Adapting the Segment Anything Model During Usage in Novel Situations

Robin Schön Julian Lorenz Katja Ludwig
Rainer Lienhart
Universität Augsburg
86159 Augsburg, Universitätsstraße 6a
{robin.schoen, julian.lorenz, katja.ludwig, rainer.lienhart}@uni-a.de

Abstract

The interactive segmentation task consists in the creation of object segmentation masks based on user interactions. The most common way to guide a model towards producing a correct segmentation consists in clicks on the object and background. The recently published Segment Anything Model (SAM) supports a generalized version of the interactive segmentation problem and has been trained on an object segmentation dataset which contains 1.1B masks. Though being trained extensively and with the explicit purpose of serving as a foundation model, we show significant limitations of SAM when being applied for interactive segmentation on novel domains or object types. On the used datasets, SAM displays a failure rate $FR_{30}@90$ *of up to* 72.6%*. Since we still want such foundation models to be immediately applicable, we present a framework that can adapt SAM during immediate usage. For this we will leverage the user interactions and masks, which are constructed during the interactive segmentation process. We use this information to generate pseudo-labels, which we use to compute a loss function and optimize a part of the SAM model. The presented method causes a relative reduction of up to* 48.1% *in the* $FR_{20}@85$ *and* 46.6% *in the* $FR_{30}@90$ *metrics.*

1. Introduction

Many computer vision systems need object segmentation masks for single images as training material. The development of such systems has especially been aided by the existence of large datasets for regular consumer images, such as COCO [27] and ADE20k [59]. Some segmentation tasks, however, need much more specific data. Example domains for such cases are sports [33, 34], agriculture [40], medical image segmentation [19], and robotic vision [60].

The annotation of instance segmentation datasets usually incurs a high effort. Not only is there a large cost associated for human annotators, but in some difficult cases the creation of a high-quality mask is a non-negligible problem. An example for this would be the annotation of mask polygons when the object edges are finely jagged. In consequence, this led to the development of interactive segmentation systems. Such systems receive a simple, low-effort user interaction to create masks. This usually happens in an iteratively interactive context: The human refines computed masks by repeatedly interacting with the system, adding progressively more guiding interactions while inspecting the mask. This process goes on until the user is satisfied with the quality of the mask. In most cases, such interactions take the form of clicks, but scribbles, bounding boxes and coarse masks constitute usable forms of user guidance as well.

The class agnostic nature of this task renders it viable for any kind of prompt. This property has been exploited to create a large foundation model which is capable of performing interactive segmentation, the Segment Anything model or SAM [23]. While SAM is trained on the large SA-1B dataset, which has been published in conjunction with the model, a lot of practical scenarios require the creation of datasets for very specific tasks. This is for example the case in smaller companies that seek to create datasets for the usage of in-house applications of computer vision, such as the automatization of processes. Here, only a small set of objects might be interesting to annotate. The go-to solution in such cases is fine-tuning the pretrained foundation model. Such a fine-tuning training, however, necessitates two factors: 1) Availability of a preexisting dataset in the target domain that can be used as training data during fine-tuning. 2) The necessary computational resources to fine-tune the interactive segmentation model. This practically entails an entirely new additional training stage. On low-performance devices, such as hardware without GPU support or mobile phones, this requirement constitutes a considerable obstacle.

Especially the latter problem occurs in situations where the annotation of data should be distributed amongst many annotators. Most of them will not have a high performance machine at their disposal. The goal is therefore to not only find a strategy for adapting an interactive segmentation model that does not require additional data, but one that is efficient in the sense that any computationally demanding fine-tuning process can be avoided completely.

In our paper we are going to present such an adaptation strategy for SAM, while viewing this problem in the light of the interactive segmentation task on scenarios which are considerably different from regular consumer images. This first and foremost means the usage of appropriate metrics: The first important metric is the Number of Clicks (NoC) we need to annotated an object mask, and the second one is the Failure Rate (FR) which tells us about the percentage of cases in which we fail to do so with a reasonable number of clicks. Out of these two metrics, we regard the failure rate as the more crucial metric since it informs us about the limits of the model's segmentation capabilities. Our adaptation strategy mostly relies on pseudo-labels which are generated during the interaction. We use the clicks created by the user as pseudo labels for single pixels. In addition to that we use the mask which results from the interaction after pruning it to avoid errouneous training signals.

We will only carry out a partial adaptation of the network. In case the user intends to annotate multiple classes, the fine-tuned part can thus be copied for every particular class. For the purpose of validating the techniques we are going to adapt SAM to miscellaneous rare situations, as well as medical image segmentation tasks. Our contributions can be summarized as follows:

1. We explore the performance of SAM as an interactive segmentation model on a variety of datasets which differ from regular consumer images.
2. We test the limit of SAM's segmentation capabilities, and show that the model displays a considerable failure rate on domains which are different from general consumer images.
3. We show possible adaptation schemes which lower the failure rate without incurring considerable costs. The low memory overhead and fast adaptation render the usage of our method effectively for free.

2. Related Work

2.1. Interactive Segmentation

Interactive Segmentation uses various kinds of user guidance, with clicks being the most popular annotation mode [7, 28, 35, 36, 55]. The method in [37] uses four extreme points, which are assumed to be exactly on the borders of the object. Building on this work, Dupont et al. [13] try to segment the object with only two non axis aligned points. Jang and Kim [18] try to improve their prediction by optimizing their interaction maps via backpropagation. The work of Sofiiuk et al. [45] extends this by introducing auxiliary variables, which are optimized instead of the interaction maps. Zhang et al. [57] combines bounding boxes with clicks on the object surface as user input. While recent work mostly uses on convolutional architectures [8, 15, 46], the general training scheme is applied to networks with ViT-based backbones by Liu et al. [31]. The methods in [1, 3, 4, 9, 32] use scribbles as a form of guidance for interactive segmentation. [30, 41] look at the problem of 3D interactive segmentation. Recently, Kirillov et al. [23] have proposed the so called Segment Anything model (SAM) together with SA-1B, the largest interactive segmentation dataset to date containing over 1.1B segmentation masks. Due to the availability of the weights of the Segment Anything Model, there have been various papers which fine-tune its weights in order to adapt the model to a specific task. Cheng et al. [10] and Wu et al. [54] adapt SAM to various medical image segmentation tasks. Wang et al. [49] use a modified version of SAM for robotic surgery. In Chen et al. [6], adapter layers are introduced at intermediate places in the SAM-Encoder in order to fine-tune SAM to unusual image segmentation tasks. The method in Ding et al. [11] adapts FastSAM [58] for the task of change detection in remote sensing. The authors of [21] improve SAM by adding a small amount of parameters to the SAM head and fine tuning these new parameters on high-quality human annotated data. It should be noted that all aforementioned methods require some additional fine-tuning on an existing annotated dataset in the target domain before they can be used. In contrast to that, our method can be used directly and adapts the network on-the-fly.

2.2. Test-Time Adaptation

The field of test-time adaptation deals with techniques to improve the model while it is already in use. Most of the existing methods are employed in contexts where there is no access to high-quality pseudo-labels, as would be the case in interactive segmentation. The methods proposed by Song et al. [47] and Wang et al. [50] leverage entropy-minimization to adapt the model. Wang et al. [53] use a consistency loss and a exponential moving average, while stochastically restoring single weights to mitigate error accumulation. The methods most strongly related to this paper, are methods which focus on the adaptation of interactive segmentation models during usage. The most commonly exploited information in these methods are the user generated clicks. Albeit very sparse, they provide immediately available ground truth information. Kontogianni et al. [24], Shi et al. [42] and Lenczner et al. [26] all exploit the clicks which are available due to the user interaction. The authors of Wang et al. [51] fine-tune their model on the basis of scribbles. The works of [16] and [29] is most similar to our method, since the authors mention that they use intermediate masks or previously created masks, respectively. They do, however, not mention any method avoiding erroneous masks or regions. In contrast to our method, both publications also introduce additional modules to their model which would require an additional previous fine-tuning stage.

3. Method

3.1. Problem Statement

First, we will provide a precise description of the interactive segmentation problem. We follow the problem description discussed in [8, 31, 46]. Afterwards, we will briefly describe how we simulate the interaction in order to test such a system. Assume that we have an image $\boldsymbol{x} \in \mathbb{R}^{H \times W \times 3}$ and wish to create a segmentation map $\boldsymbol{m} \in \{0, 1\}^{H,W}$ which delimits a desired area in said image. That is, every pixel belonging to the area in $\boldsymbol{x}$ is set to 1 in $\boldsymbol{m}$, and every other pixel to 0. In order to create such an annotation, a user will repeatedly interact with a neural network f_{Seg} by providing it with clicks that indicate pixels reliably belonging to the foreground or background of the image. In each step t the user will be shown the current estimation of the mask $\boldsymbol{m}_{t-1}$, which only consists of background pixels in the beginning ($t = 0$). The user then chooses a falsely labeled region from the mask and places a click $\boldsymbol{p}_t$ on its surface. This $\boldsymbol{p}_t$ is a triple (i_t, j_t, l_t) which indicates a position $(i, j) \in \{1, ..., H\} \times \{1, ..., W\}$ and, depending on the choice of the user, a label $l \in \{+, -\}$ marking the position as foreground or background. The model f_{Seg} is then given $\boldsymbol{m}_{t-1}$, all previously clicked pixels $\boldsymbol{p}_{1:t} = \{\boldsymbol{p}_1, ..., \boldsymbol{p}_t\}$ and the image $\boldsymbol{x}$ in order to predict an improved mask $\boldsymbol{m}_t = f_{\text{Seg}}(\boldsymbol{x}, \boldsymbol{p}_{1:t}, \boldsymbol{m}_{t-1})$. Once the user regards the quality of the mask as satisfactory, the interaction stops by saving this mask as $\boldsymbol{m}^{\text{Res}}$, and the next image is annotated. It is to be noted that this result mask $\boldsymbol{m}^{\text{Res}}$ might still be partially erroneous if the user chooses to ignore falsely annotated parts.

When it comes to evaluating the quality of such systems, we do not usually have a user at our disposal. Instead, we follow Sofiiuk et al. [46] to simulate user interaction on images for which we already have ground truth segmentation masks $\boldsymbol{m}^{\text{GT}}$. At each iteration, we first compute the false positive area $\boldsymbol{m}_{\text{FP}}$ and the false negative area $\boldsymbol{m}_{\text{FN}}$. Then we compute the euclidean distance transforms $\mathcal{D}(\boldsymbol{m}_{\text{FP}})$ and $\mathcal{D}(\boldsymbol{m}_{\text{FN}})$ of the respective error masks, and select the pixel with the largest value on both distance transforms as a click. The label of the click depends on whether it has been placed on $\boldsymbol{m}_{\text{FP}}$ or $\boldsymbol{m}_{\text{FN}}$. We stop the interaction once the overlap of the proposed mask $\boldsymbol{m}_t$ with the ground truth mask $\boldsymbol{m}^{\text{GT}}$ exceeds a certain minimum IoU. This final mask will then be treated as the result mask $\boldsymbol{m}^{\text{Res}}$.

3.2. Foundation models for Interactive Segmentation

The Segment Anything Model (SAM) is a large foundation model for the general task of *promptable segmentation*, which has been published in Kirillov et al. [23] alongside the SA-1B dataset. Promptable segmentation denotes the task of segmenting arbitrary object instances as indicated by a user interaction, such as bounding boxes, text prompts or foreground/background clicks, as well as previously available low-quality masks. The ability to improve upon previous masks and being guided by foreground/background clicks renders every promptable segmentation model compatible with click-based interactive segmentation. In addition to that, SAM has been pretrained on the SA-1B dataset, which contains 1.1B class-agnostic segmentation masks for 11M images. This causes SAM to be an extraordinarily good model for segmentation of objects on consumer images. Despite this, there is still room for improvement when it comes to more specific image domains and more obscure types of objects, as our experiments indicate.

The architecture of SAM itself is divided into three parts: An *image encoder*, a *prompt encoder* and a *mask decoder*. The image encoder receives an image $\boldsymbol{x} \in \mathbb{R}^{H \times W \times 3}$ and encodes it into a feature map independently of any user interaction. The authors of SAM use a ViT backbone for this task. The prompt encoder receives the prompt in the form of clicks, bounding boxes, and masks, and encodes them into a form which is useful for the mask decoder. The mask decoder receives the image features and the encoded prompts, and uses both to predict as segmentation mask for the object indicated by the prompts. Figure 1 contains a rough visualization of the SAM architecture.

The greatest benefit of this general architecture lies in the decoupling of the computation of prompt embeddings and image features. The image only needs to be embedded once, while additional interactions only require a reuse of the prompt encoder and mask decoder. As long as the latter two networks are sufficiently light-weight, the user will be granted a real-time experience during the interactive usage of the model.

3.3. Adapting the Model During Test-Time

When performing interactive segmentation, we generally annotate a sequence of images instead of just a single one. This opens up the possibility of exploiting information gathered from segmenting previous images, in order to get better at segmenting future images. Similar to Kontogianni et al. [24] and Lenczner et al. [26], we make use of the fact that each click on its own constitutes a single reliably correct ground truth pixel. Since this piece of ground truth is available directly after being entered by the user, we can already adapt the model while still annotating the image. Additionally, we use the mask $\boldsymbol{m}^{\text{Res}}$ which results after the user is

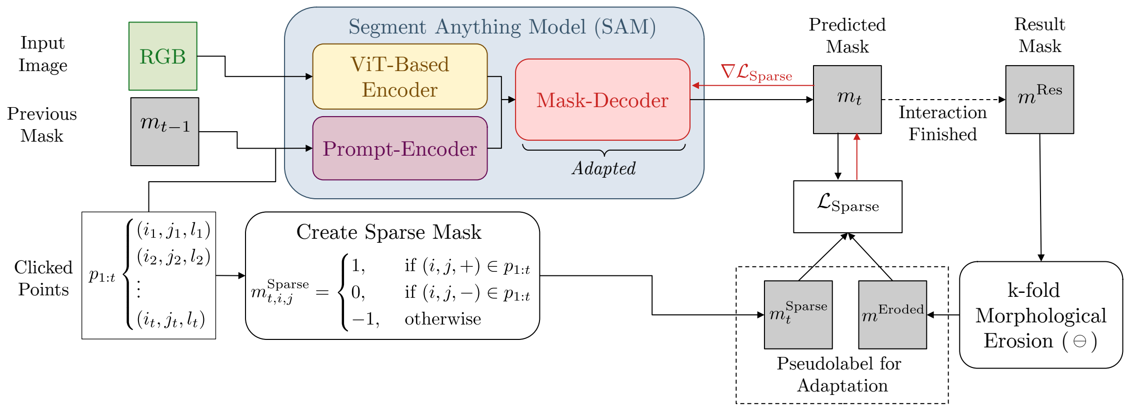

Figure 1. A rough description of the SAM architecture and the information used as pseudo-labels. Our method only adapts the mask-decoder which renders the computational effort of the backpropagation and optimization negligible. The gradient computation is displayed in red. The usage of pseudo-labels is discussed in Section 3.3.

done annotating the image. Depending on the users judgement, some areas of $\boldsymbol{m}^{\text{Res}}$ may still be erroneous. Since we especially suspect the borders between foreground and background to be faulty, we first subject the mask to multiple iterations of morphological erosion and then use this eroded mask $\boldsymbol{m}^{\text{Eroded}}$ as a pseudo-label to adapt the model to the image domain. When carrying out the adaptation, we only optimize the parameters of the decoder. A single execution of backpropagation and optimization with the Adam optimizer took 43.6 ms on a Nvidia V100 GPU vs. 13.1 ms for the corresponding forward pass. Since the accompanying optimization takes less than four times the time of the forward pass, the method doesn't impede any potential real time usage. Extracting the features with the backbone takes 116.9 ms. This operation, however, only has to be executed once per image. In the following paragraphs, we describe the variants of adaptation used by us.

Immediately using Clicks for Adaptation. As soon as the user makes a click $\boldsymbol{p}_t = (i_t, j_t, l_t)$, we have ground truth information for a particular pixel at our disposal. We can use all clicks $\boldsymbol{p}_{1:t}$ we have received up until that point in order to create a sparse mask $\boldsymbol{m}_t^{\text{Sparse}}$ with

$$\boldsymbol{m}^{\text{Sparse}}_{t,i,j} = \begin{cases} 1, & \text{if } (i,j,+) \in \boldsymbol{p}_{1:t} \\ 0, & \text{if } (i,j,-) \in \boldsymbol{p}_{1:t} \\ -1, & \text{otherwise} \end{cases} \tag{1}$$

where -1 marks unknown pixels. Let $\boldsymbol{m}_t$ be the segmentation mask that has been computed after that last click has been made. We then compute a sparse binary cross entropy loss

$$\mathcal{L}_{\text{Sparse}}(\boldsymbol{m}_t^{\text{Sparse}}, \boldsymbol{m}_t) = \frac{\sum_{i,j} 1_{\boldsymbol{m}^{\text{Sparse}}_{t,i,j}=1} \mathcal{L}_{\text{BCE}}(\boldsymbol{m}^{\text{Sparse}}_{t,i,j}, m_{t,i,j})}{\sum_{i,j} 1_{\boldsymbol{m}^{\text{Sparse}}_{t,i,j}=1}} + \frac{\sum_{i,j} 1_{\boldsymbol{m}^{\text{Sparse}}_{t,i,j}=0} \mathcal{L}_{\text{BCE}}(\boldsymbol{m}^{\text{Sparse}}_{t,i,j}, m_{t,i,j})}{\sum_{x,y} 1_{\boldsymbol{m}^{\text{Sparse}}_{t,i,j}=0}} \tag{2}$$

using $\boldsymbol{m}_t^{\text{Sparse}}$ as the label mask. We then immediately carry out an optimization step, thus progressively overfitting to the particular image as we continue annotating it. Note that this overfitting is deliberate and has to be reversed after we are done with the image. In order to achieve this, we reset the weights to their values before the image annotation, directly after we are done with the image.

Using all Clicks to adapt the Model to the Image Sequence. While the last paragraph describes a deliberate overfitting to the image, we also have the option to only carry out a single optimization step after we finish annotating the image. When doing this, we use all clicks that have been accumulated during the annotation of an image to create a single $\boldsymbol{m}^{\text{Sparse}}$ per image. The mask is created in the same fashion as before. This strategy adapts the model to the type of object and image domain of the test set, whilst acting less destructive on the parameters.

Using the Resulting Mask to Adapt the Model to the Image Sequence. Once the user regards the interactively created mask to be of sufficient quality, they stop the annotation and we obtain the result mask $\boldsymbol{m}^{\text{Res}} \in \{0,1\}^{H \times W}$. We can use this mask as a pseudo-label to adapt the model to the image sequence. In order to circumvent erroneous regions we will prune $\boldsymbol{m}^{\text{Res}}$ at the borders between foreground and background, where we estimate the risk of errors to be the highest. This is done by separating the foreground and background masks, iteratively eroding both of them and uniting them again. Let $\boldsymbol{m}^{\text{FG}} = \boldsymbol{m}^{\text{Res}}$ and $\boldsymbol{m}^{\text{BG}} = 1 - \boldsymbol{m}^{\text{Res}}$ be the foreground and background masks, respectively. We define $\gamma^k(\boldsymbol{m})$ to be a k-fold application of morphological erosion as

$$\gamma^0(\boldsymbol{m}) = \boldsymbol{m}, \tag{3}$$

$$\gamma^k(\boldsymbol{m}) = \gamma^{k-1}(\boldsymbol{m}) \ominus \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \tag{4}$$

where $\ominus$ is the symbol for the erosion operation. Then $\boldsymbol{m}^{\text{FG, Eroded}} = \gamma^k(\boldsymbol{m}^{\text{FG}})$ and $\boldsymbol{m}^{\text{BG, Eroded}} = \gamma^k(\boldsymbol{m}^{\text{BG}})$ are the eroded background and foreground masks. We will unite the two, resulting in the pruned pseudolabel mask $\boldsymbol{m}^{\text{Eroded}}$ with

$$m_{i,j}^{\text{Eroded}} = \begin{cases} 1, & \text{if } m_{i,j}^{\text{FG, Eroded}} = 1 \\ 0, & \text{if } m_{i,j}^{\text{BG, Eroded}} = 1 \\ -1, & \text{otherwise} \end{cases}. \tag{5}$$

We will carry out a single optimization step using $\mathcal{L}_{\text{Sparse}}$ after annotating each image.

Using multiple decoders for Multiple Classes. All of the aforementioned adaptation will inevitably overfit the model to a particular domain or type of object. It is however noteworthy, that the only part of the model to be adapted is the relatively lightweight decoder. This allows use to duplicate the parameters of the adapted module. In cases where we want to annotate multiple different classes, we use multiple copies of the original decoder, which are separately adapted to the respective object type or domain. We regard the memory overhead as negligible: For the version of SAM with the ViT-b backbone, we have 4.06M parameters for the decoder vs 89.7M parameters for the rest of the model. For the versions with the ViT-l and ViT-h backbones, the rest of the model has 308.3M and 637M parameters respectively, while the decoder size remains the same.

4. Experiments

4.1. Experimental Setting

Implementation Details. During training we only adapt the decoder in order to minimize the computational overhead of our method. We carry out all optimization with a sparse binary cross entropy loss, as described in Section 3.3. We use the Adam optimizer [22] with a learning rate of 10^{-6}. The resolution of the input images is 1024×1024, which is a pre-existing property of SAM. All experiments use the ViT-b backbone [12]. Whenever we use erosion, we carry out the iterative erosion with $k = 5$ iterations.

Metrics. When testing an interactive segmentation system, we want to exceed a certain IoU threshold T_{IoU} within n clicks. If the system is unable to do that, we consider the attempt at segmenting the image a failure and use n as surrogate value for the number of clicks when computing the $\text{NoC}_n@\text{T}_{\text{IoU}}$. The *Number of Clicks* ($\text{NoC}_n@\text{T}_{\text{IoU}}$) metric measures the average number of clicks on the test set, while the *Failure Rate* ($\text{FR}_n@\text{T}_{\text{IoU}}$) measures the percentage of images on which the segmentation failed. Out of the two metrics we regard the failure rate as the more important one for the following reason: While having to add an additional click on some images during the annotation process incurs a higher time effort, the failure rate measures the amount of images that cannot be segmented within a reasonable number of clicks at all.

Click Adaptation (CA): After each click, we can use all so far accumulated clicks to create a sparse label mask, with which we optimize the model to overfit to the image. We call this process *Click Adaptation (CA)*. In Section 3.3 we mentioned that this deliberate overfitting necessitates resetting the weight after each object, which we denote with an R for (R)eset in the tables. We may however choose to not perform this reset, and adapt our model continually over all images. We denote this by a C for (C)ontinual. No letter in the tables means that we do not use Click Adaptation at all.

Result Mask (RM): After being done with annotating an image, we can make use of the *Result Mask (RM)*. We could directly use the mask as a pseudolabel for optimization. We denote this with a U for (U)ntreated in the tables. As we will show however, this mask may still be erroneous and worsen our performance by subjecting our model to a partially false training signal. In order to circumvent this problem we may prune the masks foreground and background area by using iterative erosion. We denote this by an E for (E)rosion. No letter means that we do not make use of the result mask.

Click Mask (CM): After the annotation, we can use the accumulated clicks to form a sparse *Click Mask (CM)*, with which we can perform a single optimization step. In each configuration in which we do so, it is annotated by a checkmark (✓).

The table row containing no letter or checkmark means that we are not performing any form of adaptation, which constitutes our baseline. Whenever we use the Result Mask and the Click Mask in the same configuration, we merge them into a single mask. In all tables, the first line contains the baseline, while the second line contains our complete method. Figure 2 shows some qualitative examples.

4.2. Adaptation to Rare Objects

We will adapt SAM during usage on various datasets providing examples for rather obscure and uncommon sit-

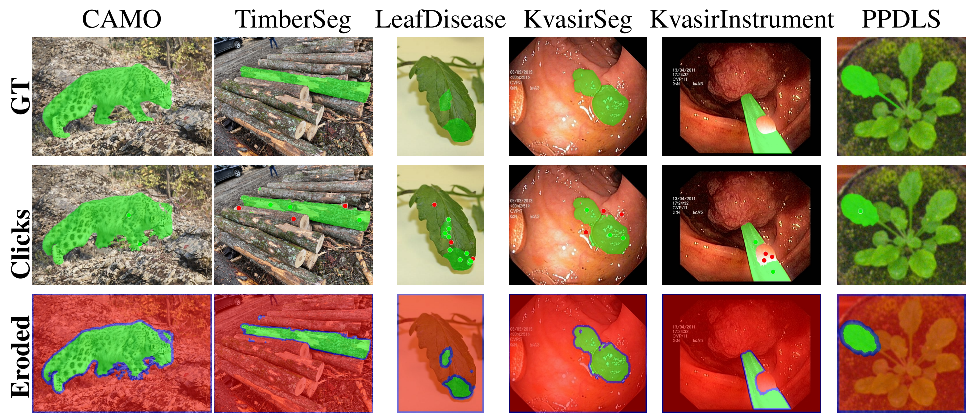

Figure 2. Examples for the masks occurring during the interaction. The *first* row contains the ground truth. The *second* row contains the annotated mask and the clicks. The *third* row contains examples for the eroded result mask. Green, red and blue correspond to foreground, background and the eroded area, respectively.

Configuration			Rooftop		DOORS		TrashCan		CAMO	
CA	RM	CM	NoC	FR	NoC	FR	NoC	FR	NoC	FR
			4.171	6.00	5.439	16.69	13.259	57.42	7.224	20.3
R	E	✓	3.667	3.93	4.877	13.50	11.488	40.49	7.310	17.2
R			3.755	3.93	5.149	12.25	11.847	39.41	7.382	18.2
C			3.834	3.93	5.222	12.73	11.932	41.42	7.212	17.1
	E		3.741	3.39	5.642	18.10	13.486	58.23	7.401	20.2
		✓	3.915	4.62	5.154	14.97	13.694	59.47	7.278	19.4
R		✓	3.707	3.70	5.326	12.83	11.796	40.38	7.402	17.0
R	U	✓	3.693	3.00	4.861	12.64	16.041	64.49	12.764	45.8
HQ-SAM			9.977	31.64	10.688	42.74	16.902	79.83	10.383	36.5
Configuration			ISTD		LeafDisease		PPDLS		TimberSeg	
CA	RM	CM	NoC	FR	NoC	FR	NoC	FR	NoC	FR
			11.584	40.68	14.624	62.07	6.239	23.76	11.564	48.50
R	E	✓	10.392	31.13	14.595	60.71	6.250	20.04	10.497	39.67
R			10.932	34.66	14.665	61.05	6.267	19.25	11.080	42.26
C			10.896	33.91	14.631	60.71	6.218	19.43	10.661	40.73
	E		11.295	38.80	14.690	61.05	5.955	21.42	10.745	43.32
		✓	11.596	41.73	14.517	60.54	5.988	21.56	10.933	43.92
R		✓	10.810	33.68	14.469	60.03	6.140	19.54	10.571	40.18
R	U	✓	15.017	57.97	14.918	62.41	14.387	49.40	16.710	74.76
HQ-SAM			18.757	89.32	16.519	74.49	10.173	3646	17.706	84.33

Table 1. The results on datasets displaying rare objects types. NoC means the $NoC_{20}@85$ metric and FR is the $FR_{20}@85$, describing the number of objects that could not be segmented after 20 clicks. For both metrics, a smaller value indicates a better performance. An explanation of the configurations can be found in Section 4.1.

uations. The Rooftop dataset [48] provides various remote sensing photos with annotated rooftops. The DOORS dataset [39] has been created for the segmentation of boulders. The TrashCan dataset [17] contains segmentation masks for underwater waste objects. CAMO [25, 56] is a dataset for the task of camouflaged object segmentation and ISTD [52] for shadow segmentation. Additionally, we have three datasets for agricultural applications: One dataset for leaf disease segmentation [2], PPDLS [38] for the segmentation of arabidopsis and tobacco leafs, and TimberSeg [14] for the segmentation of logs in forestry work.

We are first going to look at $NoC_{20}@85$ and $FR_{20}@85$ metrics. According to Table 1, our method reduces the FR on ISTD from 40.68 to 31.13, while reducing the NoC by more than one click. On TrashCan, our method even improves the FR from 57.42 to 40.49. It should also be noted

Configuration			Rooftop		DOORS		TrashCan		CAMO	
CA	RM	CM	NoC	FR	NoC	FR	NoC	FR	NoC	FR
			9.979	22.63	13.870	37.77	23.281	72.49	13.870	34.1
R	E	✓	8.891	18.21	13.163	33.62	20.527	54.06	13.488	28.3
R			8.961	18.24	14.996	36.30	20.979	53.86	13.719	29.6
C			9.358	19.86	14.623	35.35	21.032	53.40	13.573	29.1
	E		9.321	19.63	14.965	42.47	23.700	73.30	14.082	33.0
		✓	9.314	19.40	13.629	35.96	23.976	74.27	14.063	33.6
R		✓	9.127	18.94	15.533	37.33	20.925	52.20	13.503	28.5
R	U	✓	9.339	19.40	13.082	33.31	25.221	70.75	20.840	54.2
HQ-SAM			19.637	53.12	20.475	61.10	26.844	87.09	18.010	50.0
Configuration			ISTD		LeafDisease		PPDLS		TimberSeg	
CA	RM	CM	NoC	FR	NoC	FR	NoC	FR	NoC	FR
			18.744	49.02	24.255	72.62	13.260	38.55	20.358	62.64
R	E	✓	16.660	40.00	23.617	70.24	13.782	30.28	18.735	52.15
R			17.411	41.80	24.138	71.26	13.682	31.30	19.018	54.46
C			17.302	40.90	24.214	72.28	13.276	30.88	19.026	54.00
	E		18.329	47.89	24.320	72.62	12.877	36.17	19.306	58.21
		✓	19.574	53.08	24.226	71.60	12.574	35.07	19.436	58.76
R		✓	17.217	41.35	24.153	72.11	13.447	31.22	18.874	53.49
R	U	✓	22.729	59.40	24.221	72.11	22.892	56.13	26.319	79.89
HQ-SAM			28.337	91.20	26.269	81.63	18.180	49.87	27.364	88.58

Table 2. The results on datasets displaying rare objects types. NoC means the $NoC_{30}@90$ metric and FR is the $FR_{30}@90$, describing the number of objects that could not be segmented after 30 clicks. For both metrics, a smaller value indicates a better performance. An explanation of the configurations can be found in Section 4.1.

Configuration			KvasirInstrument		CVCClinicDB		GlaS		KvasirSeg	
CA	RM	CM	NoC	FR	NoC	FR	NoC	FR	NoC	FR
			2.137	1.86	4.935	8.17	7.485	14.64	3.615	2.7
R	E	✓	2.166	1.53	4.551	5.56	6.759	10.20	3.145	1.4
R			2.388	2.71	4.828	5.39	7.377	13.53	3.314	1.1
C			2.239	2.37	4.900	7.03	7.250	13.27	3.352	1.2
	E		2.136	1.69	4.471	4.41	8.437	20.65	3.123	1.2
		✓	2.178	2.37	4.637	5.39	8.539	20.72	3.281	1.2
R		✓	2.305	2.37	4.757	6.21	7.576	15.29	3.273	1.0
R	U	✓	2.251	2.20	5.087	6.70	13.946	49.15	7.684	20.3
HQ-SAM			7.973	18.31	15.789	66.01	18.845	88.89	10.504	34.1

Table 3. The results medical datasets. NoC means the $NoC_{20}@85$ metric and FR is the $FR_{20}@85$, describing the number of objects that could not be segmented after 20 clicks. For both metrics, a smaller value indicates a better performance. An explanation of the configurations can be found in Section 4.1.

that the results imply that SAM is unable to segment over half of the objects in the TrashCan and LeafDisease datasets to a satisfying degree. While our complete method slightly increases the NoC on the CAMO and PPDLS datasets, it still lowers the FR which we regard as the more crucial metric. In order to see the effect of using the untreated mask, we also run a version of our complete method without pruning the mask by erosion. As it turns out, eroding the mask is important due to potential erroneous areas at the edge of foreground and background area. The resulting false training signal manages to increase the FR by even more than two times on CAMO.

In Table 2, where the model needs to achieve an IoU of 90 within 30 clicks, we see an exacerbation of the problem SAM has with segmenting objects that are alien to its original training set. The FR values of the unadapted SAM model are 72.49, 72.62 and 62.64 on TrashCan, LeafDisease, and TimberSeg, respectively. This indicates that SAM is almost inept to segment these types of data to an IoU of 90 with the actual object surface, which would be consid-

Configuration			KvasirInstrument		CVCClinicDB		GlaS		KvasirSeg	
CA	RM	CM	NoC	FR	NoC	FR	NoC	FR	NoC	FR
			3.651	4.75	10.301	19.61	14.995	33.53	6.378	5.8
R	E	✓	3.825	4.58	8.585	10.46	11.684	19.15	5.580	3.9
R			4.063	5.42	9.343	14.05	13.341	24.12	6.397	5.7
C			4.041	5.42	9.041	12.75	13.331	23.73	6.057	4.4
	E		3.749	5.08	9.588	14.87	15.884	35.49	5.573	3.4
		✓	3.647	4.75	9.458	14.87	16.729	40.13	6.106	4.9
R		✓	4.237	5.93	9.253	13.40	13.690	25.23	6.178	5.7
R	U	✓	4.239	5.76	12.446	21.57	22.744	55.29	16.168	34.2
HQ-SAM			13.698	30.85	24.139	70.75	28.888	93.86	17.410	44.4

Table 4. The results medical datasets. NoC means the $NoC_{30}@90$ metric and FR is the $FR_{30}@90$, describing the number of objects that could not be segmented after 30 clicks. For both metrics, a smaller value indicates a better performance. An explanation of the configurations can be found in Section 4.1.

ered necessary when producing annotations for new data. In the case of TrashCan and TimberSeg we manage to reduce the FR by 18.43 and 10.49 percentage points, respectively. The largest improvements regarding the NoC are incurred on TrashCan with a reduction of 2.754 clicks. On PPDLS, we again have a reduction in the FR for the cost of slightly higher NoC. It should be noted, that our complete method (CA = R, RM = E, CM = ✓) reduces the failure rate in all cases, and thus widens the applicability of SAM for uncommon domains.

4.3. Results on Medical Image Segmentation

In order to investigate the efficacy of the adaptation method on medical image segmentation, we consider four different datasets: KvasirInstrument [20] contains segmented images of tools used in the gastrointestinal tract. CVCClinicDB [5] and KvasirSeg [19] are two datasets for the task of polyp segmentation, while the GlaS dataset [43, 44] provides data for the task of gland segmentation in colon histology. The results for using our method on medical data generally comport with the results on other rare objects. It should first be noted that our complete method causes a reduction of the failure rate in all cases. In Table 3 we see the complete method decreasing the FR on KvasirSeg from 2.7 to 1.4, which is a relative reduction of 48.1%. On GlaS, the FR is lowered from 14.64 to 10.20 and the NoC is lowered from 7.485 to 6.759. On KvasirSeg and GlaS, the untreated result mask with a partially erroneous signal causes the most damage. It increases the failure rate by 18.9 and 38.95 percentage points in comparison to the full method with the eroded mask on each of the respective datasets. In Table 4, we can see a reduction in the FR by 14.38 percentage points, as well as a reduction in the NoC by 3.311 clicks on GlaS. On CVCClinicDB the FR is lowered by 9.15 percentage points, which equates to a reduction of 46.6%, while the NoC is lowered by 1.716 clicks. On KvasirInstrument, the adaptation method causes a slightly higher NoC, but still lowers the failure rate. We also want to assure that this decreased performance does not stem from potential low-quality masks in SA-1B. For this purpose, we also tested HQ-SAM [21] on our datasets, which is a slightly altered version of SAM that has been fine-tuned on high-quality human-annotated masks. In Tables 1 to 4 we see that HQ-SAM performs drastically worse than SAM. We assume this to be the case due to a decrease in diversity which occurred during fine tuning. The novel segmentation head and HQ token have only ever been trained on the vastly smaller HQSeg-44K, rendering them particularly inept for the usage on unknown domains.

5. Conclusion

In our paper we applied the Segment Anything Model to uncommon situations. We did so for the specific task of interactive segmentation and evaluated appropriate metrics: The Number of Clicks (NoC) and the Failure Rate (FR). Despite the model being trained on the largest dataset for instance masks to date, we see considerable problems when confronting the model with data that differs from regular consumer images. In some situations the model failed to segment more than half of the objects in the dataset, as reflected by the Failure Rate. This inability to segment certain objects poses a crucial limit to the model. In order to alleviate this problem we propose an efficient test time adaptation method. All techniques are restricted to using information that occurs during usage and do not require any previous fine-tuning on existing datasets. In addition to that, they only incur a minimal computational overhead in order to not hamper any potentially required real-time capabilities. With the help of our method we manage to lower the Failure Rate on twelve different datasets and lower the NoC on ten of them. We thus conclude that the information available during test time provides a useful tool when applying a foundation model such as SAM to uncommon domains.

References

[1] Eirikur Agustsson, Jasper RR Uijlings, and Vittorio Ferrari. Interactive full image segmentation by considering all regions jointly. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11622–11631, 2019. 2

[2] Fakhre Alam. Leaf disease segmentation. https://www.kaggle.com/datasets/fakhrealam9537/leaf-disease-segmentation-dataset, 2021. Accessed: 2023-09-14. 6

[3] 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. 2

[4] Junjie Bai and Xiaodong Wu. Error-tolerant scribbles based interactive image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 392–399, 2014. 2

[5] Jorge Bernal, F Javier Sánchez, Gloria Fernández-Esparrach, Debora Gil, Cristina Rodríguez, and Fernando Vilariño. Wm-dova maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians. *Computerized medical imaging and graphics*, 43:99–111, 2015. 8

[6] Tianrun Chen, Lanyun Zhu, Chaotao Ding, Runlong Cao, Shangzhan Zhang, Yan Wang, Zejian Li, Lingyun Sun, Papa Mao, and Ying Zang. Sam fails to segment anything?–sam-adapter: Adapting sam in underperformed scenes: Camouflage, shadow, and more. *arXiv preprint arXiv:2304.09148*, 2023. 2

[7] Xi Chen, Zhiyan Zhao, Feiwu Yu, Yilei Zhang, and Manni Duan. Conditional diffusion for interactive segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7345–7354, 2021. 2

[8] Xi Chen, Zhiyan Zhao, Yilei Zhang, Manni Duan, Donglian Qi, and Hengshuang Zhao. Focalclick: Towards practical interactive image segmentation. In *CVPR*, 2022. 2, 3

[9] Xi Chen, Yau Shing Jonathan Cheung, Ser-Nam Lim, and Hengshuang Zhao. Scribbleseg: Scribble-based interactive image segmentation. *arXiv preprint arXiv:2303.11320*, 2023. 2

[10] Junlong Cheng, Jin Ye, Zhongying Deng, Jianpin Chen, Tianbin Li, Haoyu Wang, Yanzhou Su, Ziyan Huang, Jilong Chen, Lei Jiang, Hui Sun, Junjun He, Shaoting Zhang, Min Zhu, and Yu Qiao. Sam-med2d, 2023. 2

[11] Lei Ding, Kun Zhu, Daifeng Peng, Hao Tang, and Haitao Guo. Adapting segment anything model for change detection in hr remote sensing images. *arXiv preprint arXiv:2309.01429*, 2023. 2

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 5

[13] Camille Dupont, Yanis Ouakrim, and Quoc Cuong Pham. Ucp-net: unstructured contour points for instance segmentation. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 3373–3379. IEEE, 2021. 2

[14] Jean-Michel Fortin, Olivier Gamache, Vincent Grondin, François Pomerleau, and Philippe Giguère. Instance segmentation for autonomous log grasping in forestry operations. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6064–6071. IEEE, 2022. 6

[15] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1551–1560, 2021. 2

[16] Yuying Hao, Yi Liu, Juncai Peng, Haoyi Xiong, Guowei Chen, Shiyu Tang, Zeyu Chen, and Baohua Lai. Rais: Robust and accurate interactive segmentation via continual learning. *arXiv preprint arXiv:2210.10984*, 2022. 2

[17] Jungseok Hong, Michael Fulton, and Junaed Sattar. Trashcan: A semantically-segmented dataset towards visual detection of marine debris. *ArXiv*, abs/2007.08097, 2020. 6

[18] Won-Dong Jang and Chang-Su Kim. Interactive image segmentation via backpropagating refinement scheme. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5297–5306, 2019. 2

[19] Debesh Jha, Pia H Smedsrud, Michael A Riegler, Pål Halvorsen, Thomas de Lange, Dag Johansen, and Håvard D Johansen. Kvasir-seg: A segmented polyp dataset. In *MultiMedia Modeling: 26th International Conference, MMM 2020, Daejeon, South Korea, January 5–8, 2020, Proceedings, Part II 26*, pages 451–462. Springer, 2020. 1, 8

[20] Debesh Jha, Sharib Ali, Krister Emanuelsen, Steven A. Hicks, Vajira Thambawita, Enrique Garcia-Ceja, Michael A. Riegler, Thomas de Lange, Peter T. Schmidt, Håvard D. Johansen, Dag Johansen, and Pål Halvorsen. Kvasir-instrument: Diagnostic and therapeutic tool segmentation dataset in gastrointestinal endoscopy. In *MultiMedia Modeling*, pages 218–229, Cham, 2021. Springer International Publishing. 8

[21] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. In *NeurIPS*, 2023. 2, 8

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[23] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv e-prints*, pages arXiv–2304, 2023. 1, 2, 3

[24] Theodora Kontogianni, Michael Gygli, Jasper Uijlings, and Vittorio Ferrari. Continuous adaptation for interactive object segmentation by learning from corrections. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 579–596. Springer, 2020. 2, 3

[25] Trung-Nghia Le, Tam V. Nguyen, Zhongliang Nie, Minh-Triet Tran, and Akihiro Sugimoto. Anabranch network for

camouflaged object segmentation. *Journal of Computer Vision and Image Understanding*, 184:45–56, 2019. 6

[26] Gaston Lenczner, Adrien Chan-Hon-Tong, Nicolas Luminari, Bertrand Le Saux, and Guy Le Besnerais. Interactive learning for semantic segmentation in earth observation. In *ECML-PKDD 2020, MACLEAN Workshop*, 2020. 2, 3

[27] 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 *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 1

[28] Zheng Lin, Zhao Zhang, Lin-Zhuo Chen, Ming-Ming Cheng, and Shao-Ping Lu. Interactive image segmentation with first click attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13339–13348, 2020. 2

[29] Zheng Lin, Zhao Zhang, Zi-Yue Zhu, Deng-Ping Fan, and Xia-Lei Liu. Sequential interactive image segmentation. *Computational Visual Media*, 9(4):753–765, 2023. 2

[30] Qin Liu, Zhenlin Xu, Yining Jiao, and Marc Niethammer. isegformer: interactive segmentation via transformers with application to 3d knee mr images. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 464–474. Springer, 2022. 2

[31] Qin Liu, Zhenlin Xu, Gedas Bertasius, and Marc Niethammer. Simpleclick: Interactive image segmentation with simple vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22290–22300, 2023. 2, 3

[32] Mathias Micheelsen Lowes, Jakob L Christensen, Bjørn Schreblowski Hansen, Morten Rieger Hannemose, Anders Bjorholm Dahl, and Vedrana Dahl. Interactive scribble segmentation. In *Proceedings of the Northern Lights Deep Learning Workshop*, 2023. 2

[33] Katja Ludwig, Daniel Kienzle, Julian Lorenz, and Rainer Lienhart. Detecting arbitrary keypoints on limbs and skis with sparse partly correct segmentation masks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 461–470, 2023. 1

[34] Katja Ludwig, Julian Lorenz, Robin Schön, and Rainer Lienhart. All keypoints you need: Detecting arbitrary keypoints on the body of triple, high, and long jump athletes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5178–5186, 2023. 1

[35] Sabarinath Mahadevan, Paul Voigtlaender, and Bastian Leibe. Iteratively trained interactive segmentation. *arXiv preprint arXiv:1805.04398*, 2018. 2

[36] Soumajit Majumder and Angela Yao. Content-aware multi-level guidance for interactive instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11602–11611, 2019. 2

[37] K.-K. Maninis, S. Caelles, J. Pont-Tuset, and L. Van Gool. Deep extreme cut: From extreme points to object segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[38] Massimo Minervini, Andreas Fischbach, Hanno Scharr, and Sotirios A Tsaftaris. Finely-grained annotated datasets for image-based plant phenotyping. *Pattern recognition letters*, 81:80–89, 2016. 6

[39] Mattia Pugliatti and Francesco Topputo. Doors: Dataset for boulders segmentation, 2022. 6

[40] Gianmarco Roggiolani, Matteo Sodano, Tiziano Guadagnino, Federico Magistri, Jens Behley, and Cyrill Stachniss. Hierarchical approach for joint semantic, plant instance, and leaf instance segmentation in the agricultural domain. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9601–9607. IEEE, 2023. 1

[41] Luyue Shi, Xuanye Zhang, Yunbi Liu, and Xiaoguang Han. A hybrid propagation network for interactive volumetric image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 673–682. Springer, 2022. 2

[42] Qingxuan Shi, Yihang Li, Huijun Di, and Enyi Wu. Self-supervised interactive image segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 2023. 2

[43] Korsuk Sirinukunwattana, David RJ Snead, and Nasir M Rajpoot. A stochastic polygons model for glandular structures in colon histology images. *IEEE transactions on medical imaging*, 34(11):2366–2378, 2015. 8

[44] Korsuk Sirinukunwattana, Josien PW Pluim, Hao Chen, Xiaojuan Qi, Pheng-Ann Heng, Yun Bo Guo, Li Yang Wang, Bogdan J Matuszewski, Elia Bruni, Urko Sanchez, et al. Gland segmentation in colon histology images: The glas challenge contest. *Medical image analysis*, 35:489–502, 2017. 8

[45] Konstantin Sofiiuk, Ilia Petrov, Olga Barinova, and Anton Konushin. f-brs: Rethinking backpropagating refinement for interactive segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8623–8632, 2020. 2

[46] Konstantin Sofiiuk, Ilya A Petrov, and Anton Konushin. Reviving iterative training with mask guidance for interactive segmentation. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 3141–3145. IEEE, 2022. 2, 3

[47] Junha Song, Jungsoo Lee, In So Kweon, and Sungha Choi. Ecotta: Memory-efficient continual test-time adaptation via self-distilled regularization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[48] Xiaolu Sun, C Mario Christoudias, and Pascal Fua. Free-shape polygonal object localization. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 317–332. Springer, 2014. 6

[49] An Wang, Mobarakol Islam, Mengya Xu, Yang Zhang, and Hongliang Ren. Sam meets robotic surgery: An empirical study on generalization, robustness and adaptation, 2023. 2

[50] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020. 2

[51] Guotai Wang, Wenqi Li, Maria A Zuluaga, Rosalind Pratt, Premal A Patel, Michael Aertsen, Tom Doel, Anna L David,

Jan Deprest, Sébastien Ourselin, et al. Interactive medical image segmentation using deep learning with image-specific fine tuning. *IEEE transactions on medical imaging*, 37(7): 1562–1573, 2018. 2

[52] Jifeng Wang, Xiang Li, and Jian Yang. Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1788–1797, 2018. 6

[53] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7201–7211, 2022. 2

[54] Junde Wu, Rao Fu, Huihui Fang, Yuanpei Liu, Zhaowei Wang, Yanwu Xu, Yueming Jin, and Tal Arbel. Medical sam adapter: Adapting segment anything model for medical image segmentation. *arXiv preprint arXiv:2304.12620*, 2023. 2

[55] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–381, 2016. 2

[56] Jinnan Yan, Trung-Nghia Le, Khanh-Duy Nguyen, Minh-Triet Tran, Thanh-Toan Do, and Tam V. Nguyen. Mirrornet: Bio-inspired camouflaged object segmentation. *IEEE Access*, 9:43290–43300, 2021. 6

[57] Shiyin Zhang, Jun Hao Liew, Yunchao Wei, Shikui Wei, and Yao Zhao. Interactive object segmentation with inside-outside guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12234–12244, 2020. 2

[58] Xu Zhao, Wenchao Ding, Yongqi An, Yinglong Du, Tao Yu, Min Li, Ming Tang, and Jinqiao Wang. Fast segment anything, 2023. 2

[59] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017. 1

[60] Chungang Zhuang, Shaofei Li, and Han Ding. Instance segmentation based 6d pose estimation of industrial objects using point clouds for robotic bin-picking. *Robotics and Computer-Integrated Manufacturing*, 82:102541, 2023. 1