GSDet: Gaussian Splatting for Oriented Object Detection

Zeyu Ding[1,2], **Jiaqi Zhao**[1,2*], **Yong Zhou**[1,2], **Wen-liang Du**[1,2], **Hancheng Zhu**[1,2], **Rui Yao**[1,2]

[1]School of Computer Science and Technology, China University of Mining and Techology

[2]Mine Digitization Engineering Research Center of the Ministry of Education

{dingzeyu, jiaqizhao, yzhou, zhuhancheng, wldu, ruiyao}@cumt.edu.cn

1 Notation

To facilitate clarity in the paper, we present a summary of symbols along with their corresponding descriptions as utilized in this study, encapsulated in Table 9.

Notation	Description
(cx, cy)	center coordinate of object
(w, h)	width and height of object
θ	rotation angles of object
cz	scale of object, or coordicate of z-axis
s	aspect ratio of object
$\boldsymbol{\mu}_{3d}$	mean of 3D Gaussian
$\boldsymbol{\Sigma}_{3d}$	covariance matrix of 3D Gaussian
$\mathcal{N}_{3d}(\boldsymbol{\mu}_{3d}, \boldsymbol{\Sigma}_{3d})$	3D Gaussian distribution
$\mathbf{R}_{3d}$	3D rotation matrix
$\boldsymbol{\Lambda}_{3d}$	3D scaling matrix
$\lambda_1, \lambda_2, \lambda_3$	eigenvalue
σ	coefficient
$\boldsymbol{\mu}_{2d}$	mean of 2D Gaussian
$\boldsymbol{\Sigma}_{2d}$	covariance matrix of 2D Gaussian
$\mathcal{N}_{2d}(\boldsymbol{\mu}_{2d}, \boldsymbol{\Sigma}_{2d})$	2D Gaussian distribution
$\mathbf{R}_{2d}$	2D rotation matrix
$\boldsymbol{\Lambda}_{2d}$	2D scaling matrix
$\{P_2, P_3, P_4, P_5, P_6\}$	feature pyramid
$l \in \{2, 3, 4, 5, 6\}$	level of feature pyramid
$\{F_2, F_3, F_4, F_5, F_6\}$	feature map in 3D feature space
$[W_F, H_F]$	size of $\{F_2, F_3, F_4, F_5, F_6\}$
Z_l	z-axis coordinate of F_l
G^{xy}	sampling area on F_l
(i, j)	index on F_l
G_l^z	Gaussian weight of the z-axis
f_l	sampled features from F_l
f	sampled features from 3D feature space
$\mathcal{L}_{gwd}$	loss for Gaussian Wasserstein distance
$\mathcal{L}_{skewiou}$	loss for SkewIou
$\mathcal{L}_{cls}$	loss for classification
$\mathcal{L}$	total loss
$\delta_{gwd}, \delta_{skewiou}, \delta_{cls}$	loss weight
Q, K, V	query, key, value of self-attention
N_{train}	the number of Gaussians during training
N_{eval}	the number of Gaussians during inference

Table 9: The nomenclature with related notations.

2 Method

2.1 3D Gaussian representation

Structure-from-motion points. We discuss why structure-from-motion (SFM) points are not used during initialization. Oriented object detection datasets consist only of 2D RGB images, with each image captured from a single perspective in a unique scene. Unlike datasets used for rendering tasks, these datasets lack images taken from multiple perspectives of the same scene. Consequently, generating SFM points for the oriented object detection task is challenging due to these dataset characteristics.

Random initialization. In the code implementation, the function `random()` easily generates random values. Specifically, it randomly generates (cx, cy, w, h, θ) within the range [0, 1]. These values are then scaled to their actual ranges based on the image size and angle range. Finally, random 3D Gaussians are calculated based on the random values, ensuring that they are confined within the image area. This initialization does not involve learnable neuron parameters.

The design of $\mathbf{R}_{3d}$. The equation for $\mathbf{R}_{3d}$ is provided in Eq. 4. In our setting, the 3D Gaussians rotate only around the z-axis, rather than an arbitrary axis. This restriction is due to the fact that images in oriented object detection are captured from a single perspective.

The design of $\boldsymbol{\Lambda}_{3d}$. The equation for $\boldsymbol{\Lambda}_{3d}$ is provided in Eq. 4. Although λ_3 is not explicitly defined, features closer to cz are assigned greater attention through the Gaussian weight G_l^z in Eq. 10.

Tile-based rasterizer. We discuss why rasterizers are not used during splatting. Each Gaussian represents a potential object, rather than multiple Gaussians jointly determining one. Oriented object detection is an instance-level task, not a pixel-level task, so there is no need to calculate the value of every pixel or consider object occlusion. Our splatting method can be viewed as a simplified version of the rasterizer.

2.2 Architecture

Feature pyramid $\{P_2, P_3, P_4, P_5, P_6\}$. Assume the size of the original input image is $[H_o, W_o, 3]$, where 3 represents the RGB channels. The shapes of $\{P_2, P_3, P_4, P_5, P_6\}$ are as follows: $P_2 \in [\frac{W_o}{4}, \frac{H_o}{4}, 256]$, $P_3 \in [\frac{W_o}{8}, \frac{H_o}{8}, 256]$, $P_4 \in [\frac{W_o}{16}, \frac{H_o}{16}, 256]$, $P_5 \in [\frac{W_o}{32}, \frac{H_o}{32}, 256]$, and $P_6 \in [\frac{W_o}{64}, \frac{H_o}{64}, 256]$.

*Corresponding author

3D feature space. We rescale the feature maps $P_3 \sim P_6$ to the same size as P_2 by interpolation. We define $W_F = \frac{W_o}{4}$ and $H_F = \frac{H_o}{4}$. The shapes of $\{F_2, F_3, F_4, F_5, F_6\}$ are $[W_F, H_F, 256]$. The z-axis coordinates Z_l of each feature map is are as follows: $Z_2 = 0$, $Z_3 = 1$, $Z_4 = 2$, $Z_5 = 3$, $Z_6 = 4$.

3D Gaussian sampling. Assume that N Gaussians are used. The shape of the sampled features f is $[N, 256, 7, 7]$.

To prepare the input for the subsequent module, we compute f_{in} as follows:

$$f_{in} = \texttt{GAP}(f), \tag{13}$$

where $\texttt{GAP}()$ represents global average pooling. The f_{in} has s shape of $[N, 256]$.

Self-attention. The f_{in} serves as the input query, key, and value for the self-attention. The self-attention is calculated as:

$$sa = \texttt{self-attention}(f_{in}), \tag{14}$$

where sa is the result of the self-attention and has a shape of $[N, 256]$.

Dynamic convolution. Dynamic convolution takes f_{in} and sa as inputs, and its output is computed as:

$$f_{out} = \texttt{DynamicConv}(f, sa), \tag{15}$$

where f_{out} represents the result of dynamic convolution, with a shape of $[N, 256]$.

Feedforward network. The Feedforward network (FFN) consists of several simple linear layers. Instead of directly predicting the labels, $\boldsymbol{\mu}'_{3d}$, $\mathbf{R}'_{3d}$ and $\mathbf{\Lambda}'_{3d}$, the FFN predicts the confidence scores, and the offsets $(\Delta cx, \Delta cy, \Delta cz, \Delta s, \Delta\theta)$ of base elements. This approach ensures stable training. The updated elements $(cx', cy', cz', s', \theta')$ for 3D Gaussians are calculated as follows:

$$\begin{aligned} &cx' = cx + \Delta cx \cdot 2^z, \quad cy' = cy + \Delta cy \cdot 2^z, \\ &cz' = cz + \Delta cz, \qquad\quad s' = s + \Delta s, \\ &\theta' = \theta + \Delta\theta. \end{aligned} \tag{16}$$

The attributes of predicted 3D Gaussians, including mean $\boldsymbol{\mu}'_{3d}$, rotation matrix $\mathbf{R}'_{3d}$ and covariance matrix $\mathbf{\Lambda}'_{3d}$, can be easily updated based on $(cx', cy', cz', s', \theta')$.

2.3 Inference

The inference procedure for GSDet is detailed in Algorithm 2. Given input images, the model predicts object classes and oriented bounding boxes (OBBs). The image encoder extracts features only once, while the detection decoder can be reused iteratively.

The adaptive control and dynamic Gaussians in our method are inspired by adaptive density control in rendering tasks [Kerbl *et al.*, 2023]. Both approaches share the common goal of dynamically adjusting Gaussians to improve their representation. In rendering, Gaussians are adjusted to better represent the scene space, while in our method, they are adjusted to more accurately represent oriented objects.

Algorithm 2 GSDet Inference

Input: images
Output: class, obb

$index \leftarrow 0$
$\{P_2, ..., P_6\} \leftarrow$ Encoder(images)
$\{F_2, ..., F_6\} \leftarrow$ 3DFeatureSpace($\{P_2, ..., P_6\}$)
Curr3DGau $\leftarrow$ Random Initialization
while $index <$ iterations **do**
    Result3DGau $\leftarrow$ Decoder($\{F_2, ..., F_6\}$, Curr3DGau)
    Curr3DGau $\leftarrow$ AdaptiveControl(Result3DGau)
    $index = index + 1$
end while
Result2DGau $\leftarrow$ Splatting(Curr3DGau)
class, obb $\leftarrow$Transform(Result2DGau)
return class, obb

3 Experiments and Discussion

DOTA-v1.0 has 15 common categories: plane (PL), baseball diamond (BD), bridge (BR), ground track field (GTF), small vehicle (SV), large vehicle (LV), ship (SH), tennis court (TC), basketball court (BC), storage tank (ST), soccer-ball field (SBF), roundabout (RA), harbor (HA), swimming pool (SP), and helicopter (HC).

DIOR-R has 20 common categories: airplane (APL), airport (APO), baseball field (BF), basketball court (BC), bridge (BR), chimney (CH), expressway service area (ESA), expressway toll station (ETS), dam (DAM), golf field (GF), ground track field (GTF), harbor (HA), overpass (OP), ship (SH), stadium (STA), storage tank (STO), tennis court (TC), train station (TS), vehicle (VE), and windmill (WM).

Main results. All results for the comparison methods are directly taken from the original papers, as shown in Table 1. We retrain Oriented Rep and DCFL on DIOR-R, DOTA-v1.0, and DOTA-v1.5, as presented in Table 2. Additionally, we retrain H2RBox-v2 on DIOR-R in Table 2.

Inference speed. Table 1 demonstrates that the inference speed of GSDet is faster than two-stage and transformer-based methods but slightly slower than one-stage methods. Two-stage methods require additional time for generating proposals via the region proposal network, while transformer-based methods are slowed down by their transformer encoder, which consists of multiple stacked layers. In contrast, one-stage methods avoid consecutive layer structures, resulting in faster inference speeds. However, the simplicity of one-stage methods often leads to lower accuracy. We plan to further study and enhance the inference speed of our method.

Dynamic number of Gaussians. In Table 5, we retrain ARS-DETR [Zeng *et al.*, 2024] using 900 queries with the official code. Apart from increasing the number of queries from 300 to 900, all other modules remain unchanged.

Number of 3D Gaussians during training. In this ablation study, we use the same number of Gaussians during training and inference. Figures 4 and 6 show that GSDet improves the AP_{50} by 2.4% when the number of Gaussians is increased from 500 to 900 during training. In our method, each 3D Gaussian represents a potential object. A sufficient number of 3D Gaussians allows the model to better learn objects.

Why not use 100 or 300 3D Gaussians? Datasets for oriented object detection are highly complex, often containing more than 100 objects in a single image. Using only 100 Gaussians would result in fewer Gaussians than the number of objects in some images. Similarly, 300 Gaussians do not provide sufficient capacity for the model to effectively learn objects.

Detail results of DIOR-R. The AP_{50} for each category in the DIOR-R dataset using ResNet50 in our method is shown in Table 10.

Category	**APL**	**APO**	**BF**	**BC**	**BR**
(900 @ 1)	78.65	59.66	77.58	86.22	48.15
(900 @ 3)	81.07	58.94	78.06	85.91	48.64
Category	**CH**	**DAM**	**ETS**	**ESA**	**GF**
(900 @ 1)	78.65	43.48	72.80	85.62	75.44
(900 @ 3)	77.55	42.51	73.25	86.18	75.04
Categoryy	**GTF**	**HA**	**OP**	**SH**	**STA**
(900 @ 1)	79.12	41.45	60.00	82.16	77.86
(900 @ 3)	78.98	41.29	59.81	87.54	78.43
Categoryy	**STO**	**TC**	**TS**	**VE**	**WM**
(900 @ 1)	69.93	86.59	58.65	55.78	69.22
(900 @ 3)	76.96	87.22	58.06	57.30	68.25

Table 10: Experimental results on **DIOR-R** dataset.

Detail results of DOTA-v1.5. The AP_{50} for each category in the DOTA-v1.5 dataset using ResNet50 in our method is shown in Table 11.

Category	**PL**	**BD**	**BR**	**GTF**
(900 @ 1)	80.40	69.57	49.84	71.10
(900 @ 3)	80.48	72.51	49.11	71.19
Category	**SV**	**LV**	**SH**	**TC**
(900 @ 1)	58.40	77.76	87.83	90.93
(900 @ 3)	57.75	78.96	88.93	90.85
Category	**BC**	**ST**	**SBF**	**RA**
(900 @ 1)	80.56	74.95	56.98	65.96
(900 @ 3)	79.44	77.52	55.77	65.56
Category	**HA**	**SP**	**HC**	**CC**
(900 @ 1)	67.87	70.24	64.31	9.99
(900 @ 3)	71.36	70.40	65.41	10.26

Table 11: Experimental results on **DOTA-v1.5** dataset.

Layers	**mAP**	**AP_{75}**	**AP_{50}**	**Training time**
2	29.30	26.49	54.72	11
4	46.17	49.55	72.78	14
6	**47.77**	**52.15**	**75.44**	17
8	45.50	47.97	72.64	20

Table 12: Effect of **number of decoder layers** on DOTA-v1.0. The performance is the best when using 6 layers.

Number of decoder layers. Table 12 shows the effect of different number of decoder layers. The model achieves an AP_{50} of only 54.72% when 2 layers. We argue that input random Gaussians are hard to well optimized through 2 layers. As we gradually increase the number of decoder layers, the performance is saturated at 6 layers with an AP_{50} of 75.44%, the training time are gradually increase. Considering all factors, 6 decoder layers are set as the default.

More stringent metric. The metric AP_{50} has a large tolerance for angle deviation, while AP_{75} and mAP are more sensitive to such deviations. AP_{75} and mAP are stricter metrics for evaluating performance. AP_{50} reflects the model's detection ability under a looser IoU threshold, and generally, AP_{50} values are higher because lower IoU thresholds are easier to meet. In contrast, AP_{75} evaluates the model's performance under a stricter IoU threshold. A higher AP_{75} indicates that the model can more accurately localize the target, with a greater overlap between the predicted box and the real object. mAP is a comprehensive evaluation metric that reflects the model's detection ability across different levels of strictness. A higher mAP indicates that the model maintains good detection accuracy across various IoU thresholds. We compare our method with state-of-the-art approaches on DOTA-v1.0, as shown in Table 13. Our GSDet achieves an AP_{75} of 52.40% and a mAP of 48.16%, outperforming all compared methods.

Method	**Backbone**	**AP_{50}**	**AP_{75}**	**mAP**
ACM-SkewIoU [Xu *et al.*, 2024]	R50	74.21	42.83	-
O. R-CNN [Xie *et al.*, 2024]	R50	75.87	46.81	44.92
ReDet [Han *et al.*, 2021]	ReR50	76.38	50.83	47.08
COBB [Xiao *et al.*, 2024] + ReDet	ReR50	76.52	51.38	47.67
PSC [Yu and Da, 2023] + Yolov5s [Jocher *et al.*, 2022]	DarkNet	77.32	47.56	46.48
ARC [Pu *et al.*, 2023] + O. R-CNN	ARC	77.35	51.11	47.44
PKINet-T [Cai *et al.*, 2024] + O. R-CNN	PKINet-T	77.87	51.30	47.35
SM3Det [Li *et al.*, 2024]	MoE	**77.88**	48.24	46.47
GSDet (ours)	R50	75.74	**52.40**	**48.16**

Table 13: Comparasion of state-of-the-art methods on **DOTA-v1.0.**

Backbone. Our method focuses on oriented object detection, rather than on designing a specific backbone architecture. Modern backbones, such as those described in [Cai *et al.*, 2024] and [Pu *et al.*, 2023], are orthogonal to our approach. This makes it easy to integrate various backbones into our method. In our experiments, we use two commonly adopted backbones: the CNN-based ResNet [He *et al.*, 2016] and the transformer-based Swin-Transformer [Liu *et al.*, 2021]. Given that these backbones are widely used, we have not conducted additional experiments to investigate the impact of different backbones in detail.

4 Limitation and future work

Various geometric shapes, such as horizontal boxes, 3D boxes, ellipses, and spherical boxes, can be represented by Gaussians and detected through Gaussian splatting. However, more effective initialization methods need to be explored further. Additionally, we plan to improve the speed of model inference.

5 Visualization

We visualize some detection results on the DIOR-R dataset in Figure 9. The prediction results show that GSDet can accurately detect challenging objects, including densely distributed objects such as ships (SH), large-scale objects like ground track fields (GTF), and objects with extreme aspect ratios, such as airports (APO).

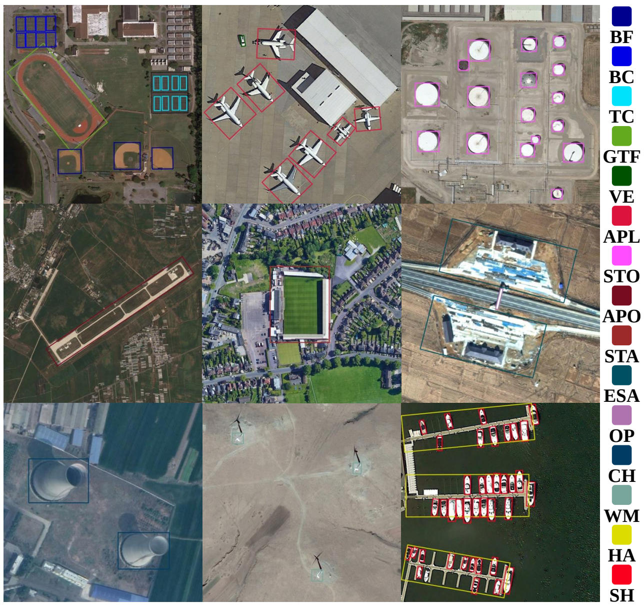

Figure 9: Examples of detection **results on DIOR-R** dataset using our GSDet.

References

[Cai *et al.*, 2024] Xinhao Cai, Qiuxia Lai, Yuwei Wang, Wenguan Wang, Zeren Sun, and Yazhou Yao. Poly kernel inception network for remote sensing detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27706–27716, 2024.

[Han *et al.*, 2021] Jiaming Han, Jian Ding, Nan Xue, and Gui-Song Xia. Redet: A rotation-equivariant detector for aerial object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2786–2795, 2021.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Jocher *et al.*, 2022] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, Yonghye Kwon, Jiacong Fang, Kalen Michael, Diego Montes, Jebastin Nadar, Piotr Skalski, et al. ultralytics/yolov5: v6. 1-tensorrt, tensorflow edge tpu and openvino export and inference. *Zenodo*, 2022.

[Kerbl *et al.*, 2023] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4), July 2023.

[Li *et al.*, 2024] Yuxuan Li, Xiang Li, Yunheng Li, Yicheng Zhang, Yimian Dai, Qibin Hou, Ming-Ming Cheng, and Jian Yang. Sm3det: A unified model for multi-modal remote sensing object detection. *arXiv preprint arXiv:2412.20665*, 2024.

[Liu *et al.*, 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[Pu *et al.*, 2023] Yifan Pu, Yiru Wang, Zhuofan Xia, Yizeng Han, Yulin Wang, Weihao Gan, Zidong Wang, Shiji Song, and Gao Huang. Adaptive rotated convolution for rotated object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6589–6600, 2023.

[Xiao *et al.*, 2024] Zikai Xiao, Guoye Yang, Xue Yang, Taijiang Mu, Junchi Yan, and Shimin Hu. Theoretically achieving continuous representation of oriented bounding boxes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16912–16922, 2024.

[Xie *et al.*, 2024] Xingxing Xie, Gong Cheng, Jiabao Wang, Ke Li, Xiwen Yao, and Junwei Han. Oriented r-cnn and beyond. *International Journal of Computer Vision*, pages 1–23, 2024.

[Xu *et al.*, 2024] Hang Xu, Xinyuan Liu, Haonan Xu, Yike Ma, Zunjie Zhu, Chenggang Yan, and Feng Dai. Rethinking boundary discontinuity problem for oriented object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17406–17415, 2024.

[Yu and Da, 2023] Yi Yu and Feipeng Da. Phase-shifting coder: Predicting accurate orientation in oriented object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13354–13363, 2023.

[Zeng *et al.*, 2024] Ying Zeng, Yushi Chen, Xue Yang, Qingyun Li, and Junchi Yan. Ars-detr: Aspect ratio-sensitive detection transformer for aerial oriented object detection. *IEEE Transactions on Geoscience and Remote Sensing*, 62:1–15, 2024.