n \leq z_{\max}\}$ is attained at one of its four corner points.

A.2. No interior extrema.

Within the domain $\mathcal{D}_{xz}$, the gradient of $f\left(x_n, z_n\right)$ is expressed as:

$$\nabla f = \left(\frac{\partial f}{\partial x_n}, \frac{\partial f}{\partial z_n}\right) = \left(\frac{1}{z_n}, -\frac{x_n}{z_n^2}\right), \tag{14}$$

since the gradient possesses neither zeros nor singularities anywhere in $\mathcal{D}_{xz}$, $f\left(x_n, z_n\right)$ admits no interior critical points, hence its extrema must occur on the boundary of the domain.

A.3. Boundary extremum analysis.

When z_n is fixed at $z_{\min}$ or $z_{\max}$, f reduces to a linear-fractional function of x_n whose maximum and minimum are attained at $x_{\min}$ and $x_{\max}$. When x_n is fixed at $x_{\min}$ or $x_{\max}$, f reduces to a linear-fractional function of z_n whose maximum and minimum are attained at $z_{\min}$ and $z_{\max}$.

In summary, the extrema of $f(x_n, z_n)$ over the point set S are necessarily attained in at least one of the four line segments $(x_{\min}, y_n, z_{\min})$, $(x_{\min}, y_n, z_{\max})$, $(x_{\max}, y_n, z_{\min})$, and $(x_{\max}, y_n, z_{\max})$. Since the vertices of the 3D bounding box are precisely the endpoints of these segments, the maximum and minimum values of u_n over S are realized at these vertices, the identical argument holds for v_n.

A.4. Perspective projection preserves convexity.

Perspective projection is a convexity-preserving linear mapping in homogeneous coordinates. Consequently, the image of any convex set under projection remains convex:

$$\pi\left(\mathrm{conv}(S)\right) = \mathrm{conv}\left(\pi(S)\right), \tag{15}$$

where π denotes the projection transformation. The 2D point set obtained by projecting S forms a convex polygon, and every point within this polygon can be represented as a convex combination of its vertices. Therefore:

$$u_{\min} \leqslant u_n \leqslant u_{\max}, v_{\min} \leqslant v_n \leqslant v_{\max}. \tag{16}$$

We denote the ground-truth 2D bounding box as $\mathcal{B}^{2D} = \left[u^{2D}_{\min}, u^{2D}_{\max}\right] \times \left[v^{2D}_{\min}, v^{2D}_{\max}\right]$. Since $\mathcal{B}^{2D}$ is the axis-aligned minimal enclosing rectangle of the projected 3D box point set, it follows that:

$$\begin{aligned} u^{2D}_{\min} &= \min u_n = u_{\min}, u^{2D}_{\max} = \max u_n = u_{\max}, \\ v^{2D}_{\min} &= \min v_n = v_{\min}, v^{2D}_{\max} = \max v_n = v_{\max}. \end{aligned} \tag{17}$$

Consequently, the extrema of the projected point set's u and v coordinates lie exactly at the edges of the corresponding 2D bounding box. Since we have already shown that these extrema occur at the vertices of the 3D box, it follows that the projected corners of the 3D box provide the extreme u and v values aligned with the boundary of the 2D box, which completes the proof.

A.5. Statistical analysis.

Fig. 6 illustrates the distribution of pixel deviations between the edges of the minimal enclosing rectangle of the projected ground-truth 3D bounding box corners and the corresponding edges of the 2D ground-truth detection boxes on the KITTI training set. Specifically, we compute the absolute errors $\Delta u_{\min}$, $\Delta u_{\max}$, $\Delta v_{\min}$, and $\Delta v_{\max}$ along the four sides and display their empirical frequency distributions.

All four histograms exhibit a sharp unimodal shape, with the absolute value of each mean below one pixel. Moreover, the worst-case deviation between $v_{\min}$ and $v^{2D}_{\min}$, as well as between $v_{\max}$ and $v^{2D}_{\max}$, is approximately three pixels. These statistical observations corroborate the validity of our adopted projection consistency constraint, whereby the minimum bounding rectangle of the projected 3D box aligns almost perfectly with its 2D counterpart.

B. Detailed Loss Function

For the 2D regression loss $\mathcal{L}_{2D}$, we adopt the focal loss [17] to supervise the learning of object categories, while an L1 loss is employed to constrain the projected 3D center (u_{3d}, v_{3d}) and the dimensions of the 2D bounding box. To impose additional geometric regularization on the 2D box localization, we further incorporate the GIoU loss. Consequently, the 2D regression loss can be formulated as:

$$\mathcal{L}_{2D} = \lambda_1 \mathcal{L}_{class} + \lambda_2 \mathcal{L}_{2Ddim} + \lambda_3 \mathcal{L}_{center} + \lambda_4 \mathcal{L}_{giou}. \tag{18}$$

The 3D regression loss follows the design philosophy of MonoDLE [23], which uses a normalized L1 loss to supervise the recovery of 3D dimensions (h_{3d}, w_{3d}, l_{3d}), while the yaw angler r_y is regularized by a multi-bin classification loss [25]. For depth estimation, we adopt the uncertainty-aware Laplace-based regression loss proposed in MonoDGP [30]:

$$\mathcal{L}_{depth} = \frac{\sqrt{2}}{\sigma_d} \left\| \frac{f \cdot h_{3d}}{h_{2d}} + Z_{err} - Z_{gt} \right\|_1 + \log(\sigma_d), \tag{19}$$

where σ_d denotes the standard deviation of the Laplace distribution. Consequently, the 3D regression loss can be compactly expressed as:

$$\mathcal{L}_{3D} = \lambda_5 \mathcal{L}_{3Ddim} + \lambda_6 \mathcal{L}_{angle} + \lambda_7 \mathcal{L}_{depth}. \tag{20}$$

The object-level depth map loss, denoted as $\mathcal{L}_{dmap}$, employs the focal loss [17] to supervise the classification of foreground depth pixels. The construction of the corresponding ground-truth foreground depth map follows the protocol detailed in MonoDETR [46]. For the region segmentation loss $\mathcal{L}_{region}$, we adopt the Dice loss introduced in MonoDGP [30], which effectively alleviates class-imbalance issues during training:

$$\begin{aligned} \mathcal{L}^i_{region} &= 1 - \frac{2 \cdot \sum_{j=1}^{N_i} p_j \cdot g_j}{\sum_{j=1}^{N_i} p_j + \sum_{j=1}^{N_i} g_j}, \\ \mathcal{L}_{region} &= \sum_{i=1}^{r} \mathcal{L}^i_{region}, \end{aligned} \tag{21}$$

where p_j denotes the predicted probability of the j-th pixel, g_j is its corresponding binary label in the ground truth mask, and N_i indicates the total number of pixels in the i-th feature map f_i, $i = 1, 2, 3, 4$.

C. Implementation of Hierarchical Task Learning

As outlined in Sec. 3.4, we decompose the training procedure of a monocular 3D object detector into a sequence

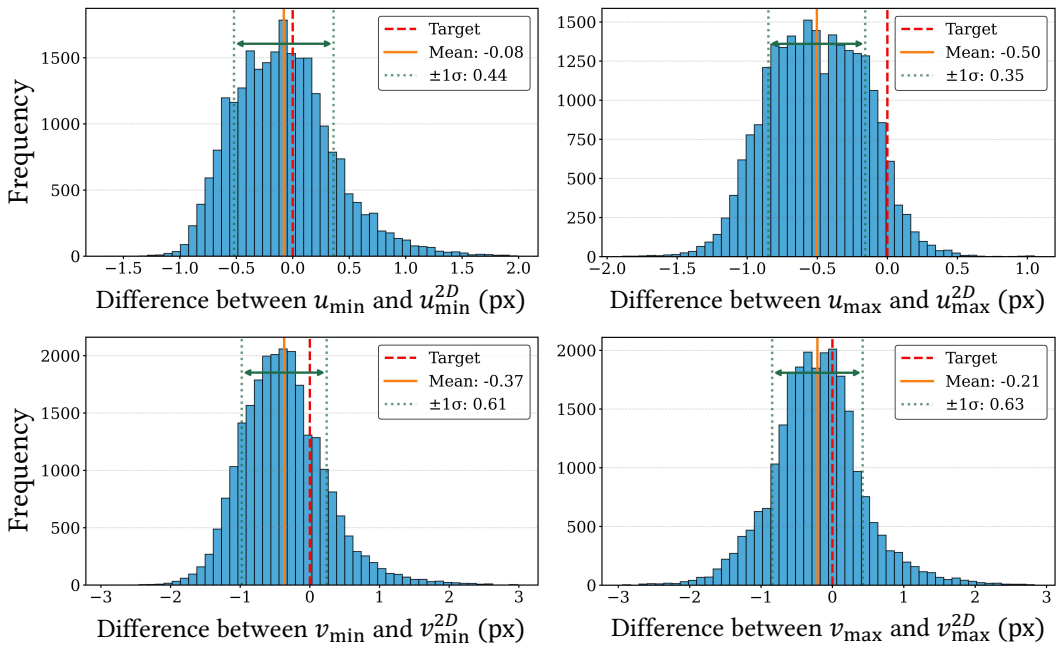

Figure 6. **Deviation statistics.** The data distribution of pixel deviation between the extreme projected coordinates of 3D bounding box corners and the corresponding 2D boundaries on the KITTI training set. Four histograms report the differences between (i) $u_{\min}$ and $u_{\min}^{2D}$, (ii) $u_{\max}$ and $u_{\max}^{2D}$, (iii) $v_{\min}$ and $v_{\min}^{2D}$, and (iv) $v_{\max}$ and $v_{\max}^{2D}$, where $u_{\min}$, $u_{\max}$ and $v_{\min}$, $v_{\max}$ denote the minimal and maximal projected u, v values among the eight 3D box vertices, and $\left[u_{\min}^{2D}, u_{\max}^{2D}\right] \times \left[v_{\min}^{2D}, v_{\max}^{2D}\right]$ denotes the corresponding 2D bounding box.

of pre- and post-dependent tasks. We now elaborate on the refined implementation of the Hierarchical Task Learning (HTL) mechanism, which dynamically modulates the per-epoch weight of each task. Under this scheme, the overall loss can be expressed as:

$$\mathcal{L} = \sum_{i \in \mathcal{T}} \lambda_i \omega_i(t) \mathcal{L}_i, \tag{22}$$

where $\mathcal{T}$ denotes the set of all tasks and $i \in \mathcal{T}$ index of an individual task. Let t be the current training epoch. $\mathcal{L}_i$ represents the loss function of the i-th task, while λ_i is its fixed base weight and $\omega_i(t)$ is the adaptive scaling factor assigned during training.

We adopt a polynomial time-scheduling function to govern the evolution of the dynamic weights:

$$\omega_i(t) = \left(\frac{t}{T_e}\right)^{1-\alpha_i(t)}, \alpha_i(t) \in [0,1], \tag{23}$$

where T_e denotes the total number of training epochs, and the normalized time variable $\frac{t}{T_e}$ automatically rescales the temporal axis. $\alpha_i(t)$ is the adjustment coefficient at epoch t, corresponding to each pre-task for the i-th task.

We now turn to the specification of the scheduling parameter α_i:

$$\alpha_i(t) = \left(\prod_{j \in \mathrm{P}_i} l_{s_j}(t)\right)^{\frac{1}{|\mathrm{P}_i|}}, |\mathrm{P}_i| \geq 1, \tag{24}$$

where P_i denote the set of pre-tasks for task i, and let $l_{s_j} \in [0,1]$ be the learning-status indicator for task $j \in \mathrm{P}_i$. Eqn. 24 implies that α_i attains a high value only when every pre-task has been sufficiently mastered. In contrast to the arithmetic-mean implementation adopted in [21], we employ the geometric mean. The formulation naturally satisfies the boundary conditions (all $l_{s_j} = 1 \Rightarrow \alpha_i = 1$, all $l_{s_j} = 0 \Rightarrow \alpha_i = 0$) and imposes a sharper suppression when any single l_{s_j} deviates from unity. The corresponding gradient flow can be derived as:

$$\frac{\partial \alpha_i}{\partial l_{s_k}} = \frac{\alpha_i}{|\mathrm{P}_i| l_{s_k}} = \frac{\left(\prod_{j \in \mathrm{P}_i} l_{s_j}(t)\right)^{\frac{1}{|\mathrm{P}_i|}}}{|\mathrm{P}_i| l_{s_k}} = \frac{\left(\prod_{j \in \mathrm{P}_i, j \neq k} l_{s_j}(t)\right)^{\frac{1}{|\mathrm{P}_i|}}}{|\mathrm{P}_i| \left(l_{s_k}\right)^{1-\frac{1}{|\mathrm{P}_i|}}}, \tag{25}$$

when a particular prerequisite indicator l_{s_j} is small, the gradient contribution of that term in Eqn. 25 is amplified, thereby providing stronger corrective feedback. This design balances the influence of all prerequisites while mitigating the "single failure to zero" pathology inherent in pure product scheduling.

The learning-status indicator $l_{s_j}(t)$ is computed as:

$$l_{s_j}(t) = \text{clamp}\left(\frac{\mathcal{DF}_j(K) - \mathcal{DF}_j(t)}{\mathcal{DF}_j(K)}, \min = 0, \max = 1\right),$$
$$\mathcal{DF}_j(t) = \frac{1}{K}\sum_{\hat{t}=t-K}^{t-1}\left|\dot{\mathcal{L}}_j(\hat{t})\right|, \tag{26}$$

in Eqn. 26, $\dot{\mathcal{L}}_j(\hat{t})$ denotes the derivative of the loss function $\mathcal{L}_j(\cdot)$ at epoch $\hat{t}$, capturing the instantaneous local trend of task j. $\mathcal{DF}_j(t)$ is defined as the moving average of these derivatives over the most recent K epochs preceding epoch t, thereby summarizing the short-term convergence behavior. When the loss of task j stabilizes, l_{s_j} approaches one, indicating that the prerequisite is reliably learned.

D. Experiments on Waymo Open Dataset

The Waymo Open Dataset [36] is a widely used benchmark for autonomous-driving scene understanding. It encompasses a diverse collection of real-world driving scenes and categorizes objects into LEVEL_1 and LEVEL_2 according to LiDAR point density. The dataset comprises 798 training sequences and 202 validation sequences, yielding approximately 160,000 and 40,000 samples, respectively. For fair comparison, we follow the protocol of [2] and subsample every third frame from both training and validation sequences, resulting in 52,386 training images and 39,848 validation images. During evaluation, we adopt the same settings as DEVIANT [9]. The AP_{3D} metric is reported under two IoU thresholds (0.5 and 0.7) across four distance ranges: Overall, 0 - 30m, 30 - 50m, and 50m - ∞.

As shown in Table 7, our method achieves state-of-the-art performance across all ranges without the use of additional data. These results further demonstrate the effectiveness and generalizability of the proposed SPAN. Notably, the approach utilizing LiDAR outperforms SPAN. We attribute this discrepancy to the higher density of the LiDAR point clouds provided by Waymo, which better compensates for geometric ambiguities, thus leading to a greater performance gain.

E. Compared with Prior Geometric Methods

While sharing geometric roots, SPAN innovates by formulating consistency as a differentiable, single-stage optimization objective, differing from prior works [25, 26]. The results of the comparative experiment are shown in Tab. 8.

Soft Constraints vs. Hard Solving. Deep3DBox [25] uses rigid algebraic solving, where 2D noise propagates to catastrophic 3D failure (-0.81%). SPAN's differentiable soft losses enable the network to learn resilience to 2D jitter, ensuring robustness.

Gradient Refinement vs. Active Regression. Shift R-CNN [26] relies on heavy multi-stage Active Regression (+15ms). SPAN achieves refinement via end-to-end gradient optimization, yielding geometric gains in a single-stage pipeline at zero inference cost.

F. Why MGIoU?

To address questions on MGIoU [10] choice and whether gains are solely from HTL, we conducted a comprehensive ablation. As shown in Tab. 9. MGIoU provides superior optimization dynamics compared to baselines. (i) Unlike L1 constraint which treats corners independently, MGIoU enforces structural volumetric integrity. (ii) Unlike Exact IoU, MGIoU utilizes projection-based approximation to provide non-vanishing gradients for disjoint boxes, ensuring superior convergence. MGIoU outperforms L1 (+0.21%) and Exact IoU (+0.14%). By rigidly coupling depth and dimensions, it effectively mitigates depth bias by preventing the network from exploiting 2D projection ambiguities.

G. Disentangling HTL & Geometric Alignment Gains

As shown in Tab. 10, we conducted an experiment replacing HTL with a simple Linear Weight Schedule to isolate gains. By using a naive Linear Weight Schedule to mitigate initial noise, SPAN achieves 22.95% AP, outperforming the HTL baseline. This confirms that gains stem primarily from the geometric losses. HTL further boosts performance (+0.31%) by adapting to training dynamics, acting as an amplifier for SPAN.

H. Interplay between Projection Alignment & Depth Bias

We analyze the interaction between Projection Alignment (PA) and depth bias.

PA mitigates Depth Bias. To address the inherent scale ambiguity ($z \propto f \cdot H_{3d}/h_{2d}$), PA utilizes reliable 2D boxes as geometric anchors, rectifying depth errors that pure regression permits. Tab. 11 validates that PA reduces long-range Depth MAE, directly contributing to AP_{3D} gains.

Depth Loss with Geometry Prior regularizes PA. Conversely, depth bias loss by MonoDGP [30] prevents PA from converging to degenerate solutions. By penalizing projection violations, it ensures the geometric optimization remains physically consistent.

I. More Qualitative Results

Additional qualitative comparisons on the KITTI validation set are presented in Fig. 7. The proposed method consistently delivers more accurate 3D localization, espe-

Difficulty	Method	Extra Data	AP_{3D}			
			All	0-30	30-50	50-∞
Level_1 ($\text{IoU}_{3D} \geq 0.7$)	CaDDN [31]	LIDAR	5.03	15.54	1.47	0.10
	PCT [37]	Depth	0.89	3.18	0.27	0.07
	GUPNet [21] in [9]	None	2.28	6.15	0.81	0.03
	DEVIANT [9]	None	2.69	6.95	0.99	0.02
	MonoUNI [8]	None	3.20	8.61	0.87	0.13
	MonoDGP [30]	None	4.28	10.24	1.15	0.16
	MonoDGP + SPAN	None	**4.36**	**10.43**	**1.17**	**0.17**
Level_2 ($\text{IoU}_{3D} \geq 0.7$)	CaDDN [31]	LIDAR	4.49	14.50	1.42	0.09
	PCT [37]	Depth	0.66	3.18	0.27	0.07
	GUPNet [21] in [9]	None	2.14	6.13	0.78	0.02
	DEVIANT [9]	None	2.52	6.93	0.95	0.02
	MonoUNI [8]	None	3.04	8.59	0.85	0.12
	MonoDGP [30]	None	4.00	10.20	1.13	0.15
	MonoDGP + SPAN	None	**4.12**	**10.51**	**1.16**	**0.16**
Level_1 ($\text{IoU}_{3D} \geq 0.5$)	CaDDN [31]	LIDAR	17.54	45.00	9.24	0.64
	PCT [37]	Depth	4.20	14.70	1.78	0.39
	GUPNet [21] in [9]	None	10.02	24.78	4.84	0.22
	DEVIANT [9]	None	10.98	26.85	5.13	0.18
	MonoUNI [8]	None	10.98	26.63	4.04	0.57
	MonoDGP [30]	None	12.36	31.12	5.78	1.24
	MonoDGP + SPAN	None	**12.48**	**31.76**	**5.89**	**1.27**
Level_2 ($\text{IoU}_{3D} \geq 0.5$)	CaDDN [31]	LIDAR	16.51	44.87	8.99	0.58
	PCT [37]	Depth	4.03	14.67	1.74	0.36
	GUPNet [21] in [9]	None	9.39	24.69	4.67	0.19
	DEVIANT [9]	None	10.29	26.75	4.95	0.16
	MonoUNI [8]	None	10.38	26.57	3.95	0.53
	MonoDGP [30]	None	11.71	31.02	5.61	1.17
	MonoDGP + SPAN	None	**11.95**	**31.33**	**5.78**	**1.20**

Table 7. Comparisons for the Vehicle Category on the Waymo val set. The best results are highlighted in **bold**, the second-best are underlined, compared with methods without extra data.

Geometric Strategy	Mechanism	Inference Cost (Batch Size of 1)	AP_{3D} (Mod.)
Baseline (MonoDGP [30])	Pure Regression	42ms (single RTX 3090)	22.34
+ Deep3DBox [25]	Hard Algebraic Solving	+5ms	21.53 (-0.81)
+ Shift R-CNN [26]	Multi-stage Active Reg.	+15ms	22.85 (+0.51)
+ SPAN (Ours)	**Soft Loss + HTL**	**+0ms**	**23.26 (+0.92)**

Table 8. Comparison of Geometric Strategies on KITTI val set.

Method Config	HTL	AP_{3D} (Mod.)	Δ
Baseline (MonoDGP)	-	22.34	-
Baseline + HTL Only	✓	22.56	+0.22
Baseline + HTL + $\mathcal{L}_{corner}$ (L1)	✓	22.68	+0.34
Baseline + HTL + $\mathcal{L}_{corner}$ (Exact 3D IoU)	✓	22.75	+0.41
Baseline + HTL + $\mathcal{L}_{corner}$ (MGIoU)	✓	**22.89**	**+0.55**

Table 9. MGIoU performance analysis.

cially for distant and small objects, thereby demonstrating its superior robustness under challenging scale variations.

Method Config	Schedule	AP_{3D} (Mod.)	Δ
Baseline	None	22.34	-
Baseline + SPAN	None	21.85	-0.49
Baseline	HTL	22.56	+0.22
Baseline + SPAN	**Linear Weight Schedule**	**22.95**	**+0.61**
Baseline + SPAN	**HTL**	**23.26**	**+0.92**

Table 10. Decoupling Schedule vs. Geometric Gains.

Method	Depth MAE (m) ↓ by Distance			AP_{3D} (Mod.)
	0-20m	20-40m	40m-∞	
MonoDGP	**0.34**	0.96	1.55	22.34
+ Proj. Align + HTL	0.36 (+0.02)	**0.92** (-0.04)	**1.50** (-0.05)	**22.97**

Table 11. Impact of Projection Alignment on Depth Bias.

(a) MonoDETR (b) MonoDGP (c) MonoDGP (+SPAN)

Figure 7. **More qualitative results on KITTI val set.** (a) MonoDETR [46], (b) MonoDGP [30], and (c) MonoDGP (+SPAN). For each image set, the top row presents the camera-view visualization, while the bottom row offers the corresponding bird's-eye view. Ground-truth bounding boxes are rendered in **green**, and predictions are shown in **red**. We also circle some objects to highlight the difference between the baseline model and our method.