Unbiased IoU for Spherical Image Object Detection

Feng Dai,[1] Bin Chen,[2,1] Hang Xu,[3] Yike Ma,[1] Xiaodong Li,[4] Bailan Feng,[4]
Peng Yuan,[4] Chenggang Yan,[3] Qiang Zhao[1]*

[1]Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
[3]Hangzhou Dianzi University, Hangzhou, China, [4]Huawei Noah's Ark Lab

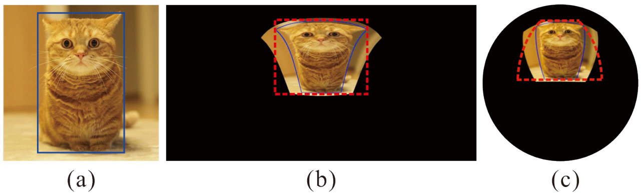
Abstract

As one of the fundamental components of object detection, intersection-over-union (IoU) calculations between two bounding boxes play an important role in samples selection, NMS operation and evaluation of object detection algorithms. This procedure is well-defined and solved for planar images, while it is challenging for spherical ones. Some existing methods utilize planar bounding boxes to represent spherical objects. However, they are biased due to the distortions of spherical objects. Others use spherical rectangles as unbiased representations, but they adopt excessive approximate algorithms when computing the IoU. In this paper, we propose an unbiased IoU as a novel evaluation criterion for spherical image object detection, which is based on the unbiased representations and utilize unbiased analytical method for IoU calculation. This is the first time that the absolutely accurate IoU calculation is applied to the evaluation criterion, thus object detection algorithms can be correctly evaluated for spherical images. With the unbiased representation and calculation, we also present Spherical CenterNet, an anchor free object detection algorithm for spherical images. The experiments show that our unbiased IoU gives accurate results and the proposed Spherical CenterNet achieves better performance on one real-world and two synthetic spherical object detection datasets than existing methods.

Introduction

Due to the development of numerous panoramic cameras in recent years, spherical multimedia data are widely used in virtual navigation, cultural heritage and entertainment industry (Anguelov et al. 2010), which also facilitate the progress in the panoramic research (Zhang et al. 2019; Gu, Sun, and Xu 2020; Shen et al. 2021). With the growing amount of these new types of data, it is also required to detect objects in spherical images for better understanding their contents. For example, Hu et al. treat detected foreground objects as targets to be followed in 360° piloting (Hu et al. 2017).

In the literature, a lot of planar image object detection algorithms (Ren et al. 2017; Zhou, Wang, and Krähenbühl 2019) have been proposed and this field has achieved significant breakthroughs. Among these algorithms, intersection-over-union (IoU) is always indispensable for the different network training or inference phrases. However, although it can be defined and calculated easily for planar image object detection task, it is challenging and unsolved for spherical ones, which leads to comparatively limited existing studies for spherical image object detection. Some existing works directly use biased representations, such as axis-aligned rectangles or circles as representations for objects and check whether a predicted detection is correct based on the IoU between two bounding boxes in unrolled spherical images (Yang et al. 2018; Wang and Lai 2019; Lee et al. 2019). These works would give large errors due to the distortions of spherical objects, which have made these evaluation criteria adopted in planar object detection, shown in Figure 1 (a), is not suitable because axis-aligned rectangles can not tightly bound objects in unrolled spherical images as shown in Figure 1 (b). Other works are based on the unbiased representations but utilize the biased IoU calculations. These works either use the rectangles on tangent planes of sphere as bounding boxes (Su and Grauman 2017; Coors, Condurache, and Geiger 2018) or represent each object using a spherical rectangle on the sphere (Chou et al. 2020; Zhao et al. 2020). Nevertheless, when computing the IoU, they adopt excessive approximate compromises for simplicity of computation, such as considering the spherical rectangles as part of spherical zones as shown in Figure 1 (c), which would give incorrect results. In a nutshell, none of existing methods gives both *unbiased* representations and *unbiased* IoU calculations for spherical image object detection.



(a) (b) (c)

Figure 1: (a) Axis-aligned rectangle used in planar image, (b) axis-aligned rectangle used as biased representation in unrolled spherical image, (c) part of spherical zone used in biased approximate calculation on the sphere.

In this paper, we propose an unbiased IoU as a new evaluation criterion for spherical image object detection task,

*Corresponding author: qiangzhao@ieee.org. This work was done when Qiang was at ICT.

which is the first to use both the spherical rectangles as unbiased representations and the unbiased IoU calculations by spherical geometry. Different from the existing evaluation criteria, our unbiased IoU is absolutely accurate and it does not make any *approximations* for spherical image object detection. Meanwhile, our IoU calculation is fast and does not depend on the resolution of unrolled spherical images due to the form of analytical solutions detailed in the following sections. Based on the new representation and calculation, we also propose an anchor-free object detection algorithm for spherical images, which simply resembles the idea of CenterNet (Zhou, Wang, and Krähenbühl 2019), but explicitly considers the geometry for spherical images. Specifically, we revisit the ground truth generation and loss function design for spherical case. We also replace the traditional convolutional layers with distortion aware spherical convolutional layers. For evaluation, we carry out experiments on three spherical datasets, including one real-world dataset and two synthetic datasets. It shows that our method can get better performance than other baseline methods.

Related Work

Planer Image Object Detection. There are numerous object detection algorithms for planar images, which can facilitate the development of other computer vision tasks (Yan et al. 2020). The readers are referred to (Liu et al. 2020) on object detection for a good survey. Here we only briefly introduce some representative ones. Faster R-CNN framework (Ren et al. 2017) is composed of two modules, where the first module proposes regions by CNNs and the second module is the detector (Girshick 2015) for classification and regression. YOLO (Redmon et al. 2016) frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. SSD (Liu et al. 2016) predicts category scores and box offsets for a fixed set of default boxes applied to multi-scale feature maps. CornerNet (Law and Deng 2018) and CenterNet (Zhou, Wang, and Krähenbühl 2019) eliminates the need for designing common anchors and uses keypoints estimation to detect objects.

Spherical Image Object Detection. Object detection for spherical images is challenging due to the image distortions. (Yang et al. 2018) transforms spherical image into four subimages through stereographic projection, then YOLO is applied on each sub-image for object detection. (Wang and Lai 2019) applies a multi-kernel layer in Faster R-CNN to alleviate image distortions and adds position information to detect spherical objects. (Su and Grauman 2017) projects the feature maps extracted by spherical convolutional layers to the tangent plane, and applies the planar proposal network for object detection. (Coors, Condurache, and Geiger 2018) proposes the spherical single shot multi-box detector to spherical images. (Lee et al. 2019) performs the vehicle detection architecture on spherical polyhedron representation of panoramic images. (Zhao et al. 2020) proposes a two-stage 360° object detector, Reprojection R-CNN. The first stage proposes coarse detections on spherical images, and the second refines proposals by applying another planar detector.

Spherical Object Detection Dataset. There are several types of methods to prepare dataset for spherical object detection. The first type of methods transforms planar datasets and annotations to panoramic ones (Su and Grauman 2017; Zhao et al. 2020). The second type of methods composites real world background spherical images with rendered images (Coors, Condurache, and Geiger 2018) or segmented images (Zhao et al. 2020). The last type of methods captures spherical images and manually annotate the objects (Coors, Condurache, and Geiger 2018; Yang et al. 2018; Yu and Ji 2019). Particularly, the 360-Indoor dataset (Chou et al. 2020), which consists of complex indoor objects, is a new benchmark for object detection in 360° spherical images.

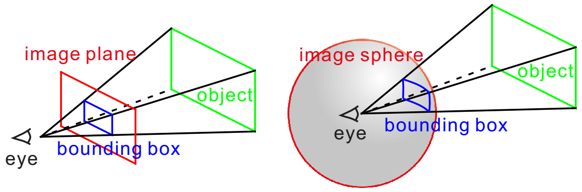


Figure 2: The bounding box of objects in planar image and spherical image. Please see text for detailed discussion.

Unbiased Spherical IoU

In this section, we first illustrate that the spherical rectangles are natural representations for spherical objects, and then explain why existing criteria are biased. Finally, we introduce our unbiased IoU for spherical image object detection.

Unbiased Bounding Box Representation

For the evaluation of generic object detection algorithms, one of the most important problem is how to represent the objects in images. In planar case, the spatial location and extent of an object are usually defined coarsely using an axis-aligned rectangle (x, y, w, h) tightly bounding the object, where (x, y) is the center point and (w, h) is the width and height. The rectangle is formed by the intersection between the image plane and the four surrounding faces of the viewing frustum as shown in Figue 2. By making an analogy, we think that the bounding box for an object on spherical image is formed by the intersection between the image sphere and the viewing frustum. As the four planes corresponding to the faces of the viewing frustum all pass through the center point of the sphere, thus the bounding box is a spherical rectangle. In this paper, we use $(\theta, \phi, \alpha, \beta)$ to denote a spherical rectangle, where θ is the azimuthal angle, ϕ is the polar angle, α and β is the horizontal and vertical field of view respectively. Although we can also use the lengths of the great-circle arcs, it is not convenient for the following IoU calculation.

Existing Biased Evaluation Criteria

Some existing evaluation criteria use biased bounding boxes to represent spherical rectangles in unrolled spherical images. The works in (Yang et al. 2018) and (Wang and Lai 2019) use axis-aligned *rectangles* to represent objects in spherical images as shown in Figure 3 (a). *Circles* are used to represent spherical objects in (Lee et al. 2019) as shown

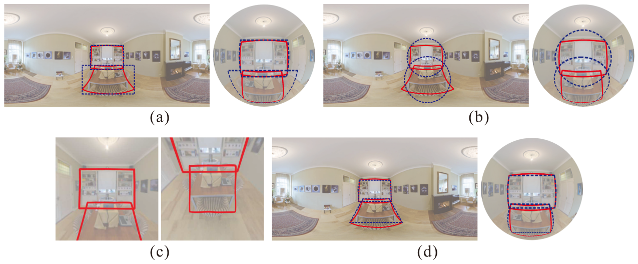


Figure 3: Existing evaluation criteria for spherical image object detection: (a) using axis-aligned rectangles on spherical image, (b) using circles on spherical image, (c) using axis-aligned rectangles on tangent planes, (d) using axis-aligned rectangles on tangent planes but computing the IoU based on polygons on spherical image. In each sub-figure, red curves are spherical rectangles used in this paper, while blue dotted curves are representations for different evaluation criteria.

in Figure 3 (b). Both of them compute the IoU based on intersection between two rectangles or circles without considering the distortions of unrolled spherical images, thus they use biased representations and have large errors.

The remaining criteria are based on unbiased representations, i.e. spherical rectangles, but use biased approximate calculations. Works in (Su and Grauman 2017; Coors, Condurache, and Geiger 2018) use axis-aligned rectangles on the tangent plane to represent spherical objects. However, it is challenging to compute the IoU, as it is unlikely that the estimated bounding box and the ground truth fall on the same tangent plane as shown in Figure 3 (c). The bounding box of one object would be a rectangle on the tangent plane, while another one is not. To deal with this problem, (Coors, Condurache, and Geiger 2018) samples evenly spaced points along the rectangular bounding boxes on the tangent plane and projects them to spherical image. Then IoU can be computed based on the intersection of two constructed *polygons* as shown in Figure 3 (d). However it is just an approximation and its accuracy is highly dependent on the point sampling density. Recent work *SphIoU* in (Zhao et al. 2020) realizes that the IoU should be calculated directly on sphere, but their solution has made too many approximations. First, they treat spherical rectangles as parts of spherical zones, which is not spherical rectangles but the rectangles on the unrolled spherical images. Second, they assume that the intersection between two spherical rectangles also forms a spherical rectangle, which is excessive and incorrect.

In summary, all existing criteria are unreasonable because of biased representations or biased calculations.

Unbiased IoU Calculation

Given two bounding boxes b_1 and b_2 represented by spherical rectangles $(\theta_1, \phi_1, \alpha_1, \beta_1)$ and $(\theta_2, \phi_2, \alpha_2, \beta_2)$, their IoU can be computed as

$$IoU(b_1, b_2) = \frac{A(b_1 \cap b_2)}{A(b_1 \cup b_2)} = \frac{A(b_1 \cap b_2)}{A(b_1) + A(b_2) - A(b_1 \cap b_2)}, \tag{1}$$

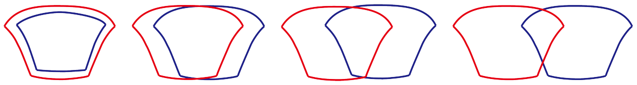


Figure 4: The intersection between two spherical rectangles in unrolled spherical images may have different shapes. From left to right, the intersection is spherical rectangle, 6-sided spherical polygon, 5-sided spherical polygon and 4-sided spherical polygon respectively. Please note that here we only give some example cases and there exist intersections with other shapes.

where $A(\cdot)$ is the area of the shape. Therefore, the IoU calculation can be formulated as the problem that computes the area of each spherical rectangle and the intersection between two spherical rectangles. One direct method for area calculation is using integral by numerical integration (Zhao et al. 2018). However, its accuracy is dependent on the resolution of spherical image, especially for the pixels falling on the boundaries of spherical rectangles. Here, we seek to propose an analytical solution for unbiased IoU calculation. The calculation of the area of each spherical rectangle is relatively easy and it can be obtained by

$$A(b_i) = 4\arccos(-\sin\frac{\alpha_i}{2}\sin\frac{\beta_i}{2}) - 2\pi, \text{for } i \in \{1, 2\}. \tag{2}$$

The derivation is given in the supplementary material.

The calculation of $A(b_1 \cap b_2)$ is very complex. This is because the intersection region may be not a spherical quadrangle, not to mention that it is not a spherical rectangle. We show the complexity in Figure 4. As the boundaries of the intersection $b_1 \cap b_2$ are all great-circle arcs, we can use the following formula (Todhunter 1863) as the most basic mathematical tool to compute the area of intersection

$$A(b_1 \cap b_2) = \sum_{i=1}^{n} \omega_i - (n-2)\pi, \tag{3}$$

if the intersection is n-sided spherical polygon. In the equation, ω_i is the angle of the spherical polygon, which equals to the angle between the planes that adjacent boundaries fall on. Then the core problem becomes determining the number n of boundaries and finding which spherical rectangle each boundary comes from. Although the algorithm proposed in (O'Rourke 1998) may be used to solve the problem, here we introduce a simpler and more robust one.

Our method first checks whether there are no intersection between two spherical rectangles or whether one spherical rectangle is inside of the other. If not, we compute all intersection points between the boundaries of spherical rectangle b_1 and those of b_2 by cross product of normal vectors of boundaries. We remove the intersection points that fall outside of b_1 or b_2 and preserve the real points that the dot products of those and all normal vectors are not less than 0, then the area of intersection between b_1 and b_2 can be computed via Equation 3. The other thing we should consider is that more than two boundaries may intersect at the same point, this case can be easily handled via loop detection, which is

Algorithm 1: Intersection Area Computation

```
Input: Two spherical rectangles $b_1$ and $b_2$ denoted as
       $(\theta_1, \phi_1, \alpha_1, \beta_1)$ and $(\theta_2, \phi_2, \alpha_2, \beta_2)$
Output: the area of intersection $A(b_1 \cap b_2)$
if $b_1 \cap b_2 = \emptyset$ then
    return 0;
end
if $b_1 \subset b_2$ or $b_2 \subset b_1$ then
    return $\min(A(b_1), A(b_2))$;
end
compute the vertices $\mathcal{V}_i$ of spherical rectangle $b_i$;
compute the set $\mathcal{P}$ of intersection points between
  boundaries of $b_1$ and those of $b_2$;
$\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{V}_1 \cup \mathcal{V}_2$;
remove the points $p$ in $\mathcal{P}$ such that $p \notin b_1$ or $p \notin b_2$;
remove duplicated points in $\mathcal{P}$ via loop detection;
for $p_i \in \mathcal{P}$ do
    compute the angle $\omega_i$
end
return $A(b_1 \cap b_2)$ computed via Equation 3;
```

a process of finding a closed loop of normal vectors by DFS algorithm. Finally, we can get the intersection area.

The outline of our method is given in Algorithm 1, which contains a complete computation process of intersection area in our unbiased IoU. More implementation details for this algorithm are given in the supplementary material. We compare our unbiased IoU with existing evaluation criteria in the Experiments to illustrate its plausibility and accuracy.

Spherical CenterNet

In this section, we propose an anchor-free detector based on CenterNet (Zhou, Wang, and Krähenbühl 2019) and make it applicable for the spherical image object detection task.

Network Structure and Loss Definition

Given an unrolled spherical image I, our goal is to predict the center point (θ_i, ϕ_i) and the field of view (α_i, β_i) of bounding box for each object $i \in \{1, 2, \cdots, N\}$. This is accomplished by a convolutional network called Spherical CenterNet shown in Figure 5.

The input spherical image is first processed by a backbone network, whose output is fed into three branches for spherical bounding boxes prediction. The first branch produces a heatmap $p \in [0, 1]^{W \times H \times C}$ for center points of all objects, where $W \times H$ is the size of the heatmap and C is the number of object categories. The score p_{xyc} at location (x, y) for class c indicates the possibility that the point (x, y) is the center point of a spherical object belonging to category c. The second branch predicts local offset $\mathbf{o}_i = (\Delta\theta_i, \Delta\phi_i)$ to slightly adjust the location of center point of each object i. The last branch is used to estimate the field of view $\mathbf{s}_i = (\alpha_i, \beta_i)$ of the spherical bounding box.

Based on the architecture of the above three branches, we design our overall training objective as

$$L = L_{cls} + \lambda_{off} L_{off} + \lambda_{fov} L_{fov}, \tag{4}$$

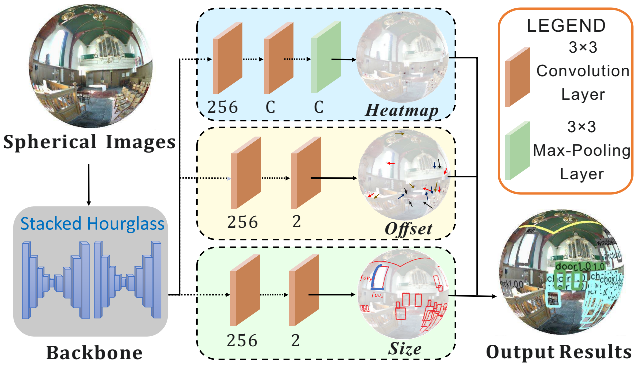


Figure 5: Our network takes spherical images as input and predicts heatmaps, offsets and sizes. With these information, we can determine the spherical rectangle bounding boxes. The number represents the number of filters in each branch.

where L_{cls} is the classification loss, L_{off} and L_{fov} are the regression loss for offset and field of view respectively. λ_{off} and λ_{fov} are the weights for the last two terms. The loss L_{cls} is similar to that of CornerNet (Law and Deng 2018) and CenterNet (Zhou, Wang, and Krähenbühl 2019), and is based on focal loss (Lin et al. 2017)

$$L_{cls} = -\frac{1}{N} \sum_{xyc} w_{xy} \begin{cases} (1 - p_{xyc})^2 \log(p_{xyc}) & \text{if } y_{xyc} = 1, \\ (1 - y_{xyc})^4 (p_{xyc})^2 & \text{otherwise.} \\ \quad \log(1 - p_{xyc}) & \end{cases} \tag{5}$$

where y_{xyc} is the value of ground truth heatmap, whose generation will be described in the following section. A difference is that we introduce a weight for each pixel at location (x, y). The pixels near the polar region, which are more distorted, have smaller weights than the pixels near the equatorial region. The weights w_{xy} are computed based on the surface area of the pixels (Zhao et al. 2018) on unit sphere

$$w_{xy} = \left(\cos \frac{y\pi}{H} - \cos \frac{(y+1)\pi}{H} \right) \frac{2\pi}{W}. \tag{6}$$

As the center points fall on the curved spherical surface, we measure L_{off} with the angle between two 3D unit vectors

$$L_{off} = \frac{1}{N} \sum_i \arccos\left(\langle \mathcal{T}(\mathbf{c}_i + \mathbf{o}_i), \mathcal{T}(\mathbf{c}_i + \hat{\mathbf{o}}_i) \rangle \right), \tag{7}$$

where $\mathbf{c}_i = (\theta_i, \phi_i)$ is the center point of object i, $\hat{\mathbf{o}}_i$ is the ground truth offset, $\mathcal{T}(\cdot)$ is the transformation that converts the azimuthal and polar angle to 3D unit vector in Cartesian coordinate system, $\langle \cdot, \cdot \rangle$ is the dot product of two input vectors. For field of view regression, we simply use the L1 loss

$$L_{fov} = \frac{1}{N} \sum_i |\mathbf{s}_i - \hat{\mathbf{s}}_i|, \tag{8}$$

where $\hat{\mathbf{s}}_i = (\hat{\alpha}_i, \hat{\beta}_i)$ is the ground truth field of view for object i. Please note that we do not incorporate the weights w_{xy} in the design of L_{off} and L_{fov}. This is because the supervisions of these two terms only act at center point locations, while the loss L_{cls} takes sum over all locations.

Implementation Details

Ground Truth Generation. To compute the ground truth offset $\hat{\mathbf{o}}_i$, we first transform the ground truth center point location from azimuthal and polar angle $(\hat{\theta}_i, \hat{\phi}_i)$ to 2D image coordinate $(\frac{\hat{\theta}_i WR}{2\pi}, \frac{\hat{\phi}_i HR}{\pi})$ of the input image, where $WR \times HR$ is the resolution of the input image and R is the downsampling factor. This location is mapped to $(\lfloor \frac{\hat{\theta}_i W}{2\pi} \rfloor, \lfloor \frac{\hat{\phi}_i H}{\pi} \rfloor)$ in predicted heatmap and corresponds to center point with azimuthal and polar angle $(\lfloor \frac{\hat{\theta}_i W}{2\pi} \rfloor \frac{2\pi}{W}, \lfloor \frac{\hat{\phi}_i H}{\pi} \rfloor \frac{\pi}{H})$. Therefore, the ground truth offset is given as

$$\hat{\mathbf{o}}_i = \left(\hat{\theta}_i - \left\lfloor \frac{\hat{\theta}_i W}{2\pi} \right\rfloor \frac{2\pi}{W}, \hat{\phi}_i - \left\lfloor \frac{\hat{\phi}_i H}{\pi} \right\rfloor \frac{\pi}{H} \right). \quad (9)$$

For the generation of ground truth heatmap, we assign non-zero values to negative locations within a radius of positive locations. The radius is determined by ensuring that the locations within the radius would generate a bounding box with at least $t = 0.7$ IoU with the ground truth. Then the ground truth heatmap is given by $\exp\left(-\frac{\arccos(\langle \mathcal{T}(\hat{\theta}_i, \hat{\phi}_i), \mathcal{T}(\theta, \phi) \rangle)}{2\sigma^2} \right)$, where $(\hat{\theta}_i, \hat{\phi}_i)$ is the ground truth positive location, (θ, ϕ) is the negative location within the radius, σ is an adaptive standard deviation depending on the radius. The complex computation details are given in the supplementary material.

Spherical Convolution. Here we use tangent images (Eder et al. 2020) to alleviate distortion problem, which facilitates transferable and scalable 360° computer vision. This representation renders a spherical image to a set of distortion-mitigated, locally-planar image grids tangent to a subdivided icosahedron. Standard CNNs can then be trained on these tangent images. Output feature maps can finally be rendered back to a sphere as feature maps of original spherical image. In our Spherical CenterNet, the heatmap, offset and FOVs are predicted for each tangent image, and then they are rendered back to the sphere for loss computation. Compared with other types of spherical convolution (Zhao et al. 2018; Tateno, Navab, and Tombari 2018; Su and Grauman 2019), we choose this type for two reasons: it keeps the parameter sharing property of convolution; it does not lead to performance degradation if more convolutional layers are added.

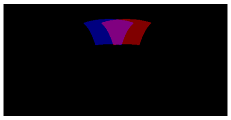
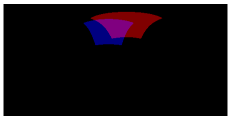
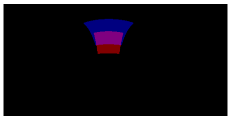
Cases	Methods	IoUs	△
	Sph. Int.	0.32006	-
	Rectangle	0.47163	0.15157
	Polygon	0.35891	0.03885
	Circle	0.24286	0.07720
	SphIoU	0.16537	0.15469
	Ours	**0.31974**	**0.00032**
	Sph. Int.	0.25801	-
	Rectangle	0.55155	0.29354
	Polygon	0.26958	0.01157
	Circle	0.24996	0.00805
	SphIoU	0.11392	0.17109
	Ours	**0.25772**	**0.00029**
	Sph. Int.	0.33966	-
	Rectangle	0.25870	0.08096
	Polygon	0.31526	0.02440
	Circle	0.35992	0.02026
	SphIoU	0.34220	0.00254
	Ours	**0.33935**	**0.00031**

Table 1: The IoUs computed with different methods for three cases. Here spherical integral (Sph. Int.) by numerical integration is taken as the **reference method**. The differences (△) are listed between each method and the reference.

Cases	Methods	$12k \times 6k$	$10k \times 5k$	$8k \times 4k$
	Sph. Int.	0.32006	0.32012	0.32022
	Ours	0.31974	0.31974	0.31974
	△	**0.00032**	**0.00038**	**0.00048**
	Sph. Int.	0.25801	0.25807	0.25816
	Ours	0.25772	0.25772	0.25772
	△	**0.00029**	**0.00035**	**0.00044**
	Sph. Int.	0.33966	0.33972	0.33981
	Ours	0.33935	0.33935	0.33935
	△	**0.00031**	**0.00037**	**0.00046**

Table 2: The IoUs computed with spherical integral (Sph. Int.) and our method for three cases. The differences (△) are given between the two methods. The precision of spherical integral by numerical integration will be degraded if we use unrolled spherical images with lower resolution.

Experimental Result

In this section, we first show that our unbiased IoU is reasonable by comparing it with existing criteria and then compare our Spherical CenterNet with other spherical image object detection methods. Finally, we give ablation studies.

Criteria Comparison

We compare different evaluation criteria and show that our IoU calculation is unbiased through some toy examples. For each example, we randomly set the parameters of the ground truth and the predicted bounding box, and compute the IoU between them with different methods. Here we take the method based on spherical integral by numerical integration (Zhao et al. 2018) as the reference. From Table 1 we can see that the IoUs computed with our method are more close to those computed with the reference method. It is no doubt that the first three methods give incorrect result, as they do not compute the IoU on the sphere and give biased results. SphIoU (Zhao et al. 2020) also gives incorrect result, as it makes too many approximations as mentioned previously.

To validate that the accuracy of spherical integral method depends on the resolution of the unrolled spherical images, we set the spherical image having three different resolutions in each case. As shown in Table 2, with the decrease of the resolution, the accuracy of the spherical integral method

Methods	Backbone	360-Indoor			360-VOC-Gaussian			360-VOC-Uniform		
		AP	AP^{50}	AP^{75}	AP	AP^{50}	AP^{75}	AP	AP^{50}	AP^{75}
CenterNet	ResNet-101	8.6	20.5	5.8	43.3	81.9	40.3	8.3	14.1	8.8
Multi-Kernel	ResNet-101	4.7	11.1	2.8	55.9	77.7	64.8	7.0	12.5	7.3
Sphere-SSD	ResNet-101	2.9	7.8	1.4	21.8	28.4	26.7	11.7	19.2	13.4
Reprojection R-CNN	ResNet-101	5.0	15.3	1.9	53.6	62.2	44.8	9.5	13.8	10.1
Ours	ResNet-101	**10.0**	**24.8**	**6.0**	**65.5**	**84.6**	**75.5**	**15.8**	**21.5**	**18.1**

Table 3: The performance of different methods on 360-Indoor, 360-VOC-Uniform and 360-VOC-Gaussian datasets.

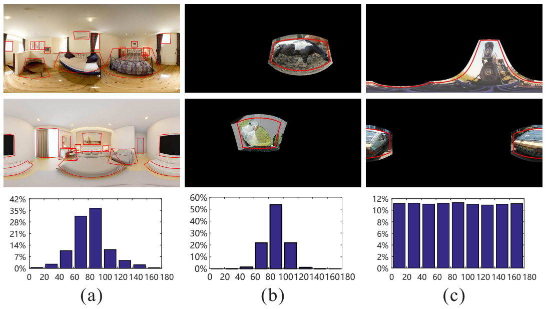


Figure 6: Some example images from 360-Indoor (a), 360-VOC-Gaussian (b) and 360-VOC-Uniform (c) datasets. Here we also plot the distribution of polar angles of the objects in each dataset.

would be degraded significantly. Furthermore, this spherical integral method is also time-consuming, which takes 37.5ms for IoU calculation, while our method is much faster and only needs 0.99ms at the same resolution (1024 × 512).

Comparison with Other Spherical Detectors

Dataset. We conduct the experiments on three datasets, including one real-world dataset *360-Indoor* (Chou et al. 2020) composed of indoor 360° spherical images for object detection, and another two synthetic spherical datasets *360-VOC-Uniform* and *360-VOC-Gaussian*.

360-Indoor is a 360° indoor dataset specially designed for object detection task with 37 categories. This dataset has approximately 3k images and 90k labels in sum, and it uses spherical rectangle $(\theta, \phi, \alpha, \beta)$ for bounding box parameterization. As shown in Figure 6, most of the objects in this dataset are located near the equatorial region.

360-VOC-Gaussian is a synthetic 360° dataset generated from PASCAL VOC 2012 (Everingham et al. 2015). It has 20 categories, and only one object instance is rendered in each spherical image. The objects in 360-VOC-Gaussian are normally distributed, and the size of background is no less than half of the object size if the object is not at the edge of the image, otherwise it is set as a random value. This dataset has 18.6k training images, 6.3k validating images, and 3.1k testing images. Some examples are shown in Figure 6.

360-VOC-Uniform is another synthetic 360° dataset, and the only difference between this dataset and 360-VOC-Gaussian, as shown in Figure 6, is that object instances are located at arbitrary position on the sphere in 360-VOC-Uniform, which is more difficult for spherical detection tasks. Other properties like dataset source, categories, image sizes and so forth, remain the same as 360-VOC-Gaussian.

Baseline Methods. We compare our Spherical CenterNet with three current spherical image object detection methods.

Multi-Kernel (Wang and Lai 2019) applies a multi-kernel layer after ROI Pooling layer in standard Faster R-CNN and incorporates position information of each proposal for object detection in spherical images.

Sphere-SSD (Coors, Condurache, and Geiger 2018) adapts SSD (Liu et al. 2016) to spherical images and defines the anchor boxes based on the tangent planes of the sphere.

Reprojection R-CNN (Zhao et al. 2020) is a two-stage spherical object detector, where the first stage outputs spherical region proposals and the second stage refines the proposals predicted by the first stage.

As our network is based on the architecture of CenterNet, we also take the planar CenterNet (Zhou, Wang, and Krähenbühl 2019) as one of the baseline methods. To make these methods comparable, we set the networks of different methods to have the same backbone.

Metric. We use standard mAP (Everingham et al. 2010) as the evaluation metric for object detection. Please note that as original evaluation metrics used in the baseline methods are biased, we convert bounding boxes they predicted to spherical rectangles and use our unbiased IoU for evaluation.

Training Details. Our method is implemented in PyTorch (Paszke et al. 2017) and 8 GeForce RTX 2080Ti GPUs are used for training with a batch size of 32 (4 images per GPU). We use Adam (Kingma and Ba 2014) to optimize the overall parameters objective for 160 epochs with the initial learning rate 1.25×10^{-4}, and the learning rate is divided by 10 at 90 and 120 epochs. The input resolution of the whole network is 1024×512, which is downsampled $4\times$ through the model. During training, we only use random flip as data augmentation because of the particularity of Equirectangular projection. For the training loss of 360-Indoor dataset, we set $\lambda_{off} = 60$ and $\lambda_{fov} = 10$ to balance the orders of magnitude for each loss term. For the other two 360-VOC-Uniform and 360-VOC-Gaussian datasets, we keep $\lambda_{off} = 1$ and $\lambda_{fov} = 0.1$ in line with the original loss weights because each image only contains one object in these two datasets.

Quantitative Results. The performance of different methods on three datasets are shown in Table 3. From the table, we can see that our method can give the best performance on all three datasets. Compared with the performance on 360-

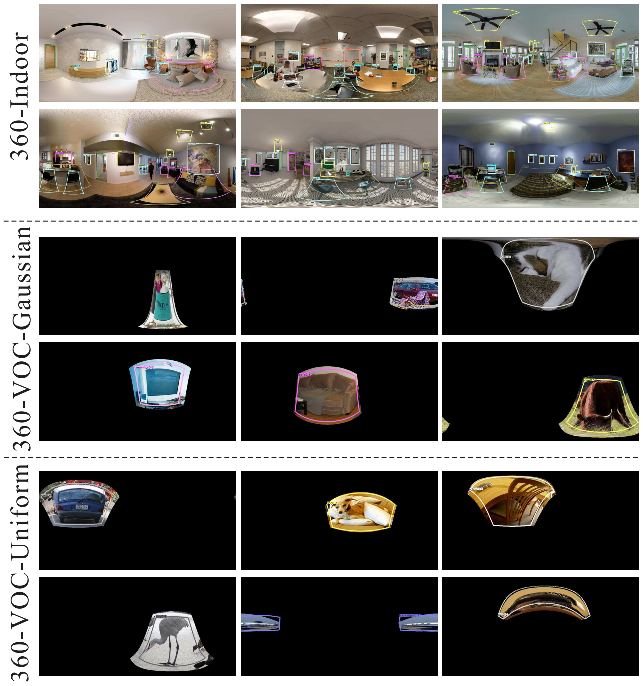


Figure 7: Visual detection results of our method on 360-Indoor, 360-VOC-Gaussian and 360-VOC-Uniform datasets.

VOC-Uniform and 360-Indoor, all methods give higher AP on 360-VOC-Gaussian. This is because only one object remains in 360-VOC-Gaussian and most instances are located at the equator on the sphere, which is consistent with our previous explanation. Meanwhile, recent work Reprojection R-CNN performs worse than our method, this is because the method is trained with the biased IoU. If bounding boxes are located near polar regions, the IoU calculation would give large errors as we discussed before. Although Sphere-SSD gives worse AP due to the biased IoU, its better performance on 360-VOC-Uniform shows that spherical convolutions it adopts to deal with image distortions is profitable, while Multi-Kernel are difficult to solve this problem.

In addition, the other three baselines have worse performances than CenterNet, and the reason is quite likely to be that they are all anchor-based methods, while CenterNet is an anchor-free method. As shown in previous work (Zhang et al. 2020), the performance of anchor-based methods is more easily affected by how to select positive/negative training samples. Based on their biased IoU, incorrect training samples may be selected, which leads to poor performance.

Backbone	Convolution	AP	AP^{50}	AP^{75}
ResNet-101	spherical	10.0	24.8	6.0
Hourglass	spherical	**14.1**	**31.4**	**11.0**
Hourglass	planar	12.7	28.4	9.3
Hourglass	spherical	**14.1**	**31.4**	**11.0**

Table 4: The performance of our network with different backbones and different types of convolutions.

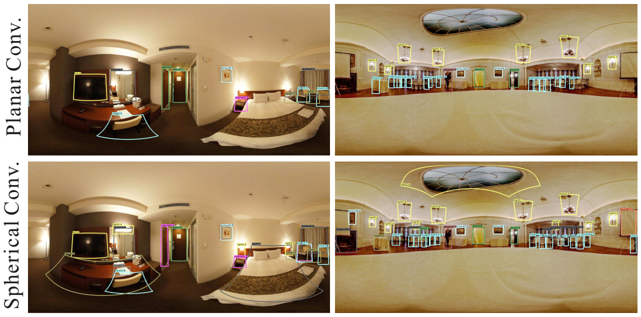


Figure 8: Compared with planar convolution, spherical convolution can detect more seriously distorted objects.

Visual Detection Results. We give some visual detection results of our method on the three datasets in Figure 7. Our method can successfully detect objects in spherical images, even if the objects have large distortions or are split by the left or right boundaries of spherical images. See the table and the bed in images from 360-Indoor, the cat and the bottle in images from 360-VOC-Gaussian, and the aeroplane and the cat in images from 360-VOC-Uniform.

Ablation Study

Backbone. We train our network with two different backbones: ResNet-101 (He et al. 2016) and Hourglass (Newell, Yang, and Deng 2016). These two backbones have about the same depth, but Hourglass uses skip layers to bring back the details to the upsampled features. This can greatly improve the performance of the detection network, especailly for the anchor-free network, as shown in Table 4.

Type of Convolution. Our network leverages spherical convolutions to deal with the distortions of spherical images. To check the effect of spherical convolutions, we have trained a network using traditional planar convolutions and compared it with the network using spherical convolutions. As shown in Table 4, the usage of spherical convolutions can significantly improve detection performance. We give some visual comparisons in Figure 8. It is obvious that spherical convolutions can let networks to detect objects with large distortions. For example, the seriously stretched table, bed and light are detected by the network using spherical convolutions.

Conclusion

In this paper, we propose the first unbiased IoU for spherical image object detection. We first illustrate that spherical rectangles are natural representations for bounding boxes of spherical objects. Then we give the unbiased IoU calculation method based on the new representation. We also present a new anchor-free object detection algorithm for spherical images, which directly output bounding boxes for objects. Extensive experiments on three datasets show that our method can get better results. In the future, we would like to apply our unbiased IoU in other tasks like visual tracking.

Acknowledgements

This work is supported by National Key R&D Program of China (2018YFB0804203) and National Natural Science Foundation of China (62072438, 61771458, U1936110).

References

Anguelov, D.; Dulong, C.; Filip, D.; Frueh, C.; Lafon, S.; Lyon, R.; Ogale, A.; Vincent, L.; and Weaver, J. 2010. Google Street View: Capturing the World at Street Level. *Computer*, 43(6): 32–38.

Chou, S.-H.; Sun, C.; Chang, W.-Y.; Hsu, W.-T.; Sun, M.; and Fu, J. 2020. 360-Indoor: Towards Learning Real-World Objects in 360 Indoor Equirectangular Images. In *WACV*, 834–842.

Coors, B.; Condurache, A. P.; and Geiger, A. 2018. SphereNet: Learning Spherical Representations for Detection and Classification in Omnidirectional Images. In *ECCV*, 525–541.

Eder, M.; Shvets, M.; Lim, J.; and Frahm, J.-M. 2020. Tangent Images for Mitigating Spherical Distortion. In *CVPR*, 12423–12431.

Everingham, M.; Gool, L. V.; Williams, C. K. I.; Winn, J.; and Zisserman, A. 2010. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 88: 303–338.

Everingham, M.; ·, S. M. A. E.; Gool, L. V.; Williams, C. K. I.; Winn, J.; and Zisserman, A. 2015. The Pascal Visual Object Classes Challenge: A Retrospective. *IJCV*, 111(1): 98–136.

Girshick, R. 2015. Fast R-CNN. In *ICCV*, 1440–1448.

Gu, X.; Sun, J.; and Xu, Z. 2020. Spherical space domain adaptation with robust pseudo-label loss. In *CVPR*, 9101–9110.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*, 770–778.

Hu, H.-N.; Lin, Y.-C.; Liu, M.-Y.; Cheng, H.-T.; Chang, Y.-J.; and Sun, M. 2017. Deep 360 Pilot: Learning a Deep Agent for Piloting through 360 Sports Videos. In *CVPR*, 1396–1405.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv:1412.6980.*

Law, H.; and Deng, J. 2018. CornerNet: Detecting Objects as Paired Keypoints. In *ECCV*, 765–781.

Lee, Y.; Jeong, J.; Yun, J.; Cho, W.; and Yoon, K.-J. 2019. SpherePHD: Applying CNNs on a Spherical PolyHeDron Representation of 360 Images. In *CVPR*, 9173–9181.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal Loss for Dense Object Detection. In *ICCV*, 2999–3007.

Liu, L.; Ouyang, W.; Wang, X.; Fieguth, P.; Chen, J.; Liu, X.; and Pietikäinen, M. 2020. Deep Learning for Generic Object Detection: A Survey. *IJCV*, 128: 261–318.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. SSD: Single Shot MultiBox Detector. In *ECCV*, 21–37.

Newell, A.; Yang, K.; and Deng, J. 2016. Stacked Hourglass Networks for Human Pose Estimation. In *ECCV*, 483–499.

O'Rourke, J. 1998. *Computational Geometry in C*. Cambridge university press.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch.

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *CVPR*, 779–788.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *TPAMI*, 39(6): 1137–1149.

Shen, Z.; Shen, T.; Lin, Z.; and Ma, J. 2021. PDO-eS2CNNs: Partial Differential Operator Based Equivariant Spherical CNNs. In *AAAI*, volume 35, 9585–9593.

Su, Y.-C.; and Grauman, K. 2017. Learning Spherical Convolution for Fast Features from 360 Imagery. In *NeurIPS*, volume 30, 529–539.

Su, Y.-C.; and Grauman, K. 2019. Kernel Transformer Networks for Compact Spherical Convolution. In *CVPR*, 9434–9443.

Tateno, K.; Navab, N.; and Tombari, F. 2018. Distortion-Aware Convolutional Filters for Dense Prediction in Panoramic Images. In *ECCV*, 732–750.

Todhunter, I. 1863. *Spherical trigonometry, for the use of colleges and schools: with numerous examples*. Macmillan.

Wang, K.-H.; and Lai, S.-H. 2019. Object Detection in Curved Space for 360-Degree Camera. In *ICASSP*, 3642–3646.

Yan, C.; Gong, B.; Wei, Y.; and Gao, Y. 2020. Deep multi-view enhancement hashing for image retrieval. *TPAMI*, 43(4): 1445–1451.

Yang, W.; Qian, Y.; Kämäräinen, J.-K.; Cricri, F.; and Fan, L. 2018. Object Detection in Equirectangular Panorama. In *ICPR*, 2190–2195.

Yu, D.; and Ji, S. 2019. Grid Based Spherical CNN for Object Detection from Panoramic Images. *Sensors*, 19(11).

Zhang, S.; Chi, C.; Yao, Y.; Lei, Z.; and Li, S. Z. 2020. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *CVPR*, 9759–9768.

Zhang, Y.; Dai, F.; Ma, Y.; Li, H.; Zhao, Q.; and Zhang, Y. 2019. Saliency Prediction Network for 360° Videos. *JSTSP*, 14(1): 27–37.

Zhao, P.; You, A.; Zhang, Y.; Liu, J.; Bian, K.; and Tong, Y. 2020. Spherical Criteria for Fast and Accurate 360 Object Detection. *AAAI*, 34(07): 12959–12966.

Zhao, Q.; Dai, F.; Ma, Y.; Wan, L.; Zhang, J.; and Zhang, Y. 2018. Spherical Superpixel Segmentation. *TMM*, 20(6): 1406–1417.

Zhao, Q.; Zhu, C.; Dai, F.; Ma, Y.; Jin, G.; and Zhang, Y. 2018. Distortion-aware CNNs for Spherical Images. In *IJCAI*, 1198–1204.

Zhou, X.; Wang, D.; and Krähenbühl, P. 2019. Objects as points. *arXiv:1904.07850.*