Contour-based Interactive Segmentation

Polina Popenova , **Danil Galeev** , **Anna Vorontsova** and **Anton Konushin**

Samsung Research

{p.popenova, d.galeev, a.vorontsova, a.konushin}@samsung.com

Abstract

Recent advances in interactive segmentation (IS) allow speeding up and simplifying image editing and labeling greatly. The majority of modern IS approaches accept user input in the form of clicks. However, using clicks may require too many user interactions, especially when selecting small objects, minor parts of an object, or a group of objects of the same type. In this paper, we consider such a natural form of user interaction as a loose contour, and introduce a contour-based IS method. We evaluate the proposed method on the standard segmentation benchmarks, our novel UserContours dataset, and its subset UserContours-G containing difficult segmentation cases. Through experiments, we demonstrate that a single contour provides the same accuracy as multiple clicks, thus reducing the required amount of user interactions.

1 Introduction

IS aims to segment an arbitrary object in an image according to a user request. IS has numerous applications in image editing and labeling: it can significantly speed up labeling images with per-pixel masks and ease the burden of annotating large-scale databases [Acuna *et al.*, 2018; Agustsson *et al.*, 2019a; Benenson *et al.*, 2019]. In graphical editors, IS might allow users selecting objects of interest to manipulate them.

Recent IS works [Hao *et al.*, 2021; Jang and Kim, 2019; Sofiiuk *et al.*, 2020; Sofiiuk *et al.*, 2022; Xu *et al.*, 2016] consider user input in the form of clicks. A simple and intuitive form of user interaction, clicks are not always the best option for object selection. For instance, it is difficult to click precisely on a tiny object on the small smartphone screen. Lately, sliding with a finger tends to replace tapping: Word Flow keyboard featuring shape writing was officially certified as the fastest smartphone keyboard, and many users prefer drawing a pattern to unlock the screen rather than entering a PIN-code with several clicks. Hence, we assume that switching from discrete to continuous input speeds up the interaction, and for smartphone users, it would be more convenient to draw a contour rather than to click for several times.

Accordingly, we focus on the **contour-based IS**. We address this task with a novel trainable method that segments

Figure 1: An example of an image where selecting objects with contours is much more efficient than with clicks. It takes about 5 seconds to select a flock of birds with a single contour, whereas it takes about 40 seconds with clicks (clicking on *each* bird in the flock!).

an object of interest given a single contour. Our method does not require manually annotated contours for training but makes use of conventional segmentation masks, so it can be trained on the standard segmentation datasets such as LVIS [Gupta *et al.*, 2019], COCO [Lin *et al.*, 2014], OpenImages [Kuznetsova *et al.*, 2020], and SBD [Hariharan *et al.*, 2011]. Our experiments show that a single contour allows achieving the same accuracy as 3-5 clicks on the standard benchmarks: GrabCut [Rother *et al.*, 2004], Berkeley [Martin *et al.*, 2001; McGuinness and O'connor, 2010], and DAVIS [Li *et al.*, 2018; Perazzi *et al.*, 2016]. We also present **UserContours**, a collection of 2000 images of common objects in their usual environment, annotated with real-user contours. Besides, we create the **UserContours-G** dataset by selecting 50 images especially difficult to segment: those depicting small objects, overlapped objects, and groups of objects. We empirically prove that our contour-based approach has an even greater advantage on such challenging human-annotated data compared to the click-based approach.

Overall, our contribution can be summarized as follows:

- To the best of our knowledge, we are the first to formulate the task of IS given a single contour;

- We adapt a state-of-the-art click-based model for contours, while not sacrificing its inference speed;

- We introduce a UserContours dataset and a challenging UserContours-G, manually labeled with contours;

- We develop an evaluation protocol that allows comparing the contour-based and click-based methods, and show that a single contour is equivalent to multiple clicks (up to 20!) in terms of segmentation accuracy.

2 Related Work

IS aims at obtaining a mask of an object given an image and an additional user input. Early methods [Boykov and Jolly, 2001; Grady, 2006; Gulshan *et al.*, 2010; Rother *et al.*, 2004] tackle the task via minimizing a cost function defined on a graph over image pixels.

Click-based Methods

Xu et al. [Xu *et al.*, 2016] introduced a CNN-based method and a click simulation strategy for training click-based IS methods on the standard segmentation datasets without additional annotation. In [Li *et al.*, 2018; Liew *et al.*, 2017; Liew *et al.*, 2019; Lin *et al.*, 2020], network predictions are refined through attention. BRS [Jang and Kim, 2019] minimized a discrepancy between the predicted mask and the map of clicks after each click, while in [Sofiiuk *et al.*, 2020; Kontogianni *et al.*, 2020], inference-time optimization was applied to higher network levels. Recent click-based methods [Hao *et al.*, 2021; Jang and Kim, 2019; Sofiiuk *et al.*, 2020; Sofiiuk *et al.*, 2022] show impressive accuracy, but still may require a lot of interactions. Among them, the best results are obtained via iterative click-based approaches [Jang and Kim, 2019; Sofiiuk *et al.*, 2020; Sofiiuk *et al.*, 2022] that leverage information about previous clicks. In such methods, model weights are updated after each user input, which increases the computational cost per click.

Alternative User Inputs

Alongside numerous click-based methods, other types of user input have been investigated. Strokes were widely employed as a guidance [Andriluka *et al.*, 2020; Bai and Wu, 2014; Batra *et al.*, 2010; Boykov and Jolly, 2001; Freedman and Zhang, 2005; Grady, 2006; Gueziri *et al.*, 2017; Gulshan *et al.*, 2010; Kim *et al.*, 2008; Lin *et al.*, 2016]; however, no comparison with click-based approaches was provided. Putting a stroke requires a lot of effort, and most stroke-based methods employed training-free techniques to imitate user inputs. DEXTR [Maninis *et al.*, 2018a] used extreme points: left, right, top, and bottom pixels of an object. In a recent work [Agustsson *et al.*, 2019b], strokes were combined with extreme points. However, placing extreme points in the right locations is non-trivial and definitely harder than clicking on an object, and the predictions cannot be corrected as well. Bounding boxes were used either for selecting large image areas [Cheng *et al.*, 2015; Rother *et al.*, 2004; Wu *et al.*, 2014; Xu *et al.*, 2017] or segmenting thin objects [Liew *et al.*, 2021]. The main drawbacks of bounding boxes are lack of specific object reference inside the selected area and no support for correcting predicted mask. However, it was shown that a model trained on bounding boxes could generalize to arbitrary closed curves [Xu *et al.*, 2017]. In [Zhang *et al.*, 2020], bounding boxes are combined with

clicks giving more specific object guidance and facilitating corrections. PhraseClick [Ding *et al.*, 2020] combined clicks with text input to specify object attributes and reduce the number of clicks.

We consider user input in the form of contours, and build a network capable of processing contours by slightly modifying a click-based IS network. This approach is proved to outperform click-based methods: particularly, we show that a single contour provides better results than several clicks.

3 Contour-based IS Method

Our method is inherited from the state-of-the-art click-based RITM [Sofiiuk *et al.*, 2022], having an interaction generation module, a backbone, and an interactive branch (Fig. 2).

3.1 Contour Generation Module

The contour generation module is designed to emulate real user behavior. As we are unaware of a user study of contouring patterns, we develop a generation procedure based on general assumptions. Our goal is that a network trained on generated samples performs well on real user inputs.

Contour Generation

A user is not expected to outline an object as accurate as possible: a real contour might cross object boundaries, do not cover some extruding object parts, or even lie within the object area instead of enclosing the object. Accordingly, to generate a contour from a ground truth instance mask, we need to distort it rather heavily. Overall, we generate contours via the following algorithm:

1. First, we fill all holes in the mask.

2. After that, we randomly select either dilation or erosion. For a chosen morphological transform, we randomly sample a kernel size depending on the image size. Hence, the transformed mask does not stretch or shrink too much, yet close-by objects might merge, so that the contour encloses a group of objects.
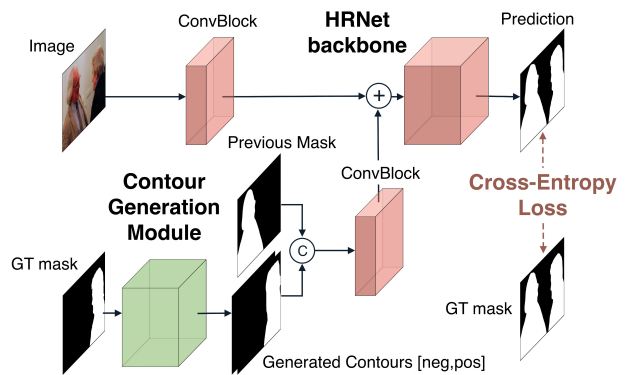

Figure 2: The architecture of the proposed method. The contour generation module simulates user contours. The generated contours are encoded as binary masks, stacked with a mask from a previous interaction, and fed into the network via a novel interactive branch. The network is trained to minimize binary cross-entropy between a predicted and a ground truth mask.
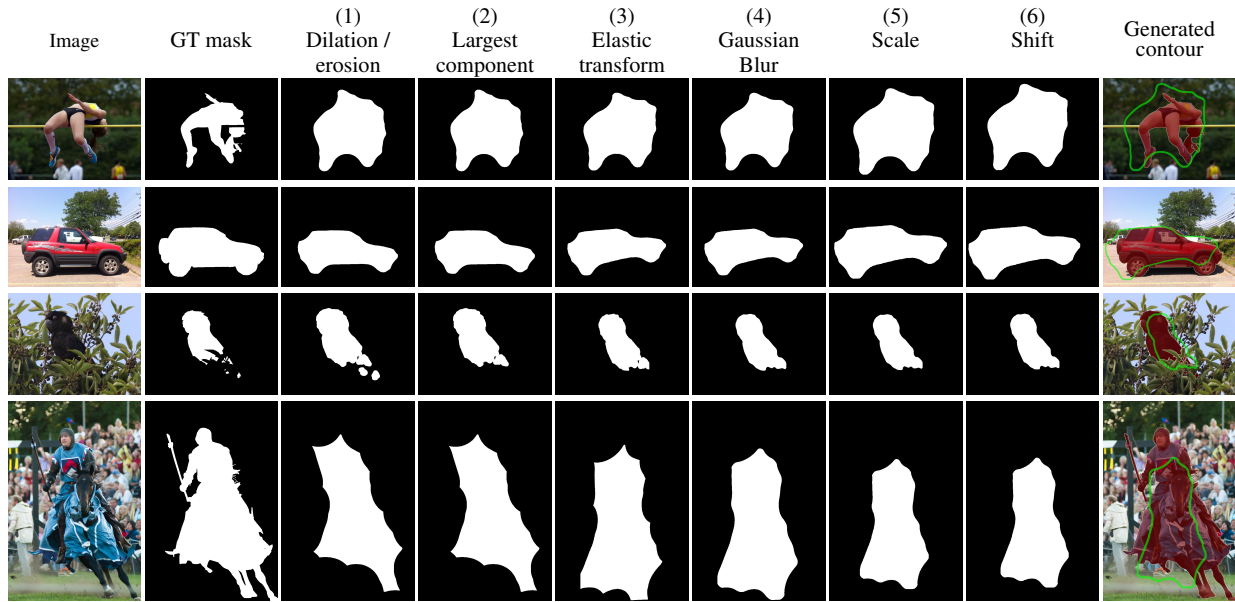
Figure 3: Step-by-step contour generation given a ground truth segmentation mask.

3. A contour should not outline distant parts of an object or even different objects, so we do not consider disconnected areas of the transformed mask. So, we search for connected components and select the largest one.

4. Then, we distort the mask via an elastic transform with random parameters depending on the object size. This might divide the mask into several disconnected parts, so we select the largest connected component yet again.

5. We smooth the mask via GaussianBlur with a random kernel size chosen according to the object size.

6. Next, we apply random scaling. We assume objects of a simple shape might be outlined rather coarsely, while complex shapes require a more thoughtful approach. Accordingly, we define a ratio r reflecting the "complexity" of the object shape: it is calculated as the area of the current mask divided by the area of its convex hull. If $r < 0.6$, we assume an object has a complex, non-convex shape. In this case, we cannot apply severe augmentations to the mask, since the distorted mask would match the object badly. If $r \geq 0.6$, an object seems to be "almost convex", so intense augmentations would not affect its shape so dramatically. Accordingly, we randomly sample a scaling factor within a narrow range for complex, "non-convex" objects, and from a wider range for less complex, "almost convex" objects.

7. Finally, we select a shift based on the object size. Particularly, we consider bounding boxes enclosing the transformed and the ground truth masks, and compute d_x as a minimum distance along x-axis between vertical sides of the ground truth and transformed boxes. An integer shift along the x-axis is sampled from $[-2d_x, 2d_x]$ for "almost convex" or $[-d_x, d_x]$ for "non-convex" objects, respectively. A shift along the y-axis is selected simi-

larly. The resulting mask defines a filled contour.

To clarify our generation procedure, we visualize the intermediate results in Fig. 3.

Fig. 4 depicts multiple contours generated for a single mask. Generated contours may vary in size and shape significantly due to the randomized generation procedure.

Contours Encoding

We formulate contours encoding guided by clicks encoding in click-based approaches. According to the study on clicks encoding [Benenson *et al.*, 2019], the best way is to encode clicks as binary masks with click positions marked with disks of a fixed radius (as in RITM [Sofiiuk *et al.*, 2022]). So we also represent contours as binary masks, considering two ways of encoding contours: "filled" contour masks with ones inside and zeros outside the contour, and contours drawn as lines. Our ablation study of contours encoding (Tab. 5) shows that "filled" contours provide higher segmentation quality than lines. We attribute this to filled masks providing explicit
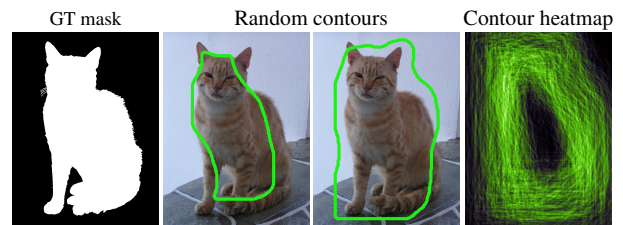

Figure 4: To emulate real user input, we aim to generate as diverse contours as possible. The only requirement is that they should adequately represent the desired object and allow unambiguously identifying it. A heatmap with 1000 random contours illustrates the range of variation. Here, we visualize contours as lines for clarity.

information about whether each pixel lies within a contour. Due to a limited receptive field, convolutional neural networks might not derive this information as effectively from contours drawn as lines.

Each user input is encoded as two binary masks: one for a positive contour and another for a negative one (one empty and one filled, depending on whether a positive or negative contour is drawn). We also leverage the history of user inputs contained in the mask predicted at the previous interaction. Prior to the first input, we pass an all-zeros mask. Overall, the network input is two binary maps stacked with the previous mask channel-wise, as in RITM [Sofiiuk *et al.*, 2022].

3.2 Backbone

Following RITM [Sofiiuk *et al.*, 2022], we use HRNet18 with an OCR module [Wang *et al.*, 2019; Yuan *et al.*, 2020] as a backbone. We also examine other HRNet backbones: a lightweight HRNet18s and more powerful HRNet32 and HR-Net48 models. The results evident that the network complexity has a minor effect on the segmentation quality (Tab. 4).

3.3 Interactive Branch

We modify a segmentation network by adding an interactive branch that processes an additional user input. This is implemented via Conv1S, a network modification proposed in RITM [Sofiiuk *et al.*, 2022]. Specifically, we pass the interaction through the interactive branch made up of Conv+LeakyReLU+Conv layers. Then, we sum up the result with the output of the first convolutional backbone layer. Yet, we observe that the interactive branch output might confuse the network at the beginning of the training. To avoid this, we extend the interactive branch with a scaling layer, that multiplies the output by a learnable coefficient just before summation. Through scaling, we can balance the relative importance of image features and user inputs in a fully data-driven way.

4 Experiments

4.1 Standard Benchmarks

Following RITM [Sofiiuk *et al.*, 2022], we train our models on the standard segmentation datasets. Specifically, we use Semantic Boundaries Dataset, or SBD [Hariharan *et al.*, 2011], and the combination of LVIS [Gupta *et al.*, 2019] and COCO [Lin *et al.*, 2014] for training. SBD [Hariharan *et al.*, 2011] consists of 8498 training samples. COCO and LVIS share the same set of 118k training images; COCO contains a total of 1.2M instance masks of common objects, while LVIS [Gupta *et al.*, 2019] is annotated with 1.3M instance masks with long-tail object class distribution. Respectively, the combination of COCO and LVIS contains small yet diverse set of classes from LVIS and general and large set of classes from COCO. In an ablation study of training data, we use the test+validation split of OpenImages [Kuznetsova *et al.*, 2020] (about 100k samples); we do not consider the train split since it is annotated quite inaccurately.

We evaluate our method on standard IS benchmarks: Grab-Cut [Rother *et al.*, 2004] (50 samples), the test subset of Berkeley [Martin *et al.*, 2001; McGuinness and O'connor, 2010] (100 samples), a set of 345 randomly sampled frames

from DAVIS [Perazzi *et al.*, 2016; Jang and Kim, 2019], and the test subset of SBD (539 samples). Originally, these benchmarks do not contain contour annotations, so we manually label them with contours by ourselves.

4.2 Proposed Datasets

We present the **UserContours** dataset with 2000 images depicting common objects in their natural context. Besides, we manually select 50 samples containing objects groups to create an especially challenging **UserContours-G**. The examples of images along with instance segmentation masks and user-defined contours are shown in Fig. 5 and Fig. 8.

Source of Data

The images of UserContours are taken from the train subset of OpenImages V6. We selected 2000 diverse images depicting common objects in various scenarios. The only restriction is imposed on image resolution: we consider only images with a shorter side of between 400px and 1600px, and of an aspect ratio between 1:5 and 5:1.

Instances Labeling

We decomposed the labeling task into two subtasks. The first subtask implies creating instance segmentation masks for the given images. For the test subsets of the standard benchmarks, we use pre-defined instance segmentation masks already present in these datasets. The second subtask is to outline instances with contours.

First, we label images with instance masks. Since either a single object or a group of close-by objects can be a subject of interest, we request our annotators to label 50% of instance masks as groups and 50% as individual objects. At least one

Figure 5: Examples from UserContours. User-defined contours are green, instance segmentation masks are red. Contours might be loose and non-closed.

Figure 6: The instances (marked with red) segmented by different users in the same image might also be different.

Figure 7: If an overlapped object cannot be selected with a single positive contour (green), negative contours (red) are allowed.
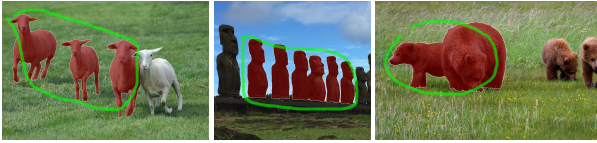

Figure 8: Images containing object groups from UserContours-G.

object per image should be annotated. To make the annotation process more alike with real image editing scenarios, we do not explicitly formulate what is a desired object. Instead, we ask annotators to label any objects that stand out (Fig. 6). We also do not restrict the object size or the location in an image.

Contours Labeling

We asked our annotators to outline each segmented instance with a *contour*: a line loosely following object boundaries. There should be no intermediate breaks in a contour; however, its start may not coincide with its end: it this case, we close the contour by connecting the first and the last points with a line. We aim to emulate real user interactions, so we requested for the contours that are not as precise as possible, but drawn in a natural relaxed manner. Nevertheless, the correspondence between instances and contours should be clear and unambiguous. Negative contours might be used only when necessary (Fig. 7).

UserContours-G

We hand-picked 50 images depicting groups of objects from UserContours to create a small yet extremely complex UserContours-G (Fig. 8).

4.3 Evaluation

Click-based Evaluation

Click-based IS methods are typically evaluated with NoC@k – the number of clicks made to achieve a predefined IoU=k [Hao *et al.*, 2021; Jang and Kim, 2019; Maninis *et al.*, 2018b; Sofiiuk *et al.*, 2020; Sofiiuk *et al.*, 2022]. For contours, the equivalent number of contours can be reported. However, this seems controversial as there exists a crucial difference between clicks and contours. A click is a pair of coordinates with a fixed and limited complexity, while a contour might be an arbitrarily long curve of an extremely complex shape. Since there is no conventional approach to measuring the curve complexity, we cannot formulate the relative complexity of clicks and contours and explicitly incorporate

it into the evaluation metrics (e.g., in the form of scaling coefficients). Neither, we cannot treat clicks and contours equally.

Contour-based Evaluation

Since the proposed method is the first contour-based approach, there are no established evaluation protocols. Accordingly, we formulate an evaluation metric based on observations of human behavior. A user is not assumed to spend much time on image editing in mobile applications, so speed is a crucial factor of usability. Since the speed depends on the number of interactions, the fewer interactions, the better. Accordingly, it seems important to provide decent predictions even after the first contour. Since we manually annotate test datasets with contours, we can calculate IoU achieved with a single contour and use it as the main metric for assessing the contour-based IS. Or, we can apply click-based models and find the number of clicks required to achieve the same accuracy as with a single contour. This way, contour-based methods can be compared with click-based methods non-directly.

4.4 Implementation Details

Training

We train a binary segmentation model using a BCE loss. Input images are resized to 320px $\times$ 480px. During training, we randomly crop and rescale images, use horizontal flip, and apply random jittering of brightness, contrast, and RGB values. With an equal probability, we choose an object of interest with a positive contour or erase the unwanted object with a negative contour (passing the ground truth object mask as the previous mask, and treating the generated contour as a negative one).

The models are trained for 140 epochs using Adam [Kingma and Ba, 2014] with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$. The learning rate is initialized with $5 \cdot 10^{-4}$ and reduced by a factor of 10 at epochs 119 and 133.

In a study of training data, we fine-tune our models for 10 epochs. We use stochastic weight averaging [Izmailov *et al.*, 2018], aggregating the weights at every second epoch starting from the fourth epoch. During fine-tuning, we set the learning rate to $1 \cdot 10^{-5}$ for the backbone and $1 \cdot 10^{-4}$ for the rest of the network, and reduce it by a factor of 10 at epochs 8 and 9.

Evaluation

We follow RITM [Sofiiuk *et al.*, 2020] for the evaluation, using Zoom-In and averaging predictions from the original and the horizontally flipped images. Unlike a single click, a single contour allows hypothesizing about the object size, so we can apply Zoom-In at the first interaction; this minor change tends to improve the results significantly.

4.5 Comparison with Previous Works

We present quantitative results for GrabCut (Tab. 1), Berkeley (Tab. 2), and DAVIS (Tab. 3). We report a mean IoU for from 1 to 5 clicks for the click-based models, and an IoU after the first interaction for our contour-based models. Apparently, a single contour provides the same accuracy as 5 clicks on GrabCut and 3 clicks on Berkeley and DAVIS, compared with the state-of-the-art RITM.

Note, that our method has the same backbone and interactive branch as RITM, so their computational efficiency and

Method	Training data	IoU				
		@1	@2	@3	@4	@5
BRS [Jang and Kim, 2019]	SBD	80.0*	87.0*	89.0*	90.0*	90.0*
f-BRS [Sofiiuk *et al.*, 2020]	SBD	80.0*	85.0*	87.0*	91.0*	92.0*
EgdeFlow [Hao *et al.*, 2021]	LC	85.0*	92.0*	94.0*	95.5*	96.5*
RITM [Sofiiuk *et al.*, 2022]	LC	87.46	91.76	95.39	96.28	97.20
Ours	SBD	96.42				
Ours	LC	96.32				

Table 1: A quantitative comparison of IS methods on GrabCut. LC denotes LVIS+COCO. "*" means an approximate metric value determined from the IoU plots from the original papers. The results better than ours are underlined.

Method	Training data	IoU				
		@1	@2	@3	@4	@5
BRS [Jang and Kim, 2019]	SBD	80.0*	85.0*	87.0*	89.0*	91.0*
f-BRS [Sofiiuk *et al.*, 2020]	SBD	77.0*	83.0*	85.0*	88.0*	90.0*
EgdeFlow [Hao *et al.*, 2021]	LC	80.0*	90.0*	93.5*	94.5*	95.0*
RITM [Sofiiuk *et al.*, 2022]	LC	82.88	91.51	94.46	95.57	95.83
Ours	SBD	93.35				
Ours	LC	93.08				

Table 2: A quantitative comparison of IS methods on Berkeley. LC denotes LVIS+COCO. "*" means an approximate metric value determined from the IoU plots from the original papers. The results better than ours are underlined.

Method	Training data	IoU				
		@1	@2	@3	@4	@5
BRS [Jang and Kim, 2019]	SBD	72.0*	80.0*	85.0*	86.0*	86.0*
f-BRS [Sofiiuk *et al.*, 2020]	SBD	71.0*	79.0*	79.0*	82.0*	83.0*
EgdeFlow [Hao *et al.*, 2021]	LC	74.0*	83.0*	86.0*	86.05	89.0*
RITM [Sofiiuk *et al.*, 2022]	LC	73.37	82.51	86.45	88.46	89.62
Ours	SBD	85.44				
Ours	LC	86.05				

Table 3: A quantitative comparison of IS methods on DAVIS. LC denotes LVIS+COCO. "*" means an approximate metric value determined from the IoU plots from the original papers. The results better than ours are underlined.

inference speed are on par. However, RITM needs more inference rounds to achieve the same segmentation quality.

We compare an IoU per click for RITM with an IoU of our method achieved with a single contour on UserContours (Fig. 9) and UserContours-G (Fig. 10). For UserContours, a single contour is as effective as 5 clicks on average. Apparently, for hard segmentation cases present in UserContours-G, using contours is even more beneficial than using clicks: one contour is equivalent to 20(!) clicks in terms of IoU. A qualitative comparison is presented in Fig. 11.

4.6 Ablation Study

Backbones
We use the same backbone as RITM [Sofiiuk *et al.*, 2022] to guarantee our method can be fairly compared with the pre-
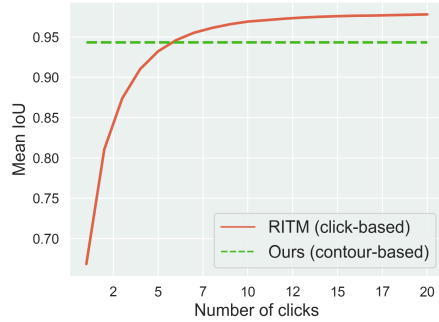

Figure 9: IoU of RITM (per click) and IoU of our method (a single contour) on UserContours.
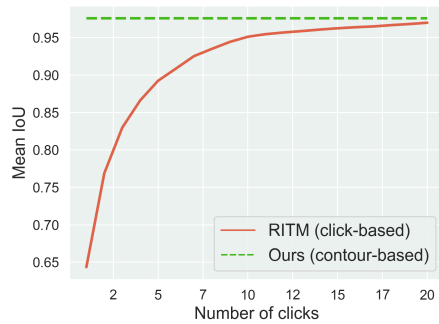

Figure 10: IoU of RITM (per click) and IoU of our method (a single contour) on UserContours-G.

vious state-of-the-art. Our experiments with HRNet18s, HRNet32, and HRNet48 reveal that a model complexity does not affect accuracy much (Tab. 4), so we opt for the efficient yet accurate HRNet18.

Backbone	GrabCut	Berkeley	DAVIS	SBD
HRNet18	95.14	91.16	83.66	**87.52**
HRNet18s	94.28	**91.48**	**83.78**	86.85
HRNet32	**95.22**	90.94	83.17	86.94
HRNet48	94.84	90.87	82.70	87.39

Table 4: Ablation study of backbones. We train our models on LVIS+COCO and represent contours as filled masks. The best results are **bold**.

Contours Encoding
We compare filled contour masks with contours as lines of varying width. According to the Tab. 5, a width of 2% of the length of the shorter side of an image provides the best results among all contour representations in the form of lines. However, they are still inferior to the results obtained with filled contour masks. Respectively, we use "filled" contours in all other experiments, as they facilitate more accurate predictions on all benchmarks.
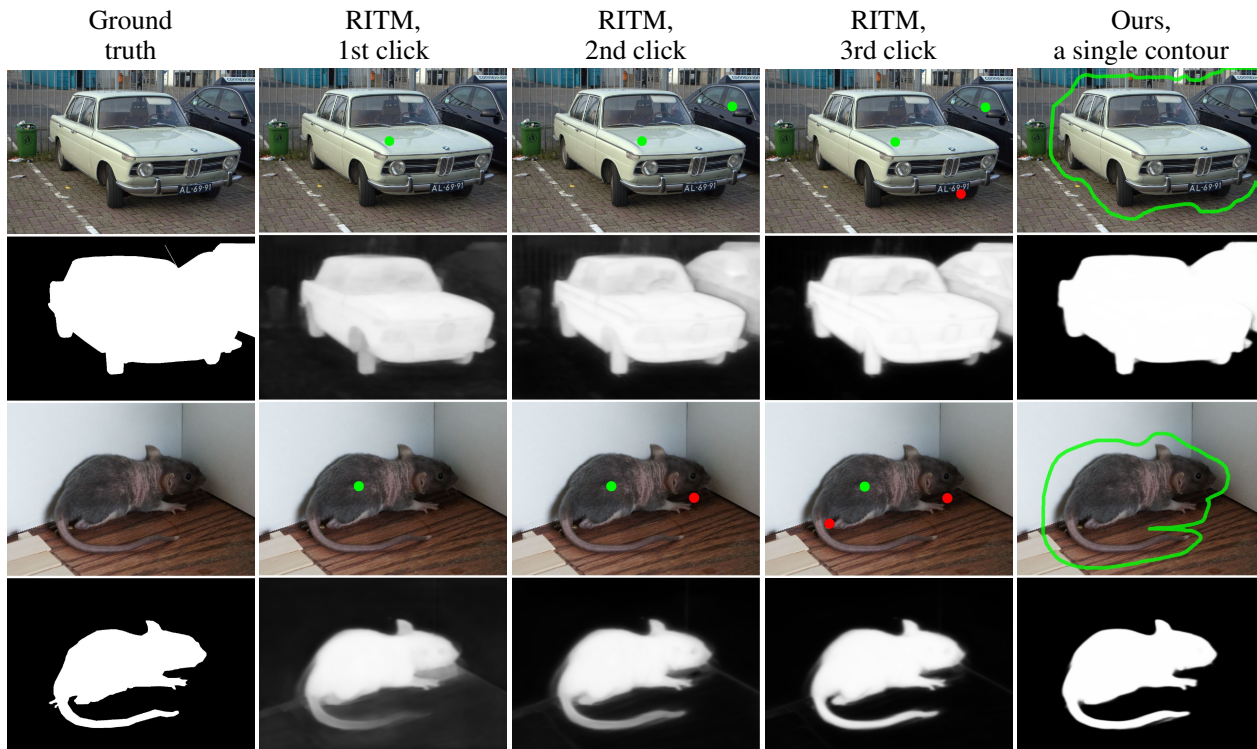
Ground truth	RITM, 1st click	RITM, 2nd click	RITM, 3rd click	Ours, a single contour

Figure 11: Randomly generated user interactions and corresponding predictions obtained with RITM and our method.

Contours encoding	GrabCut	Berkeley	DAVIS	SBD
Filled	**95.14**	**91.16**	**83.66**	**87.52**
Line, 0.005	25.48	16.44	25.58	21.06
Line, 0.01	94.55	90.99	81.98	87.02
Line, 0.02	94.68	91.13	82.44	87.07
Line, 0.05	94.08	89.94	81.71	86.82
Line, 0.1	94.92	89.63	80.18	86.65

Table 5: Ablation study of contours encoding. We train our models on LVIS+COCO and employ HRNet18 as a backbone. "Line, w" means that the contour is represented as a line of a width $= (w \times$ the length of a shorter image side). The best results are **bold**.

Training Datasets

We measure the performance gain from using additional data sources. We leverage the OpenImages data in two different ways. First, we simply combine it with LVIS+COCO for training, following the same training procedure as for LVIS+COCO only. The results are comparable to those obtained with training only on LVIS+COCO (Tab. 6). Alternatively, we fine-tune on a part of the OpenImages data. To compose a fine-tuning set, we utilize our best model (HRNet18-based, trained on LVIS+COCO, "filled" contours). For each image from OpenImages test split, we generate a random contour and pass it through the model. If a predicted mask has an IoU>97 with a ground truth mask, we save a sample consisting of an image, a ground truth mask, and the generated contour. This way, we obtain a set of 2533

contours for 2253 images. It appears to be more profitable to fine-tune on a carefully selected minor subset of OpenImages, than to use the entire test+validation split for training.

Training data	Fine-tuning	GrabCut	Berkeley	DAVIS	SBD
LC	-	95.14	91.16	83.66	87.52
LC+OI	-	95.14	92.31	83.61	87.16
LC	+	**96.32**	**93.08**	**86.05**	**87.84**

Table 6: Ablation study of training data. We employ HRNet18 as a backbone, and represent contours as filled masks. LC stands for LVIS+COCO, OI means OpenImages. The best results are **bold**.

5 Conclusion

We presented a novel contour-based interactive segmentation method. We tested our approach on the standard benchmarks against click-based methods and showed that a single contour provides the same accuracy as several clicks. Moreover, we introduced a novel UserContours containing human-annotated contours for common objects in the wild, and UserContours-G, featuring difficult segmentation cases. We empirically proved that our contour-based approach has an even greater advantage over click-based methods on challenging data. Overall, we demonstrated that contours could reduce the required number of interactions and significantly simplify image editing and labeling.

References

[Acuna *et al.*, 2018] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-RNN++. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 859–868. IEEE, jun 2018.

[Agustsson *et al.*, 2019a] Eirikur Agustsson, Jasper R. Uijlings, and Vittorio Ferrari. Interactive full image segmentation by considering all regions jointly. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11622–11631. IEEE, jun 2019.

[Agustsson *et al.*, 2019b] Eirikur Agustsson, Jasper R. R. Uijlings, and Vittorio Ferrari. Interactive full image segmentation by considering all regions jointly. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[Andriluka *et al.*, 2020] Mykhaylo Andriluka, Stefano Pellegrini, Stefan Popov, and Vittorio Ferrari. Efficient full image interactive segmentation by leveraging within-image appearance similarity. *arXiv preprint arXiv:2007.08173*, 2020.

[Bai and Wu, 2014] Junjie Bai and Xiaodong Wu. Error-tolerant scribbles based interactive image segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[Batra *et al.*, 2010] Dhruv Batra, Adarsh Kowdle, Devi Parikh, Jiebo Luo, and Tsuhan Chen. icoseg: Interactive co-segmentation with intelligent scribble guidance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[Benenson *et al.*, 2019] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. Large-scale interactive object segmentation with human annotators. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11700–11709, 2019.

[Boykov and Jolly, 2001] Y.Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, volume 1, pages 105–112. IEEE Comput. Soc, 2001.

[Cheng *et al.*, 2015] M. M. Cheng, V. A. Prisacariu, S. Zheng, P. H. S. Torr, and C. Rother. DenseCut: Densely connected CRFs for realtime GrabCut. *Computer Graphics Forum*, 34(7):193–201, oct 2015.

[Ding *et al.*, 2020] Henghui Ding, Scott Cohen, Brian Price, and Xudong Jiang. PhraseClick: Toward achieving flexible interactive segmentation by phrase and click. In *IEEE/CVF European Conference on Computer Vision (ECCV)*, pages 417–435. Springer International Publishing, 2020.

[Freedman and Zhang, 2005] D. Freedman and Tao Zhang. Interactive graph cut based segmentation with shape priors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 755–762. IEEE, 2005.

[Grady, 2006] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, nov 2006.

[Gueziri *et al.*, 2017] Houssem-Eddine Gueziri, Michael McGuffin, and Catherine Laporte. Latency management in scribble-based interactive segmentation of medical images. *IEEE Transactions on Biomedical Engineering*, 2017.

[Gulshan *et al.*, 2010] Varun Gulshan, Carsten Rother, Antonio Criminisi, Andrew Blake, and Andrew Zisserman. Geodesic star convexity for interactive image segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3129–3136. IEEE, jun 2010.

[Gupta *et al.*, 2019] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5356–5364, 2019.

[Hao *et al.*, 2021] Yuying Hao, Yi Liu, Zewu Wu, Lin Han, Yizhou Chen, Guowei Chen, Lutao Chu, Shiyu Tang, Zhiliang Yu, Zeyu Chen, et al. Edgeflow: Achieving practical interactive segmentation with edge-guided flow. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1551–1560, 2021.

[Hariharan *et al.*, 2011] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 991–998. IEEE, 2011.

[Izmailov *et al.*, 2018] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *Conference on Uncertainty in Artificial Intelligence*, 2018.

[Jang and Kim, 2019] Won-Dong Jang and Chang-Su Kim. Interactive image segmentation via backpropagating refinement scheme. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5297–5306, 2019.

[Kim *et al.*, 2008] Tae Hoon Kim, Kyoung Mu Lee, and Sang Uk Lee. Generative image segmentation using random walks with restart. In *IEEE/CVF European Conference on Computer Vision (ECCV)*, pages 264–275. Springer Berlin Heidelberg, 2008.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Kontogianni *et al.*, 2020] Theodora Kontogianni, Michael Gygli, Jasper Uijlings, and Vittorio Ferrari. Continuous adaptation for interactive object segmentation by learning from corrections. In *IEEE/CVF European Conference on Computer Vision (ECCV)*, pages 579–596. Springer International Publishing, 2020.

[Kuznetsova *et al.*, 2020] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4. *International Journal of Computer Vision (IJCV)*, 128(7):1956–1981, 2020.

[Li *et al.*, 2018] Zhuwen Li, Qifeng Chen, and Vladlen Koltun. Interactive image segmentation with latent diversity. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Liew *et al.*, 2017] JunHao Liew, Yunchao Wei, Wei Xiong, Sim-Heng Ong, and Jiashi Feng. Regional interactive image segmentation networks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2746–2754. IEEE, oct 2017.

[Liew *et al.*, 2019] Jun Hao Liew, Scott Cohen, Brian Price, Long Mai, Sim-Heng Ong, and Jiashi Feng. MultiSeg: Semantically meaningful, scale-diverse segmentations from minimal user input. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 662–670. IEEE, oct 2019.

[Liew *et al.*, 2021] Jun Hao Liew, Scott Cohen, Brian Price, Long Mai, and Jiashi Feng. Deep interactive thin object selection. In *EEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[Lin *et al.*, 2014] 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 *IEEE/CVF European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014.

[Lin *et al.*, 2016] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. ScribbleSup: Scribble-supervised convolutional networks for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3159–3167. IEEE, jun 2016.

[Lin *et al.*, 2020] Zheng Lin, Zhao Zhang, Lin-Zhuo Chen, Ming-Ming Cheng, and Shao-Ping Lu. Interactive image segmentation with first click attention. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13339–13348. IEEE, jun 2020.

[Maninis *et al.*, 2018a] K.-K. Maninis, S. Caelles, J. Pont-Tuset, and L. Van Gool. Deep extreme cut: From extreme points to object segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 616–625. IEEE, jun 2018.

[Maninis *et al.*, 2018b] K.-K. Maninis, S. Caelles, J. Pont-Tuset, and L. Van Gool. Deep extreme cut: From extreme points to object segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Martin *et al.*, 2001] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, volume 2, pages 416–423. IEEE, 2001.

[McGuinness and O'connor, 2010] Kevin McGuinness and Noel E O'connor. A comparative evaluation of interactive segmentation algorithms. *Pattern Recognition*, 43(2):434–444, 2010.

[Perazzi *et al.*, 2016] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 724–732, 2016.

[Rother *et al.*, 2004] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "grabcut" interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314, 2004.

[Sofiiuk *et al.*, 2020] Konstantin Sofiiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. f-brs: Rethinking back-propagating refinement for interactive segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8623–8632, 2020.

[Sofiiuk *et al.*, 2022] Konstantin Sofiiuk, Ilia A Petrov, and Anton Konushin. Reviving iterative training with mask guidance for interactive segmentation. In *IEEE International Conference on Image Processing (ICIP)*, 2022.

[Wang *et al.*, 2019] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2019.

[Wu *et al.*, 2014] Jiajun Wu, Yibiao Zhao, Jun-Yan Zhu, Siwei Luo, and Zhuowen Tu. MILCut: A sweeping line multiple instance learning paradigm for interactive image segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 256–263. IEEE, jun 2014.

[Xu *et al.*, 2016] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 373–381, 2016.

[Xu *et al.*, 2017] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas Huang. Deep grabcut for object selection. *ArXiv*, abs/1707.00243, 2017.

[Yuan *et al.*, 2020] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *IEEE/CVF European Conference on Computer Vision (ECCV)*, 2020.

[Zhang *et al.*, 2020] Shiyin Zhang, Jun Hao Liew, Yunchao Wei, Shikui Wei, and Yao Zhao. Interactive object segmentation with inside-outside guidance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12234–12244. IEEE, jun 2020.