Declutterization

Nathan Vance

Overview of the Declutterizer

1. Generate cluttered images using the Clutterizer
2. Train an SVM using clutter images
3. Use the SVM to identify clutter in a novel clutter image
4. Merge adjacent bounding boxes with the same label

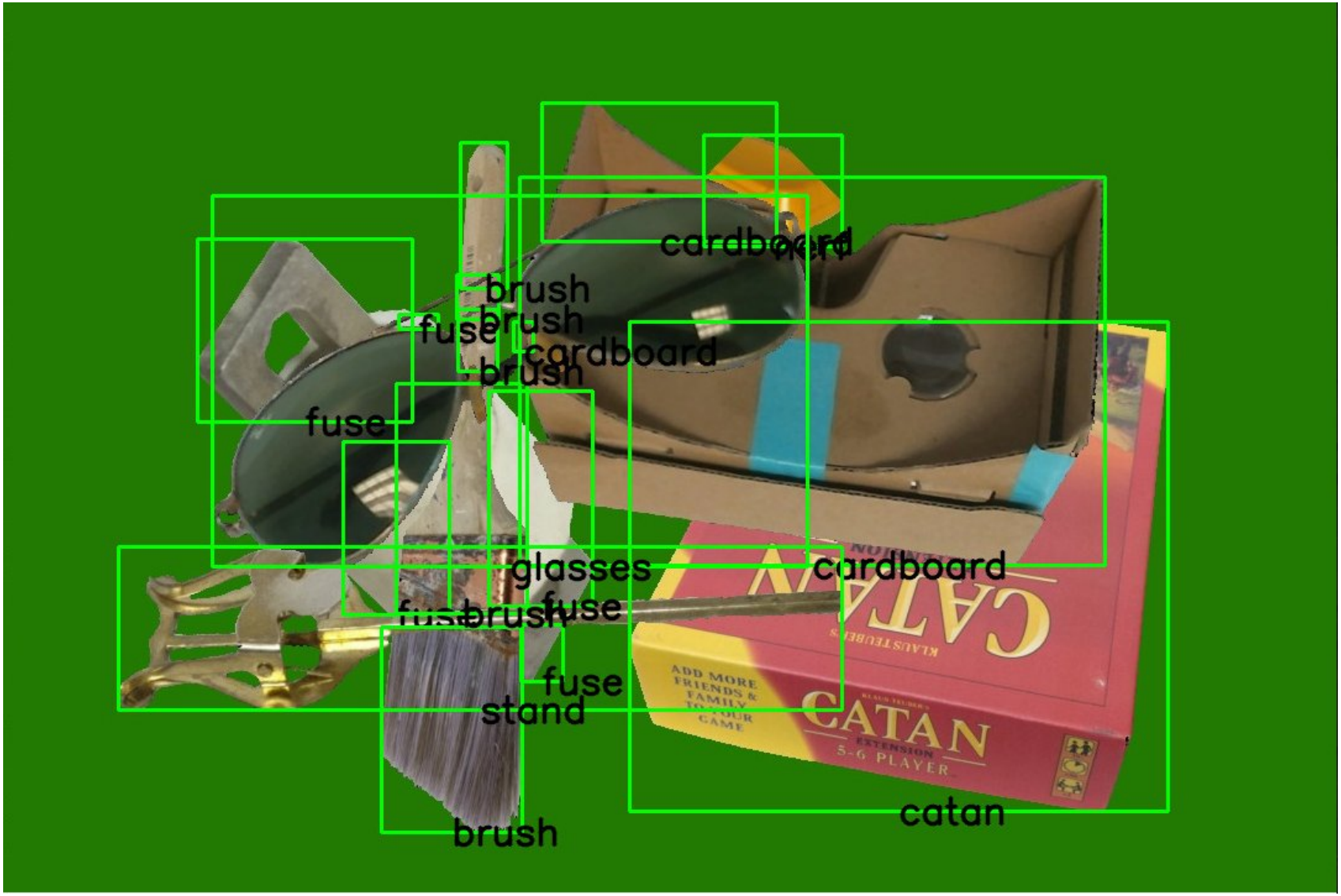
Clutterization Process

1. Images of individual objects were converted to PNGs with transparent backgrounds by hand.
2. A scene is initialized to a random color.
3. A random subset of objects are selected, resized, and rotated.
4. The modified objects are added to the scene in a random order. The algorithm keeps track of their boundaries and occlusions.
5. A labelme-formatted json file is written which corresponds to the generated clutter.

Clutterization Example

Clutterization Example (cont)

Clutterization Example (cont)



Feature Extraction

The following features are extracted:

- Normalized color histogram for the Hue channel in HSV colorspace
- Normalized LBP histograms for:
  - 24 points at a radius of 8 pixels
  - 16 points at a radius of 4 pixels
  - 12 points at a radius of 2 pixels
  - 8 points at a radius of 1 pixel (standard LBP)
- Both feature spaces are rotation invariant.

SVMs

Two models were employed in the solution:

- An SVM with an RBF kernel for classification (NuSVC in sklearn)
- A K-nearest-neighbors model for anomaly detection (LocalOutlierFactor in sklearn)

Training

The models were trained using two different sources:

- The 92 images of the 10 objects that were converted to PNG, with backgrounds masked
- 70 clutterized images (generated from the PNG images)

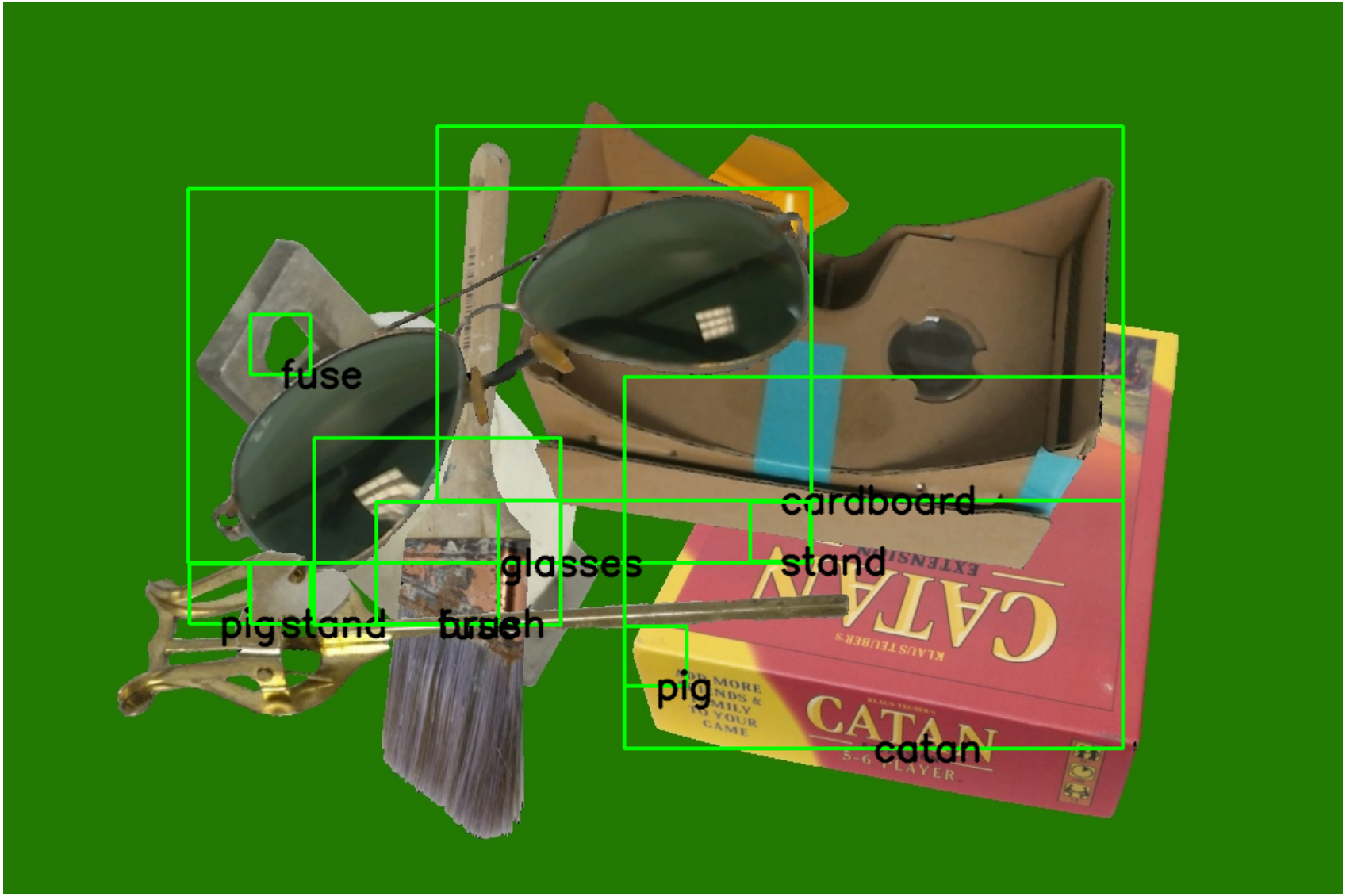
In each case, the images were traversed with a sliding window. The system was then validated on 30 clutterized images.

Additional parameters

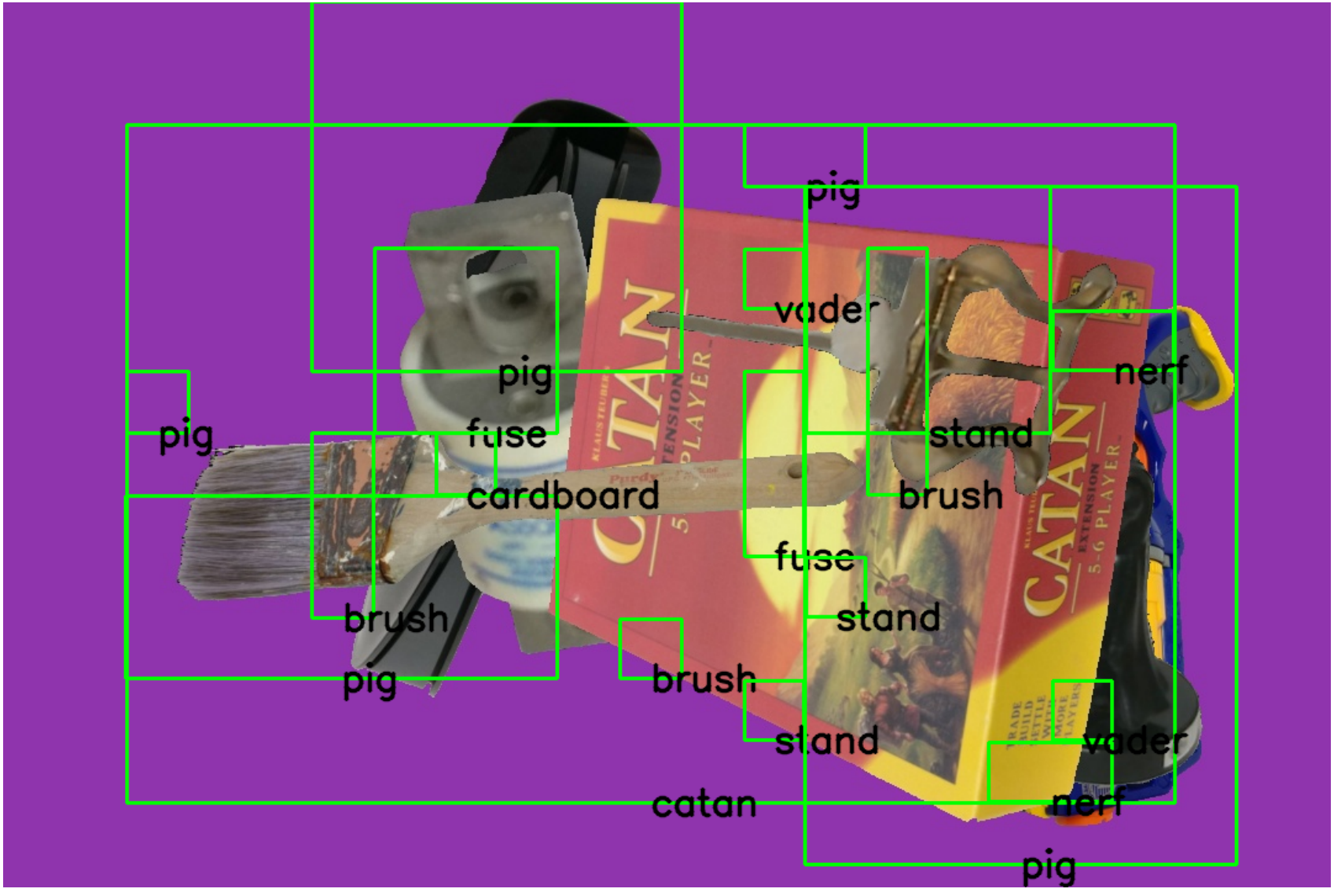
The following additional parameters were used:

- Sliding window size: 100x100 pixels
- Sliding window stride: 50 pixels
- Likelihood threshold to classify: 0.15

Validation example



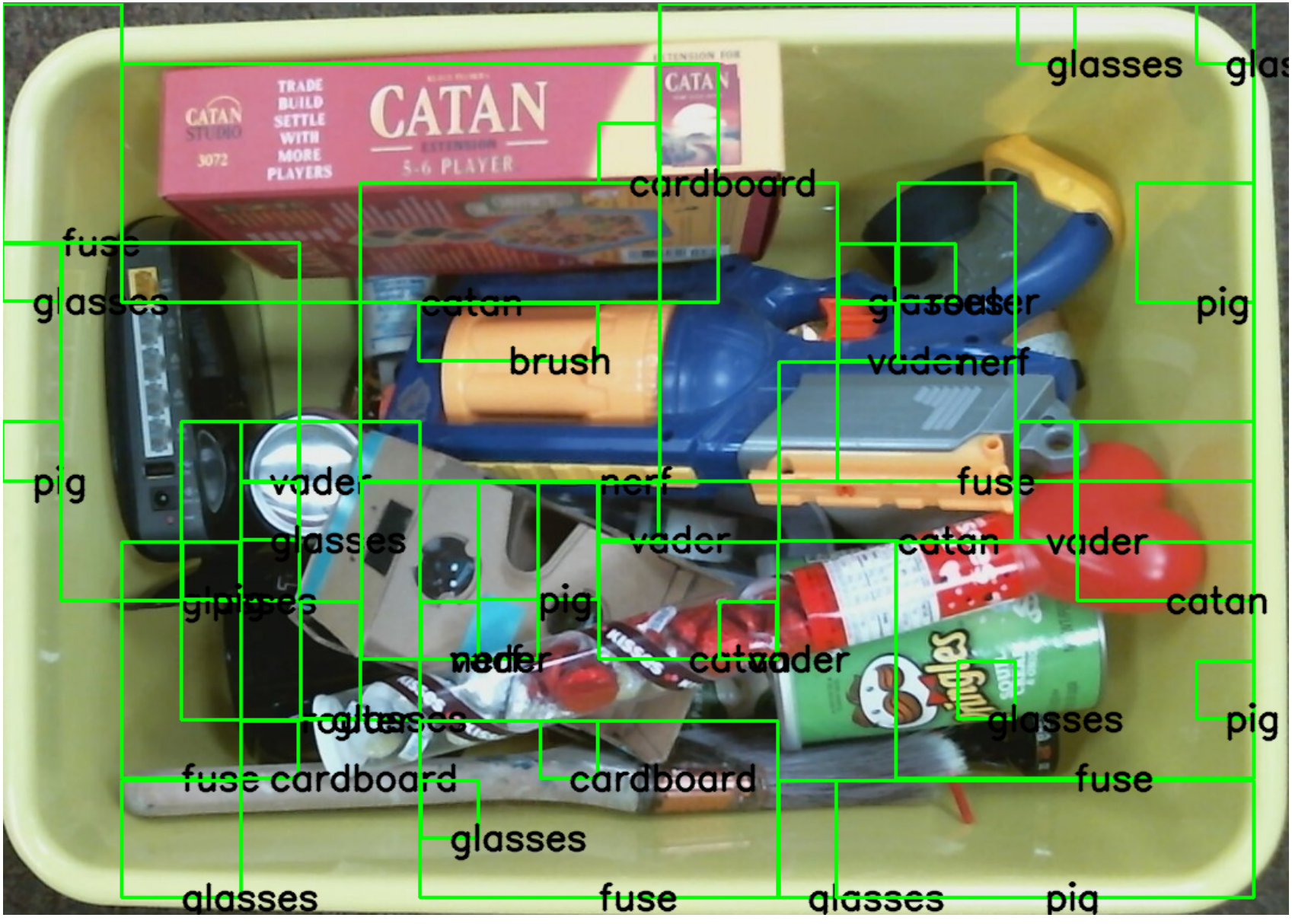
Validation example (bad)



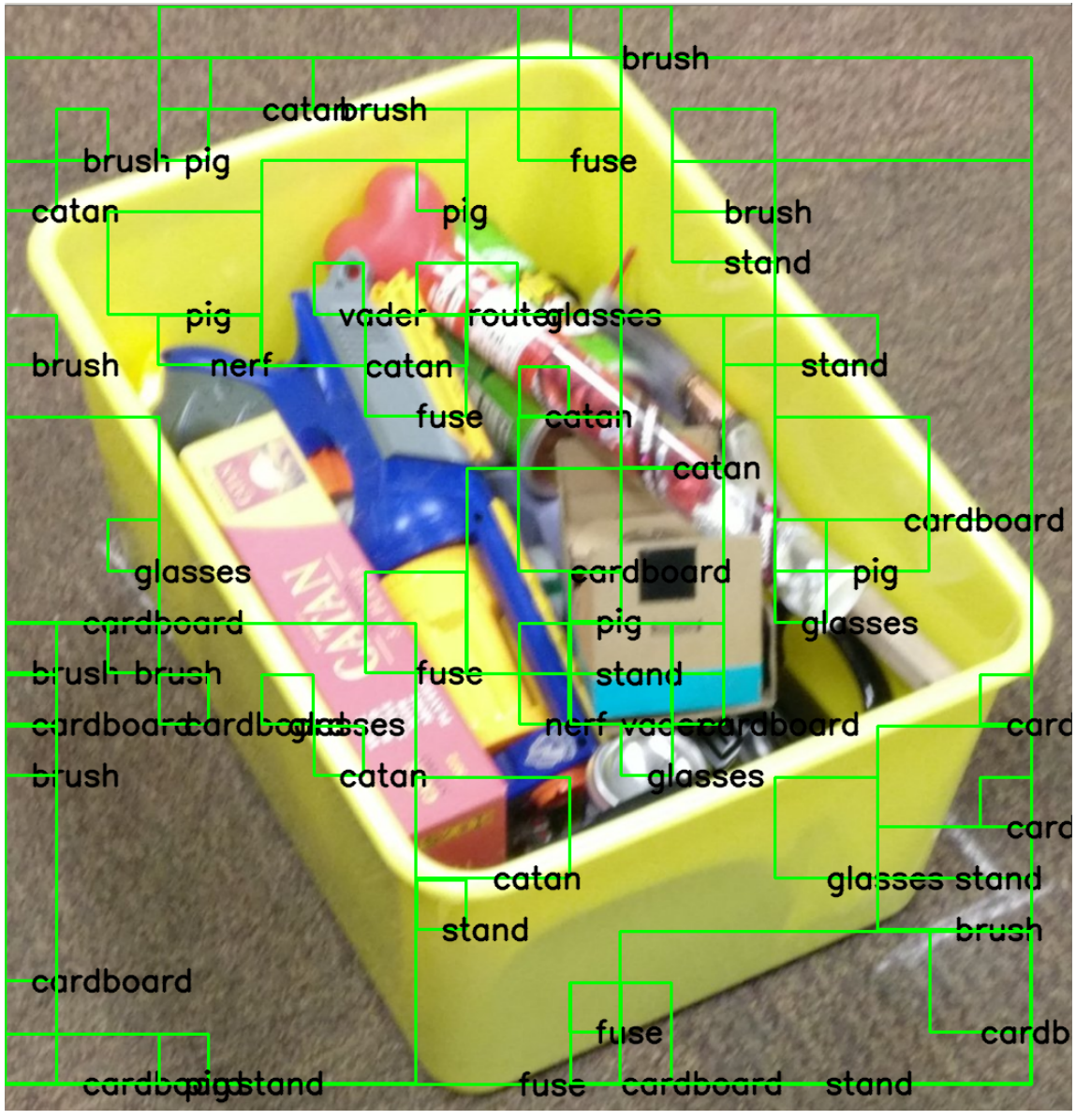
Validation results

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.008
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.028
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.003
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.021
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.030
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.040
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.040
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.003
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.070
```

Declutterization example



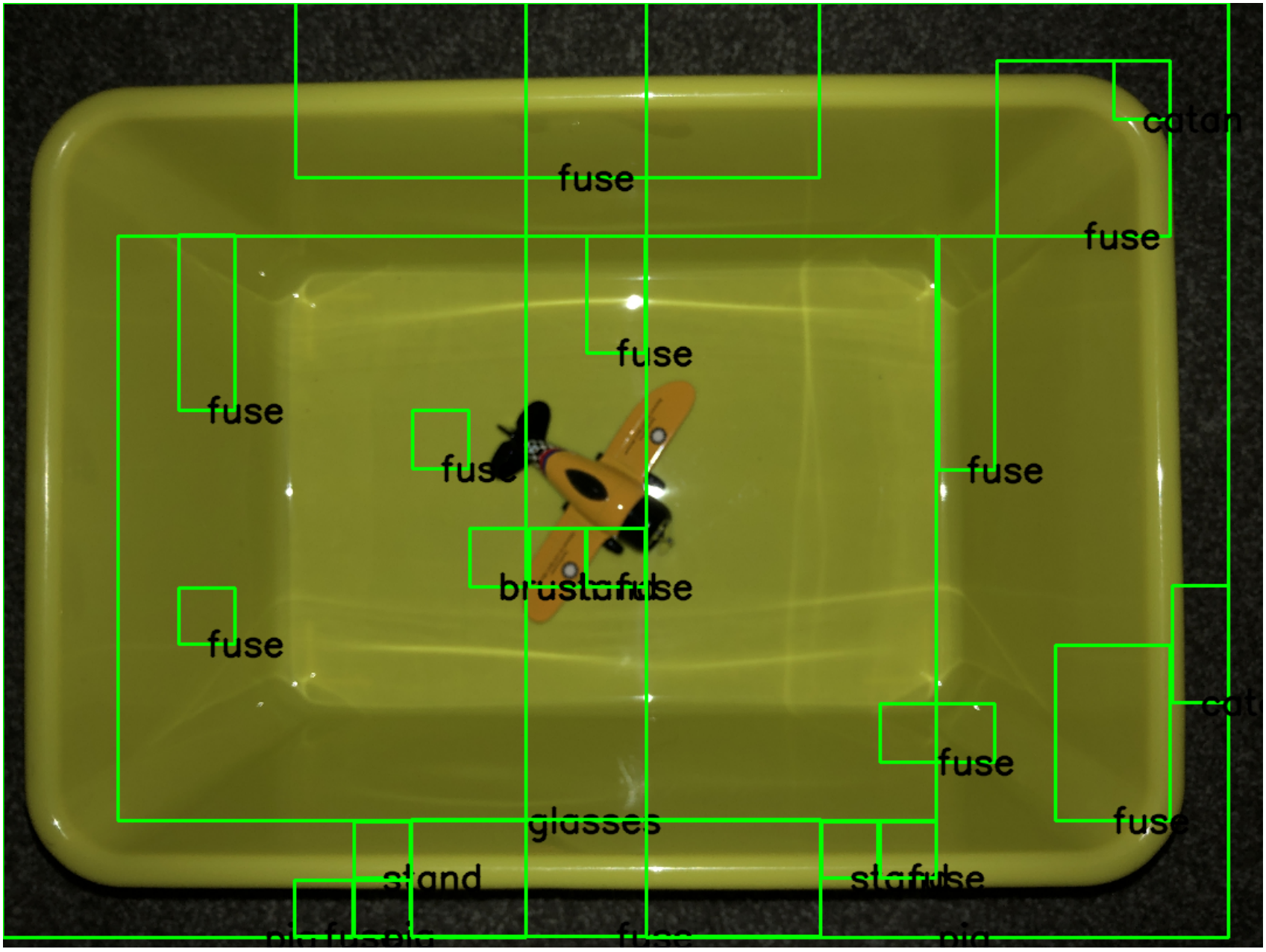
Declutterization example (bad)



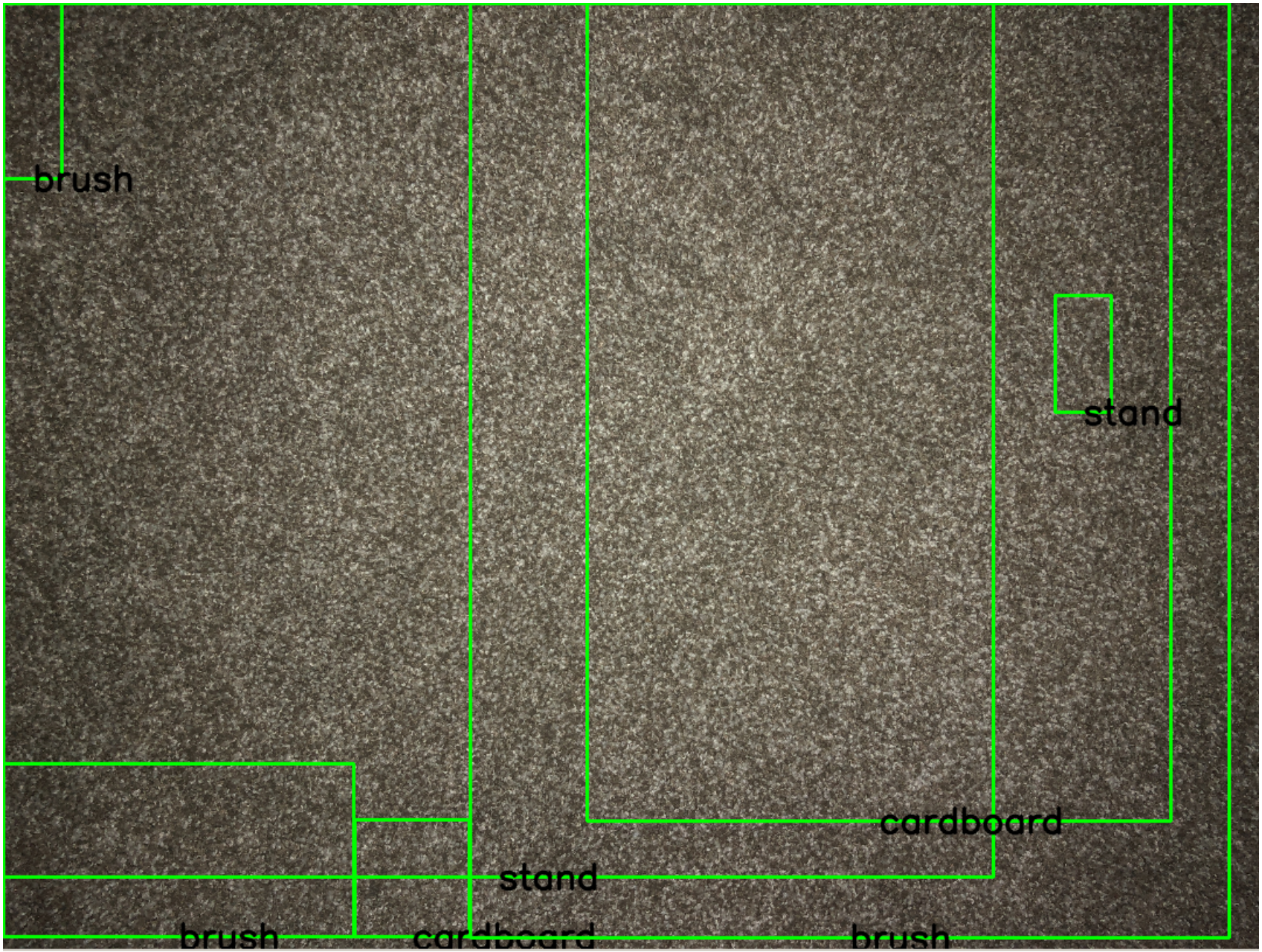
Declutterization results

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.001
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.002
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.001
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.009
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.016
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.016
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.019
```

Unknown input



Unknown input



Unknown input

Conclusions

- Qualitatively, the system does provide useful results.
- Using the official COCO scoring method, the system looks really bad.
- When tested on the unknown input, the system produces many false-positives.