ByLabel: A Boundary Based Semi-Automatic Image Annotation Tool

Xuebin Qin, Shida He, Zichen Zhang, Masood Dehghan and Martin Jagersand
Department of Computing Science
University of Alberta
{xuebin, shida3, zichen2, masood1, mj7}@ualberta.ca

Abstract

This paper presents a novel boundary based semi-automatic tool, ByLabel, for accurate image annotation. Given an image, ByLabel first detects its edge features and computes high quality boundary fragments. Current labeling tools require the human to accurately click on numerous boundary points. ByLabel simplifies this to just selecting among the boundary fragment proposals that ByLabel automatically generates. To evaluate the performance of ByLabel, 10 volunteers, with no experiences of annotation, labeled both synthetic and real images. Compared to the commonly used tool LabelMe, ByLabel reduces image-clicks and time by 73% and 56% respectively, while improving the accuracy by 73% (from 1.1 pixel average boundary error to 0.3 pixel). The results show that our ByLabel outperforms the state-of-the-art annotation tool in terms of efficiency, accuracy and user experience. The tool is publicly available: `http://webdocs.cs.ualberta.ca/~vis/bylabel/`.

1. Introduction

In image segmentation and visual tracking, well annotated image and video ground truth are essential for performance evaluation and comparison of methods [11]. Labeled images are also used in supervised learning. Although many ground-truth datasets [32, 6, 15, 17, 4, 21, 23, 26] have been published, they are still few compared with the diversity of images and applications of interest in the real world. Fast and accurate image annotation remains an open problem in computer vision and related fields. Image annotation tools seek to maximize labeling accuracy while minimizing human labour and time [31]. Existing annotation tools can be categorized into three main classes: (1) bounding box/quadrilateral based labeling; (2) pixel-wise labeling; (3) boundary based labeling. The annotation tool proposed in this paper belongs to the third class, as shown in Figure 1.

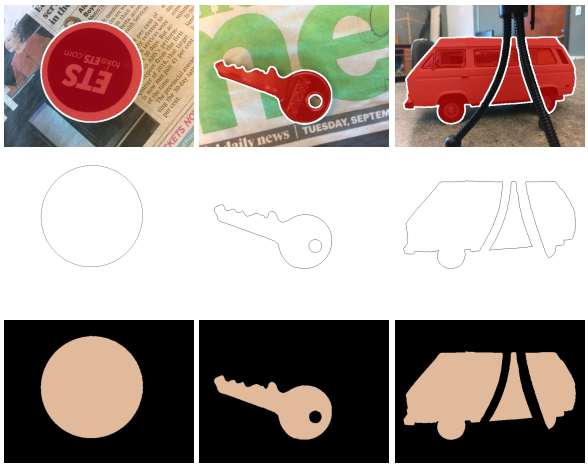
A simple bounding box, defined by two corners (top-left and bottom-right), is usually used in object recognition [6] and two degree of freedom (DoF) tracking [13]. In registration based high DoF tracking, a quadrilateral, which is defined by four corners, describes planar geometric transformations, such as rotation, affine and homography [25, 7, 14, 16]. Bounding boxes labeling is easy to implement. Doermann and Mihalcik developed ViPER, a video annotation tool which allows users to perform annotation frame by frame [5]. Vondrick *et al.* designed a crowdsource video labeling tool, VATIC, which introduces inter frames interpolation to generate bounding boxes semi-automatically [30].

Compared to bounding boxes, pixel-wise labeling provide detailed shape descriptions of target objects [32]. Graph cuts [2], watershed segmentation [12], active contour

(a) input image (b) edge map

(c) region mask (d) region mask overlap

Figure 1. Annotation of a bike. Given an input image, ByLabel computes boundary proposals and the human selects the correct ones. Then, its region mask are generated from these boundaries.



(a) circular box (b) key (c) toy van

Figure 2. Three basic types of objects. Column (a) is a circular box (one closed boundary). Column (b) is a key with a hole (two boundaries). Column (c) is a partially occluded toy van (three boundaries). The top, middle and bottom row are their original images, edge maps and region masks respectively.

[31], partition trees [8] and even mixed techniques [20, 19] have been introduced to decrease the need for user intervention. When to-be-annotated images have salient foregrounds and relatively flat backgrounds, these methods perform well. Otherwise the region masks produced are often inaccurate and noisy.

Closed boundaries are used in many annotation tools, such as KAT [28], PhotoStuff [9], M-Ontomat Annotizer [22] and iVAT[1]. Closed boundaries are usually approximated by polygons. Russell developed a web-based image annotation tool, LabelMe, based on manual polygon drawing [27]. Lluis *et al.* used Recurrent Neural Networks (RNN) to reduce human intervention in polygon annotation [3]. The annotation accuracy depends on the number of the sampled control points and their localization error. More control points are required to describe curved boundaries accurately. Human labelers have to localize each point very accurately [2]. These two factors limits the annotation efficiency. To address these problems, Yang *et al.* proposed a constrained random walk algorithm, which combined unified combinatorial user inputs, to obtain relatively smooth segmentations [33]. Maji *et al.* refined the manually labeled coarse polygons using random maximum a-posteriori perturbations [18]. However, their results usually have similar problems as the aforementioned pixel-wise annotation methods.

To achieve highly accurate annotations while minimizing user interventions, we present an edge fragment based annotation tool called ByLabel. Given an image, edges

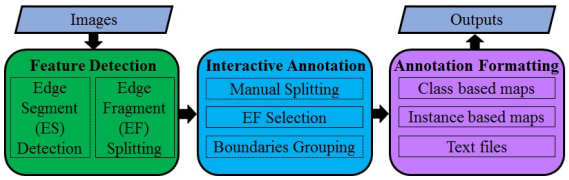


Figure 3. Annotation workflow of ByLabel

are detected and split into high quality fragments. Compared with manually sampled polygon control points, detected edge fragments fit curved boundaries more accurately. Then, users create closed boundaries by selecting subsets of those edge fragments sequentially. The selection operation requires no careful localization of boundaries, hence it greatly reduces the annotation work load. One or multiple boundaries are grouped to describe different types of objects including simple objects with single boundary, objects with holes and objects divided by occlusions. Finally, region masks are generated and outputted based on the corresponding boundary groups.

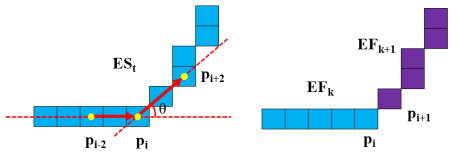
The remainder of this paper is organized as follows. In section 2, we describe the workflow and basic functionalities of our semi-automatic annotation tool, ByLabel. Section 3 shows the experimental results and evaluations. The conclusion is presented in section 4.

2. ByLabel

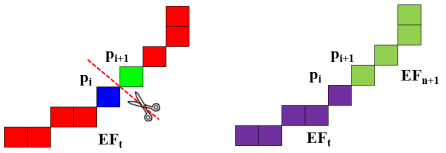
Boundary maps and region masks are the two most commonly used image ground truth types. The goal of ByLabel is to produce these ground truths semi-automatically interactively with a human user. We categorize to-be-annotated objects into three basic types:

- simple objects, which can be defined by one closed boundary or a piece of contiguous region, as shown in Figure 2a.
- objects with holes, which can be described by several nested closed boundaries or a piece of contiguous region, as shown in Figure 2b.
- objects divided by occlusions, which can be determined by multiple closed boundaries or isolated regions, as shown in Figure 2c.

As we can see in Figure 1 and Figure 2, holes and occlusions often exist. Multiple boundaries or regions are necessary to describe these kinds of complicated targets. We take closed boundaries and the regions enclosed by them as "dual" representations of targets, since they can be determined by each other. Hence, we can label either boundaries or regions and generate the other one automatically. ByLabel is designed to label boundaries. Region masks are generated from the labeled boundaries.



(a) splitting by turning angle rule (b) automatic splitting result



(c) manual splitting (d) manual splitting result

Figure 4. Edge segment splitting. (a) and (b) show the automatic splitting of an Edge Segment (ES) based on turning angle. As illustrated in (a), if the turning angle of the pixel $\mathbf{p_i}$ is larger than a threshold (35), the edge segment will be split between pixel $\mathbf{p_i}$ and $\mathbf{p_{i+1}}$. (c) and (d) illustrate the manual splitting of an Edge Fragment (EF). The edge fragment is split between the blue and green pixels

To achieve high annotation accuracy and efficiency, we propose to use automatically detected edge fragments instead of the manually sampled control points to describe object boundaries. Given an image, annotating a new object involves three steps, as illustrated in Figure 3. First, the edge features of the image are detected and split properly. Then, users are allowed to annotate boundaries interactively with well-designed keyboard and mouse interface. Finally, grouped boundaries, generated region masks and inputted class names are organized and formatted in an output fold to represent the objects.

2.1. Feature Detection

Edge fragments (EF) are basic elements in our annotation process. They are obtained by splitting detected edge segments (ES). Given a image, edge segments are extracted by Edge Drawing [29], which is able to produce well-localized, clean, contiguous and one-pixel wide edge segments. Each (chain-wise) edge segment is outputted in vector form as an array of pixels. Compared with manual control points, these pixel chains are smoother and more accurate in fitting object boundaries. However, in the automatically detected fragments some foreground and background edge pixels are often improperly identified as one long and complex edge segment. Our annotation tool provides two ways of splitting an edge segment into multiple well organized edge fragments, as illustrated in Figure 4: (1) automatic splitting using turning angle rules; (2) manual splitting.

Pixels which have large curvatures are more likely to be incorrect connections of foreground and background edges. Here, we search for these pixels using a simple measure, turning angle θ, which is computed as:

$$\theta_{\mathbf{p_i}} = arccos(\frac{(\mathbf{p_i} - \mathbf{p_{i-2}}) \cdot (\mathbf{p_{i+2}} - \mathbf{p_i})}{|\mathbf{p_i} - \mathbf{p_{i-2}}||\mathbf{p_{i+2}} - \mathbf{p_i}|}). \quad (1)$$

where $\mathbf{p_i}$ is the current pixel, $\mathbf{p_{i-2}}$ and $\mathbf{p_{i+2}}$ are two pixels sampled near $\mathbf{p_i}$. $\theta_{\mathbf{p_i}}$ is the angle between vector $\overrightarrow{\mathbf{p_{i-2}p_i}}$ and $\overrightarrow{\mathbf{p_ip_{i+2}}}$, as illustrated in Figure 4a. All of the detected edge segments are split at pixels where turning angles are larger than a certain threshold (set to 35 throughout our experiments), as shown in Figure 4b. Compared with line fitting-based edge splitting methods, this method retains relatively long smooth edge curves, and hence prevents over splitting.

Sometimes, foreground and background edges also have smooth connections, leading to a boundary that cannot be inferred from the turning angle. To handle this case, ByLabel allows manual splitting. Users can split an edge fragment by moving the mouse cursor over the expected splitting position and pressing key "b". As shown in Figures 4c and 4d, the expected splitting position is indicated by two pixels: the blue pixel $\mathbf{p_i}$, which is the closest one to the mouse cursor, and the green pixel $\mathbf{p_{i+1}}$, which is the next one of $\mathbf{p_i}$ in the vector of edge fragment $\mathbf{EF_t}$.

2.2. Interactive Annotation

In ByLabel, one or multiple closed boundaries are employed to describe an object of arbitrary shape. Closed boundaries are defined by sequentially connected multiple edge fragments. In the annotation process, detected edge fragments are superimposed on the original image, and users sequentially select a subset of them to form the object boundaries. If there is a small gap, two successively selected edge fragments are connected with their two closest endpoints by a short straight line segment that is automatically added. Sometimes, not all of the object boundaries can be detected successfully. To address this problem, ByLabel provides a "drawing" mode. Users can switch the labeling mode between "selecting" and "drawing" by press key "a". Similar to LabelMe, the "drawing" mode allows users to manually enter multiple control points to fit those missing arcs. After selecting all necessary fragments, users can click the middle button of the mouse (mouse wheel) to finish the labeling and close the boundary. The simple process of annotation is shown in Figure 5.

An object can consist of multiple boundaries. To group these boundaries into an object entity, users have to label them successively. The annotation processes in ByLabel is sequentially ordered. After completing a closed boundary, there will be a pop-up window asking whether the labeled boundary is the last one of the current object. The input

(a) edge fragments (EF) (b) labeled boundary (c) region mask

Figure 5. Interactive annotation. (a) shows the detected edge fragments. (b) shows the labeled boundary, red edges are selected EF, green edges are generated connections, pink edges are drawn segments. (c) is the region mask generated from the labeled boundary.

of "n" means the object has more boundaries to label. The input of "y" means the current object labeling is finished and there will be another pop-up window for inputting the object's class name or identity.

2.3. Annotation Formatting

There are mainly seven types of annotation outputs stored in their corresponding folders: (1) *color_im_overlap*, (2) *edge_map_classes*, (3) *edge_map_instances*, (4) *region_map_classes*, (5) *region_map_instances*, (6) *text_EF_pixels*, (7) *text_shape_pixels*. The pixel coordinates of detected edge fragments are outputted into *text_EF_pixels*. The direct results of interactive annotation are boundaries represented by multiple edge fragments. These boundaries are all one-pixel-wide. They are written in text files (*text_shape_pixels*) and drawn as boundary maps. In addition to label boundaries, ByLabel is also able to generate region masks according to labeled boundaries. Here, both edge maps and region masks are outputted as two types of color images: class based (*edge_map_classes*, *region_map_classes*) and instance based (*edge_map_instances*, *edge_map_instances*). Users are free to choose any type of output depending on the application. More details and instructions can be found: `https://github.com/NathanUA/ByLabel`.

3. Experimental Results

To demonstrate the advantages of our boundary based annotation tool ByLabel, we conduct a set of user tests to quantitatively compare its performance with that of a popular web-based annotation tool LabelMe. Additionally, we show the results of some typical annotation cases to demonstrate the effectiveness of ByLabel qualitatively.

3.1. User Tests and Evaluation

In our user experiments, ten volunteers are asked to annotate five synthetic images (Figure 6), and ten real images (Figure 7 and Figure 8), as fast and accurate as possible using both LabelMe and ByLabel. These ten volunteers have no prior experiences on image annotation. To reduce the total work load of the annotation test, each testing image contains only one target object, which is defined by a single boundary.

We evaluate ByLabel and compare it with LabelMe on the following four aspects: *clicks*, *time costs*, *error* and *user experience*. Mouse clicks are the most commonly used operation in image annotation. Generally, more *clicks* produce more detailed annotation results and higher geometric accuracy. However, a large number of clicks require more patience and time. Therefore, *clicks* and *time costs* reflect the annotation work load directly. The geometric annotation *error* is defined by the average Alignment Error ($aveAE$) [24]

$$aveAE = \frac{B_{usr} \otimes Dist_{gt}}{P_{usr}} \tag{2}$$

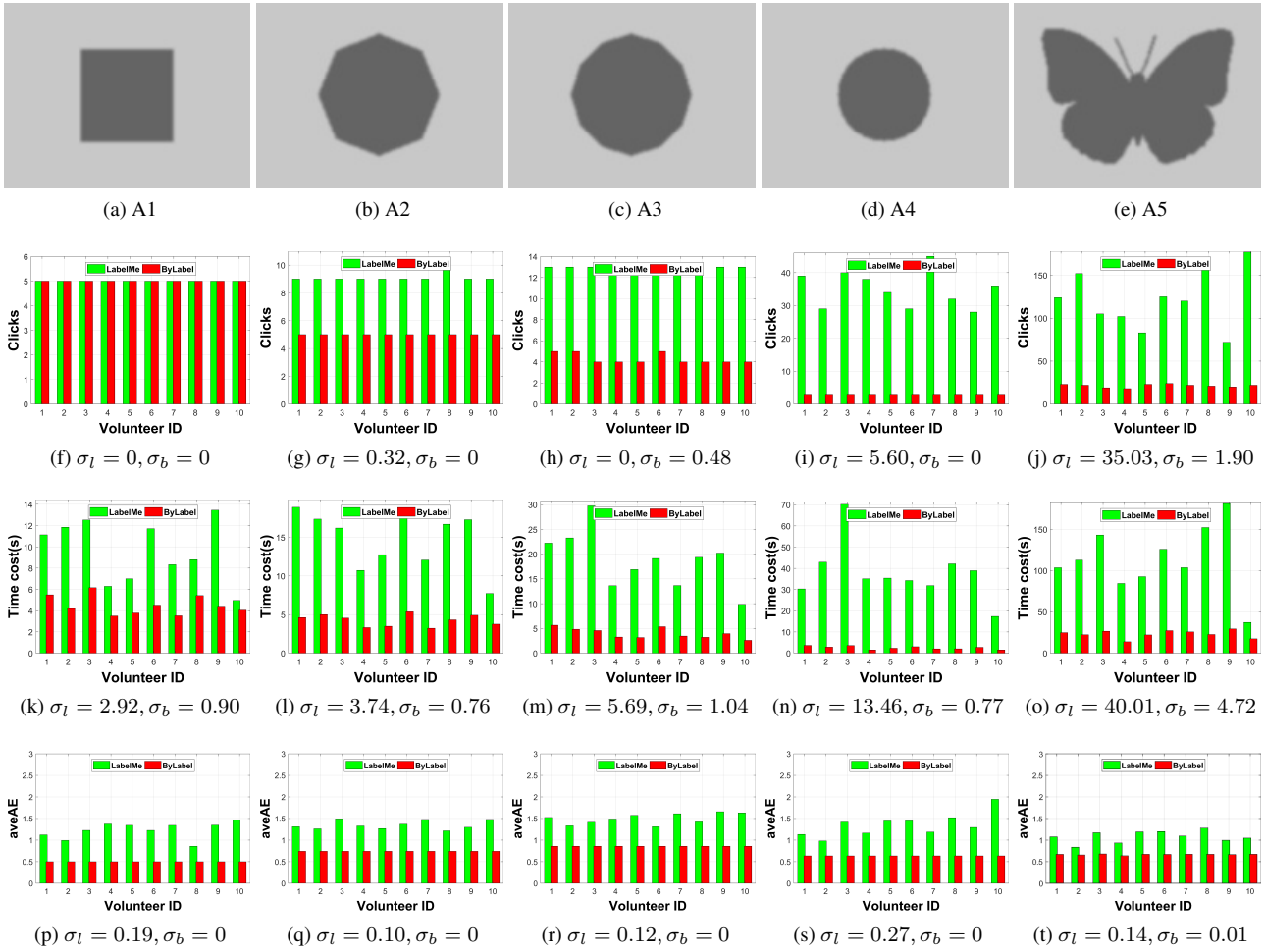
where $\otimes$ indicates the summation of element(pixel)-wise multiplication, B_{usr} is the annotated binary boundary map, $Dist_{gt}$ is the distance map of the ground truth, P_{usr} is the edge pixels' number of B_{usr}.

In our tests, we recorded users' *clicks*, *time costs* and *errors* on each testing image, as shown in Figure 6, Figure 7 and Figure 8. After annotating, they were also asked to fill the forms of NASA Task Load Index (TLX) [10] for *user experience* evaluation (see the results in Figure 11).

1) Tests on synthetic images

Image annotation are usually simplified as polygon drawing. As mentioned above, there are two factors that affect the annotation work load. One is the number of boundary control points, which determines the required *clicks* along the boundary. The other one is the boundary saliency. Here, the boundary saliency denotes the intensity change rate (image gradient) along the perpendicular direction of local boundary. It affects the difficulty of key points localization and therefore their accuracy.

To study the influences of the two factors on image annotation, we first conduct our user tests on five synthetic images of increasing complexity. These five images are square, octagon, dodecagon, circle and a shape of butterfly respectively (see the first row of Figure 6). They are generated by blurring corresponding binary shape maps using a Gaussian filter (kernel size = 5, $\sigma = 1$). The Gaussian filter blurs the binary map to simulate the appearance of real image boundaries. The binary shape maps are retained as ground truth of synthetic images.



(a) A1 (b) A2 (c) A3 (d) A4 (e) A5

(f) $\sigma_l = 0, \sigma_b = 0$ (g) $\sigma_l = 0.32, \sigma_b = 0$ (h) $\sigma_l = 0, \sigma_b = 0.48$ (i) $\sigma_l = 5.60, \sigma_b = 0$ (j) $\sigma_l = 35.03, \sigma_b = 1.90$

(k) $\sigma_l = 2.92, \sigma_b = 0.90$ (l) $\sigma_l = 3.74, \sigma_b = 0.76$ (m) $\sigma_l = 5.69, \sigma_b = 1.04$ (n) $\sigma_l = 13.46, \sigma_b = 0.77$ (o) $\sigma_l = 40.01, \sigma_b = 4.72$

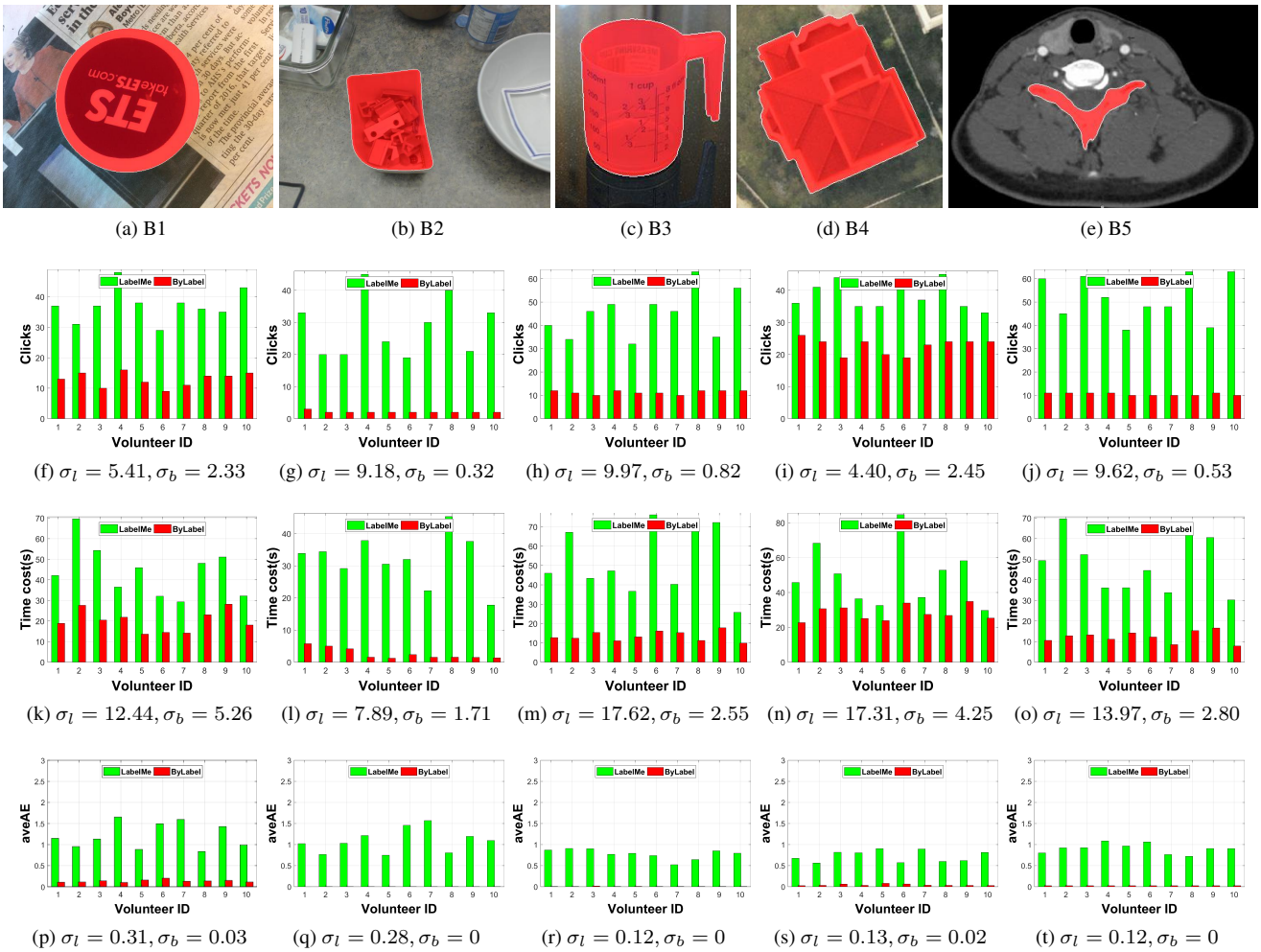
(p) $\sigma_l = 0.19, \sigma_b = 0$ (q) $\sigma_l = 0.10, \sigma_b = 0$ (r) $\sigma_l = 0.12, \sigma_b = 0$ (s) $\sigma_l = 0.27, \sigma_b = 0$ (t) $\sigma_l = 0.14, \sigma_b = 0.01$

Figure 6. The results of user tests on synthetic images. The top row shows the synthetic images. The second to fourth row show the *clicks*, *time costs* and *average Alignment Error (aveAE)* respectively. σ_l and σ_b are standard deviations of LabelMe and ByLabel.

In LabelMe, to annotate regular polygons whose boundaries are straight line segments as shown in Figures 6a, 6b and 6c, users just have to click through the corners sequentially. But as the number of corners goes up, more *clicks* and *time costs* are required (see Figures 6i and 6n). The extreme case of the regular polygon is a circle, as shown in Figure 6d. Theoretically, there are infinitely many corners along its boundaries. The users would need to click many times to obtain an accurate annotation. Compared with LabelMe, ByLabel annotates targets simply by selecting the detected edge fragments, which is more efficient for smooth boundary annotation. As can be seen in Figures 6f - 6i, using ByLabel reduces *clicks* greatly and therefore saves the annotation time (see Figures 6k - 6n). Although ByLabel requires the same number of clicks as LabelMe in annotating A1, it costs less time because selecting detected edge fragments is easier than localizing the exact control points. The butterfly shape in Figure 6e is comprised of many corners and smooth arcs. It is designed as a comprehensive annotation test. The results show that ByLabel achieves significant improvement in terms of *clicks* and *time cost*, see Figure 6j, Figure 6o and Table 1, 2.

Overall, as the shape complexity increases, more *clicks* and *time costs* are required to obtain relatively detailed annotations using LabelMe. However, with ByLabel, the number of clicks and the time costs stayed at a low level, as shown in the second and third row of Figure 6,

As shown in Figures 6p - 6t, users produce smaller errors (0.5 - 1 pixel) when using ByLabel than using LabelMe (over 1 pixel). Besides, different users achieve almost the same error on each testing image when using ByLabel which suggests that ByLabel is able to reduce the annotation uncertainties. The quantitative evaluation of the uncertainties is the standard deviation σ.



(a) B1 (b) B2 (c) B3 (d) B4 (e) B5

(f) $\sigma_l = 5.41, \sigma_b = 2.33$ (g) $\sigma_l = 9.18, \sigma_b = 0.32$ (h) $\sigma_l = 9.97, \sigma_b = 0.82$ (i) $\sigma_l = 4.40, \sigma_b = 2.45$ (j) $\sigma_l = 9.62, \sigma_b = 0.53$

(k) $\sigma_l = 12.44, \sigma_b = 5.26$ (l) $\sigma_l = 7.89, \sigma_b = 1.71$ (m) $\sigma_l = 17.62, \sigma_b = 2.55$ (n) $\sigma_l = 17.31, \sigma_b = 4.25$ (o) $\sigma_l = 13.97, \sigma_b = 2.80$

(p) $\sigma_l = 0.31, \sigma_b = 0.03$ (q) $\sigma_l = 0.28, \sigma_b = 0$ (r) $\sigma_l = 0.12, \sigma_b = 0$ (s) $\sigma_l = 0.13, \sigma_b = 0.02$ (t) $\sigma_l = 0.12, \sigma_b = 0$

Figure 7. The results of user tests on the first group of real images. The top row shows the test images. The highlighted red regions are to-be-annotated targets. The second to fourth rows show the *clicks*, *time costs* and *average Alignment Error (aveAE)* respectively. σ_l and σ_b are standard deviations of LabelMe and ByLabel.

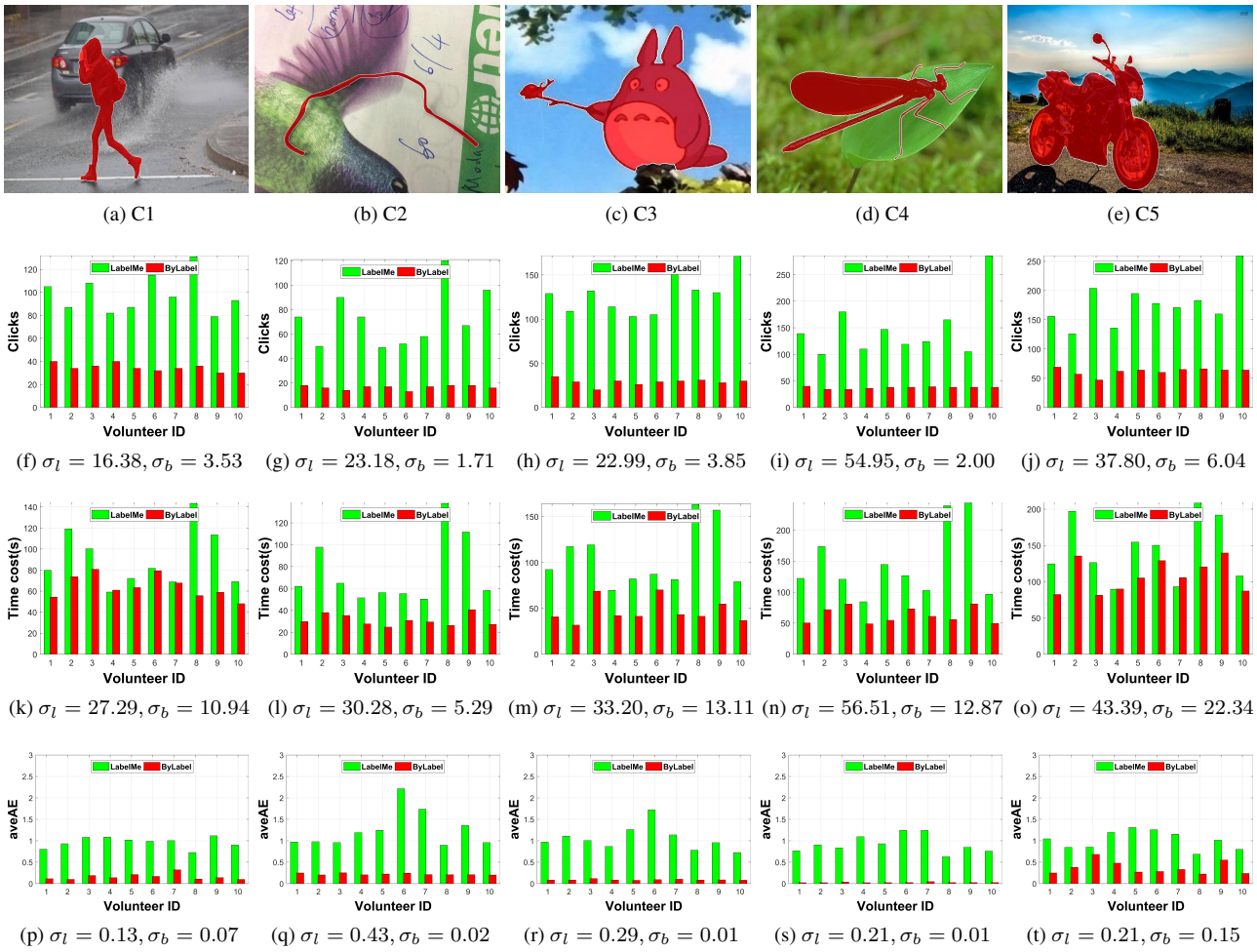
2) Tests on real images

The 10 real images including nature images, satellite image, medical image, and manga are selected for user tests. These images are divided into two groups according to their targets' shape complexity. The targets in the first group are relatively simple, as shown in Figures 7a - 7e. Those in the second group are more complicated and challenging, see Figures 8a - 8e.

B1, B2, B4 and B5 are circular box, rounded rectangular container, measure cup and MRI image of human neck respectively. Their boundaries are all smooth curves. Figures 7f, 7g, 7h, 7j and Figures 7k, 7l, 7m, 7o show that ByLabel reduces both *clicks* and *time costs* on this kind of targets. B3 (see Figure 7d) is an aerial image with a building roof whose boundary is comprised of multiple straight edges and sharp corners. Our algorithm split its boundary into many edge fragments at its sharp corners. As a result, the annotation *clicks* and *time costs* of ByLabel and LabelMe are similar. But using ByLabel achieves almost zero errors while the errors when using LabelMe are all close to 1 pixel, as shown in Figures 7p - 7t.

Targets in Figures 8a - 8e are pedestrain, cable, manga totoro, insect and motorcycle respectively. Compared with the targets in the first group, this five images have finer structures, which are challenging to annotate. As shown in Figure 8f - Figure 8j, more than 100 clicks are required to annotate each of these targets by LabelMe. ByLabel reduces *clicks* to close to or fewer than 50. Figures 8k - 8o illustrate the advantage of ByLabel in terms of *time costs*.

As can be seen in Figure 8k and Figure 8o, some volunteers spend the same or even more time in annotating object C1 and C5 when using ByLabel compared to LabelMe. The



(a) C1 (b) C2 (c) C3 (d) C4 (e) C5

(f) $\sigma_l = 16.38, \sigma_b = 3.53$ (g) $\sigma_l = 23.18, \sigma_b = 1.71$ (h) $\sigma_l = 22.99, \sigma_b = 3.85$ (i) $\sigma_l = 54.95, \sigma_b = 2.00$ (j) $\sigma_l = 37.80, \sigma_b = 6.04$

(k) $\sigma_l = 27.29, \sigma_b = 10.94$ (l) $\sigma_l = 30.28, \sigma_b = 5.29$ (m) $\sigma_l = 33.20, \sigma_b = 13.11$ (n) $\sigma_l = 56.51, \sigma_b = 12.87$ (o) $\sigma_l = 43.39, \sigma_b = 22.34$

(p) $\sigma_l = 0.13, \sigma_b = 0.07$ (q) $\sigma_l = 0.43, \sigma_b = 0.02$ (r) $\sigma_l = 0.29, \sigma_b = 0.01$ (s) $\sigma_l = 0.21, \sigma_b = 0.01$ (t) $\sigma_l = 0.21, \sigma_b = 0.15$

Figure 8. The results of user tests on the second group of real images.

Table 1. Average Clicks

Image	A1	A2	A3	A4	A5	B1	B2	B3	B4	B5	C1	C2	C3	C4	C5	Average
LabelMe	5	9	13	35	123	37	28	45	38	51	98	73	129	147	177	**67**
Ours	5	5	4	3	21	13	2	11	23	11	35	16	29	37	62	**18**

Table 2. Average Time Costs (s)

Image	A1	A2	A3	A4	A5	B1	B2	B3	B4	B5	C1	C2	C3	C4	C5	Average
LabelMe	9.61	14.80	18.80	37.85	113.71	44.13	32.08	52.63	49.62	47.80	90.71	74.56	104.79	145.52	144.39	**65.04**
Ours	4.51	4.26	4.01	2.56	23.25	20.03	2.60	13.49	28.19	12.26	64.30	31.02	47.07	62.76	107.62	**28.53**

Table 3. Average Alignment Error (pixel)

Image	A1	A2	A3	A4	A5	B1	B2	B3	B4	B5	C1	C2	C3	C4	C5	Average
LabelMe	1.23	1.35	1.50	1.35	1.08	1.21	1.09	0.78	0.73	0.91	0.97	1.25	1.05	0.92	1.02	**1.10**
Ours	0.50	0.74	0.86	0.64	0.67	0.14	0.00	0.01	0.04	0.02	0.16	0.22	0.09	0.03	0.37	**0.30**

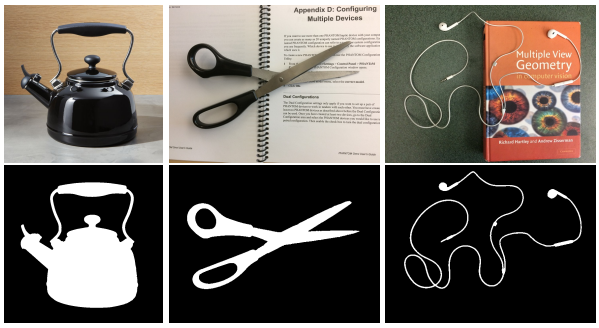
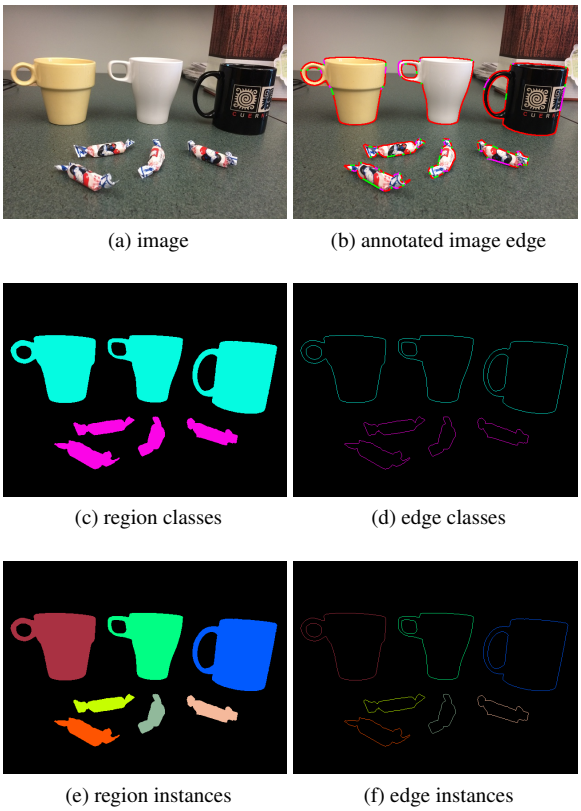


Figure 9. Annotation of objects with multiple boundaries.



(a) image (b) annotated image edge

(c) region classes (d) edge classes

(e) region instances (f) edge instances

Figure 10. Classes and instances.

reason is that some part of the boundary in C1 and C5 are missing. Volunteers have to draw the missing parts using "drawing" mode of ByLabel. A lot of the time are spent in figuring out how to switch between "selecting" and "drawing" mode due to their limited experience with the tool. This can be reduced for experienced users.

3) Overall Evaluation

We summarize the average *clicks*, *time costs* and *errors* of 10 volunteers on each testing image in Table 1 and Table 2. Our ByLabel reduces 73% of *clicks* (from 67 to 18), meanwhile saves 56% *time costs* (from 65.04 s to 28.53 s). Table 3 shows that ByLabel achieves an overall average Alignment Error (ave_AE) of 0.30. Compared with that (1.10 pixels) of using LabelMe, the annotation error is decreased by 73%.

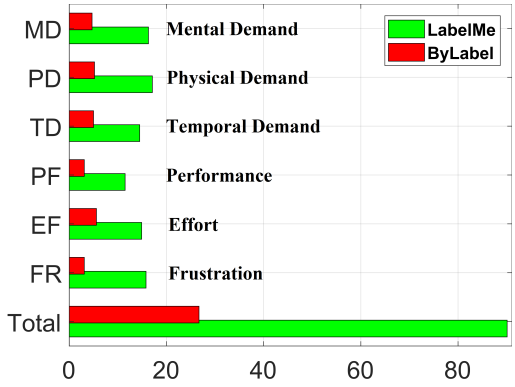


Figure 11. The average value of NASA Task Load Index (TLX) produced by those 10 volunteers.

Figure 11 shows the NASA Task Load Index results generated by these 10 volunteers. Lower scores denote more friendly *user experience*. As shown in Figure 11, ByLabel achieves lower scores than LabelMe in all six aspects.

3.2. More Annotation Examples

In section 3.1, the testing targets are all defined by single boundaries. To further illustrate the capability of ByLabel, Figure 9 shows annotations of some commonly used objects which are defined by multiple boundaries.

Additionally, ByLabel is able to output both class based and instance based annotations. Figure 10 depicts a scence with three cups and four candies. Figure 10c and Figure 10d show the class based annotations. Objects belonging to the same class are encoded in the same color. Figure 10e and Figure 10f show the instance based annotations. Each object instance has a unique color. The color codes and their corresponding classes and instances are outputted into accompanying text files.

4. Conclusions

In this paper, we develop a novel semi-automatic boundary based image annotation tool, ByLabel. Instead of annotating images directly, ByLabel introduces edge detection and splitting algorithms to assist annotation, which greatly improves the annotation efficiency and accuracy. The results of user tests show that ByLabel outperforms the state-of-the-art annotation tool LabelMe in terms of *time costs*, *accuracy* and *user experience*. Additionally, ByLabel can also be used to annotate video streams frame by frame.

References

[1] S. Bianco, G. Ciocca, P. Napoletano, and R. Schettini. An interactive tool for manual, semi-automatic and automatic video annotation. *Computer Vision and Image Understanding*, 131:88–99, 2015.

[2] Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*, pages 105–112, 2001.

[3] L. Castrejon, K. Kundu, R. Urtasun, and S. Fidler. Annotating object instances with a polygon-rnn. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4485–4493, 2017.

[4] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] D. S. Doermann and D. Mihalcik. Tools and techniques for video performance evaluation. In *15th International Conference on Pattern Recognition, ICPR'00, Barcelona, Spain, September 3-8, 2000.*, pages 4167–4170, 2000.

[6] M. Everingham, S. M. A. Eslami, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[7] S. Gauglitz, T. Höllerer, and M. Turk. Evaluation of interest point detectors and feature descriptors for visual tracking. *International Journal of Computer Vision*, 94(3):335–360, 2011.

[8] X. Giró i Nieto, N. Camps, and F. Marqués. GAT: a graphical annotation tool for semantic regions. *Multimedia Tools Appl.*, 46(2-3):155–174, 2010.

[9] C. Halaschek-Wiener, J. Golbeck, A. Schain, M. Grove, B. Parsia, and J. A. Hendler. Annotation and provenance tracking in semantic web photo libraries. In *Provenance and Annotation of Data, International Provenance and Annotation Workshop, IPAW 2006, Chicago, IL, USA, May 3-5, 2006, Revised Selected Papers*, pages 82–89, 2006.

[10] S. G. Hart. Nasa task load index (tlx). volume 1.0; paper and pencil package. 1986.

[11] F. D. Julca-Aguilar and N. S. T. Hirata. Expressmatch: A system for creating ground-truthed datasets of online mathematical expressions. In *10th IAPR International Workshop on Document Analysis Systems, DAS 2012, Gold Coast, Queenslands, Australia, March 27-29, 2012*, pages 155–159, 2012.

[12] B. Klava and N. S. T. Hirata. A model for simulating user interaction in hierarchical segmentation. In *2014 IEEE International Conference on Image Processing, ICIP 2014, Paris, France, October 27-30, 2014*, pages 4358–4362, 2014.

[13] M. Kristan, A. Leonardis, and J. Matas. The visual object tracking VOT2016 challenge results. In *Computer Vision - ECCV 2016 Workshops - Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II*, pages 777–823, 2016.

[14] J. Kwon, H. S. Lee, F. C. Park, and K. M. Lee. A geometric particle filter for template-based visual tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(4):625–643, 2014.

[15] F. Li, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[16] P. Liang, Y. Wu, and H. Ling. Planar object tracking in the wild: A benchmark. *CoRR*, abs/1703.07938, 2017.

[17] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, pages 740–755, 2014.

[18] S. Maji, T. Hazan, and T. S. Jaakkola. Active boundary annotation using random MAP perturbations. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, pages 604–613, 2014.

[19] B. Marcotegui, P. L. Correia, F. Marqués, R. Mech, R. Rosa, M. Wollborn, and F. Zanoguera. A video object generation tool allowing friendly user interaction. In *Proceedings of the 1999 International Conference on Image Processing, ICIP '99, Kobe, Japan, October 24-28, 1999*, pages 391–395, 1999.

[20] K. McGuinness and N. E. O'Connor. Toward automated evaluation of interactive segmentation. *Computer Vision and Image Understanding*, 115(6):868–884, 2011.

[21] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016.

[22] K. Petridis, D. Anastasopoulos, C. Saathoff, N. Timmermann, Y. Kompatsiaris, and S. Staab. M-ontomat-annotizer: Image annotation linking ontologies and multimedia low-level features. In *Knowledge-Based Intelligent Information and Engineering Systems, 10th International Conference, KES 2006, Bournemouth, UK, October 9-11, 2006, Proceedings, Part III*, pages 633–640, 2006.

[23] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017.

[24] X. Qin, S. He, Z. Zhang, M. Dehghan, and M. Jagersand. Real-time salient closed boundary tracking using perceptual grouping and shape priors. *28th British Machine Vision Conference, BMVC, London, UK, September 4-7*, 2017.

[25] A. Roy, X. Zhang, N. Wolleb, C. Perez, Quenterio, and M. Jagersand. Tracking benchmark and evaluation for manipulation tasks. In *International Conference on Robotics and Automation*. IEEE, 2015.

[26] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[27] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image

annotation. *International Journal of Computer Vision*, 77(1-3):157–173, 2008.

[28] C. Saathoff, S. Schenk, and A. Scherp. Kat: the k-space annotation tool. 2008.

[29] C. Topal and C. Akinlar. Edge drawing: A combined real-time edge and segment detector. *J. Visual Communication and Image Representation*, 23(6):862–872, 2012.

[30] C. Vondrick, D. J. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation - A set of best practices for high quality, economical video labeling. *International Journal of Computer Vision*, 101(1):184–204, 2013.

[31] T. Wang, B. Han, and J. P. Collomosse. Touchcut: Fast image and video segmentation using single-touch interaction. *Computer Vision and Image Understanding*, 120:14–30, 2014.

[32] J. M. Winn, A. Criminisi, and T. P. Minka. Object categorization by learned universal visual dictionary. In *10th IEEE International Conference on Computer Vision (ICCV 2005), 17-20 October 2005, Beijing, China*, pages 1800–1807, 2005.

[33] W. Yang, J. Cai, J. Zheng, and J. Luo. User-friendly interactive image segmentation through unified combinatorial user inputs. *IEEE Trans. Image Processing*, 19(9):2470–2479, 2010.