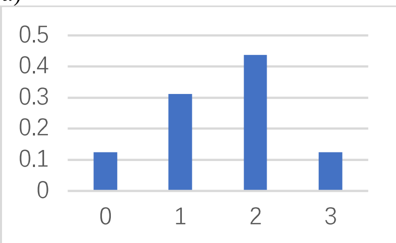
IMAGE PROCESSING- 2019 - SOLUTION

1. (10pt) For the image in the left figure (which only takes integer values of 0 to 3): a) (2pt) determine its histogram, b) (4pt) determine a transformation function that will attempt to equalize the histogram; c) (2pt) Show the equalized image using your transformation function in the right figure; d) (2pt) Show the histogram of the equalized image.

0	0	1	2
1	1	2	2
1	2	2	3
2	2	3	1

Solution:

a)



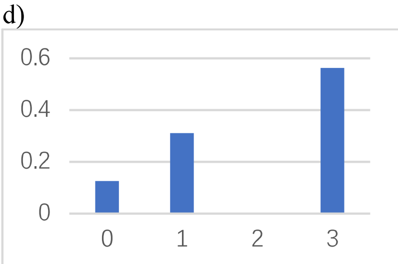
b)

	p	Σp	[Σp*3]
0	2/16	2/16	[6/16]=0
1	5/16	7/16	[21/16]=1
2	7/16	14/16	[42/16]=3
3	2/16	16/16	[48/16]=3

c)

0	0	1	3
1	1	3	3
1	3	3	3
3	3	3	1

d)



2. (10 pt) For the 2D filter H given below, where the center position corresponds to m=n=0: a) (2pt) Is this filter separable? If yes, what is the horizontal and vertical filter? b) (1pt) Based on the filter coefficients, can you tell what is the function of this filter overall, and what its function in the horizontal and vertical directions? c) (6pt) Determine the discrete space Fourier Transform $H(u,v)$ of the filter H, and sketch the one dimensional profiles $H(u,0)$, $H(u,1/2)$, $H(0,v)$, *and* $H(1/2,v)$. *You should use the property of separable functions for computing the DSFT when possible. Note that u represents the vertical frequency, v the horizontal frequency.* (d) (1pt) Based on the frequency response you derived, can you tell what is the function of the filter? Is this observation consistent with what you get in part (b)?

$$H = \begin{bmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{bmatrix}$$

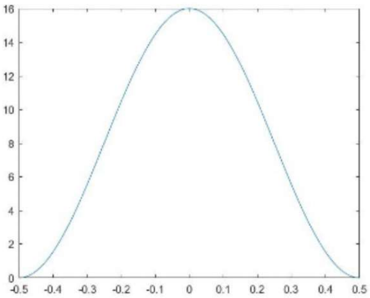
Solution:

a) Yes, this filter is separable. The horizontal filter is [-1 2 -1], and the vertical filter is [1 2 1].

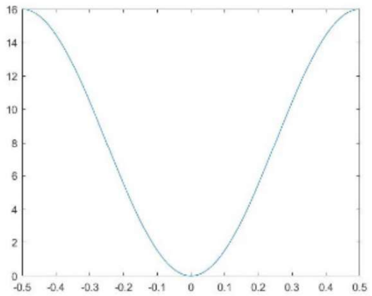
b) The filter is a vertical edge detector. It's low-pass in vertical direction, high-pass in horizontal direction.

c) H(u)=2+2cos2πu, H(v)= 2-2cos2πv. The filter is separable, so the Fourier transform H(u,v)= (2+2cos2πu)(2-2cos2πv)

H(u,0)=H(0.5,v)=0, H(u,0.5)=8(1+cos2πu), H(0,v)=8(1-cos2πv)



H(u,0.5):



H(0,v):

d) The same as part b.

3.

$$G_3 = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 6 \\ 3 & 4 & 5 & 6 \\ 4 & 6 & 6 & 6 \end{bmatrix} \longrightarrow G_2 = \begin{bmatrix} 2 & 4 \\ \frac{17}{4} & \frac{23}{4} \end{bmatrix} \longrightarrow G_1 = \begin{bmatrix} 4 \end{bmatrix}$$

$$UG_3 : \begin{bmatrix} 2 & 2 & 4 & 4 \\ 2 & 2 & 4 & 4 \\ \frac{17}{4} & \frac{17}{4} & \frac{23}{4} & \frac{23}{4} \\ \frac{17}{4} & \frac{17}{4} & \frac{23}{4} & \frac{23}{4} \end{bmatrix} \qquad L_3 = G_3 - UG_3 = \begin{bmatrix} -1 & 0 & -1 & 0 \\ 0 & -1 & 0 & -1 \\ -\frac{5}{4} & -\frac{1}{4} & \frac{-3}{4} & \frac{1}{4} \\ -\frac{1}{4} & \frac{7}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

$$UG_1 : \begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix} \qquad L_2 = G_2 - UG_2 = \begin{bmatrix} -2 & 0 \\ \frac{1}{4} & \frac{7}{4} \end{bmatrix}$$

Reconstruction:

① Upsample $\overset{G_1}{[4]}$: $\overset{UG_1}{\begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix}}$

② Find $G_2 = UG_1 + L_2 = \begin{bmatrix} 2 & 4 \\ \frac{17}{4} & \frac{23}{4} \end{bmatrix}$

③ Upsample $\overset{G_2}{\begin{bmatrix} 2 & 4 \\ \frac{17}{4} & \frac{23}{4} \end{bmatrix}}$: $\overset{UG_2}{\begin{bmatrix} 2 & 2 & 4 & 4 \\ 2 & 2 & 4 & 4 \\ \frac{17}{4} & \frac{17}{4} & \frac{23}{4} & \frac{23}{4} \\ \frac{17}{4} & \frac{17}{4} & \frac{23}{4} & \frac{23}{4} \end{bmatrix}}$

④ $G_3 = UG_2 + L_3$

$$= \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 6 & 6 & 6 \end{bmatrix}$$

4. (a) Let's assume the original image is $M \times N$, then

$x \in \mathbb{R}^{MN}$, $H \in \mathbb{R}^{MN \times MN}$.

$$\underbrace{\begin{bmatrix} \overbrace{1\ 1\ 0\ 0 \cdots 1}^{N \text{ pixel away}}\ 0\ 0 & \cdots & 0 \\ 1\ \overbrace{1\ 1\ 0 \cdots 0\ 1}^{N \text{ pixel away}} & & \\ & \ddots & \\ 0\ 0\ 0\ 0 \cdots & & \overbrace{1\ 0\ 0\ 1\ 1}^{N \text{ pixel away}} \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} x_{11} \\ x_{12} \\ \vdots \\ x_{1N} \\ \vdots \\ x_{M1} \\ \vdots \\ x_{MN} \end{bmatrix}}_{x} = b$$

In general for pixel i, j corresponds to the $(i-1)N + j^{th}$ row in H. $H = 1$ for following pixel:

$i, j \rightarrow ((i-1)N + j$

$i-1, j \rightarrow (i-2)N + j$ if $i-2 < 0$, 0

$i+1, j \rightarrow (i)N + j$ if $i \geq M$, 0

$i, j-1 \rightarrow (i-1)N + j - 1$ if $j-1 < 0$, 0

$i, j+1 \rightarrow (i-1)N + j + 1$ if $j+1 \geq N$, 0

pixel index in image　index in H $(i-1)N + j^{th}$ row

(b) $J = |b - Hx|^2$

$$\frac{\partial J}{\partial x} = 2 \cdot (-H^T)(b - Hx)$$

$$\frac{\partial J}{\partial x} = 0 \Leftrightarrow H^T(Hx - b) = 0$$

$$\Rightarrow H^T H x = H^T b$$

$$\Rightarrow x^* = (H^T H)^{-1} H^T b$$

(c) $\min_{z} |z|$

s.t. $z = b - Hx$

$$L_\rho(x, z, y) = |z| + y^T(Hx + z - b) + \frac{\rho}{2}(Hx + z - b)_2^2$$

$$\frac{\partial L}{\partial x} = H^T y + \rho H^T(Hx + z - b)$$

$$\frac{\partial L}{\partial x} = 0 \Leftrightarrow H^T y + \rho H^T H x + \rho H^T z - \rho H^T b = 0$$

$$\Leftrightarrow \rho H^T H x = \rho H^T b - \rho H^T z - H^T y$$

$$\Leftrightarrow \hat{x} = (H^T H)^{-1}(H^T b - H^T z - \frac{1}{\rho} H^T y)$$

$$\frac{\partial L}{\partial z} = \text{sign}(z) + y + \rho(Hx + z - b)$$

$$\frac{\partial L}{\partial z} = 0 \Leftrightarrow \text{sign}(z) + y + \rho(Hx + z - b) = 0$$

$$\Leftrightarrow Hx + z - b = -\frac{1}{\rho}\text{sign}(z) - \frac{1}{\rho} y$$

$$\Leftrightarrow z = b - Hx - \frac{1}{\rho} y - \frac{1}{\rho}\text{sign}(z)$$

$$\hat{z} = \text{Soft}(b - Hx - \frac{1}{\rho} y, \frac{1}{\rho})$$

So the iterative steps are:

$$x^{t+1} = (H^T H)^{-1}(H^T b - H^T z^t - \frac{1}{\rho} H^T y^t)$$

$$z^{t+1} = \text{Soft}(b - Hx^{t+1} - \frac{1}{\rho} y^t, \frac{1}{\rho})$$

$$y^{t+1} = y^t + \rho(Hx^{t+1} + z^{t+1} - b)$$

5) a) A point along an edge has change in only one direction. All points along the edge will have similar descriptor, this makes it undesirable for use as a feature, You cannot accurately match edge features in 2 images.

b) Harris feature detector detects points where there are large changes in more than one directions (i.e. corners).

c) It is intensity invariant because only the change in intensity between adjacent pixels (i.e. I_x I_y) is used to calculate the Harris score. An increase in intensity for all pixels in a patch will not affect this score. (local maxima)

Note that the gradient value can change if the image intensity is scaled by a factor. But since we detect local maxima of Harris image, it is invariant to intensity variation

d) It is rotation invariant because it is affected by the relative magnitude of the eigenvectors. Not how they are oriented. A corner ~~patch~~ will have large gradients in two different directions, no matter how it is rotated

e) It is scale variant. Because a corner will start to look like an edge if it is zoomed in.

f) It can be scale invariant by applying it on a Gaussian pyramid. Because a Gaussian pyramid has the image at different scales, corners that are scaled differently can be detected at one of the levels.

Then detect local maxima across scale

Problem 6:

(a) $\cancel{x}\ x = \frac{a_0+a_1u+a_2v}{c_0+c_1u+c_2v}$ $y = \frac{b_0+b_1u+b_2v}{c_0+c_1u+c_2v}$

Assuming $c_0 = 1$

then we can get

$a_0+a_1u+a_2v-c_1ux-c_2vx = x$

$b_0+b_1u+b_2v-c_1uy-c_2vy = y$

5

$$\begin{bmatrix} \cdots \\ \cdots \\ 1 & u_n & v_n & 0 & 0 & 0 & -u_nx_n & -v_nx_n \\ 0 & 0 & 0 & 1 & u_n & v_n & -u_ny_n & -v_ny_n \\ \cdots \\ \cdots \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ b_0 \\ b_1 \\ b_2 \\ c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} \vdots \\ x_n \\ y_n \\ \vdots \end{bmatrix} \Rightarrow Aa = x$$

As we get 4 pairs of corresponding feature points.

So A^TA is invertible. $A^TAa = A^Tx$

$a = (A^TA)^{-1}A^Tx$.

a is the homography mapping parameters.

b)

1. Given matching pairs (u_i, x_i), $i = 1, 2, \ldots, N$.
2. Randomly select K pairs from candidate pairs.
3. compute the geometric mapping parameters a.
4. apply resulting mapping $h(u, a)$ to all candidate pairs and evaluate the mapping error (Euclidean distance) at each pair

   $E_n = \|x_n - h(u_n, a)\|_2$ If $E_n > T$ consider pair n outlier.

   T is the inlier threshold.

8+2

5. count the number of inliers for this trial
6. ~~Return~~ Repeat process S times.
7. Return the mapping with the largest inliers count.
8. Refit the model using least square fit for all inlier points. Flowchart?

a)

7

```
% f1 : anchor frame ; f2: target frame; fp ; predicted image.
% mvx, mvy : store the MV image

f3 = imresize (f2, 2, 'bilinear')
for i = 1:N: h-N
    for j = 1:N: W-N
        MAD_min = 256 x N x N;
        mvx=0
        mvy =0
        for k= max(-i,-R) :0.5 : min(R, h-N-i)
            for l = max (-j ,-R) :0.5 : min (R , W-N-j)
                MAD= sum(sum (f1 (i:i+N-1, j:j+N-1)
                    - f3 (2*(i+k):2:2*(i+k+N-1), 2*(j+l):2:2*(j+l
                                                              +N-1))))
                if MAD_min > MAD
                    MAD_min = MAD
                    dy=k.
                    dx=l.
                end.
            end
        end.
        fp(i:i+N-1 , j:j+N-1 )=f3 (2*(i+dy):2: 2*(i+dy+N-1),
                                   2* (j+dx) :2: 2* (j+dx+N-1))
        iblk = (floor) (i-1)/N+1;
        jblk = (floor) (j-1) /N+1:
        mvx (iblk, jblk) = dx,
        mvy(iblk,jblk)=dy
```

7 b) $\frac{H}{N} \cdot \frac{W}{N} \cdot (4R+1)^2 \cdot N^2$

$= HW(4R+1)^2$

8)

(a) Trainable parameters. $\nearrow$ Bias $\rightarrow$ Batch Norm

$$3\times3\times3\times16 + \overset{(B)}{16} + \overset{(BN)}{32} = 480$$

$$16\times3\times3\times32 + \overset{(B)}{32} + \overset{(BN)}{64} = 4704$$

$$(32\times3\times3\times32) + \overset{(B)}{32} + \overset{(BN)}{64} = 9312$$

$$(64\times3\times3\times16) + \overset{(B)}{16} + \overset{(BN)}{32} = 9264$$

$$(32\times3\times3\times16) + \overset{(B)}{16} + \overset{(BN)}{32} = 4656$$

$$(16\times3\times3) + \overset{(B)}{1} = 145$$

28,561 # Params.

Para - B - BN = 28,224

Para - BN = 28337

We have considered both with bias and without bias solutions as correct, but deducted points for not including batch norm

(b) Perceptive field

(i) Top Layer: 3×3
(ii) Middle Layer: 8×8
(iii) Bottom Layer: 18×18

A

We have also considered the foll. Solution:

(i) Top Layer: 3×3
(ii) Middle Layer: 7×7
(iii) Bottom Layer: 15×15

B

A is the true Solution as the DownSampling is done using MaxPool & not standard Down Sampling

(c) Soft Dice Loss:

$$L = \frac{\sum_{i,j} P_{ij} G_{ij}}{\sum_{i,j} P_{ij} + \sum_{i,j} G_{ij}}$$

, $P_{ij} \rightarrow$ Prediction @ Pixel(i,j)

$G_{i,j} \rightarrow$ Ground Truth @ Pixel(i,j)

(c) we have also accepted Binary CrossEntropy Loss as a Loss function

(d) (i) Tackles the Problem of vanishing Gradients

(ii) Provides spatial Information from shallow Layers.

———x———x———x———x———

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(3, 16, 3)
        self.bn1 = nn.BatchNorm2d(16)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(16, 32, 3)
        self.bn2 = nn.BatchNorm2d(32)
        self.conv3 = nn.Conv2d(32,32,3)
        self.bn3 = nn.BatchNorm2d(32)
        self.up = nn.Upsample(scale_factor=2)
        self.conv4 = nn.Conv2d(64,16,3)
        self.bn4 = nn.BatchNorm2d(16)
        self.conv5 = nn.Conv2d(32,16,3)
        self.bn5 = nn.BatchNorm2d(16)
        self.conv6 = nn.Conv2d(16,1,3)

print("total parameters:  ",sum(p.numel() for p in net.parameters()))
```

Code to print N/W Parameters in PyTorch.