BUBL: An Effective Region Labeling Tool Using a Hexagonal Lattice

Carolina Galleguillos Peter Faymonville* Serge Belongie
Department of Computer Science and Engineering
University of California, San Diego
{cgallegu,pfaymonville,sjb}@cs.ucsd.edu

Abstract

We propose a data labeling tool that permits accurate labeling of images using less time and effort. Our tool, BUBL, uses a hexagonal grid with a variable size tiling for accurate labeling of object contours. The hexagonal lattice is superimposed by a bubble wrap interface in order to make the labeling task enjoyable. The resulting label mask is represented by a Gaussian kernel density estimator which provides accurate bounding contours, even for objects that include hollow regions. Furthermore, multiple annotations from different users are collected for every image, making it possible to "hint" a partial labeling so the user can finish labeling in less time. We show accuracy results by simulating the application of our labeling tool for the MSRC dataset and to a subset data set of Caltech-101.

1. Introduction

Image databases are an important part of object recognition research, as they are required for learning object models and testing recognition algorithm performance. Current databases present images from real world scenes containing objects that may vary in scale, position, and viewpoint; in addition, they may be surrounded by background clutter, occluded by other objects, and obscured by poor image quality. In order to model these sources of variability, many approaches to object recognition require large labeled data sets of fully annotated images. Typical annotations in these data sets provide masks or bounding boxes that specify the locations, scales, and orientations of objects in each image.

Today the web is offering us an immensely large amount of visual data. Thanks to the popularity of digital photography, images are captured at every moment and they are made publicly available by users that want to share their pictures in web albums. Therefore gathering images for new image databases has become an easier task, however obtaining accurate labels is still a problem. Currently there exist several annotation tools that can help to label a collection of images. They can gather different types of information, that range from captions [11] to object outlines [6, 10]. Some of them even make the annotation process completely public [6] and also entertaining [11, 12] (as shown in Table 1).

(a) (b) (c) (d)

Figure 1. Comparison of labels between LabelMe (b) tool and BUBL (d). Figures (a) and (c) correspond to interfaces of LabelMe and BUBL respectively. Popped bubbles in (d) are shown in green.

LabelMe [6] is a database and an online annotation tool. The tool provides functionalities such as drawing polygons, querying images, and browsing the database. Although LabelMe provides object contours for a large amount of real world images, the labels tend to be less accurate as labeling is done using polygon drawing which requires a great deal of effort. Users need to upload the images to the website and then hand label the objects with the correct contours. This labeling tends to be tedious as concentration and good skill are needed.

Another way of labeling large amounts of data is through online games. The ESP game [11] collects image captions by pairing two random online users who view the same target image. The goal is to try to "read each other's mind" by agreeing on an appropriate name for a given image as quickly as possible. The ESP game has collected over 10 million image captions since 2003, however information about object's location and pounding contours are not acquired. Another online game, Peekaboom [12], provides approximate location information of objects and parts. In this game two random players participate by taking different roles in the game: one reveals parts of the image to the

*Visiting Student from University of Potsdam, Germany

	Annotation Type	Effort Level	Boundary Accuracy	Hollow Objects	Label Access	# Labels per Image	Goal/ Incentive
LabelMe	object contours	high	medium	no	public	many	labels
ESP Game	image caption	low	-	no	private	many	fun
Peekaboom	object parts/area	medium	medium	-	private	many	fun
SQ-PIX	object contour	high	low	no	private	one	solve CAPTCHA
Mechanical Turk [7]	polygon/ super-pixel	high/ medium	high/ medium	no	private	one	money
BUBL	object mask /contours	low	high	yes	public/ private	one	obtain labels/ fun

Table 1. Comparison between different labeling tools. We can see that BUBL has many advantages with respect to other tools.

other user so that the other user can guess the associated word. While location information is provided for a large number of images, often only small discriminant regions are labeled and not entire object outlines.

SQ-PIX [10] is an image-based CAPTCHA that also serves as an online image labeling tool. The tool requires users first to pick out the right image from three and then to trace the outline of the object within the image. Even though this tool serves two purposes – obtain object labels and serve as a CAPTCHA – users require a degree of manual skill for labeling objects. Therefore object labels are not accurate as the goal of the user is to pass the CAPTCHA as quickly as possible.

Outsourced labeling is another option for obtaining accurate object labels in images. The work by [7] outsourced the annotation work to the online worker community of Amazon Mechanical Turk. Advantages of this idea include more accurate object contours as there are multiple annotations collected for every image and more reliable annotations as users get a monetary compensation for each labeling. Also the tool obtains super-pixels in addition to polygon labels. When using this method we need to consider the learning curve to get familiar with Amazon Mechanical Turk, some time delays as labeling is done without time commitment, and the availability of sufficient funds to pay users for their service.

In this paper we propose a data labeling tool that permits accurate labeling of images using less time and effort than other available tools. Our tool, BUBL, uses a hexagonal grid with a variable size tiling for an accurate labeling of object contours. Over the hexagonal lattice we place a bubble wrap interface that makes the labeling task enjoyable. The final labeling is computed using kernel density estimation, which results in accurate object contours. Moreover BUBL can support objects that include hollow regions (as shown in Figure 1). BUBL uses multiple annotations from different users for each image, making it possible to "hint" a partial labeling so that a user can finish labeling in less time. We show accuracy results of our labeling tool for the MSRC database and for a subset of Caltech-101 [1] which presents some of the most challenging geometric object classes [8].

2. Our Labeling Model

Our labeling tool comprises two main parts: the hexagonal lattice interface and the label collector. The former presents a bubble wrap interface with an underlying hexagonal grid that the user "pops" in order to specify the support of an object. The label collector obtains the labeling information from the interface and once it has collected enough labeling data, it proceeds to merge the labels. It also implements strategies for hinting an initial labeling for images such that successive users take less time and effort when labeling. Both modules are explained next in detail.

2.1. Hexagonal Lattice Interface

Several image processing and graphics algorithms have considered the use of hexagonal lattice as a way of representing and describing objects and images [2, 3, 4, 9, 14]. Hexagon-based descriptions of images are considered useful as they present many advantages with respect to rectangular grids: (i) Sampling density of a hexagonal lattice is higher than that of a square lattice (isoperimetry). Hexagons enclose more area than any other closed planar curve of equal perimeter, except a circle. (ii) Every hexagon in the lattice has six equidistant neighbors with a shared edge. Therefore curves can be represented in a better fashion and the lattice can more easily follow edges (greater angular resolution). (iii) Less ambiguity in defining boundaries and regions (uniform connectivity).

Since a hexagonal lattice presents superior symmetry, a more definite neighborhood and fewer samples to define boundaries compared to a rectangular lattice, we adopt this representation for the interface of our labeling tool.

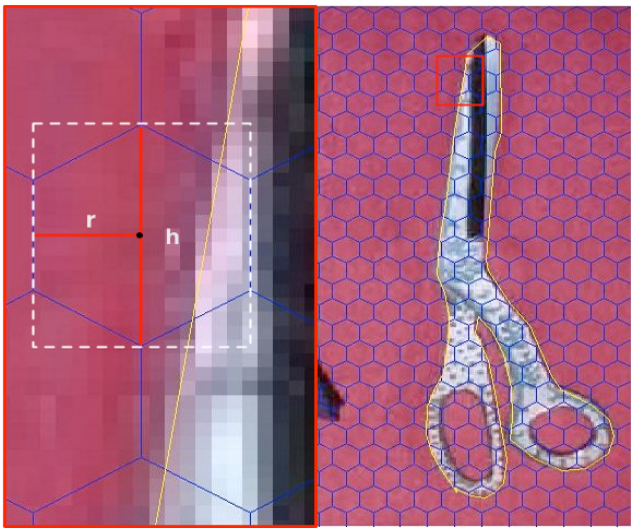


Figure 2. Hexagonal grid. In the figure r is half of the width of the surrounding rectangle, h is the height of the surrounding rectangle.

We model the center of each hexagon (c_x, c_y) in the grid as:

$$c_x = \begin{cases} 2xrs + o_x & \text{even} \\ 2xrs + rs + o_x & \text{odd} \end{cases} \quad (1)$$

$$x \in [0, \frac{N_x}{2rs+1}]$$

$$c_y = 2yhs + o_y \qquad y \in [0, \frac{N_y}{(2r-2)s}] \quad (2)$$

Where $r \geq 7$ is half of the width of the surrounding rectangle, h is the height of the surrounding rectangle, s is the scale with $s \in [1 \ldots n]$, o_i is the number of pixels to translate the grid in direction i and N_i the number of pixels in coordinate i. Figure 2 shows in detail the hexagonal lattice.

Over this hexagonal lattice we place a bubble wrap interface that permits the selection of hexagons for labeling. We selected the bubble wrap interface given the increasing popularity of online bubble wrap games[1]. Popping bubbles seems to be an enjoyable and fun activity, and since it naturally fits the hexagonal lattice, we include it as part of our labeling tool. Moreover, when a bubble is clicked, the user hears a satisfying "pop" sound. Each bubble's centroid is aligned with each hexagon centroid, therefore following the different scales and translations of the lattice. In this way, labeling of objects is still accurate for irregular object shapes (including objects containing holes), and time and effort still decrease, but now labeling is done in a more enjoyable setting.

We consider three different strategies when choosing a hexagon (bubble) for labeling. The user can select the bubbles that: (i) are completely contained inside the object, (ii) are contained at least 50% inside the object or (iii) contain some part of the object. Figure 3 shows an example of the different strategies for a given labeling.

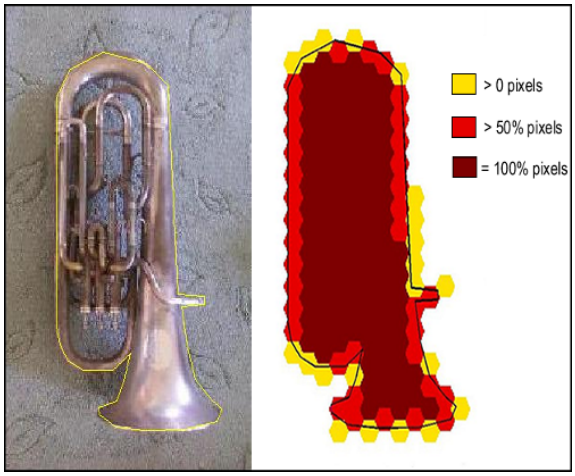


Figure 3. Different strategies for selecting "bubbles". This particular examples shows a specific size of hexagon and translation, representing one user labeling. We can observe that the contour approximation depends on the strategy chosen. The polygon contour corresponds to the ground truth label from Caltech-101 [1].

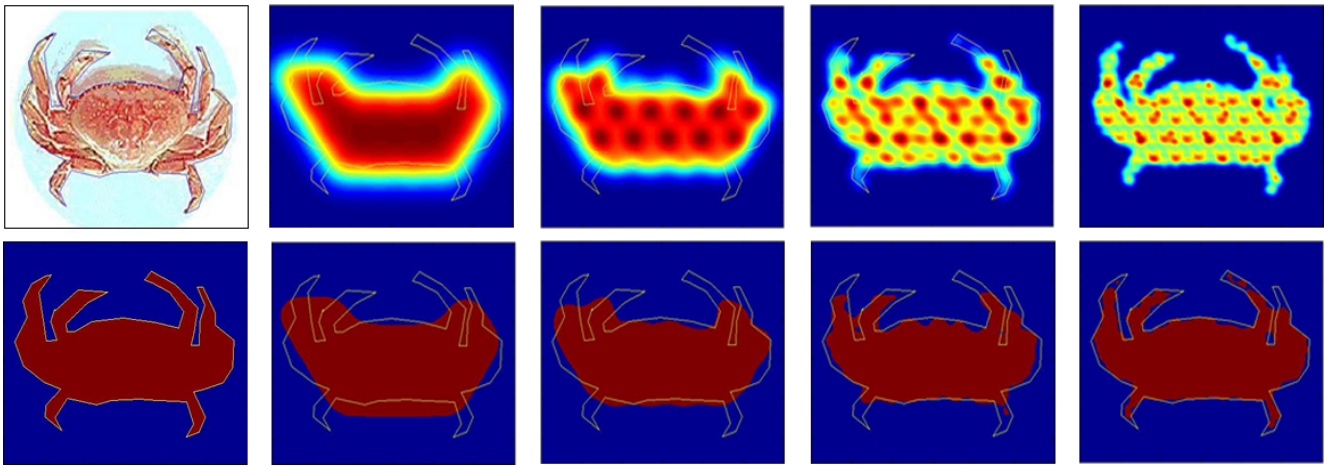
2.2. Collecting and Merging Labels

We collect several labels per image in order to obtain accurate contour masks of objects. Given an image, the tool presents different bubble sizes and global lattice translations for each user. The size of the bubbles gets smaller as the number of users increases, and the translations are randomly chosen. Figure 4 shows the different bubble sizes for a given image. For a given image (a), the first user will encounter a bubble size like (b), a second user will have a bubble size (c) and a third user will see (d). Although we agree that collecting only one label using the grid could lead to a less accurate mask, we consider more than one label with different granularities so we obtain a more detailed contour of the mask. Also when having more than one user's labeling the same object, we can obtain a less biased labeling.

With respect to translating the grid, one could either use deterministic lattice shifting or random shifting; for simplicity, we chose to use the latter.

Once we have obtained several labels for an image, we proceed to combine all the information in order to obtain the final labeling of the object. Although in this case we assume that there is one object to label in each image, we

[1] www.virtual-bubblewrap.com, www.puffgames.com/bubblewrap



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

Figure 5. Densities estimated for different number of users considering bubbles that are contained at least 50% inside the object . The first row shows the estimated densities and the second row shows the final mask, which is obtained by thresholding the densities. Column (a) shows the image and object to label (crab) and the ground truth mask (Caltech-101). Column (b) is the density and label estimated for the first user, using the biggest size bubble. Column (c) corresponds to the density and mask obtained once a third user has labeled the image with medium size bubbles. Column (d) corresponds to the density and mask obtained once a sixth user has labeled the image (with a small sized bubble). Finally, column (e) corresponds to the density and mask obtained once the ninth user has labeled the image. We can observe that the accuracy improves dramatically even for difficult objects such as a crab.

(a) (b) (c) (d)

Figure 4. Different scales in the hexagonal grid for a given image. Each grid encodes different information of the contours of an object without the inconvenience of selecting a large amount of pixels or drawing polygons.

could extend the tool for labeling multiple objects in the image. For a given scale and translation in the hexagonal grid, we keep the centers and scale (c_x, c_y, s) of each selected hexagon. This information is saved for each image in the data set. The final labeling is computed using kernel density estimation over all (c_x, c_y), which is an approximation of its probability density function. Therefore we compute Gaussian kernels (with identical masses) on each training data point and we adjust the "widths" to maximize the sum of leave-one-out log densities. Let X_i be a sequence of independent and identically distributed random variables with an unknown density function f where $\{x_i\}_{i=1}^{m}$ is a sequence of observations on X_i. We define our kernel density estimator as follows:

$$\hat{f}_h(x) = \frac{1}{mh_x^2} \sum_{i=1}^{m} K\left(\frac{x - x_i}{h_x}\right), \quad x \in \mathbb{R}^2 \tag{3}$$

$$K(x) = \frac{1}{2\pi} \exp\big(-\frac{1}{2}x^T x\big) \tag{4}$$

Where m is the number of Gaussian centroids of the object labeled and h_x is the variable bandwidth of the Gaussian kernel computed using a Newton's method in the log of initial guesses of h_x [5]. The initial guesses for the bandwidth are driven by the number of users that have already labeled the object. For example, if we obtain only one labeling, we consider a bigger bandwidth than when we have two or more labels from users. Using this method we can estimate a more precise final density with respect to the available information. Figure 5 shows the different density estimation for different number of users. We can observe that the bandwidth decreases inversely with respect to the number of users that have already labeled the image.

Once we obtain the estimated density for an object labeling, we find a threshold to obtain the final label. When thresholding, we throw out noisy points and possible mistakes that users could have made in the process of labeling the object. We can observe in Figure 6 that our labeling method also supports hollow objects. In Figure 6 we had to manually label the gap because the ground truth mask did not account for the hole that should be present there. We can see that the estimated density accounts for hollow regions. We can observe also that difference between the estimated mask and final mask is minimal, which shows the good labeling precision of BUBL.

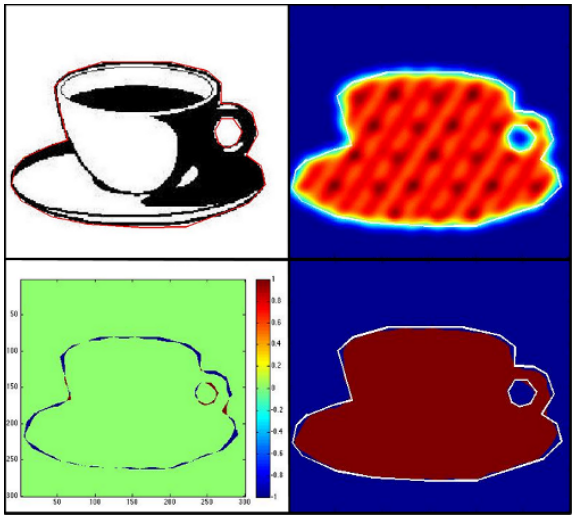


Figure 6. Support for objects with hollow regions. BUBL is able to label objects and consider possible holes inside objects. In this figure we hand label the cup's gap in order to check the accuracy of our labeling. In clockwise order: image to label with ground truth polygon, estimated density for 6 users, difference between the estimated mask and final mask with the superimposed ground truth polygon.

2.3. Supporting Labeling Effort

In order to make the labeling experience less demanding on the user's side, we propose three different approaches for computing an initial labeling for a given image. We take advantage of our multi-labeling scheme and of our global-to-local scale adjustment. Next we describe in detail each approach.

2.3.1 Using Available Labels

Since BUBL collects several labels per images, we can use this information to "hint" an initial labeling once we have obtained at least one labeling for that object. Given n previous labels, we compute the estimated density using Equation 3 and threshold it using a higher value t_1 than the one used for the final labeling. This way we are sure that the hinted labels are inside the object, in an area of higher confidence. Once we compute the mask, we superpose the mask over the new hexagonal lattice. Therefore we can present "pre-popped" bubbles to the new user, making the labeling faster and easier.

2.3.2 Using Partial Labeling for Refinement

In the same spirit as the approach before, we use the previous labels to "pre-pop" the bubble wrap interface. The major difference here is that the threshold value t_2 here is less than the one used for computing the final labeling ($t_2 < t_1$). Therefore we try to overestimate the object region by "over popping" some of the bubbles. This way the new user can correct the labeling by "unpopping" the bubbles the are not part of the object and continue popping the ones that are inside the object. By making the user correct the labels, we gain extra information about the location of the object and obtain a more refined label.

2.3.3 Using Other Databases to Learn Object Models

Another approach for hinting initial labels can be implemented by using other databases as examples to learn object models. In this case we don't use any previous labels, only the output of object classifiers. Any off-the-shelf object categorization method can be plugged into BUBL and use the bounding box or segment mask to "pre-pop" bubbles in the grid. Then the user can refine the labeling by "popping" and "unpopping" bubbles to get the object correctly. In the case where multiple objects appear in the image and the object categories are difficult to learn, we advocate using the first two approaches.

3. Experiments

We performed experiments using two current state-of-the-art image databases: Caltech-101 [1] and MSRC [13]. We use all categories of MSRC and a subset of 10 classes from Caltech-101 that are characterized by possesing difficult shapes [8], namely *accordion*, *crab*, *cannon*, *electric guitar*, *euphonium*, *gramophone*, *inline skate*, *revolver*, *watch* and *windsor chair*. We simulate bubble popping by considering the bubbles inside the contours of the object; thus there are no actual human users using the bubble wrap tool.

Ground truth masks from MSRC and polygon annotations from Caltech-101 (converted into masks) represent an ideal user in the experiments. Given the mask M_{gt} corresponding to the ground truth mask of the polygon region and M_b the mask corresponding to BUBL, we measure the labeling accuracy as:

$$A_{cc} = \frac{(M_{gt} \cap M_b)}{(M_{gt} \cup M_b)} \tag{5}$$

First we evaluate our method without labeling support. We use our classes of geometric object from Caltech-101 using 3 different scales ($r = 7{*}s,\ s = 1, 2, 3$) and 9 random translations in total (2 for $s = 3$, 3 for $s = 2$ and 4 for $s = 1$). The scales go from coarse to fine so $s = 3$ is the biggest and $s = 1$ is the smallest. We use the three different bubble pop strategies and also compare to the square lattice in order to show the superiority of the hexagonal grid.

Table 2 shows results corresponding to each selection strategy. We perform a 4-fold experiment to account for the randomness introduced by the lattice translation. We

observe that strategy 2 is the one that has the highest labeling accuracy, even for difficult shapes. Lowest accuracy is found for categories such as *crab*, *electric guitar*, *gramophone* and *revolver*. These categories are the most difficult, as they have more complicated shapes, like small corners and extensions. Some examples are shown in Figure 8.

Class	Strat. 1	Strat. 2	Strat. 3
accordion	0.87	0.97	0.88
crab	0.65	0.90	0.58
cannon	0.77	0.95	0.77
electric guitar	0.69	0.91	0.66
euphonium	0.78	0.94	0.77
gramophone	0.76	0.93	0.74
inline skate	0.83	0.96	0.82
revolver	0.68	0.91	0.65
watch	0.84	0.96	0.84
winsor chair	0.77	0.93	0.79
average accuracy	0.76	**0.94**	0.75

Table 2. Results for Caltech-101 using three different strategies. Strategy 1 corresponds to popping all bubbles that have any overlap with the object. Strategy 2 corresponds to popping all bubbles that overlap more than 50% with the object. Strategy 3 corresponds to popping all bubbles that overlap completely with the object.

In order to measure the contribution of the hexagonal lattice, we compare our results to those obtained using a rectangular lattice. The same number of scales and translations are used in both experiments. Table 3 shows the average accuracy of performing 4-fold experiments with a strategy of selecting bubbles with an overlap of 50% or more with the object. We can observe that choosing the hexagonal lattice gives better accuracy than using a rectangular grid.

Class	Number Images	Hexagonal Grid	Square Grid
accordion	55	0.97	0.84
crab	43	0.90	0.86
cannon	73	0.95	0.93
electric guitar	75	0.91	0.89
euphonium	64	0.94	0.93
gramophone	51	0.93	0.90
inline skate	31	0.96	0.94
revolver	82	0.91	0.89
watch	239	0.96	0.81
winsor chair	56	0.93	0.91
average accuracy	769	**0.94**	0.89

Table 3. Results for Caltech-101 using a hexagonal and a rectangular grid. We can see that the hexagonal grid gives superior accuracy for the same number of scales and translations.

We also performed experiments on the state-of-the-art dataset MSRC. This dataset is used for object categorization and presents multiple objects in each image. We use the same parameters as before and perform a 4-fold experiment. Table 4 presents results for labeling using the hexagonal and the rectangular grid. BUBL can accurately label most of all categories of this challenging dataset. We can also observe that BUBL has a lower accuracy compared to that of Caltech-101, as the object sizes in the dataset tend to be small (*face*, *cows*, *book*, *flower*, *boat*, *bird*).

Class	Number Images	Hexagonal Grid	Square Grid
building	15	0.88	0.84
grass	15	0.91	0.87
tree	15	0.90	0.86
cow	15	0.72	0.68
sheep	15	0.94	0.92
sky	15	0.86	0.83
aeroplane	15	0.88	0.84
water	15	0.90	0.88
face	15	0.70	0.67
car	15	0.92	0.90
bicycle	15	0.79	0.75
flower	15	0.92	0.89
sign	15	0.95	0.94
bird	15	0.88	0.85
book	15	0.96	0.94
chair	15	0.90	0.88
road	15	0.91	0.89
cat	15	0.93	0.90
dog	15	0.91	0.88
body	15	0.90	0.86
boat	15	0.68	0.60
average accuracy	315	**0.87**	0.76

Table 4. Results for MSRC using a hexagonal and a rectangular grid. We observe the superiority of hexagonal grid over the rectangular lattice for the same number of scales and translations. We notice that the average accuracy is lower than that of Caltech-101, as there are multiple objects in each image, of which many are small and difficult.

We implemented the first approach for reducing the labeling effort in order to measure the real contribution of this idea. We perform a 4-fold experiment and computed the average number of bubbles to pop in each category, first using the normal BUBL labeling and then using the labeling support approach using available labels. We picked a threshold $t_1 = 0.7$ are in order to compute the intermediate labels, as all values p_i in the density are $p_i \in [0, 1]$. Table 5 shows the results for this experiment. We observe that the effort for the labeling task is considerably reduced for the user by more than 58% on average. The percentage of popped bubbles yields a significant reduction in the labeling effort. For example, for the class *accordion* we need to pop only 30% of the original bubbles. Figure 7 shows in more detail the difference in effort for each scale in the labeling process. We observe that the last scale, the finest, gets the most benefit from using previous labels for an initial "hinting". This is a great advantage as the smaller scale has the most number of bubbles to select.

Class	Effort Hex Grid	Pre-popped Hex	Percentage Popped Bubbles
accordion	1071	321	30%
crab	605	274	42%
cannon	645	263	41%
electric guitar	354	201	57%
euphonium	747	283	38%
gramophone	770	317	41%
inline skate	1075	367	34%
revolver	424	216	51%
watch	713	300	42%
windsor chair	714	277	39%

Table 5. Results for Caltech-101 using available labels for pre-popping. We observe a significant decrease in the number of bubbles to pop in each category. Therefore the effort for the labeling task is considerably reduced for the user.

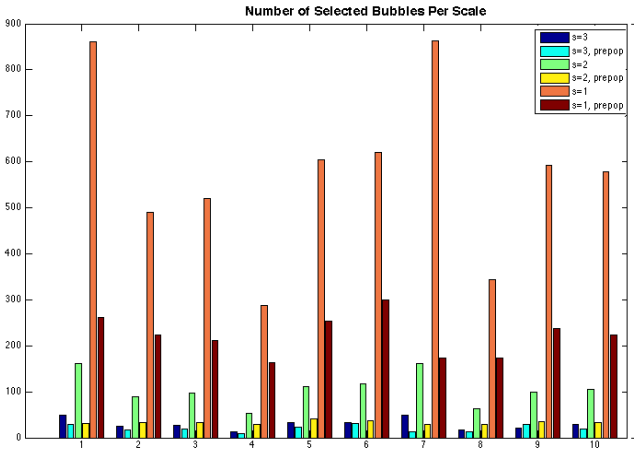


Figure 7. Effort difference for each scale in the labeling process. On the x axis we observe each category (accordion, crab, etc.) and on the y axis the number of bubbles that are needed to pop to label the object correctly. We can see a significant difference between using the previous available labels and not using them. We can see a decrease in the number of bubbles to select for all different scales.

4. Conclusion

We propose a data labeling tool, BUBL, that permits labeling images accurately using less time and effort. We use a hexagonal grid with a variable size tiling and a bubble wrap interface which makes labeling efficient and entertaining. The final labeling is computed using kernel density estimation, which results in accurate object contours. Also, multiple annotations from users are collected for every image, making it possible to "hint" a partial labeling so successive users can finish labeling in less time. Some remaining challenges for our method include labeling small and irregular objects, as well as labeling multiple objects at once. As future work we propose to address these drawbacks and make BUBL publicly available.

References

[1] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006.

[2] M. Golay. Hexagonal parallel pattern transformations. *IEEE Transactions on Computers*, 100(18):733–740, 1969.

[3] I. Her and C. Yuan. Resampling on a pseudohexagonal grid. *CVGIP. Graphical models and image processing*, 56(4):336–347, 1994.

[4] L. Middleton and J. Sivaswamy. *Hexagonal Image Processing: A Practical Approach*. Springer, 2005.

[5] V. Raykar and R. Duraiswami. Fast optimal bandwidth selection for kernel density estimation. In *Proceedings of the sixth SIAM International Conference on Data Mining*, pages 524–528. Citeseer, 2006.

[6] B. Russell, A. Torralba, K. Murphy, and W. Freeman. LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1):157–173, 2008.

[7] A. Sorokin and D. Forsyth. Utility data annotation with Amazon Mechanical Turk. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008. CVPR Workshops 2008*, pages 1–8, 2008.

[8] M. Stark and B. Schiele. How good are local features for classes of geometric objects. In *Proceedings of the International Conference on Computer Vision*, 2007.

[9] R. Staunton. One-pass parallel hexagonal thinning algorithm. *IEE Proceedings-Vision, Image and Signal Processing*, 148(1):45–53, 2001.

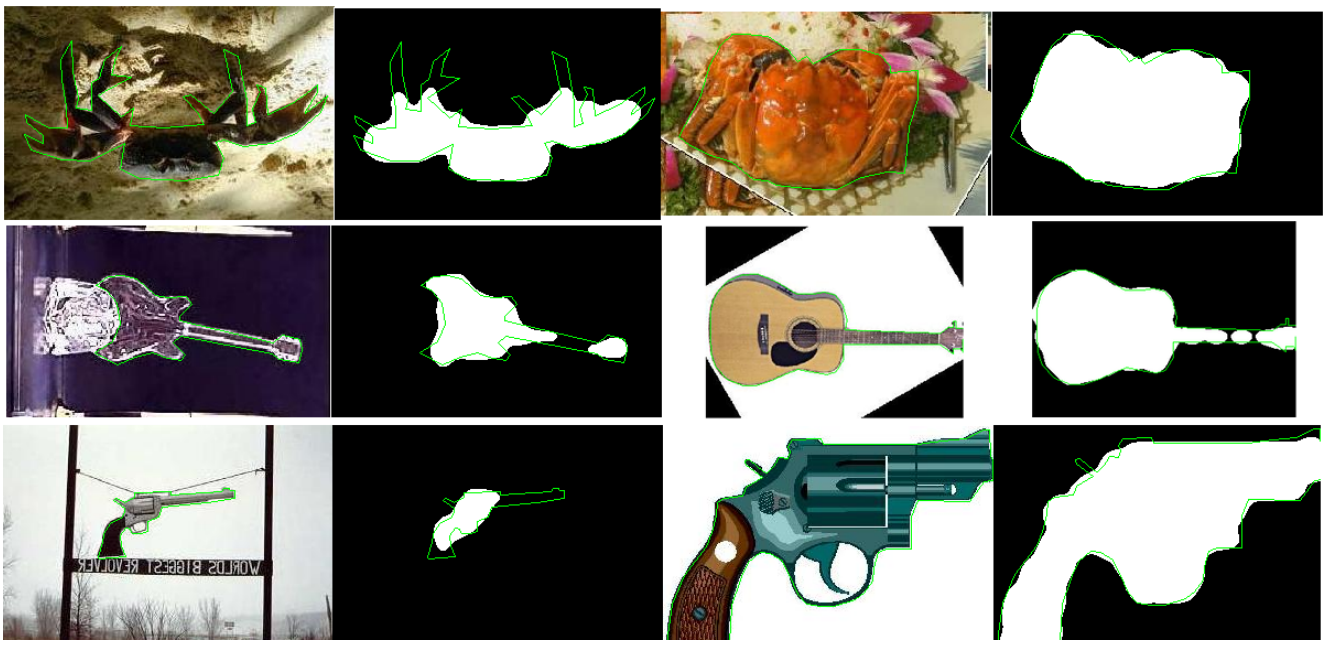
[10] A. Thomas, A. Rusu, and V. Govindaraju. Synthetic handwritten CAPTCHAs. *Pattern Recognition*, 2009.

[11] L. Von Ahn and L. Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM New York, NY, USA, 2004.

[12] L. Von Ahn, R. Liu, and M. Blum. Peekaboom: a game for locating objects in images. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 55–64. ACM New York, NY, USA, 2006.

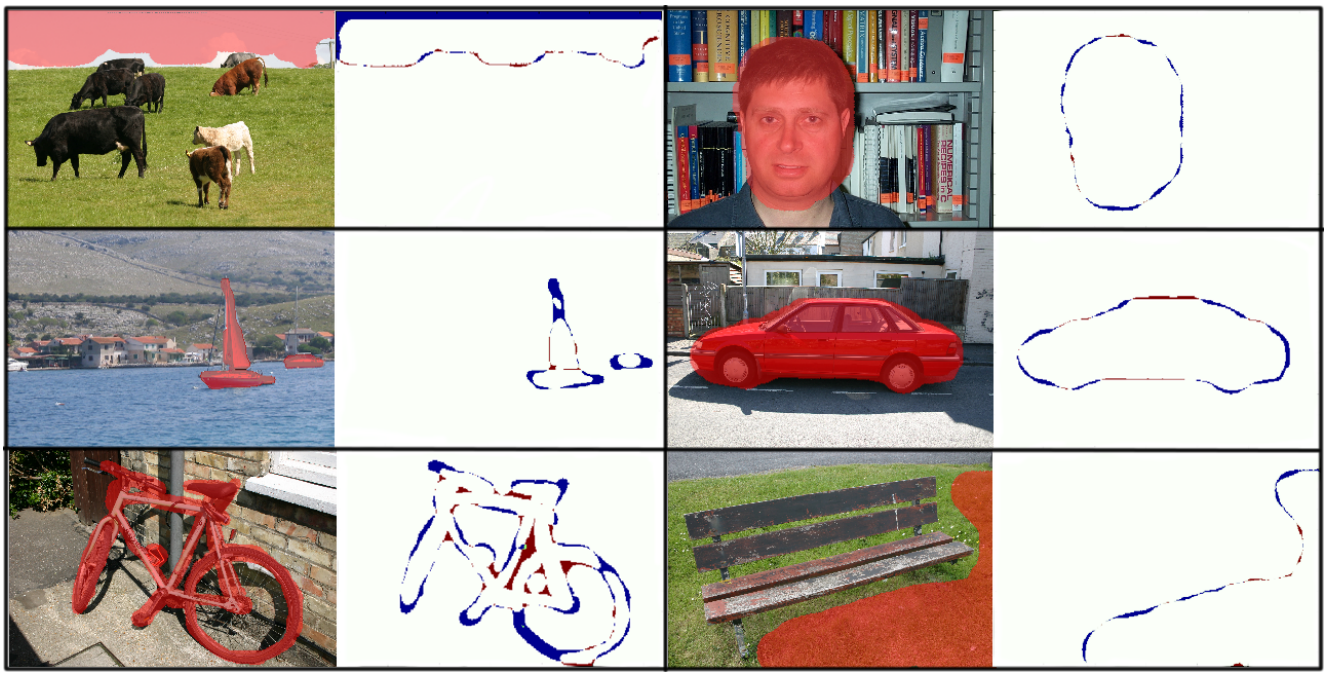
[13] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 2, 2005.

[14] C. Wüthrich and P. Stucki. An algorithmic comparison between square-and hexagonal-based grids. *CVGIP: Graphical Models and Image Processing*, 53(4):324–339, 1991.



(a) (b) (c) (d)

Figure 8. Examples of labeling using BUBL on Caltech-101. Column (a) shows an image with its ground truth polygon and (b) represents the labeling made by BUBL. Column (d) shows the labelings made by BUBL.



(a) (b) (c) (d)

Figure 9. Examples of labeling using BUBL on MSRC. Column (a) shows an image with its ground truth mask overlayed and (b) represents the bad examples of the difference between labeling made by BUBL and the ground truth mask. Column (d) shows successful examples. In (b) and (d), red color represents the an overestimation made by BUBL, and blue is the under estimation with respect to the ground truth mask.