Guorun Huang, Project3 Report
Pattern Recognition and Computer Vision, 25Fall
(choose to use 1 travel day to adjust the ddl)

1. Project Overview

This project implements a real-time shape-based object recognition system in C++ with OpenCV. Adaptive HSV/gray hybrid thresholding is used for robust object/background separation under varying illumination, followed by connected-component region filtering (implemented a Mat segmentRegions function manually). A feature vector is computed for each detected region, including centroid, area, orientation from central moments, oriented minimum bounding box. The system supports interactive one-shot learning — pressing ‘n’ stores the current object’s feature vector with a user-provided label into a lightweight text-based database.

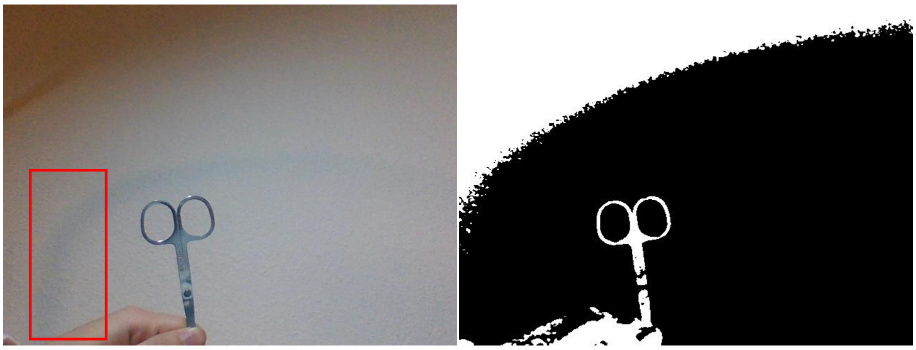
For customized extension experiments, the system is evaluated on objects such as scissors, money, sunglasses, pens, phone camera modules, and cards. It performs well on objects with distinctive geometric signatures, but struggles on color/texture-dominated items (e.g. fruits, boxes, cables). A one-shot ResNet18 embedding was also tested.

Overall, the results indicate that segmentation quality and region filtering thresholds are critical, and that shape-based descriptors are highly effective for structurally distinctive objects, while more general categories may require color/texture or deep embeddings to achieve stable recognition.

2. Results

1 Threshold

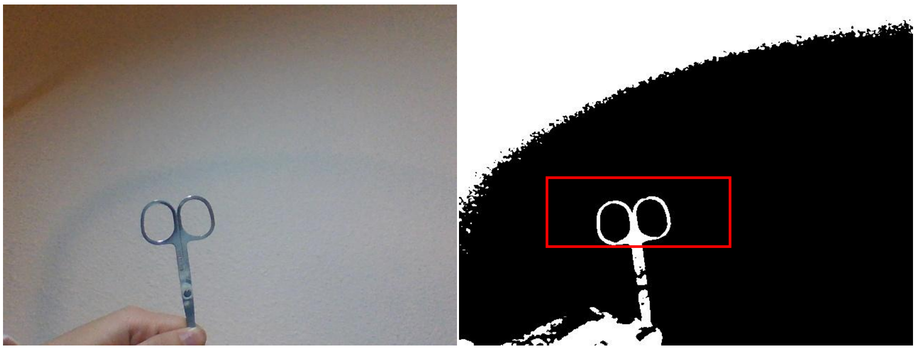
Light-colored shadows are removed.

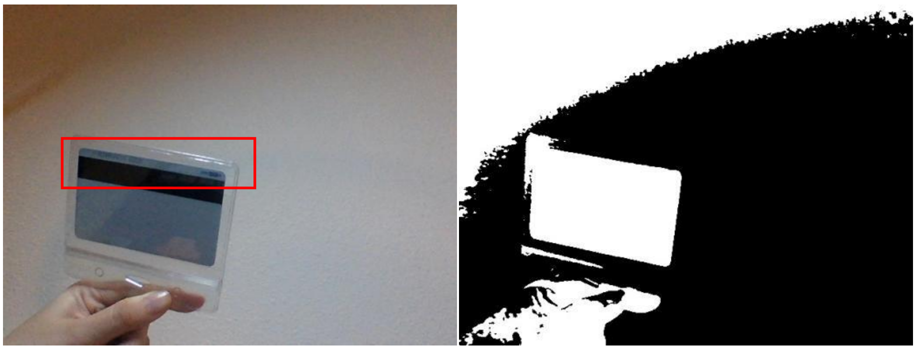


2 Binarization

Reflections can reconnect gaps caused by lighting, for example with scissors. Note the scissors still show missing parts because I filtered out pixels with grayscale values above 180 in an earlier step.

Very fine text is removed as well.

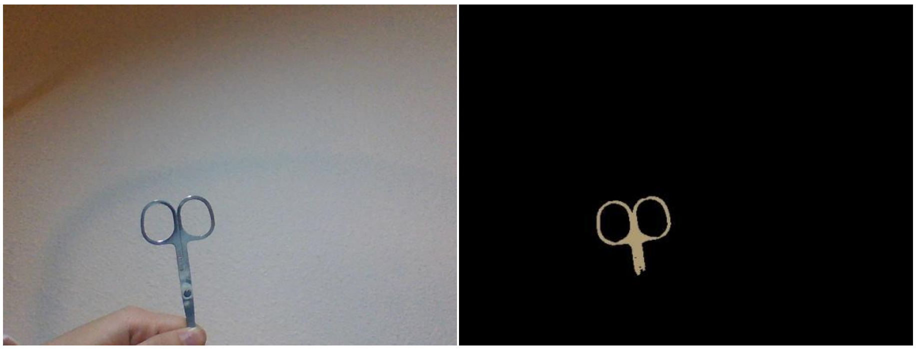


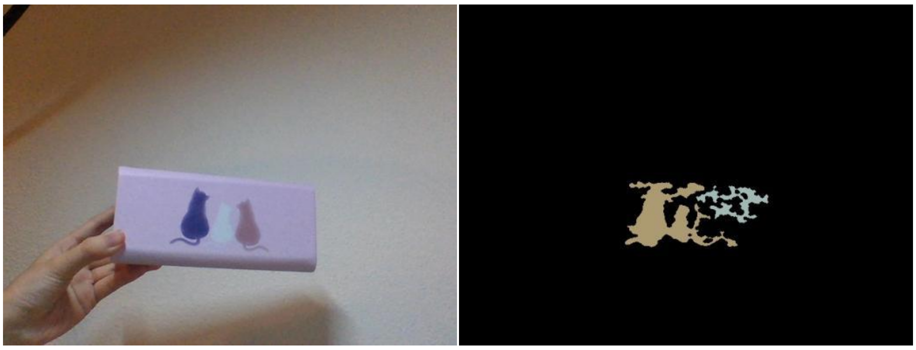


3 Region segmentation

I implemented a Mat segmentRegions function manually; it segments the image by performing two passes over the pixels.

Unevenly lit backgrounds and fingers do not affect the region segmentation at this stage.

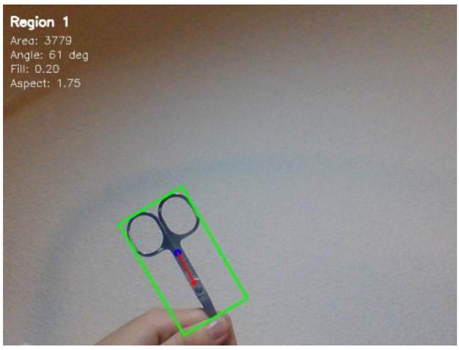




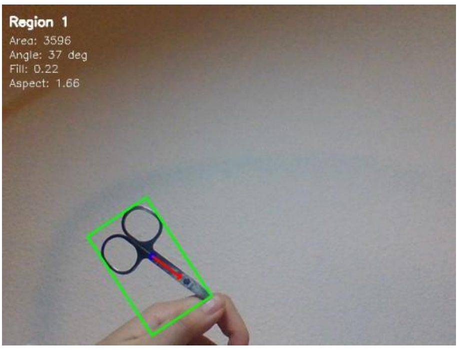
4 Computed features

Centroid, area, angle of the principal central moment, oriented minimum bounding box, percent filled (area ratio), aspect ratio (scale invariant), and Hu moments (invariant to rotation, scale, and translation).

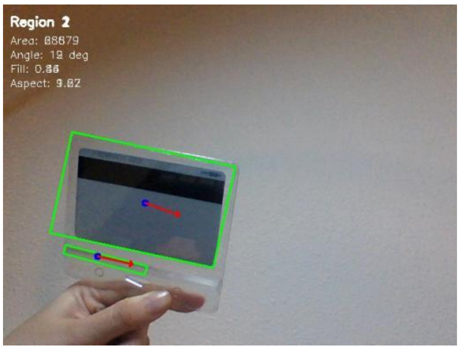
The principal axis angle and the oriented bounding box change with the object’s orientation.

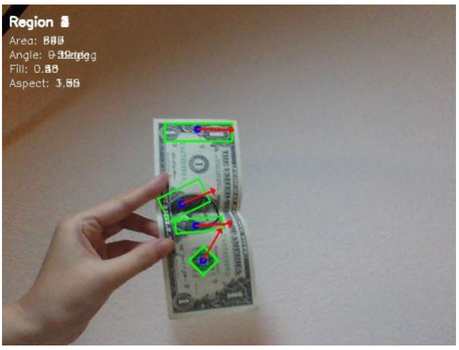


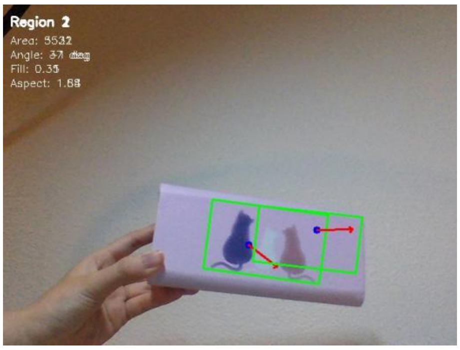
```
Region 1
Area: 3779
Angle: 61 deg
Fill: 0.20
Aspect: 1.75
```



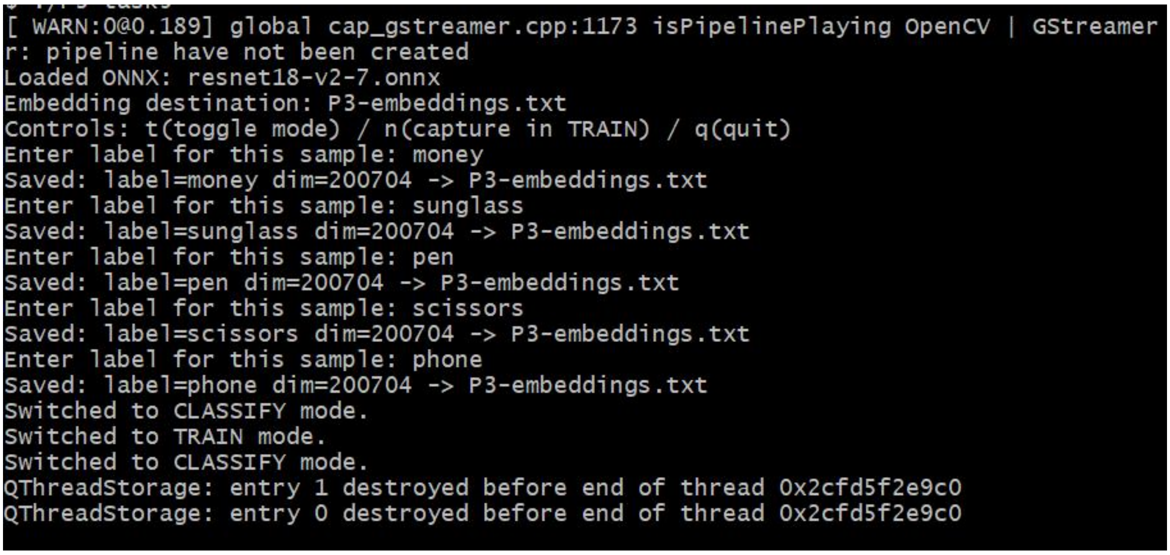
```
Region 1
Area: 3596
Angle: 37 deg
Fill: 0.22
Aspect: 1.66
```







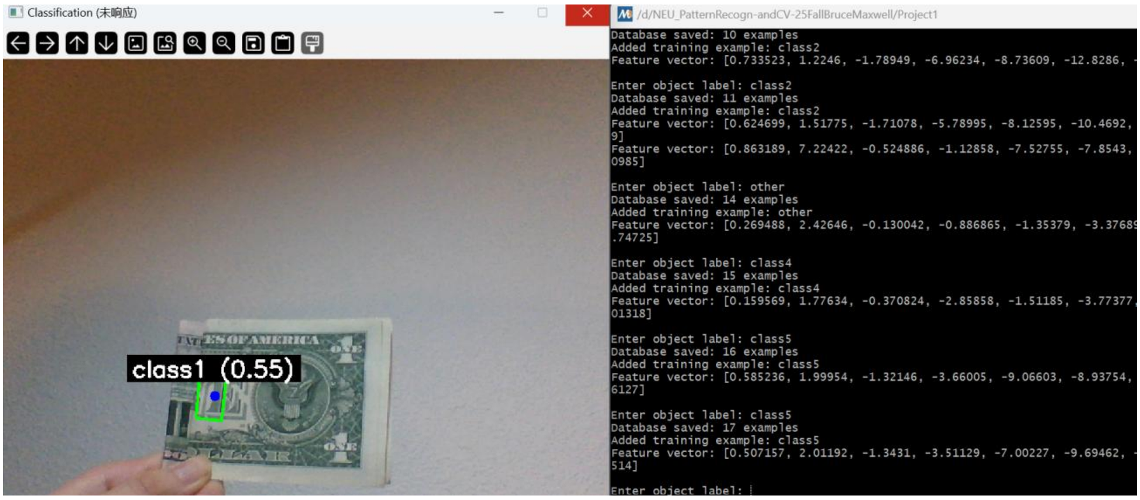
5 Interaction and labeling workflow In the default training mode the system detects objects and displays their features in the image window. While watching the window, press “n” to save the current frame; then type a label in the msys terminal and the result will be appended to a text file. The system then resumes capturing objects with the camera.



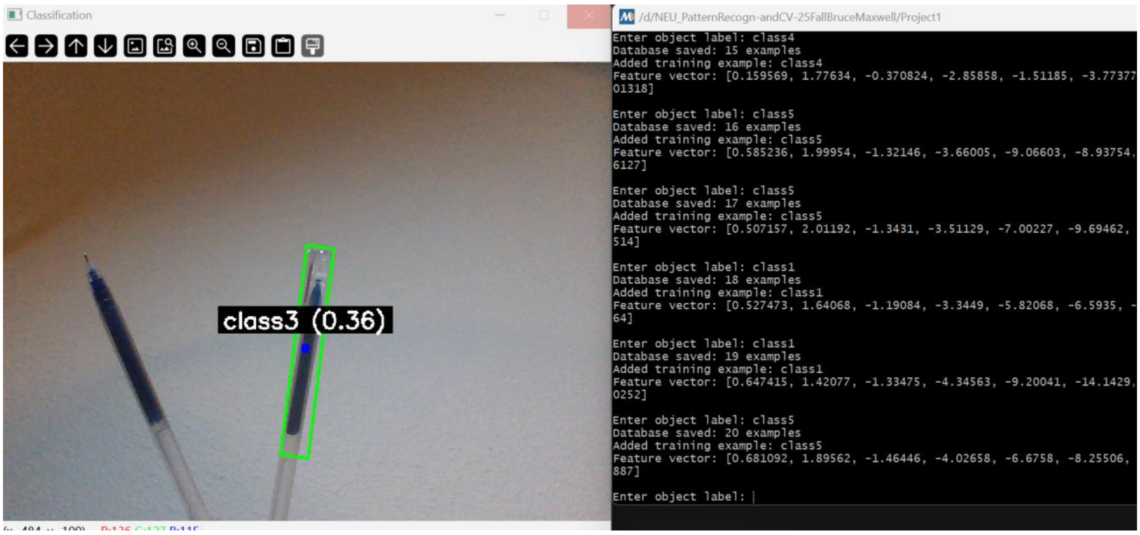
```
[ WARN:0@0.189] global cap_gstreamer.cpp:1173 isPipelinePlaying OpenCV | GStreame
r: pipeline have not been created
Loaded ONNX: resnet18-v2-7.onnx
Embedding destination: P3-embeddings.txt
Controls: t(toggle mode) / n(capture in TRAIN) / q(quit)
Enter label for this sample: money
Saved: label=money dim=200704 -> P3-embeddings.txt
Enter label for this sample: sunglass
Saved: label=sunglass dim=200704 -> P3-embeddings.txt
Enter label for this sample: pen
Saved: label=pen dim=200704 -> P3-embeddings.txt
Enter label for this sample: scissors
Saved: label=scissors dim=200704 -> P3-embeddings.txt
Enter label for this sample: phone
Saved: label=phone dim=200704 -> P3-embeddings.txt
Switched to CLASSIFY mode.
Switched to TRAIN mode.
Switched to CLASSIFY mode.
QThreadStorage: entry 1 destroyed before end of thread 0x2cfd5f2e9c0
QThreadStorage: entry 0 destroyed before end of thread 0x2cfd5f2e9c0
```

6 Classification results

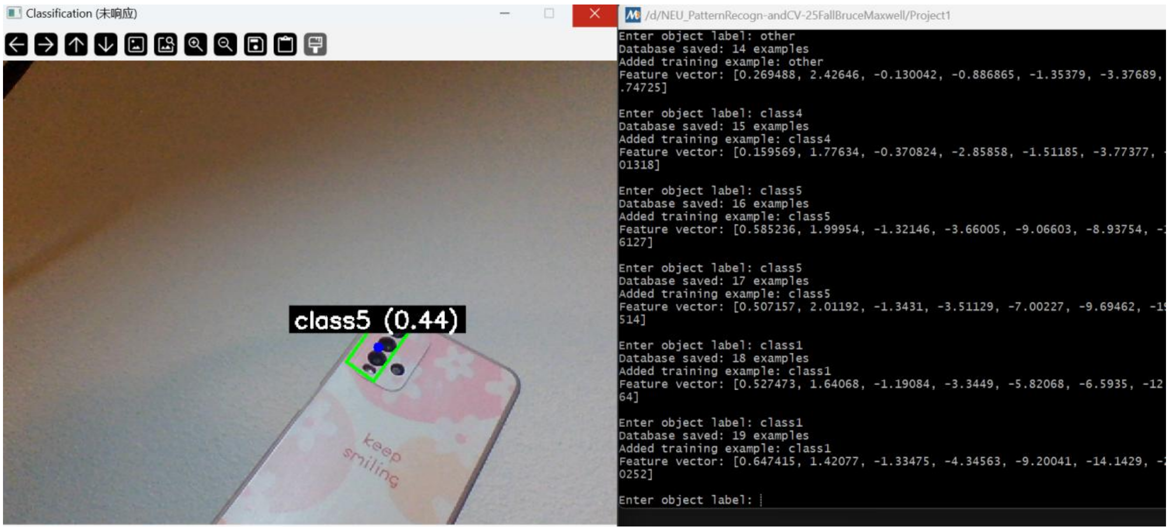
Accuracy is acceptable, but the system sometimes misclassifies objects with similar shapes and principal axes. For example, sunglasses (class 2) were partially detected and misclassified as money (class 1).



```
Database saved: 10 examples
Added training example: class2
Feature vector: [0.733523, 1.2246, -1.78949, -6.96234, -8.73609, -12.8286, -

Enter object label: class2
Database saved: 11 examples
Added training example: class2
Feature vector: [0.624699, 1.51775, -1.71078, -5.78995, -8.12595, -10.4692,
9]
Feature vector: [0.863189, 7.22422, -0.524886, -1.12858, -7.52755, -7.8543,
0985]

Enter object label: other
Database saved: 14 examples
Added training example: other
Feature vector: [0.269488, 2.42646, -0.130042, -0.886865, -1.35379, -3.3768
.74725]

Enter object label: class4
Database saved: 15 examples
Added training example: class4
Feature vector: [0.159569, 1.77634, -0.370824, -2.85858, -1.51185, -3.77377,
01318]

Enter object label: class5
Database saved: 16 examples
Added training example: class5
Feature vector: [0.585236, 1.99954, -1.32146, -3.66005, -9.06603, -8.93754,
6127]

Enter object label: class5
Database saved: 17 examples
Added training example: class5
Feature vector: [0.507157, 2.01192, -1.3431, -3.51129, -7.00227, -9.69462, -
514]

Enter object label:
```



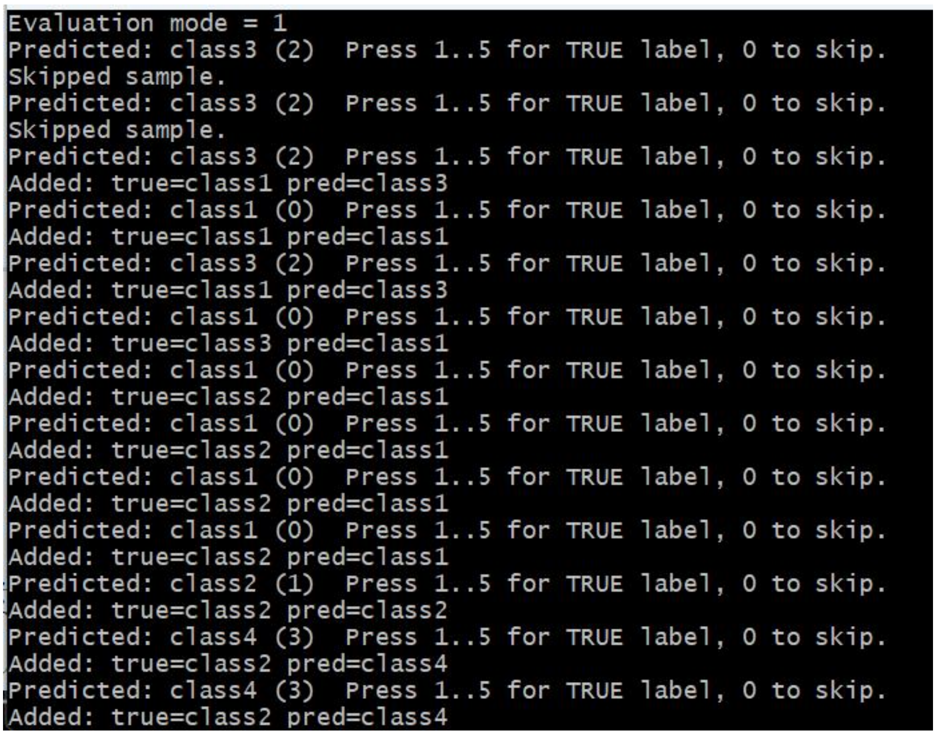
```
Enter object label: class4
Database saved: 15 examples
Added training example: class4
Feature vector: [0.159569, 1.77634, -0.370824, -2.85858, -1.51185, -3.77377
01318]

Enter object label: class5
Database saved: 16 examples
Added training example: class5
Feature vector: [0.585236, 1.99954, -1.32146, -3.66005, -9.06603, -8.93754,
6127]

Enter object label: class5
Database saved: 17 examples
Added training example: class5
Feature vector: [0.507157, 2.01192, -1.3431, -3.51129, -7.00227, -9.69462,
514]

Enter object label: class1
Database saved: 18 examples
Added training example: class1
Feature vector: [0.527473, 1.64068, -1.19084, -3.3449, -5.82068, -6.5935, -
64]

Enter object label: class1
Database saved: 19 examples
Added training example: class1
Feature vector: [0.647415, 1.42077, -1.33475, -4.34563, -9.20041, -14.1429,
0252]

Enter object label: class5
Database saved: 20 examples
Added training example: class5
Feature vector: [0.681092, 1.89562, -1.46446, -4.02658, -6.6758, -8.25506,
887]

Enter object label:
```



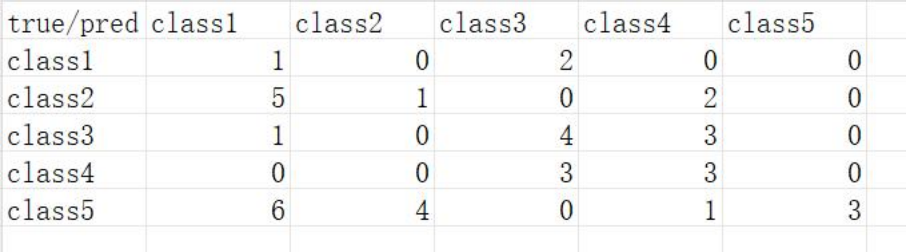
```
Enter object label: other
Database saved: 14 examples
Added training example: other
Feature vector: [0.269488, 2.42646, -0.130042, -0.886865, -1.35379, -3.37689,
.74725]

Enter object label: class4
Database saved: 15 examples
Added training example: class4
Feature vector: [0.159569, 1.77634, -0.370824, -2.85858, -1.51185, -3.77377, -
01318]

Enter object label: class5
Database saved: 16 examples
Added training example: class5
Feature vector: [0.585236, 1.99954, -1.32146, -3.66005, -9.06603, -8.93754, -1
6127]

Enter object label: class5
Database saved: 17 examples
Added training example: class5
Feature vector: [0.507157, 2.01192, -1.3431, -3.51129, -7.00227, -9.69462, -1
514]

Enter object label: class1
Database saved: 18 examples
Added training example: class1
Feature vector: [0.527473, 1.64068, -1.19084, -3.3449, -5.82068, -6.5935, -12
64]

Enter object label: class1
Database saved: 19 examples
Added training example: class1
Feature vector: [0.647415, 1.42077, -1.33475, -4.34563, -9.20041, -14.1429, -
0252]

Enter object label:
```

7 Performance Evaluation

Results and reflection: accuracy is reasonable but confusions occur when shape and axis are similar.



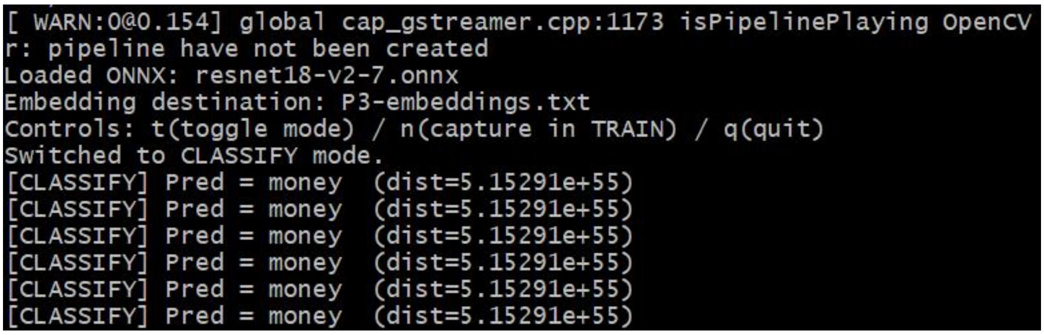
```
Evaluation mode = 1
Predicted: class3 (2)  Press 1..5 for TRUE label, 0 to skip.
Skipped sample.
Predicted: class3 (2)  Press 1..5 for TRUE label, 0 to skip.
Skipped sample.
Predicted: class3 (2)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class1 pred=class3
Predicted: class1 (0)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class1 pred=class1
Predicted: class3 (2)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class1 pred=class3
Predicted: class1 (0)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class3 pred=class1
Predicted: class1 (0)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class2 pred=class1
Predicted: class1 (0)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class2 pred=class1
Predicted: class1 (0)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class2 pred=class1
Predicted: class1 (0)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class2 pred=class1
Predicted: class2 (1)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class2 pred=class2
Predicted: class4 (3)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class2 pred=class4
Predicted: class4 (3)  Press 1..5 for TRUE label, 0 to skip.
Added: true=class2 pred=class4
```



true/pred	class1	class2	class3	class4	class5
class1	1	0	2	0	0
class2	5	1	0	2	0
class3	1	0	4	3	0
class4	0	0	3	3	0
class5	6	4	0	1	3

8 Extention
https://drive.google.com/file/d/1GN4SAtCPGYRyjZ0OQe8UH3L3M7cpLV-4/view?usp=sharing

9 One-shot CNN training Training results were normal.
Possible causes:
(1) Important features ResNet relies on (color, texture) may have been lost during image preprocessing.
(2) Training samples have unusual angles or lighting; for the same object, variations in lighting and viewpoint produce embeddings that differ more than embeddings of different but visually similar objects.



```
[ WARN:0@0.154] global cap_gstreamer.cpp:1173 isPipelinePlaying OpenCV
r: pipeline have not been created
Loaded ONNX: resnet18-v2-7.onnx
Embedding destination: P3-embeddings.txt
Controls: t(toggle mode) / n(capture in TRAIN) / q(quit)
Switched to CLASSIFY mode.
[CLASSIFY] Pred = money  (dist=5.15291e+55)
[CLASSIFY] Pred = money  (dist=5.15291e+55)
[CLASSIFY] Pred = money  (dist=5.15291e+55)
[CLASSIFY] Pred = money  (dist=5.15291e+55)
[CLASSIFY] Pred = money  (dist=5.15291e+55)
[CLASSIFY] Pred = money  (dist=5.15291e+55)
```

10 Extension — other objects tested Recognizable: scissors, money, sunglasses, pens, camera lenses (2 – 3 concentric circles), square cards. Not recognized: orange, light-blue charging cable, red soap case, avocado, metal key, silver box, red box, black-and-red coffee can.
Reflection: the first threshold step and the region-detection step are crucial. Proper parameter tuning is required to accurately identify object regions.

3. Reflection

- Thresholding: using grayscale-only thresholding performed poorly. Under a warm white background many colored objects were not detected (for example cyan toothpaste, pale red liquid soap, light purple phone cases). These items have distinctive shapes well suited to morphological classification, but grayscale thresholding missed them. Switching to HSV or combined HSV+grayscale thresholds and using neutral white lighting produced a clear improvement.
- Cleanup step: I found that skipping the cleanup step actually improved region detection. The background was fairly uniform and objects had little removable noise. For example, the small loops on scissors are prone to partial removal during thresholding because of metal reflection; applying the cleanup step caused those loops to disappear more often.
- Connected components filtering: I initially ignored regions smaller than 500 pixels to suppress noise, but this threshold proved too strict so I relaxed it to 300 pixels. I also avoided treating regions that touch the image border as background, because when I hold objects up for the camera my hand often connects the object to the border. These adjustments reduce missed detections.
- Feature storage: normalization is important when saving features.
- Classification stability: classification often fluctuates. Shape-based recognition works best for objects with distinctive silhouettes, for example glasses, large keys, stamped money, phone cases with 2 – 3 identical camera holes, scissors, and hammers. For objects with subtle shape differences or variable appearance, the model is less stable.

4. Acknowledgements

Thanks to Professor Maxwell (the provided ResNet18 onnx were essential for the deep-feature experiments) and the TAs for their guidance and support. I also referred to the official OpenCV documentation, Stack Overflow discussions, and various online tutorials throughout the project.