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

Abstract

Oriented object detection has advanced with the development of convolutional neural networks (CNNs) and transformers. However, modern detectors still rely on predefined object candidates, such as anchors in CNN-based methods or queries in transformer-based methods, which struggle to capture spatial information effectively. To address the limitations, we propose GSDet, a novel framework that formulates oriented object detection as Gaussian splatting. Specifically, our approach performs detection within a 3D feature space constructed from image features, where 3D Gaussians are employed to represent oriented objects. These 3D Gaussians are projected onto the image plane to form 2D Gaussians, which are then transformed into oriented boxes. Furthermore, we optimize the mean, anisotropic covariance, and confidence scores of these randomly initialized 3D Gaussians, using a decoder that incorporates 3D Gaussian sampling. Moreover, our method exhibits flexibility, enabling adaptive control and a dynamic number of Gaussians during inference. Experiments on 3 datasets indicate that GSDet achieves AP_{50} gains of 0.7% on DIOR-R, 0.3% on DOTA-v1.0, and 0.55% on DOTA-v1.5 when evaluated with adaptive control and outperforms mainstream detectors. Code link https://github.com/wokaikaixinxin/GSDet.

1 Introduction

Oriented object detection aims to identify and locate objects with arbitrary orientations in images [Murrugarra-LLerena *et al.*, 2025; Xie *et al.*, 2024]. Unlike objects in horizontal object detection, oriented objects often appear in various scales, aspect ratios, and orientations, which significantly increases the difficulty of accurate detection [Cheng *et al.*, 2022a; Xia *et al.*, 2018]. These challenges necessitate the development of techniques that can effectively localize and classify oriented objects regardless of their complexity.

Over the years, oriented object detection has evolved significantly, driven by advancements in both CNN-based [Xie *et al.*, 2024; Han *et al.*, 2021] and transformer-based [Zeng *et al.*, 2024] methods. CNN-based object detectors can be broadly categorized into one-stage and two-stage detectors. One-stage detectors [Han *et al.*, 2021] predict object categories and bounding boxes directly from the feature map in a single pass, utilizing dense object candidates like anchors to cover potential object locations. In contrast, two-stage detectors [Xie *et al.*, 2024] first generate region proposals using a region proposal network and then refine these proposals to produce the final results. Recently, transformer-based methods [Zeng *et al.*, 2024] have gained traction in this field, which treat object detection as a set prediction problem and use a set of learnable queries from the encoder to directly predict the final bounding boxes and class labels.

Although CNN-based and transformer-based methods have demonstrated promising performance, they still rely on manually designed object candidates, such as anchors, region proposals, and queries. Moreover, these candidates, which represent potential objects, struggle to effectively capture spatial information of objects.

To address these issues, we propose a novel framework that directly detects oriented objects from random 3D Gaussians.

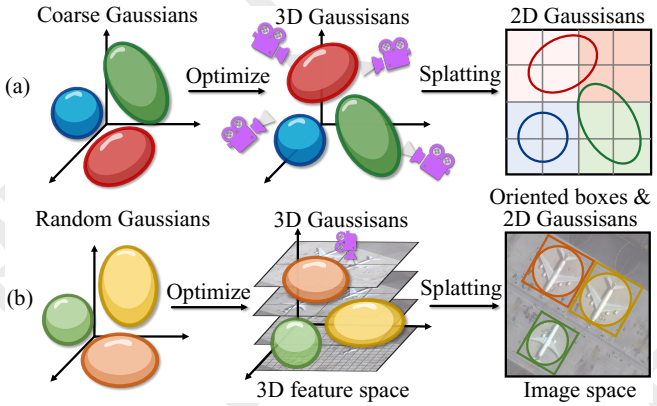

Figure 1: (a) **GS for rendering**. The 3D Gaussians are initialized through SFM or random points and optimized for mean, covariance, opacity, and color. All image pixels are computed through a tile-based rasterizer. (b) **GS for oriented object detection**. Randomly initialized 3D Gaussians are placed in a 3D feature space and trained to optimize mean, covariance, and confidence scores. The 3D Gaussians are then projected onto the image plane to form 2D Gaussians, which are subsequently transformed into oriented boxes.

*Corresponding author

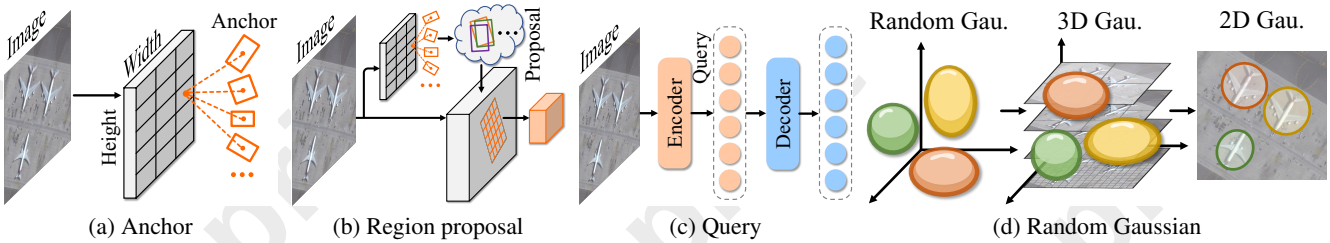

Figure 2: Comparisons of **different schemes for object candidates**. (a) In one-stage CNN-based detectors, dense anchors are enumerated across all image grids. (b) In two-stage CNN-based detectors, region proposals are selected through region proposal networks. (c) In transformer-based detectors, learnable queries are produced by the encoder. (d) In our GSDet, 3D Gaussians are randomly initialized.

The random initialization does not involve learnable neuron parameters. Regarding spatial information, a 3D feature space is constructed from image features. Within this space, the 3D Gaussians are optimized to represent oriented objects and can be projected onto the image plane to form 2D Gaussians, which are then transformed into oriented boxes. This *Gaussian-to-box* method eliminates the need for prior anchors, proposals, or queries, which simplifies the generation of object candidates and establishes a streamlined pipeline.

Our method is inspired by Gaussian splatting (GS), as illustrated in Figure 1. The philosophy of *Gaussian-to-box* paradigm is analogous to the *Gaussian-to-image* process in Gaussian splatting [Kerbl *et al.*, 2023], where a scene is represented by 3D Gaussians and high-quality rendering is achieved by projecting these 3D Gaussians onto the 2D image space. Gaussian splatting has demonstrated significant advancements in 3D scene representation tasks. However, to the best of our knowledge, no prior work has successfully applied Gaussian splatting to oriented object detection.

In this paper, we propose GSDet, a novel approach that formulates oriented object detection as Gaussian splatting. A 3D feature space is constructed by treating multi-scale feature maps from the encoder as supporting planes within a 3D coordinate system. In this space, 3D Gaussians serve as a flexible representation of objects and are initialized randomly, without relying on structure-of-motion (SFM) points. The Gaussian attributes, including the mean, anisotropic covariance, and confidence scores, are optimized using a decoder that incorporates 3D Gaussian sampling to extract spatial features. The predicted 3D Gaussians are projected onto the image plane to form 2D Gaussians, which are further transformed into predicted boxes via decomposition.

Our GSDet exhibits remarkable flexibility during inference. Firstly, it allows for a dynamic number of initial 3D Gaussians as input. We can train the model with a certain number of Gaussians and then use a different number during inference. Secondly, GSDet enables iterative reuse of the decoder to refine the predicted Gaussians through adaptive control, allowing Gaussians to focus on foreground objects.

Experiments on 3 datasets validate the effectiveness of GSDet. Its performance is further improved through the use of adaptive control and an increased number of Gaussians during inference. Contributions are summarized as follows:

- We propose a detector GSDet, which is the first work to formulate oriented object detection as Gaussian splatting to the best knowledge of our knowledge.
- We perform detection within a 3D feature space constructed from image features, where the designed 3D Gaussian sampling captures spatial information.
- Anisotropic 3D Gaussians are employed to represent oriented objects through their attributes. They are projected onto the image plane to form 2D Gaussians, which are then transformed into oriented boxes.
- Our Gaussian-to-box detection paradigm exhibits favorable properties, including adaptive control and dynamic Gaussians during inference, owing to the random initialization of 3D Gaussians.

2 Related Work

Oriented object detection. The mainstream methods in oriented object detection can be broadly categorized into CNN-based and transformer-based approaches. CNN-based methods, which have achieved significant success, are further divided into single-stage and two-stage methods. Two-stage methods [Ding *et al.*, 2019; Xie *et al.*, 2024] utilize a region proposal network (RPN) to generate dense proposals, which are then refined by a region of interest (RoI) head. In contrast, single-stage methods [Hou *et al.*, 2022; Nie and Huang, 2023] bypass the need for region proposals and detect objects directly from dense anchors.

To eliminate the need for dense anchor or proposal candidates, transformer-based methods establish a paradigm based on a set of queries [Zeng *et al.*, 2024; Hu *et al.*, 2023]. These queries represent both the semantic and location information of objects, and are selected through the encoder [Zhu *et al.*, 2020]. Mainstream transformer architectures consist of an encoder and a decoder, with attention mechanisms serving as their core. Self-attention in the decoder enables queries to interact with one another, while cross-attention integrates encoder features with queries.

Unlike these methods, our approach is the first to predict oriented boxes directly from random Gaussians using Gaussian splatting, eliminating the need for pre-designed anchors, proposals, or learned queries, as illustrated in Figure 2.

Gaussian splatting has emerged as a significant technique in computer graphics, offering explicit scene representation and efficient rendering capabilities [Kerbl *et al.*, 2023]. It represents objects and surfaces as collections of Gaussians, allowing for high-quality image synthesis. This method has

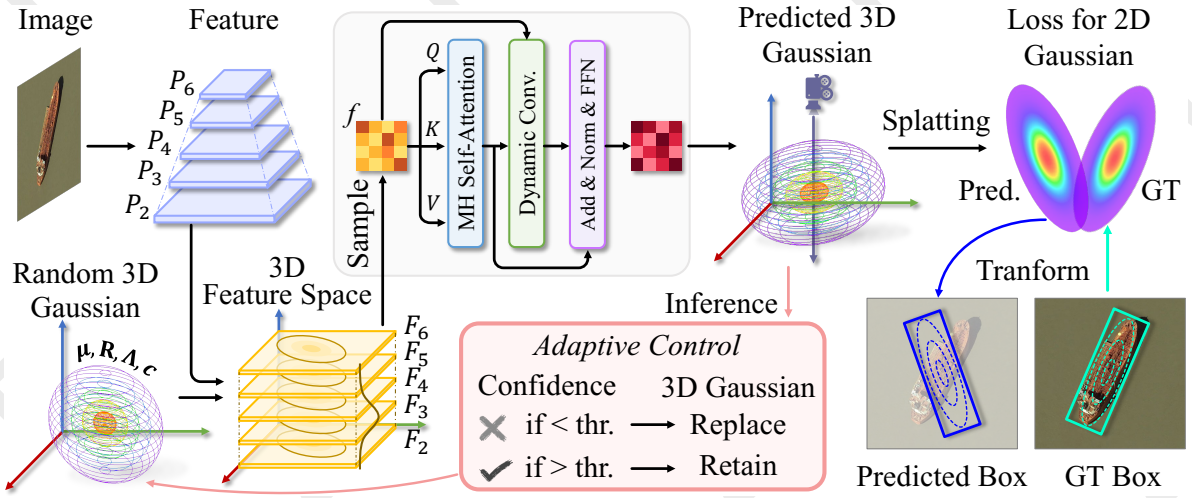

Figure 3: **The overview of GSDet**. 3D Gaussians, defined by their mean, covariance, and confidence scores, are employed to represent objects and are initialized randomly. A 3D feature space is constructed from image features, where 3D Gaussian sampling is designed to capture spatial information. The decoder outputs predicted 3D Gaussians, which are projected into 2D Gaussians, supervised via loss functions, and ultimately transformed into oriented boxes. During inference, adaptive control and a dynamic number of Gaussians are introduced.

been explored in various applications, including scene reconstruction [Yang *et al.*, 2024; Wu *et al.*, 2024], SLAM [Yan *et al.*, 2024; Matsuki *et al.*, 2024], and autonomous driving [Zhou *et al.*, 2024b; Zhou *et al.*, 2024a], demonstrating its versatility and potential for real-time rendering. Despite the success of Gaussian splatting in these domains, there are no previous solutions that successfully adapt Gaussian splatting for oriented object detection. We believe that the slow progress of Gaussian splatting in oriented object detection is primarily due to the different optimization objectives, *i.e.*, Gaussians and oriented boxes. To the best of our knowledge, this is the first work that adopts Gaussian splatting for oriented object detection.

3 Method

Oriented object detection identifies oriented boxes and class labels for objects in images. The core of our GSDet is to address oriented object detection through Gaussian splatting. In this section, we provide a detailed description of our detector. The overview of our GSDet is illustrated in Figure 3.

3.1 Gaussian for Oriented Object

Oriented box. The objects rotate arbitrarily in oriented object detection. Typically, an oriented box is represented as (cx, cy, w, h, θ), where cx and cy denote the center coordinates, w and h represent the width and height, and θ specifies the angle of orientation. To make better use of spatial features, we introduce another representation (cx, cy, cz, s, θ), where cz denotes the scale of objects and s denotes the aspect ratio. The cz and s are calculated as:

$$cz = \log_2(\sqrt{wh}), s = \log_2(\sqrt{\frac{h}{w}}), \tag{1}$$

where wh denote the area, and $\frac{h}{w}$ denote the aspect ratio of objects. The $\log_2(\cdot)$ is used to limit the range of value. Furthermore, the (cx, cy, cz) directly represents the center of a 3D Gaussian in a 3D space, which will be introduced below.

3D Gaussian representation. In our approach, anisotropic 3D Gaussians are initialized randomly. The basic 3D Gaussian $\mathcal{N}_{3d}(\boldsymbol{\mu}_{3d}, \boldsymbol{\Sigma}_{3d})$ is described by its mean $\boldsymbol{\mu}_{3d}$, covariance matrix $\boldsymbol{\Sigma}_{3d}$, and confidence score $\boldsymbol{c}$.

Geometrically, the mean $\boldsymbol{\mu}_{3d}$ represents the center coordinates of ellipsoids, and can be defined as:

$$\boldsymbol{\mu}_{3d} = (cx, cy, cz)^\top. \tag{2}$$

The covariance matrices $\boldsymbol{\Sigma}_{3d}$ have geometric meaning only when they are positive semi-definite. Typically, it is difficult to constrain the learnable model parameters using gradient descent to generate such valid matrices. To avoid invalid matrices during training, we optimize the factorized form of the covariance matrix. The covariance matrix can be factorize into a rotation matrix $\mathbf{R}_{3d}$ and scaling matrix $\boldsymbol{\Lambda}_{3d}$:

$$\boldsymbol{\Sigma}_{3d} = (\mathbf{R}_{3d}\boldsymbol{\Lambda}_{3d})(\mathbf{R}_{3d}\boldsymbol{\Lambda}_{3d})^\top, \tag{3}$$

where the rotation matrix $\mathbf{R}_{3d}$ and scaling matrix $\boldsymbol{\Lambda}_{3d}$ are expressed as:

$$\mathbf{R}_{3d} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}, \boldsymbol{\Lambda}_{3d} = \begin{pmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \end{pmatrix}. \tag{4}$$

The rotation matrix $\mathbf{R}_{3d}$ describes the rotation of the entire Gaussian by angle θ around a specified axis geometrically. In our setting, the 3D Gaussians rotate only around the z-axis, as the image is captured from a single perspective. The scaling matrix $\boldsymbol{\Lambda}_{3d}$ changes the size of a Gaussian by scaling it along the x-, y-, and z-axis. The λ_1, λ_2, and λ_3 are scaling factors in different eigenvector directions and determine how much the Gaussian is stretched or shrunk in each direction. Specifically, the λ_1 and λ_2 are calculated as:

$$\lambda_1 = \frac{1}{2\sigma} \cdot 2^{cz-s}, \ \lambda_2 = \frac{1}{2\sigma} \cdot 2^{cz+s}, \tag{5}$$

where $\sigma = 3$ by default according to the 3-sigma rule. The λ_3 is not explicitly defined, but features closer to cz receive more attention, which will be elaborated in Sec. 3.2.

The confidence score $\boldsymbol{c}$ is used for classification.

Splatting: 3D Gaussian → 2D Gaussian. In oriented object detection, only the original image view is considered, not new viewpoints. Consequently, the 3D Gaussians are projected onto the image plane along the z-axis to produce 2D Gaussians $\mathcal{N}_{2d}(\boldsymbol{\mu}_{2d}, \boldsymbol{\Sigma}_{2d})$. The mean $\boldsymbol{\mu}_{2d}$ is calculated as:

$$\boldsymbol{\mu}_{2d} = (cx, cy)^\top. \tag{6}$$

The covariance $\boldsymbol{\Sigma}_{2d}$ can also be factorized by rotation matrix $\mathbf{R}_{2d}$ and scaling matrix $\boldsymbol{\Lambda}_{2d}$, *i.e.*, $\boldsymbol{\Sigma}_{2d} = (\mathbf{R}_{2d}\boldsymbol{\Lambda}_{2d})(\mathbf{R}_{2d}\boldsymbol{\Lambda}_{2d})^\top$. The $\mathbf{R}_{2d}$ and $\boldsymbol{\Lambda}_{2d}$ can be calculated as:

$$\mathbf{R}_{2d} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \boldsymbol{\Lambda}_{2d} = \begin{pmatrix} \lambda_1 & \\ & \lambda_2 \end{pmatrix}. \tag{7}$$

The confidence score $\boldsymbol{c}$ is consistent between each 2D Gaussian and its corresponding 3D Gaussian.

The time consumption in the splatting process is negligible, as 3D Gaussians can directly achieve orthogonal projection using these concise formulas.

Transformation: 2D Gaussian → Oriented box. Each Gaussian characterizes a potential object in our method, rather than multiple Gaussians jointly determining a potential object. Thus, given the mean $\boldsymbol{\mu}_{2d}$, rotation matrix $\mathbf{R}_{2d}$, and scaling matrix $\boldsymbol{\Lambda}_{2d}$, the oriented boxes as final results are transformed from 2D Gaussians by the reverse process of Eq. 1, Eq. 5, Eq. 6 and Eq. 7 through decomposition.

3.2 Architecture

Our framework consists of an image encoder and a detection decoder, which is a simple paradigm. The image encoder provides 3D feature space and the detection decoder is trained to refine the random 3D Gaussians.

Image encoder. The image encoder is composed of a backbone [He *et al.*, 2016; Liu *et al.*, 2021] and a feature pyramid network (FPN) [Lin *et al.*, 2017a]. The backbone takes in the raw image as its input and extracts image features for the subsequent FPN. The FPN is utilized to generate multi-scale feature maps with 256 channels. Specifically, we construct the pyramid with levels ranging from P_2 to P_6, where P_l has a resolution that is 2^l times lower than that of the image. The $l \in \{2, 3, 4, 5, 6\}$ represents the pyramid level.

3D feature space. Since oriented object detection lacks real depth information, we construct a 3D feature space based on the feature pyramid. The z-axis coordinates Z_l of each feature map are computed as:

$$Z_l = \log_2(2^l/4). \tag{8}$$

We rescale the feature maps $P_3 \sim P_6$ to the same size $[W_F, H_F]$ as P_2 by interpolation. These feature maps, acting as supporting planes, are aligned along the x- and y-axes in the 3D feature space, denoted as $\{F_2, F_3, F_4, F_5, F_6\}$.

3D Gaussian sampling. We implement 3D Gaussian sampling in a decomposition manner. We first sample in the feature maps and then integrate the Gaussian weights of the z-axis. The main sampling area $G^{xy}(i, j)$ based on the 3-sigma rule for each feature map is calculated as follows:

Algorithm 1 GSDet Training

Input: images, GT
Output: loss $\mathcal{L}$

```
{P_2, ..., P_6} ← Encoder(images)
{F_2, ..., F_6} ← 3DFeatureSpace({P_2, ..., P_6})
Initial3DGau ← Random Initialization
Result3DGau ← Decoder({F_2, ..., F_6}, Initial3DGau)
Result2DGau ← Splatting(Result3DGau)
GT2DGau ← Transform(GT)
L ← Loss(Result2DGau, GT2DGau)
return L
```

$$\texttt{sum}(((\sigma\boldsymbol{\Lambda}_{2d})^{-1}\mathbf{R}_{2d}((i,j)^\top - \boldsymbol{\mu}_{2d}))^{\circ 2}) \leq 1, \tag{9}$$

where $(i, j) \in [0, W_F] \times [0, H_F]$. $(\cdot)^{\circ 2}$ denotes squaring each element. $\texttt{sum}(\cdot)$ denotes the summation of all elements. The Gaussian weight on the z-axis is calculated as follows:

$$G_l^z = \frac{\exp(-(Z_l - cz)^2)}{\Sigma_l \exp(-(Z_l - cz)^2)}. \tag{10}$$

The sampling area G^{xy} and weight G_l^z assigns high importance to spatial locations near the center (cx, cy, cz). The 3D Gaussian sampling is calculated as follows:

$$f_l = G_l^z \cdot \texttt{GauAlign}(G^{xy}, F_l), \tag{11}$$

where f_l denotes the sampled features from F_l. Inspired by `RRoIAlign()` [Ding *et al.*, 2019], we introduce `GauAlign()`, which first crops main regional features based on G^{xy} and then resizes them into a uniform size of 7×7 with 256 channels using bilinear interpolation. This operator naturally aligns the oriented Gaussians with axis-aligned features. Last, the sampled features f are obtained by summing f_l from all levels, calculated as $f = \sum_{l=2}^{6} f_l$.

Detection decoder. The decoder stacks multiple layers, each integrating 3D Gaussian sampling, self-attention [Carion *et al.*, 2020], and dynamic convolution [Sun *et al.*, 2021]. The f sampled through 3D Gaussian sampling is passed into self-attention for the relations between Gaussians. Dynamic convolution further enhances feature fusion. Finally, a feed-forward network outputs 3D Gaussians as predictions.

3.3 Training

The training procedure for GSDet is shown in Algorithm 1. Given input images and ground truth (GT), the model is trained to predict both locations and classes of objects.

Loss for 2D Gaussians. We assign multiple predictions to GT and apply the set prediction loss [Carion *et al.*, 2020]. We measure the distance and overlap between the predicted and GT 2D Gaussians. The GT 2D Gaussians are transformed from GT annotations. Specifically, the Gaussian Wasserstein distance (gwd) is employed to quantify the distance [Yang *et al.*, 2021b], and the SkewIoU is used to calculate the overlap [Yang *et al.*, 2023]. Additionally, Focal loss [Lin *et al.*, 2017b] is used for classification. The loss is formulated as:

$$\mathcal{L} = \delta_{gwd}\mathcal{L}_{gwd} + \delta_{skewiou}\mathcal{L}_{skewiou} + \delta_{cls}\mathcal{L}_{cls}, \tag{12}$$

where δ_{gwd}, $\delta_{skewiou}$ and δ_{cls} are weights of each component. We set $\delta_{gwd} = 2$, $\delta_{skewiou} = 5$, and $\delta_{cls} = 2$ by default.

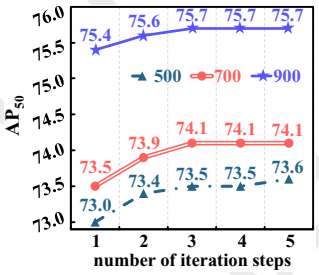

Figure 4: The dynamic number of iterations in **adaptive control** during inference.

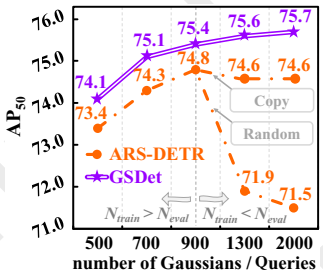

Figure 5: **GSDet vs. ARS-DETR** with dynamic number of Gaussians during inference.

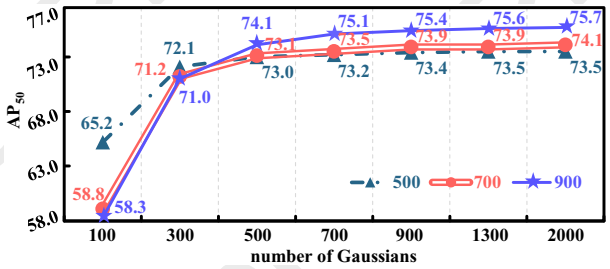

Figure 6: **Dynamic number of Gaussians** during inference.

3.4 Inference

Adaptive control of Gaussian. We can evaluate GSDet with a dynamic number of iterations by reusing the decoder during inference. After each iteration, some predicted 3D Gaussians focus on foreground objects, while others are located in the background. Predicted Gaussians with lower confidence scores are replaced with random Gaussians for the next iteration. Meanwhile, high-confidence Gaussians are retained for final results and then replaced with relatively random Gaussians centered within their areas for the next iteration.

Dynamic number of Gaussian. Thanks to the random initialization, we can evaluate GSDet with a dynamic number of random Gaussians during inference which do not need to be equal to the training stage. In contrast, previous methods rely on the same number of anchors or queries during training and inference, which is inflexible.

4 Experiments

4.1 Experimental Setups

Datasets. We conduct extensive experiments on three datasets DOTA-v1.0 [Xia *et al.*, 2018], DOTA-v1.5 [Xia *et al.*, 2018] and DIOR-R [Cheng *et al.*, 2022a].

- **DOTA-v1.0** [Xia *et al.*, 2018] comprises 1,869 images in the `trainval` set and 937 images in the `test` set, annotated with 188,282 instances across 15 categories.
- **DOTA-v1.5** [Xia *et al.*, 2018] dataset extends the DOTA-v1.0 dataset by adding a new category named "Container Crane" while keeping the same images. The number of instances is increased to 403,318 in total.
- **DIOR-R** [Cheng *et al.*, 2022a] dataset consists of 11,725 training images in the `trainval` set, 11,738 test images in the `test` set and 192,512 instances belonging to 20 categories.

Implementation details. Our code is built on MMrotate with pytorch. The optimizer AdamW [Loshchilov and Hutter, 2018] is used with the learning rate as 2.5×10^{-5} and the weight decay as 10^{-4}. All models are trained with the batchsize 4 on two Nvidia 2080ti (2 images per GPU). The training schedule is 24 epochs, with the learning rate divided by 16 and 22 epochs. Data augmentation strategies contain only random flips. For DOTA-v1.0 and DOTA-v1.5 datasets, images are cropped into patches of 1024×1024 with overlaps of 200. The predictions of DOTA-v1.0/v1.5 are submitted to the official website for evaluation.

Evaluation metrics. The average precision at various intersections over union (IoU) thresholds is utilized to measure the precision and recall rates of the detected objects, including AP_{50}, AP_{75}, and mAP. Additionally, frames per second (FPS) is employed to evaluate the speed of methods.

4.2 Properties of GSDet

Adaptive control. We investigate the effect of adaptive control by increasing the number of iterations from 1 to 5 during inference, as shown in Figure 4. The model is trained with 500, 700, and 900 Gaussians, respectively. As the number of iterations increases from 1 to 5, the model achieves a 0.3% AP_{50} improvement with 900 training Gaussians and a 0.6% AP_{50} improvement with both 500 and 700 training Gaussians.

Dynamic number of Gaussians. Figure 5 compares our GSDet with ARS-DETR [Zeng *et al.*, 2024]. Both models are trained with 900 Gaussians or queries and evaluated using 500, 700, 900, 1300, 2000 Gaussians or queries. To enable ARS-DETR to evaluate with 1300 or 2000 queries, we employ two strategies: cloning existing queries with non-maximum suppression and concatenating random queries. With the former strategy, ARS-DETR experiences a slight decrease in accuracy, whereas the latter strategy results in a significant accuracy drop. In contrast, as the number of Gaussians increases from 500 to 2000 during inference, GSDet achieves a 1.6 AP_{50} improvement.

Additionally, we train GSDet with 500, 700, and 900 Gaussians and evaluate its performance across varying numbers of Gaussians during inference. Figure 6 illustrates that as the number of Gaussians increases from 100 to 2000, the AP_{50} consistently improves, demonstrating the flexibility.

4.3 Main Results

We compare GSDet with modern CNN-based one-stage, two-stage, and transformer-based detectors. We train and evaluate GSDet with 900 Gaussians in both Table 1 and 2. Table 1 presents the detailed results for each category on DOTA-v1.0. GSDet achieves an AP_{50} of 75.44% with one iteration and 75.74% with three iterations using the ResNet50 [He *et al.*, 2016]. Notably, its performance improves further when using the Swin-T [Liu *et al.*, 2021]. Furthermore, with a multi-scale training strategy, GSDet achieves an AP_{50} of 80.04%.

Table 2 provides the AP_{50}, AP_{75}, mAP, and FPS results on DIOR-R, DOTA-v1.0, and DOTA-v1.5. GSDet achieves a AP_{50} of 70.05% on DIOR-R, 75.74% on DOTA-v1.0, and 67.84% on DOTA-v1.5 with three iterations. These results confirm that Gaussian splatting for oriented object detection

	Method	B.	PL	BD	BR	GTF	SV	LV	SH	TC	BC	ST	SBF	RA	HA	SP	HC	AP_{50}
One-Stage	PSC [Yu and Da, 2023]	R50	88.24	74.42	48.63	63.44	79.98	80.76	87.59	90.88	82.02	71.58	59.12	60.78	65.78	71.21	53.06	71.83
	R3Det [Yang *et al.*, 2021a]	R101	88.76	83.09	50.91	67.27	76.23	80.39	86.72	90.78	84.68	83.24	61.98	61.35	66.91	70.63	53.94	73.79
	S2A-Net [Han *et al.*, 2021]	R50	89.11	82.84	48.37	71.11	78.11	78.39	87.25	90.83	84.90	85.64	60.36	62.60	65.26	69.13	57.94	74.12
	H2RBox [Yang *et al.*, 2022]	R50	88.93	78.89	46.27	68.79	81.12	75.45	86.68	90.89	86.71	**87.33**	64.15	68.83	62.81	69.39	59.79	74.40
	DHRec [Nie and Huang, 2023]	R50	88.58	77.90	53.84	72.93	78.45	78.84	87.64	90.88	**88.78**	85.46	56.11	66.74	67.58	70.25	57.53	74.57
	SASM [Hou *et al.*, 2022]	R50	86.42	78.97	52.47	69.84	77.30	75.99	86.72	90.89	82.63	85.66	60.13	68.25	73.98	72.22	62.37	74.92
	FRED [Lee *et al.*, 2024]	Re50	89.37	82.12	50.84	73.89	77.58	77.38	87.51	90.82	86.30	84.25	62.54	65.10	72.65	69.55	63.41	75.56
Two-Stage	SCRDet [Yang *et al.*, 2019]	R101	**89.98**	80.65	52.09	68.36	68.36	60.32	72.41	90.85	87.94	86.86	65.02	66.68	66.25	68.24	65.21	72.61
	RoI Trans. [Ding *et al.*, 2019]	R50	88.65	82.60	52.53	70.87	77.93	76.67	86.87	90.71	83.83	52.81	53.95	67.61	74.67	68.75	61.03	74.61
	Gliding V. [Xu *et al.*, 2020]	R101	89.64	85.00	52.26	77.34	73.01	73.14	86.82	90.74	79.02	86.81	59.55	**70.91**	72.94	70.86	57.32	75.02
	AOPG [Cheng *et al.*, 2022a]	R50	89.27	83.49	52.50	69.97	73.51	82.31	87.95	90.89	87.64	84.71	60.01	66.12	74.19	68.30	57.80	75.24
	DODet [Cheng *et al.*, 2022b]	R50	89.34	84.31	51.39	71.04	79.04	82.86	88.15	**90.90**	86.88	84.91	62.69	67.63	75.47	72.22	45.54	75.49
	O. R-CNN [Xie *et al.*, 2024]	R50	89.46	82.12	54.78	70.86	78.93	83.00	88.20	90.90	87.50	84.68	63.97	67.69	74.94	68.84	52.28	75.87
Query	D. DETR-O [Zhu *et al.*, 2020]	R50	84.89	70.71	46.04	61.92	73.99	78.83	87.71	90.07	77.97	78.41	47.07	54.48	66.87	67.66	55.62	69.48
	EMO2-DETR [Hu *et al.*, 2023]	R50	88.08	77.91	43.17	62.91	74.01	75.09	79.21	90.88	81.50	54.04	51.92	59.44	64.74	71.81	58.96	70.91
	ARS-DETR [Zeng *et al.*, 2024]	R50	86.97	75.56	48.32	69.20	77.92	77.94	87.69	90.50	77.31	82.86	60.28	64.58	74.88	71.76	66.62	74.16
	RQFormer [Zhao *et al.*, 2025]	R50	87.45	78.57	47.36	69.01	79.58	81.27	88.53	90.89	82.80	86.21	58.68	64.20	75.21	74.44	61.39	75.04
	O. Former [Zhao *et al.*, 2024]	R50	88.14	79.13	51.96	67.34	81.02	83.26	88.29	90.90	85.57	86.25	60.84	66.36	73.81	71.23	56.49	75.37
GS	**GSDet (900 @ 1)**	R50	88.65	81.31	51.11	73.48	78.59	82.76	88.17	90.83	84.04	81.35	59.68	62.27	74.68	69.91	64.89	75.44
	GSDet (900 @ 3)	R50	88.57	81.17	51.70	73.19	80.22	83.54	88.64	90.82	84.10	81.59	59.48	64.69	75.08	70.00	63.33	75.74
	GSDet (900 @ 1)	Swin-T	85.05	80.85	52.90	72.31	80.22	83.65	87.58	89.65	83.95	85.65	62.15	62.94	76.28	72.19	61.64	75.80
	GSDet (900 @ 3)	Swin-T	85.89	81.41	52.98	72.52	80.59	83.12	87.96	90.04	83.92	85.86	62.51	63.13	76.53	72.41	61.61	76.03
	GSDet (900 @ 1)*	Swin-T	88.04	**85.55**	**57.22**	**79.44**	**81.28**	**84.77**	**88.73**	90.82	87.11	87.32	**68.64**	68.54	**79.57**	**80.81**	**72.79**	**80.04**

Table 1: Comparison with state-of-the-art methods on **DOTA-v1.0**. * denotes multi-scale training and testing. (900@3) indicates that GSDet is evaluated with 900 Gaussians and 3 iterations in adaptive control. The **best** and second-best results are highlighted in **bold** and underlined.

Type	Method	Backbone	DIOR-R				DOTA-v1.0			DOTA-v1.5		
			AP_{50}	AP_{75}	mAP	FPS	AP_{50}	AP_{75}	mAP	AP_{50}	AP_{75}	mAP
One-stage	Oriented Rep [Li *et al.*, 2022]	R50	66.31	44.36	42.81	29	74.38	46.56	44.57	64.04	38.05	37.43
One-stage	DCFL [Xu *et al.*, 2023]	R50	66.41	41.83	40.88	29	73.13	39.62	41.14	67.00	39.13	38.60
One-stage	H2RBox-v2 [Yu *et al.*, 2023]	R50	57.13	32.13	32.77	**30**	72.60	39.67	40.53	64.71	32.18	34.97
Two-stage	COBB-sig [Xiao *et al.*, 2024]	R50	65.65	42.78	-	14	75.52	48.35	45.61	66.25	41.34	40.04
Query	ARS-DETR [Zeng *et al.*, 2024]	R50	66.12	45.81	43.89	12	74.16	49.41	46.21	64.29	38.23	37.08
GS	**GSDet (900@1)**	R50	69.35	48.27	46.19	28	75.44	52.15	47.77	67.29	42.49	41.11
GS	**GSDet (900@3)**	R50	**70.05**	**48.91**	**46.69**	21	**75.74**	**52.40**	**48.16**	**67.84**	**43.69**	**41.60**

Table 2: Comparisons with different oriented object detectors on **DIOR-R**, **DOTA-v1.0**, and **DOTA-v1.5**.

Metrics	Center	Full	Grid	Isotropic	Random
mAP	5.44	7.61	46.33	46.83	47.77
AP_{75}	4.06	6.13	50.85	51.12	52.15
AP_{50}	11.60	15.11	73.01	73.59	75.44

Table 3: Comparison with other **initialization** strategies on DOTA-v1.0. The random initialization works best.

3D Gau. Represent	3D Gau. Sample	mAP	AP_{75}	AP_{50}
		38.96	39.03	67.01
✓		47.03	50.61	74.01
✓	✓	**47.77**	**52.15**	**75.44**

Table 4: The effect of **3D Gaussian representation** and **sampling** on DOTA-v1.0.

Self-Attention	Dynamic Conv.	mAP	AP_{75}	AP_{50}
		33.24	32.07	61.91
✓		37.87	38.41	66.21
✓	✓	**47.77**	**52.15**	**75.44**

Table 5: The effect of **self-attention** and **dynamic convolution** on DOTA-v1.0.

outperforms both CNN-based and transformer-based methods. Additionally, the inference speed of GSDet is faster than that of two-stage and transformer-based methods owing to random initialization and simple architecture.

4.4 Ablation Study

Gaussian initialization. We study the effect of different Gaussian initialization strategies in Table 3, including: (1) "Center" means all Gaussians are located in the central region, with a height and width set to 10% of the image size; (2) "Full" indicates that all Gaussians are initialized to cover the entire image; (3) "Grid" means that each Gaussian is distributed within each grid, with a total of 30×30 grids arranged across the image; (4) "Isotropic" signifies that all Gaussians are isotropic and randomly initialized. The performance of the "Center" and "Full" strategies is notably poor, with AP_{50} of only 11.60% and 15.11%, respectively. We argue that Gaussians under these two strategies can not effectively cover potential object regions. In contrast, the "Random" initialization achieves the best performance, with an AP_{50} of 75.44%, followed by the "Isotropic" and "Grid" strategies. The results indicate that anisotropic random Gaussians are more likely to align with potential object areas.

3D Gaussian representation and sampling. We evaluate the impact of 3D Gaussian representation and 3D Gaussian sampling in Table 4. This ablation begins with an additional angle branch [Yang and Yan, 2020] and a set of randomly

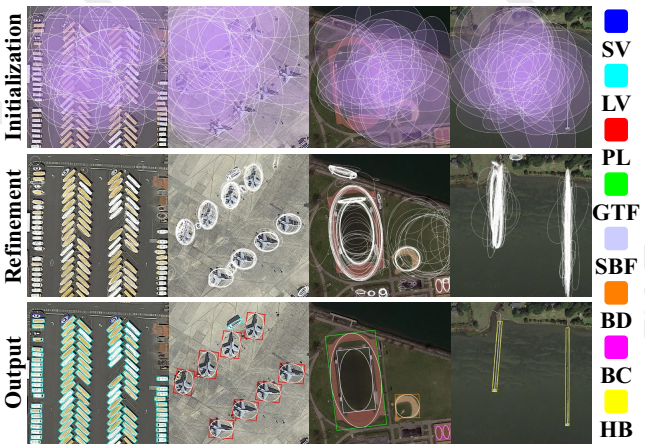

Figure 7: **Top:** random Gaussians. Only a small number of random 2D Gaussians are shown for better visualization. **Middle:** refined Gaussians from the middle decoder layer focus on foreground objects. **Bottom:** final Gaussians and predicted results.

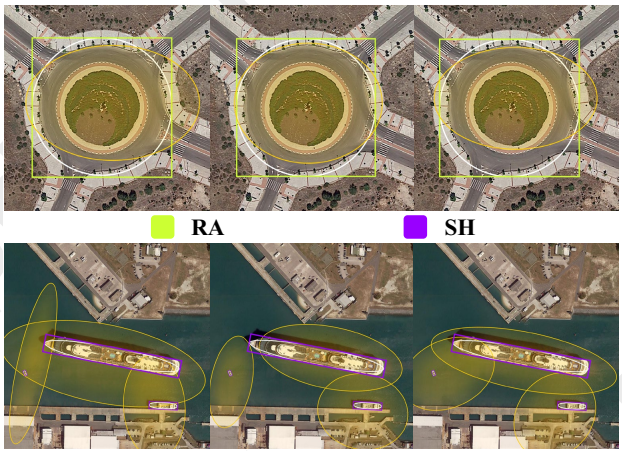

Figure 8: Different randomly initialized Gaussians with a yellow color and corresponding predicted results.

Method	AP	AP_{75}	AP_{50}
Cross-Attention	38.22	38.42	69.29
Dynamic Conv.	**47.77**	**52.15**	**75.44**

Table 6: The comparison between **cross-attention** and **dynamic conv.** on DOTA-v1.0.

Method	AP	AP_{75}	AP_{50}
Feature pyramid	45.13	47.17	73.21
3D feature space	**47.77**	**52.15**	**75.44**

Table 7: The comparison between **feature pyramid** and **3D feature space** on DOTA-v1.0.

High-conf.	Low-conf.		AP_{50}	FPS		AP_{50}	FPS
		1 Iter.	75.44	25	3 Iter.	74.78	18
✓			75.44	25		75.53	18
	✓		75.44	25		75.65	18
✓	✓		75.44	25		75.74	18

Table 8: The effect of Gaussian update strategy in **adaptive control**.

initialized rectangles, achieving an AP_{50} of 67.01%. Under these conditions, learning object orientation proves challenging. Incorporating the 3D Gaussian representation results in a 7% improvement, demonstrating the feasibility of Gaussian splatting. Further applying 3D Gaussian sampling results in a 1.43% improvement, confirming that 3D Gaussian sampling captures spatial information more effectively than relying solely on `GauAlign()`.

Self-attention and dynamic convolution. We check the effect of self-attention and dynamic convolution in the decoder, as shown in Table 5. Our method achieves an AP_{50} of 61.91% without self-attention and dynamic convolution. Sequentially incorporating self-attention results in AP_{50} of 66.21%. When both of them are applied, the model achieves an AP_{50} of 75.44%. Additionally, we compare an alternative implementation, replacing dynamic convolution with cross-attention, as shown in Table 6. The performance degrades with cross-attention. We argue that the input keys and values here are not processed by a standard transformer encoder.

3D feature space. We compare the effect of the 3D feature space with the feature pyramid $\{P_2, \ldots, P_6\}$, as shown in Table 7. In this ablation study, we represent oriented objects with 2D Gaussians and employ `RRoIAlign()` to extract features from transformed rectangles in $\{P_2, \ldots, P_6\}$. The results indicate a 2.23% drop in AP_{50} when using the pyramid, attributed to the lack of spatial information.

Gaussian update strategy. Table 8 illustrates the effect of the update strategy for predicted high- and low-confidence 3D Gaussians in adaptive control. The results indicate a 0.66% drop in AP_{50} with more iterations, when the output of the current step is directly used as input for the next step. We argue that the output distribution is not learned during training. By gradually applying the two update strategies, our GSDet achieves performance gains with only a slight drop in FPS, which verifies the effectiveness of adaptive control.

Qualitative results. We visualize the Gaussians and predictions in Figure 7. The initial Gaussians vary each time due to randomness. After refinement, the Gaussians focus more effectively on potential object areas. The predictions demonstrate that GSDet accurately detects dense, large scale and aspect ratio objects. In addition, Figure 8 illustrates various randomly initialized Gaussians and their corresponding predictions. We observe that initial Gaussians positioned near objects contribute to the final predictions, and GSDet effectively handles diverse inputs. Interestingly, relatively large-scale Gaussians are able to predict small-sized objects.

5 Conclusion

This paper introduces GSDet, a novel detector that is the first to formulate oriented object detection as Gaussian splatting. By leveraging anisotropic 3D Gaussians for object representation within a 3D feature space, GSDet effectively captures spatial information through Gaussian sampling to optimize Gaussians' attributes. 3D Gaussians are projected onto the image plane to form 2D Gaussians, which are then transformed into oriented boxes. Benefiting from the random initialization, the proposed Gaussian-to-box paradigm eliminates the reliance on pre-designed object candidates, allowing for adaptive control and dynamic number of Gaussians during inference. Extensive experiments validate the superior performance of GSDet over mainstream methods.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62272461, Grant 62172417, and Grant 62277046; in part by the Double First-Class Project of China University of Mining and Technology for Independent Innovation and Social Service under Grant 2022ZZCX06; in part by the Six Talent Peaks Project in Jiangsu Province under Grant 2015-DZXX-010 and Grant 2018-XYDXX-044; and Funded by the Graduate Innovation Program of China University of Mining and Technology and the Postgraduate Research & Practice Innovation Program of Jiangsu Province under Grant 123625313.

References

[Carion *et al.*, 2020] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[Cheng *et al.*, 2022a] Gong Cheng, Jiabao Wang, Ke Li, Xingxing Xie, Chunbo Lang, Yanqing Yao, and Junwei Han. Anchor-free oriented proposal generator for object detection. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–11, 2022.

[Cheng *et al.*, 2022b] Gong Cheng, Yanqing Yao, Shengyang Li, Ke Li, Xingxing Xie, Jiabao Wang, Xiwen Yao, and Junwei Han. Dual-aligned oriented detector. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–11, 2022.

[Ding *et al.*, 2019] Jian Ding, Nan Xue, Yang Long, Gui-Song Xia, and Qikai Lu. Learning roi transformer for oriented object detection in aerial images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2849–2858, 2019.

[Han *et al.*, 2021] Jiaming Han, Jian Ding, Jie Li, and Gui-Song Xia. Align deep features for oriented object detection. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–11, 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.

[Hou *et al.*, 2022] Liping Hou, Ke Lu, Jian Xue, and Yuqiu Li. Shape-adaptive selection and measurement for oriented object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 923–932, 2022.

[Hu *et al.*, 2023] Zibo Hu, Kun Gao, Xiaodian Zhang, Junwei Wang, Hong Wang, Zhijia Yang, Chenrui Li, and Wei Li. Emo2-detr: Efficient-matching oriented object detection with transformers. *IEEE Transactions on Geoscience and Remote Sensing*, 2023.

[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.

[Lee *et al.*, 2024] Chanho Lee, Jinsu Son, Hyounguk Shon, Yunho Jeon, and Junmo Kim. Fred: Towards a full rotation-equivariance in aerial image object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 2883–2891, 2024.

[Li *et al.*, 2022] Wentong Li, Yijie Chen, Kaixuan Hu, and Jianke Zhu. Oriented reppoints for aerial object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1829–1838, 2022.

[Lin *et al.*, 2017a] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[Lin *et al.*, 2017b] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[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.

[Loshchilov and Hutter, 2018] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[Matsuki *et al.*, 2024] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18039–18048, 2024.

[Murrugarra-LLerena *et al.*, 2025] Jeffri Murrugarra-LLerena, Jose Henrique Lima Marques, and Claudio R Jung. Gaucho: Gaussian distributions with cholesky decomposition for oriented object detection. *arXiv preprint arXiv:2502.01565*, 2025.

[Nie and Huang, 2023] Guangtao Nie and Hua Huang. Multi-oriented object detection in aerial images with double horizontal rectangles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4932–4944, 2023.

[Sun *et al.*, 2021] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14454–14463, 2021.

[Wu *et al.*, 2024] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024.

[Xia *et al.*, 2018] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3974–3983, 2018.

[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.*, 2020] Yongchao Xu, Mingtao Fu, Qimeng Wang, Yukang Wang, Kai Chen, Gui-Song Xia, and Xiang Bai. Gliding vertex on the horizontal bounding box for multi-oriented object detection. *IEEE transactions on pattern analysis and machine intelligence*, 43(4):1452–1459, 2020.

[Xu *et al.*, 2023] Chang Xu, Jian Ding, Jinwang Wang, Wen Yang, Huai Yu, Lei Yu, and Gui-Song Xia. Dynamic coarse-to-fine learning for oriented tiny object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7318–7328, 2023.

[Yan *et al.*, 2024] Chi Yan, Delin Qu, Dan Xu, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. Gs-slam: Dense visual slam with 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19595–19604, 2024.

[Yang and Yan, 2020] Xue Yang and Junchi Yan. Arbitrary-oriented object detection with circular smooth label. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII 16*, pages 677–694. Springer, 2020.

[Yang *et al.*, 2019] Xue Yang, Jirui Yang, Junchi Yan, Yue Zhang, Tengfei Zhang, Zhi Guo, Xian Sun, and Kun Fu. Scrdet: Towards more robust detection for small, cluttered and rotated objects. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8232–8241, 2019.

[Yang *et al.*, 2021a] Xue Yang, Junchi Yan, Ziming Feng, and Tao He. R3det: Refined single-stage detector with feature refinement for rotating object. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 3163–3171, 2021.

[Yang *et al.*, 2021b] Xue Yang, Junchi Yan, Qi Ming, Wentao Wang, Xiaopeng Zhang, and Qi Tian. Rethinking rotated object detection with gaussian wasserstein distance loss. In *International conference on machine learning*, pages 11830–11841. PMLR, 2021.

[Yang *et al.*, 2022] Xue Yang, Gefan Zhang, Wentong Li, Yue Zhou, Xuehui Wang, and Junchi Yan. H2rbox: Horizontal box annotation is all you need for oriented object detection. In *The Eleventh International Conference on Learning Representations*, 2022.

[Yang *et al.*, 2023] Xue Yang, Yue Zhou, Gefan Zhang, Jirui Yang, Wentao Wang, Junchi Yan, Xiaopeng Zhang, and Qi Tian. The kfiou loss for rotated object detection, 2023.

[Yang *et al.*, 2024] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20331–20341, 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.

[Yu *et al.*, 2023] Yi Yu, Xue Yang, Qingyun Li, Yue Zhou, Feipeng Da, and Junchi Yan. H2rbox-v2: Incorporating symmetry for boosting horizontal box supervised oriented object detection. In *Thirty-seventh Conference on Neural Information Processing Systems*, 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.

[Zhao *et al.*, 2024] Jiaqi Zhao, Zeyu Ding, Yong Zhou, Hancheng Zhu, Wen-Liang Du, Rui Yao, and Abdulmotaleb El Saddik. Orientedformer: An end-to-end transformer-based oriented object detector in remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 62:1–16, 2024.

[Zhao *et al.*, 2025] Jiaqi Zhao, Zeyu Ding, Yong Zhou, Hancheng Zhu, Wen-Liang Du, Rui Yao, and Abdulmotaleb El Saddik. Rqformer: Rotated query transformer for end-to-end oriented object detection. *Expert Systems with Applications*, 266:126034, 2025.

[Zhou *et al.*, 2024a] Hongyu Zhou, Jiahao Shao, Lu Xu, Dongfeng Bai, Weichao Qiu, Bingbing Liu, Yue Wang, Andreas Geiger, and Yiyi Liao. Hugs: Holistic urban 3d scene understanding via gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21336–21345, 2024.

[Zhou *et al.*, 2024b] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21634–21643, 2024.

[Zhu *et al.*, 2020] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2020.