Unbiased IoU for Spherical Image Object Detection

Feng Dai[1], Bin Chen[1,2], 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]Noah's Ark Lab, Huawei, China

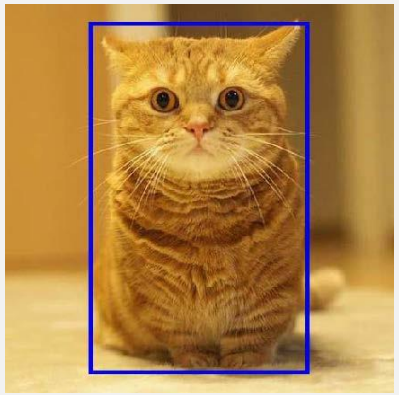
*Qiang Zhao is the Corresponding Author.

CONTENTS

PART 01 | Introduction

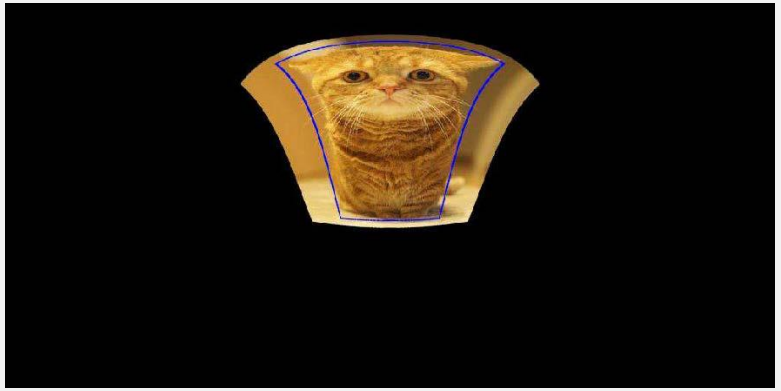
Introduction

Planar Image Object Detection



Determine the categories and locations of planar objects

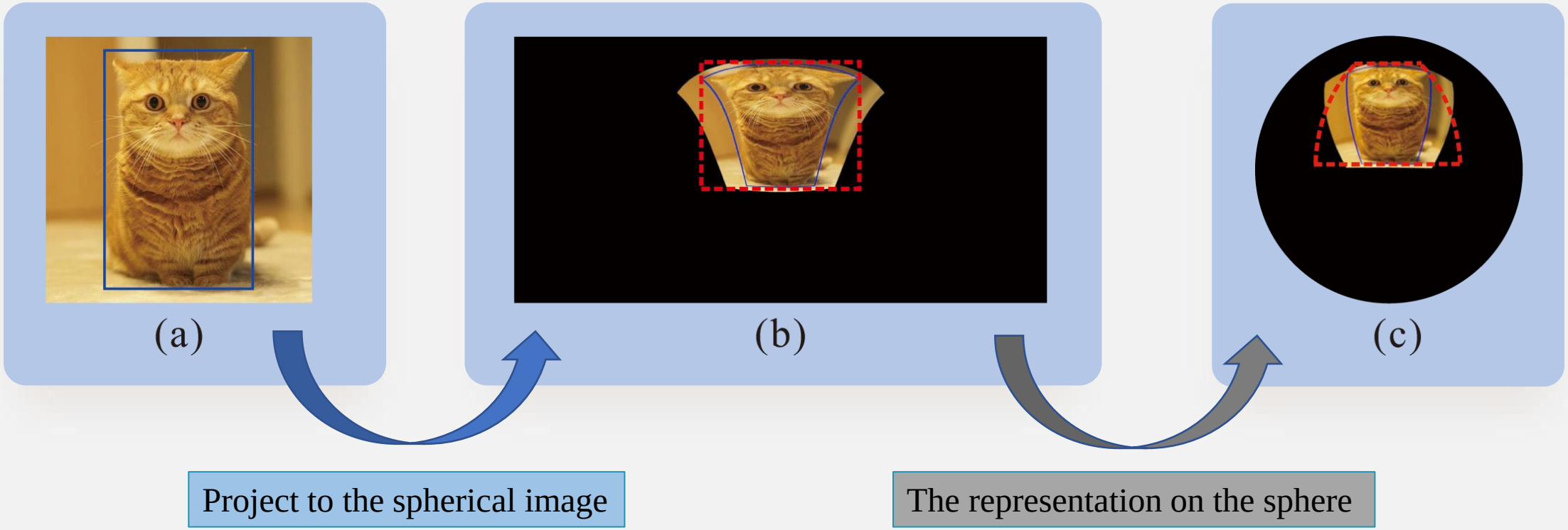
Spherical Image Object Detection



Determine the categories and locations of distorted spherical objects

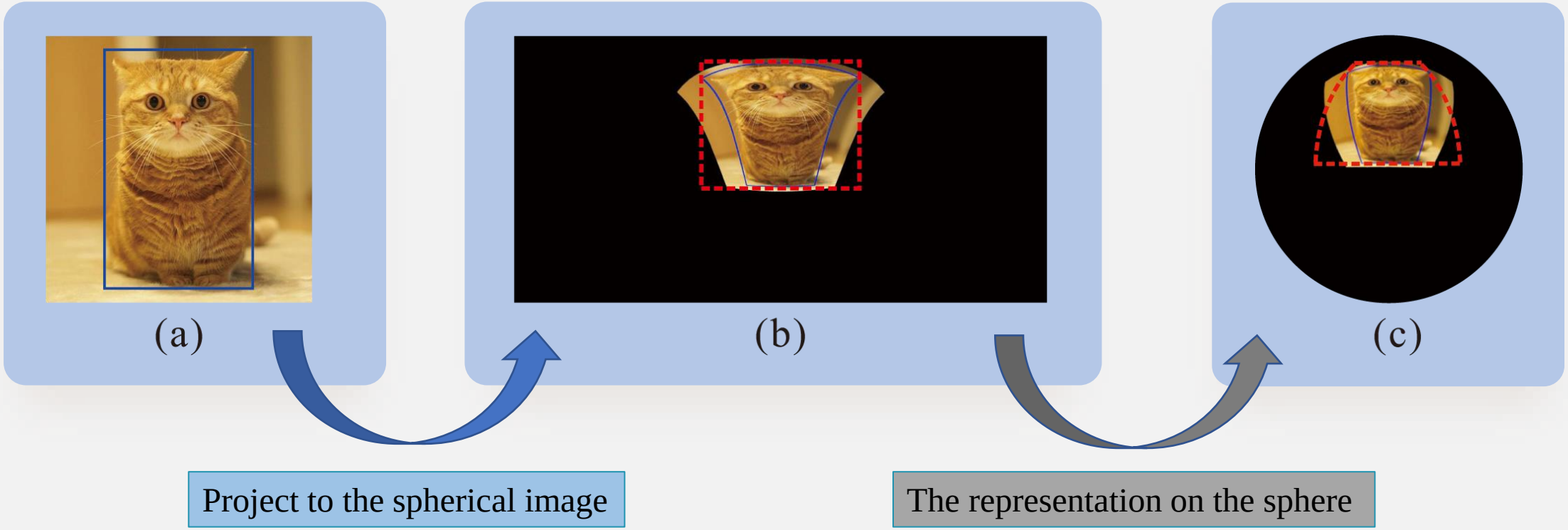
Introduction

Spherical Image Object Detection



Introduction

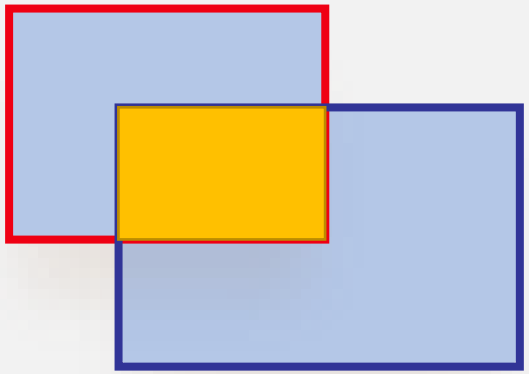
Spherical Image Object Detection



Objects in spherical images cannot be bound tightly by planar rectangles or other biased representations.

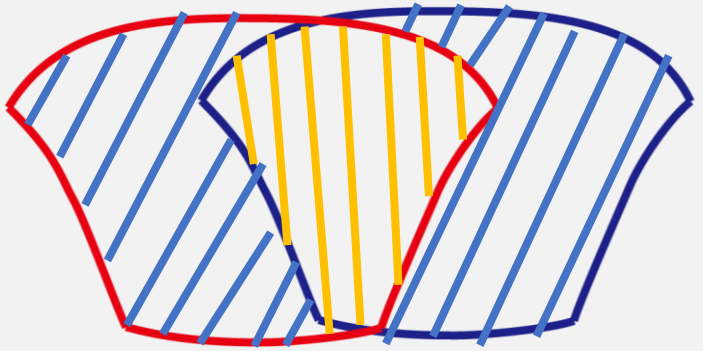
Introduction

IoU for Rectangles



$$IoU = \frac{Area(\square)}{Area(\square) + Area(\square) - Area(\square)}$$

IoU for Spherical Rectangles



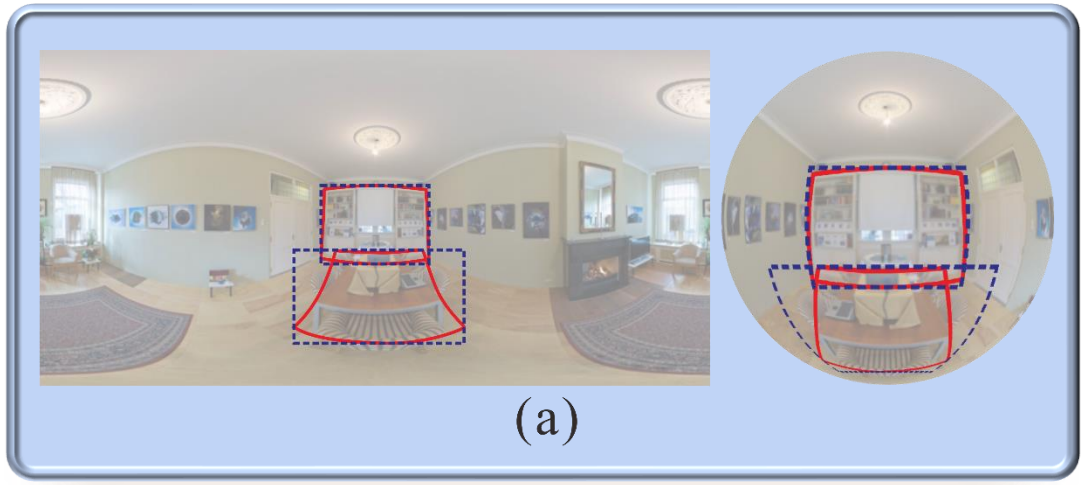
$$IoU = \frac{Area(\square)}{Area(\square) + Area(\square) - Area(\square)}$$

- The IoU plays an important role in positive/negative samples selection, NMS operation in training stage and mAP calculation in evaluation, thus an unbiased IoU is essential to object detection task.
- Rectangles can tightly bound objects in planar images, and it is easy to compute its IoU.
- Spherical Rectangles can tightly bound objects in spherical images, but it is not easy to compute its IoU due to its complex boundaries and the intersection area.

PART 02 | Motivation

Motivation

1. Existing Biased IoU Representation

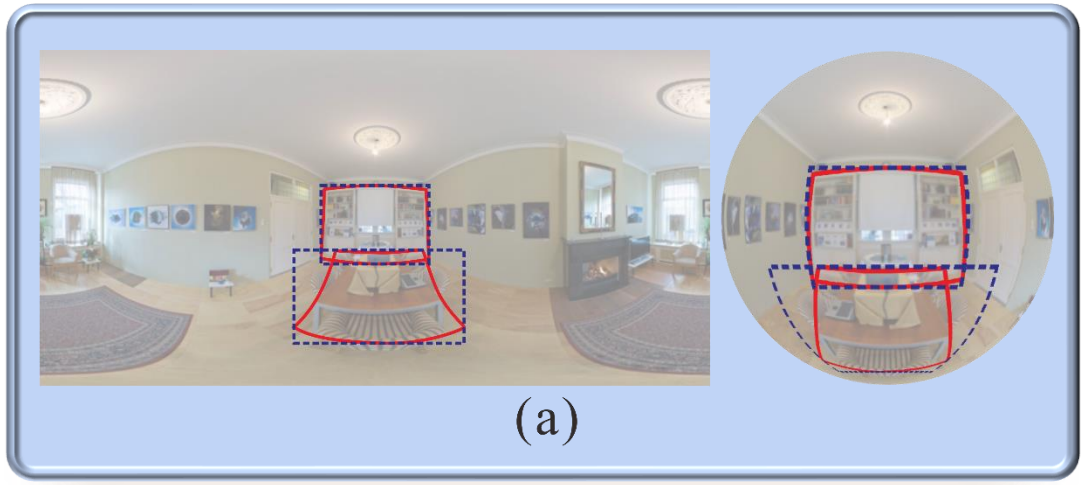


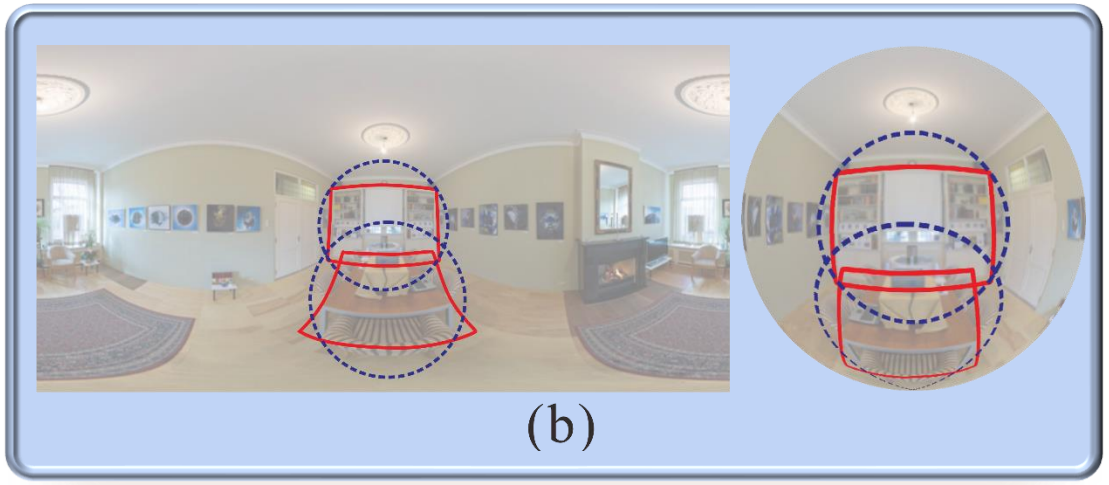
Some existing evaluation criteria use biased bounding boxes to represent spherical rectangles in spherical images.

(a) Using axis-aligned rectangles to represent spherical objects in (Yang et al. 2018; Wang and Lai 2019).

Motivation

1. Existing Biased IoU Representation





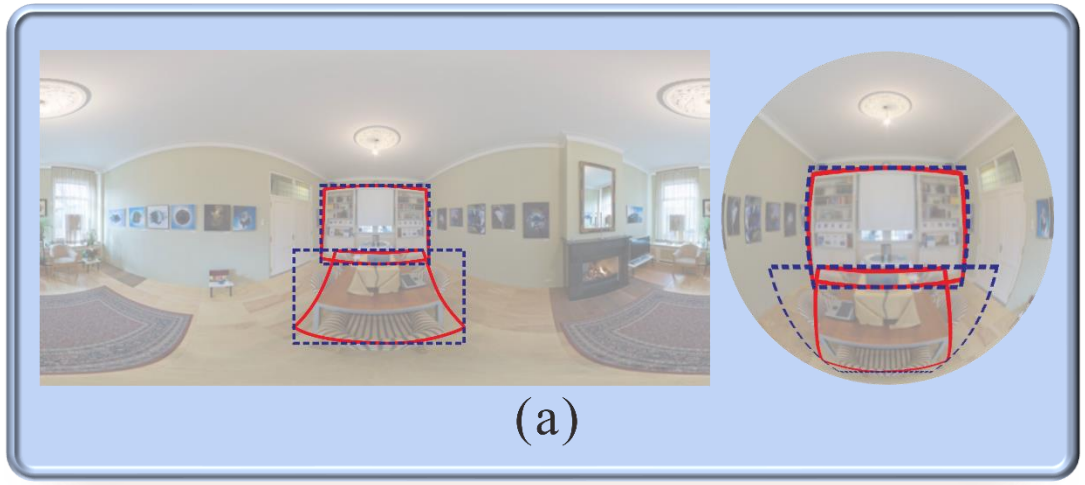
Some existing evaluation criteria use biased bounding boxes to represent spherical rectangles in spherical images.

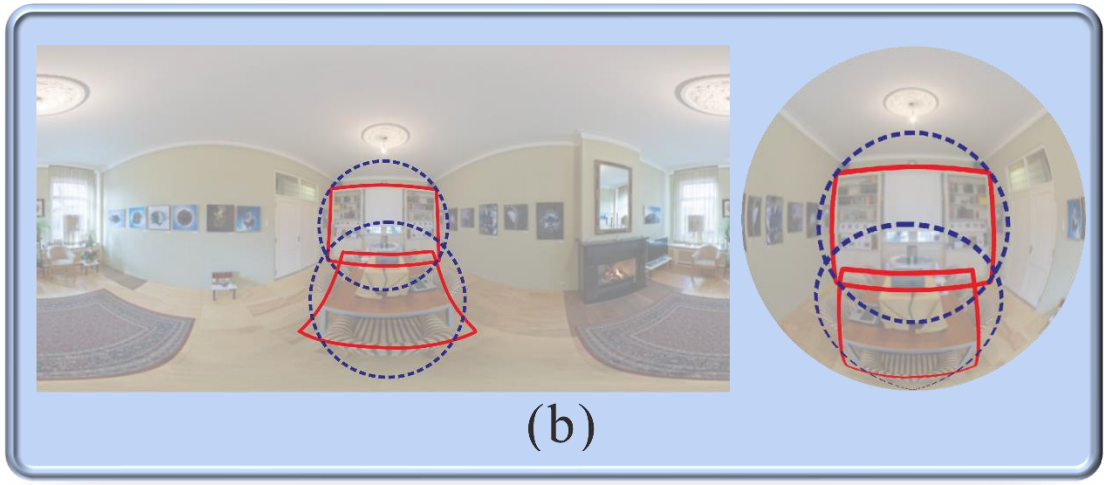
(a) Using axis-aligned rectangles to represent spherical objects in (Yang et al. 2018; Wang and Lai 2019).

(b) Using circles to represent spherical objects in (Lee et al. 2019).

Motivation

1. Existing Biased IoU Representation





Some existing evaluation criteria use biased bounding boxes to represent spherical rectangles in spherical images.

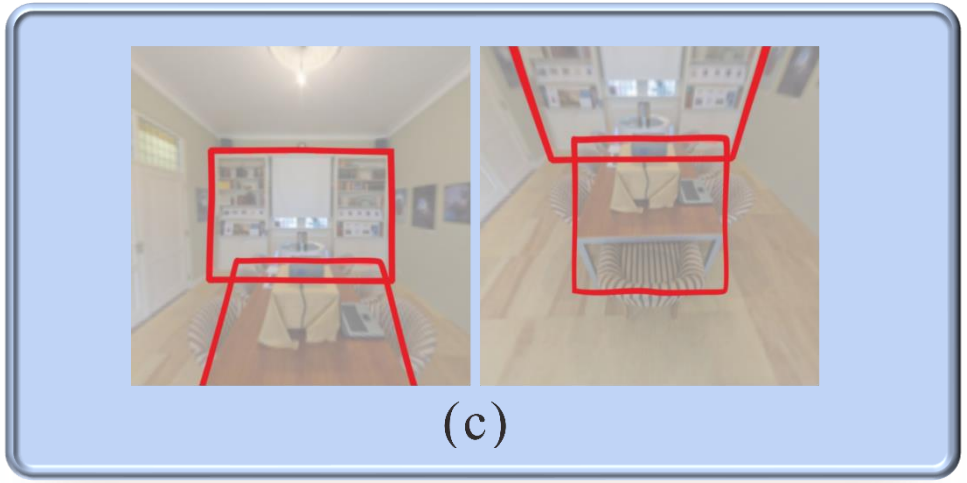
(a) Using axis-aligned rectangles to represent spherical objects in (Yang et al. 2018; Wang and Lai 2019).

(b) Using circles to represent spherical objects in (Lee et al. 2019).

Both of them compute the IoU between two rectangles or circles without considering the distortions of unrolled spherical images, thus they use biased representations and have large errors.

Motivation

2. Existing Biased IoU Calculation

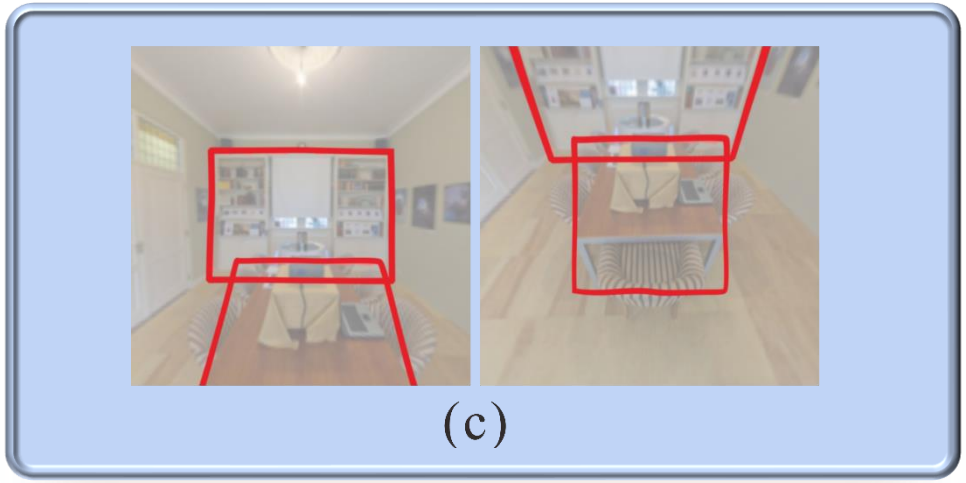


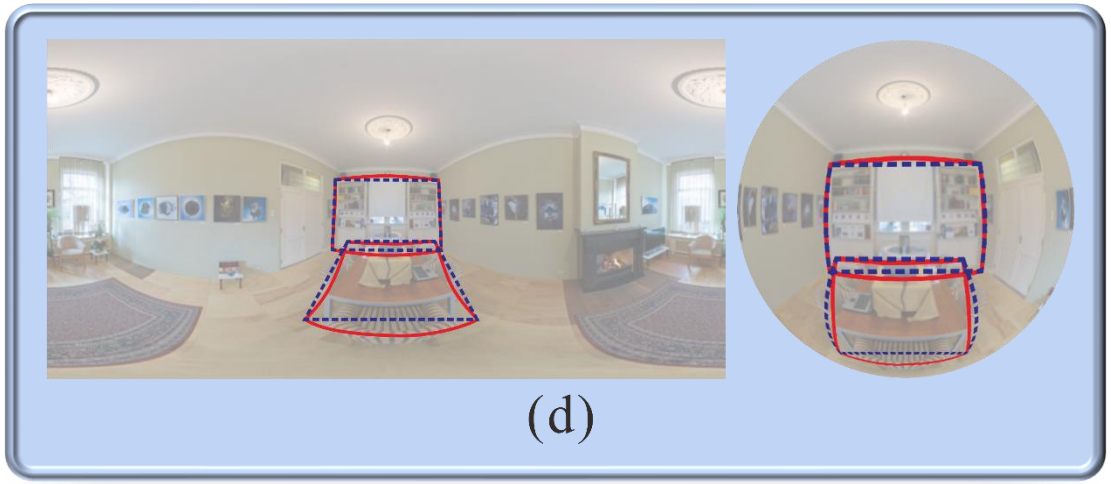
Some existing evaluation criteria use biased bounding boxes to represent spherical rectangles in spherical images.

(c) Using axis-aligned rectangles on tangent planes (Su and Grauman 2017; Coors, Condurache, and Geiger 2018).

Motivation

2. Existing Biased IoU Calculation





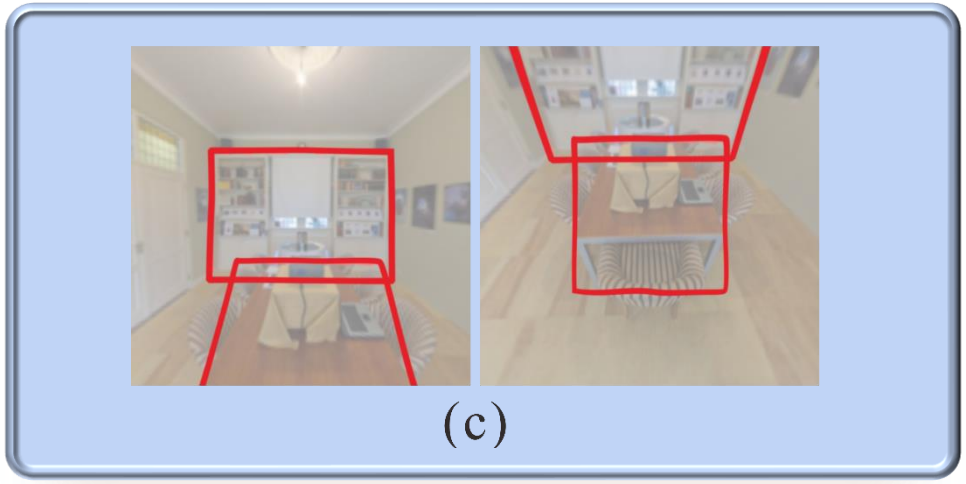
Some existing evaluation criteria use biased bounding boxes to represent spherical rectangles in spherical images.

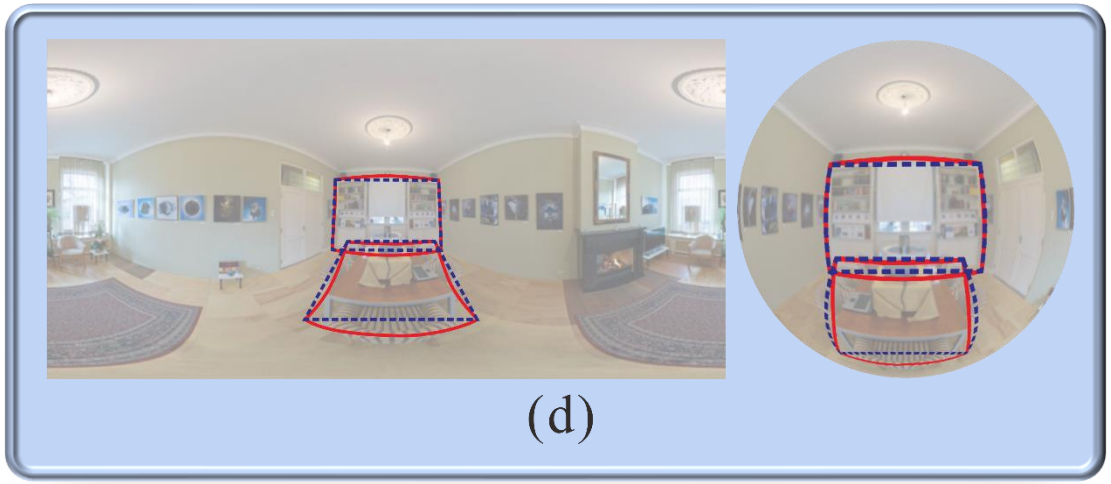
(c) Using axis-aligned rectangles on tangent planes (Su and Grauman 2017; Coors, Condurache, and Geiger 2018).

(d) Using sampled evenly spaced points on tangent planes and projecting them to spherical image, then computing IoUs based on polygons on spherical images (Coors, Condurache, and Geiger 2018).

Motivation

2. Existing Biased IoU Calculation





Some existing evaluation criteria use biased bounding boxes to represent spherical rectangles in spherical images.

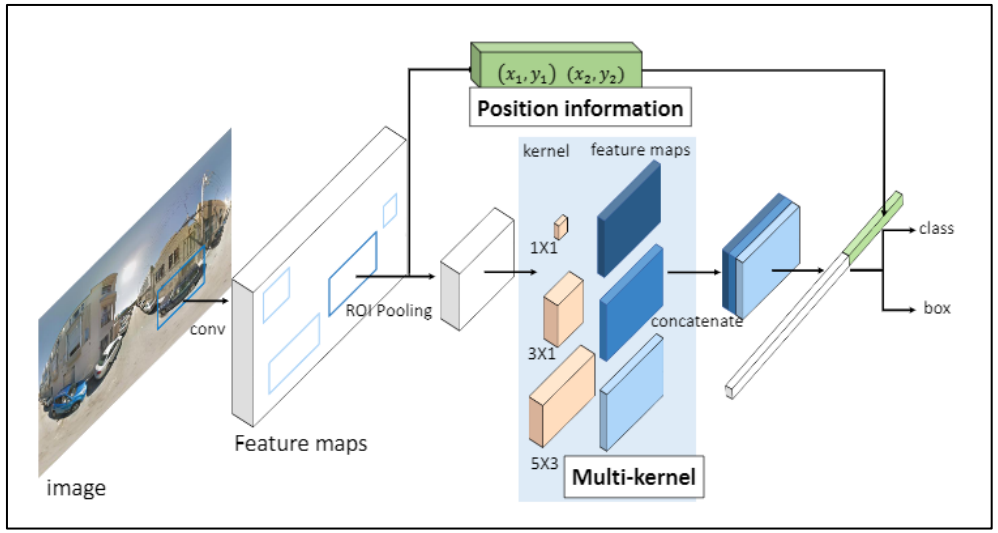
(c) Using axis-aligned rectangles on tangent planes (Su and Grauman 2017; Coors, Condurache, and Geiger 2018).

(d) Using sampled evenly spaced points on tangent planes and projecting them to spherical image, then computing IoUs based on polygons on spherical images (Coors, Condurache, and Geiger 2018).

Both of them make excessive approximations when computing the IoU, thus they give biased calculations and incorrect results.

Motivation

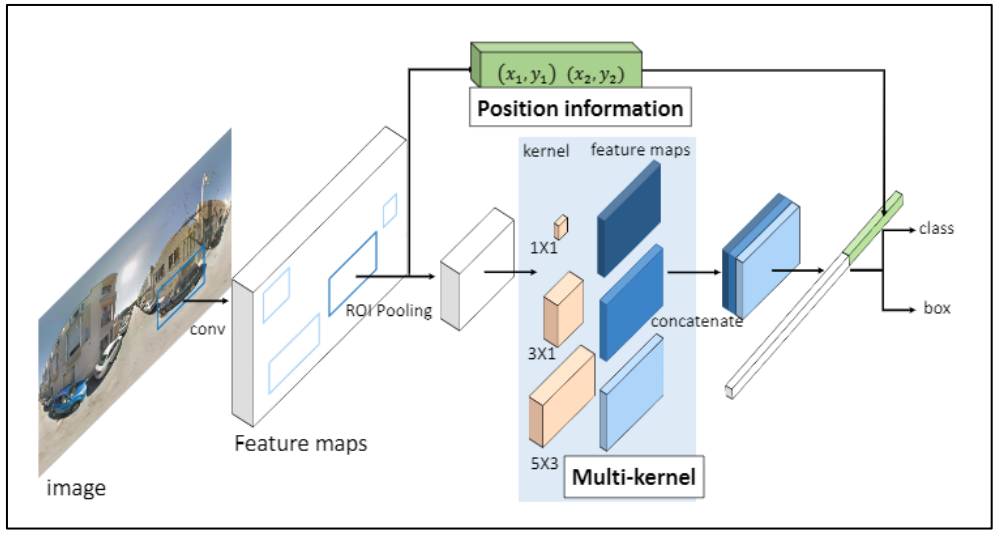
3. Existing Detectors for Spherical Image Object Detection Task



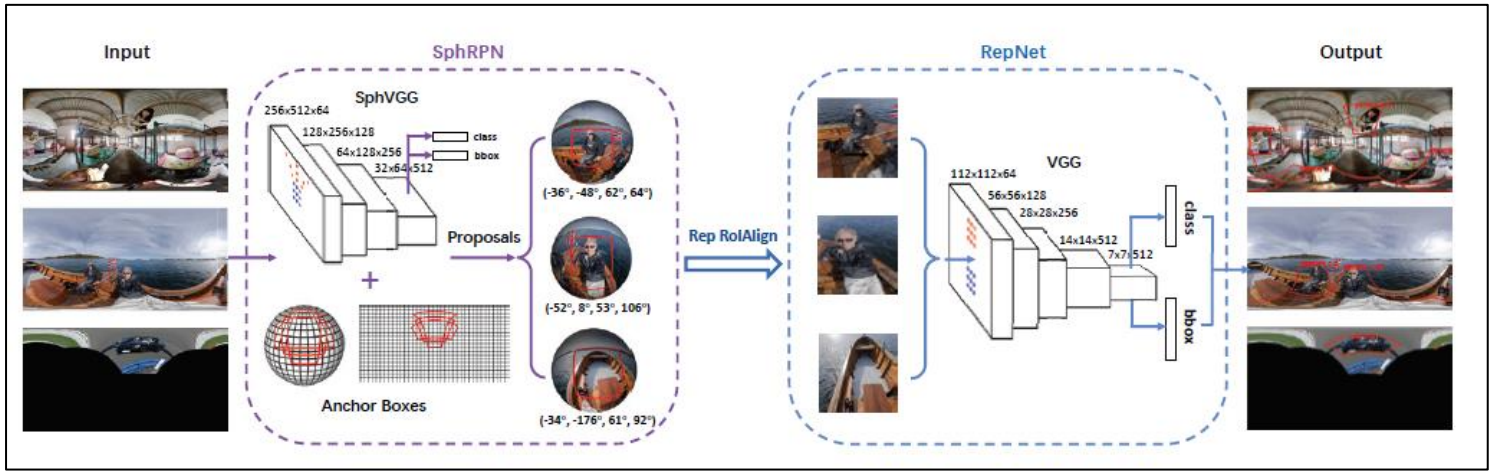
- Multi-kernel (Wang and Lai 2019): Using convolution kernels of different sizes and position information. It gives poor ability to deal with distortion, which leads to poor detection performance.

Motivation

3. Existing Detectors for Spherical Image Object Detection Task

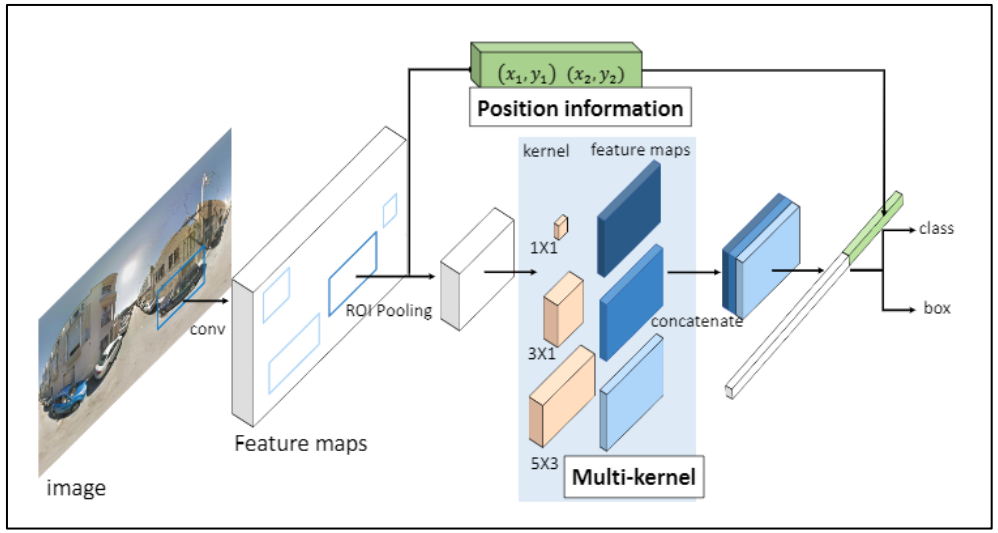


- Multi-kernel (Wang and Lai 2019): Using convolution kernels of different sizes and position information. It gives poor ability to deal with distortion, which leads to poor detection performance.
- Reprojection R-CNN (Zhao et al. 2020): A two-stage detector, the anchor selection and sampling operation is complex. The network training is slow and does not support the end-to-end training mode.

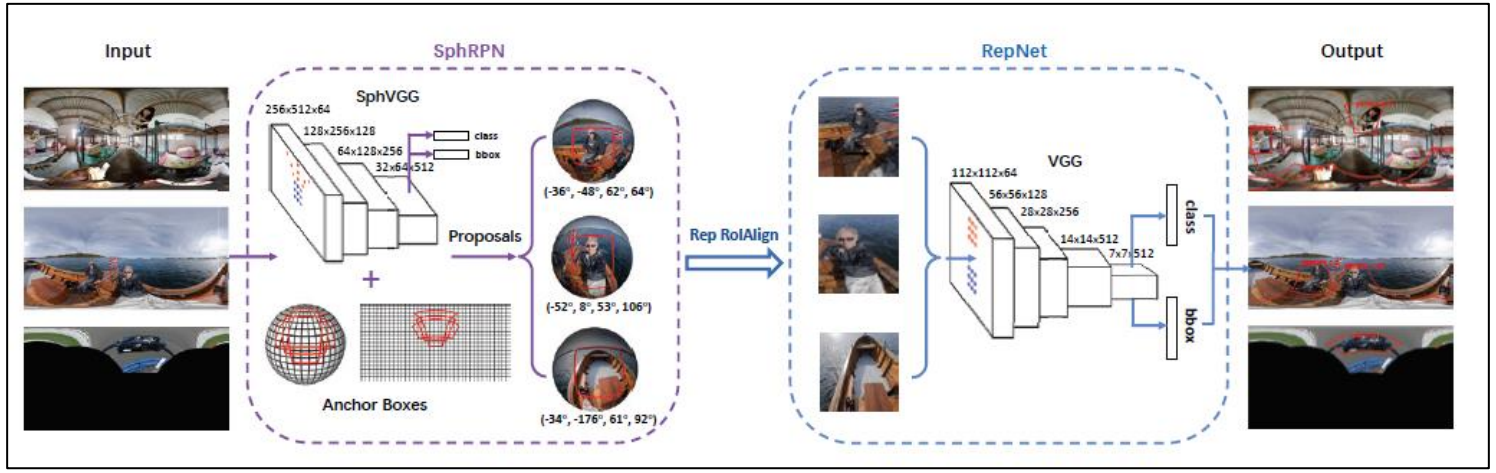


Motivation

3. Existing Detectors for Spherical Image Object Detection Task



- Multi-kernel (Wang and Lai 2019): Using convolution kernels of different sizes and position information. It gives poor ability to deal with distortion, which leads to poor detection performance.
- Reprojection R-CNN (Zhao et al. 2020): A two-stage detector, the anchor selection and sampling operation is complex. The network training is slow and does not support the end-to-end training mode.



The existing detection methods either gives poor performance or contains complex network structure and unsupported end-to-end training mode.

PART 03 | Contributions

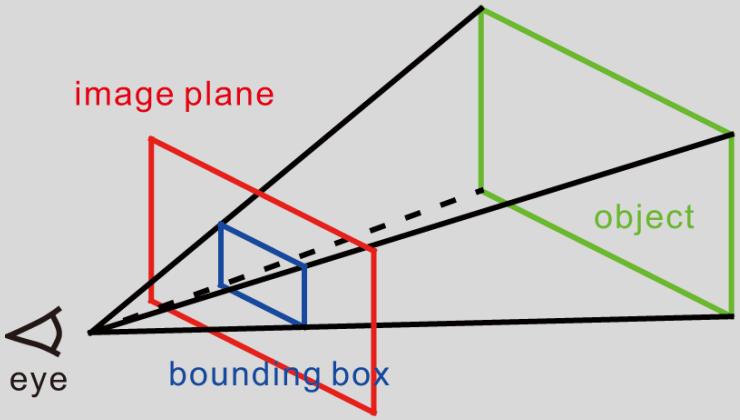
Contributions

Our main contributions

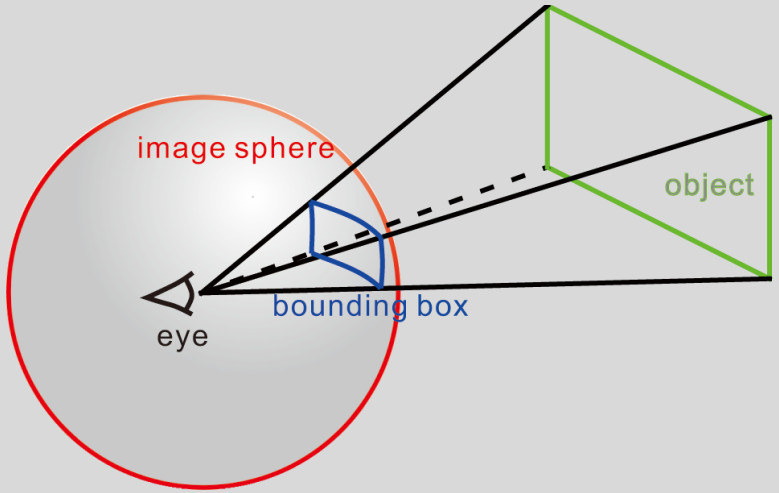
- We propose an unbiased IoU as a novel evaluation criterion for spherical image object detection, which is based on the unbiased representations and utilize unbiased analytical method for IoU calculation.
- To the best of our knowledge, our work on proposed unbiased IoU is the first absolutely accurate spherical IoU both in the representation and in the calculation, which is applied to the training and evaluation for spherical image object detection task, and thus spherical image object detection algorithms can be correctly evaluated.
- We also present Spherical CenterNet, an anchor free object detection algorithm for spherical images. The experiments show that the proposed Spherical CenterNet achieves better performance on one real-world and two synthetic spherical object detection datasets than existing methods.

PART 04 | Methodology

Unbiased IoU Representation



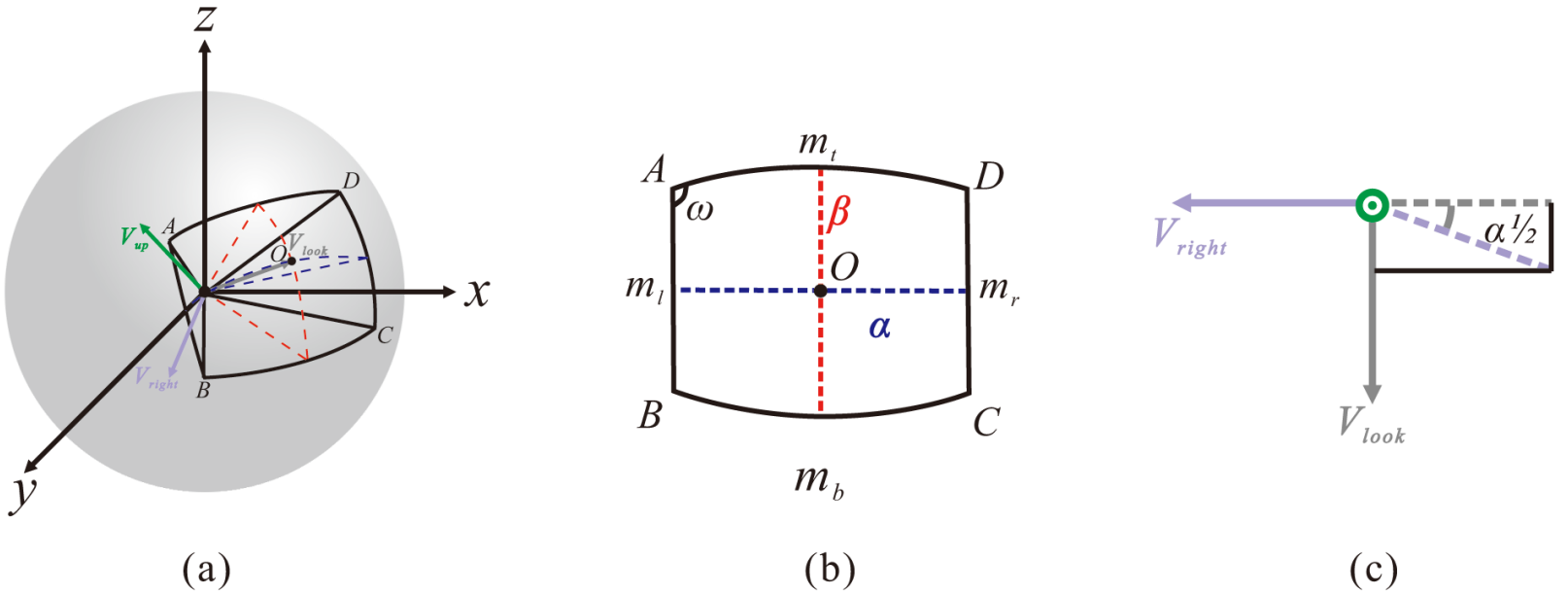
- In planar case, the spatial location and extent of an object are defined using an axis-aligned rectangle: (x, y, w, h), where (x, y) is the center point and (w, h) is the width and height.



- In spherical case, the tightly bounded spherical rectangle can be defined as: (θ, φ, α, β), where θ is the azimuthal angle, φ is the polar angle, α and β is the horizontal and vertical field of view respectively.

Methodology

Unbiased IoU Calculation – Area of Spherical Rectangles



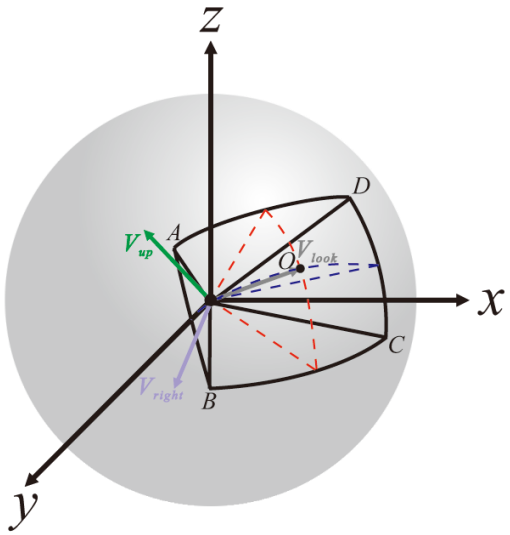
(a) (b) (c)

1. Construct the new coordinate system:

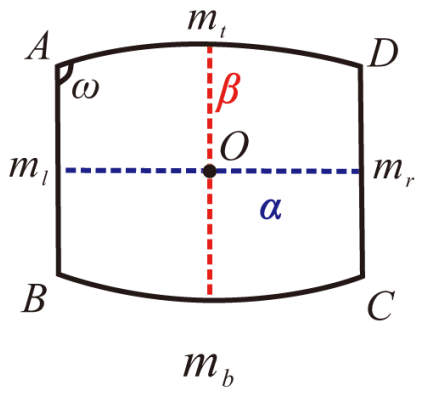
$$\begin{cases} V_{look} = [\sin(\phi)\cos(\theta), \sin(\phi)\sin(\theta), \cos(\phi)]^T \\ V_{right} = [-\sin(\theta), \cos(\theta), 0]^T \\ V_{up} = [-\cos(\phi)\cos(\theta), -\cos(\phi)\sin(\theta), \sin(\phi)]^T \end{cases}$$

Methodology

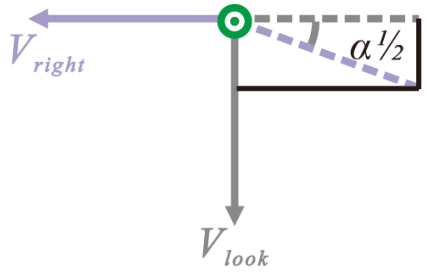
Unbiased IoU Calculation – Area of Spherical Rectangles



(a)



(b)



(c)

1. Construct the new coordinate system:

$$\begin{cases} V_{look} = [\sin(\phi)\cos(\theta), \sin(\phi)\sin(\theta), \cos(\phi)]^T \\ V_{right} = [-\sin(\theta), \cos(\theta), 0]^T \\ V_{up} = [-\cos(\phi)\cos(\theta), -\cos(\phi)\sin(\theta), \sin(\phi)]^T \end{cases}$$

2. Calculate the normal vector of each plane:

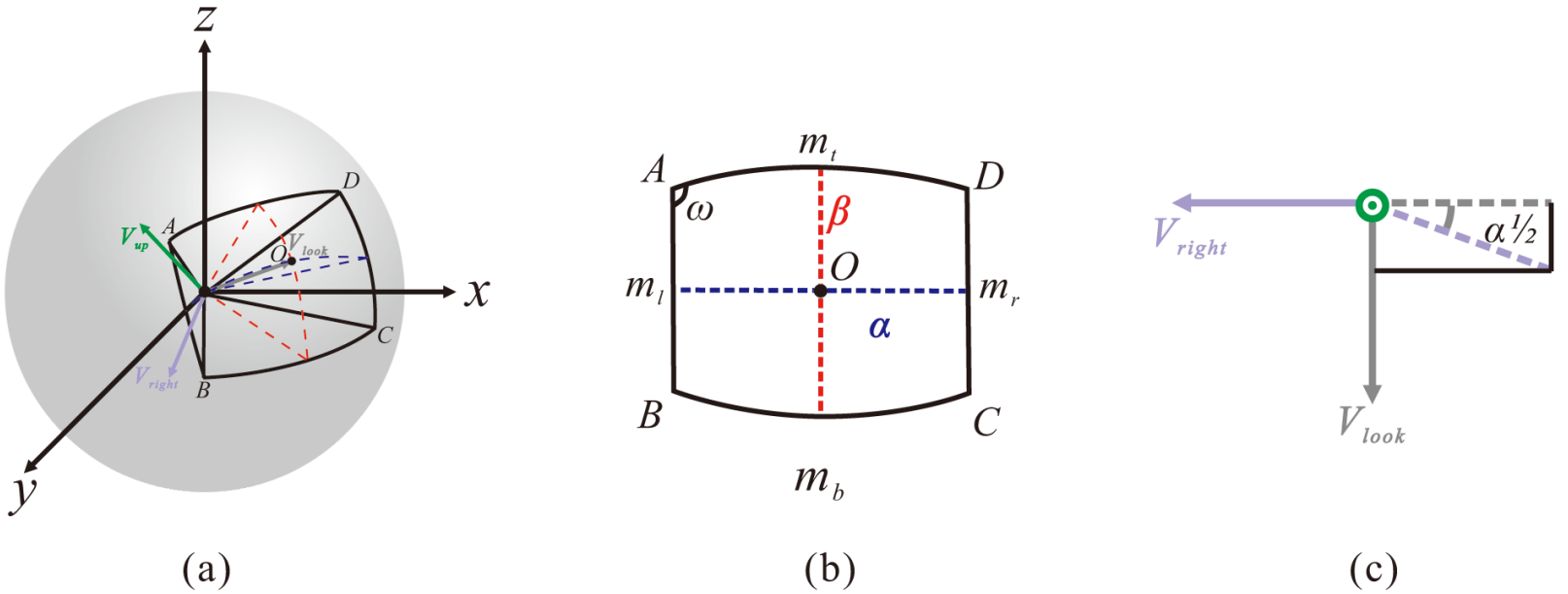
$$N_l = \sin\frac{\alpha}{2}V_{look} - \cos\frac{\alpha}{2}V_{right}$$

$$N_t = \sin\frac{\beta}{2}V_{look} - \cos\frac{\beta}{2}V_{up}$$

$$\vdots$$

Methodology

Unbiased IoU Calculation – Area of Spherical Rectangles



(a) (b) (c)

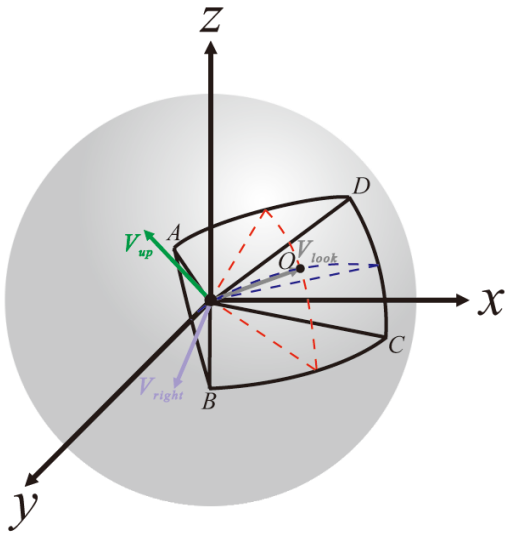
3. Calculate the boundary angles on the sphere:

$$\omega = \pi - \arccos(\langle N_l, N_t \rangle) = \arccos(-\sin\frac{\alpha}{2}\sin\frac{\beta}{2})$$

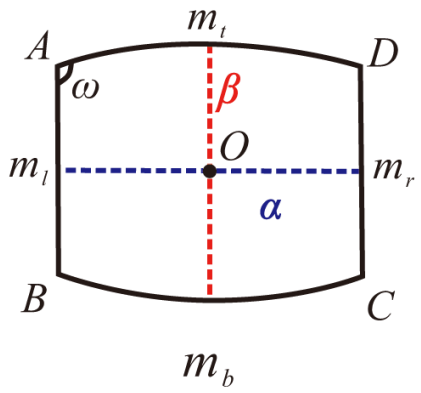
$$\vdots$$

Methodology

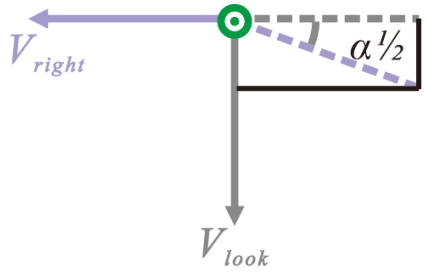
Unbiased IoU Calculation – Area of Spherical Rectangles



(a)



(b)



(c)

3. Calculate the boundary angles on the sphere:

$$\omega = \pi - \arccos(\langle N_l, N_t \rangle) = \arccos(-\sin\frac{\alpha}{2}\sin\frac{\beta}{2})$$

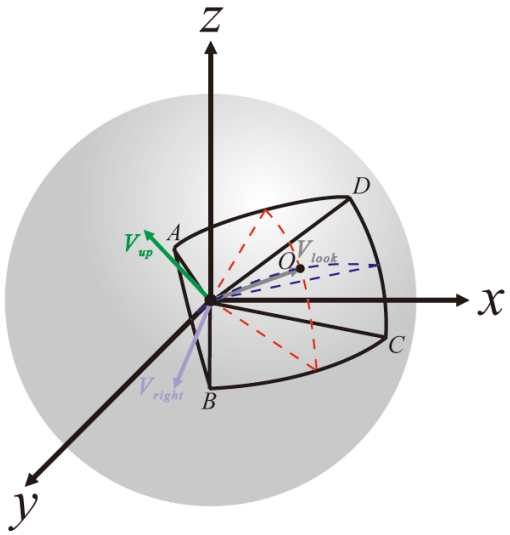
$$\vdots$$

4. Compute the area of spherical rectangle:

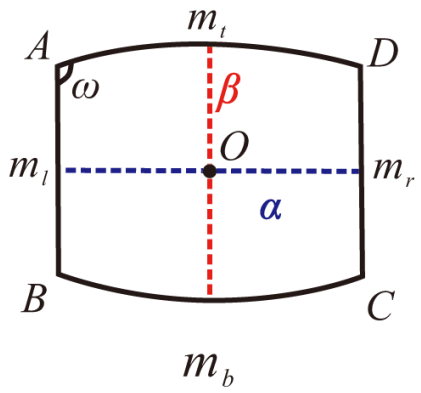
$$A(b) = 4\arccos(-\sin\frac{\alpha}{2}\sin\frac{\beta}{2}) - 2\pi$$

Methodology

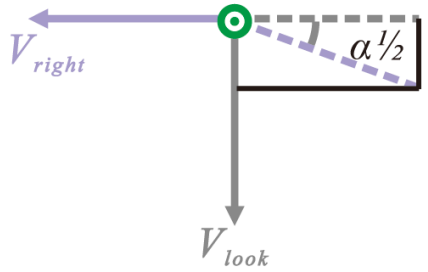
Unbiased IoU Calculation – Area of Spherical Rectangles



(a)



(b)



(c)

3. Calculate the boundary angles on the sphere:

$$\omega = \pi - \arccos(\langle N_l, N_t \rangle) = \arccos(-\sin\frac{\alpha}{2}\sin\frac{\beta}{2})$$

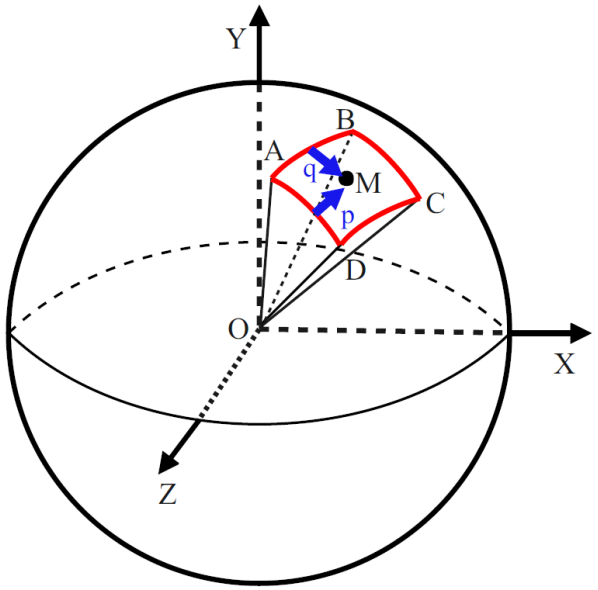
$$\vdots$$

4. Compute the area of spherical rectangle:

$$A(b) = 4\arccos\left(-\sin\left(\frac{\alpha}{2}\right)\sin\left(\frac{\beta}{2}\right)\right) - 2\pi$$

Now we have derived the analytical solution of the area of spherical rectangle.

Unbiased IoU Calculation – Intersection Area Computation Algorithm



Example:

- Normal Vector **p** and **q** of plane *OAD* and *OAB*
- *The vertex A can be computed by:* **p × q**

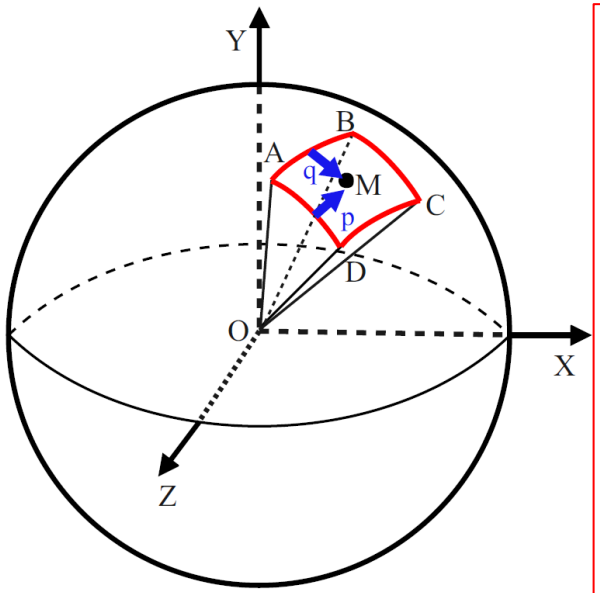
Step 7 & 8: Compute the normal vectors as before, and then compute 8 vertices (4 vertices for each spherical rectangle) and other 32 vertices (4 × 4 × 2) by cross product of the normal vectors of two intersecting planes.

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 ω_i
end
return $A(b_1 \cap b_2)$ computed via Equation 3;

Unbiased IoU Calculation – Intersection Area Computation Algorithm



Example:

- The point **M** is an inner point because of cos(**OM**, $\mathbf{nv_i}$).all() ≥ 0, where $\mathbf{nv_i}$ is the normal vector of the i-th spherical rectangle boundary.

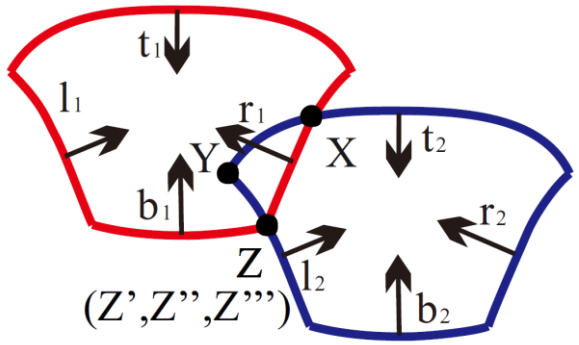
Step 10: Remove points outside the intersection region by dot product.

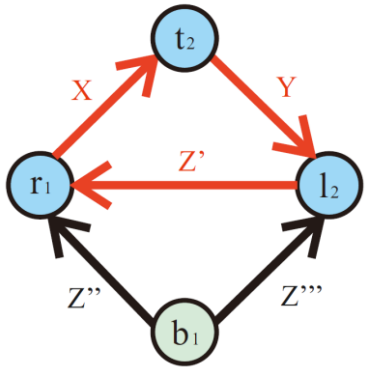
Algorithm 1: Intersection Area Computation

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 ω_i
end
return $A(b_1 \cap b_2)$ computed via Equation 3;

Unbiased IoU Calculation – Intersection Area Computation Algorithm





Example:

- Z is the duplicated point (3 intersecting boundaries)
- After Step 10, there are five points left:
  $\{X(\vec{r_1}, \vec{t_2}), Y(\vec{t_2}, \vec{l_2}), Z'(\vec{l_2}, \vec{r_1}), Z''(\vec{b_1}, \vec{r_1}), Z'''(\vec{b_1}, \vec{l_2})\}$
- By DFS algorithm, the only closed-loop $Z' \rightarrow X \rightarrow Y$ has remained and three intersection points are given.

Step 11: Remove duplicated points (more than two boundaries intersect at the duplicated points) via loop detection.

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 ω_i
end
return $A(b_1 \cap b_2)$ computed via Equation 3;

Unbiased IoU Calculation – Intersection Area Computation Algorithm

Step 12-15: Compute the angles by cross product of the normal vectors of left points, and then the final intersection area can be computed via the following formula.

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

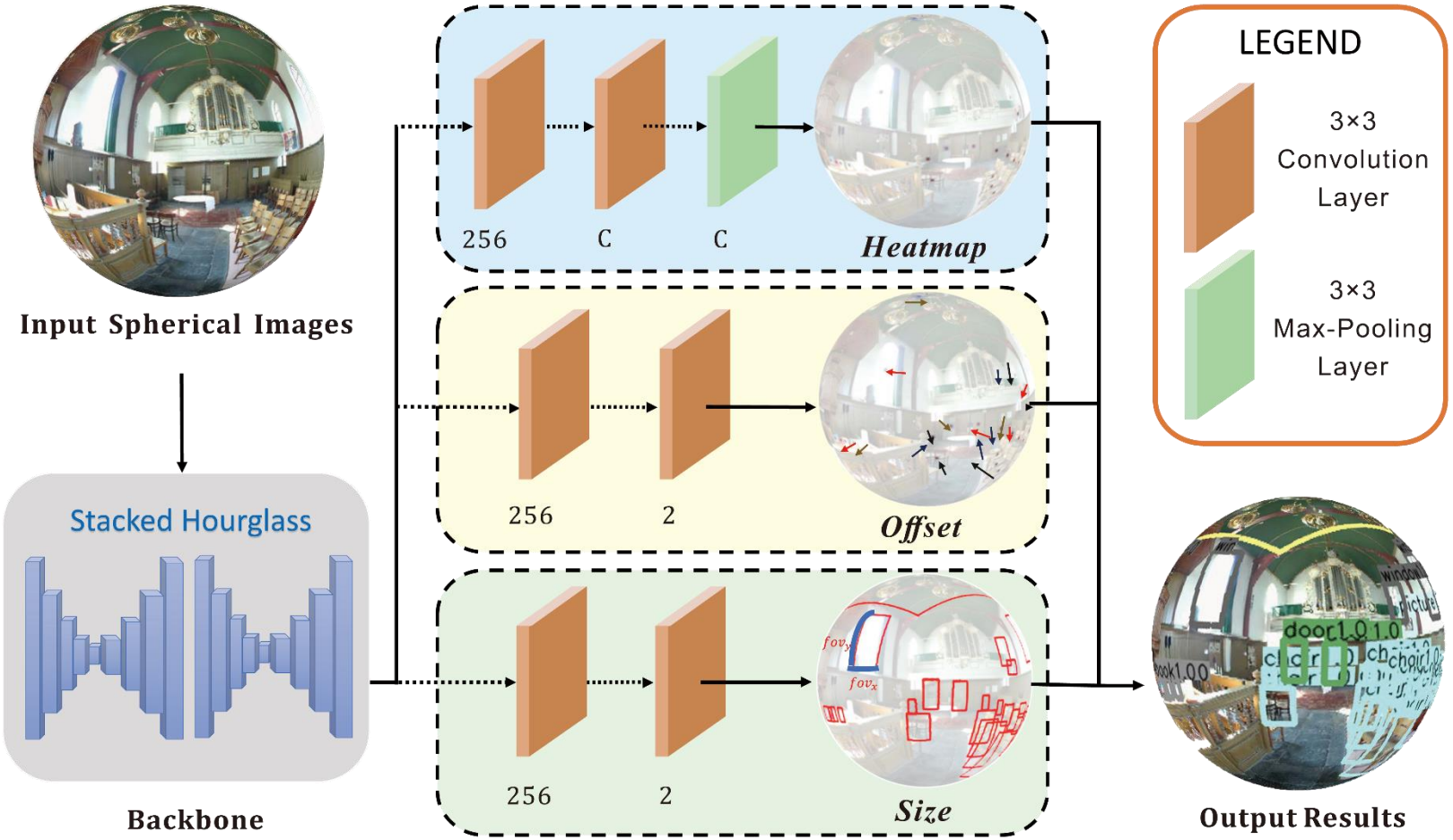
Algorithm 1: Intersection Area Computation

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 ω_i
end
return $A(b_1 \cap b_2)$ computed via Equation 3;

Methodology

An Anchor-free Spherical Detection Method: Spherical CenterNet



An Anchor-free Spherical Detection Method – Loss Definition

- The classification loss (based on Focal Loss(Lin et al. 2017))

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

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

We introduce a weight w_{xy} 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.

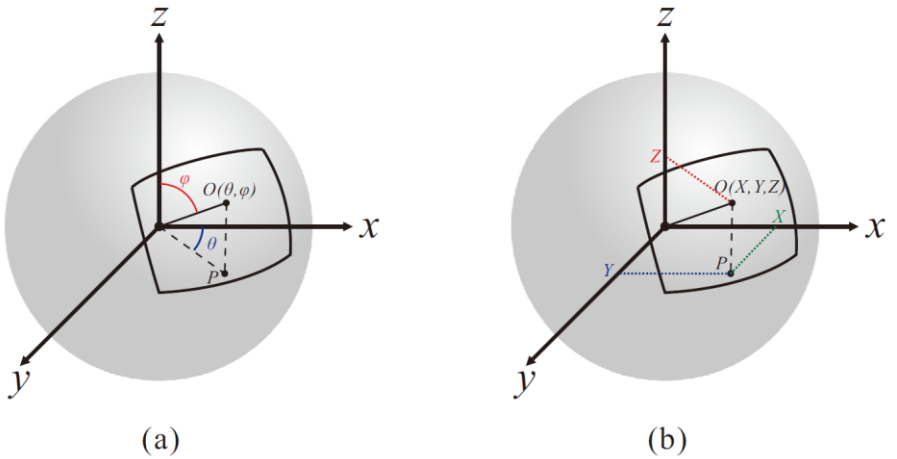
- The field of view regression loss (L1 Loss)

$$L_{fov} = \frac{1}{N}\sum_{i} |\mathbf{s}_i - \hat{\mathbf{s}}_i|$$

An Anchor-free Spherical Detection Method – Loss Definition

- The offset regression loss (Measure the angle between two 3D unit vectors)

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

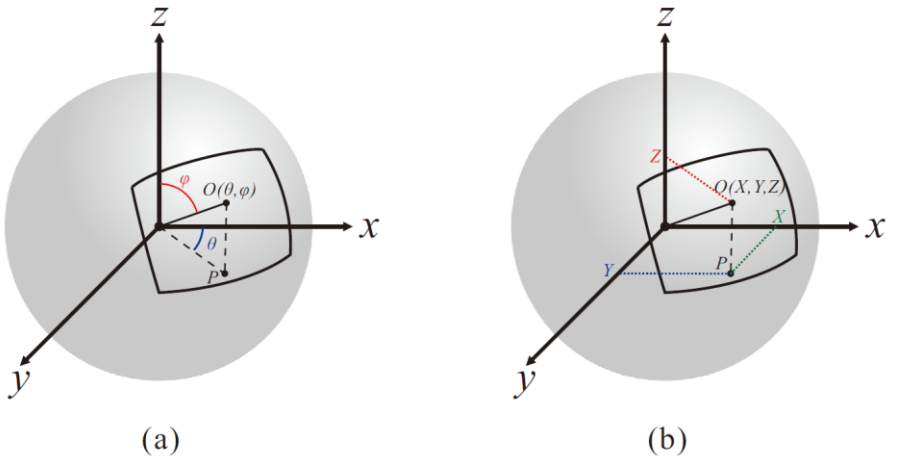


$$\mathcal{T}: \begin{cases} X = \sin\varphi\cos\theta \\ Y = \sin\varphi\sin\theta \\ Z = \cos\varphi \end{cases}$$

An Anchor-free Spherical Detection Method – Loss Definition

- The offset regression loss (Measure the angle between two 3D unit vectors)

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



$$\mathcal{T}: \begin{cases} X = \sin\varphi\cos\theta \\ Y = \sin\varphi\sin\theta \\ Z = \cos\varphi \end{cases}$$

- The overall training objective (The final loss)

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

An Anchor-free Spherical Detection Method – Ground Truth Generation

- The generation of the ground truth offset

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

- The generation of the ground truth heatmap

We use gaussian kernel and assign nonzero values to negative locations within a radius of positive locations. The ground truth heatmaps is given by

$$\exp\left(-\frac{\arccos\left(\langle \mathcal{T}(\hat{\theta}_i, \hat{\phi}_i), \mathcal{T}(\theta, \phi) \rangle \right)}{2\sigma^2} \right)$$

An Anchor-free Spherical Detection Method – Ground Truth Generation

- The generation of the ground truth offset

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

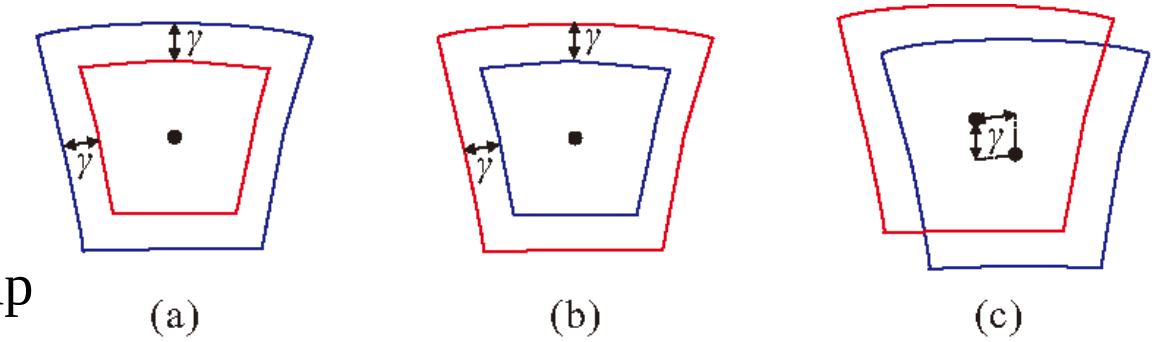
- The generation of the ground truth heatmap

We use gaussian kernel and assign nonzero values to negative locations within a radius of positive locations. The ground truth heatmaps is given by

$$\exp\left(-\frac{\arccos\left(\langle \mathcal{T}(\hat{\theta}_i,\hat{\phi}_i), \mathcal{T}(\theta,\phi)\rangle\right)}{2\sigma^2}\right)$$

The radius is determined by ensuring the IoU within a threshold with the ground truth. There are three cases, and the final radius is the minimum of them.

Case (a): $\dfrac{4\arccos(-\sin\frac{\alpha}{2}\sin\frac{\beta}{2}) - 2\pi}{4\arccos\left(-\sin(\frac{\alpha+2\gamma}{2})\sin(\frac{\beta+2\gamma}{2})\right) - 2\pi} = t$



Case (a):

$$\gamma = \frac{1}{2}\arccos\left(-2\sin\left(\frac{\arcsin(\sin\frac{\alpha}{2}\sin\frac{\beta}{2})}{t}\right) + \cos(\frac{\alpha-\beta}{2})\right) - \frac{\alpha+\beta}{4}.$$

An Anchor-free Spherical Detection Method – Ground Truth Generation

- The generation of the ground truth offset

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

- The generation of the ground truth heatmap

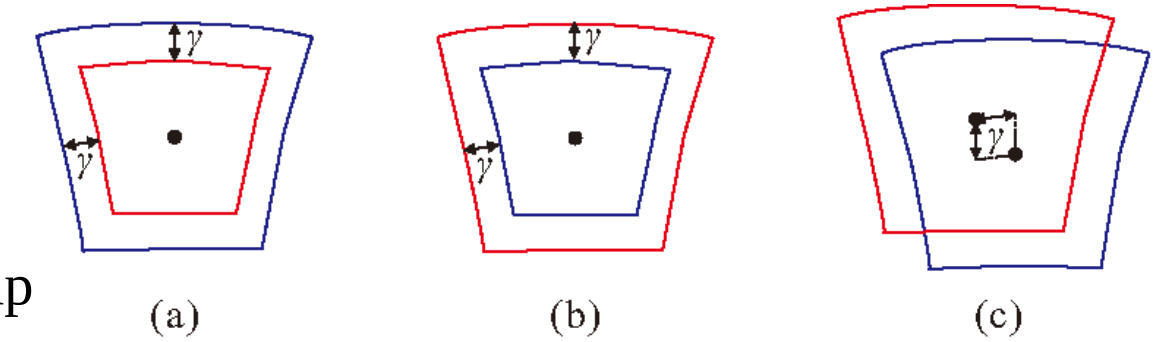
We use gaussian kernel and assign nonzero values to negative locations within a radius of positive locations. The ground truth heatmaps is given by

$$\exp\left(-\frac{\arccos\left(\langle \mathcal{T}(\hat{\theta}_i,\hat{\phi}_i), \mathcal{T}(\theta,\phi)\rangle\right)}{2\sigma^2}\right)$$

The radius is determined by ensuring the IoU within a threshold with the ground truth. There are three cases, and the final radius is the minimum of them.

Case (a): $\dfrac{4\arccos(-\sin\frac{\alpha}{2}\sin\frac{\beta}{2}) - 2\pi}{4\arccos\left(-\sin(\frac{\alpha+2\gamma}{2})\sin(\frac{\beta+2\gamma}{2})\right) - 2\pi} = t$

Final Radius: $\gamma = \min(\gamma_a, \gamma_b, \gamma_c)$



Case (a):

$$\gamma = \frac{1}{2}\arccos\left(-2\sin\left(\frac{\arcsin(\sin\frac{\alpha}{2}\sin\frac{\beta}{2})}{t}\right) + \cos(\frac{\alpha-\beta}{2})\right) - \frac{\alpha+\beta}{4}.$$

Case (b):

$$\gamma = -\frac{1}{2}\arccos\left(-2\sin\left(t\arcsin(\sin\frac{\alpha}{2}\sin\frac{\beta}{2})\right) + \cos(\frac{\alpha-\beta}{2})\right) + \frac{\alpha+\beta}{4}.$$

Case (c):

$$\gamma = -\arccos\left(-2\sin\left(\frac{2t\,(\arccos(-\sin\frac{\alpha}{2}\sin\frac{\beta}{2}) - 2\pi)}{1+t}\right) + \cos(\frac{\alpha-\beta}{2})\right) + \frac{\alpha+\beta}{2}.$$

Methodology

Ground Truth Heatmaps & Spherical Convolution

- Ground Truth Heatmaps

  There are two ground truth heatmaps. The shape of the Gaussian kernels is consistent with the distortion of spherical objects. See the bed and the fan for example.

- Spherical Convolutions

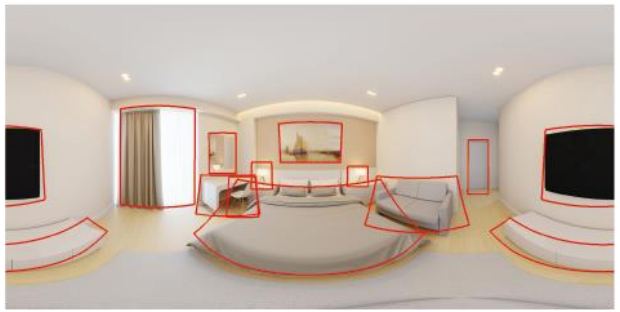
  We use tangent images (Eder et al. 2020) to alleviate distortion problem, which facilitates transferable and scalable 360° computer vision. We choose this type for two reasons: it keeps the parameter sharing property of convolution; it does not lead to performance degradation if more convolutional layers are added.

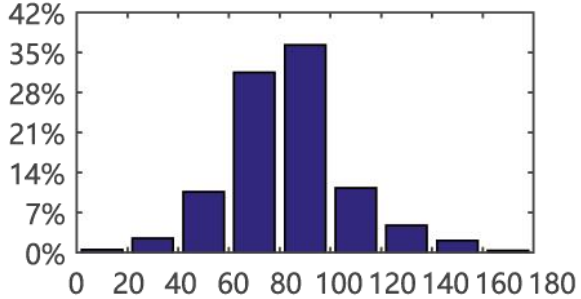
PART 05 | Experiments

Experiments

Datasets

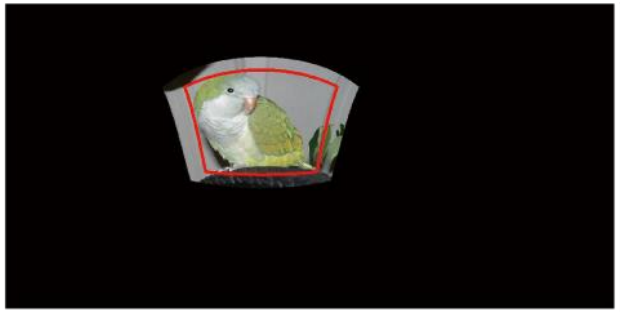
360-Indoor

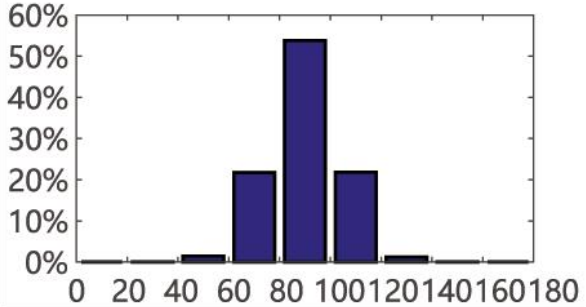




Real-world dataset:
- 3k images
- 37 categories
- Latitude distribution: Gaussian distribution

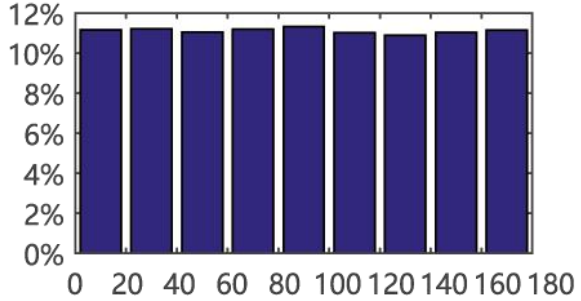
360-VOC-Gaussian





Synthetic VOC dataset1:
- 28k images
- 20 categories
- Latitude distribution: Gaussian distribution

360-VOC-Uniform



Synthetic VOC dataset2:
- 28k images
- 20 categories
- Latitude distribution: Uniform distribution

Experiments

Experimental Results on different IoU computation methods

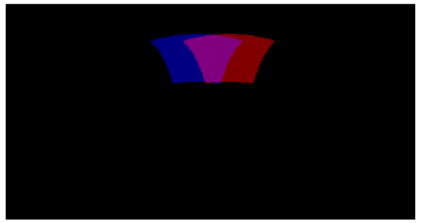
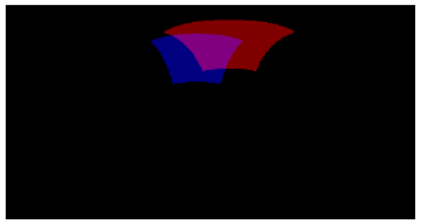
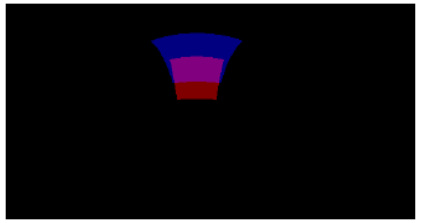
Cases	Methods	IoUs	△
	Sph. Integral	0.32006	-
	Rectangle	0.47163	0.15157
	Polygon	0.35891	0.03885
	Circle	0.24286	0.07720
	SphIoU	0.16537	0.15469
	Ours	**0.31974**	**0.00032**
	Sph. Integral	0.25801	-
	Rectangle	0.55155	0.29354
	Polygon	0.26958	0.01157
	Circle	0.24996	0.00805
	SphIoU	0.11392	0.17109
	Ours	**0.25772**	**0.00029**
	Sph. Integral	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 by numerical integration is taken as the reference method. The differences (△) are listed between each method and the reference method.

- The first three methods give incorrect result, as they do not compute the IoU on the sphere.
- SphIoU (Zhao et al. 2020) also gives incorrect result, as it treats spherical rectangle as parts of spherical zones and has made too many approximations.

Experimental Results on spherical integral and our unbiased IoU

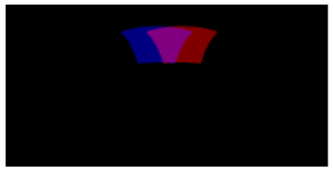
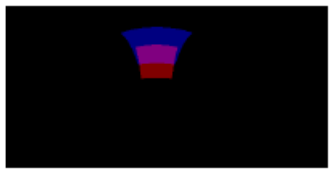
Cases	Methods	$12k \times 6k$	$10k \times 5k$	$8k \times 4k$
	Sph. Integral	0.32006	0.32012	0.32022
	Ours	0.31974	0.31974	0.31974
	△	**0.00032**	**0.00038**	**0.00048**
	Sph. Integral	0.25801	0.25807	0.25816
	Ours	0.25772	0.25772	0.25772
	△	**0.00029**	**0.00035**	**0.00044**
	Sph. Integral	0.33966	0.33972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

Table 2: The IoUs computed with spherical integral 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.

Note: With the decrease of the resolution, the accuracy of the spherical integral method 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).

Experiments

Experimental Results on different spherical image object detection methods

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.

- In order to ensure the fairness, we conduct the experiments on different spherical image object detection methods in its original implementation, and evaluate the performance on our unbiased IoU (Those biased representations is converted to spherical rectangles for the IoU and the final mAP metric computation).
- Our method can give the best performance on all three datasets compared with other methods.

Ablation Study

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.

Ablation Study

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

Experiments

Visualization Results

360-Indoor

Figure 2: Visual detection results of our method on 360-Indoor.

Experiments

Visualization Results

Figure 3: Visual detection results of our method on 360-VOC-Gaussian and 360-VOC-Uniform datasets.

PART 06 | Conclusions

Conclusions

- We propose the first unbiased IoU for spherical image object detection. We illustrate that spherical rectangles are natural representations for bounding boxes of spherical objects, and then give the unbiased IoU calculation method based on the new representation.
- We present a new anchor-free object detection algorithm for spherical images, which directly output bounding boxes for objects.
- Extensive experiments show that our unbiased IoU gives accurate results and the proposed Spherical CenterNet can get better results.
- In the future, we would like to apply our unbiased IoU in other computer vision tasks, such as visual tracking.

Thank you!