SharpContour: A Contour-based Boundary Refinement Approach for Efficient and Accurate Instance Segmentation

Chenming Zhu[1†], Xuanye Zhang[2†], Yanran Li[4], Liangdong Qiu[1,2], Kai Han[5], Xiaoguang Han[1,3‡]
[1]SSE, CUHK-Shenzhen [2]Shenzhen Research Institute of Big Data [3]FNii, CUHK-Shenzhen
[4]Birmingham University [5]The University of Hong Kong

Abstract

Excellent performance has been achieved on instance segmentation but the quality on the boundary area remains unsatisfactory, which leads to a rising attention on boundary refinement. For practical use, an ideal post-processing refinement scheme are required to be accurate, generic and efficient. However, most of existing approaches propose pixel-wise refinement, which either introduce a massive computation cost or design specifically for different backbone models. Contour-based models are efficient and generic to be incorporated with any existing segmentation methods, but they often generate over-smoothed contour and tend to fail on corner areas. In this paper, we propose an efficient contour-based boundary refinement approach, named SharpContour, to tackle the segmentation of boundary area. We design a novel contour evolution process together with an Instance-aware Point Classifier. Our method deforms the contour iteratively by updating offsets in a discrete manner. Differing from existing contour evolution methods, SharpContour estimates each offset more independently so that it predicts much sharper and accurate contours. Notably, our method is generic to seamlessly work with diverse existing models with a small computational cost. Experiments show that SharpContour achieves competitive gains whilst preserving high efficiency.

1. Introduction

Instance segmentation is a fundamental topic in computer vision which plays important role in scene understanding [45], intelligent robots [36], clinical analysis [6, 46] and autonomous driving [10, 14]. The mainstream of instance segmentation approaches follow the design of detection-then-segmentation framework like Mask R-CNN [13] and achieve promising performance. However, high fidelity segmentation of fine details especially in the

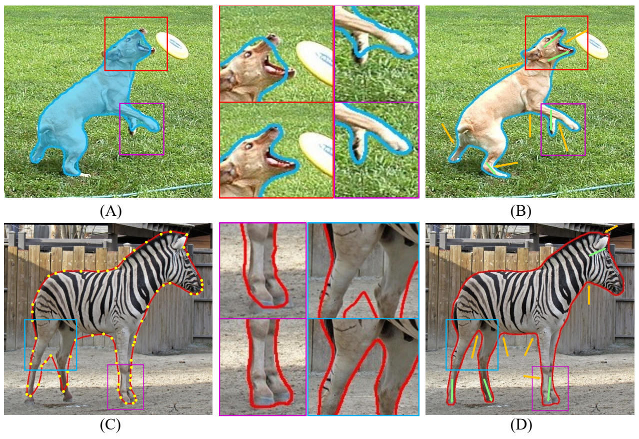

Figure 1. **Instance segmentation with SharpContour. Top**: A is the coarse mask predicted by Mask R-CNN and B is the refinement result of SharpContour. **Bottom**: C is the coarse contour generated by DANCE and D is the refinement result of SharpContour. In the corner areas, SharpContour yields significant improvements.

boundary area remains extremely challenging.

To address this issue, the refinement of boundaries is raised as a new topic in recent years. A novel evaluation metric called Boundary AP [7] and a number of approaches [8, 18, 33, 42] are proposed. Boundary AP [7] focuses on the accuracy of the object boundary region thus can better reflect boundary refinement. As for the approaches, refinement can be generally recognized as a post-processing operation which is expected to meet three basic requirements: accurate, efficient and generic. It is worth mentioning that "generic" is also a significant factor. However, existing top-performing refinement methods mainly focus on accuracy and lacks consideration of other two factors which are actually important in the practical application scenarios. For example, Boundary Patch Refinement method [33] proposed to additionally process patches along boundaries after the instance segmentation model, resulting in massive computational cost. RefineMask [42] designed a new architecture built upon Mask-RCNN [13] to refine the quality of instance mask with fine-grained features. Though the computational cost is not extremely expensive, this method is

[†]The two authors contribute equally to this paper.
[‡]Corresponding author: hanxiaoguang@cuhk.edu.cn

too specific to be applied to other types of instance segmentation models. It is intrinsically hard for the pixel-wise refinement methods to achieve the three requirements at once, since the dense pixel maps usually bring excessive computation cost.

Another group of trending segmentation approaches proposed contour-based segmentation methods [1, 3, 23, 27, 29, 38, 39] which directly process and generate sparse points along the boundaries. They are advantageous on efficiency and naturally generic to be appended after mask-based algorithms since it is straightforward to extract a contour from a mask. Thus, an interesting question can be raised up: " Is it possible to use contour-based methods to address the boundary refinement problem?"

However, existing contour-based methods, like DANCE [26] and DeepSnake [10], tend to produce over-smoothed contours especially on the sharp turner areas (as shown in Fig. 1) so that their performances on boundary areas is still lagging behind. We figure out the major reason which leads to the oversmoothing problem is that all the vertices on the contour are tied together by their feature learning strategies (e.g. circular convolutions [10]) and the smoothness regularization in their contour evolution process. As a result, a slight offset on one vertex will cause a wide chain reaction on all vertices of the contour. To address this, we elaborate a drastically different contour evolution process that estimates the deformation offsets independently for each vertex. Based on this, we propose an *accurate, efficient and generic* refinement approach, named *SharpContour*, for boundary area using contour-based representation.

Specifically, our SharpContour takes a coarse contour as input and deforms each vertex on the contour individually. To avoid artefacts caused by the involved "independency", we constrain each vertex to move along its normal direction and conduct the deformation in an iterative procedure, making the deformation stably. For the sake of balancing between efficiency and accuracy for each vertex's movement, instead of directly regressing the offset, we propose to determine the target position by performing classification on a few discrete points. Specifically, we first sample some points along the moving direction and classify them into "inner/outer" status to obtain the flipping position which indicates the boundary. The classifier plays a pivotal role in this contour evolution process. Thus, we carefully design an Instance-aware Point Classifier (IPC), which can predict "inner/outer" of a vertex with respect to different instance boundaries in the form of a probability score. There are two important factors for IPC to achieve high fidelity: 1) It is required to be instance-aware, which is essential to enable generating different results for a pixel with respect to different instances. For this purpose, the parameters of the IPC are predicted on-the-fly for each instance. 2) It should capture the information of the boundary details. Thus IPC takes the fine-grained feature, which is derived from the high-resolution feature map, as input to predict the vertex state.

With IPC, we can determine how to move each vertex during the deformation. The moving distance is further defined by the object size and IPC probability score. We iteratively adjust the vertices of the predicted contour, until they match the object boundary. In this process, SharpContour avoids the high-dependency over neighbouring vertices without using any smooth term, thus overcoming the over-smoothing issue. Notably, SharpContour only introduces a handful of parameters and involves a negligible amount of points in the calculation. Therefore, our approach is considerably efficient. Experiments manifest that SharpContour can work seamlessly with various segmentation models by introducing a small computational cost and producing high-quality object boundaries.

We validate the effectiveness of our SharpContour approach quantitatively and qualitatively on various large scale segmentation benchmarks. On COCO datasets [22], we use both AP and the Boundary AP metrics. Coupled with DANCE [26], SharpContour brings significant improvements of **1.5 AP** and **3.2 Boundary AP**. Coupled with the Mask R-CNN [13] and CondInst [34] as the refinement approach, our approach surpasses the baseline by a significant gain of **2.3, 2.2 AP** and **3.4, 3.3 Boundary AP** respectively. On Cityscapes datasets [9], SharpContour achieves **3.9 AP** boost when combined with Mask R-CNN [13]. Compared with other boundary refinement models, SharpContour consistently produces high-quality boundaries with a low computational cost. Taking the output of the state-of-the-art contour refinement method, RefineMask, as initial contour, SharpContour still gains **0.5 AP** and **1.1 Boundary AP**. The qualitative results further demonstrate the effectiveness of our approach.

2. Related Work

Mask-based methods. The mainstream mask-based approaches can be broadly categorized into top-down and bottom-up methods. The typical top-down approaches [2, 4, 5, 8, 13, 19, 24, 34, 37, 43] follow the pipeline of Mask-RCNN [13], which detects each instance first and predicts a pixel-wise binary map for each instance. Models can be categorized into two-stage [5, 8, 24] and one-stage [2, 4, 19, 34, 37, 40] according to their pipelines. However, they usually rely on a pooling operation to extract canonical-size features from the feature maps for each instance, which loses many boundaries details. CondINST [34] removed the RoI operation and use dynamic convolution layer to generate segmentation masks, but it still losses many details around boundaries. Bottom-up approaches [11, 17, 25, 28, 32, 44] first generate semantic segmentation and then cluster pixels

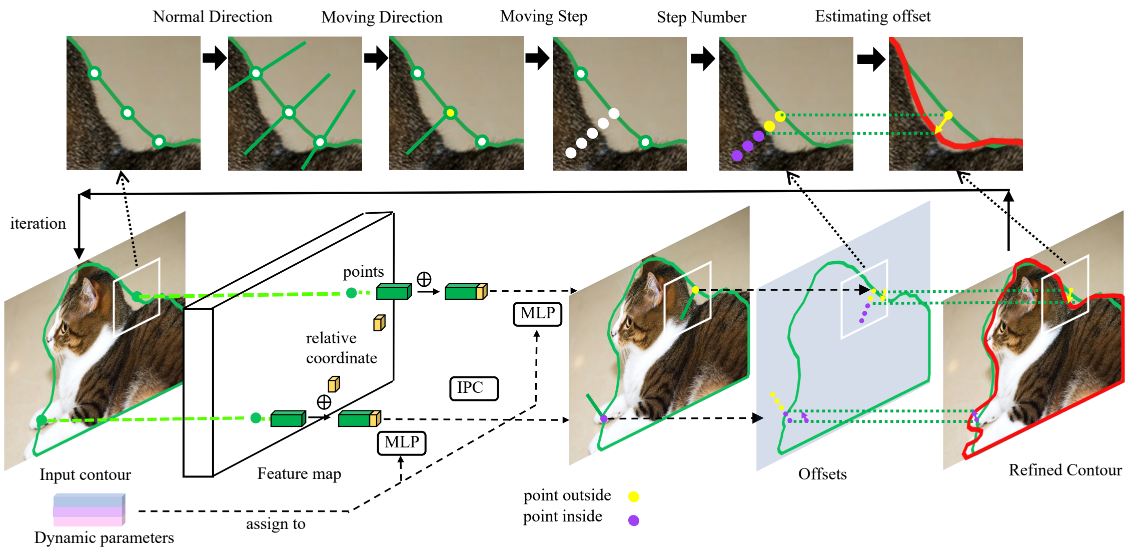

Figure 2. **Contour Evolution of SharpContour.** SharpContour obtains the initial contour from coarse segmentation results and deforms the contour to achieve boundary refinement. In the deformation process, 1) SharpContour obtains the normal direction. 2) SharpContour predicts the inner/outer state of vertex to decide negative/positive normal direction of deformation. 3) SharpContour decides the step size. 4) SharpContour obtains the moving step number based upon the position of the *flipping point*. 5) SharpContour estimates the offsets and deforms the contour.

into different instances. In all the pixel-wised methods, the boundary points only occupy a very small proportion (less than 1% [17]) so that their accuracy is usually over compromised during the optimization. Our method avoids these drawbacks and can work together with various models to enhance their performance on boundaries.

Boundary refinement To tackle the aforementioned limitations, various boundary refinement methods [8,18,20,33,41,42] are proposed. Meanwhile, to evaluate the boundary performance more accurate, a novel evaluation metric Boundary AP [7] is proposed, which can solve the desensitization problem of the current evaluation metric (AP) for object boundary region. As for the approaches, BPR [33] adopts a post-processing scheme, refining predicted instance boundary patches in detail to improve mask quality. Another top-performing model, PolyTransform [20] proposes the first contour-based refinement approach which uses the results of the mask-based model as initial contour and refines the contour by Transformer network [35]. These methods achieve superior performance while introducing a large computational cost and scarifying the efficiency. Some other methods try to balance the performance enhancement and the extra computational overheads. For example, PointRend [18] only performs point-based segmentation around the boundary blurred areas to obtain high-quality mask. RefineMask [42] achieves high-quality results by introducing fine-grained features during the upsampling process to refine entire objects. However, these two methods lack of generality since they can only refine the results of specific mask-based methods. In contrast, our refinement approach can not only achieve accurate and efficient refinement but also be generic to refine the results of both mask-based and contour-based models.

Contour-based methods Neural contour-based models [15,23,27,30,38,39] for segmentation are widely studied in recent years due to their potential advantage on efficiency. Most of them formulate the contour as polygons and propose various regression methods to estimate the coordinate of vertices on the polygons. [27, 39] propose CNN-based models which can learn high-level and expressive features for deformation. [23] proposes a contour-based model which utilizes Graph Convolution Networks (GCN) to regress the offsets of vertices on the contour. Following the pipeline of traditional Snake model [16], DeepSnake [29] proposes a two-stage contour evolution process and designs circular convolution to exploit the features on the contour. DANCE [26] follows the pipeline of DeepSnake [29] and introduces edge attention module and improves the matching scheme in the contour evolution process. DANCE [26] achieves state-of-the-art results for the contour-based instance segmentation methods. However, all of these regression-based contour evolution methods still suffer severe performance degradation in approximating the corners or cusps of instance. This is because their regression method needs to balance the regression error and the smoothness of the current contour and the offsets are highly dependent on each other during feature learning. Our approach elaborates a drastically different evolution method. In our deformation stage, the offset of each vertex on the polygon is estimated by discrete moving steps rather than regression. Incorporated with DANCE [26], our approach is proved to be more effective than all the existing contour-based methods, yielding accurate and high-fidelity segmentation results even for significant challenging areas.

3. Methodology

We propose a contour refinement approach, SharpContour, which coupled with existing instance segmentation models (either mask-based or contour-based), can produce high-quality boundary segmentation results. Let $C^{(0)}$: $\{x_i \mid i = 1, \ldots, N\}$ be an initial contour, which is obtained by off-the-shelf instance segmentation methods. The contour is defined by a sequence of vertices x_i. SharpContour iteratively moves the vertices to approach the actual boundary of an instance. To perform the contour evolution, we design an iteration process to deform the vertices along their normal. The key challenge of the contour evolution methods is to predict the offset for each vertex accurately. In contrast to existing methods that directly regress the offsets for all the vertices at once, which is often error-prone and leads to over-smoothed contours, we propose to adjust the vertices iteratively in a discrete manner inspired by [31]. Due to the inherent challenge of directly regressing offsets, we cast this problem to be a classification-based formula. Specifically, we propose an Instance-aware Point Classifier (IPC) $\varphi(x_i)$ which predicts a state for the vertex x_i indicating its relative position to the actual object boundary (i.e., inside or outside), such that we can determine whether a vertex should march along the positive or negative normal direction. Our approach greatly reduces the dependency on the offsets of neighbour vertices and allows for delicate segmentation details on complex and challenging boundary areas, such as the corner and cusps areas where the existing methods struggle. An overview of SharpContour can be found in Fig. 2.

3.1. Contour Evolution

Given a vertex x_i which is off the object boundary, its evolution process in one iteration can be written as

$$x_i' = x_i + m s_i d_i, \tag{1}$$

where d_i is the moving direction, s_i the moving step and m the step number. For each vertex, $\varphi(x_i)$ will output a scalar value in $[0, 1]$, indicating the probability of x_i being outside (i.e., $\varphi(x_i)$ = 1) or inside (i.e., $\varphi(x_i)$ = 0) the object. $\varphi(x_i)$ = 0.5 means that IPC is not sure whether the point is inside or outside the object, implying the point is probably located on the boundary. More details of IPC will be introduced in Sec. 3.2. Next, we describe how we obtain the moving step s_i and the step number m. They together define the moving distance.

To define the moving step, we take into account the object size (in terms of the area of the bounding box, denoted as A) and the uncertainty of the vertex state from IPC. Intuitively, if the object is larger, the step size should also be

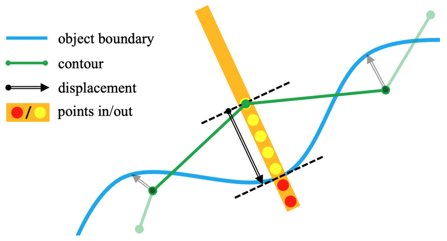

Figure 3. **Deformation of SharpContour.** Value flipping indicates the position of instance boundary. SharpContour uses the position of flipping points to update the contour.

larger, and vice versa. If the uncertainty of vertex is higher, the step size should be smaller to reach finer prediction, and vice versa. To reflect these, concretely, we define the step size as $s_i = \lambda\sqrt{A}|\varphi(x_i) - 0.5|$, where λ is an empirical deformation ratio which we set to 0.003 in experiments. $|\varphi(x_i) - 0.5|$ indicates the uncertainty of state of x_i for IPC, the closer $\varphi(x_i)$ is to 0.5, the higher the uncertainty of state of x_i for IPC, so smaller step size for x_i should be assigned for accuracy. For the moving step number, ideally, we expect the vertex to progressively move towards the actual object boundary. Therefore, we move the vertex along d_i step-by-step, and after each move, we examine the value of φ on the current location. If we reach a location that φ indicates the vertex is moved from inside to outside of the boundary or the other way around, then we stop. We call this location *flipping point*. The number of moves from the original location to the flipping point is then our moving step number m. To avoid the vertex moving towards an improper direction too far away, we set an upper bound for the move, i.e., for each move if after M steps, a flipping point is still not reached, we then set $m = M$.

Iteratively Evolution We perform the above evolution process for each vertex of $C^{(0)}$ to obtain an updated contour $C^{(1)}$. We then run another evolution cycle for $C^{(1)}$ to update the vertices. We iteratively run the evolution process and generate a series of contours $\{C^{(0)}, C^{(1)}, \ldots, C^{(n)}, \ldots\}$ to approach actual object boundary progressively. Note that during each iteration, if a vertex reaches a *flipping point*, which indicates that it has reaches the object boundary, it will not participate the subsequent deformation process anymore. This further improves the efficiency of SharpContour.

3.2. Instance-aware Point Classifier

Here, we introduce our Instance-aware Point Classifier (IPC) φ, which aims at predicting a state for a given vertex x_i to tell its relative position to the actual object boundary. In a nutshell, IPC takes the fine-grained feature of x_i derived from the high-resolution feature map and the relative location of x_i to its corresponding bounding box as input to predict the probability indicating whether x_i is located inside or outside the object. Importantly, the parameters of φ

are dynamically predicted for each instance on the fly, such that φ is instance-aware, which plays an important role at inferring vertices relative location to the instance.

Fine-grained Feature To be able to identify the subtle details for the vertex evolution, we employ the fine-grained features from the high-resolution feature map produced by the instance segmentation backbone as the input for φ. We append a convolution layer on the high-resolution feature map to further encode feature to benefit φ for contour refinement and reduce the dimension. Let the encoded fine-grained feature for x_i be f_i.we concatenate f_i with the relative coordinates c_i of x_i to the instance bounding box, forming a new fine-grained location-aware feature vector $f_i^c = [f_i;\ c_i]$, which serves as the input of φ to predict the state probability. Taking Mask R-CNN as example, we utilize the highest resolution feature map from the feature pyramid network (FPN), which is $1/4$ of the image size. To decrease the parameters number of IPC and enhance the utilization of the high-resolution feature, we attach a 3×3 convolution layer to this feature map and reduce its channel from 256 to 16. From the resulting feature map, we extract the fine-grained feature for each vertex.

Instance-aware Dynamic Parameters Only equipped with the fine-grained feature, the classifier still struggles at the determination. For example, a point with the same feature can be outside one instance but inside another instance. To deal with this problem, our classifier needs to be instance-aware and has strong ability to grasp holistic information to identify each instance. Inspired by [34], we propose to dynamically *predict* the parameters θ of the classifier φ for each instance, based on the feature of each instance. This strategy can capture strong instance-aware information therefore it can discriminate the state of a vertex w.r.t different instance. For example, we incorporate a boundary controller head inspired by [34] for Mask R-CNN to obtain the dynamic parameters for each instance. The boundary controller head is a very compact and lightweight network with three fully connected layers. The output dimension of this controller head is the same with the parameter number of IPC. By sharing feature with the original mask head of Mask R-CNN, we can make better use the instance-aware feature while only introducing a small computation cost. Our IPC can therefore be written as $\varphi(x_i) = \varphi_\theta([f_i, c_i])$. If $\varphi(x_i) > 0.5$, x_i is considered as outside the object predicting the label $\hat{y}_i$ as 1, otherwise inside the object predicting the label $\hat{y}_i$ as 0. In our implementation, IPC is simply realized an MLP with three hidden layers with the ReLU activation. The sigmoid activation applied to its output layer. Hence, the extra cost introduced by the IPC is marginal.

3.3. Instance Segmentation with SharpContour

In this section, we describe how to combine our SharpContour with different types of instance segmentation models, including mask-based models and contour-based models. In the training stage, we train the head which generates parameters of IPC by randomly sampling pixels near the ground-truth instance boundary. In the inference stage, SharpContour obtains the initial contour from the coarse segmentation results and the contour evolution process can be iterated for refinement. In general, when working with existing instance segmentation methods, the SharpContour takes three inputs: initial contour, fine-grained features and dynamic parameters. Methods to obtain these three inputs vary from different instance segmentation methods. Here we introduce how to apply SharpContour to three typical instance segmentation methods. For contour-based models such as DANCE [26], the predicted contour of their model can be used directly as the initial contour for our SharpContour. For Mask-based models such as Mask R-CNN [13] and CondInst [34], we extract the initial contour by transferring the predicted mask into contour representation. To obtain the fine-grained features, we use convolution layers to encode the feature maps of highest resolution from their backbone and use the encoded feature map to extract fine-grained features. To obtain the dynamic parameters, we tailor the boundary controller head to incorporate it with various instance segmentation models. The detailed network structure and parameters settings are provided in the supplementary.

3.4. Loss Function

We use the focal loss [21] to train the boundary controller head which generates parameters of IPC and the convolution layer which encodes the fine-grained features. Since the points sampled around ground-truth boundary for training is random, the ratio between positive and negative samples is not fixed. To compensate this, we exploit the focal loss with a dynamic coefficient α, which is the ratio of current positive and negative samples. The loss can be written as

$$L_{IPC} = \begin{cases} -\alpha(1-\hat{y}_i)^\gamma log(\hat{y}_i),\, y_i = 1 \\ -(1-\alpha)(\hat{y}_i)^\gamma log(1-\hat{y}_i),\, y_i = 0 \end{cases} \tag{2}$$

where $\hat{y}$ denotes the predicted label, α is dynamically determined by the proportion of positive and negative samples in the current batch and γ is the difficulty factor which we set to 2 in our experiments.

After combining our SharpContour with a baseline instance segmentation model, we jointly train our SharpContour and with the coupled instance segmentation model. The overall loss function can be expressed as

$$L = L_s + \mu L_{IPC}, \tag{3}$$

where L_s is the original loss of the instance segmentation model. We set the weight $\mu = 10$ in our experiments.

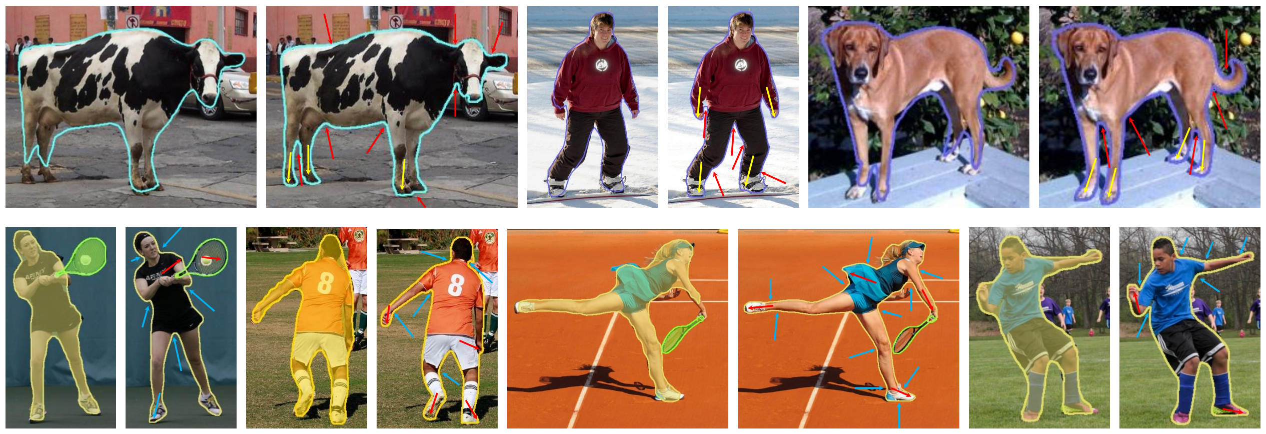

Figure 4. **Qualitative results on COCO dataset.** We use SharpContour to refine the segmentation results of different models. The top line is the results of DANCE while the bottom line is the results of Mask-RCNN. For each example, the left is the result before refinement while the right is our result. As is illustrated, SharpContour can refine the segmentation results near instance boundary.

	AP_{dev}	AP_{val}	Boundary AP	AP*	$AP*_S$	$AP*_M$	$AP*_L$	FPS	Inference time (ms/per image)
Mask R-CNN	34.6	34.7	-	36.8	22.6	43.7	52.0	12.3	81.3
Mask R-CNN*	35.5	35.2	21.2	37.6	22.8	44.7	53.8	17.5	57.1
DANCE	34.6	34.5	20.2	-	-	-	-	16.5	60.6
CondInst	35.4	35.7	21.6	-	-	-	-	17.4	57.5
PointRend	36.8	36.3 (+1.1)	23.5(+2.3)	39.7	22.9	46.7	57.4	13.0	76.9 (+19.8)
RefineMask	37.6	37.3 (+2.1)	24.7(+3.5)	40.9	24.1	48.8	58.0	13.0	76.9 (+19.8)
DANCE + SharpContour	36.3	36.1 (+1.5)	23.9(+3.7)	-	-	-	-	13.6	73.5
Mask R-CNN* + SharpContour	37.7	37.5 (+2.3)	24.6(+3.4)	41.2	24.2	49.1	58.5	15.0	66.7 (+9.6)
CondInst + SharpContour	37.7	**37.9** (+2.2)	24.9(+3.3)	-	-	-	-	15.4	64.9
RefineMask + SharpContour	**38.0**	37.8 (+0.5)	**25.8(+1.1)**	**41.9**	**24.3**	**49.4**	**59.1**	12.1	82.6 (+5.7)

Table 1. **Comparisons on COCO val2017 and test-dev.** AP_{dev} denotes the evaluation results on $test-dev$, and other columns denotes the evaluation results on $val2017$. "Mask R-CNN" is a original Mask R-CNN, and "Mask R-CNN*" is the improved version in Detectron2. All methods are trained with 1x schedule using R50-FPN backbone. The FPS is measured on a single Tesla V100 GPU. SharpContour brings significant AP enhancement for DANCE, Mask R-CNN and CondInst. Moreover, SharpContour can achieve competitive performance compared with other boundary refinement approaches with the highest efficiency.

Method	AP	AP_S	AP_M	AP_L
Mask R-CNN	33.8	12.0	31.5	51.8
PointRend	35.8(+2.0)	-	-	-
RefineMask	37.6(+3.8)	**14.6**	34.0	58.1
Mask R-CNN + SharpContour	**37.7(+3.9)**	14.4	**34.2**	**58.3**

Table 2. **Results on Cityscapes validation set.** The training setting for all models are same: trained on fine annotations for 64 epochs, using multi-scale training and ResNet-50 with FPN.

4. Experiments

4.1. Impementation Details

Training strategy We train the SharpContour with the DANCE, Mask R-CNN and the CondInst together to enhance the performance. For each combination model, we conduct experiments using totally the same training settings with coupled model, including training epoch, learning rate schedule, data augmentation methods, etc, to ensure fairness. We set the deformation ratio $\lambda = 0.003$, the number of sampling points $M = 10$ and the resolution of the polygon $N = 128$. The iteration number of evolution is set to be 3. More detailed training settings will be described in *supp*.

4.2. Benchmarks and Metrics

We evaluate the efficiency and the effectiveness of SharpContour on three standard benchmarks, which are Cityscapes [9] and Microsoft COCO [22]. Following the prior work, we use Mask AP as the evaluation metrics. To further demonstrate the effectiveness of SharpContour, we report Boundary AP on the COCO datasets. (Unless explicitly stated, AP denotes Mask AP in this paper.)

COCO [22] is one of the most common benchmarks to evaluate the models for object detection and segmentation tasks, and we mainly report the results on COCO. Our models were trained on train2017. Following PointRend and RefineMask, we also report AP*, which evaluates the COCO categories using LVIS annotations, since LVIS annotations have much higher quality mask. The results of AP* are obtained by the same model trained on COCO.

Cityscapes [9] is a large-scale dataset for pixel-level and instance-level segmentation evaluation, which provides a massive amount of video records of daytime urban street scenes from 50 cities. Cityscapes contain a rich set of annotations including $5,000$ images with fine pixel-level annotations and $20,000$ images with coarse annotations. It is one of the most widely used and challenging benchmarks.

4.3. Comparison with the state-of-the-art

Effectiveness for contour-based model We apply SharpContour to the DANCE [26], which follows the idea of DeepSnake [29] and achieves current state-of-the-art performance for contour-based instance segmentation meth-

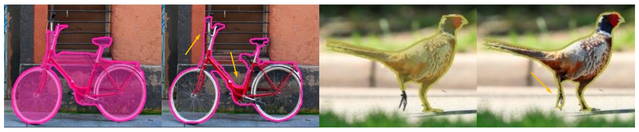

Figure 5. **Qualitative comparison results of complex contours** SharpContour+MaskRCNN (Right) performs better than RefineMask(Left) at challenging regions.

Training Scheme	AP	AP*
joint training	37.5(+2.3)	41.2(+3.6)
training alone	36.2(+1.0)	39.8(+2.2)

Table 3. **Results of Different training scheme on COCO val2017.** The performance is reported on "Mask R-CNN + SharpContour" combination.

	Mask R-CNN	Mask R-CNN$^+$	Mask R-CNN+SharpContour
AP	35.2	35.5	37.5
AP^*	37.6	38.0	41.2

Table 4. **Results on COCO val2017**. Mask R-CNN$^+$ is the results of Mask R-CNN trained with SharpContour. It demonstrates that original model can get benefits from joint training, and our proposed module indeed improve the mask quality.

ods. We conduct experiments on the COCO datasets. As is shown in the Tab. 1, SharpContour brings 1.5AP and 3.7 Boundary AP improvement.

Effectiveness for Mask-based methods We apply SharpContour to the Mask R-CNN [13] and the CondInst [34] and complete experiments on the COCO and Cityscapes datasets. On COCO datasets, we further report the AP* measured using the higher quality LVIS [12] annotations. 1) For Mask R-CNN [13], SharpContour outperforms baseline model 2.3AP, 3.4 Boundary AP and 3.6AP* on COCO datasets (Tab. 1) and yields 3.9AP enhancement on CityScapes datasets(Tab. 2). 2) For CondInst [34], SharpContour achieves an improvement of 2.2AP and 3.3 Boundary AP on the COCO [22] datasets. Besides, 'RefineMask + SharpContour' combination in Tab. 1 represents: we directly run 'Mask R-CNN + SharpContour' model on the contour simply extracted from the output mask of RefineMask, it still gains 0.5AP and 1.1 Boundary AP improvement. SharpContour gains larger improvements on Cityscapes compared with COCO. The probable reason is that the classes of Cityscapes is much less than COCO, which leads to a better IPC. This further proves the effectiveness of SharpContour.

Efficiency We compare the efficiency with other boundary refinement methods, including PointRend [18] and RefineMask [42], on the COCO and Cityscapes dataset. For fairness, 1) These methods are built on Mask R-CNN. 2) All the methods adopt ResNet-50 with FPN as the backbone, 3) All the models are trained for the same epochs. As is shown in Tab. 1, 2, comparing with PointRend [18]and RefineMask [42], SharpContour achieves better performance with higher efficiency. Notably, PointRend [18] and RefineMask [42] incorporate their methods after Mask R-CNN and introduce additional inference time of 19.8ms. Compared with them, our SharpContour only introduce inference time of 9.6ms, half of theirs, but brings more gains.

Qualitative Comparison Fig. 4 shows the qualitative results of SharpContour on the COCO datasets. We use SharpContour to refine two models: DANCE (top line) and Mask R-CNN (bottom line). For each instance in these figures, the left one is the result without the SharpContour while the right one is the result with the SharpContour. As is illustrated, the SharpContour can ameliorate the contour near instance boundary since it can extract extra information in vertex neighbourhood so that achieve high-fidelity and accurate results for diverse instances. More results can be found in the supplementary materials. Fig.5 shows qualitative comparison results between RefineMask and SharpContour+MaskRCNN. SharpContour can deal with objects with complex contour and perform better at challenging regions (e.g., thin structure region).

4.4. Ablation Study

We conduct ablation experiments to analyze the proposed SharpContour on the COCO datasets. We explore the influence of different choices of SharpContour.

Training Scheme To demonstrate the effectiveness of SharpContour in advance, we also adopt the training scheme which freezes all parameters of the instance segmentation models and only trains the SharpContour independently. Specifically, we only train the boundary controller head (Sec. 3.2) used to generate parameters of the IPC and the convolution layer (Sec. 3.2) used to generate fine-grained feature. As shown in Tab. 3, SharpContour can still refine the instance segmentation results of these models by adopting such training scheme.

Effectiveness for feature extraction SharpContour shares the backbone with the instance segmentation model to refine. We explore what effects such training strategy casts to the feature extraction process of the backbone. We examine the instance segmentation accuracy of the model w/o training with the SharpContour. As illustrated in Tab. 4, Training with SharpContour can actually enhance the performance of the instance segmentation models, which indicates that SharpContour can ameliorate feature extraction process.

Effects of dynamic step size The output probability of the IPC indicates the uncertainty of inner/outer state. We argue that such uncertainty can be used to better control the moving distance in contour evolution process. We conduct ablation experiments w/o such dynamic step size on COCO datasets. As reported in Tab. 5, the dynamic step size yields 0.5AP and 0.9AP* enhancements. Moreover, we notice that dynamic step size can also stable the inference process.

Larger models, longer training We train SharpContour together with Mask-RCNN using different backbones, including ResNet-101 with FPN and ResNeXt-101 with FPN, with longer 3x schedule. In the Tab. 6, SharpContour consistently improves the performance of baseline model.

adaptive step	AP	AP*
w	37.5	41.2
w/o	37.0(-0.5)	40.3(-0.9)

Table 5. **Effectiveness of adaptive step on COCO val2017**. When combining Mask R-CNN with our SharpContour, the adaptive step strategy can bring significant 0.5AP and 0.9AP* improvements, which can also stable the inference process.

	backbone	AP	AP*
Mask R-CNN	R50-FPN	37.2	39.5
Mask R-CNN + SharpContour	R50-FPN	**39.3(+2.1)**	**43.1(+3.6)**
Mask R-CNN.	R101-FPN	38.6	41.4
Mask R-CNN + SharpContour	R101-FPN	**40.8(+2.2)**	**45.2(+3.8)**
Mask R-CNN.	X101-FPN	39.5	42.1
Mask R-CNN + SharpContour	X101-FPN	**41.8(+2.3)**	**46.0(+3.9)**

Table 6. **Larger models and longer 3x schedule**. Even with a stronger backbone (ResNet-101) and a longer training schedule (3x), a consistent improvement brought by SharpContour is achieved as well. AP* is COCO mask AP evaluated using the higher-quality LVIS annotations.

λ	AP_{val}	M	AP_{val}	FPS	N	AP_{val}	FPS
0.0015	37.1	5	36.8	15.4	128	**37.5**	**15.0**
0.003	**37.5**	10	**37.5**	**15.0**	256	37.8	14.6
0.006	36.6	15	37.2	14.7	348	38.0	14.2
		20	37.1	14.3	512	38.0	13.7

Table 7. AP_{val} **and FPS for different parameters of sampling strategy on COCO val2017 datasets**. The parameters selected achieve the balance of accuracy and efficiency.

Combined Method	n=1	n=2	n=3	n=4
DANCE	35.2	35.7	36.1	36.2
Mask R-CNN	36.6	37.1	37.5	37.7
CondInst	36.9	37.4	37.9	38.0

Table 8. **Results of different iteration numbers of Evolution Process.** It can be seen that the Iterative Evolution can consistently boost the performance, as well as preserve stability on the COCO val2017 datasets. As the number of iterations increases, the improved performance of each iteration continues to decrease, indicating the convergence of contour.

	Reg.1	Reg. 2	SharpContour
AP	36.1(+0.9)	36.2(+1.0)	**37.5(+2.3)**

Table 9. **Results of Regression-based model** Both regression-based designs can only slightly enhance the AP.

Different parameters of Contour Evolution process There are three main parameters in the design of the points sampling strategy, which are the deformation ratio λ, the number of sampling points M and the resolution of the polygon N. In final version, we set $\lambda = 0.003$, $M = 10$ and $N = 128$. In this ablation study, we conduct control experiments for each one of these parameters (preserving the other two parameters the same with the final version) and the results of three evolution iterations are in Tab. 7.

For λ, a) a larger deformation ratio causes performance degradation with more iterations, which is due to the fast deformation; b) a smaller deformation ratio requires more iterations for convergence. From these two experiments, we can see that our final choice achieves the best performance.

For M, a) a larger number of points will increase the inference time; b) a smaller number of points will decrease the running time but require more iterations for better performance; c) if the number of points becomes very large, the performance will decrease as more iterations are performed. This is because the polygon deforms too fast so that errors tend to appear in the result.

For N, it shows that denser sampling on the polygon contributes to minor performance improvements, while greatly increasing the inference time. We use 128-resolution in the final model for a better trade-off between accuracy and efficiency.

Different number of Stacked Contour Evolution Process To comprehensively evaluate the effectiveness of the SharpContour, we explore the performance of different iterations of the Contour Evolution Process. The results are reported in Table. 8. As is shown, if stacking the contour evolution process of SharpContour more times, a stable enhancement of accuracy is validated for our methods.

Regression-based Baselines We have explored two regression-based designs before adopting current discretization design: 1) regressing offset vectors (**Reg.1**); 2)regressing the distance along the normal direction (**Reg. 2**), where the proposed instance-aware feature is also used. We adopted the Mask-RCNN as the basic framework and carried out experiments on COCO datasets. As is shown in Tab. 9, both regression-based designs can only slightly enhance the AP comparing with the SharpContour.

5. Conclusion

We propose a novel contour-based refinement approach called SharpContour to improve the boundary quality of instance segmentation. The existing mask-based refinement methods are lack of efficiency or generality and contour-based approaches prone to generate over-smoothed contours around sharp corners. We address all of their limitations by designing a new contour evolution method and an Instance-aware Points Classifier. In contrast to the previous approaches, our approach shows superior accuracy in a considerable efficient fashion and can refine both mask-based and contour-based methods. We extensively evaluate the qualitative and quantitative performance of SharpContour on two large-scale public benchmarks for instance segmentation. It achieves significant improvement on all benchmarks when incorporated with different models.

Acknowledgement The work was supported in part by the Basic Research Project No. HZQB-KCZYZ-2021067 of Hetao Shenzhen-HK S&T Cooperation Zone, National Key R&D Program of China with grant No. 2018YFB1800800, by Shenzhen Outstanding Talents Training Fund 202002, and by Guangdong Research Projects No. 2017ZT07X152 and No. 2019CX01X104. This work was also supported by NSFC-62172348, 61902334, Shenzhen General Project (JCYJ20190814112007258) and Shenzhen Sustainable Development Project(KCXFZ20201221173008022). We thank the ITSO in CUHKSZ for their High-Performance Computing Services.

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 859–868, 2018. 2

[2] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9157–9166, 2019. 2

[3] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5230–5238, 2017. 2

[4] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. Blendmask: Top-down meets bottom-up for instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8573–8581, 2020. 2

[5] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4974–4983, 2019. 2

[6] Xu Chen, Bryan M Williams, Srinivasa R Vallabhaneni, Gabriela Czanner, Rachel Williams, and Yalin Zheng. Learning active contour models for medical image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11632–11640, 2019. 1

[7] Bowen Cheng, Ross Girshick, Piotr Dollár, Alexander C Berg, and Alexander Kirillov. Boundary iou: Improving object-centric image segmentation evaluation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15334–15342, 2021. 1, 3

[8] Tianheng Cheng, Xinggang Wang, Lichao Huang, and Wenyu Liu. Boundary-preserving mask r-cnn. In *European conference on computer vision*, pages 660–676. Springer, 2020. 1, 2, 3

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 2, 6

[10] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 2020. 1, 2

[11] Naiyu Gao, Yanhu Shan, Yupei Wang, Xin Zhao, Yinan Yu, Ming Yang, and Kaiqi Huang. Ssap: Single-shot instance segmentation with affinity pyramid. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 642–651, 2019. 2

[12] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5356–5364, 2019. 7

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1, 2, 5, 7

[14] Markus Hofmarcher, Thomas Unterthiner, José Arjona-Medina, Günter Klambauer, Sepp Hochreiter, and Bernhard Nessler. Visual scene understanding for autonomous driving using semantic segmentation. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 285–296. Springer, 2019. 1

[15] Saumya Jetley, Michael Sapienza, Stuart Golodetz, and Philip HS Torr. Straight to shapes: real-time detection of encoded shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6550–6559, 2017. 3

[16] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988. 3

[17] Alexander Kirillov, Evgeny Levinkov, Bjoern Andres, Bogdan Savchynskyy, and Carsten Rother. Instancecut: from edges to instances with multicut. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5008–5017, 2017. 2, 3

[18] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9799–9808, 2020. 1, 3, 7

[19] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13906–13915, 2020. 2

[20] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9131–9140, 2020. 3

[21] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2, 6, 7

[23] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. Fast interactive object annotation with curve-gcn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5257–5266, 2019. 2, 3

[24] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018. 2

[25] Yiding Liu, Siyu Yang, Bin Li, Wengang Zhou, Jizheng Xu, Houqiang Li, and Yan Lu. Affinity derivation and graph merge for instance segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 686–703, 2018. 2

[26] Zichen Liu, Jun Hao Liew, Xiangyu Chen, and Jiashi Feng. Dance: A deep attentive contour model for efficient instance segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 345–354, 2021. 2, 3, 5, 6

[27] Diego Marcos, Devis Tuia, Benjamin Kellenberger, Lisa Zhang, Min Bai, Renjie Liao, and Raquel Urtasun. Learning deep structured active contours end-to-end. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8877–8885, 2018. 2, 3

[28] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. *arXiv preprint arXiv:1611.05424*, 2016. 2

[29] Sida Peng, Wen Jiang, Huaijin Pi, Xiuli Li, Hujun Bao, and Xiaowei Zhou. Deep snake for real-time instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8533–8542, 2020. 2, 3, 6

[30] Christian Rupprecht, Elizabeth Huaroc, Maximilian Baust, and Nassir Navab. Deep active contours. *arXiv preprint arXiv:1607.05074*, 2016. 3

[31] Andrei Sharf, Thomas Lewiner, Ariel Shamir, Leif Kobbelt, and Daniel Cohen-Or. Competing fronts for coarse–to–fine surface reconstruction. In *Computer Graphics Forum*, volume 25, pages 389–398. Wiley Online Library, 2006. 4

[32] Konstantin Sofiiuk, Olga Barinova, and Anton Konushin. Adaptis: Adaptive instance selection network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7355–7363, 2019. 2

[33] Chufeng Tang, Hang Chen, Xiao Li, Jianmin Li, Zhaoxiang Zhang, and Xiaolin Hu. Look closer to segment better: Boundary patch refinement for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13926–13935, 2021. 1, 3

[34] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. *arXiv preprint arXiv:2003.05664*, 2020. 2, 5, 7

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 3

[36] Weiwei Wan, Kensuke Harada, and Fumio Kanehiro. Planning grasps with suction cups and parallel grippers using superimposed segmentation of object meshes. *IEEE Transactions on Robotics*, 2020. 1

[37] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. Solo: Segmenting objects by locations. In *European Conference on Computer Vision*, pages 649–665. Springer, 2020. 2

[38] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12193–12202, 2020. 2, 3

[39] Wenqiang Xu, Haiyang Wang, Fubo Qi, and Cewu Lu. Explicit shape encoding for real-time instance segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5168–5177, 2019. 2, 3

[40] Hui Ying, Zhaojin Huang, Shu Liu, Tianjia Shao, and Kun Zhou. Embedmask: Embedding coupling for one-stage instance segmentation. *arXiv preprint arXiv:1912.01954*, 2019. 2

[41] Yuhui Yuan, Jingyi Xie, Xilin Chen, and Jingdong Wang. Segfix: Model-agnostic boundary refinement for segmentation. In *European Conference on Computer Vision*, pages 489–506. Springer, 2020. 3

[42] Gang Zhang, Xin Lu, Jingru Tan, Jianmin Li, Zhaoxiang Zhang, Quanquan Li, and Xiaolin Hu. Refinemask: Towards high-quality instance segmentation with fine-grained features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6861–6869, 2021. 1, 3, 7

[43] Rufeng Zhang, Zhi Tian, Chunhua Shen, Mingyu You, and Youliang Yan. Mask encoding for single shot instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10226–10235, 2020. 2

[44] Ziyu Zhang, Alexander G Schwing, Sanja Fidler, and Raquel Urtasun. Monocular object instance segmentation and depth ordering with cnns. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2614–2622, 2015. 2

[45] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 267–283, 2018. 1

[46] Yanhao Zhu, Zhineng Chen, Shuai Zhao, Hongtao Xie, Wenming Guo, and Yongdong Zhang. Ace-net: Biomedical image segmentation with augmented contracting and expansive paths. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 712–720. Springer, 2019. 1