iShape: A First Step Towards Irregular Shape Instance Segmentation Appendix

1 Algorithm Supplement

1.1 Constructing the ASIS Affinity Kernel

Algorithm 1 describes the process of constructing the ASIS affinity kernel. The input of the algorithm contains the kernel radius r_k, the affinity neighbor gap g, and the initial radius r_0 (default is 0). The output is the affinity neighbor set P which represents the relative coordinates of the current point $(0, 0)$. Above all, we use $r_0 + g$ as the initial value of the radius and $r_k + g$ as the maximum value of the radius to calculate the radius r of a semicircle every affinity neighbor gap g. For a semicircle whose radius is r, the second line of the Algorithm will calculate n, the number of affinity neighbors that will be assigned to the semicircle. Further, we calculate the polar coordinate angle interval $\Delta\theta$ of two adjacent affinity neighbors that belong to one semicircle. In particular, each semicircle has a perturbation value θ_0 that makes affinity neighbors on adjacent semicircles staggered. The polar angle of ith neighbor θ is determined by $i\Delta\theta - \theta_0$. At last, we employ the polar coordinate formula to obtain the Cartesian coordinate values of the affinity neighbor and merge them into the affinity neighbor set P. To enhance short range affinity, the ASIS kernel has the same structure as the GMIS [1] when the radius is 1, 2, 4, and 8.

Algorithm 1 ASIS Affinity Kernel construction algorithm

Require: Kernel radius r_k, affinity neighbor gap g, initial radius r_0 (default is 0)
Ensure: Affinity neighbor set P

for $r = r_0 + g; r < r_k + g; r = r + g$ **do**
  $n = \lceil \frac{\pi r}{g} \rceil$
  $\Delta\theta = \frac{\pi}{n}$
  $\theta_0 = (\frac{r}{2g}\ mod\ 1)\Delta\theta$
  for $i = 1; i <= n; i++$ **do**
    $\theta = i\Delta\theta - \theta_0$
    Update $P : P = P \bigcup \{(-\lfloor rcos\theta \rceil, -\lfloor rsin\theta \rceil)\}$
  end for
end for

1.2 Search Parameters for Affinity Kernel

Since each dataset has its optimal affinity kernel, we propose Algorithm 2 that could adaptively generate appropriate r_k and g based on the dataset property.

Where $GetCircleRadius$ is defined as:

$$\begin{aligned} GetCircleRadius(c) =& [MinEnclosingCircleRadius(c) \\ &+ MaxInscribedCircleRadius(c)]/2 \end{aligned} \tag{1}$$

and $GetThreshold(L, \alpha, \beta)$ first sort the input list L with N_L items in an ascending order, then get the No. $\lfloor \alpha \times N_L \rceil$ item and multiply it with β to get the output result, denoted as:

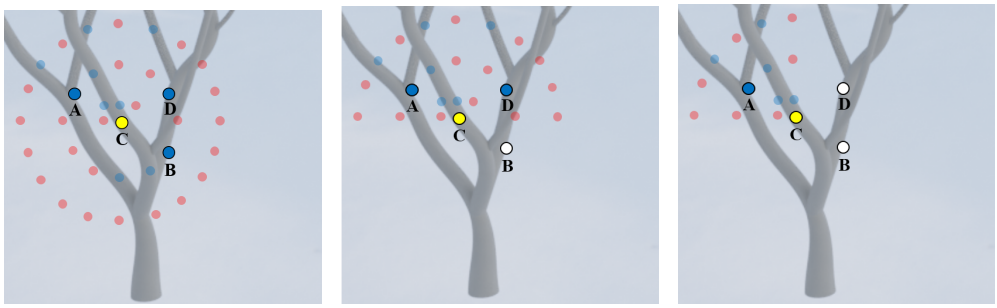
$$GetThreshold(L, \alpha, \beta) = \beta Sorted(L)_{\lfloor \alpha |L| \rceil} \tag{2}$$

Submitted to the 35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks. Do not distribute.

Algorithm 2 Search Parameters for Affinity Kernel Based on Dataset Property

Require: All instance masks in the dataset M
Ensure: Kernel radius r_k, affinity neighbor gap g

Initialize list $R = [\,]$
for $m \in M$ **do**
  $C = ConnectedComponents(m)$
  $r_{max} = \max\left[\bigcup_{c \in C} GetCircleRadius(c)\right]$
  $R.append(r_{max})$
end for
$r_{min} = GetThreshold(R, \alpha = 0.2, \beta = 0.8)$
$g = \sqrt{2} r_{min}$
Initialize list $D = [\,]$
for $m \in M$ **do**
  $C = ConnectedComponents(m)$
  $C_{big} = \{c | c \in C, GetCircleRadius(c) \geq r_{min}\}$
  for $c \in C$ **do**
    $C'_{big} = C_{big} \setminus c$
    if $|C'_{big}| \geq 1$ **then**
      $d_{min} = \min\left[\bigcup_{c' \in C'_{big}} MinDistanceBetween(c, c')\right]$
      $D.append(d_{min})$
    end if
  end for
end for
$r_k = GetThreshold(D, \alpha = 0.8, \beta = 1.2)$



(a) (b) (c)

Figure 1: Example redundancy removal of affinity outputs. (a) The full centrosymmetric and axisymmetric affinity kernel. (b) The kernel that the centrosymmetric part is removed. (c) The kernel that both the centrosymmetric and axisymmetric parts are removed. The affinity between C and D can no longer be built.

1.3 Redundancy Removal of Affinity Outputs

As shown in Figure 4 in the paper, we shake off the redundancy of affinity outputs by removing the centrosymmetric part of the affinity kernel. Here we show why we can remove the centrosymmetric part but can not remove the axisymmetric one. As shown in Figure 1(a), regarding the kernel center C, point A and D are axisymmetric while A and B are centrosymmetric. In Figure 1(b), we show that when we remove the centrosymmetric part, including point B, we can still establish the affinity between C and B when the half kernel moves on the image and the kernel center reach B. However, if we further remove the axisymmetric part, as shown in Figure 1, we can no longer establish the affinity between point C and D even when the kernel center moves on to pixel D. Therefore, we can only remove half of the redundant affinity kernel, not limited to the horizontal or vertical direction.

2 Detailed ASIS Experimental

We use PSPNet [2] with Resnet-50 [3] as backbone and the weight is initialized with ImageNet [4] pretrained model. The input image is cropped to 512×512. And the image data augmentation includes horizontal and vertical flipping with a probability of 0.5, and random rotation within 360 degrees. In the training phase, we set $\lambda = 0.9$ in $\mathcal{L}$. In this way, a weight of 0.9 is assigned to the loss of affinity head.

All experiments are trained in 4 2080Ti GPUs and the batch sizes are set to 8. The stochastic gradient descent (SGD) solver is adopted in 50K iterations. The momentum is set to 0.9 and weight decay is set to 0.0005. The learning rate is initially set to 0.01 and adopt warm-up [5] strategy that the learning rate gradually rises during the first 625 iterations. After 625 iterations, the learning rate decreases linearly.

Similar to GMIS [1], we do graph merge sequentially from short-range to long-range affinity. And each stage has a different graph merge threshold according to the distance of the affinity neighbor. In particular, when $r \leq 4$ the threshold is set to 0.97, when $4 < r \leq 16$ the threshold is set to 0.7, and when $r > 16$ the threshold is set to 0.3.

3 Details of Synthetic Sub-datasets Creation

Table 1: Details of iShape-Branch creation.

Item	Description
Polygon Mesh	Blender's [6] add-on: Sapling Tree Gen.
Texture	Slight color change.
Background & Lighting	Choose randomly from [7] outdoor category.
Physic Engine	Off
Objects Number	10

Table 2: Details of iShape-Fence creation.

Item	Description
Polygon Mesh	Single fence 3D model.
Texture	Same texture in green color.
Background & Lighting	Choose randomly from [7].
Physic Engine	Rigid body with gravity.
Objects Number	Sample from $[2, 4]$

Table 3: Details of iShape-Hanger creation.

Item	Description
Polygon Mesh	Single hanger 3D model.
Texture	Same metal material.
Background & Lighting	Choose randomly from [7] indoor category.
Physic Engine	Rigid body with gravity.
Objects Number	Sample from $[25, 30]$

Table 4: Details of iShape-Log creation.

Item	Description
Polygon Mesh	Cylinders with same length but different radius.
Texture	Choose randomly from 5 wood textures from [8].
Background & Lighting	Choose randomly from [7] outdoor category.
Physic Engine	Rigid body with gravity.
Objects Number	Sample from $[70, 90]$

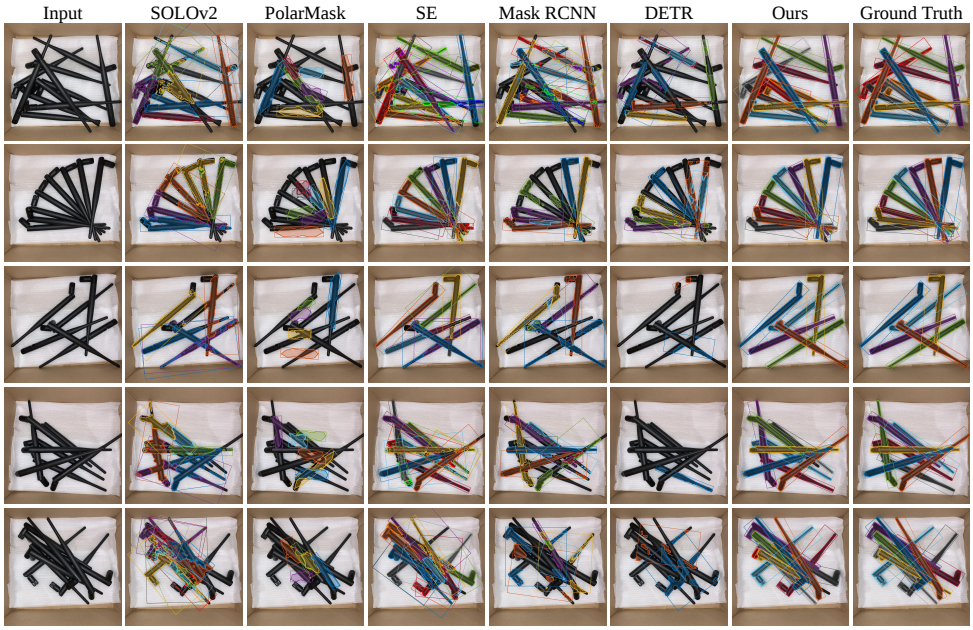
Table 5: Details of iShape-Wire creation.

Item	Description
Polygon Mesh	Single wire 3D model.
Texture	Choose color from red, green, and blue.
Lighting	Choose randomly from [7] indoor category.
Background	Random color plane.
Physic Engine	Soft body with gravity.
Objects Number	Sample from $[6, 8]$

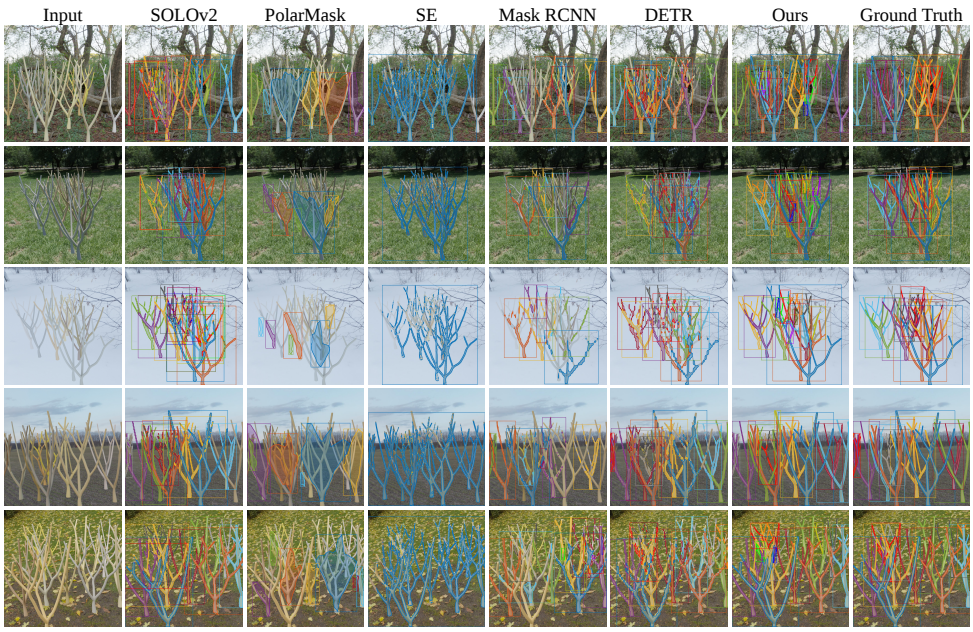
4 More Qualitative Results

We show additional qualitative results of all methods on iShape.

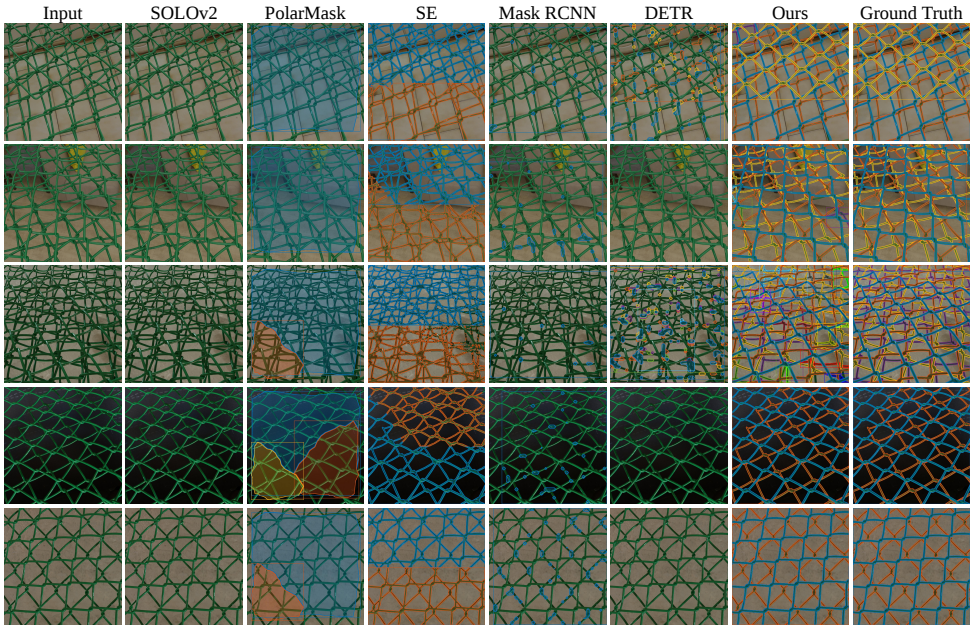
iShape-Antenna



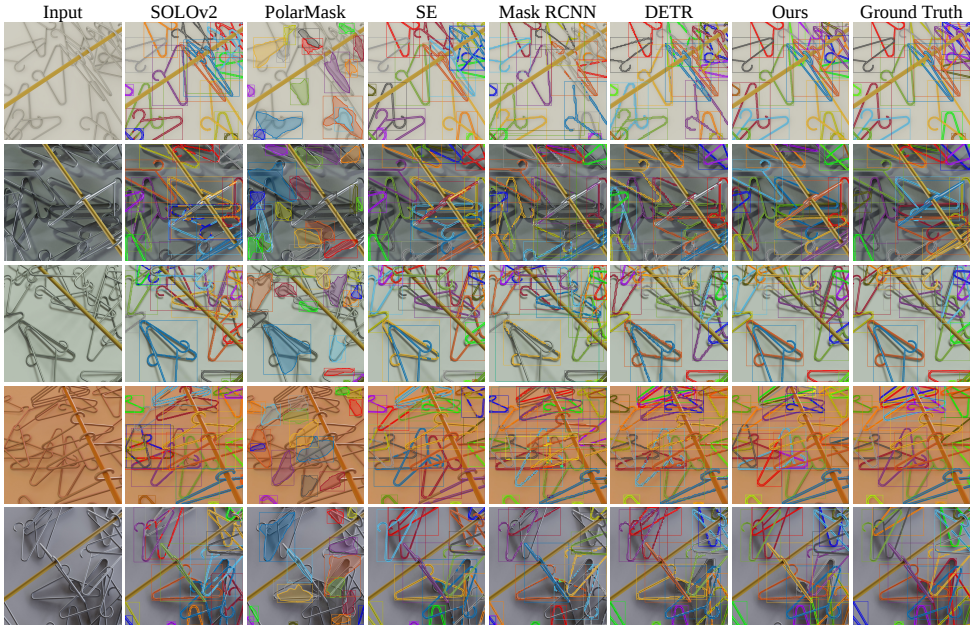
iShape-Branch



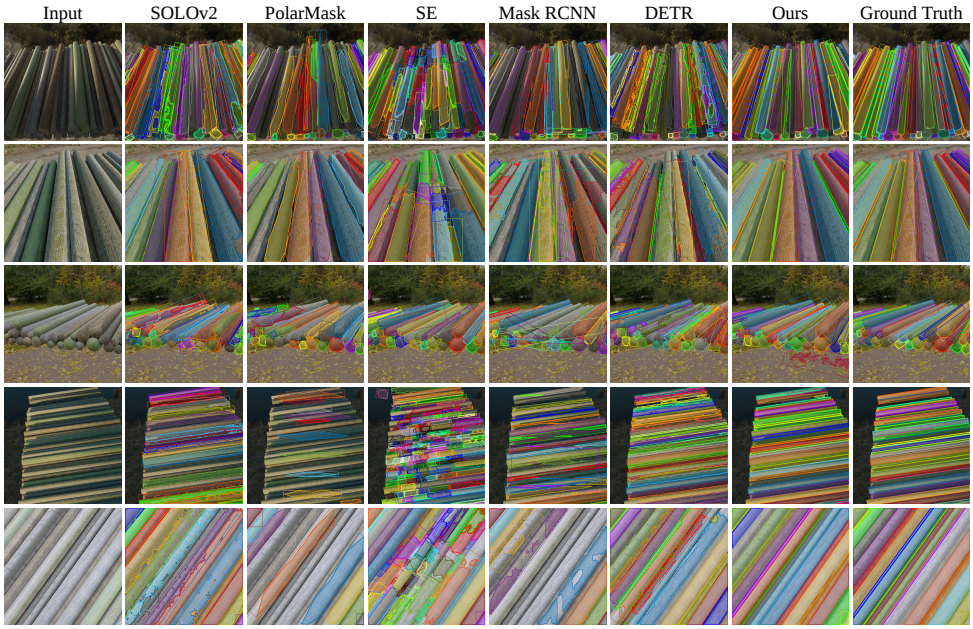
iShape-Fence



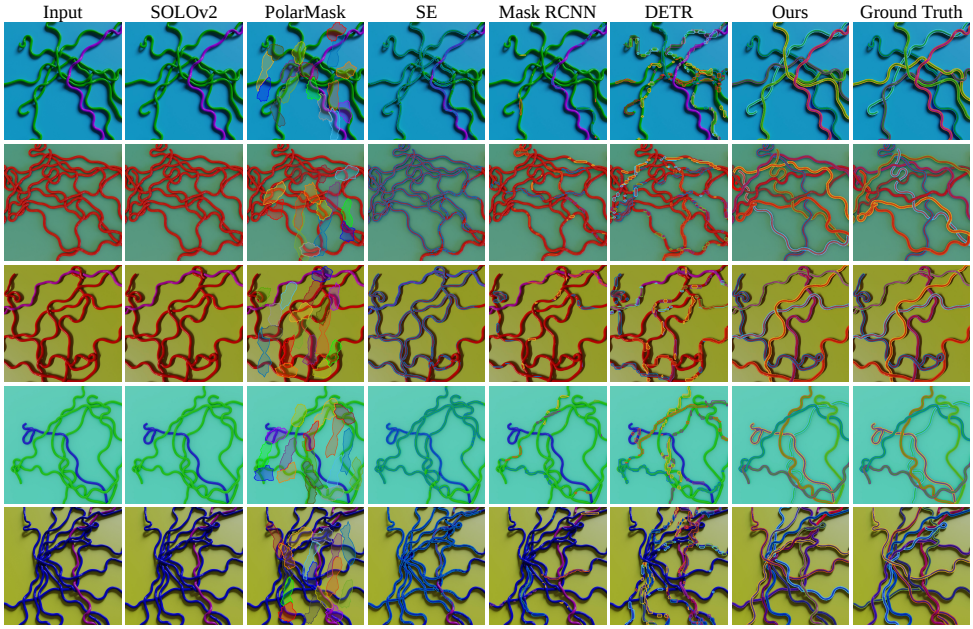
iShape-Hanger



iShape-Log



iShape-Wire



5 Misc

Dataset documentation. Dataset documentation is hosted on `https://ishape.github.io/`.

Intended uses. iShape focus on evaluating the performance of algorithms on the irregular shape

Statement. We bear all responsibility in case of violation of rights.

Hosting, licensing, and maintenance plan. iShape dataset is temporarily hosted on a private server. After camera ready version, we will release iShape on `https://www.kaggle.com/` under Public domain (CC0) license.

Data format. iShape provides both Cityscapes style and COCO style dataset format.

References

[1] Yiding Liu, Siyu Yang, Bin Li, Wengang Zhou, Jizheng Xu, Houqiang Li, and Yan Lu. Affinity derivation and graph merge for instance segmentation, 2018.

[2] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network, 2017.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[4] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015.

[5] Akhilesh Gotmare, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation, 2018.

[6] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2019.

[7] Rob Tuytel. Texturehaven. `https://hdrihaven.com/`.

[8] Greg Zaal. Hdrhaven. `https://texturehaven.com/`.