Yuguang Yao
CSE

Homework 2

CSE 803
October 16, 2021

Problem 1

Image Patches

1. I chose the first three patches normalized and showed here.

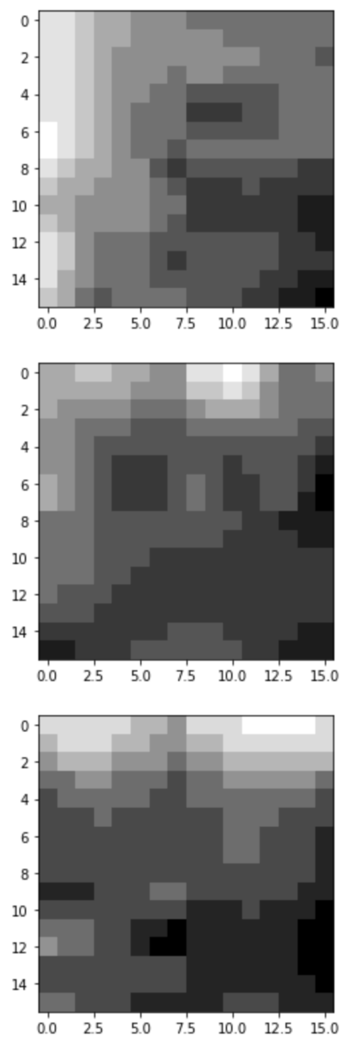


Figure 1: 3 patch

2. Advantages: Patches keep all the information in the original image. Different patch shows the detail in the different locations of the original image.

   Disadvantage: With the change of pose, scale, illumination, patches from the same object may differ a lot, this variance may affect the recognition of the same object.

Yuguang Yao
CSE

Homework 2

Gaussian Filter

1. Consider 2D sequence $f(x,y)$, then

   $G(x,y) * f(x,y) = \int\int G(u,v)f(x-u,y-v)dudv = \int\int G(u)G(v)f(x-u,y-v)dudv$
   $= G(x) * \int G(v)f(x,y-v)dv = G(x) * G(y) * f(x,y)$

   2D variance matrix has 1D variance values as its diagonal. 2D Gaussian filter equals applying 1D Gaussian filter for each vertical dimension and each horizontal dimension respectively.

2. Gaussian filter smooths the original image.

Figure 2: Gaussian filter

3. $k_x = [1/2, 0, -1/2]$

   $k_y = [1/2, 0, -1/2]^T$

4. Original edge detection detects more edges and details; Gaussian filtered edge detection detects blurred and weaker edges; Gaussian filtered edge detection focuses on the major edge.

(a) Gaussian filtered edge detection

(b) Original edge detection

Figure 3: Edge detection

Sobel Operator

1. $G_x \approx (I * k_{Gaussian}) * k_x = I * \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix} * \begin{pmatrix} \frac{1}{2} & 0 & -\frac{1}{2} \end{pmatrix} \approx I * \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}$

   $G_y \approx (I * k_{Gaussian}) * k_y = I * \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix} * \begin{pmatrix} \frac{1}{2} \\ 0 \\ -\frac{1}{2} \end{pmatrix} \approx I * \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$

2. Gx:

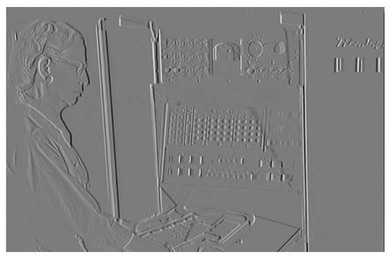


Figure 4: Gx

Gy:

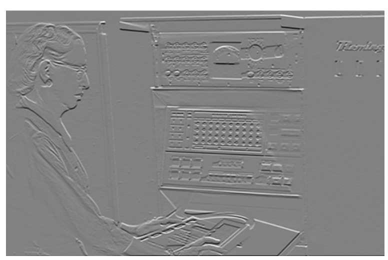


Figure 5: Gy

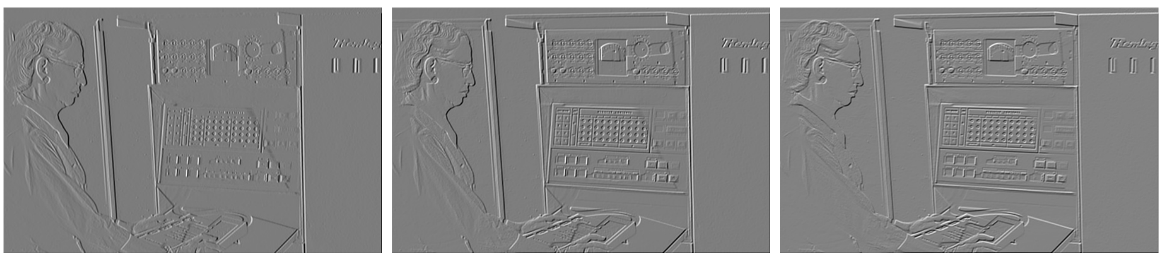
gradient_mag:

Figure 6: sobel

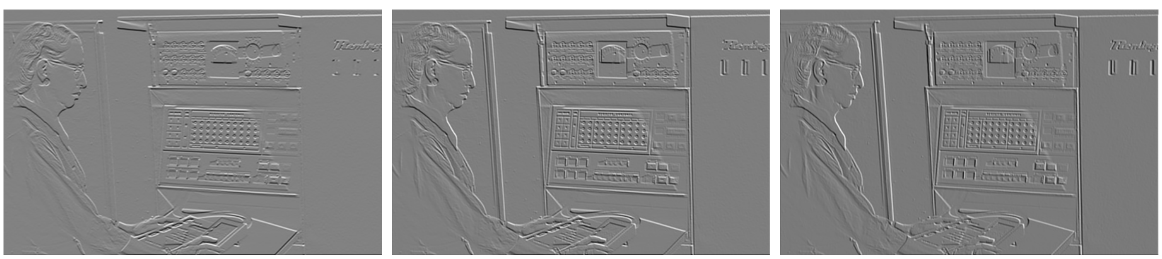
3. (a) $S(I,\alpha) = I*(G_x\cos\alpha + G_y\sin\alpha) \approx I*(\begin{pmatrix}1 & 0 & -1\\ 2 & 0 & -2\\ 1 & 0 & -1\end{pmatrix}\cos\alpha + \begin{pmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{pmatrix}\sin\alpha)$

   $K(\alpha) = \begin{pmatrix}\cos\alpha+\sin\alpha & 2\sin\alpha & -\cos\alpha+\sin\alpha\\ 2\cos\alpha & 0 & -2\cos\alpha\\ \cos\alpha-\sin\alpha & -2\sin\alpha & -\cos\alpha-\sin\alpha\end{pmatrix}$

   (b) Six different-angle filtered images:



(a) (b) (c)



(d) (e) (f)

Figure 7: $0 \sim \frac{5\pi}{6}$

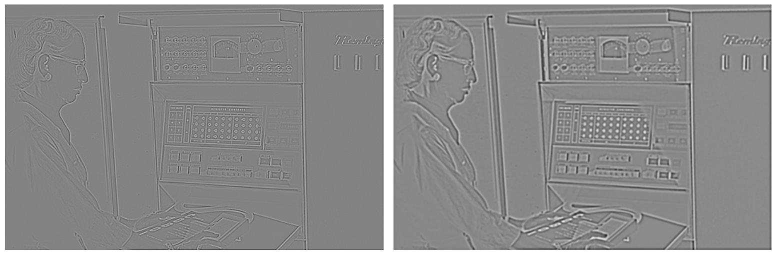
   (c) It detects the gradient towards different angles. $\alpha = 0$ refers to the gradient along x-axis and $\alpha = \frac{\pi}{2}$ refers to the one along y-axis.

Sobel Operator

1. The first output has more fine-grained edges while the second one is brighter but blurred.

   The reasons for this might be: the second kernel matrix is having a larger size and may consider more area in the original image which may result in a more informative but blurred output.

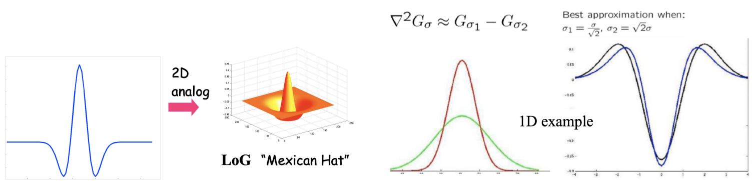
   These filters can detect edges obviously. By the way, they can also detect some corners, delivering lighter views like ears and buttons.



(a) (b)

Figure 8: LoG

2. As we can see, both LoG and DoG are really similar to Mexican Hat.



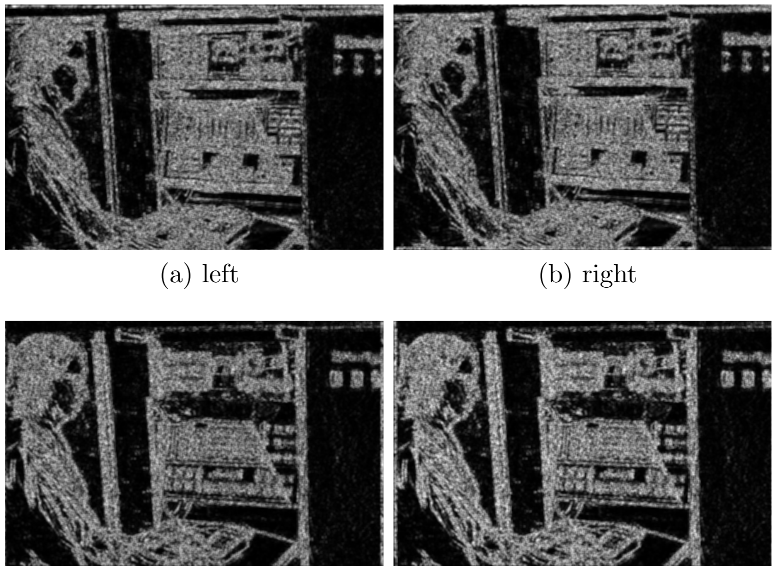
(a) LoG (b) DoG

Figure 9: LoG vs DoG

Problem 2

Corner Score

(a) With window size equal to $(5, 5)$, I plot the output that shift things left/right/up/down for 5 pixels.



(a) left (b) right (c) up (d) down

Figure 10: Corner Score

(b) Computation cost is really high even for only one offset given a small window size but for all the pixels. If iterating over different offsets, the corner might be found by a constantly high score, but the cost is too much.

Harris Corner Detector

To be mentioned, I use 5X5-size Gaussian kernel to sum each window up. Below is the result of the response generated by Harris Corner Detector using $\alpha = 0.05$:

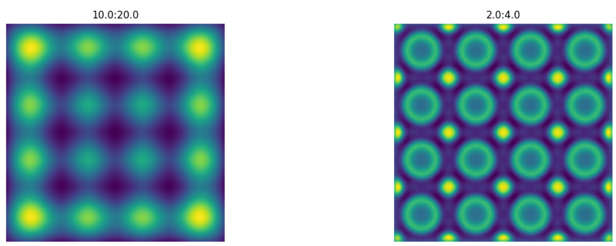
Figure 11: Harris Corner Detector

It is clear to see that the corners are brighter and the edges are darker while the others are flat.

Problem 3

Single-scale Blob Detection

I use $\sigma_1 = 10$ and $\sigma_2 = 20$ for large circle detection. I use $\sigma_1 = 2$ and $\sigma_2 = 4$ for small circle detection.



(a) Large (b) Small

Figure 12: Single-scale Blob Detection

I detect 16 maxima for large circles and 25 for small circles. I think there might be some high values in the small circle detection being actually the edges of large circles.

Scale Space Representation

I use $\sigma_{\min} = 2$. Yes. I am able to clearly see the different maxima in the image. In the fourth image 5.7:8.0 I can find 25 small maxima while in the last image 16.0:22.6 I can find 16 large maxima. Further in the fifth image 8.0:11.3 I can find both maxima clearly.

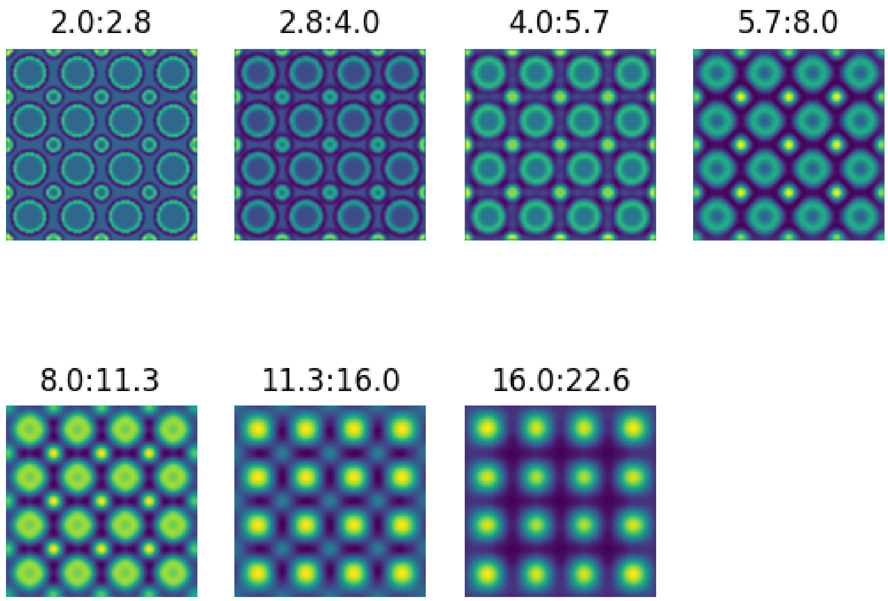
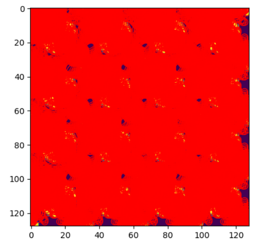
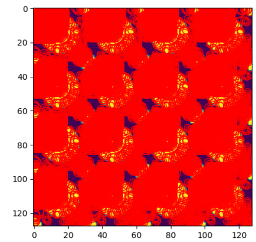


Figure 13: Scale space

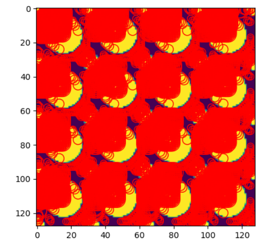
Blob Detection



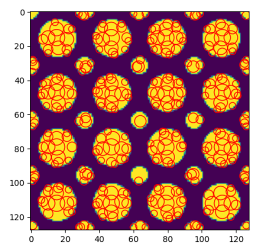
(a) kxy=1, ks=1



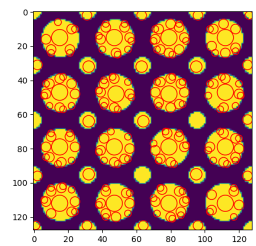
(b) kxy=1, ks=2



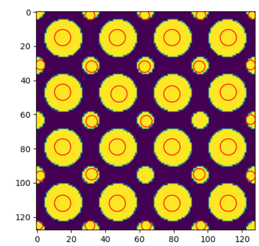
(c) kxy=1, ks=3



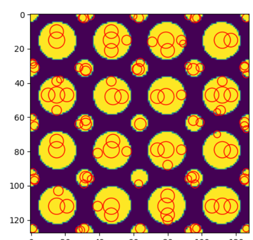
(d) kxy=5, ks=1



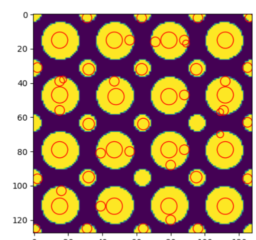
(e) kxy=5, ks=2



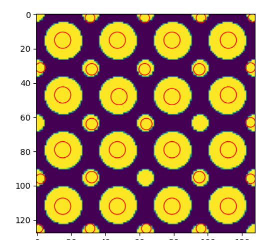
(f) kxy=5, ks=3



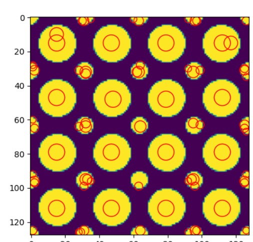
(g) kxy=9, ks=1



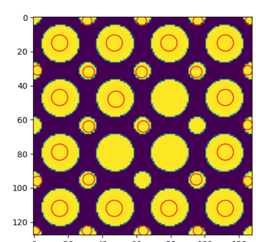
(h) kxy=9, ks=2



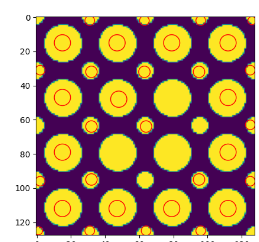
(i) kxy=9, ks=3



(j) kxy=15, ks=1



(k) kxy=15, ks=2



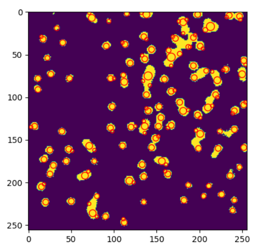
(l) kxy=15, ks=3

Figure 14: kxy = 1,5,9,15, ks = 1,2,3

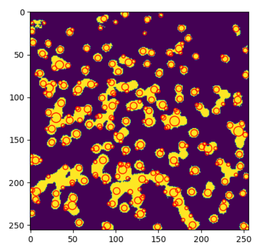
I tried kxy = 1,5,9,15 and ks = 1,2,3 respectively. Then I find the best choice is around kxy=9 and ks=3. With a smaller kxy and ks, there is a higher potential that a false positive is detected, as seen in the first row of pictures in the above figure.

Cell Counting

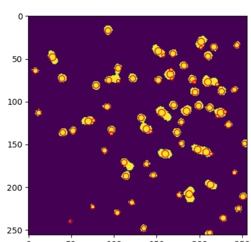
I chose cell1,2,4,5 as my targets. I chose $\sigma_{\min} = 2$, $k_{xy} = 8$, $k_s = 2$ and $k = \sqrt{2}$. To preprocess the image, I binarize each cell image with the threshold equal to the average of the highest pixel value and the lowest, i.e., $\frac{\min + \max}{2}$. If without the preprocess, there might be even more false positives due to the weak blob around in the background. In current counting, there are much less false positives to detect really small dark fake blobs although there are some missing blobs when many blobs are gathering.



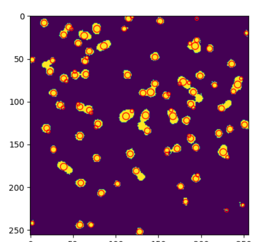
(a) 151 cells for cell 1



(b) 251 cells for cell 2



(c) 75 cells for cell 4



(d) 97 cells for cell 5

Figure 15: Cell Counting