Color Classification and Recycling Bin Detection

Rishabh Bhattacharya
Department of Mechanical & Aerospace Engineering
University of California, San Diego
ribhattacharya@ucsd.edu

***Abstract*—Object detection and Image recognition have become technologies of paramount importance in today's world of autonomous driving, intelligent robots, etc. which co-exist with humans in our living space. Object detection has applications like parcel tracking (for logistics robots), pedestrian tracking (for autonomous cars), and many more, rendering it an all important concept which demands continuous research and improvements, especially to guarantee safety.**

***Index Terms*—classifier, segmentation, RGB colorspace, Gaussian Discriminant Analysis, computer vision, bin detection, Maximum Likelihood Estimation, MLE**

I. Introduction

Any robot or autonomous system, has to be able to perceive the environment in order to interact with it. It is this interaction that makes them useful, like handling goods in a factory, sorting between different colours of glass in a recycling facility, making turns at the right intersections, etc. We have different kinds of sensors at our disposal to make this happen, like LIDAR, ultrasonic or even a regular camera capturing visible light. This project will be focusing on perceiving and detecting objects using RGB images captured by the robot.

Our task is to detect a blue recycling bin from an input image. This is particularly challenging due to the following,

- Our model has to learn the difference between blue colored recycling bins and objects that are blue but not recycling bins like cars, pieces of clothing, etc
- Our model has to segregate the colors from challenging images like a bright sky, recycling bin inside dark shade, partially visible bins, etc
- We also need to develop a one-size-fits-all shape statistic filter to filter out the infeasible detections

In order to tackle these, the problem is decomposed into two stages. The first step comprises of using a supervised learning technique like Gaussian Discriminant Analysis (II-A, III-A) to learn our model parameters. Once this is achieved, the learned parameters are used to create a binary mask (using maximum likelihood) which highlights the color of interest (blue in our case). This mask is then fed to the second part of the problem, which is bin detection (II-B, III-B). Using high level heuristics and shape statistics, we are able to successfully detect blue bins with relatively high accuracy, but with occasional failures as well (IV).

II. Problem Formulation

The overall problem is to identify *blue recycling bins* from a series of images. This problem statement can be further broken down into two smaller problems; blue color classification i.e. *pixel classification* (II-A) and bin detection i.e. *object detection* (II-B), which can be tackled individually.

A. Pixel classification

The task is to distinguish blue pixels from a set of RGB images ($I \in \mathbb{R}^{m \times n \times 3}$), where m & n are the dimensions of the image. We have pixel training data in the form $X \in \mathbb{R}^{401175 \times 3}$ with associated labels $y \in \{1, 2, 3, 4\}^{401175 \times 1}$, collected from a set of 60 RGB images using the *roipoly* [1] function. The y labels correspond to different classes of objects, which has been discussed III-A. We learn a probabilistic generative model $p(y, X \mid \omega, \theta)$ using Gaussian Discriminant Analysis, where ω and θ are parameters that are learned using MLE (Maximum Likelihood Estimation) approach. The training objective can be stated as :

$$\omega_{MLE}, \theta_{MLE} = \arg\max_{\omega, \theta} p(Y, X \mid \omega, \theta)$$

subject to

$$\sum_{k=1}^{K} \theta_k = 1$$

where $K = 4$ in this problem. For a new pixel x, the label y^* can be predicted as:

$$y^* = \arg\max_{k \in \{1,2,3,4\}} p(y_k, x \mid \omega_k^{MLE}, \theta_k^{MLE})$$

A binary mask ($B \in \mathbb{R}^{m \times n}$) is generated, which can then be used for object detection.

Note: For Part 1 of the project, we perform a similar analysis, but on a different dataset. The training dataset comprises of 3694 RGB pixels, which are extracted from 3694 RGB images ($I \in \mathbb{R}^{28 \times 28 \times 3}$) comprising of a single RGB value throughout. The labels in this case are $y \in \{1, 2, 3\}^{3694 \times 1}$, which belong to red, green and blue respectively. The parameter learning procedure is henceforth the same, with the exception that we don't return an image mask for this part.

B. Object detection

Using the binary image $B \in \mathbb{R}^{m \times n}$, we can employ high level image heuristics by employing the functions in *scikit-image* [2], such as *label, regionprops*, etc to enumerate the regions of interest in the masked image, and store those regions along with their properties in say, $g(B)$. We can then use shape statistics like i) aspect ratio of bounding box $\left(r = \frac{\text{height}}{\text{width}}\right)$ and

ii) % area covered by the image $\left(a = \frac{\text{bbox area}}{\text{image area}} \times 100\right)$ to determine whether the region is a valid bin or not.

$$\text{recycle bin} = \{g(B) \mid 1 \leq r \leq 2, a \geq 2\%\}$$

For all regions that have been classified as recycling bins, we want to collect the vertices of the smallest rectangle (or *bounding box*) (x_1, y_1, x_2, y_2) containing the object, where (x_1, y_1) and (x_2, y_2) are the vertices of the top-left and bottom right corners of the rectangle respectively.

III. Technical Approach

There are several technical approaches to solve the problem that has been stated above. Below we elaborate on the approaches that have been considered for the purposes of this project.

A. Color Classification

We use a quaternary classification system for blue recycle-bin segmentation. The classes in use are,

$$k = \begin{cases} 1 \rightarrow \text{blue bin} \\ 2 \rightarrow \text{blue but not bin} \\ 3 \rightarrow \text{other bins eg. green, brown, black} \\ 4 \rightarrow \text{shade on road, trees, etc} \end{cases}$$

The individual pixels are classified into one of the segments by training a generative probabilistic model using Gaussian Discriminant Analysis. We use a set of two parameters for this approach,

1) θ: to model the marginal probability distribution function (pdf) $p(y)$ of a label y
2) ω: to model the conditional pdf $p(x_i \mid y, \omega)$

$$\begin{aligned} p(\mathbf{y}, X|w, \theta) &= p(\mathbf{y} \mid \theta)p(X \mid \mathbf{y}, \omega) \\ &= \prod_{i=1}^{n} p(y_i \mid \theta)p(x_i \mid y_i, \omega) \end{aligned} \quad (1)$$

where, n is the number of examples in training set. The marginal and conditional probability densities are given by,

$$p(y_i \mid \theta) = \prod_{k=1}^{K} \theta_k^{\mathbb{1}\{y_i=k\}} \qquad p(x_i \mid y_i = k, \omega) = \phi(x_i; \mu_k, \Sigma_k)$$

where K is the number of classes ($K = 4$ for this problem), $\phi(x_i; \mu_k, \Sigma_k)$ is a Gaussian distribution with mean μ_k and Σ_k, given by,

$$\begin{aligned} &\phi(x_i; \mu_k, \Sigma_k) = \\ &\frac{1}{\sqrt{(2\pi)^d |\Sigma_k|}} \exp\left(-\frac{1}{2}(x_i - \mu_k)\Sigma_k^{-1}(x_i - \mu_k)^T \right) \end{aligned} \quad (2)$$

where d denotes the dimensions of our pixel data ($d = 3$ for this problem).

We obtain the MLE optimized values of θ_k, μ_k and Σ_k as follows,

$$\theta_k^{MLE} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{y_i = k\} \quad (3)$$

$$\mu_k^{MLE} = \frac{\sum_{i=1}^{n} x_i \mathbb{1}\{y_i = k\}}{\sum_{i=1}^{n} \mathbb{1}\{y_i = k\}} \quad (4)$$

$$\begin{aligned} &\Sigma_k^{MLE} = \\ &\frac{\sum_{i=1}^{n} (x_i - \mu_k^{MLE})(x_i - \mu_k^{MLE})^T \mathbb{1}\{y_i = k\}}{\sum_{i=1}^{n} \mathbb{1}\{y_i = k\}} \end{aligned} \quad (5)$$

An important advantage of this methodology is the availability of closed-form solutions for the modeling parameters. Now that we have our learned parameters, we can use it to classify any test pixel x_i by selecting the label k that maximizes the log likelihood.

$$y_i = \arg\max_k \ \log \theta_k^{\mathbb{1}\{y_i=k\}} \phi(x_i; \mu_k, \Sigma_k)$$

$$\begin{aligned} y_i = \arg\max_k \Bigg(&\log \theta_k - \frac{1}{2} \log |\Sigma_k| \\ &- \frac{1}{2}(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k) \Bigg) \end{aligned} \quad (6)$$

We are only interested in detecting blue recycling bins. Thus we use the segmentation to create a binary mask image, which has 1 in all positions where $y = 1$ (blue bin), and 0 otherwise. This mask is then further processed and analysed for the bin detection.

B. Recycling bin detection

The objective is to detect recycling bin shaped objects from the masked image $B \in \mathbb{R}^{m \times n}$. We use the functions *label* and *regionprops* from the scikit-image [2] *measure* library. After recognising the regions, we use shape level statistics to fine tune the results.

1) The *label* function labels the connected regions of the mask and assigns them the same value.
2) The *regionprops* function returns various properties of the labelled image, like coordinates of bounding boxes, area of labels, etc. Let us consider the properties to be stored in $g(B)$.
3) We now consider two shape level statistics to filter out positive detections;
   a) Aspect ratio: aspect ratio of bounding box $\left(r = \frac{h}{w}\right)$. We consider positive examples to have $1 \leq r \leq 2$, based on the usual dimensions of a recycling bin, and also accounting for tilted/non-erect orientations.
   b) % area covered by the image: $\left(a = \frac{\text{label area}}{\text{image area}} \times 100\right)$. The aspect ratio condition can be fulfilled by small but erroneous detections as well. Thus in order to eliminate those, we add a constraint that places a lower bound on the area of the detected label w.r.t the image size. For our purposes, we have considered a to be 2% i.e. the area of the detected region should be at least 2% of the total image size.

Thus to determine whether the labelled region is a valid bin or not, we have

$$\text{valid detections} = \{g(B) \mid 1 \leq r \leq 2, a \geq 2\%\} \quad (7)$$

IV. Results

We will present results for Part 1 of the project (pixel classification) and then part 2 (blue-bin segmentation) and bin detection.

A. Part 1: Pixel Classification

θ	0.36599892	0.3245804	0.30942068

TABLE I: Learnt parameters for θ^{MLE}

μ_1	0.75250609	0.34808562	0.34891229
μ_2	0.35060917	0.73551489	0.32949353
μ_3	0.34735903	0.33111351	0.73526495

TABLE II: Learnt parameters for μ^{MLE}

Σ_1	0.0370867	0.01844078	0.01863285
	0.01844078	0.06201456	0.00858164
	0.01863285	0.00858164	0.06206846
Σ_2	0.05578115	0.01765327	0.00873955
	0.01765327	0.03481496	0.0170234
	0.00873955	0.0170234	0.05606864
Σ_3	0.05458538	0.00855282	0.0171735
	0.00855282	0.05688308	0.01830849
	0.0171735	0.01830849	0.0357719

TABLE III: Learnt parameters for Σ^{MLE}

	Training Set		
	R	G	B
R	**100%**		
G	0.33%	**99.66%**	
B	1.13%		**98.86%**

TABLE IV: Precision results for the training dataset (actual colors in each row)

	Validation Set		
	R	G	B
R	**100%**		
G		**100%**	
B			**100%**

TABLE V: Precision results for the validation dataset (actual colors in each row)

	Test Set		
	R	G	B
R	**100%**		
G		**100%**	
B			**100%**

TABLE VI: Precision results for the test dataset (actual colors in each row)

We can observe that the precision results are very good for the training dataset, while we achieve perfect accuracy in the validation and testset. However, these results are to be taken with a grain of salt since the images here are singular valued RGB images, which are fairly straightforward to classify.

B. Part 2: Blue bin segmentation

We try to normalize the pixel values in our dataset by dividing with 255, so that the pixel values are now in the range of $[0, 1]$, which makes computation faster, especially when we are dealing with a huge number of data points, such as in this problem.

θ	0.30169128	0.07513928	0.11621861	0.50695083

TABLE VII: Learnt parameters for θ^{MLE}

μ_1	0.21842954	0.35367636	0.70068307
μ_2	0.33505552	0.48272488	0.71232841
μ_3	0.43780456	0.46381431	0.43328699
μ_4	0.53680871	0.4898032	0.37993842

TABLE VIII: Learnt parameters for μ^{MLE}

Σ_1	0.03267098	0.03278538	0.01612912
	0.03278538	0.04163134	0.0257891
	0.01612912	0.0257891	0.04023598
Σ_2	0.03775463	0.03521093	0.02225895
	0.03521093	0.04209727	0.03538976
	0.02225895	0.03538976	0.04533566
Σ_3	0.06813322	0.03549058	0.02034214
	0.03549058	0.0620168	0.04199292
	0.02034214	0.04199292	0.04886996
Σ_4	0.06365647	0.04622912	0.04324299
	0.04622912	0.04734501	0.04121957
	0.04324299	0.04121957	0.04874271

TABLE IX: Learnt parameters for Σ^{MLE}

Figures 1-10 below show the mask that has been applied to our images using color segmentation. The hypothesis behind the FAIL results have been discussed in the corresponding image captions.

C. Part 3: Bin detection

We can deduce from figs. 1-10 that our mask is performing well over a wide range of pictures, albeit some challenging parts where there are multiple recycling bins (7), bins kept in shadows (4, 5, 7), etc. The mask and the labels clearly support our reasoning in most cases of success and failure, and can be used to further improve upon the algorithm. The images with a successful detection have the bounding box co-ordinates in the image title. Discussions and possible explanations on the behaviour of the algorithm have been discussed in the image captions that follow.

V. Conclusions

We implemented a probabilistic generative algorithm, to perform image recognition on a set of test images. The problem was tackled in two parts, by building a color segmentation algorithm which generated a binary mask. The mask was then fed into a shape detection algorithm, which grouped similar regions of the image. We then used few shape statistic measures to filter the desired bounding boxes which enclosed the blue recycling bins. We realise that the mask implemented using Gaussian Discriminant Analysis is performing very well with a variety of images. While our shape statistic filter has successfully filtered out noise from a lot of images, it has also been stringent enough to not recognize a couple of positive examples. The next part of this project could be focused on a more rigorous fine-tuning of the the statistic filter. A balanced filter should be conservative enough to recognise positive examples, while being stringent enough to discard noise/erroneous detections. That could help robustify the implementation, leading to more successful positive detections.

References

[1] Jdoepfert. Jdoepfert/roipoly.py: Select a polygonal region of interest (roi) with python and matplotlib, similar to the roipoly.m function from matlab.

[2] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014.

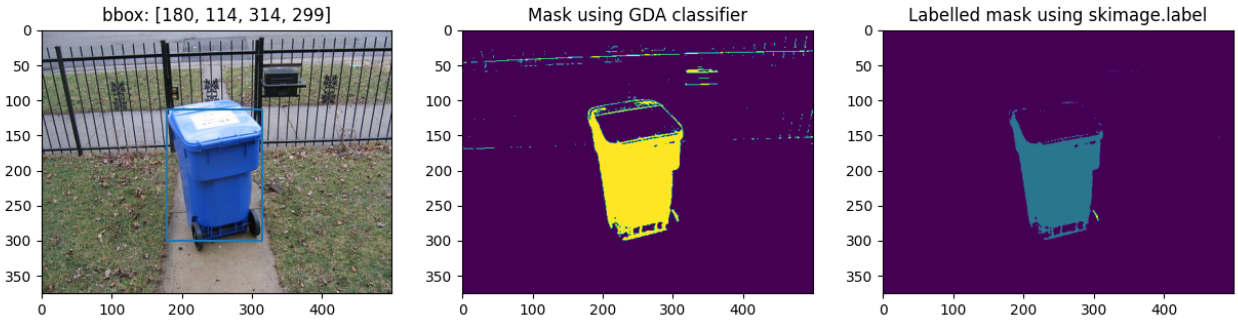

Fig. 1: Result: PASS

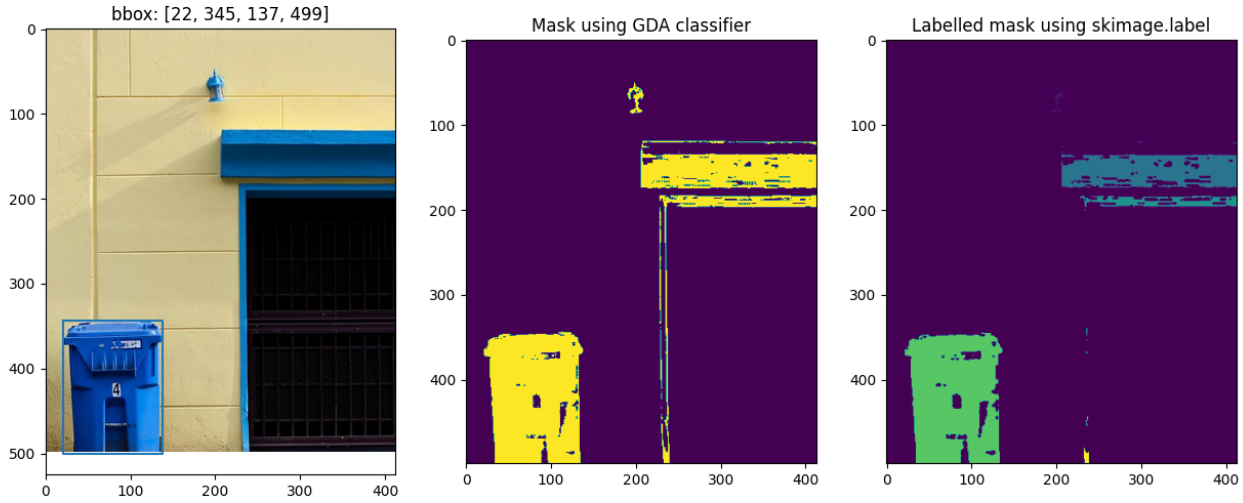

Fig. 2: Result: PASS

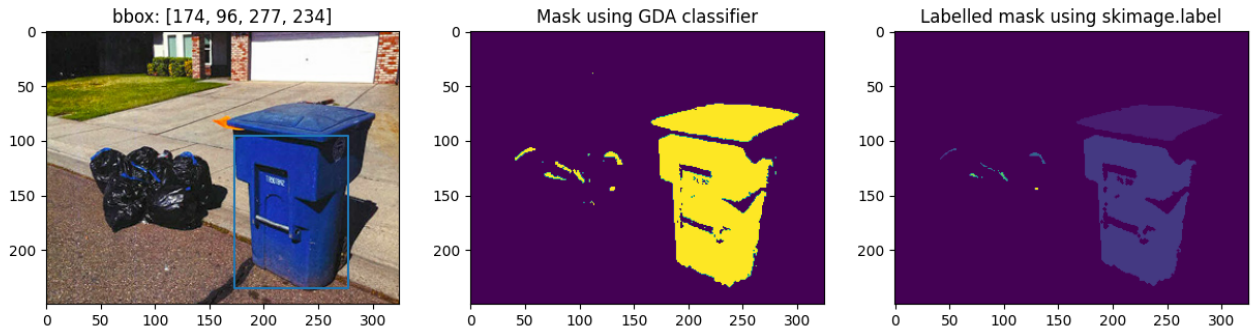

Fig. 3: Result: PASS

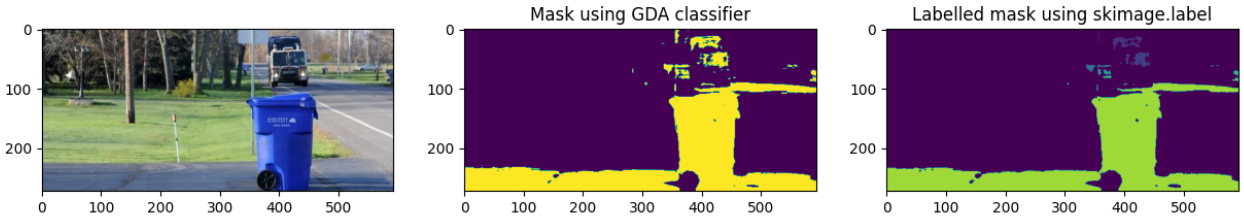

Fig. 4: Result: FAIL; Comments: The mask also includes a large portion of the road. The resulting bounding box was too wide. This directly conflicted with the aspect ratio filter, leading to a fail result.

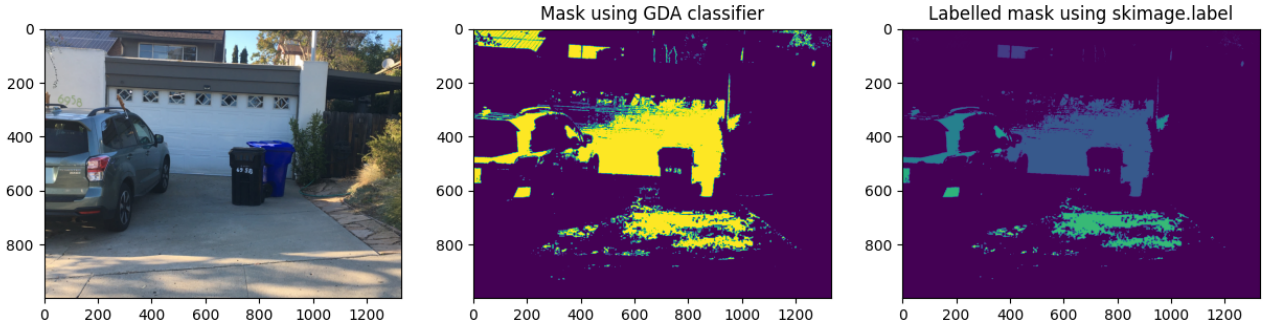

Fig. 5: Result: FAIL; Comments: The bin is in the shade, which means it is present in a place with predominantly dark pixels, which are not being captured by the mask.

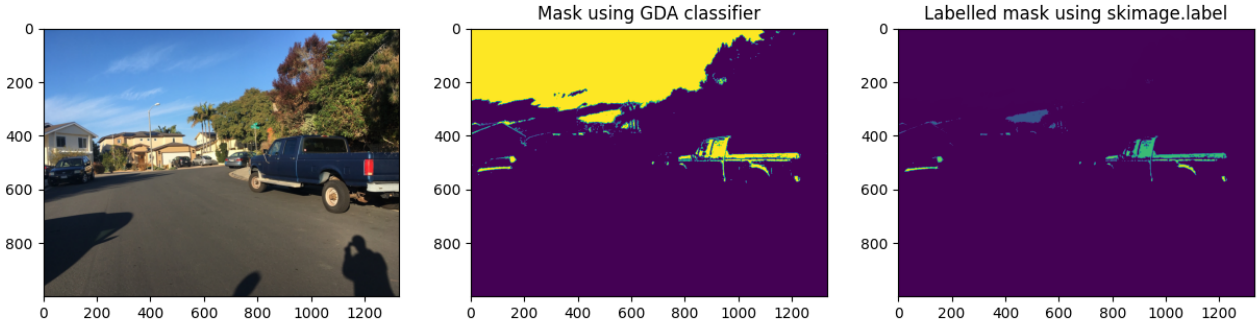

Fig. 6: Result: PASS

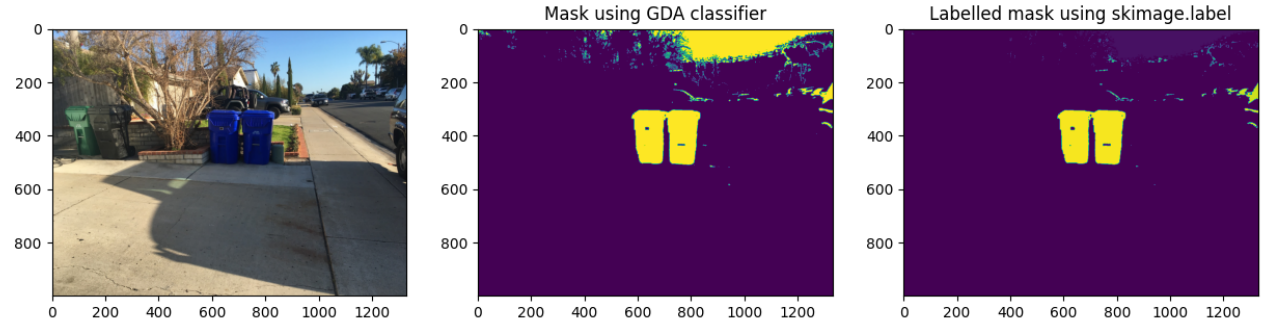

Fig. 7: Result: FAIL; Comments: Even though the bins are clearly masked as shown, the bounding box included both the bins. This was too wide according to our aspect ratio filter, so the bounding box was dropped. A successful adjustment to the ratio was performed (lowering the minimum value to 0.7), but it led to overall poor results from the other images. This problem can be solved if somehow the bins are detected as two separate units, instead of a single one.

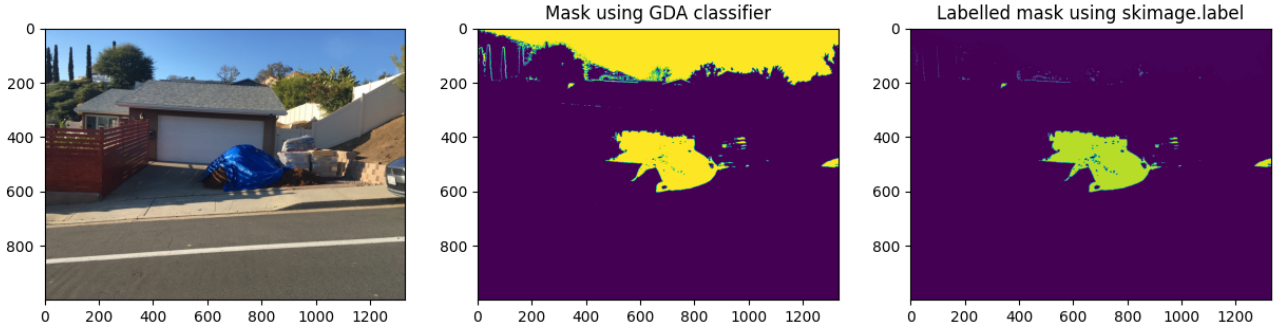

Fig. 8: Result: PASS

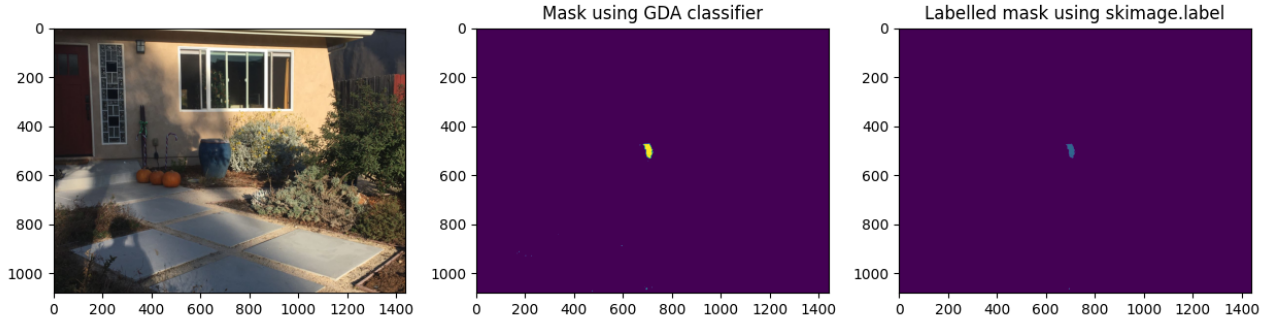

Fig. 9: Result: PASS

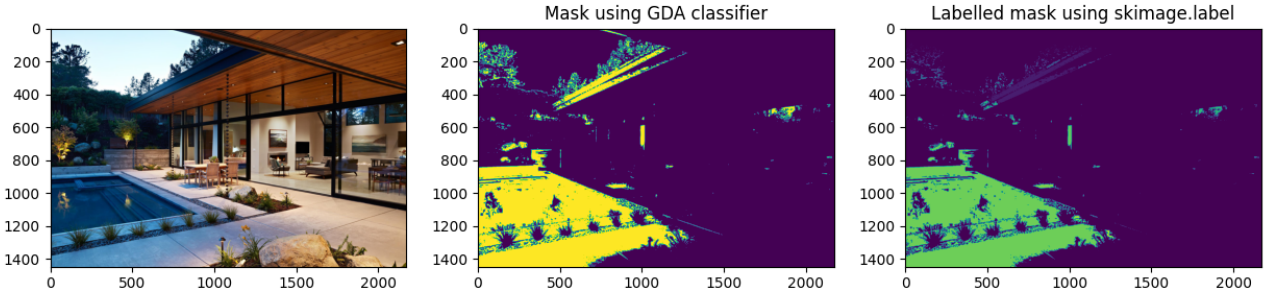

Fig. 10: Result: PASS