COLOR CLASSIFICATION AND RECYCLING BIN DETECTION

Joseph Chang

Department of Electrical and Computer Engineering, University of California, San Diego

ABSTRACT

Most existing object detection methods train models to recognize objects using convolutional neural networks. We utilize a probabilistic algorithm to detect recycling bins. First, we collect data of most common colors in RGB images containing recycling bins. Gaussian Naive Bayes is used to train the MLE parameters. The parameters are used to perform image segmentation on RGB images to classify each pixel as a specific color. Then, possible recycling bin regions are filtered with known high-level features, shape, and pixel density. Finally, we find the bounding boxes of recycling bins in a given image.

***Index Terms*—** computer vision, object detection, image segmentation

1. INTRODUCTION

Object detection is an important topic in computer vision. It is used to gain a 2D understanding of the world from camera images. Detection of recycling bins in particular could be used for autonomous garbage trucks in the future. Although convolutional neural networks (CNN) have experienced huge success in recent years, probabilistic models are still viable for object detection when the target object does not vary too much in color and shape as is the case with recycling bins. Generative and discriminative models can both be used for image segmentation including Logistic Regression, Gaussian Naive Bayes, and Gaussian Discriminant Analysis. We choose Gaussian Naive Bayes rather than Logistic Regression as it trains a quadratic decision boundary such as an ellipse, parabola, or hyperbola. This generates a model that is better fitted to training data which is optimal for recycling bin detection since the blue color does not vary significantly. Logistic Regression also has high bias and low variance while Naive Bayes has low bias and high variance. Naive Bayes again is the better choice as we prioritize low variance for image segmentation. Even if the color is not accurately trained, it's important it doesn't' overlap with other colors which will increase sparsity and noise in segmentation. On the other hand, Gaussian Discriminant Analysis removes the naive assumption that predictors are independent which is worse for training segmentation since colors are mostly uncorrelated. Hence, we use Naive Bayes to train our parameters.

In this paper, we perform multiple steps. First, we collect label and collect training data of recycling bin blue and various colors. Then, we use Naive Bayes to train MLE parameters for each color class. These parameters are fed into an algorithm which takes an RGB image as input and performs image segmentation to generate a matrix of labeled classes. This matrix is filtered to produce a segmentation mask displaying the recycling bin pixels. The pixel regions are filtered with morphological operations, height-width ratio, and area to output bounding box coordinates of all recycling bins in the image. In the future, the proposed algorithm can be used for object detection and classification of recycling bins in daytime autonomous driving scenarios.

2. PROBLEM FORMULATION

We train our MLE parameters using Gaussian Naive Bayes. For the 5 to 6 colors to classify, we label a training dataset X_i for each. There is 5 to 6 because we include $skyblue$ as a class in our first experiment and do not in include it in our second experiment. Given data from each color class $X_i \epsilon R^{n \times d}$ where n is the number of pixel samples and d is the number of color classes, we train the Gaussian Naive Bayes parameters θ_k^{MLE}, μ_k^{MLE}, and σ_k^{MLE} where μ is mean and σ is covariance. The parameters are determined by solving the following constrained optimization problem shown below.

$$\max_{\boldsymbol{\theta},\boldsymbol{\omega}} \log p(\mathbf{y}, X \mid \boldsymbol{\omega}, \boldsymbol{\theta}) \quad \text{subject to} \quad \sum_{k=1}^{K} \theta_k = 1$$

$$\theta_k^{MLE} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{y_i = k\} \tag{1}$$

$$\mu_{kl}^{MLE} = \frac{\sum_{i=1}^{n} x_{il} \mathbb{1}\{y_i = k\}}{\sum_{i=1}^{n} \mathbb{1}\{y_i = k\}} \tag{2}$$

$$\sigma_{kl}^{MLE} = \sqrt{\frac{\sum_{i=1}^{n} (x_{il} - \mu_{kl}^{MLE})^2 \mathbb{1}\{y_i = k\}}{\sum_{i=1}^{n} \mathbb{1}\{y_i = k\}}} \tag{3}$$

Then, these parameters are used to evaluate validation and test images. It evaluates which color class y_* each pixel likely belongs to where $y \epsilon R^n$ contains an integer label between 1 and the number of color classes for each of the n pixels in an image. This is performed by minimizing the difference between the Gaussian estimate of the selected pixel and each color class as shown in the equation below.

$$y_* = \underset{k \in \{-1,+1\}}{\arg\min} \left\{ \log \frac{1}{\theta_k^2} + \sum_{l=1}^{d} \log \sigma_{kl}^2 + \frac{(x_{*l} - \mu_{kl})^2}{\sigma_{kl}^2} \right\}$$

After image segmentation, we create a mask from $y*$ by setting pixels classified as recycling bin blue to 255 and pixels classified as other colors to 0. Then, morphological operations are performed to remove noise from the images using erosion then dilation. The recycling bin bounding boxes coordinates are x, y, w, h where x, y are the x and y coordinates at the bottom left of the bounding box and w, h are the width and height of the bounding box. The coordinates are determined using by finding contour regions within specified height-width ratio and area bounds.

3. TECHNICAL APPROACH

Our algorithm requires initial training data collection given a dataset of 60 training images in various environments where 50 images contain 1 to 2 blue recycling bins. Before labeling, we need to decide which color classes to use for segmentation. We choose binblue, green, brown, gray, darkgray, and skyblue as these cover most colors commonly found in recycling bin images. Grass is green, dirt and wood are brown, sidewalks are gray, streets are dark gray, and the sky is blue. Next, we choose to run our entire algorithm in the HSV colorspace as saturation and brightness are separate channels from hue or color unlike RGB. This will capture more variation in the binblue in brighter or darker lighting conditions without affecting the hue channel. We utilize roipoly to select regions of interest in each image. Because this is a memory-consuming task, we label 20 images at a time and save 3 array files for each color class. Once training data is collected, we load the color data for each class into a X and begin training our Gaussian Naive Bayes parameters θ_k^{MLE}, μ_k^{MLE}, and σ_k^{MLE}. These values are saved for each color class.

Next, we perform image segmentation given a validation image that was not used in training. We convert the input image to HSV and classify each pixel as one of the color classes using the parameters obtained. The resulting mask is matrix of integers from 1 to the number of color classes. Next, we obtain a recycling bin mask by filtering the segmented image. Binblue pixels are set to white and other pixels are set to black. Since there is a lot of noise from incorrectly classified pixels, we perform 2 iterations of erosion with a 3x3 and 5x5 kernel respectively followed by three iterations of dilation where the first two are with 5x5 kernels and the third is with a 3x3 kernel. This reduces noise and empty space between suspected recycling bin regions making detection easier. To find the bounding box, we find contours in the mask. Only regions with $contourArea > areaImg * 0.005$ are kept. Then, the bounding box coordinates are calculated. To filter the bounding boxes, we calculate the height and width ratio of 32, 64, and 95 gallon recycling bins to be 1.44-1.66. Given the segmentation is not perfect, however, we give some breathing room and keep bounding boxes with $height < 2.5 * width$ and $height > width$ as recycling bins.

4. RESULTS

We perform two experiments. In the first, we use 6 color classes: binblue, green, brown, gray, darkgray, and skyblue. The sky is visible in many images with recycling bins and can be misclassified as binblue. Hence, adding skyblue as a class for segmentation is expected to increase accuracy for images where recycling bins are not excessively light due to bright sunlight. We train our Gaussian Naive Bayes HSV MLE parameters θ_k^{MLE}, μ_k^{MLE}, and σ_k^{MLE} on these 5 classes shown in Fig. 1. Running our algorithm with 6 classes on validation images, we achieve 100% accuracy on the validation dataset where accurately detected images return one bounding box around each recycling bin and no other bounding boxes. However, with our autograder's test images, we correctly detect recycling bins in 5.25/10 images. The results on validation images are shown in Fig. 3-12 below.

	theta	mu
binblue	[[0.27835517]	[[115.13352253 114.63545031 160.94565961]
skyblue	[0.10378015]	[106.07478677 131.41896163 208.9642748]
green	[0.15038521]	[34.33449253 58.1265553 147.6993827]
brown	[0.04629963]	[60.21707871 91.18379368 101.00369535]
gray	[0.25591744]	[82.09343652 18.06988164 154.66284485]
darkgray	[0.16526239]]	[99.22357304 91.66035677 70.54801375]]

	sigma
binblue	[[14.7687992 72.7330568 38.86671294]
skyblue	[3.17002513 45.41522738 28.22643781]
green	[45.44781378 46.30584937 55.75221084]
brown	[21.95448049 56.82583749 43.46790927]
gray	[43.44944725 20.47648272 38.12967518]
darkgray	[26.40357461 34.41451395 27.32424598]]

Fig. 1: GNB MLE parameters with skyblue class

In the second experiment, we use 5 color classes: binblue, green, brown, gray, and darkgray. We remove skyblue as it could misclassify recycling bin pixels as skyblue in bright environments with sunlight reflection. We train our Gaussian Naive Bayes HSV MLE parameters on these 5 classes as shown in Fig. 2 and run our algorithm on validation images. We achieve 80% accuracy on the validation dataset where accurately detected recycling bin images return one bounding box around each recycling bin and no other bounding boxes. The errors occur when the algorithm draws bounding boxes around a door frame door frame with color similar to binblue and fails to detect an occluded recycling bin. However, with our autograder's test images, removing the skyblue class correctly detects recycling bins in 7.25/10 images. The results on validation images are shown in Fig. 13-22 below.

```
binblue   theta          mu
darkgray  [[0.31058804]  [[115.13352253 114.63545031 160.94565961]
green      [0.18439939]   [ 99.22357304  91.66035677  70.54801375]
brown      [0.16779946]   [ 34.33449253  58.1265553  147.6993827 ]
gray       [0.05166102]   [ 60.21707871  91.18379368 101.00369535]
           [0.28555208]]  [ 82.09343652  18.06988164 154.66284485]]
          sigma
binblue   [[14.7687992  72.7330568  38.86671294]
darkgray   [26.40357461 34.41451395 27.32424598]
green      [45.44781378 46.30584937 55.75221084]
brown      [21.95448049 56.82583749 43.46790927]
gray       [43.44944725 20.47648272 38.12967518]]
```

Fig. 2: GNB MLE parameters without skyblue class

Comparing the two experiments on validation data, we observe the detection algorithm without a skyblue class performs worse and accurately detects only recycling bins in 8/10 images while the algorithm with a skyblue class accurately detects only recycling bins in 10/10 images. Including the skyblue color class helps remove white from the image such as the garage door in 05.jpg which is detected as binblue without the class. Skyblue is detected on multiple recycling bins causing the binblue region to be smaller. This leads to a smaller bounding box than the ground truth. In real applications, however, it is better to detect the recycling bin with a slightly inaccurate bounding box than to detect a recycling bin when there is none. Hence, it makes more sense to have a skyblue class.

The door frame in 02.jpg is classified as a recycling bin when not using a skyblue class. It's hard to avoid classifying incorrectly as it has the shape dimensions and color as a recycling bin. A possible way to avoid classifying it wrong is to check the pixel density inside the frame, but it's difficult since only half of it is shown in the image. In real life though, blue door frames are rare. In general, having more color classes helps avoid classifying non-binblue pixels as binblue since similar colors such as gray or green are classified. Adding white, tan, and medium gray color classes would likely further increase accuracy. This is especially true for medium gray which is incorrectly classified as binblue in images 01, 04, and 05.

Segmentation accuracy is important, but filtering detected bounding boxes is where most progress was made in finding the recycling bins. Removing noise through morphological operations and filtering based on area and height-width ratio allowed us to get the accurate detection. The brown color class is not as prominent in segmentation as expected. Although dirt, wood, and brick are in several images, the brown color is mixed with green tinges. Hence, if one class was to be removed, it would be brown.

5. CONCLUSION

In this paper, we utilized probabilistic algorithm to detect recycling bins. Data was first collected for commmon color classes. Then, we used Gaussian Naive Bayes to train MLE parameters in the HSV colorspace. Using the parameters, we performed image segmentation classifying each pixel as a color. The matrix is thresholded to create a mask of possible recycling bin regions. The regions are filtered with morphological operations, height-width ratio, and by area size. Contours are detected to return the bounding box coordinates of all recycling bins. Further research towards accurate recycling bin detection include increasing the number of color classes and additional filtering of mask regions.

Fig. 3: 01.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: [[196, 154, 310, 291]]
actual box: [[182, 101, 313, 295]]

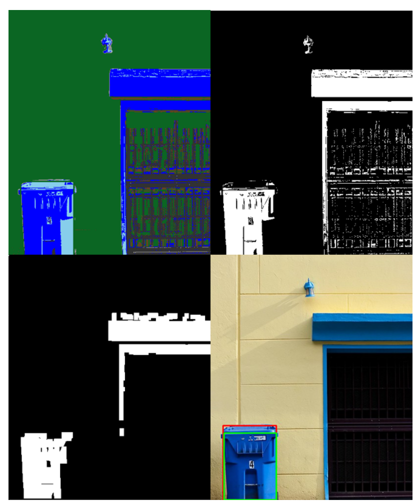

Fig. 4: 02.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: [[27, 362, 133, 497]]
actual box: [[25, 347, 133, 497]]

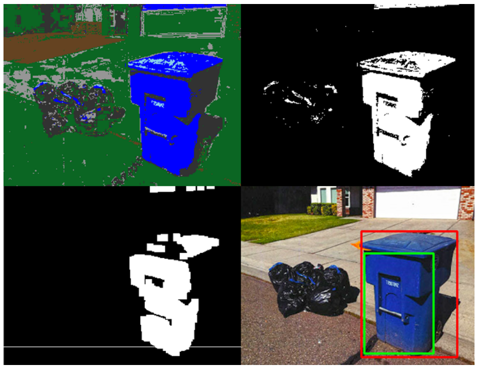

Fig. 5: 03.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: [[172, 95, 267, 234]]
actual box: [[168, 64, 300, 239]]

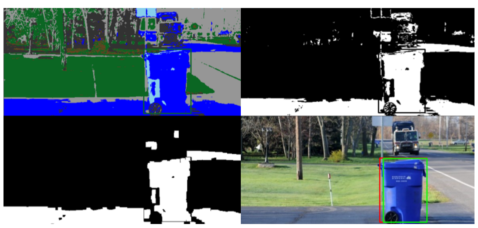

Fig. 6: 04.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: [[358, 105, 467, 264]]
actual box: [[349, 104, 467, 264]]

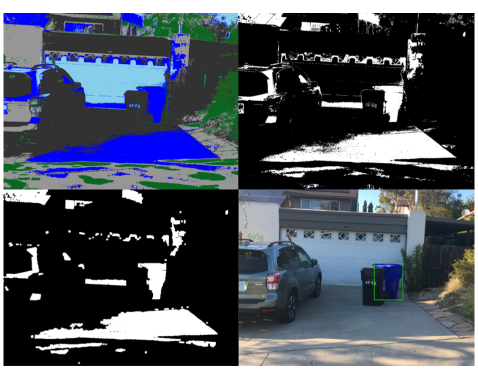

Fig. 7: 05.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: [[763, 417, 924, 622]]
actual box: [[762, 416, 924, 622]]

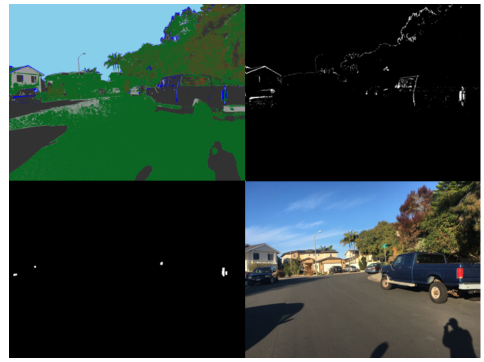

Fig. 8: 06.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: []
actual box: []

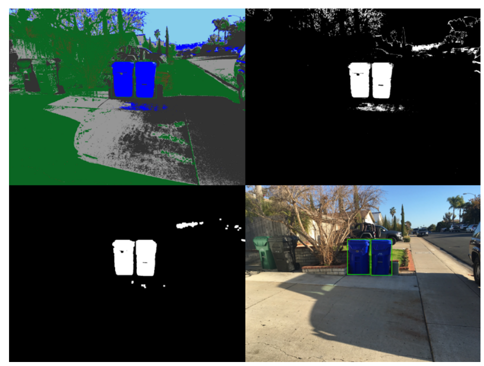

Fig. 9: 07.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: [[712, 306, 829, 509], [579, 306, 706, 504]]
actual box: [[578, 305, 706, 504], [711, 305, 830, 509]]

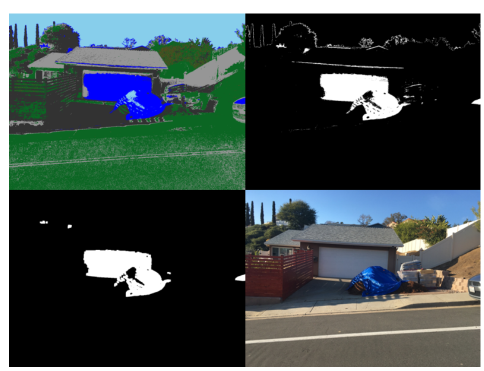

Fig. 10: 08.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: []
actual box: []

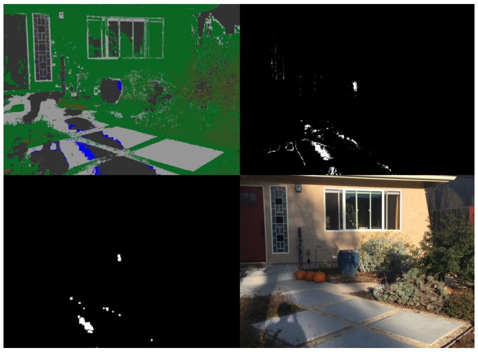

Fig. 11: 09.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: []
actual box: []

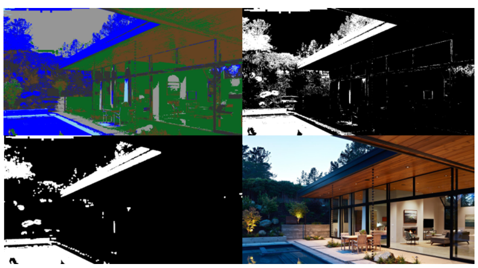

Fig. 12: 10.jpg, Segmentation, Mask, Filter, Bounding box, with sky blue class
estimated boxes: []
actual box: []

Fig. 13: 01.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: [[183, 102, 313, 291]]
actual box: [[182, 101, 313, 295]]

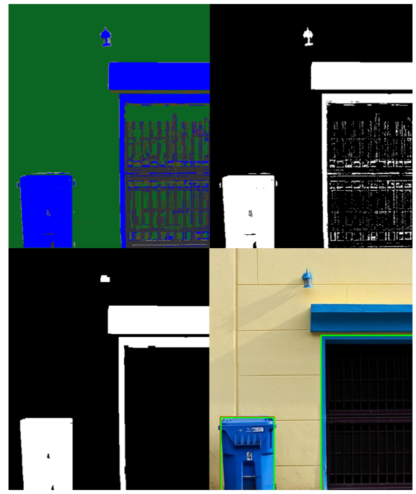

Fig. 14: 02.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: [[26, 348, 133, 497], [227, 183, 413, 499]]
actual box: [[25, 347, 133, 497]]

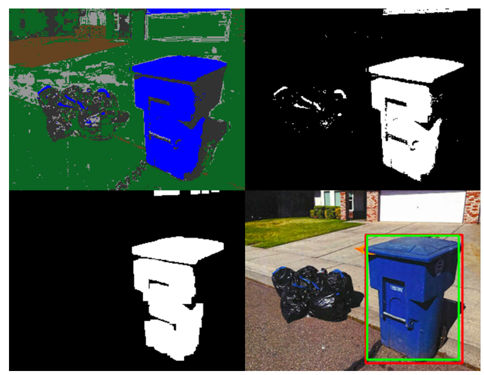

Fig. 15: 03.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: [[170, 65, 295, 234]]
actual box: [[168, 64, 300, 239]]

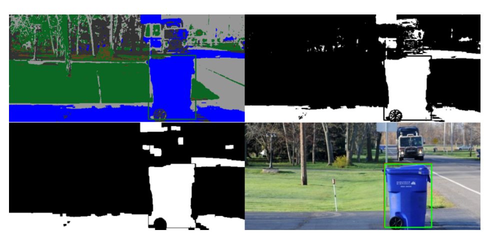

Fig. 16: 04.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: [[350, 105, 467, 264]]
actual box: [[349, 104, 467, 264]]

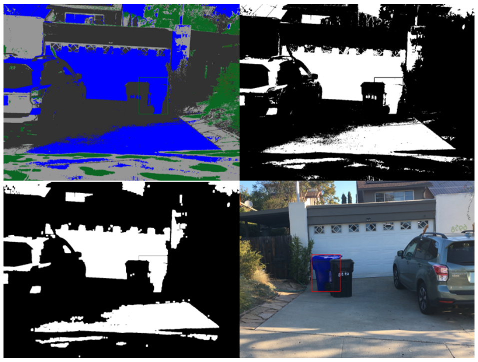

Fig. 17: 05.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: []
actual box: [[762, 416, 924, 622]]

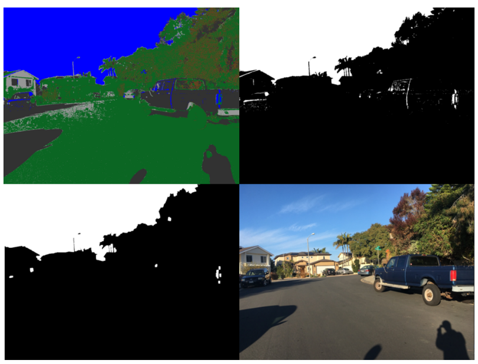

Fig. 18: 06.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: []
actual box: []

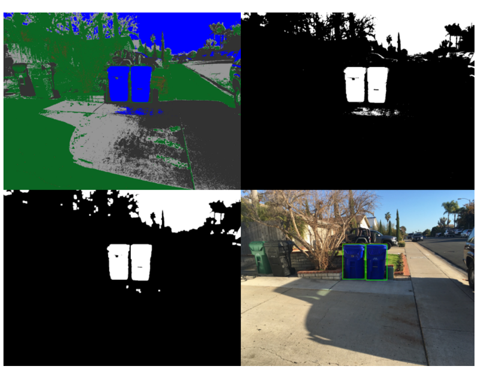

Fig. 19: 07.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: [[712, 306, 830, 509], [579, 306, 706, 504]]
actual box: [[578, 305, 706, 504], [711, 305, 830, 509]]

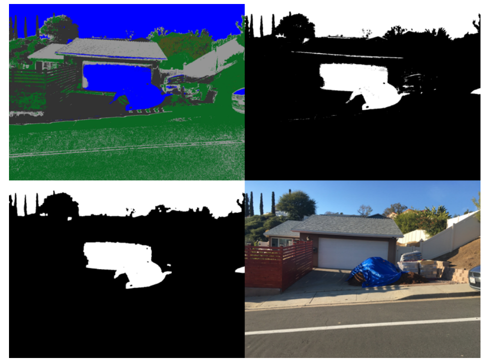

Fig. 20: 08.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: []
actual box: []

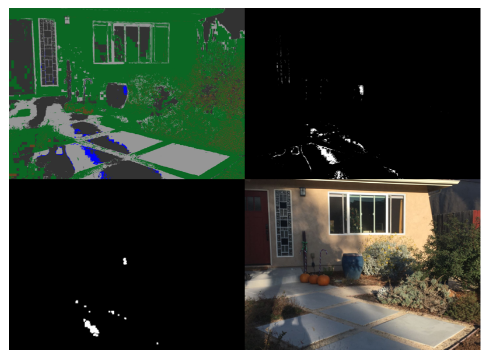

Fig. 21: 09.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: []
actual box: []

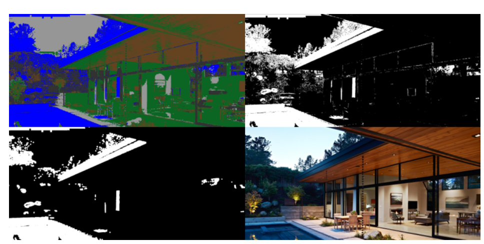

Fig. 22: 10.jpg, Segmentation, Mask, Filter, Bounding box, without sky blue class
estimated boxes: []
actual box: []