Report

Xinran TANG (20126518)

shyxt1@nottingham.edu.cn

Category

1. Task1

1.1 SNR Recording

Image	Gaussian filter	Median filter	Anisotropic filter	Bilateral filter	Combinations
Image1.bmp	18.1167	24.2886	19.7050	17.4048	N.A.
Image2.bmp	19.4083	18.5014	19.5643	19.6886	N.A.
Image3.bmp	15.7059	16.0351	16.5461	16.0388	16.6476

Table 1: SNRs for three distorted images at optimal filter parameters.

1.2 Optimal Filter Parameters Recording

Image	Gaussian filter	Median filter	Anisotropic filter	Bilateral filter	Combinations
Image1.bmp	hsize=9 sigma=1.2	N=3	Iter=25	r=5 sigma1=4 sigma2=1	N.A.
Image2.bmp	hsize=7 sigma=1.1	N=5	Iter=10	r=5 sigma1=1 sigma2=1	N.A.
Image3.bmp	hsize=9 sigma=1.45	N=5	Iter=13	r=3 sigma1=2 sigma2=1	Anisotropic: Iter=7 Median: N=5

Table 2: Optimal parameters for three distorted images.

1.3 Best Filter Choose

- **image1**
  Median filter. Because the noise in image1 is salt and pepper noise, not an additive noise. The best filter for salt and pepper noise is the median filter.

- **image2**
  Bilateral filter. Because the noise in image2 is additive, the bilateral filter can remove additive noise. Furthermore, it not only considers the weighs pixels but also considers the intensity. And can also help preserve the edge.

- **image3**
  Combination filter. Because the kind of noises in image3 have both additive noise and non-additive noise. The anisotropic filter can remove the additive noise and the median filter can remove the non-additive

noise. Having both filters can reduce the noise in image 3 more efficiently.

1.4 Parameters Choose

For parameters choosing, I first set the different sizes of loops for all filters. Within the loop, if the biggest SNR were found, it would be returned with optimal parameter values.

- **Gaussian filter**
  hsize is the size of the filter, sigma is the standard deviation of gaussian.

- **Median filter**
  N represents the N-N neighborhood around the corresponding pixel in the input image.

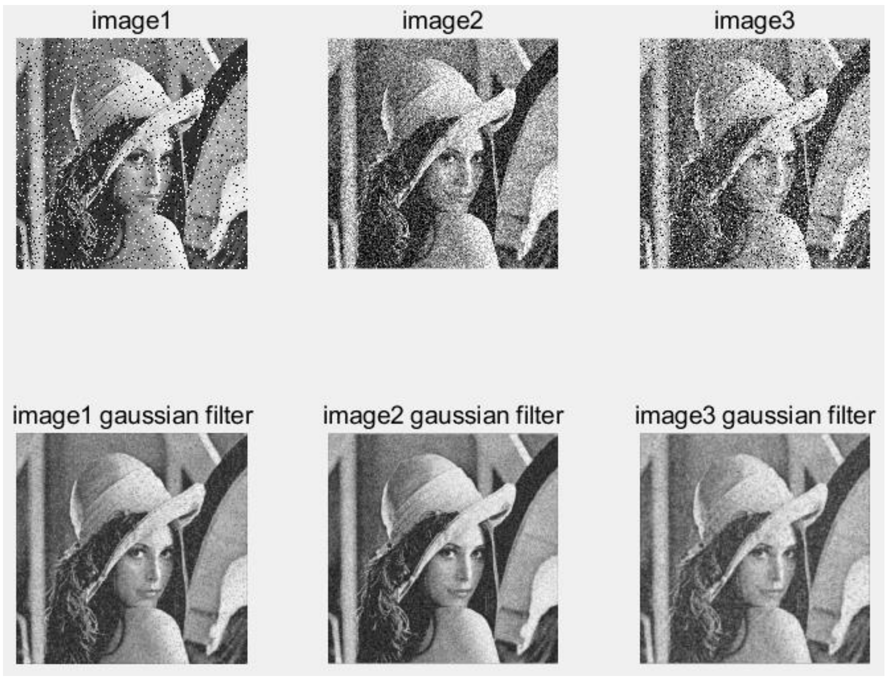
- **Anisotropic filter**
  Iter represents the times of applying the anisotropic filter on a specific image.

- **Bilateral filter**
  r is the amount of padding to add to each dimension, sigma1 is the standard deviation of gaussian in space weight, sigma2 is the standard deviation of gaussian in range weight.

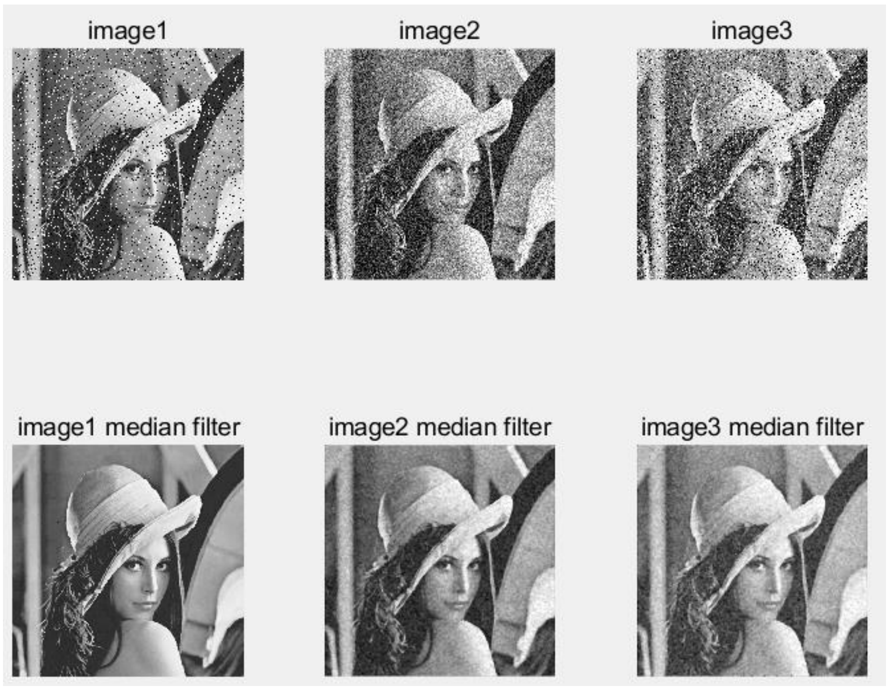
- **Combination filter**
  For filters chosen in combination filter, the anisotropic and median filters are chosen because the median filter can help remove the non-additive noise in image3, and the anisotropic filter performs better than other filters to image3 to remove the additive noise.

1.5 Pictures

- **Gaussian Filter**



- **Median Filter**



- **Anisotropic Filter**

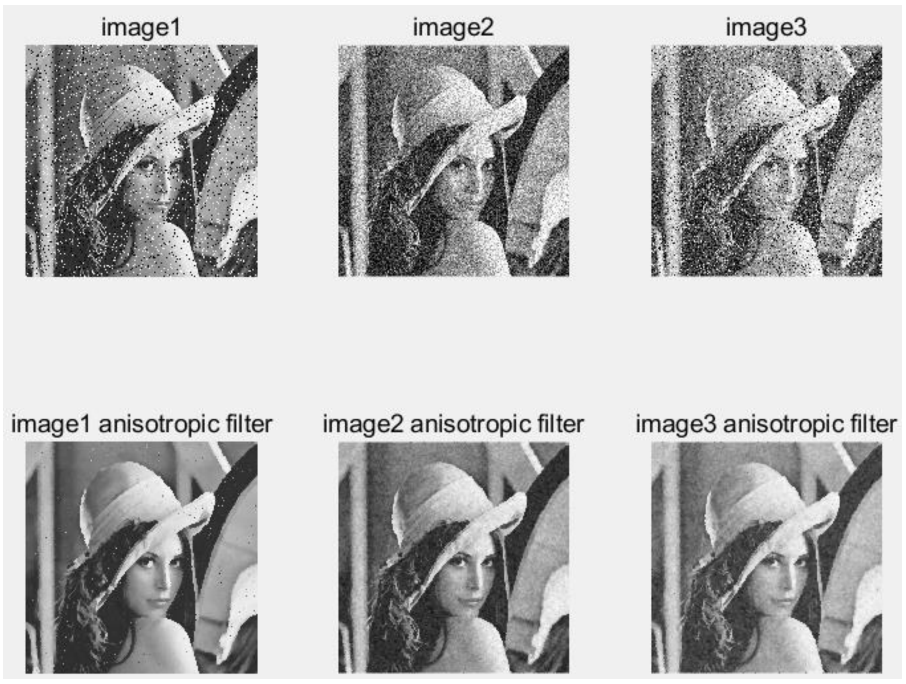
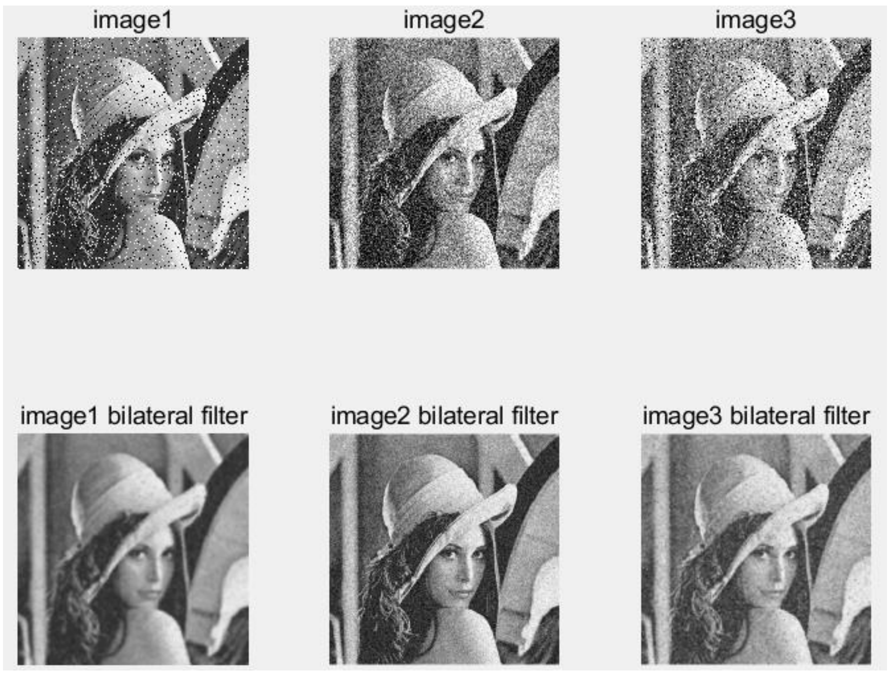
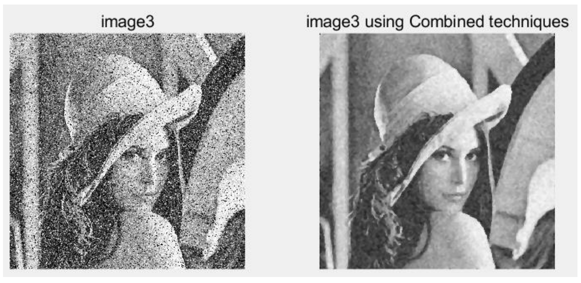


image1
image2
image3
image1 anisotropic filter
image2 anisotropic filter
image3 anisotropic filter

- **Bilateral Filter**



- ## Combinations



2. Task 2

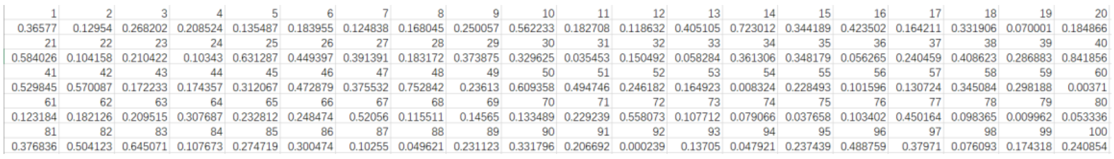
2.1 Quality of Detection Results

After running a total of 100 images, and calculated all corresponding IOUs, the average IOU is 0.2658 by my program.

- **overall IOU**

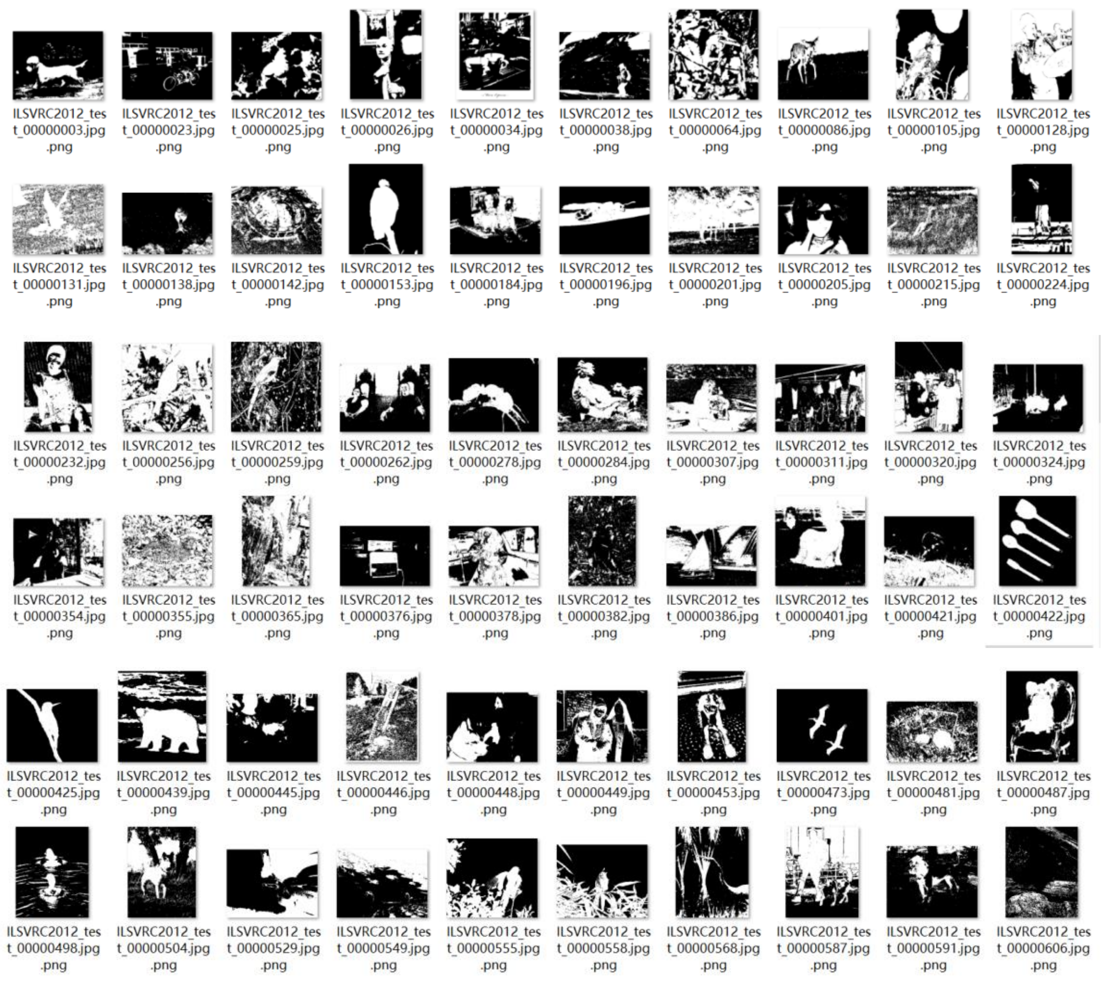
Average IOU	0.2658

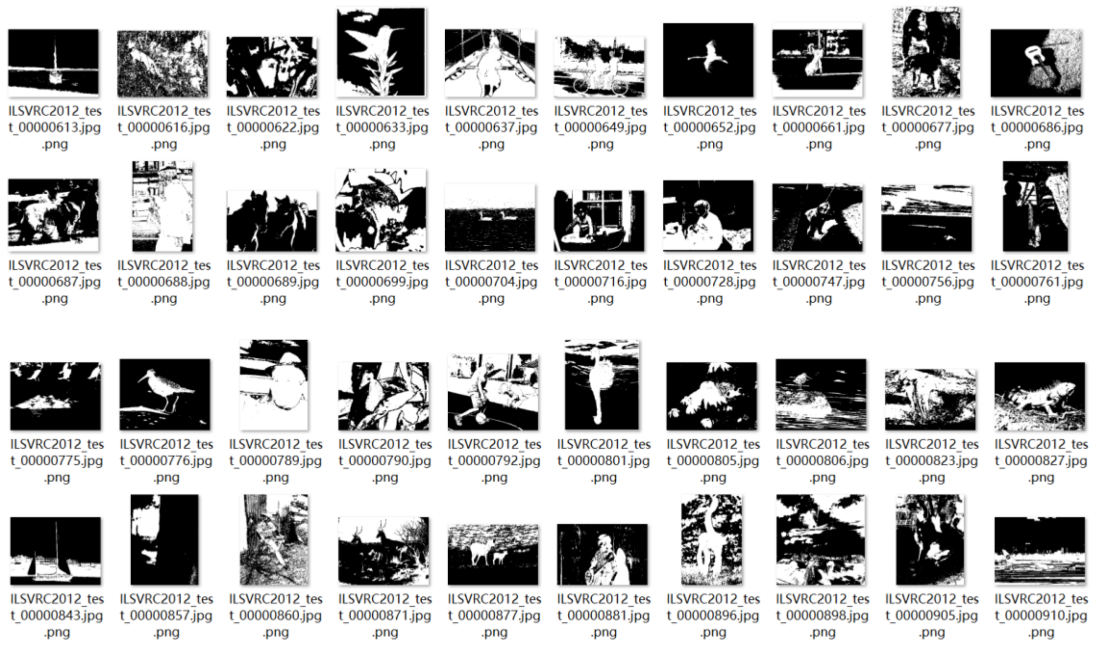
- **detail IOUs**



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0.36577	0.12954	0.268202	0.208524	0.135487	0.183955	0.124838	0.168045	0.250057	0.562233	0.182708	0.118632	0.405105	0.723012	0.344189	0.423502	0.164211	0.331906	0.070001	0.184866
21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
0.584026	0.104158	0.210422	0.10343	0.631287	0.449397	0.391391	0.183172	0.373875	0.329625	0.035453	0.150492	0.058284	0.361306	0.348179	0.056265	0.240459	0.408623	0.286883	0.841856
41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60
0.529845	0.570087	0.172233	0.174357	0.312067	0.472879	0.375532	0.752842	0.23613	0.609358	0.494746	0.246182	0.164923	0.008324	0.228493	0.101596	0.130724	0.345084	0.298188	0.00371
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
0.123184	0.182126	0.209515	0.307687	0.232812	0.248474	0.52056	0.115511	0.14565	0.133489	0.229239	0.558073	0.107712	0.079066	0.037658	0.103402	0.450164	0.098365	0.009962	0.053336
81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
0.376836	0.504123	0.645071	0.107673	0.274719	0.300474	0.10255	0.049621	0.231123	0.331796	0.206692	0.000239	0.13705	0.047921	0.237439	0.488759	0.37971	0.076093	0.174318	0.240854

- **detected pictures**


ILSVRC2012_test_00000003.jpg.png
ILSVRC2012_test_00000023.jpg.png
ILSVRC2012_test_00000025.jpg.png
ILSVRC2012_test_00000026.jpg.png
ILSVRC2012_test_00000034.jpg.png
ILSVRC2012_test_00000038.jpg.png
ILSVRC2012_test_00000064.jpg.png
ILSVRC2012_test_00000086.jpg.png
ILSVRC2012_test_00000105.jpg.png
ILSVRC2012_test_00000128.jpg.png
ILSVRC2012_test_00000131.jpg.png
ILSVRC2012_test_00000138.jpg.png
ILSVRC2012_test_00000142.jpg.png
ILSVRC2012_test_00000153.jpg.png
ILSVRC2012_test_00000184.jpg.png
ILSVRC2012_test_00000196.jpg.png
ILSVRC2012_test_00000201.jpg.png
ILSVRC2012_test_00000205.jpg.png
ILSVRC2012_test_00000215.jpg.png
ILSVRC2012_test_00000224.jpg.png
ILSVRC2012_test_00000232.jpg.png
ILSVRC2012_test_00000256.jpg.png
ILSVRC2012_test_00000259.jpg.png
ILSVRC2012_test_00000262.jpg.png
ILSVRC2012_test_00000278.jpg.png
ILSVRC2012_test_00000284.jpg.png
ILSVRC2012_test_00000307.jpg.png
ILSVRC2012_test_00000311.jpg.png
ILSVRC2012_test_00000320.jpg.png
ILSVRC2012_test_00000324.jpg.png
ILSVRC2012_test_00000354.jpg.png
ILSVRC2012_test_00000355.jpg.png
ILSVRC2012_test_00000365.jpg.png
ILSVRC2012_test_00000376.jpg.png
ILSVRC2012_test_00000378.jpg.png
ILSVRC2012_test_00000382.jpg.png
ILSVRC2012_test_00000386.jpg.png
ILSVRC2012_test_00000401.jpg.png
ILSVRC2012_test_00000421.jpg.png
ILSVRC2012_test_00000422.jpg.png
ILSVRC2012_test_00000425.jpg.png
ILSVRC2012_test_00000439.jpg.png
ILSVRC2012_test_00000445.jpg.png
ILSVRC2012_test_00000446.jpg.png
ILSVRC2012_test_00000448.jpg.png
ILSVRC2012_test_00000449.jpg.png
ILSVRC2012_test_00000453.jpg.png
ILSVRC2012_test_00000473.jpg.png
ILSVRC2012_test_00000481.jpg.png
ILSVRC2012_test_00000487.jpg.png
ILSVRC2012_test_00000498.jpg.png
ILSVRC2012_test_00000504.jpg.png
ILSVRC2012_test_00000529.jpg.png
ILSVRC2012_test_00000549.jpg.png
ILSVRC2012_test_00000555.jpg.png
ILSVRC2012_test_00000558.jpg.png
ILSVRC2012_test_00000568.jpg.png
ILSVRC2012_test_00000587.jpg.png
ILSVRC2012_test_00000591.jpg.png
ILSVRC2012_test_00000606.jpg.png


ILSVRC2012_test_00000613.jpg.png
ILSVRC2012_test_00000616.jpg.png
ILSVRC2012_test_00000622.jpg.png
ILSVRC2012_test_00000633.jpg.png
ILSVRC2012_test_00000637.jpg.png
ILSVRC2012_test_00000649.jpg.png
ILSVRC2012_test_00000652.jpg.png
ILSVRC2012_test_00000661.jpg.png
ILSVRC2012_test_00000677.jpg.png
ILSVRC2012_test_00000686.jpg.png
ILSVRC2012_test_00000687.jpg.png
ILSVRC2012_test_00000688.jpg.png
ILSVRC2012_test_00000689.jpg.png
ILSVRC2012_test_00000699.jpg.png
ILSVRC2012_test_00000704.jpg.png
ILSVRC2012_test_00000716.jpg.png
ILSVRC2012_test_00000728.jpg.png
ILSVRC2012_test_00000747.jpg.png
ILSVRC2012_test_00000756.jpg.png
ILSVRC2012_test_00000761.jpg.png
ILSVRC2012_test_00000775.jpg.png
ILSVRC2012_test_00000776.jpg.png
ILSVRC2012_test_00000789.jpg.png
ILSVRC2012_test_00000790.jpg.png
ILSVRC2012_test_00000792.jpg.png
ILSVRC2012_test_00000801.jpg.png
ILSVRC2012_test_00000805.jpg.png
ILSVRC2012_test_00000806.jpg.png
ILSVRC2012_test_00000823.jpg.png
ILSVRC2012_test_00000827.jpg.png
ILSVRC2012_test_00000843.jpg.png
ILSVRC2012_test_00000857.jpg.png
ILSVRC2012_test_00000860.jpg.png
ILSVRC2012_test_00000871.jpg.png
ILSVRC2012_test_00000877.jpg.png
ILSVRC2012_test_00000881.jpg.png
ILSVRC2012_test_00000896.jpg.png
ILSVRC2012_test_00000898.jpg.png
ILSVRC2012_test_00000905.jpg.png
ILSVRC2012_test_00000910.jpg.png

2.2 Key Advantages and Limitations

- **Advantages**
  The saliency detection method in the program can detect the saliency of 100 pictures with an average IOU 0.2658 in around 8 minutes, which is the key advantage.

  The current algorithm is chosen by considering balancing accuracy and time spent. Compared to the LC algorithm, the implemented algorithm also considered the color information which can help detect the saliency more accurately. Although the RC algorithm can detect saliency more accurately than implemented algorithm, it took a much longer time than it.

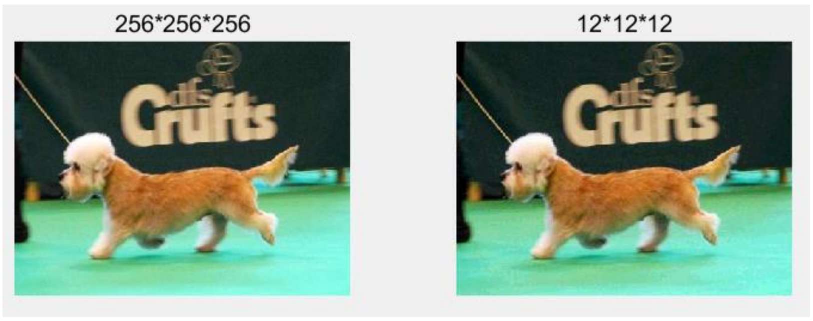
- **Limitations**
  Although the implemented algorithm can detect the saliency with an average IOU of 0.2658, it has some limitations. For example, there may be many unnecessary white regions except those that need to be highlighted; Or the detected regions included some impurities.

  For some small saliencies or the saliencies have similar colors to the background, the implemented algorithm cannot work well.

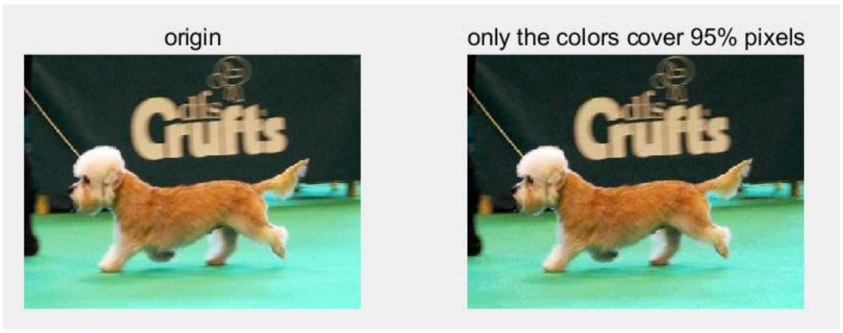
2.3 Justification of Algorithm

The implemented algorithm is based on the HC algorithm in the paper[1]. Apart from that, I had added some improvements to help increase the accuracy of saliency detection.

- **Histogram Acceleration**
  By mapping the large number of colors into a relatively small range can help reduce the time complexity of the algorithm. Therefore, I mapped 256*256*256 colors into 12*12*12 colors by even distribution. The following picture shows the image before and after mapping:



- **Reduce Number of Colors**
  The time complexity will be large if calculate the saliency value of all colors in a picture. Therefore, reducing a number of useful colors is important. In paper[1], it suggests preserving the colors which can cover more than 95% of total pixels is enough. By implementing the different percentages of preserving colors. The 95% coverage can be proven to be the most effective. The comparison images are shown below:



- **Color Distance Calculation**
  Quantization steps were done in RGB color space while the color distances were calculated in LAB color space. The equation to calculate color distance is defined as follows:

$$D(i,j) = \sqrt{(L(i) - L(j))^2 + \ (A(i) - A(j))^2 + (A(i) - A(j))^2} \qquad (1)$$

equation 1: i, j are two colors, and L, A, B are lab values of each color.

- **Saliency Calculation**

Saliency values can be used to differentiate the objective and background of a picture because objectives and backgrounds have different saliency values. In this algorithm, the saliency of color is calculated by the following function:

$$S(I_k) = S(c_l) = \sum_{j=1}^{n} f_i D(c_i, c_j) \tag{2}$$

equation 2: c_l is the color value of I_k pixel; n is the total number of colors in the picture; f_i is the probability occurrence of c_j in picture I

- **Color Space Smoothing**

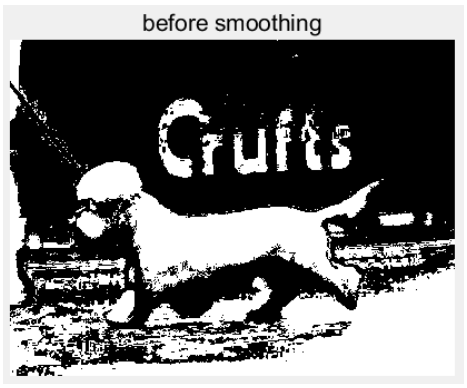
After preserving the colors covering more than 95% of total pixels, the rest of the colors are assigned as the nearest color. However, it may cause some noise. To avoid the noise caused by randomly assign new colors, smoothing is needed.

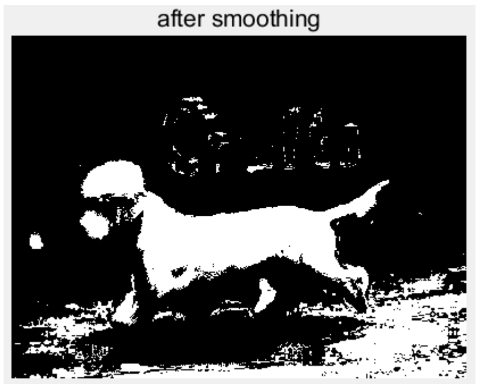
In the smoothing part, the new saliency value of each color is calculated as follows:

$$S'(i) = \frac{1}{(m-1)T} \sum_{i=1}^{m} (T - D(c, c_i)) S(c_i) \tag{3}$$

equation 3: m=n/4, n is the total number of preserved colors, T is the total distance between color c and its m nearest colors c_i

The following picture shows the detected salience before and after smoothing:

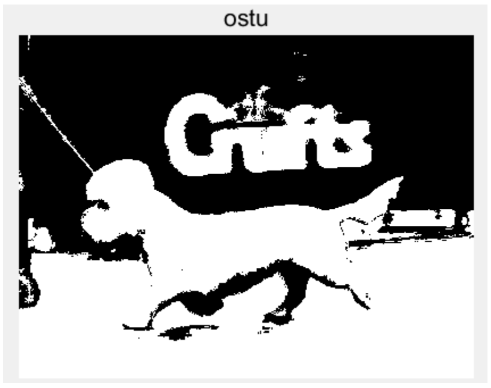


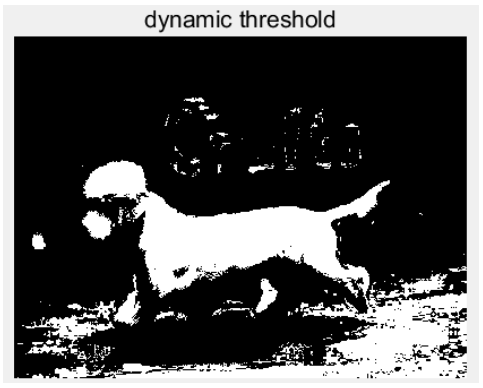


- **Threshold Definition**

Firstly, I implemented the Otsu algorithm to find the best threshold. Because Otsu algorithm have three assumptions: the image histogram is bimodal; two regions can be separated by one threshold; the histogram is made of two normal distribution. However, perhaps in violation of these three assumptions, this method cannot play well in finding the best threshold of saliency and background. After that, I decided to define the threshold dynamically. The optimal cut-off introduced in paper[1] is point 20%, which means the pixel with top 20% max saliency can be defined as saliency while others are background. Therefore, I choose the 0.8*max_saliency value as my first threshold point.

The problem of only has one threshold is that different pictures may have different characters, the static threshold can cause total white or total black. Therefore, I made some assumptions: if the total white pixels occupied more than 80% of total pixels, it is needed to assign a new threshold to avoid total white; on the other hand, it is also needed to assign a new threshold to avoid total black if total black pixels occupied more than 70% of total pixels. After the first update, there may be some following problems. So, I set a new pair of dynamic thresholds to avoid the problem that may be caused by the first threshold-reassign. The following picture shows the saliency detection with Otsu threshold and dynamic threshold:





2.4 Output Images

The output images will be stored in the new folder named: **detected saliencies**.

3. Reference

[1] M.M Cheng, et al. “Global contrast based salient region detection”, CVPR 2011.