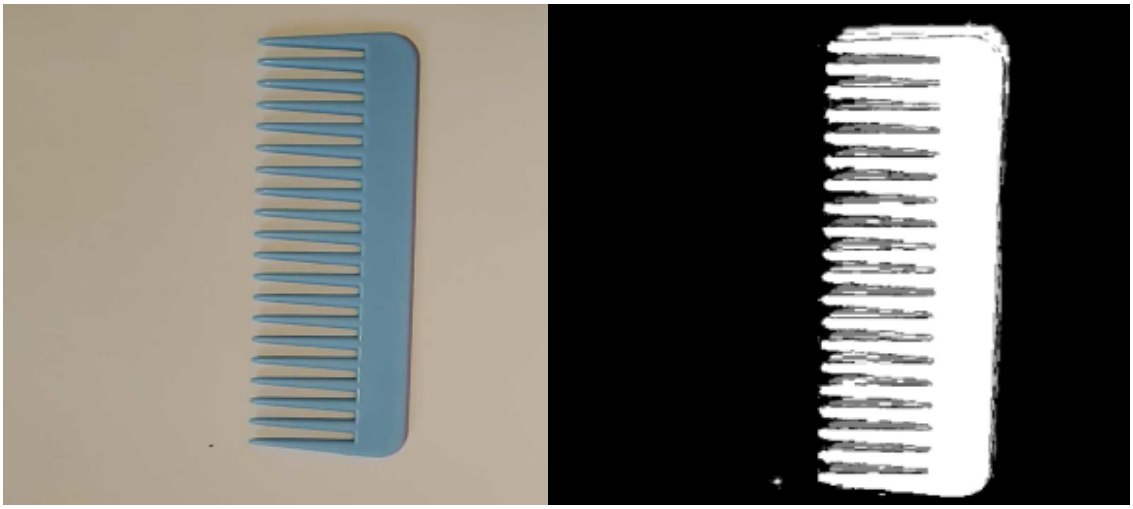
# Project 3: Real-time Object 2-D Recognition

This task is designed to teach you about 2D object recognition. The main goal of this project is to program a computer to recognize a group of items that have been placed on a white surface while maintaining translation, scaling, and rotation invariance. To do this, we carry out a number of procedures, such as thresholding, which turns the picture into a binary video sequence (for easy analysis). Following thresholding, we clean up the input video sequence's background noise and use picture segmentation to pinpoint various items in the frame. Using the features dataset, we later conduct object identification by computing the scaled Euclidian distance between the items in the film. For object recognition, we also used a different classifier (KNN).

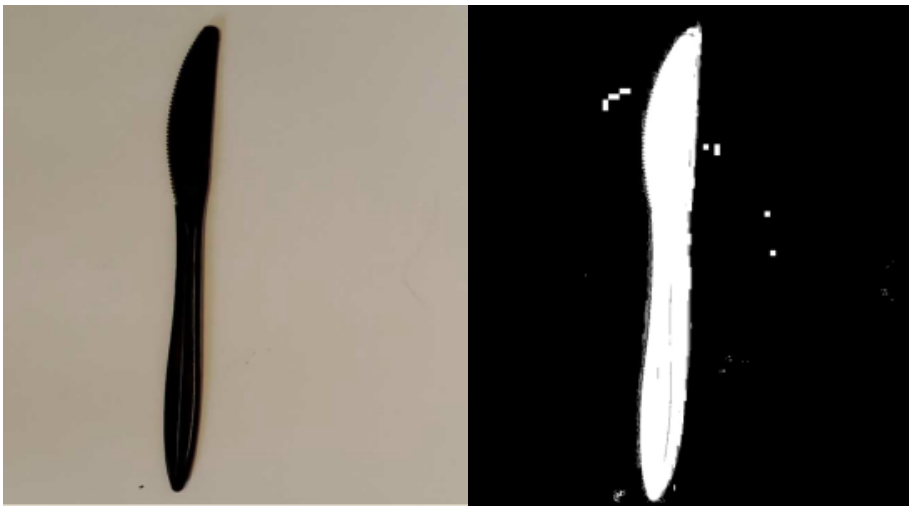
**Task 1: Threshold the input video:**

The thresholding is implemented using  background subtractor object. This method takes the current frame  as input and produces a binary foreground mask.

Example 1.



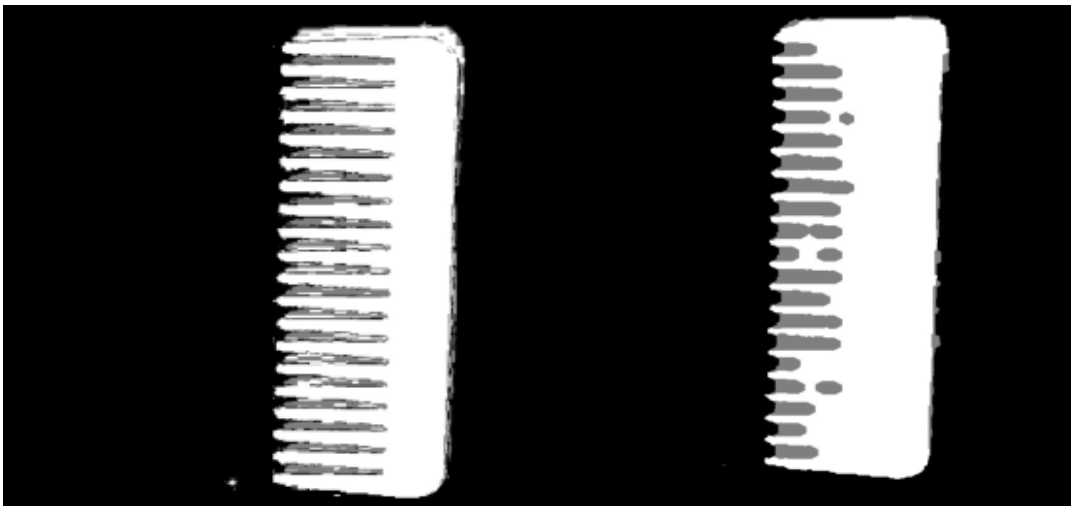
Example 2.



**Task 2: Clean up the binary image:**

After thresholding, the binary picture had some gaps and holes in the object region and salt and pepper noise in the background. I utilized growing and shrinking as my morphological filters to clean up the binary pictures.

Example1:



Example 2:

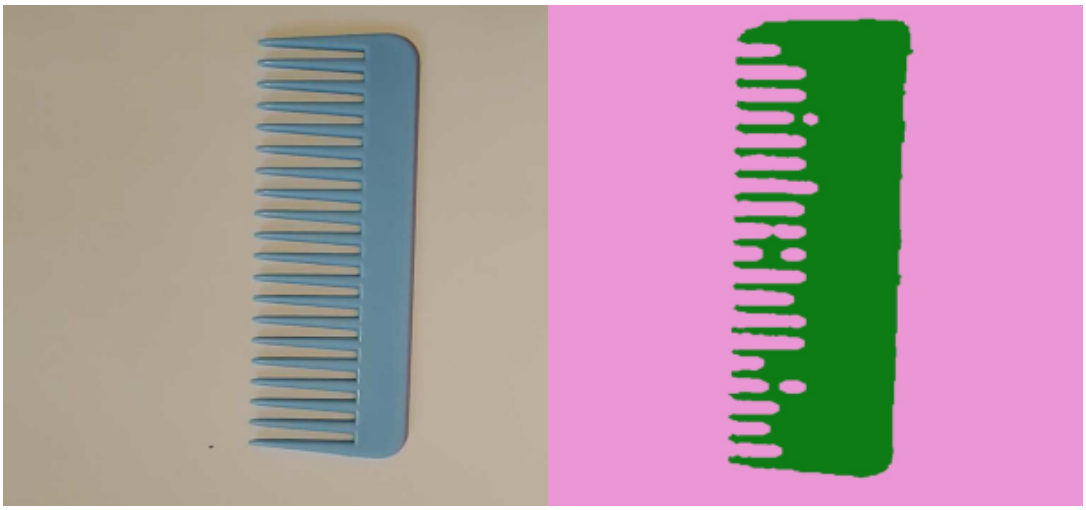
**Task 3: Segment the image into regions**

This code uses a two-pass technique to do segmentation.

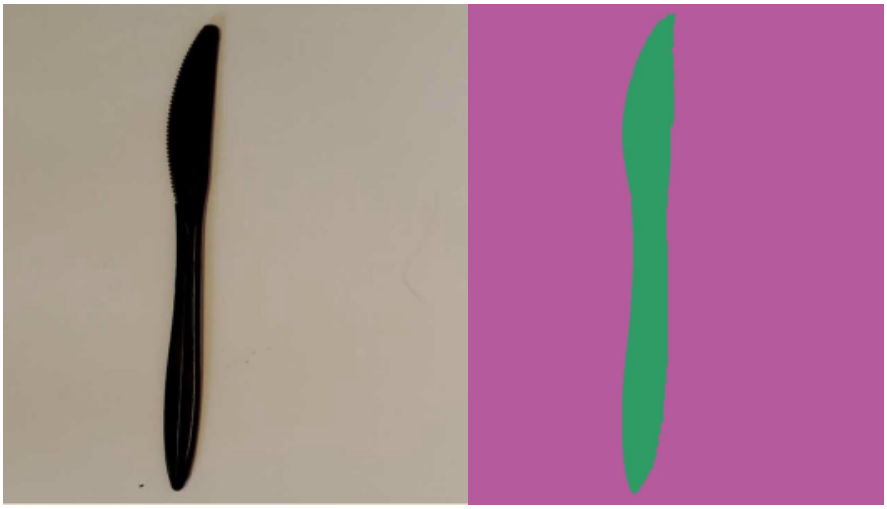
Each foreground pixel in the first pass is given a distinct integer label or, in the case of neighbors with the same label, is given that label. A new label is given to a foreground pixel if it doesn't have any labeled neighbors. This is accomplished by iteratively going over each binary picture pixel, determining whether or not it is a foreground pixel, and looking at its neighbors.

The identified pixels in the binary picture are replaced with their matching colors in the second pass after each labeled region is given a random color. To do this, a lookup table of colors for each label is created, and after that, each iteration of the binary image's pixels receives a color assignment depending on its label. The end result is a segmented picture with distinct colors for each linked area of foreground pixels.

Example 1:



Example 2:

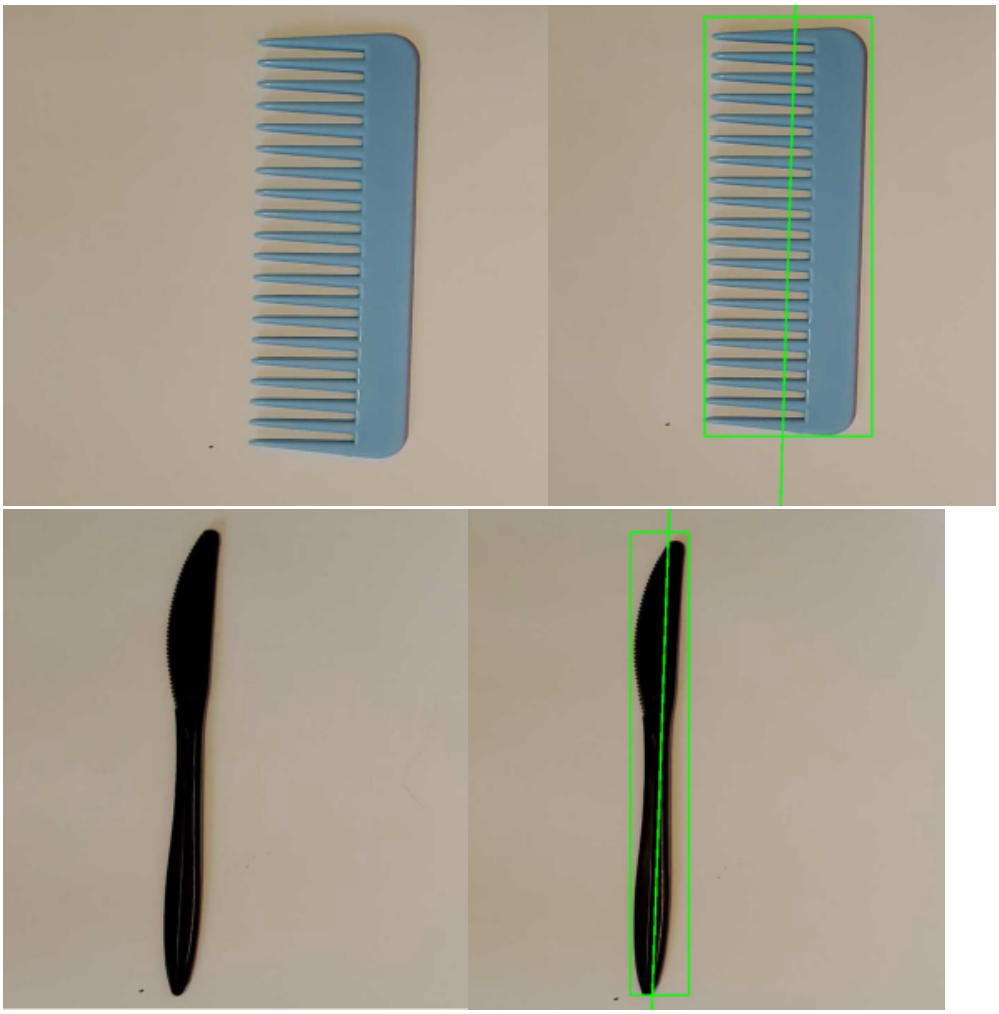


**Task 4: Compute features for each major region**

1. Using a region map and a region ID, a collection of features for a certain area were calculated.
2. To capture the object boundaries, we first locate the bounded box region and then draw an orientated rectangular box.
3. Using the HU Moment Method in OpenCV, we generated Hu Moments for the features by supplying a vector of white pixels that were present in the oriented bounding boxes of each object.
4. As a translation invariant, we determine the central moments and centroid for each object.
5. Using the HU Moment Method in OpenCV, we generated Hu Moments for the features by supplying a vector of white pixels that were present in the oriented bounding boxes of each object.
6. As a translation invariant, we determine the central moments and centroid for each object. As we want to compute moments that are invariant to translation, scaling, and rotation since central moments are insufficient for form matching, we calculate HU moments.

I computed the following features:

1. Hu moments
2. Normalized moments



**Task 5: Collect training data**

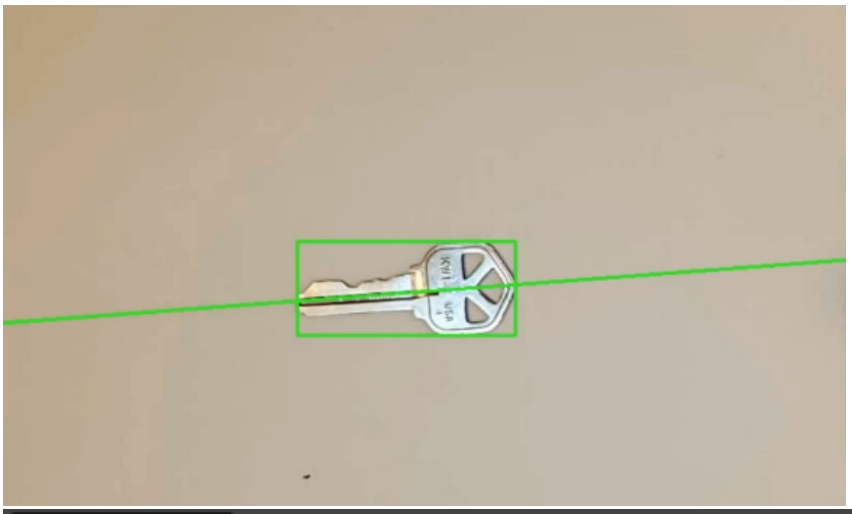
1. The image captured stores the Hu Moment feature and Normalized feature vector for each image in a csv file.
2. The csv file now acts as the training data for object classification in images.

```
Hu Moments: 0.87913 0.743605 0.00249111 0.00216446 5.02559e-06 0.00186287 -6.13804e-08
mu20_norm mu02_norm mu11_norm: 489254 207501 290017
Hu Moments: 0.89161 0.765582 0.00287536 0.00250282 6.7137e-06 0.00218665 -7.56371e-08
mu20_norm mu02_norm mu11_norm: 488873 207195 289523
Hu Moments: 0.889319 0.761489 0.00331512 0.00289965 8.98971e-06 0.00252733 -9.19984e-08
mu20_norm mu02_norm mu11_norm: 489203 206413 288886
Hu Moments: 0.889595 0.761882 0.00273723 0.00239014 6.11328e-06 0.00208537 -5.49828e-08
mu20_norm mu02_norm mu11_norm: 489048 207797 289909
Hu Moments: 0.895044 0.771601 0.00374378 0.0032882 1.15365e-05 0.0028862 -1.02776e-07
mu20_norm mu02_norm mu11_norm: 489264 207447 289565
Hu Moments: 0.898572 0.777873 0.00288218 0.00251465 6.76941e-06 0.0022149 -7.33829e-08
mu20_norm mu02_norm mu11_norm: 488770 207966 289973


Hu Moments: 0.046447 0.109091 2.54028 2.59952 5.16945 2.65465 -7.13441

mu20_norm mu02_norm mu11_norm: 5.6891 5.31799 5.46236
```

.CSV file with feature vector database.

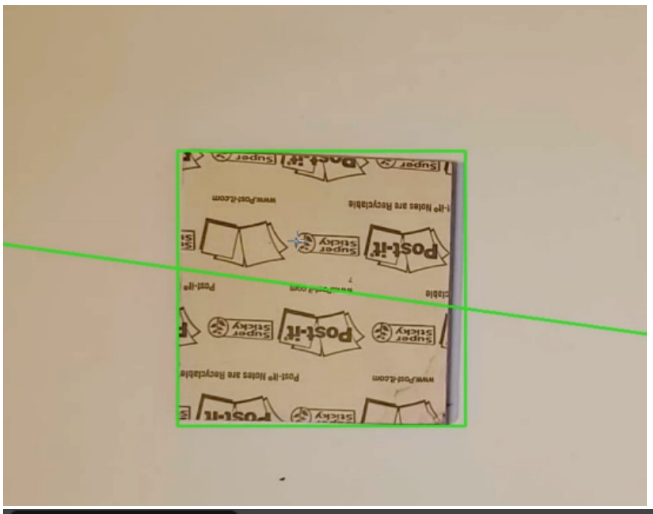
| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Lighter | 0.5391 | 1.2483 | 4.7687 | 5.3544 | 10.4731 | 6.1791 | 10.7332 | 5.6106 | 5.2082 | 5.4031 |
| Lighter | 0.5644 | 1.3285 | 4.5385 | 4.8373 | 9.5277 | 5.5017 | -10.495 | 5.7021 | 5.0359 | 5.3506 |
| mouse | 0.7775 | 2.6437 | 5.2753 | 6.8718 | -13.0096 | -8.2325 | -13.2409 | 5.7262 | 5.0126 | 5.342 |
| mouse | 0.7445 | 2.182 | 5.6829 | 6.6707 | 13.0593 | 7.9349 | 12.9502 | 5.7045 | 4.8001 | 5.2325 |
| mouse | 0.7733 | 2.5506 | 6.6941 | 8.0727 | 15.6516 | 9.6151 | -15.5693 | 5.7414 | 4.8806 | 5.2754 |
| mouse | 0.7428 | 2.1661 | 4.8357 | 6.1646 | 11.6724 | 7.2501 | -12.3962 | 5.7233 | 4.9135 | 5.3017 |
| Leaf | 0.5072 | 1.1509 | 3.1758 | 3.4326 | 6.7371 | 4.0136 | 8.0951 | 5.6396 | 4.9317 | 5.2845 |
| Leaf | 0.6045 | 1.4468 | 3.5945 | 4.0368 | 7.8525 | 4.7602 | -9.6957 | 5.7289 | 4.9411 | 5.3297 |
| Leaf | 0.5063 | 1.1475 | 3.5232 | 3.7521 | 7.3899 | 4.3278 | -8.994 | 5.7152 | 5.0088 | 5.3608 |
| Leaf | 0.6149 | 1.4823 | 3.5765 | 4.065 | 7.8903 | 4.8272 | 8.7322 | 5.7217 | 5.0289 | 5.3711 |
| Toothbrush | -0.0679 | -0.12 | 2.0106 | 2.2198 | 4.3362 | 2.1683 | -5.4723 | 5.6954 | 5.3664 | 5.4758 |
| Toothbrush | -0.1934 | -0.3767 | 1.6099 | 1.7493 | 3.429 | 1.5631 | -4.9958 | 5.6336 | 5.1588 | 5.3679 |
| Toothbrush | -0.1924 | -0.3745 | 1.4044 | 1.5296 | 2.9969 | 1.3487 | 4.4255 | 5.5888 | 5.098 | 5.3129 |
| Toothbrush | -0.0775 | -0.1384 | 1.8744 | 2.0922 | 4.0776 | 2.0387 | 5.0854 | 5.654 | 5.3751 | 5.4616 |
| Pencil | -0.1294 | -0.2521 | 2.3913 | 2.4116 | 4.8131 | 2.2864 | 7.9449 | 5.7283 | 5.2169 | 5.4219 |
| Pencil | -0.2669 | -0.5303 | 3.1167 | 3.1484 | 6.281 | 2.8846 | -8.6885 | 5.646 | 5.0878 | 5.3503 |
| Pencil | -0.2632 | -0.5229 | 2.2964 | 2.317 | 4.6236 | 2.0565 | 7.2988 | 5.6737 | 5.1057 | 5.3724 |
| comb | 0.5557 | 1.3179 | 5.081 | 5.5177 | -11.9442 | -7.0948 | -10.8183 | 5.7676 | 5.1788 | 5.423 |
| comb | 0.5666 | 1.3476 | 4.438 | 4.8754 | 9.5951 | 5.6207 | -9.8314 | 5.7072 | 5.0718 | 5.3699 |
| comb | 0.5716 | 1.3668 | 5.0236 | 5.5967 | -11.4754 | -6.6061 | 10.9233 | 5.6482 | 5.2386 | 5.3999 |
| fork | 0.2118 | 0.4642 | 1.4558 | 1.5506 | 3.0539 | 1.7835 | -5.0503 | 5.6387 | 5.259 | 5.4204 |
| fork | 0.0857 | 0.2003 | 0.7578 | 0.812 | 1.5968 | 0.9122 | -4.1577 | 5.5046 | 5.0388 | 5.2541 |
| fork | 0.0951 | 0.2199 | 0.7799 | 0.8343 | 1.6414 | 0.9443 | -4.3958 | 5.7105 | 5.1663 | 5.4281 |
| knief | -0.0759 | -0.141 | 2.2782 | 2.3404 | 4.6498 | 2.2743 | 6.7809 | 5.6842 | 5.2833 | 5.4406 |
| knief | -0.0695 | -0.1275 | 1.166 | 1.2116 | 2.4004 | 1.1498 | 4.6804 | 5.6622 | 5.2421 | 5.4085 |
| knief | -0.099 | -0.1886 | 1.4038 | 1.4498 | 2.8766 | 1.3583 | 5.0808 | 5.6441 | 5.0906 | 5.3497 |
| knief | -0.0988 | -0.1882 | 1.4051 | 1.4505 | 2.8783 | 1.3589 | -5.1184 | 5.6427 | 5.1719 | 5.3909 |

**Task 6: Classify new images**



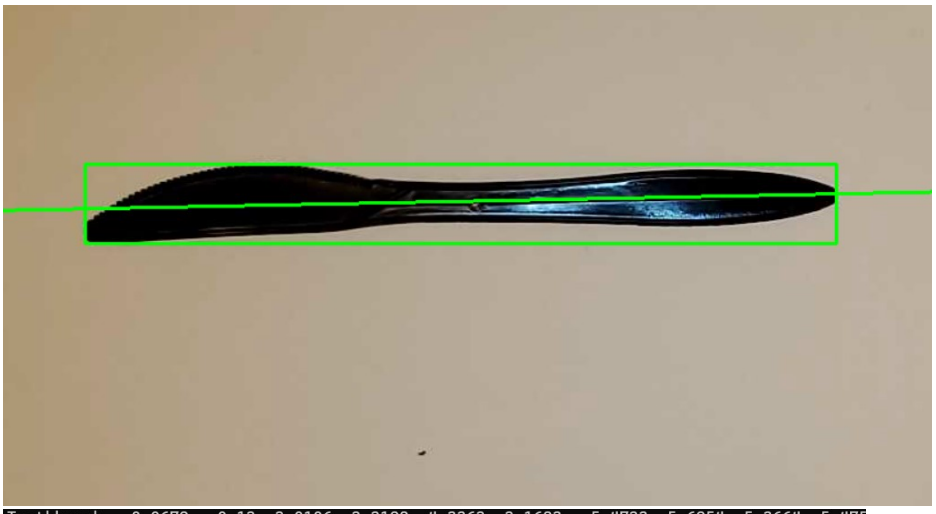
Microsoft Visual Studio Debu

```
Leaf  0.5072  1.1509  3.1758  3.4326  6.7371  4.0136  8.0951  5.6396  4.9317  5.2845
Leaf  0.6045  1.4468  3.5945  4.0368  7.8525  4.7602  -9.6957  5.7289  4.9411  5.3297
Leaf  0.5063  1.1475  3.5232  3.7521  7.3899  4.3278  -8.994  5.7152  5.0088  5.3608
Leaf  0.6149  1.4823  3.5765  4.065  7.8903  4.8272  8.7322  5.7217  5.0289  5.3711
Toothbrush  -0.0679  -0.12  2.0106  2.2198  4.3362  2.1683  -5.4723  5.6954  5.3664  5.4758
Toothbrush  -0.1934  -0.3767  1.6099  1.7493  3.429  1.5631  -4.9958  5.6336  5.1588  5.3679
Toothbrush  -0.1924  -0.3745  1.4044  1.5296  2.9969  1.3487  4.4255  5.5888  5.098  5.3129
Toothbrush  -0.0775  -0.1384  1.8744  2.0922  4.0776  2.0387  5.0854  5.654  5.3751  5.4616
Pencil  -0.1294  -0.2521  2.3913  2.4116  4.8131  2.2864  7.9449  5.7283  5.2169  5.4219
Pencil  -0.2669  -0.5303  3.1167  3.1484  6.281  2.8846  -8.6885  5.646  5.0878  5.3503
Pencil  -0.2632  -0.5229  2.2964  2.317  4.6236  2.0565  7.2988  5.6737  5.1057  5.3724
comb  0.5557  1.3179  5.081  5.5177  -11.9442  -7.0948  -10.8183  5.7676  5.1788  5.423
comb  0.5666  1.3476  4.438  4.8754  9.5951  5.6207  -9.8314  5.7072  5.0718  5.3699
comb  0.5716  1.3668  5.0236  5.5967  -11.4754  -6.6061  10.9233  5.6482  5.2386  5.3999
fork  0.2118  0.4642  1.4558  1.5506  3.0539  1.7835  -5.0503  5.6387  5.259  5.4204
fork  0.0857  0.2003  0.7578  0.812  1.5968  0.9122  -4.1577  5.5046  5.0388  5.2541
fork  0.0951  0.2199  0.7799  0.8343  1.6414  0.9443  -4.3958  5.7105  5.1663  5.4281
knief  -0.0759  -0.141  2.2782  2.3404  4.6498  2.2743  6.7809  5.6842  5.2833  5.4406
knief  -0.0695  -0.1275  1.166  1.2116  2.4004  1.1498  4.6804  5.6622  5.2421  5.4085
knief  -0.099  -0.1886  1.4038  1.4498  2.8766  1.3583  5.0808  5.6441  5.0906  5.3497
knief  -0.0988  -0.1882  1.4051  1.4505  2.8783  1.3589  -5.1184  5.6427  5.1719  5.3909
knief  -0.0077  -0.0027  2.5281  3.4149  -6.7521  -3.6065  -6.431  5.6691  5.4931  5.5657 [mjpeg @ 000002b1dd
rread 8

Top match:Key
```




Microsoft Visual Studio Debu

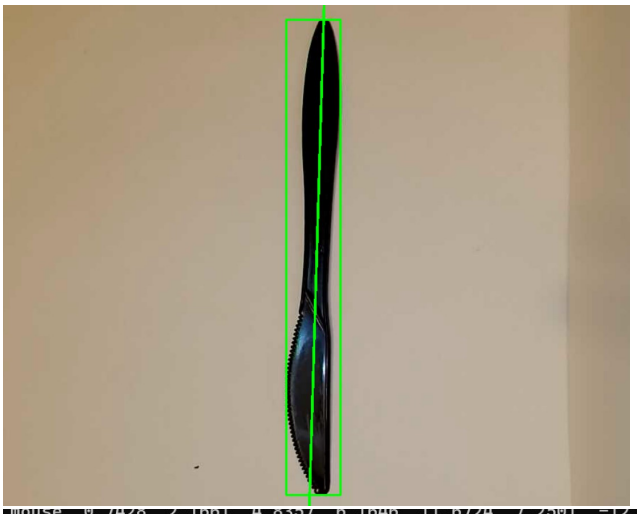
```
mouse  0.7733  2.5506  6.6941  8.0727  15.6516  9.6151  -15.5693  5.7414  4.8806  5.2754
mouse  0.7428  2.1661  4.8357  6.1646  11.6724  7.2501  -12.3962  5.7233  4.9135  5.3017
Leaf  0.5072  1.1509  3.1758  3.4326  6.7371  4.0136  8.0951  5.6396  4.9317  5.2845
Leaf  0.6045  1.4468  3.5945  4.0368  7.8525  4.7602  -9.6957  5.7289  4.9411  5.3297
Leaf  0.5063  1.1475  3.5232  3.7521  7.3899  4.3278  -8.994  5.7152  5.0088  5.3608
Leaf  0.6149  1.4823  3.5765  4.065  7.8903  4.8272  8.7322  5.7217  5.0289  5.3711
Toothbrush  -0.0679  -0.12  2.0106  2.2198  4.3362  2.1683  -5.4723  5.6954  5.3664  5.4758
Toothbrush  -0.1934  -0.3767  1.6099  1.7493  3.429  1.5631  -4.9958  5.6336  5.1588  5.3679
Toothbrush  -0.1924  -0.3745  1.4044  1.5296  2.9969  1.3487  4.4255  5.5888  5.098  5.3129
Toothbrush  -0.0775  -0.1384  1.8744  2.0922  4.0776  2.0387  5.0854  5.654  5.3751  5.4616
Pencil  -0.1294  -0.2521  2.3913  2.4116  4.8131  2.2864  7.9449  5.7283  5.2169  5.4219
Pencil  -0.2669  -0.5303  3.1167  3.1484  6.281  2.8846  -8.6885  5.646  5.0878  5.3503
Pencil  -0.2632  -0.5229  2.2964  2.317  4.6236  2.0565  7.2988  5.6737  5.1057  5.3724
comb  0.5557  1.3179  5.081  5.5177  -11.9442  -7.0948  -10.8183  5.7676  5.1788  5.423
comb  0.5666  1.3476  4.438  4.8754  9.5951  5.6207  -9.8314  5.7072  5.0718  5.3699
comb  0.5716  1.3668  5.0236  5.5967  -11.4754  -6.6061  10.9233  5.6482  5.2386  5.3999
fork  0.2118  0.4642  1.4558  1.5506  3.0539  1.7835  -5.0503  5.6387  5.259  5.4204
fork  0.0857  0.2003  0.7578  0.812  1.5968  0.9122  -4.1577  5.5046  5.0388  5.2541
fork  0.0951  0.2199  0.7799  0.8343  1.6414  0.9443  -4.3958  5.7105  5.1663  5.4281
knief  -0.0759  -0.141  2.2782  2.3404  4.6498  2.2743  6.7809  5.6842  5.2833  5.4406
knief  -0.0695  -0.1275  1.166  1.2116  2.4004  1.1498  4.6804  5.6622  5.2421  5.4085
knief  -0.099  -0.1886  1.4038  1.4498  2.8766  1.3583  5.0808  5.6441  5.0906  5.3497
knief  -0.0988  -0.1882  1.4051  1.4505  2.8783  1.3589  -5.1184  5.6427  5.1719  5.3909
knief  -0.0077  -0.0027  2.5281  3.4149  -6.7521  -3.6065  -6.431  5.6691  5.4931  5.5657
Top match:Post-it
```



```
Toothbrush  -0.0679  -0.12  2.0106  2.2198  4.3362  2.1683  -5.4723  5.6954  5.3664  5.475
Toothbrush  -0.1934  -0.3767  1.6099  1.7493  3.429  1.5631  -4.9958  5.6336  5.1588  5.36
Toothbrush  -0.1924  -0.3745  1.4044  1.5296  2.9969  1.3487  4.4255  5.5888  5.098  5.312
Toothbrush  -0.0775  -0.1384  1.8744  2.0922  4.0776  2.0387  5.0854  5.654  5.3751  5.461
Pencil  -0.1294  -0.2521  2.3913  2.4116  4.8131  2.2864  7.9449  5.7283  5.2169  5.4219
Pencil  -0.2669  -0.5303  3.1167  3.1484  6.281  2.8846  -8.6885  5.646  5.0878  5.3503
Pencil  -0.2632  -0.5229  2.2964  2.317  4.6236  2.0565  7.2988  5.6737  5.1057  5.3724
comb  0.5557  1.3179  5.081  5.5177  -11.9442  -7.0948  -10.8183  5.7676  5.1788  5.423
comb  0.5666  1.3476  4.438  4.8754  9.5951  5.6207  -9.8314  5.7072  5.0718  5.3699
comb  0.5716  1.3668  5.0236  5.5967  -11.4754  -6.6061  10.9233  5.6482  5.2386  5.3999
fork  0.2118  0.4642  1.4558  1.5506  3.0539  1.7835  -5.0503  5.6387  5.259  5.4204
fork  0.0857  0.2003  0.7578  0.812  1.5968  0.9122  -4.1577  5.5046  5.0388  5.2541
fork  0.0951  0.2199  0.7799  0.8343  1.6414  0.9443  -4.3958  5.7105  5.1663  5.4281
knief  -0.0759  -0.141  2.2782  2.3404  4.6498  2.2743  6.7809  5.6842  5.2833  5.4406
knief  -0.0695  -0.1275  1.166  1.2116  2.4004  1.1498  4.6804  5.6622  5.2421  5.4085
knief  -0.099  -0.1886  1.4038  1.4498  2.8766  1.3583  5.0808  5.6441  5.0906  5.3497
knief  -0.0988  -0.1882  1.4051  1.4505  2.8783  1.3589  -5.1184  5.6427  5.1719  5.3909
knief  -0.0077  -0.0027  2.5281  3.4149  -6.7521  -3.6065  -6.431  5.6691  5.4931  5.5657
rread 2
[mjpeg @ 000001b491b9cec0] overread 8
[mjpeg @ 000001b491b9cec0] overread 8
[mjpeg @ 000001b491b9cec0] overread 8
[mjpeg @ 000001b491b9cec0] overread 8
[mjpeg @ 000001b491b9cec0] overread 8

Top match:Toothbrush
```

**Task 7: Implement a different classifier**

As comparison to a simple classifier, the KNN classifier is utilized to provide results that are more accurate.



```
mouse  0.7428  2.1681  4.8357  8.1646  11.6724  7.2501  -12.3962  5.7233  4.9135  5.3617
Leaf  0.5072  1.1509  3.1758  3.4326  6.7371  4.0136  8.0951  5.6396  4.9317  5.2845
Leaf  0.6045  1.4468  3.5945  4.0368  7.8525  4.7602  -9.6957  5.7289  4.9411  5.3297
Leaf  0.5063  1.1475  3.5232  3.7521  7.3899  4.3278  -8.994  5.7152  5.0088  5.3608
Leaf  0.6149  1.4823  3.5765  4.065  7.8903  4.8272  8.7322  5.7217  5.0289  5.3711
Toothbrush  -0.0679  -0.12  2.0106  2.2198  4.3362  2.1683  -5.4723  5.6954  5.3664  5.4758
Toothbrush  -0.1934  -0.3767  1.6099  1.7493  3.429  1.5631  -4.9958  5.6336  5.1588  5.3679
Toothbrush  -0.1924  -0.3745  1.4044  1.5296  2.9969  1.3487  4.4255  5.5888  5.098  5.3129
Toothbrush  -0.0775  -0.1384  1.8744  2.0922  4.0776  2.0387  5.0854  5.654  5.3751  5.4616
Pencil  -0.1294  -0.2521  2.3913  2.4116  4.8131  2.2864  7.9449  5.7283  5.2169  5.4219
Pencil  -0.2669  -0.5303  3.1167  3.1484  6.281  2.8846  -8.6885  5.646  5.0878  5.3503
Pencil  -0.2632  -0.5229  2.2964  2.317  4.6236  2.0565  7.2988  5.6737  5.1057  5.3724
comb  0.5557  1.3179  5.081  5.5177  -11.9442  -7.0948  -10.8183  5.7676  5.1788  5.423
comb  0.5666  1.3476  4.438  4.8754  9.5951  5.6207  -9.8314  5.7072  5.0718  5.3699
comb  0.5716  1.3668  5.0236  5.5967  -11.4754  -6.6061  10.9233  5.6482  5.2386  5.3999
fork  0.2118  0.4642  1.4558  1.5506  3.0539  1.7835  -5.0503  5.6387  5.259  5.4204
fork  0.0857  0.2003  0.7578  0.812  1.5968  0.9122  -4.1577  5.5046  5.0388  5.2541
fork  0.0951  0.2199  0.7799  0.8343  1.6414  0.9443  -4.3958  5.7105  5.1663  5.4281
knief  -0.0759  -0.141  2.2782  2.3404  4.6498  2.2743  6.7809  5.6842  5.2833  5.4406
knief  -0.0695  -0.1275  1.166  1.2116  2.4004  1.1498  4.6804  5.6622  5.2421  5.4085
knief  -0.099  -0.1886  1.4038  1.4498  2.8766  1.3583  5.0808  5.6441  5.0906  5.3497
knief  -0.0988  -0.1882  1.4051  1.4505  2.8783  1.3589  -5.1184  5.6427  5.1719  5.3909
knief  -0.0077  -0.0027  2.5281  3.4149  -6.7521  -3.6065  -6.431  5.6691  5.4931  5.5657
Top match:knief
```

**Task 8: Evaluate the performance of your system**

The confusion matrix for a group of items using the KNN classifier is displayed in the table below. Five input images of each object were examined. The KNN classifier is producing strong results, as can be seen from the confusion matrix, which has the majority of its weight along the diagonal.

| | comb | fork | knief | spoon | pen | hard_disk | Key | Chocolate | Hand-crea | Post-it | Watch | Lighter | mouse | Leaf | Toothbrus | Pencil |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| comb | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| fork | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| knief | 0 | 2 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| spoon | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| pen | 0 | 0 | 1 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| hard_disk | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Key | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Chocolate | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Hand-crea | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Post-it | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 |
| Watch | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 |
| Lighter | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 2 | 0 | 0 | 3 | 0 | 0 | 0 | 0 |
| mouse | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 |
| Leaf | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 |
| Toothbrus | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 |
| Pencil | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |

https://northeastern-my.sharepoint.com/:v:/r/personal/dhaduti_k_northeastern_edu/Documents/obj_recognition.mp4?csf=1&web=1&e=hJ7j9J

**Reflection**

I became more familiar with several relevant OpenCV methods while learning about the object identification technique. I now have a better knowledge of how changing light sources and the 2D shapes of things may affect the outcome of object recognition thanks to this study. I gained a lot of knowledge regarding how an object's moments are computed as well.

**Acknowledgement**

I would like to express my sincere gratitude and appreciation to my fellow classmate Jose Thandapral who has provided guidance throughout the course of this project. We had several discussions and brainstorming sessions, where we explored different strategies and approaches to tackle the challenges of this project.

I used the following resources to get a better understanding of the project:

https://www.geeksforgeeks.org/connect-your-android-phone-camera-to-opencv-python/

https://docs.opencv.org/3.4/d1/dc5/tutorial_background_subtraction.html

https://learnopencv.com/shape-matching-using-hu-moments-c-python/

https://answers.opencv.org/question/32910/image-moment-normalization/

https://docs.opencv.org/4.x/dd/de1/classcv_1_1ml_1_1KNearest.html