Computer Vision 1 Lab Assignment 2

Anne Chel id:10727477, Bobbie van Gorp id:11161108,
Marvin Lau id:12364282 and Milan Klaasman id:11431474

February 2019

1 Introduction

This document consists of the answers of the second lab assignment of the course Computer Vision 1. Containing the following three different sections of the assignment: neighborhood processing, low-level filters and applications in image processing. It includes graphs, tables and code. The rest of the code can be found in the accompanied zip file.

2 Neigborhood Processing

Q1.1

The convolution and correlation operators differ in sign of the offset for I. First, a variant of the convolution formula can be written as:

$$g(i,j) = \sum_{k,l} f(i-k, j-l)h(k,l) = \sum_{k,l} f(k,l)h(i-k, j-l)$$

In this variant, the offset is present for h. When considering the difference between the operators with the sign of the offset in this case, the convolution operator with the minus sign can be considered flipping the mask both horizontally and vertically, while for the correlation case the flipping of the mask does not need to happen. So for the original formulas that was given, the convolution operator would flip the values in the neighborhood of the image and multiply it with each coefficient in h. In addition, the convolution operator both are associative and commutative [6].

Q1.2

In the previous question we have seen that the general formulas can be rewritten such that the offset are in the mask h. For the convolution this means flipping the mask in both horizontal and vertical directions. The resulting mask can then be applied as equivalent to the non flipped mask for the correlation case. The case where the flipped and non flipped mask are the same is when the mask is symmetrical, so the mask should have the form of a square matrix that is symmetrical.

3 Low-level filters

Q2

Because the Gaussian kernel is separable, the result of convolving an image with a 2D Gaussian kernel is exactly the same as the result of convolving an image with a 1D Gaussian kernel in the

x-direction and a 1D Gaussian kernel in the y-direction. The computational complexity of these two methods is not the same, however. With 2D kernel size $K \times K$, image width W and image height H, the computational complexity is $O(K^2WH)$, because for each pixel in the image K^2 operations are necessary. The other option of convolving a 1D Gaussian kernel of size K with the x- and y-direction separately has computational complexity $O(2KWH)$, because each pixel needs K operations for the x-direction and K for the y-direction.

Q3

The second order derivative of the Gaussian kernel can, like the first order derivative, detect edges in an image. More interestingly, because of its circular shape and similarities with ganglion cells, it can be used to detect blobs in the image.

Q4

First, λ controls the wavelength of the sinus, the central frequency of the carrier, to which wavelength it will respond best [1]. A smaller wavelength leads to a higher frequency and a larger wavelength leads to a lower frequency. The ψ is the amount of phase shift or offset for the sinus carrier signal with respect to the center [2]. The parameter θ determines the filter orientation, the angle between x axes and the sinus [1, 2]. With this you can search for textures with a certain orientation angle. The parameter σ defines the standard deviation of the Gaussian, changing the shape and size of the region to be checked in an image [1, 2]. Lastly, γ, defines the aspect ratio of the Gaussian (ellipse) form, for example can make the ellipse form more stretched in one direction than the other or make it about equal.

Q5

The effects of different θ, σ and γ values are visualized in Figures 1,2 and 3 respectively. In each case, the other parameters stayed fixed on certain values as specified in the caption of the figures. For θ it can be observed that it changes the orientation. The orientation value in radiance translates to an angle of 0,60,90,120 and 360 degrees from left to right. For σ, it can be observed that range of the Gaussian form increases. Lastly, for γ, low values present a stretched form compared to the ratio of 1.

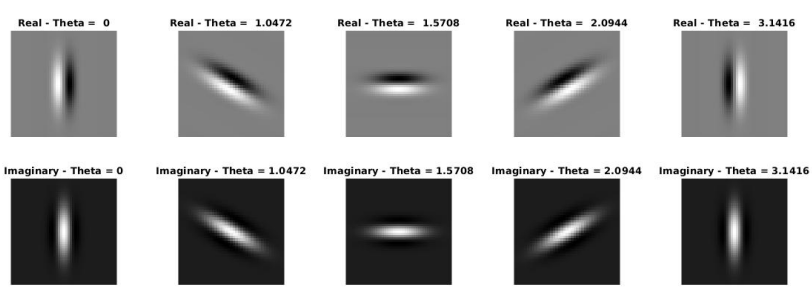


Figure 1: Overview of different theta results on filter - with other parameters fixed as; σ is 3, λ is 15, ψ is 1.5707 and γ is 0.4

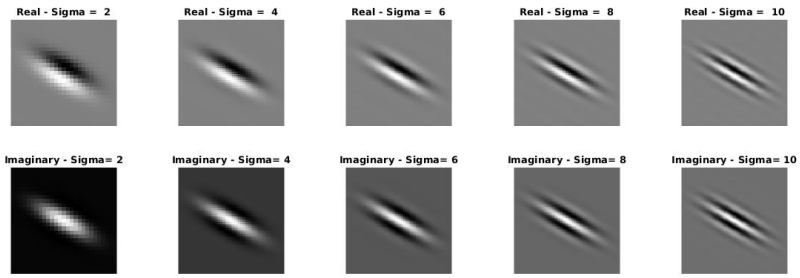


Figure 2: Overview of different sigma results on filter - with other parameters fixed as; θ is 1.0471, λ is 15, ψ is 1.5707 and γ is 0.4

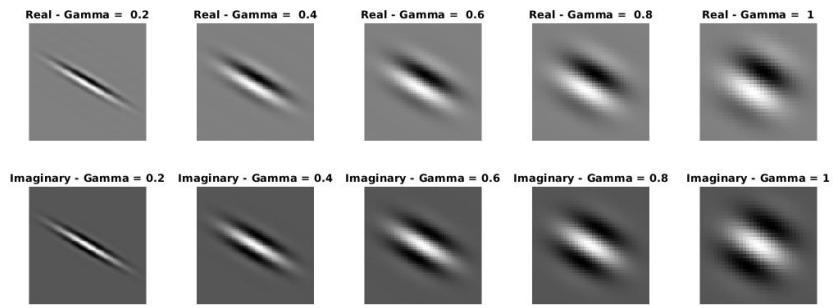


Figure 3: Overview of different gamma results on filter - with other parameters fixed as; σ is 6, θ is 1.0471, λ is 15, ψ is 1.5707

4 Image Denoising

4.1 Quantitative evaluation

Q6

Using the implemented myPSNR function (can be found in the corresponding data folder) When computing the PSNR between 'image1_saltpepper.jpg' and image1.jpg it returns a value of 16.1079 dB. The PSNR between image1_gaussian.jpg and 'image1.jpg' is 20.5835 dB.

4.2 Quantitative evaluation

1. In figure 4 the results of both box and median filtering for sizes 3x3, 5x5 and 7x7 are shown for the images saltpepper and gaussian .

2. The peak signal-to-noise ratio (PSNR) is often used as a quality measure in which a higher PSNR generally indicates a higher quality picture.[4]

   In Table 1 the PSNRs are shown for the four different combinations between images (saltpepper and gaussian) and type of filter (box and median). The filter size has as an effect that it reduces the PSNR score. This is the case for all the four combinations, in which 3x3 filters

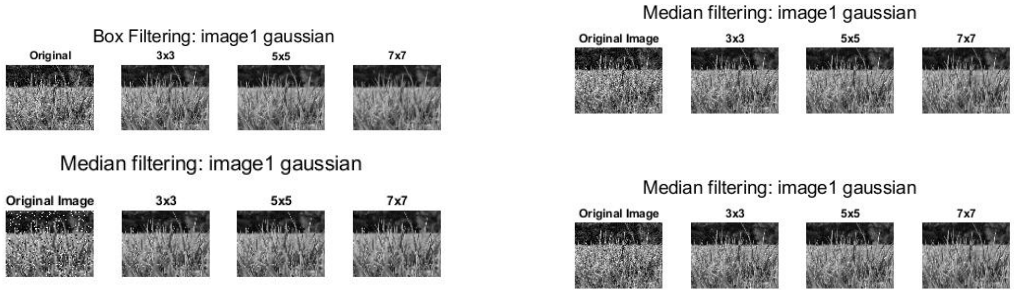


Figure 4

Filter size	PSNR Box saltpeper	PSNR Box Gaussian	PSNR Median saltpepper	PSNR Median Gaussian
3x3	16.4394	**20.3716**	16.1201	**20.2763**
5x5	15.7926	19.1919	16.0105	20.0087
7x7	15.4254	18.4129	15.9004	19.7217

Table 1: Caption

have a higher PSNR score than 5x5 and 5x5 has a higher PSNR score than 7x7. This would imply that in these cases the 3x3 filter size would be the best choice.

3. In terms of which filter type is better. The Median filter performs better, when looking at the PSNR. For every filter size and image combination the median filter outperforms the box filter in terms of the PSNR.

4. To answer this exercise it was not found beneficial to use the PSNR, this because a high score didn't correspond with a good image. Thus, was the best image depicted from plotting the images over different ranges of the input arguments. First we started with checking the difference in kernel size for a sigma of 1, this turned out to be unexplanatory, thus we switched to a set kernel size of 3x3 and tried to manipulate the sigma value. After tweaking the standard deviation (a result is shown in Figure 5), in combination with on optical evaluation a sigma value of approximately 0.75 turned out to be the best value. Now we used the sigma value of 0.75 to examine which kernel size would be most appropriate. This turned out to be not that important anymore, because as shown in Figure 6 the pictures didn't differ that significant anymore with the set sigma. Thus a kernel size of 3 would be most appropriate, due to the fact that the difference in quality isn't improving with a higher gradient kernel and thus it was found unnecessary and counterproductive. We are aware that the above explained approach has its limitations, because the best standard deviation might be influenced by the corresponding kernel that has been chosen in the first manipulation. However, this path was chosen due to the PSNR not giving an adequate answer.

5. The PSNR score for the sigma ranging from 0.1 to to ten for three different kernels (3x3, 5x5 and 7x79) can be seen in Table 2. Based on this data it seems that the PSNR increases for the three different kernels till 0.7-0.9 depending on the kernel and then decreases again. This is in confluence with the findings in Q 7.4. However the difference in PSNR values is very small. Thus it is difficult to make hard conclusions on this data.

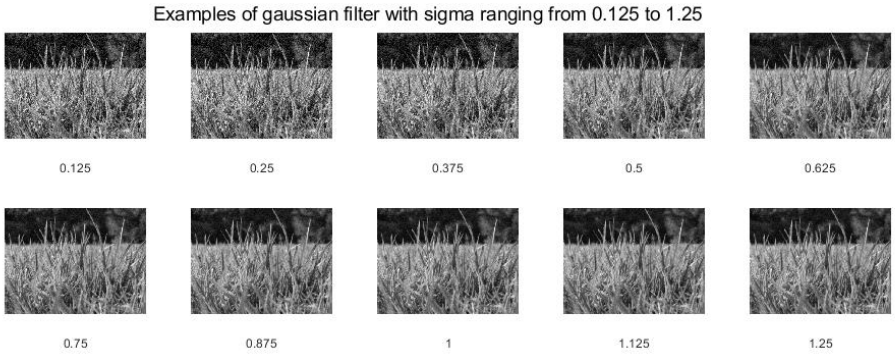


Figure 5: This figure shows ten images of the Gaussian filter with kernel

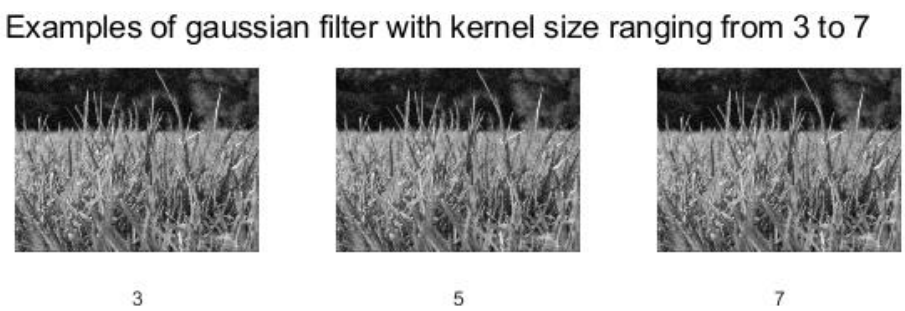


Figure 6: This figure shows three images of the manipulation of sigma values (3, 5 and 7).

Standard deviation (σ) :	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
3x3	7.1167	7.1167	7.1168	7.1171	7.1174	7.1176	7.1175	7.1176	7.1175	7.1175
5x5	7.1167	7.1167	7.1168	7.1171	7.1174	7.1176	7.1175	7.1175	7.1174	7.1172
7x7	7.1167	7.1167	7.1168	7.1171	7.1174	7.1176	7.1175	7.1175	7.1173	7.1172

Table 2: This table represents the PSNR for a standard deviation ranging from 0.1 to 1 for three different kernels, namely 3x3, 5x5, 7x7.

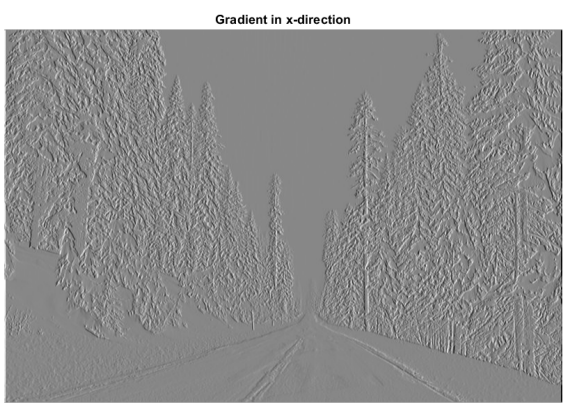
6. Median filter, operates over a window by selecting the median intensity/value of that window. It is effective for removing noise with extreme values like salt and pepper (random occurrences of black and white pixels), but still preserving edges.[7] Gaussian filter, replaces each pixel by a weighted average of its neighbors. The mask weights are computed by sampling a Gaussian function. Most commonly used to blur an image or to reduce noise. Box filter, the middle pixel of the mask is replaced by a value equal to the average value of its neighboring pixels in the input image. It is a simple and intuitive method of smoothing images. I.e. the reduction of the intensity variation between neighbouring pixels. It is also often used to reduce noise.

   If two filtering methods give a fairly similar PSNR, it is possible that there are qualitative differences. The PSNR isn't particular good at predicting human visual response (Qualitative differences) to image quality. It is a ratio between the maximum possible power of a signal and the power of noise that affects the signal. From the qualitative perspective, the structural similarity index measurement (SSIM) and other variations are commonly considered better.[3]

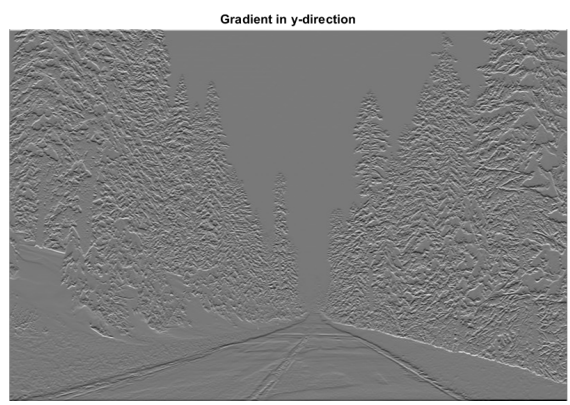
4.3 Edge detection

Q8

The gradient of an image can be used for edge detection. In figure 7 the gradients of image2.jpg in the x- and y-direction are shown. The gradient images show the change in intensity in either the x- or y-direction. Gray parts of the image have a small gradient, while black or white parts have higher gradients. Figure 7 (a) highlights intensity changes in the x-direction, so vertical edges, like tree trunks, are clearly visible. In figure 7 (b), the tree trunks are hardly visible at all. Although there are hardly any fully horizontal features in this image, the parts of the image that are closest to being horizontal, like the lines where the snow starts and ends and some of the tree branches, are highlighted in the y-direction gradient image.



(a) Gradient of image2.jpg in the x-direction approximated with Sobel kernel

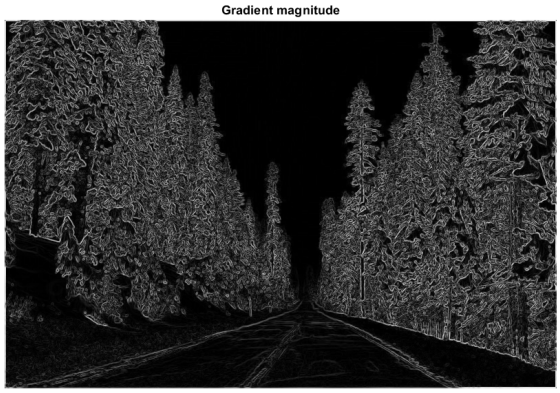


(b) Gradient of image2.jpg in the y-direction approximated with Sobel kernel

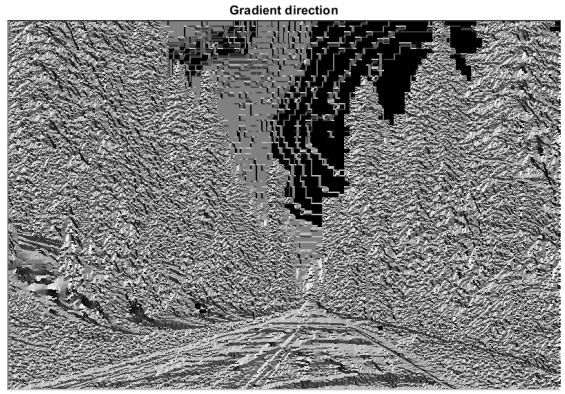
Figure 7

Figure 8 displays the gradient magnitude and gradient direction information for each pixel in the original image. The gradient magnitude indicates how large the change in intensity is in a certain pixel. Black parts of the gradient magnitude image indicate hardly any change, while white

parts show parts where there is a large change in intensity. As can be seen in this image, the parts with the largest gradient magnitude are the lines where snow starts or ends. The gradient direction is shown in (b), where the angles are encoded in grayscale. The scale goes from black to white, representing an angle going from 0 degrees to 360 degrees. Because all gradients directions are shown on this same scale, irrespective of the gradient magnitude, the gradient direction image can and does shows large color differences in parts of the image wherer there is hardly any gradient, like in the sky above the trees.



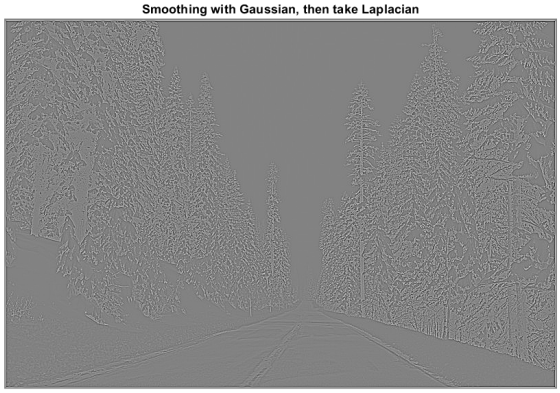
(a) Gradient magnitude of each pixel from image2.jpg



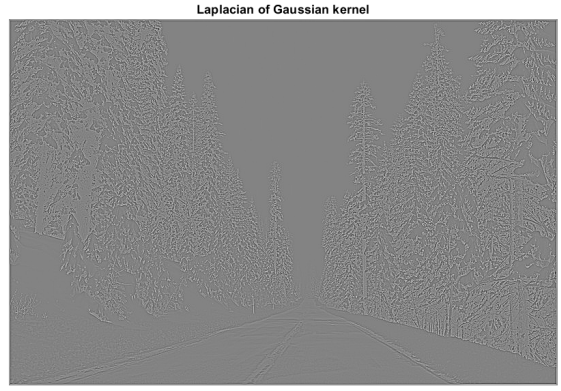
(b) Gradient direction of each pixel from image2.jpg

Figure 8

Q9



(a) LoG of image2.jpg by first smoothing with Gaussian kernel, then taking the Laplacian.



(b) LoG of image2.jpg by convolving with Laplacian of Gaussian kernel.

Figure 9

1. The results of the three methods are displayed in figures 9 and 10.

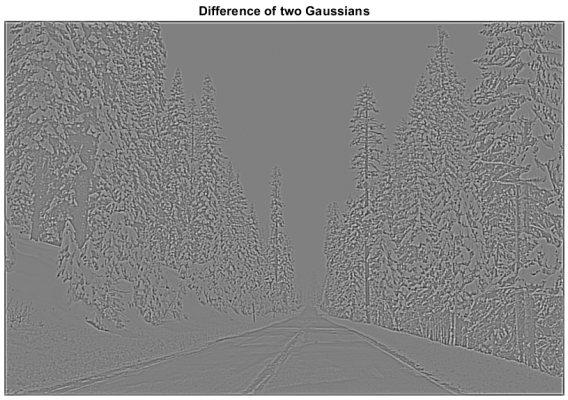


Figure 10: LoG of image2.jpg by taking difference of two Gaussians

2. The first two methods are applying the 2nd order derivative operator in a different order. Method 1 means first smoothing the image by convolving it with a Gaussian kernel and then taking the second order derivative of the result. The Laplacian of Gaussian method applies the 2nd order derivative to the Gaussian kernel itself and then convolves the image with this Laplacian kernel. Although the order of applying the 2nd derivative is different, the end result is the same since the convolution operation is associative. The difference of two Gaussians (DoG) method is not similar, because it does not involve derivatives. However, by substracting two images convolved with different Gaussian kernels from each other, the Mexican Hat function and thus the effect of the Laplacian of Gaussian is approximated.

3. Convolving the image with a Gaussian first reduces high-frequency noise in the image that the 2nd order derivative approximation is sensitive to.

4. It is difficult to see the difference between the standard deviations in our own implementation: many different values of σ result in images similar to the LoG method. According to the literature a ratio of 1.6 produces the best approximation [5]. Because this method involves substracting images convolved with two different standard deviatons, it does not only remove high-frequency noise, but it becomes a band-pass filter that removes low-frequency noise as well.

5. Gabor filters can be used to further isolate the road by setting the parameters such that it looks for features in the direction of the road edges. As with Gabor filters are mostly used for texture analysis, the orientation of θ for example can be set to match the road in the middle. In addition, the γ and σ can be adjusted such that it matches the road that stretched in the distance and creates the illusion that it converges.

4.4 Foreground-background separation

Running gabor segmentation results into images with a lot of grey or black dots that are detected in images with filters. Using k-means, these can be clustered. Since k = 2 here, it can be considered the clustering of fore and background. Objects that are the foreground may have certain edges for example that can be detected and everything with that can be "filled" in and belong to that cluster.

Thus it tries to distinguish objects that are the fore or background. Examples of running the algorithm with default parameters on Kobi can be seen in Figure 11 and 12.

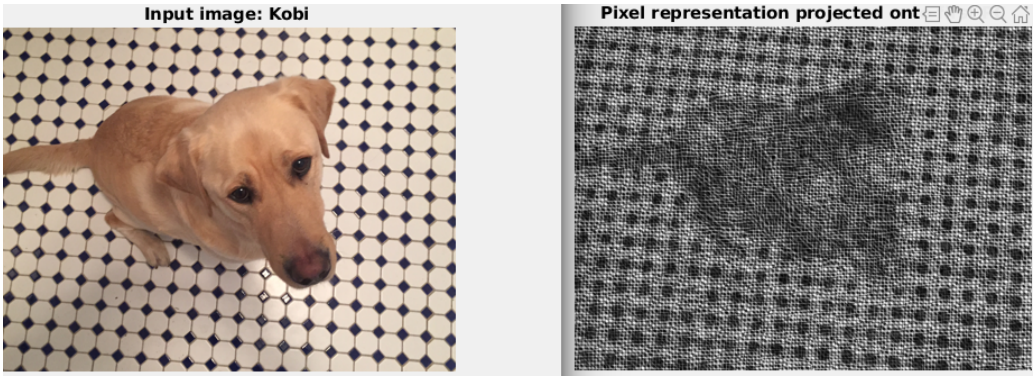


Figure 11: The saliency map using the first principal component of the features matrix on image Kobi

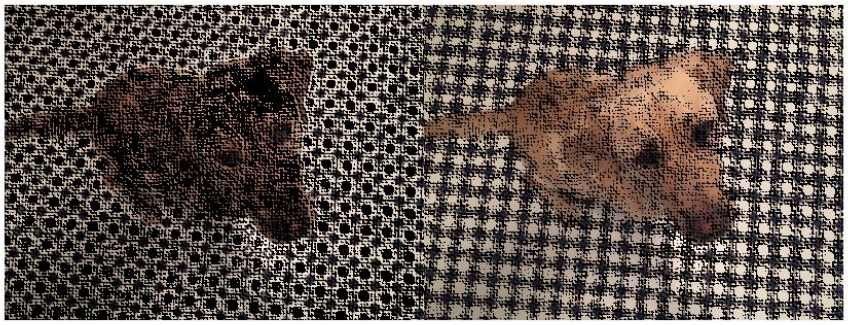


Figure 12: Running gabor segmentation with default parameters on Kobi

5 Conclusion

In this assignment three topics were covered: neighborhood processing, low-level filters and applications in image processing. The distribution of workload of this assignment is as follows: Marvin did Question 1, 4 and 5, consisting of 20 points, Bobbie answered question 2,3,8 and 9, consisting of 30 points, Milan did question 6 and 7, consisting of 30 points, Anne did question 10, consisting of 20 points.

References

[1] Bianconi, F., and Fernández, A. Evaluation of the effects of gabor filter parameters on texture classification. *Pattern Recogn. 40*, 12 (Dec. 2007), 3325–3335.

[2] Fogel, I., and Sagi, D. Gabor filters as texture discriminator. *Biological Cybernetics 61*, 2 (Jun 1989), 103–113.

[3] Hore, A., and Ziou, D. Image quality metrics: Psnr vs. ssim. In *2010 20th International Conference on Pattern Recognition* (2010), IEEE, pp. 2366–2369.

[4] Huynh-Thu, Q., and Ghanbari, M. Scope of validity of psnr in image/video quality assessment. *Electronics letters 44*, 13 (2008), 800–801.

[5] Marr, D., and Hildreth, E. Theory of Edge Detection. *Proceedings of the Royal Society of London Series B 207* (Feb. 1980), 187–217.

[6] Szeliski, R. *Computer Vision: Algorithms and Applications*. Springer, 2010.

[7] Yin, L., Yang, R., Gabbouj, M., and Neuvo, Y. Weighted median filters: a tutorial. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing 43*, 3 (1996), 157–192.

APPENDIX