Interactive Segmentation for Diverse Gesture Types Without Context - Supplementary Materials

Josh Myers-Dean[1], Yifei Fan[2], Brian Price[2], Wilson Chan[2], and Danna Gurari[1,3]
[1]University of Colorado Boulder [2]Adobe Research [3]University of Texas at Austin

1. Supplementary Materials

This document supplements the main paper with the following:

1. Alternative inputs for interactive segmentation. (supplements **Section 2**)
2. Examples of annotations from our user study. (supplements **Section 3**)
3. Examples of ground truth region segmentations that illustrate the diversity supported in our DIG dataset. (supplements **Section 4.1**)
4. Expanded discussion of the different gesture types supported in our DIG dataset. (supplements **Section 4.1**)
5. Implementation details for creating previous segmentations for our DIG dataset. (supplements **Section 4.1**)
6. Characterization of segmentations included in DIG with respect to the setting of segmentation creation versus segmentation refinement. (supplements **Section 4.2**)
7. Expanded discussion of our evaluation metric, RICE. (supplements **Sections 5 and 6**)
8. Expanded discussion of models benchmarked on our DIG dataset. (supplements **Section 6**)
9. Additional results for benchmarked models. (supplements **Section 6**)

2. Non-Gesture Segmentation Methods

Other forms of segmentation (e.g., automatic) and interaction (e.g., language) could be used to select a region in an image. With that said, we discuss here their critical shortcomings that are not present when utilizing gestures.

For automatic methods [1, 9, 12, 15], they could allow a user to segment an entire image and then choose their

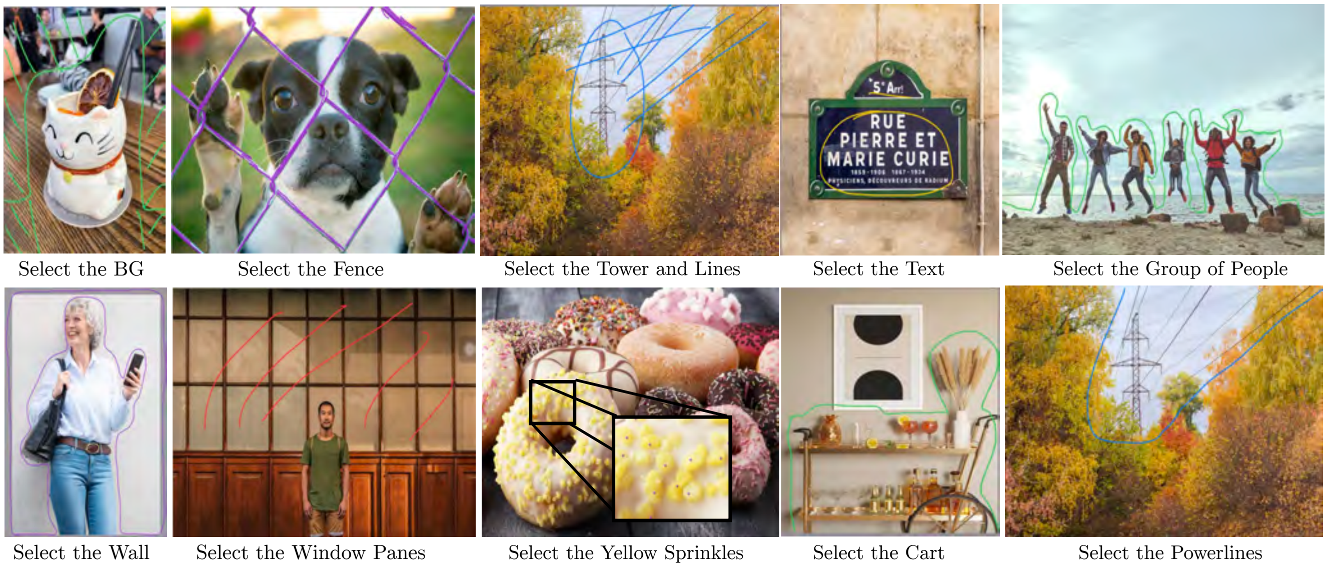

Figure 1. Examples of annotations generated in our user study about gestures. The text below each annotated image is a truncated description of the task described to the study participants. BG stands for background.

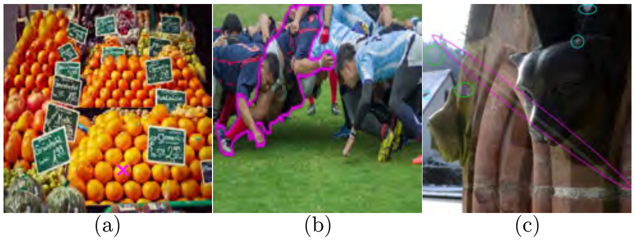

Figure 2. Scenarios representing the difficulty of using language instead of gestures: (a) selecting a specific object, (b) specifying distractors to remove from an image, (c) performing corrections on a previous segmentation.

selection based off of the set of segmentations. However, these methods not only lack the ability to infer user intent, as noted in the main paper, but may be unable to segment to a user's desired level of granularity (e.g., a user wants to segment a petal on a flower but a model may not support part segmentation). Furthermore, automatic methods lack the capacity for ongoing adjustments to a segmentation, as they segment an entire scene in a single operation. Conversely, the capability for users to iteratively refine segmentation until satisfaction represents a fundamental aspect of interactive segmentation. This divergence renders automatic methods inappropriate for our proposed task.

Language models have had significant impact in many vision problems such as generation [7,14], and can certainly be valuable in aiding selection [2,5,11,17]. However, it brings important limitations. We highlight three key limitations below.

1. Language models target only a tiny fraction of the 7000+ languages spoken in the world [2], and primarily focus on the most successful nations. Relying only on language would discriminate against most language groups, many of which encompass the most disenfranchised peoples, at least until vision-language models achieve the same level of accuracy for all languages as they have for English. Gestures, in contrast, can be easily understood and used by those speaking any language.
2. Many objects/parts/regions are difficult to identify using language. To illustrate, consider three scenarios: (a) Describing which objects to manipulate is tricky as images could have repeated objects (like many oranges in a fruit stand), and users might want to adjust a subset of these objects (e.g., the magenta "X" marked orange in Figure 2(b)). The intended objects may be difficult to specify in cluttered scenes. (b) Users might struggle to articulate the desired changes to an image or region, like removing distracting color blobs (cyan-encircled areas) or unwanted elements (green-encircled lens flare) in Figure 2(b). Simply marking these would be easier than verbalizing. (c) Correcting errors from a previous segmentation could be hard to verbalize, whereas gesturing at the issue is simple. For instance, refining the player selection (e.g., removing body parts from other players) in Figure 2(c) is clearer through gestures than words.
3. Using language alone could result in an inequitable experience for disabled users. For example, typing a sentence to segment a region could result in more work for users with motor impairments. This lack of accessibility means that users could be excluded from participating in or benefiting from interactive segmentation applications that rely solely on language. In comparison, using the gesture that fits a user's unique needs (which is supported under our task) could be considerably faster and less limiting.

To summarize, while language is a valuable tool for computer vision tasks, it is limited in the scope of interactive segmentation. As highlighted above, language-based models are: unable to achieve comparable performance across different languages, tedious when specifying the potentially many corrections a segmentation may have, and could lead to inequitable experiences among users. In contrast, gestures are: language agnostic, trivial to specify corrections with, and can be adapted to suite the needs of users with varying speech, motor, and visual [10] abilities.

3. User Study Annotation Examples

We show examples of the annotations for eight different tasks in Figure 1. These exemplify the observation in our main paper that multiple gesture types are used, with lasso the most popular (e.g., tower, powerlines, large amounts of text, group of people, car). We also found other gestures occur – for example, we observed individuals use a combination of a lasso and scribbles to denote the background behind a cup, scribbles to select powerlines, and multiple clicks to select the many sprinkles on a donut. The diversity of gesture types underscores the need for algorithms to support multiple gesture types simultaneously.

4. Ground Truth Segmentations of Regions

To exemplify the diversity of types of regions supported in our paper, we show here examples of ground truths for (1) non-occluded regions in Figure 3(a), (2) occluded regions (i.e., *parts*) in Figure 3(b), and (3) multi-region segmentations in Figure 3(c). We also exemplify ground truth corrections for a previous segmentation in Figure 4; i.e., each correction has its own ground truth.

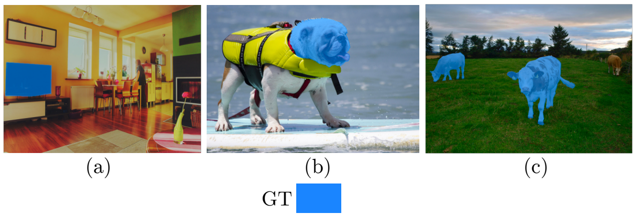
(a) (b) (c)

Figure 3. Examples of ground truth segmentation for multiple segmentation region types captured in our DIG dataset: (a) object (i.e., non-occluded object), (b) object part (i.e., occluded object), and (c) multi-region (i.e., multiple objects).

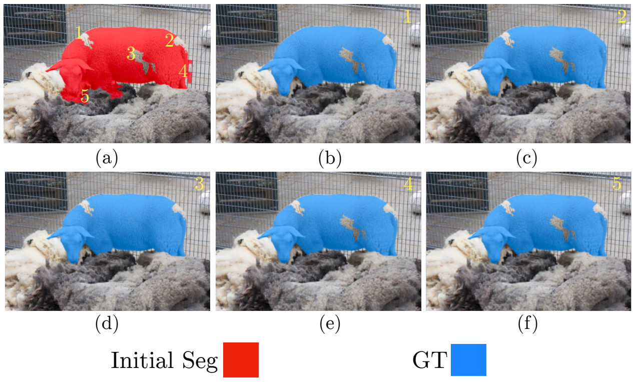
(a) (b) (c) (d) (e) (f)

Figure 4. Examples of ground truth used for training algorithms for segmentation refinement. Each correction in an initial segmentation has a corresponding ground truth. (a) is the initial segmentation with each correction denoted with a number, (b) is the ground truth for correction 1, (c) is the ground truth for correction 2, and (d)-(f) are the ground truths for corrections 3-5.

5. Gesture Types Supported in DIG

5.1. Gesture Implementation Details

For gesture annotations, we wanted a thickness that would neither be too small (i.e., a single pixel is difficult for a human to discern where the gesture is) nor too large that gestures would be indiscernible (e.g., a lasso that looks like a scribble). We chose as a heuristic a radius of 5 pixels from the center of each point in a gesture annotation. We used tools from Scikit-Image [19] and NumPy [8] to create all markings.

Lasso Generation. We describe here how we sample points from a region boundary to create both coarse and tight lassos. To construct lassos, we uniformly sample N points from the relevant boundary based on the interactive segmentation setting, randomly "jitter" points to simulate user annotation noise, and then interpolate between the points. For segmentation creation, the points are sampled from the ground truth segmentation of the target region. When refining a segmentation, the points are sampled from an erroneous region (e.g., the missing leg of the turtle in Figure 1 of the main paper) such that gestures target a specific region to correct (i.e., add or subtract pixels). Coarser lassos are simulated by applying a morphological dilation operator to the boundary. Formally, let L be the set of boundary points around a region and L' be the totally ordered set of newly sampled points. We then model the number of sampled points as $N \sim \mathcal{U}(\frac{|L|}{512}, \frac{|L|}{8})$, where $\mathcal{U}(\cdot)$ is a discrete uniform distribution over $[a, b]$ with $a, b \in \mathbb{Z}$ and $N = |L'|$. In the case of a degenerate lasso (i.e., all points in L' are colinear), we resample points up to 10 times. In order to simulate user noise, we randomly "jitter" a point, p, in L' before interpolating between points. Formally, $p = p + \epsilon$, $\forall_p \in L'$, where $\epsilon \sim \mathcal{U}(-J, J)$ and J is our "jitter" parameter. We use $J = 4$ for loose lassos and $J = 0$ for tight lassos.

Scribble Generation. We describe here how we permit scribbles to pass outside a region's boundary as well as how we simulate smoother curves. First, we create the B-spline by randomly sampling 4 to 6 (x, y) pairs from a target region or previous segmentation. Then, to permit scribbles to pass outside of the target region's boundaries and to simulate simpler curves, we perturb the sampled points by independently sorting the x and y coordinates before interpolating. We chose as heuristics to sort the x coordinates with 30% probability and the y coordinates with 60% probability. This allows the curves to pass outside of the boundaries of regions as the new (x, y) pairs may not exist within the region. Moreover, sorted points lead to visually less complex curves.

Rectangle Generation. We describe how we augment perfect bounding boxes into diverse rectangles that encapsulate a region of interest. Given a bounding box, B of the form $[x_{min}, y_{min}, x_{max}, y_{max}]$, we augment it as follows:

$$x_{min/max} = x_{min/max} + v \cdot g_i \cdot (x_{max} - x_{min}) \quad (1)$$

$$y_{min/max} = y_{min/max} + v \cdot g_i \cdot (y_{max} - y_{min}) \quad (2)$$

where the hyper-parameter v controls the variation of the rectangle and $g_i \sim \mathcal{N}(0, 1)$, $\forall_i \in \{0, 1, 2, 3\}$, where i corresponds to an index (e.g., $0 = y_{min}$) in B. We chose to randomly set $v \in [0.10, 0.15]$. Since g_i can go both positive and negative, we can "jitter" the rectangle across multiple directions.

5.2. Gesture Timing

We show the breakdown of how long it takes to generate each gesture type on average in Table 1. We report the mean time (± standard deviation) of each gesture type for approximately 4,740 regions (i.e., a random region from each image in the DIG validation and test splits). Computations are performed on an Intel Xeon Platinum 8275CL CPU. Overall, we observe that scribbles take approximately double the time of clicks and rectangles while lassos take 5-8 times longer. This underscores the impracticality of generating diverse gestures on the fly.

	Click	Scribble	Loose Lasso	Tight Lasso	Rectangle
Seconds	.004 ± .001	.009 ± .010	.035 ± .019	.023 ± .022	.004 ± .001

Table 1. Mean time ± standard deviation to generate each gesture type. We observe a large difference between the slowest gestures (i.e., lassos) and the quickest (i.e., clicks and rectangles).

5.3. Gesture Examples

We show additional examples of gesture annotations in Figure 5 to further highlight the diversity of gestures in our DIG dataset. For example, in the second row of Figure 5, we observe a nearly perfect circle lasso in (a) followed by an incomplete lasso in (b). Additionally, in the second row of Figure 5(c), we observe a scribble going out of the frame and rejoining (i.e., disconnected), contrasting the simpler scribble in the top row. Finally, we observe for clicks a diversity of positions and for rectangles varying amount of background content contained within the rectangle. The diversity of gestures present in DIG supports training models to account for the wide range of possible human interactions.

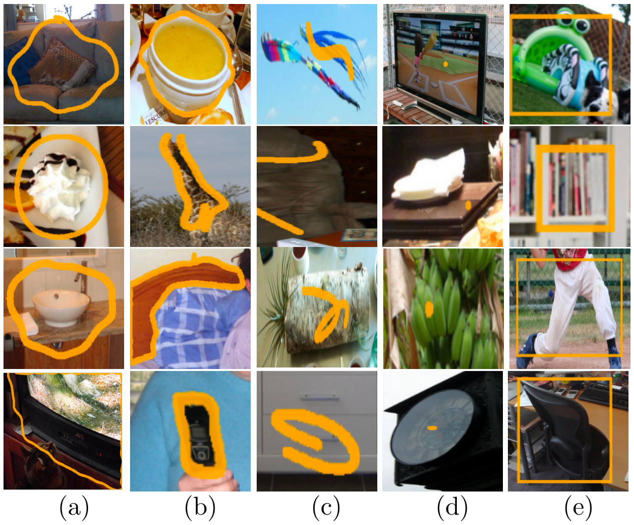

Figure 5. Examples of diversity within gesture types. We crop the image to the desired region to focus on the gesture annotations. The figure shows (a) loose lassos, (b) tight lassos, (c) scribbles, (d) clicks, and (e) rectangles.

6. Previous Segmentation Construction

For the construction of previous segmentations, we follow the approach employed by FocalClick [3]. Specifically, we only retain previous segmentations results that have IoU scores between 0.75 and 0.85 with a region's ground truth. This lower bound is motivated by prior work which shows that users of interactive segmentation methods tended to discard previous segmentations when IoU scores fell below 0.75. The upper bound is motivated by click-based segmentation methods which use 0.85 as the target IoU when evaluating using the NoC metric.

7. DIG Segmentation Setting Characterization

We show the frequency of gesture types for segmentation creation and refinement in Table 2. For the setting where no previous segmentation is present, we observe a balanced number of gesture types with a mean of 5 gestures per region. When no previous segmentation is present, we are always adding pixels to a segmentation, thus we observe no gestures intended for subtraction. For the setting when a previous segmentation is present, there are 18.96% fewer objects than when a previous segmentation is not present. A reason for this is that we disregard all corrections smaller than 100 pixels in areas as those corrections could be unnecessarily challenging for interactive segmentation methods. We again observe a balanced number of gesture types with a mean of 9.1 gestures per region in an image. We observe slightly more gestures whose intended context is *addition* as we include multi-region segmentation scenarios, which always involve adding pixels.

8. Evaluation Metric: RICE

8.1. RICE Implementation

We define RICE to take into account both the starting IoU of the previous segmentation and the ground truth, as well as the IoU of the prediction with the ground truth. Specifically, let $\hat{y}$ be the segmentation output of an interactive segmentation model, g be the ground truth of a region, and m be the previous segmentation. Additionally, let

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3}$$

be the intersection-over-union for some region A and some ground truth B. Then, $\alpha = IoU(\hat{y}, g)$, $\alpha \in [0, 1]$ and $\beta = IoU(m, g)$, $\beta \in [0, 1)$ given that $\hat{y} \in \{0, 1\}^{H \times W}$ is the output of an interactive segmentation method, $g \in \{0, 1\}^{H \times W}$ is the ground truth for the region of interest, $m \in \{0, 1\}^{H \times W}$ is an initial segmentation to refine. We only consider initial segmentations with $\beta \in [0, 1)$ for this metric since $\beta = 1$ would be a perfect segmentation with no available refinements. When creating a segmentation with no previous segmentation, RICE simplifies to $IoU(\cdot)$.

RICE ranges from a minimum value of -1 to a maximum value of 1. Intuitively, a positive score means that an algorithm corrected a previous segmentation (i.e., $\alpha > \beta$), and a score of 0 means the algorithm either did not change the previous segmentation or that the algorithm produced a similar result to the previous segmentation (i.e., $\alpha = \beta$). A

previous segmentation	# Filtered Regions	# Clicks	# Scribbles	# LL	# TL	# Rectangles	# Subtractions	# Additions	Mean # Gestures per Region
✗	1,091,467	1,091,467	1,091,467	1,082,165	1,085,329	1,091,467	0	5,411,894	4.98
✓	884,551	1,611,618	1,611,618	1,611,506	1,611,578	1,611,618	4,009,524	4,048,382	9.1

Table 2. Analysis of the interactive segmentation settings of segmentation creation (i.e., create segmentation from scratch) and segmentation refinement (i.e., refine a given segmentation) with respect to the frequency of different gesture types and modes. The larger number of samples for segmentation refinement stems from having multiple errors in previous segmentations needing correction. (LL = loose lasso, TL = tight lasso)

negative score means that the algorithm damaged the previous segmentation (i.e., $\alpha < \beta$). This contrasts previous metrics that only range from 0 to 1 and do not quantify if an algorithm degrades an initial segmentation.

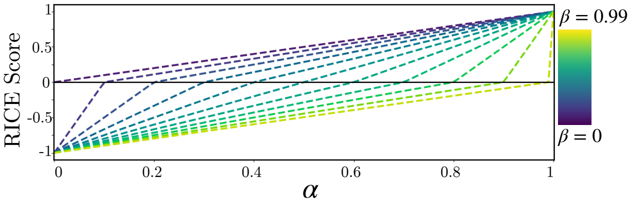

Figure 6. Visualization of the level set of RICE with fixed β values. As β increases so does the slope of RICE.

When examining Figure 6, it is clear that the slope of RICE changes depending on β. Formally, we calculate the rate of change as:

$$\frac{\partial}{\partial \alpha} RICE(\alpha, \beta) = \begin{cases} \frac{1}{1-\beta}, & \text{if } \alpha \geq \beta \\ \frac{1}{\beta}, & \text{else} \end{cases} \tag{4}$$

First, we observe that as β goes to 1, then the slope of RICE trends towards ∞ when $\alpha \geq \beta$. When β goes to 0, the slope of RICE also trends to ∞ when $\alpha < \beta$. Then let, $|\alpha - \beta| = \delta$. When β is large, then a small δ will lead to a higher RICE score than a small δ with a low β.

To support intuitively understanding this metric, we visualize the level set of RICE in two dimensions with multiple fixed β. Results are shown in Figure 6 with differing colors representing RICE at a fixed β. Of note, when the functions cross the x-axis, $\alpha = \beta$ and $RICE(\alpha, \beta) = 0$. As β increases, the slope of RICE increases, meaning a small δ will have a larger effect. To put this concept concretely, let $\beta_1 = 0.20$, $\beta_2 = 0.90$, $\alpha_1 = 0.21$, and $\alpha_2 = 0.91$ such that $|\alpha_1 - \beta_1| = |\alpha_2 - \beta_2| = \delta$. Then, RICE($\alpha_1$, β_1) $= 0.002$ and RICE(α_2, β_2) $= 0.10$. RICE appropriates takes into account that a small change to a relatively good previous segmentation (i.e., β_2) should signify better algorithm performance than a small change to a poor previous segmentation (i.e., β_1). Intuitively, positive changes to a relatively good previous segmentation would be more difficult (i.e., smaller areas to correct) than positive changes to a relatively poor previous segmentation.

8.2. Local Ground Truth for Evaluation

Let $g_r \in \{0, 1\}^{H \times W}$ be the entire ground truth of a given region, $g_a \in \{0, 1, 255\}^{H \times W}$ be the augmented ground truth from Section 9.3, $m \in \{0, 1\}^{H \times W}$ be a previous segmentation, and $g_v \in \{0, 1\}^{H \times W}$ be g_a with all void pixels set to 0 when creating a segmentation. In this formulation, we define void pixels as regions in an image that do not contribute to the loss when training a model or the evaluation when evaluating a model for model selection. Details about which pixels get labeled "void" are provided in Section 9.3. To isolate intended parts of regions for local evaluation, we remove the void pixels for both segmentation creation and segmentation refinement. When refining a segmentation, the void pixels are set to 1 if they are an erroneous region or 0 if they are a missing region. We compute the ground truth used for local evaluation (i.e., RICE$_{\text{local}}$), g_l, as:

$$g_l = (g_r \cdot m) + g_v \tag{5}$$

where $\cdot$ denotes element-wise multiplication. We clip the values of g_l to be in $\{0, 1\}$. We show comparisons of g_l and g_v in Figure 7 for both segmentation creation and segmentation refinement.

9. Expanded Benchmarking Discussion

9.1. Architecture Details for HRNet-dataAug

For this proposed model, we exclude the post-processing module because it was designed to take in multiple clicks to locate an area intended for modification and then generate a cropped area, which could result in partially excluding the gesture and so valuable information; e.g., given that a lasso surrounds the region of interest, a crop may either not include the lasso at all, or contain parts of a region that a user does not wish to change.

9.2. Architecture Details for Proposed Multi-Task Models

To extend *HRNet-base* to predict the intended context of a gesture in addition to the gesture type, we add in two MLPs after the encoder of HRNet [20]. Specifically, after the encoder, we reduce the feature size with a $1x1$ convolution from 96 dimensions to 48 and reduce the spatial size from 128 to 32 with a max pooling operation. We then use two separate 3-layer MLPs to output a binary classification

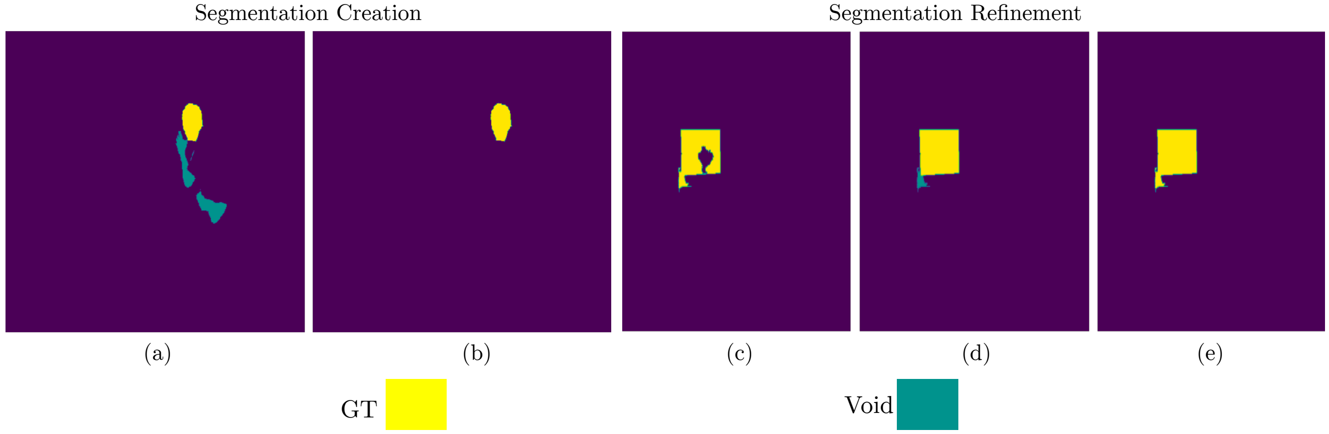

Figure 7. Examples of ground truths used for training (i.e., g_a) in a and d. These ground truths allow for the inclusion of void pixels. Examples of ground truths used for *local* evaluation (i.e., g_l) are shown in b and e, these ground truths remove void pixels for evaluation. We show an example of a previous segmentation in (c).

for the intended context and a 5-class classification for the gesture type. For the intended context, the MLPs go from a dimension of 49,152 to 1024, then 1024 to 512, then 512 to 1 (i.e., add or subtract pixels) with ReLU [6] activations between linear layers. The MLP for the gesture type follows the same architecture as the MLP for context but with an output size of 5 (i.e., one class for each gesture type). We also tested a variant that combines multi-task learning (i.e., *HRNet-multiHead*) with multi-region data augmentation (i.e., *HRNet-dataAug*). We follow the same training procedure as *HRNet-multiHead* while employing the data augmentation described for *HRNet-dataAug*.

9.3. Data Augmentation and Training Details for Proposed Models

We train all variants on our DIG dataset for 15 epochs using the AdamW [16] optimizer with a learning rate of 3e$-$4 and batch size of 32, resize all inputs to 512×512, use a value of 0.5 for non-maximal suppression, and initialize the weights using those publicly available for HRNet [20] pretrained on ImageNet [4]. For data augmentation, as described in Section 8.2, we define void pixels as regions that do not contribute to the loss when training a network and are ignored when evaluating models for model selection. To encourage algorithms to respond locally to user interactions, we make use of void pixels in the ground truth of our regions when applicable. For example, given that we target *parts* of regions when creating a segmentation, we set the remaining connected components within a region as void. We exemplify this in Figure 3(b). Similarly, when performing refinements, we consider a specific correction targeted by an interaction as ground truth (i.e., add or subtract pixels) in addition to any part of the region that is not corrupted. We use a value of 255 to represent void pixels. We show an example of this for an initial segmentation in Figure 4.

9.4. Input Augmentation

Due to existing methods only training with a small number of points (e.g., at most 24 [3]), they are not designed to handle the larger number of points available in some of our gesture annotations. Therefore, when applicable, we reduce the number of points in all annotations using skeletonization [24].

10. Additional Benchmarking Results

10.1. Deep GrabCut and IOG Results

We show results for both algorithms in Table 6. When analyzing *IOG [23]* and *Deep GrabCut [21]*), a plausible explanation for their worse performance is that those techniques used restrictive settings. Deep GrabCut's training process relied solely on single bounding boxes without any provision for refinement, which may have restricted the model's capability to learn general regions. Moreover, this may have limited its suitability for complex real-world settings involving small regions, multiple regions or intricate boundaries. Similarly, IOG's reliance on a unique combination of gestures, specifically bounding box and center click, may not be universally generalizable to each gesture independently, potentially limiting its practical applicability.

10.2. Analysis for Multi-Region Segmentation

For this task, we collect one previous segmentation region and then apply a gesture on a second disconnected region in the image to reference the second, disconnected object we also want to segment. For ground truth, both the disconnected and the target regions of interest are needed to evaluate this set-up of multiple regions. We show results

		Average		Click		Scribble		Loose Lasso		Tight Lasso		Rectangle	
	Method	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**
Multi-Region	*RITM [18] - positive*	-5.81	-5.78	**28.86**	**28.67**	22.57	21.89	-57.40	-56.89	0.82	0.78	-23.91	-23.35
	RITM [18] - negative	-5.70	-5.54	-9.02	-8.72	-9.51	-9.24	-11.23	-11.02	-2.83	-2.75	4.09	4.05
	RITM [18] - random	-5.88	-5.78	9.71	9.75	6.31	6.10	-34.31	-33.93	-1.02	-0.99	-10.08	-9.82
	FocalClick [3] - positive	-16.44	-16.27	23.06	23.12	15.95	15.53	-69.75	-69.26	-16.44	-16.23	-35.05	-34.49
	FocalClick [3] - negative	-15.25	-14.81	-14.58	-14.08	-16.02	-15.51	-17.41	-17.16	-15.98	-15.41	-12.27	-11.91
	FocalClick [3] - random	-16.01	-15.70	4.09	4.39	-0.06	-0.04	-43.60	-43.23	-16.57	-16.18	-23.90	-23.43
	SAM [13]-R - positive	-22.07	-21.84	-12.92	-11.16	-30.05	-30.40	-38.39	-38.15	-14.86	-14.97	-14.12	-14.52
	SAM [13]-R - negative	-37.88	-37.91	-91.99	-91.49	-30.05	-30.40	-38.39	-38.15	-14.86	-14.97	-14.12	-14.52
	SAM [13]-R - random	-29.75	-29.65	-51.33	-50.19	-30.05	-30.40	-38.39	-38.15	-14.86	-14.97	-14.12	-14.52
	SAM [13]-C - positive	-41.14	-39.93	-12.92	-11.16	-43.28	-41.70	-73.89	-72.71	-61.49	-59.56	-14.14	-14.53
	SAM [13]-C - negative	-67.30	-66.90	-91.99	-91.49	-86.74	-86.37	-73.21	-72.43	-70.44	-69.71	-14.14	-14.54
	SAM [13]-C - random	-54.16	-53.35	-51.81	-50.66	-64.96	-63.96	-73.79	-72.82	-66.10	-64.75	-14.14	-14.54
	HRNet-base	-26.25	-46.33	-65.48	-64.95	-60.05	-60.09	-47.85	-47.75	-17.25	-18.17	-40.63	-40.68
	HRNet-dataAug	**28.57**	**27.58**	-2.99	-3.00	**24.26**	**23.36**	**36.91**	**36.08**	**53.34**	**50.92**	**31.31**	**30.52**
	HRNet-multiHead	-54.16	-54.32	-72.32	-72.19	-60.00	-60.01	-57.23	-57.37	-29.61	-30.36	-51.66	-51.68
	HRNet-multiHeadAug	23.25	22.33	5.53	5.24	20.73	19.95	23.6	23.08	54.16	51.55	12.22	11.81

Table 3. Results on the test set of DIG for multi-region segmentation.

for multi-region segmentation in Table 3. Overall, we observe similar trends as single and multi-region segmentation with *HRNet-dataAug* performing the best across the majority of gesture types and single-region methods struggling to support multiple gesture types. We observe that single-gesture methods capable of refinement (i.e., FocalClick [3] and RITM [18]) perform better when using clicks, likely due to the unfair advantage we provided such methods in knowing the context of the interaction (i.e., include versus exclude annotated region). Additionally, we observe that HRNet [20] with multiple classification heads and multi-region data augmentation performs the second best, further showing the efficacy of the proposed data augmentation. Finally, we observe that our other models without data augmentation (i.e., *HRNet-base* and *HRNet-head*) perform the worst, likely due to having neither the context of the interaction nor the multi-region setting included in training.

10.3. IoU for Segmentation Refinement

We show the IoU for each method capable of segmentation refinement (i.e., all methods except IOG [23] and Deep GrabCut [21]) in Table 4. We exclude single region results as RICE simplifies to IOU when no previous segmentation is present. Overall, we observe significantly higher scores for IoU than the corresponding RICE scores in the main paper. This demonstrates why IoU may be misleading as an evaluation metric. For example, an algorithm may receive a high IoU score despite not improving the segmentation but rather because the previous segmentation initially had a high IoU with the region ground truth. On the other hand, RICE provides a more accurate assessment by taking into account if an algorithm improved or damaged a previous segmentation.

10.4. Results for FocalClick with Post-Processing Module

We tested FocalClick [3] with the inclusion of their proposed post-processing module when performing refinements with clicks. Overall, we observe a RICE$_{local}$ score of -72.52 and a RICE$_{global}$ score of -71.13. One plausible explanation for the comparatively lower score may lie in the module's objectives of refining probabilities. Given that our previous segmentations are binary, it is possible that the module cannot fully exploit the probabilistic information, thereby contributing to the observed outcome. For the NoG setting, we reintroduce this module as this setting allows for multiple sequential interactions.

10.5. Results for HRNet Multi-Task Variants

We show results for *HRNet-mutliHead* and *HRNet-multiHeadAug* in Table 6. Overall, we observe a small boost in performance when performing local refinements (i.e., RICE$_{local}$) over *HRNet-dataAug* (2.51 percentage points), but worse performance across all other metrics. We also observe that *HRNet-multiHead* has slightly improved performance over *HRNet-base* for segmentation refinement, illustrating that the additional tasks of gesture and context classification (in *HRNet-multiHead*) play a role in helping the algorithm infer the intention of a user when no context is available. Despite these algorithms achieving top (refinement) or near-top (creation) performance, there still remains room for improvement. This indicates that our proposed dataset challenge presents a challenging, open problem for the research community.

10.6. NoG Evaluation

For backwards compatibility in evaluation, we examine how long an algorithm takes to reach a sufficient quality, if at all. We examine the NoG metric, as described in the main paper, for IoU thresholds of 80, 85, and 90. In line with previous research [3, 18, 23], we also report the number of

		Average		Click		Scribble		Loose Lasso		Tight Lasso		Rectangle	
	Method	**IoU$_{\text{local}}$**	**IoU$_{\text{global}}$**	**IoU$_{\text{local}}$**	**IoU$_{\text{global}}$**	**IoU$_{\text{local}}$**	**IoU$_{\text{global}}$**	**IoU$_{\text{local}}$**	**IoU$_{\text{global}}$**	**IoU$_{\text{local}}$**	**IoU$_{\text{global}}$**	**IoU$_{\text{local}}$**	**IoU$_{\text{global}}$**
Refinement	*RITM [18] - positive*	70.71	67.36	82.62	79.76	81.21	77.96	51.52	48.13	66.04	62.57	72.16	68.40
	RITM [18] - negative	73.32	70.51	83.74	80.55	81.32	78.29	66.46	64.12	68.10	65.39	67.00	64.20
	RITM [18] - random	72.00	68.92	83.16	80.14	81.26	78.12	59.12	56.24	66.97	63.89	69.50	66.24
	FocalClick [3] - positive	62.73	62.68	73.09	73.38	74.18	74.32	40.58	40.23	60.54	60.44	65.24	65.04
	FocalClick [3] - negative	65.50	65.62	77.90	78.20	73.38	73.60	55.77	55.68	61.31	61.41	59.17	59.20
	FocalClick [3] - random	64.08	64.10	75.53	75.83	73.75	73.90	48.03	47.77	60.95	60.95	62.16	62.05
	SAM [13]-R - positive	20.03	20.89	38.49	41.45	6.63	6.69	25.60	26.40	18.87	19.25	10.58	10.69
	SAM [13]-R - negative	13.29	13.59	4.78	4.94	6.63	6.69	25.60	26.40	18.87	19.25	10.58	10.69
	SAM [13]-R - random	16.94	17.54	23.01	24.66	6.63	6.69	25.60	26.40	18.87	19.25	10.58	10.69
	SAM [13]-C - positive	23.89	25.39	41.01	44.08	33.07	35.24	14.79	15.60	20.07	21.41	10.49	10.59
	SAM [13]-C - negative	8.35	8.56	4.78	4.94	6.94	7.24	10.11	10.32	9.45	9.73	10.49	10.59
	SAM [13]-C - random	16.12	16.98	22.92	24.58	19.87	21.12	12.57	13.08	14.72	15.53	10.49	10.59
	HRNet-base	89.03	93.84	89.23	93.59	89.08	93.9	88.97	93.11	89.35	94.54	88.51	94.07
	HRNet-dataAug	**90.41**	94.07	**90.57**	93.91	**90.38**	94.18	**90.57**	**93.59**	**90.59**	94.60	**89.92**	94.06
	HRNet-multiHead	89.58	94.19	89.79	**94.05**	89.74	94.25	89.61	93.55	89.76	94.85	89.01	94.27
	HRNet-multiHeadAug	88.97	**94.47**	89.33	94.42	89.08	**94.56**	88.77	93.52	89.32	**95.24**	88.33	**94.63**

Table 4. IoU results on the test set of DIG for segmentation refinement.

instances where an algorithm fails to achieve the specified IoU within 20 interactions. For every interaction we follow previous work [3,18,22] by always targeting the largest error. We analyze algorithms capable of refinement both settings of starting from segmentation creation and starting from an imperfect superpixel mask supplied by DAVIS585 [3]. For the proposed models, we analyze the performance with respect to each gesture type supplied during training, as well as the setting of starting with a tight lasso and having each subsequent interaction be a click (i.e., mixed). We omit *SAM-R* and *SAM-C* for this experiment and just consider SAM [13] out of the box. We adopt this modification as clicks are the only supported gesture capable of indicating content to not include in the final segmentation.

Results are shown in table 5. As noted in the main paper, algorithms that require context exhibit more failures than our proposed context-free models when refining a previous segmentation, while also requiring more interactions on average to achieve a specified IoU.

Analysis with Respect to Context Augmentation. Augmenting context for existing algorithms yields flipped results for the settings of segmentation creation and refinement. For example, FocalClick [3] and RITM [18] observe the best results (outside of knowing the context) when always assuming the interaction is positive during segmentation creation. Conversely, these methods yield the best results during refinement when assuming the input context is negative. A potential rationale for this observation is that these models may be adept at adding content with minimal 'cruft', leading to better results with positive context. For refinement, 77.44% of DAVIS585 [3] samples contain a false positive, while RITM and FocalClick fail at reaching a sufficient IoU 22.39% to 64.79% of the time when assuming the context is negative, indicating that resolivng the false negatives is sufficient to reach a lower IoU (i.e., 80%) but fails as the desired quality of the segmentation increases. For SAM [13], we observe consistent results with positive context performing the best in both scenarios. This can be partially attributed to SAM exploiting click history as mentioned in the main paper. Across all methods that take in context, we observe that random context exhibits the worst performance in the number of gestures to reach a specific IoU, while also failing the most. This observation may be attributed to the context being selected as the opposite choice of what is desired (i.e., removal when the interaction should be addition) as well as potentially undoing any progress made.

10.7. Qualitative Results

We show qualitative results for segmentation creation when the entire region is the target in Figure 8, segmentation creation when the region *part* is the target in Figure 9, segmentation refinement in Figure 10, and multi-region segmentation in Figure 11.

In the context of creating segmentations where the target region encompasses the entire area (Figure 8), all of the proposed multiple-gesture variants demonstrate an inclination towards selecting an object region when provided with minimal guidance in the form of clicks and scribbles. This tendency may be attributed to the presence of void pixels in the training data, which aids in the development of algorithms that respond to local interactions as opposed to learning to identify the entirety of the region. Conversely, when the same degree of guidance is provided, techniques that employ a single gesture exhibit a proclivity towards selecting the entirety of the region. For the SAM [13] variants, we observe a bias of selecting the entire player for all gesture types aside from scribbles. A potential reason for this is that for consistent evaluation, we always pick the output mask with the highest IoU with the ground truth.

	Creation						**Refinement**					
Method	**NoG@80**	**NoG@85**	**NoG@90**	**NoF@80**	**NoF@85**	**NoF@90**	**NoG@80**	**NoG@85**	**NoG@90**	**NoF@80**	**NoF@85**	**NoF@90**
FocalClick [3] - positive	**10.15**	**11.87**	**14.45**	**273**	**325**	405	9.07	10.76	12.95	242	295	364
FocalClick [3] - negative	-	-	-	585	585	585	7.95	9.75	12.58	209	266	353
FocalClick [3] - random	13.31	14.83	16.67	353	404	469	**7.80**	**8.98**	**11.48**	**198**	**236**	**312**
RITM [18] - positive	**8.00**	**9.77**	**12.89**	**197**	**251**	**352**	6.74	10.26	**13.64**	166	274	379
RITM [18] - negative	-	-	-	585	585	585	**5.52**	**9.39**	13.83	**131**	**241**	**379**
RITM [18] - random	13.62	14.83	16.35	378	416	466	14.90	16.07	17.32	408	451	495
SAM [13] - positive	**1.69**	**1.79**	**1.99**	**88**	**114**	**181**	**1.78**	**1.84**	**2.01**	286	311	350
SAM [13] - negative	-	-	-	585	585	585	-	-	-	585	585	585
SAM [13] - random	5.82	6.31	6.33	129	172	269	5.78	6.06	6.24	**157**	**200**	**290**
HRNet-base - clicks	7.29	7.08	6.66	329	434	526	**1.09**	**1.14**	**1.53**	**45**	**73**	**114**
HRNet-base - scribbles	5.23	6.06	7.23	362	431	511	4.12	4.62	4.92	205	284	395
HRNet-base - loose lassos	1.96	2.41	2.45	225	329	453	3.51	3.89	4.08	362	421	489
HRNet-base- tight lassos	**1.28**	1.30	1.40	**85**	**122**	**195**	4.30	5.05	6.26	130	203	331
HRNet-base - rectangles	2.85	3.43	4.35	457	499	553	2.05	2.15	2.46	407	445	492
HRNet-base - mixed	1.30	**1.23**	**1.30**	108	144	223	8.18	9.70	11.01	272	409	530
HRNet-dataAug - clicks	6.99	7.26	7.40	202	309	456	**1.06**	**1.17**	**1.42**	**36**	**50**	**88**
HRNet-dataAug - scribbles	5.52	6.20	6.69	239	333	454	2.58	3.20	3.89	94	164	266
HRNet-dataAug - loose lassos	2.04	2.38	2.42	182	273	399	2.06	2.44	2.78	119	186	298
HRNet-dataAug - tight lassos	**1.30**	**1.32**	**1.34**	**69**	**109**	**177**	3.36	4.22	5.33	99	178	283
HRNet-dataAug - rectangles	3.80	3.94	4.12	350	420	493	1.72	2.09	2.44	107	175	271
HRNet-dataAug - mixed	1.40	1.44	1.29	88	122	195	2.76	3.87	5.23	72	150	340
HRNet-multiHead - clicks	9.82	10.93	13.16	248	347	491	**1.05**	**1.12**	**1.47**	**38**	**55**	**89**
HRNet-multiHead - scribbles	7.82	8.53	10.03	261	356	463	2.97	3.47	3.90	141	222	333
HRNet-multiHead- loose lassos	3.49	3.89	3.99	251	347	459	3.67	4.07	4.05	237	311	427
HRNet-multiHead - tight lassos	**1.34**	**1.38**	**1.62**	**78**	**110**	**187**	2.87	3.08	3.70	145	243	345
HRNet-multiHead - rectangles	5.44	6.50	8.09	430	490	546	2.60	2.71	2.83	242	318	398
HRNet-multiHead - mixed	1.51	1.69	1.66	89	128	208	10.15	10.84	11.34	224	325	464
HRNet-multiHeadAug - clicks	10.17	11.47	13.82	236	332	481	**1.05**	**1.12**	**1.47**	**38**	**55**	**89**
HRNet-multiHeadAug - scribbles	7.23	8.76	10.64	266	351	459	3.19	3.77	3.96	130	213	328
HRNet-multiHeadAug - loose lassos	3.64	3.83	4.05	255	350	455	3.62	4.13	4.30	239	318	425
HRNet-multiHeadAug - tight lassos	**1.36**	**1.37**	**1.70**	**78**	**110**	**185**	2.92	3.24	4.15	139	248	338
HRNet-multiHeadAug - rectangles	5.64	6.85	8.11	442	486	538	2.82	2.99	3.16	249	305	397
HRNet-multiHeadAug - mixed	1.55	1.71	1.66	93	130	204	10.39	11.10	12.44	216	315	451

Table 5. Algorithmic benchmarking results on DAVIS585 [3]. We report the number of gestures to reach a given IoU as well as the number of failures. Best results for each method are in bold.

	Average		**Click**		**Scribble**		**Loose Lasso**		**Tight Lasso**		**Rectangle**	
Method	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**	**RICE$_{local}$**	**RICE$_{global}$**
GrabCut [21]	10.23	10.60	16.21	16.51	11.66	12.09	6.16	6.48	9.01	9.49	8.13	8.44
IOG [23]	13.68	13.64	23.72	23.12	19.12	19.05	3.59	3.73	12.68	12.89	9.29	9.40
HRNet-multiHead	**63.84**	**61.28**	**55.20**	**52.32**	**57.22**	**54.70**	**66.48**	**64.89**	**81.93**	**78.16**	**58.37**	**56.34**
HRNet-multiHeadAug	58.85	56.46	48.91	46.26	53.87	51.48	60.63	59.19	78.96	75.29	51.86	50.07
HRNet-multiHead	38.55	**45.86**	36.39	**46.68**	37.84	**46.46**	38.20	**46.38**	42.43	**46.53**	37.90	**43.24**
HRNet-multiHeadAug	**41.06**	43.61	**38.90**	44.49	**40.35**	43.36	**39.33**	43.20	**45.54**	44.41	**41.20**	40.38

Table 6. Results on the test set of DIG. Above the dashed line represents segmentation creation, and below represents segmentation refinement.

Regarding the task of segmentation creation where the target region is a *part* of a region, the majority of methods tend to appropriately select the upper half of the surfer, except for Deep GrabCut [21], which includes the majority of the surfer as well as the board in Figure 9. As multiple-gesture methods receive more refined guidance (e.g., lassos, rectangles), they tend to select a larger portion of the target region, thus improving the segmentation accuracy. On the other hand, the performance of single-gesture methods is suboptimal when utilizing gestures other than clicks.

In the context of segmentation refinement, our findings suggest that the multiple-gesture variants exhibit better performance in filling the missing region of the elephant while leaving other corrections relatively untouched. Conversely, single-gesture methods are observed to be ineffective in addressing the region of interest, as evidenced by FocalClick [3] filling in missing content on the ear of the elephant in the second row of Figure 10(b), instead of the intended target region on the body. Moreover, all SAM [13] variants fill in the majority of corrections. We also note that single-gesture methods tend to degrade the segmentation rather than improve it when utilizing gestures other than clicks, as illustrated by the results in the first and second rows of Figure 10(d).

In the realm of multi-region segmentation, our observations suggest that single-gesture methods methods that are provided with contextual information or trained using multi-region augmentation techniques (i.e., *HRNet-dataAug, HRNet-multiHeadAug*) exhibit a superior ability to maintain the previous segmentation, which is a disjoint region, while accurately segmenting a new region of interest. We observe different fail cases for the remaining methods (i.e., *HRNet-*

base, *HRNet-multiHead*, SAM [13] variants). For example, the SAM [13] variants segment only the cat while ignoring the previously segmented region.

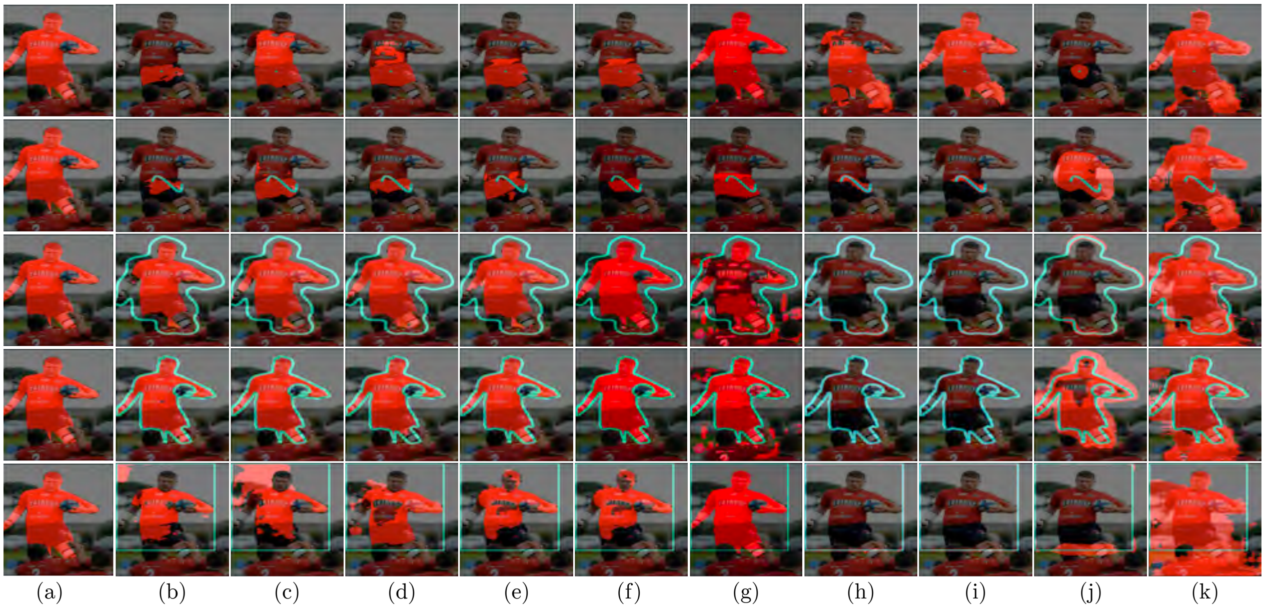

Figure 8. Qualitative results for each gesture type for segmentation creation for all methods when the target is the entire region. From top to bottom: click, scribble, loose lasso, tight lasso, rectangle. (a) input image with region ground truth overlayed, (b) *HRNet-multiHeadAug*, (c) *HRNet-multiHead*, (d) *HRNet-dataAug*, (e) *HRNet-base*, (f) *SAM [13]-R - positive*, (g) *SAM [13]-C - positive*, (h) *FocalClick [3] - positive*, (i) *RITM [18] - positive*, (j) *IOG [23]*, (k) *Deep GrabCut [21]*.

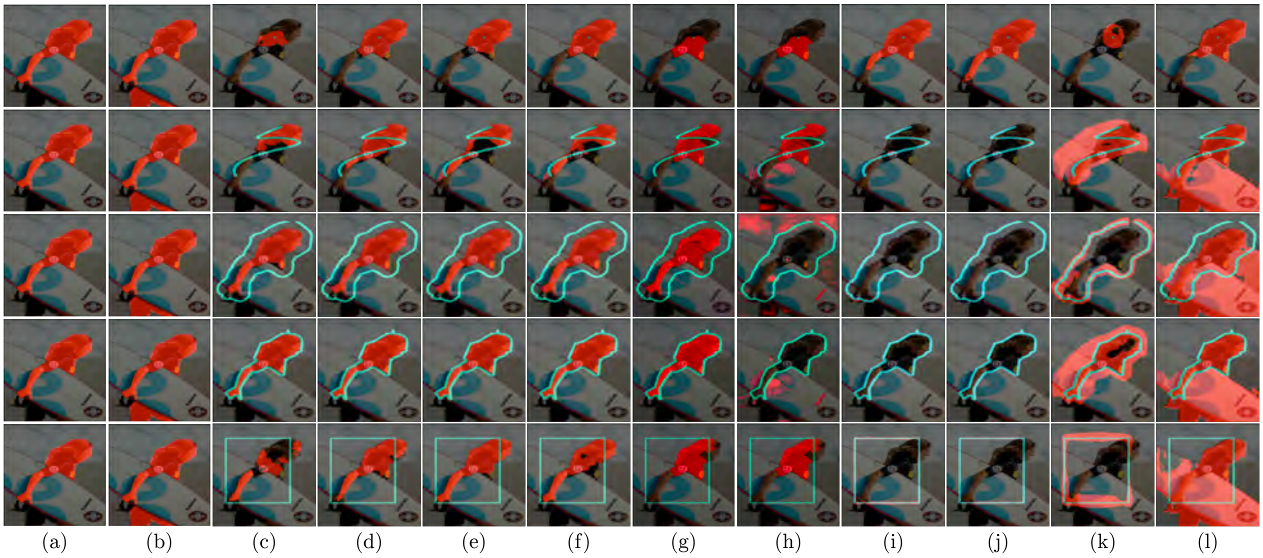

Figure 9. Qualitative results for each gesture type for segmentation creation for all methods when the target is a region *part*. From top to bottom: click, scribble, loose lasso, tight lasso, rectangle. (a) input image with region ground truth overlayed, (b) input with with region part ground truth overlayed (c) *HRNet-multiHeadAug*, (d) *HRNet-multiHead*, (e) *HRNet-dataAug*, (f) *HRNet-base*, (g) *SAM [13]-R - positive*, (h) *SAM [13]-C - positive*, (i) *FocalClick [3] - positive*, (j) *RITM [18] - positive*, (k) *IOG [23]*, (l) *Deep GrabCut [21]*.

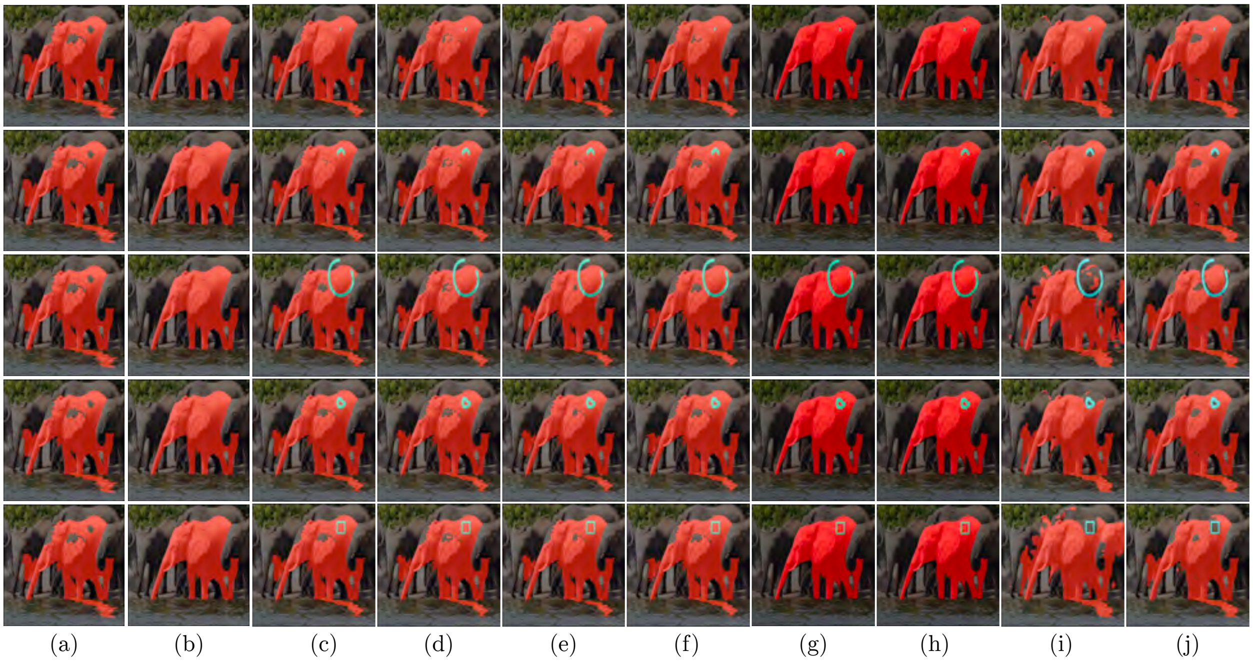

Figure 10. Qualitative results for each gesture type on DIG for segmentation refinement. (a) input image with previous segmentation overlayed, (b) input with with region ground truth overlayed (c) *HRNet-multiHeadAug*, (d) *HRNet-multiHead*, (e) *HRNet-dataAug*, (f) *HRNet-base*, (g) *SAM [13]-R - positive*, (h) *SAM [13]-C - positive*, (i) *FocalClick [3] - positive*, (j) *RITM [18] - positive*.

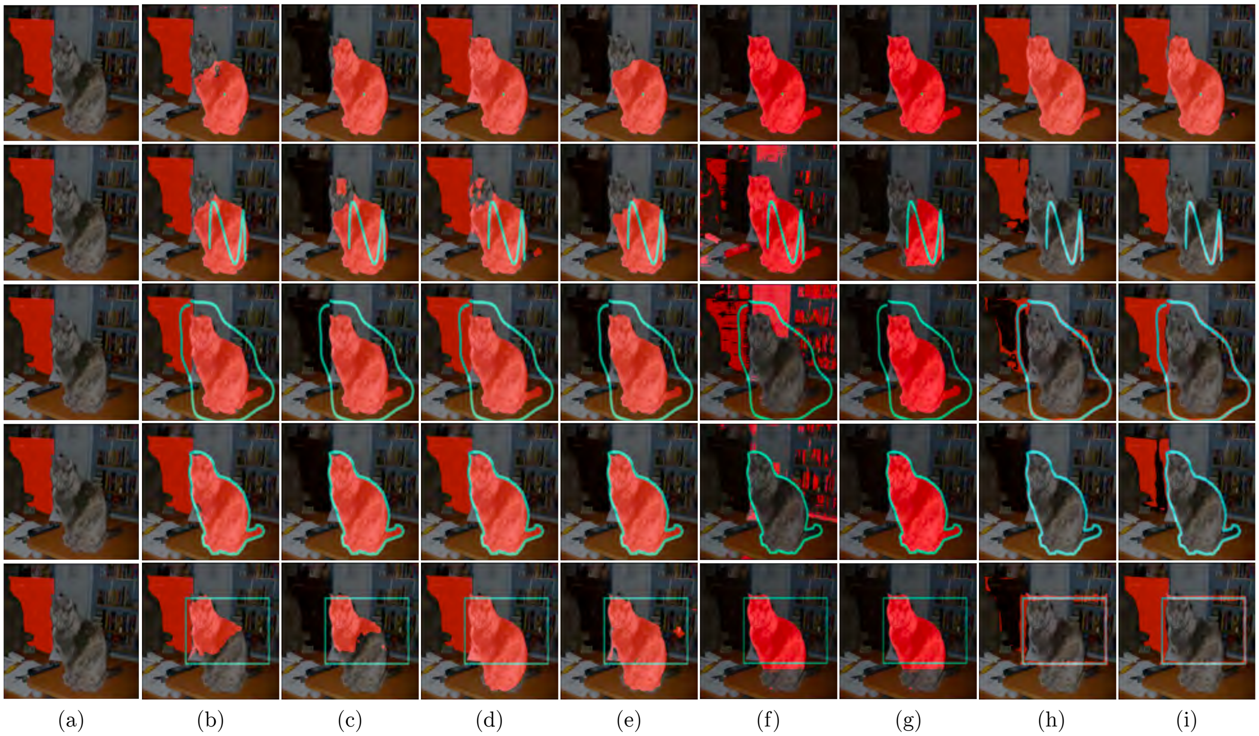

Figure 11. Qualitative results for each gesture type on DIG for multi-region segmentation. (a) input with with previous segmentation overlayed (b) *HRNet-multiHeadAug*, (c) *HRNet-multiHead*, (d) *HRNet-dataAug*, (e) *HRNet-base*, (f) *SAM [13]-R - positive*, (g) *SAM [13]-C - positive*, (h) *FocalClick [3] - positive*, (i) *RITM [18] - positive*, (j) *IOG [23]*, (k) *Deep GrabCut [21]*.

References

[1] Ali Borji, Ming-Ming Cheng, Huaizu Jiang, and Jia Li. Salient object detection: A benchmark. *IEEE transactions on image processing*, 24(12):5706–5722, 2015.

[2] Fredrik Carlsson, Philipp Eisen, Faton Rekathati, and Magnus Sahlgren. Cross-lingual and multilingual clip. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6848–6854, 2022.

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

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[5] Henghui Ding, Scott Cohen, Brian Price, and Xudong Jiang. Phraseclick: toward achieving flexible interactive segmentation by phrase and click. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 417–435. Springer, 2020.

[6] Kunihiko Fukushima. Cognitron: A self-organizing multilayered neural network. *Biological cybernetics*, 20(3):121–136, 1975.

[7] Danna Gurari, Yinan Zhao, Meng Zhang, and Nilavra Bhattacharya. Captioning images taken by people who are blind. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 417–434, Cham, 2020. Springer International Publishing.

[8] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020.

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

[10] Shaun K Kane, Jacob O Wobbrock, and Richard E Ladner. Usable gestures for blind people: understanding preference and performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 413–422, 2011.

[11] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 787–798, 2014.

[12] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9404–9413, 2019.

[13] 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 preprint arXiv:2304.02643*, 2023.

[14] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. 2023.

[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[16] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[17] Timo Lüddecke and Alexander Ecker. Image segmentation using text and image prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7086–7096, 2022.

[18] 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.

[19] Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

[20] 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. *TPAMI*, 2019.

[21] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas Huang. Deep grabcut for object selection. In *28th British Machine Vision Conference, BMVC 2017*. BMVA Press, 2017.

[22] 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.

[23] 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.

[24] Tongjie Y Zhang and Ching Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984.