Assignment 1: Instance Search Report

LI Xiaoyang

April 5, 2023

1 Introduction

In this project, scale invariant feature transform (SIFT) [1], local binary pattern (LBP) [3] and color histogram is implemented to facilitate the instance search task and the experiment is mainly based on SIFT implementation. Due to the lack of training data with annotations and the device limit, it is rather challenging and time-consuming to prepare the training data set and to train a model from scratch. Therefore, CNN-based methods are not considered in this assignment.

2 Methodology

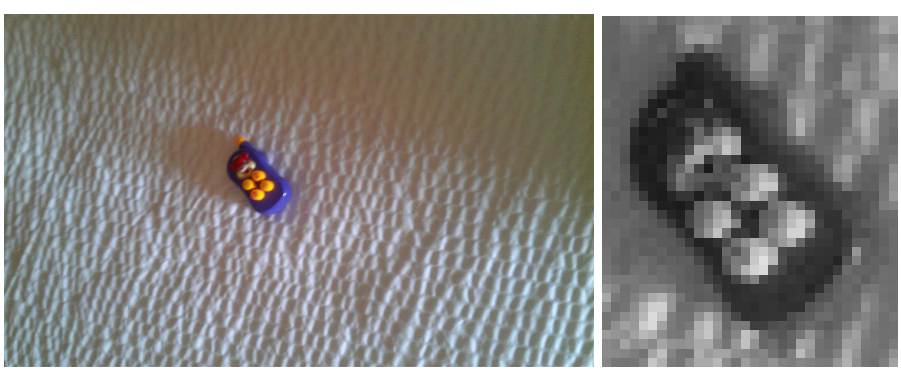
2.1 Scale Invariant Feature Transform

Basically, the implementation of SIFT follows the ideas from the original paper, including scale space construction, difference of Gaussian (DoG), peak detection, keypoint localization and descriptor generation with orientation.

In my default setting, the number of octaves is 9 and the number of scales sampled per octave is 6, which is different from the suggested setting in original paper (#octaves = 4, #scale = 5) [1], in order to increase the number of keypoints per image. In terms of prior smoothing for each octave, my initial sigma is 1.52 ($\sigma_0 = 1.52$). Firstly, a base image is created by upsampling by two in both directions and blurring with σ_0. Notably, the initial blur is assumed to be 1.0 ($\sigma_{assume} = 1.0$). Therefore, prior smooth $\hat{\sigma}_0$ is applied to the image base, where

$$\hat{\sigma}_0 = \sqrt{\sigma_0^2 - \sigma_{assume}^2}$$

The reason base image is upsampled is that the instance in ground truth bounding box can be quite small, which leads to few keypoints per image. Nevertheless, the experiment results reveals that, for some cases, instance can still be too small to generate enough keypoints after upsampling, which is demonstrated in the following sections.



(a) original image (b) GT

Figure 1: Tiny Instance

The image base will be processed with multiple Gaussian filters, of which the output is a scale. Then the algorithm moves on to the next octave by downscale the image by two and produce another S ($S = 6$) Gaussian blurred images. This process continues until the image size reach certain limit. In my implementation, the top octave is discarded due to its small size.

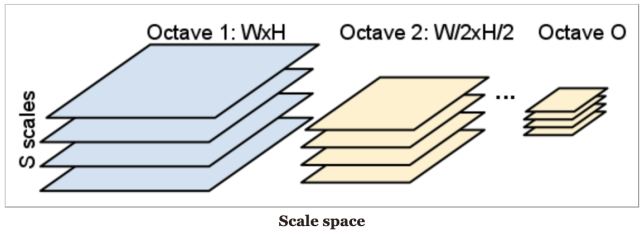


Figure 2: Scale Space Construction, adapted from Google

Next, difference of Gaussian is approximated by subtracting the laplacian of Gaussian, essentially adjacent scales per octave. After calculating DoG, peak detection is conducted in suggested manner, where 26 neighboring pixels across 3 scales are compared. If the pixel value of one certain pixel is either higher or lower than that of any of the 26 neighboring pixel, it is marked as **keypoint**. To avoid low contrast keypoints and edge response, normally a large number of keypoints are eliminated by deduplicating.

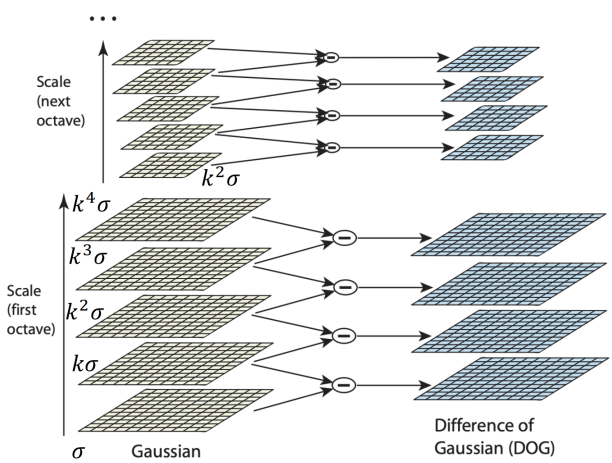


Figure 3: Difference of Gaussian, adapted from IJCV by David Lowe [1]

Lastly, the orientation assignment step calculates orientation of each keypoint base on gradient. The magnitude and direction calculations for the gradient are done for every pixel in a neighboring window around the keypoint in the Gaussian-blurred image. Each pixel votes in a orientation histogram. In my implementation, only the maximum orientation in the histogram is selected as the orientation of a keypoint for simplicity.

2.2 Local Binary Pattern

A more robust variation of Local Binary Pattern (LBP) [4] method is implemented in this assignment to make it rotation-invariant. Traditional local binary pattern descriptor is scale-invariant but not rotation-invariant as defined. Therefore, the rotation of the image can lead to arbitrary LBP value. To tackle this issue, a improvement is to revolve the circle neighborhood and to calculate the minimum as the local binary pattern value of this neighborhood, which makes arbitrary LBP patterns generated by rotation match an identical pattern.

Apart from the extension mentioned above, the step of deriving feature vectors is generally identical. In a 3-by-3 square neighborhood, the pixel value of the center pixel is selected as threshold. Compared with 8 adjacent pixels, if one is larger than the threshold, it is marked as 1, otherwise

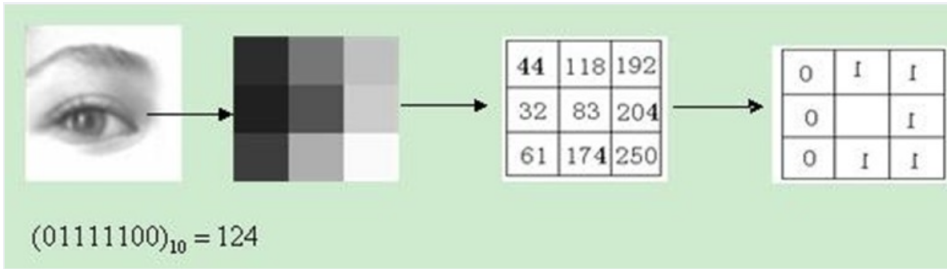


Figure 4: 8-digit LBP code, adapted from Google

0. In this way, a 8-digit binary code is generated ranging from 0 to 255, to represent the texture feature in corresponding square neighborhood. A popular variation is to adopt circle neighborhood instead of square neighborhood, which is not implemented in this project for simplicity. A pseudo-code implementation is as follows.

Algorithm 1 calculate neighborhood feature

```
Initialize vector v, pixel position (i,j)
threshold ← image[i, j]
for pixel ∈ neighborhood do
    (x, y) ← pixel
    if image[x, y] ≤ threshold then
        v appends 0
    else
        v appends 1
    end if
end for
vector v
```

2.3 Color Histogram & Combination of Methods

Since SIFT and LBP based methods only focus on local texture features and ignore the color features of the image, color histogram method is implemented to make use of the information about the color feature of images. In my implementation, color histogram is normalized. Notably, chi-square distance is utilized to measure the similarity.

$$\chi^2(x,y) = \sqrt{\sum_i (\frac{x_i - E(x_i)}{E(x_i)})^2 + \sum_j (\frac{y_j - E(y_j)}{E(y_j)})^2}$$

However, due to the lack of bounding box ground truth in search dataset, it is impossible to merely extract the color feature of the instance without involving color features from the background. Therefore, the color feature of images to be matched can be extremely unreliable in real practice, taking the noisy into account. **For above reasons, color histogram method will only be a support criterion to filter some trivially unmatched images for SIFT or LBP based methods in experiments.**

Figure 5: Example

3 Future Improvements

As mentioned in previous sections, color features of images are not fully utilized in my current methodologies since both SIFT and LBP are texture-feature-focused. Instead of extracting features from gray scale image, Colored Scale Invariant Feature Transform (CSIFT) [2] is proposed to extend the original algorithm into color domain, which enables us to eliminate trivially unmatched candidates in terms of color feature.

Another intuition is that further improvements can be made in the implementation of feature clustering and classification. For instance, RANSAC can be utilized in regression of keypoint matching. Last but not least, Query Extension can also be used to search for results in dataset iteratively until enough matches, say 10, are found.

4 Benchmark

The experiment result of top 10 queries is as follows.
Q1: 3457 4125 2443 902 3313 3559 4847 147 189 224
Q2: 3259 672 1876 4127 2389 829 889 1287 1396 2003
Q3: 3113 2489 3399 184 297 307 473 586 791 851
Q4: 173 1323 4456 1799 2427 357 1724 1929 2149 2473
Q5: 2403 4937 4371 1187 2940 3065 3689 151 286 360
Q6: 2003 934 1227 839 991 2490 3875 1952 3682 52
Q7: 2857 2726 79 2444 3005 2402 2683 1603 2701 1234
Q8: 770 1295 3660 426 4277 4346 4173 41 239 403
Q9: 2666 2458 2948 3747 2727 1020 2989 3164 4164 2179
Q10: 2176 4018 139 1200 2106 1482 2488 4781 11 67 394
Q11: 4386 1044 2491 968 2220 2365 1467 2003 2500 2842
Q12: 4256 1130 670 3268 3908 4967 1903 2399 585 1326
Q13: 3331 3370 4228 990 584 739 4576 4569 3493 1954
Q14: 4325 3166 716 291 3481 4978 4256 1020 2214 810
Q15: 1038 1368 3071 3586 648 1380 1809 2489 985 4438
Q16: 456 22 3904 166 181 200 386 388 477 577
Q17: 2 19 613 606 2460 2735 1977 340 3003 4011
Q18: 1276 1164 623 4792 649 1030 756 2151 2245 2574
Q19: 4332 4286 1459 1028 4170 1784 1884 4184 2807 3522
Q20: 3259 672 1876 4127 2389 829 889 1287 1396 2003

References

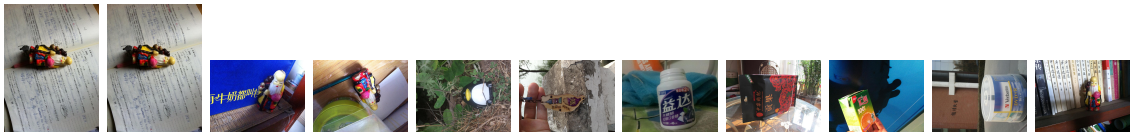
[1] Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision 60, 91–110. URL: https://doi.org/10.1023/B:VISI.0000029664.99615.94, doi:10.1023/B:VISI.0000029664.99615.94.

[2] Luke, R.H., Keller, J.M., Chamorro-Martinez, J., 2008. Extending the scale invariant feature transform descriptor into the color domain. Proc ICGST Int J Graph Vis Image Process, GVIP 8, 35–43.

[3] Ojala, T., Pietikainen, M., Harwood, D., 1994. Performance evaluation of texture measures with classification based on kullback discrimination of distributions, in: Proceedings of 12th International Conference on Pattern Recognition, pp. 582–585 vol.1. doi:10.1109/ICPR.1994.576366.

[4] Ojala, T., Pietikainen, M., Maenpaa, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 971–987. doi:10.1109/TPAMI.2002.1017623.

Supplementary Material



(a) **27** (b) 3457 (c) 4125 (d) 2443 (e) 902 (f) 3313 (g) 3559 (h) 4847 (i) 147 (j) 189 (k) 224

Figure 6: Q1(P=0.4)

(a) **35** (b) 86 (c) 930 (d) 1238 (e) 1860 (f) 1192 (g) 630 (h) 3544 (i) 4523 (j) 1950 (k) 2082

Figure 7: Q2(P=0.9)

(a) **316** (b) 3113 (c) 2489 (d) 3399 (e) 184 (f) 297 (g) 307 (h) 473 (i) 586 (j) 791 (k) 851

Figure 8: Q3(P=0.4)

(a) **776** (b) 173 (c) 1323 (d) 4456 (e) 1799 (f) 2427 (g) 357 (h) 1724 (i) 1929 (j) 2149 (k) 2473

Figure 9: Q4(P=0.6)

(a) **1258** (b) 2403 (c) 4937 (d) 4371 (e) 1187 (f) 2940 (g) 3065 (h) 3689 (i) 151 (j) 286 (k) 360

Figure 10: Q5(P=0.6)

(a) **1656** (b) 2003 (c) 934 (d) 1227 (e) 839 (f) 991 (g) 2490 (h) 3875 (i) 1952 (j) 3682 (k) 52

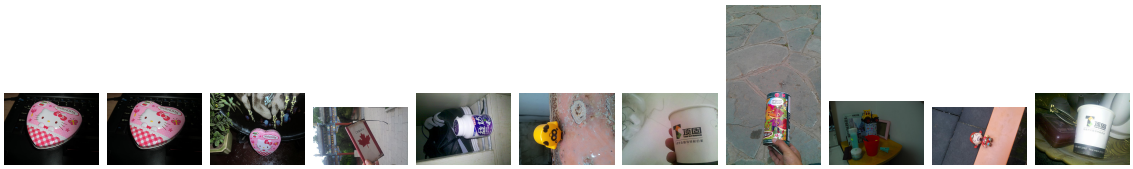
Figure 11: Q6(P=0.7)

(a) **2032** (b) 770 (c) 1295 (d) 3660 (e) 426 (f) 4277 (g) 4346 (h) 4173 (i) 41 (j) 239 (k) 403

Figure 12: Q8(P=0.6)

(a) **1709** (b) 2857 (c) 2726 (d) 79 (e) 2444 (f) 3005 (g) 2402 (h) 2683 (i) 1603 (j) 2701 (k) 1234

Figure 13: Q7(P=1.0)

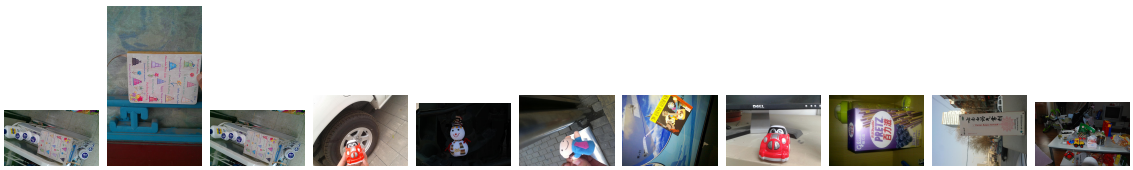


(a) **2040** (b) 2666 (c) 2458 (d) 2948 (e) 2179 (f) 3747 (g) 2727 (h) 1020 (i) 2989 (j) 3164 (k) 4164

Figure 14: Q9(P=0.2)

(a) **2176** (b) 4018 (c) 139 (d) 1200 (e) 2106 (f) 1482 (g) 2488 (h) 4781 (i) 11 (j) 67 (k) 394

Figure 15: Q10(P=0.6)



(a) **2461** (b) 4386 (c) 1044 (d) 2491 (e) 968 (f) 2220 (g) 2365 (h) 1467 (i) 2003 (j) 189 (k) 2842

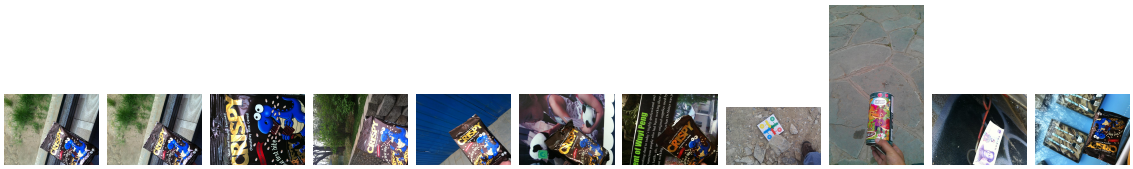
Figure 16: Q11(P=0.2)

(a) **2714** (b) 4256 (c) 1130 (d) 670 (e) 3268 (f) 1192 (g) 4967 (h) 1903 (i) 2399 (j) 585 (k) 1326

Figure 17: Q12(P=0.7)

(a) **3502** (b) 3331 (c) 3370 (d) 4228 (e) 990 (f) 584 (g) 739 (h) 4576 (i) 4569 (j) 3493 (k) 1954

Figure 18: Q13(P=1.0)



(a) **3557** (b) 4325 (c) 3166 (d) 716 (e) 291 (f) 3481 (g) 4978 (h) 4256 (i) 1020 (j) 2214 (k) 810

Figure 19: Q14(P=0.7)

(a) **3833** (b) 1038 (c) 1368 (d) 3071 (e) 3586 (f) 648 (g) 1380 (h) 1809 (i) 2489 (j) 985 (k) 4438

Figure 20: Q15(P=0.4)

(a) **3906** (b) 456 (c) 22 (d) 3904 (e) 166 (f) 181 (g) 200 (h) 386 (i) 388 (j) 477 (k) 577

Figure 21: Q16(P=0.2)

(a) **4354** (b) 2 (c) 19 (d) 613 (e) 606 (f) 2460 (g) 2735 (h) 1977 (i) 340 (j) 3003 (k) 4011

Figure 22: Q17(P=0.7)

(a) **4445** (b) 1276 (c) 1164 (d) 623 (e) 4792 (f) 649 (g) 1030 (h) 756 (i) 2151 (j) 2245 (k) 2574

Figure 23: Q18(P=0.6)

(a) **4716** (b) 4332 (c) 4286 (d) 1459 (e) 1028 (f) 4170 (g) 1784 (h) 1884 (i) 4184 (j) 2807 (k) 3522

Figure 24: Q19(P=0.9)

(a) **4929** (b) 3259 (c) 672 (d) 1876 (e) 4127 (f) 2389 (g) 829 (h) 889 (i) 1287 (j) 1396 (k) 2003

Figure 25: Q20(P=0.5)