Computer Vision: Representation and Recognition
Assignment 1

161180038, 广进, guangjin1998@gmail.com

March 26, 2019

1 Linear Filters (30 points)

1.1 question1(a)

思路：使用卷积的 flip and shift 来解这道题更加直观，而不是用卷积公式来解：

step 1: pad I and flip F （padding 过程中，I 要上下各增加 $m-1=1$ 行，左右各增加 $n-1=1$ 行，其中 F 的大小是 $m\times n=2\times 2$）：

$$I_{padding}=\begin{bmatrix}0&0&0&0&0\\0&2&0&1&0\\0&1&-1&2&0\\0&0&0&0&0\end{bmatrix}, F_{flip}=\begin{bmatrix}-1&1\\-1&1\end{bmatrix} \tag{1.1}$$

step 2: F_{flip} 在 $I_{padding}$ 上移动，对应位置相乘再整体求和：示例如下：

$$I_{left\_top}=\begin{bmatrix}0&0\\0&2\end{bmatrix}, F_{flip}=\begin{bmatrix}-1&1\\-1&1\end{bmatrix} \Rightarrow 0*(-1)+0*1+0*(-1)+2*1=2 \tag{1.2}$$

step3: 就这样 F_{flip} 一个一个向右移动得到第一行结果：

$$\begin{bmatrix}2&-2&1&-1\end{bmatrix} \tag{1.3}$$

step4: 再把 F_{flip} 在 $I_{padding}$ 向下移动一行再做，以此类推，直到所有完成，得到最终答案：

$$I*F=\begin{bmatrix}2&-2&1&-1\\3&-4&4&-3\\1&-2&3&-2\end{bmatrix} \tag{1.4}$$

1.2 question1(b)

思路：类似于上一题的做法，采用 flip and shift 的计算方法

step 1: pad I and flip F_1（padding 过程中，I 只要上下各增加 1 行）：

$$I_{padding} = \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 1 \\ 1 & -1 & 2 \\ 0 & 0 & 0 \end{bmatrix}, F_{1\_flip} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{1.5}$$

step 2: F_{1_flip} 在 $I_{padding}$ 上移动，对应位置相乘再整体求和：示例如下：

$$I_{left\_top} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, F_{1\_flip} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Rightarrow 0*1+2*1=2 \tag{1.6}$$

step3: 就这样 F_{1_flip} 一个一个向右移动得到第一行结果：

$$\begin{bmatrix} 2 & 0 & 1 \end{bmatrix} \tag{1.7}$$

step4: 再把 F_{1_flip} 在 $I_{padding}$ 向下移动一行再做，以此类推，直到所有完成，得到最终答案：

$$I * F_1 = \begin{bmatrix} 2 & 0 & 1 \\ 3 & -1 & 3 \\ 1 & -1 & 2 \end{bmatrix} \tag{1.8}$$

下面计算 $(I * F_1) * F_2$：

step 1: pad $(I * F_1)$ and flip F_2（padding 过程中，$(I * F_1)$ 只要左右各增加 1 列）：

$$(I * F_1)_{padding} = \begin{bmatrix} 0 & 2 & 0 & 1 & 0 \\ 0 & 3 & -1 & 3 & 0 \\ 0 & 1 & -1 & 2 & 0 \end{bmatrix}, F_{2\_flip} = \begin{bmatrix} -1 & 1 \end{bmatrix} \tag{1.9}$$

step 2: F_{2_flip} 在 $(I * F_1)_{padding}$ 上移动，对应位置相乘再整体求和：示例如下：

$$(I * F_1)_{left\_top} = \begin{bmatrix} 0 & 2 \end{bmatrix}, F_{2\_flip} = \begin{bmatrix} -1 & 1 \end{bmatrix} \Rightarrow 0*(-1)+2*1=2 \tag{1.10}$$

step3: 就这样 F_{2_flip} 一个一个向右移动得到第一行结果：

$$\begin{bmatrix} 2 & -2 & 1 & -1 \end{bmatrix} \tag{1.11}$$

step4: 再把 F_{2_flip} 在 $(I * F_1)_{padding}$ 向下移动一行再做，以此类推，直到所有完成，得到最终答案：

$$(I * F_1) * F_2 = \begin{bmatrix} 2 & -2 & 1 & -1 \\ 3 & -4 & 4 & -3 \\ 1 & -2 & 3 & -2 \end{bmatrix} \tag{1.12}$$

1.3 question1(c)

思路：将左右都套卷积公式展开

左式：

$$(I * F)[i,j] = \sum_{k,l} I[i-k, j-l]F[k,l]$$

右式：

$$(I * F_1)[i,j] = \sum_{k} I[i-k, j]F_1[k,0]$$

$$\begin{aligned}((I * F_1) * F_2)[i,j] &= \sum_{l}(I * F_1)[i, j-l]F_2[0,l] \\ &= \sum_{l}(\sum_{k} I[i-k, j-l]F_1[k,0])F_2[0,l] \\ &= \sum_{l}(\sum_{k} I[i-k, j-l]F_1[k,0]F_2[0,l]) \\ &= \sum_{k,l} I[i-k, j-l](F_1[k,0]F_2[0,l])) \\ &= \sum_{k,l} I[i-k, j-l]F[k,l] \\ &= (I * F)[i,j]\end{aligned}$$

证毕

1.4 question1(d)

思路：首先推出一个对于计算 flip and shift 乘法数目的一般性结论，再带入解决以下小问。

Theorem 1. 对于 $I_{a\times b}$ $F_{m\times n}$，如果要 *flip and shift* F，一共会做 $[(a+m-1)\times(b+n-1)]\times(m\times n)$ 次乘法运算。

Proof. 对于 $I_{a\times b}$ $F_{m\times n}$，如果要 flip and shift F，则 I 需要 pad：I 上下各增加 $m-1$ 行，左右各增加 $n-1$ 列，进而 $I_{padding}$ 的大小为 $(a+(m-1)\times 2)\times(b+(n-1)\times 2)$。所以，$F$ 每行会滑动 $(a+(m-1)\times 2)-(m-1) = a+m-1$ 次，每列会滑动 $(b+(n-1)\times 2)-(n-1) = b+n-1$ 次，一共会做 $(a+m-1)\times(b+n-1)$ 个框的运算。而每个框大小为 F 的大小，会做 $m\times n$ 次乘法，所以一共会做 $[(a+m-1)\times(b+n-1)]\times(m\times n)$ 次乘法运算。 □

(i) 如果 flip and shift I，应用 Theorem 1，需要做 $[(2+2-1)\times(3+2-1)]\times(2\times 3) = 12\times 6 = 72$ 次乘法运算。

(ii) 不相同。因为如果 flip and shift F，应用 Theorem 1，需要做 $[(2+2-1)\times(3+2-1)]\times(2\times 2) = 12\times 4 = 48$ 次乘法运算。

(iii) 对于 part(b), $(I*F_1)$ 我们选择乘法运算次数更少的, 即 flip and shift F_1, 应用 Theorem 1, 需要做 $[(2+2-1)\times(3+1-1)]\times(2\times1) = 9\times2 = 18$ 次乘法运算; $(I*F_1)*F_2$ 我们选择乘法运算次数更少的, 即 flip and shift F_2, 应用 Theorem 1, 需要做 $[(3+1-1)\times(3+2-1)]\times(1\times2) = 12\times2 = 24$ 次乘法运算。一共要做 $18+24 = 42$ 次乘法运算。

(iv) 由于 $42 < 48$, method (b) 需要更少的计算次数。

1.5 question1(e)

(i) 一般 image 比 filter 大, 所以我们选择乘法次数少的, 即 flip and shift F, 应用 Theorem 1, 需要做 $[(M_1+M_2-1)\times(N_1+N_2-1)]\times(M_2\times N_2)$ 次乘法运算; 而 flip and shift I, 应用 Theorem 1, 需要做 $[(M_1+M_2-1)\times(N_1+N_2-1)]\times(M_2\times N_2)$ 次乘法运算。

(ii) 设 $F = F_1F_2, where F_1 : M_2\times1 and F_2 : 1\times N_2$, 我们选择乘法次数少的, 即 flip and shift F_1 和 F_2。首先计算 $(I*F_1)$ 应用 Theorem 1, 需要做 $[(M_1+M_2-1)\times(N_1+1-1)]\times(M_2\times1) = (M_1+M_2-1)\times N_1\times M_2$ 次乘法运算, 得到 $(I*F_1)$ $: [M_1+M_2-1]\times N_1$ (见 Theorem 1的证明, 最后的框数)。然后计算 $(I*F_1)*F_2$ 应用 Theorem 1, 需要做 $[([M_1+M_2-1]+1-1)\times(N_1+N_2-1)]\times(1\times N_2) = (M_1+M_2-1)\times(N_1+N_2-1)\times N_2$ 次乘法运算。所以一共需要 $(M_1+M_2-1)\times N_1\times M_2 + (M_1+M_2-1)\times(N_1+N_2-1)\times N_2 = (M_1+M_2-1)\times[N_1\times(M_2+N_2)+(N_2-1)\times N_2]$

(iii) 对于直接 2D 卷积:

$$\begin{aligned}&[(M_1+M_2-1)\times(N_1+N_2-1)]\times(M_2\times N_2)\\&= O((M_1+M_2)(N_1+N_2)M_2N_2\\&= O(M_1N_1M_2N_2 + M_1M_2N_2^2 + M_2^2N_1N_2 + M_2^2N_2^2)\end{aligned}$$

两次连续 1D 卷积:

$$\begin{aligned}&(M_1+M_2-1)\times[N_1\times(M_2+N_2)+(N_2-1)\times N_2]\\&= O((M_1+M_2)\times[N_1\times(M_2+N_2)+N_2^2])\\&= O(M_1M_2N_1 + M_2^2N_1 + M_1N_1N_2 + M_2N_1N_2 + M_1N_2^2 + M_2N_2^2)\end{aligned}$$

由于

$$\begin{aligned}&O(M_1M_2N_1 + M_1N_1N_2 + M_2N_1N_2) = O(M_1N_1M_2N_2)\\&O(M_2^2N_1) = O(M_2^2N_1N_2) \quad O(M_1N_2^2 + M_2N_2^2) = O(M_1M_2N_2^2)\end{aligned}$$

所以两次连续 1D 卷积更有效率:

$$\begin{aligned}&O(M_1M_2N_1 + M_2^2N_1 + M_1N_1N_2 + M_2N_1N_2 + M_1N_2^2 + M_2N_2^2)\\&= O(M_1N_1M_2N_2 + M_1M_2N_2^2 + M_2^2N_1N_2 + M_2^2N_2^2)\end{aligned}$$

2 Question 2

观看效果，采用如下滤波器：

$$h_0 = \begin{bmatrix} 0.0625 & 0.125 & 0.0625 \\ 0.125 & 0.25 & 0.125 \\ 0.0625 & 0.125 & 0.0625 \end{bmatrix}, h_1 = \begin{bmatrix} -1 & -2 & 0 & 2 & 1 \\ -2 & -4 & 0 & 4 & 2 \\ -1 & -2 & 0 & 2 & 1 \end{bmatrix}, h_2 = \begin{bmatrix} -1 & -2 & -1 \\ -2 & -4 & -2 \\ 0 & 0 & 0 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.1}$$

效果如下：

Figure 1: Lenna 灰度图

Figure 2: 经过高斯滤波器 h_0 卷积的效果

Figure 3: 经过滤波器 h_1 卷积的效果

Figure 4: 经过滤波器 h_2 卷积的效果

3 Question 3

3.1 Global Histogram Equalization (三张图片示例)

3.1.1 Lenna 图片

(i) 转化为灰度图，采用书上 (2.112) 转换公式：$Y_{601} = 0.299R + 0.587G + 0.114B$。转化后见图 5：

(ii) 原图的直方图见图 6 ，累积分布函数见图 7 ，compensation transfer function 为 $f(I) = \alpha \times c(I) + (1-\alpha) \times I$（局部补偿，见课本公式 (3.9) 后面），最终直方图均衡化（$f(I) = c(I)$ $\alpha = 1$）结果见图 8，其直方图和累积分布函数分别见图 9 和图 10 。图 11 图 12 和图 13 分别是 α 分别为 0.75, 0.5, 0.25 的图片。

Figure 5: Lenna 灰度图

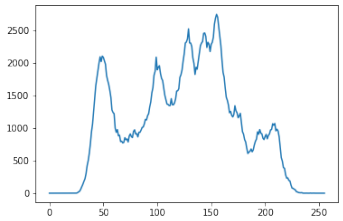

Figure 6: Lenna 直方图

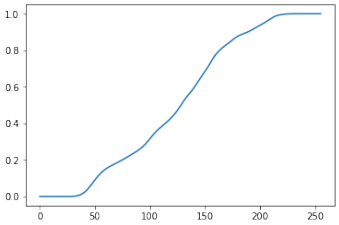

Figure 7: 累积分布函数

Figure 8: Lenna $\alpha = 1.00$

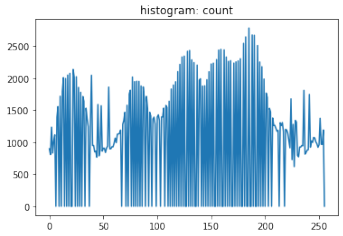

Figure 9: Lenna 直方图

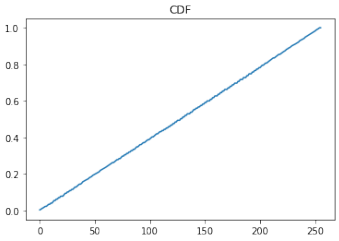

Figure 10: 累积分布函数

Figure 11: $\alpha = 0.75$

Figure 12: $\alpha = 0.50$

Figure 13: $\alpha = 0.25$

(iii) punch: 一般化，假如 pixel 前 $a\%$ 的变为纯白，pixel 后 $b\%$ 的变为纯黑，则有：

$$compensation\ transfer\ function\ f(I) = \begin{cases} 0 & if\ c(I) < b\% \\ \dfrac{c(I) - b\%}{1 - a\% - b\%} & if\ b\% < c(I) < 1 - a\% \\ 1 & if\ c(I) > 1 - a\% \end{cases} \tag{3.1}$$

punch 纯黑纯白各 $5\%, 25\%, 40\%$ 分别见图 14 、图 15 和图 16 。

Figure 14: punch 5% - 95%

Figure 15: punch 25% - 75%

Figure 16: punch 40% - 60%

(iv) limit local gain 就是要限制 transfer function $F(I) = cdf[I] < \lambda I$, 也就是限制其倒数 $f'(x) < \lambda$。由于 cdf 是离散的，倒数用差分代替，而由于公式 (3.9)，cdf 的差分便是 $\frac{1}{N}h(I)$，再将截去的那些 $h(I)$ 平均分布在每个 I 上。分别将 $N \times \lambda$ 限制在 10，100，1000，对于 Lenna 区别不是很大，见下图：

Figure 17: $\lambda = 10$

Figure 18: $\lambda = 100$

Figure 19: $\lambda = 1000$

(v) gray image to color 方法如下：由书上公式 (2.116)，$r = \dfrac{R}{R+G+B}$,$g = \dfrac{G}{R+G+B}$,$b = \dfrac{B}{R+G+B}$ 和灰度转换公式 (2.112):$Y_{601} = 0.299R + 0.587G + 0.114B$，我们假设在直方图均衡化等处理中，不改变颜色比率。我们可以得到：

$$\begin{aligned} Y' &= 0.299R' + 0.587G' + 0.114B' \\ &= (0.299r + 0.587g + 0.114b)(R' + G' + B') \\ &= (0.299R + 0.587G + 0.114B)\frac{R' + G' + B'}{R + G + B} \\ &= Y\frac{R' + G' + B'}{R + G + B} \end{aligned}$$

从而有 $R' = r(R'+G'+B') = r\dfrac{Y'}{Y}(R+G+B) = \dfrac{Y'}{Y}R$，同理 $G' = \dfrac{Y'}{Y}G$，$B' = \dfrac{Y'}{Y}B$。直方图均衡化、punch 纯黑纯白各 25% 以及 $\lambda = 10$，分别见下图：

Figure 20: 直方图均衡化

Figure 21: punch 纯黑纯白各 25%

Figure 22: $\lambda = 10$

3.1.2 night 图片

(i) 转化为灰度图，转化后见图 23 ：

(ii) 原图的直方图见图 24 ，累积分布函数见图 25 ，compensation transfer function 为 $f(I) = \alpha\times c(I)+(1-\alpha)\times I$（局部补偿，见课本公式 (3.9) 后面），最终直方图均衡化（$f(I) = c(I)$ $\alpha = 1$）结果见图 26，其直方图和累积分布函数分别见图 27 和图 28 。

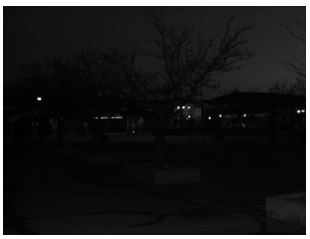

Figure 23: night 灰度图

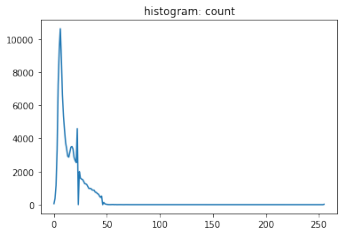

Figure 24: night 直方图

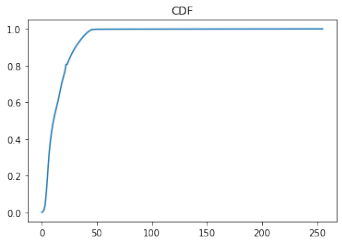

Figure 25: 累积分布函数

Figure 26: night $\alpha = 1.00$

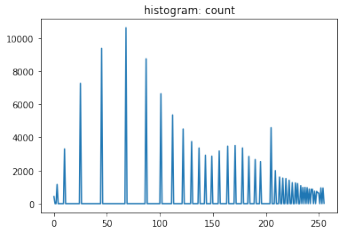

Figure 27: night 直方图

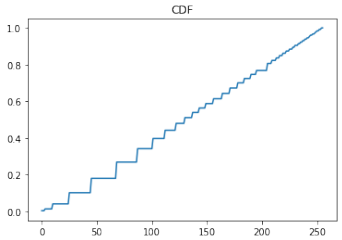

Figure 28: 累积分布函数

图 29 图 30 和图 31 分别是 α 分别为 0.75,0.5,0.25 的图片。

Figure 29: $\alpha = 0.75$

Figure 30: $\alpha = 0.50$

Figure 31: $\alpha = 0.25$

(iii) punch: punch 纯黑纯白各 5%, 25%, 40% 分别见图 32 、图 33 和图 34 。

Figure 32: punch 5% - 95%

Figure 33: punch 25% - 75%

Figure 34: punch 40% - 60%

(iv) limit local gain: 分别将 λN 限制在 1000，2000，5000，见下图：

Figure 35: $\lambda = 1000$

Figure 36: $\lambda = 2000$

Figure 37: $\lambda = 5000$

(v) gray image to color: 由于是灰色图，不适用

3.1.3 car 图片

(i) 转化为灰度图，转化后见图 38 ：

(ii) 原图的直方图见图 39 ，累积分布函数见图 40 ，compensation transfer function 为 $f(I) = \alpha \times c(I) + (1-\alpha) \times I$ (局部补偿, 见课本公式 (3.9) 后面), 最终直方图均衡化 ($f(I) = c(I)$ $\alpha = 1$) 结果见图 41，其直方图和累积分布函数分别见图 42 和图 43 。图 44 图 45 和图 46 分别是 α 分别为 0.75,0.5,0.25 的图片。

Figure 38: car 灰度图

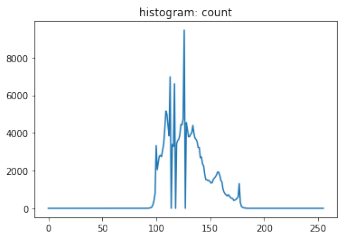

Figure 39: car 直方图

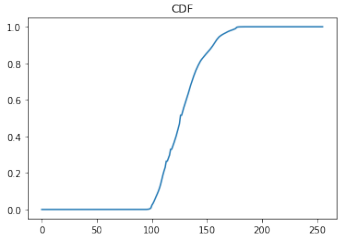

Figure 40: 累积分布函数

Figure 41: car $\alpha = 1.00$

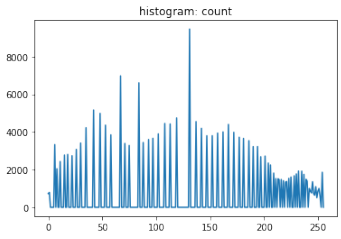

Figure 42: car 直方图

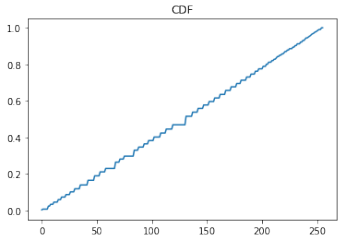

Figure 43: 累积分布函数

Figure 44: $\alpha = 0.75$

Figure 45: $\alpha = 0.50$

Figure 46: $\alpha = 0.25$

(iii) punch: punch 纯黑纯白各 5%, 25%, 40% 分别见图 47 、图 48 和图 49 。

Figure 47: punch 5% - 95%

Figure 48: punch 25% - 75%

Figure 49: punch 40% - 60%

(iv) limit local gain: 分别将 λ 限制在 100，1000，2000，见下图：

Figure 50: $\lambda = 100$

Figure 51: $\lambda = 1000$

Figure 52: $\lambda = 2000$

(v) gray image to color: 由于是灰色图，不适用

3.2 Local Histogram Equalization

在 Exercise 3.1 基础上，首先将图片转为灰色，对于每个图片分块进行直方图均衡化得到每块的 lookup table，再借助双线性插值法得到每个 pixel 的像素值（不使用低通滤波器）。

1. 使用 Lenna 图像进行展示，下面分别为 block 为 16*16，32*32，64*64 得到的局部直方图均衡化图像：

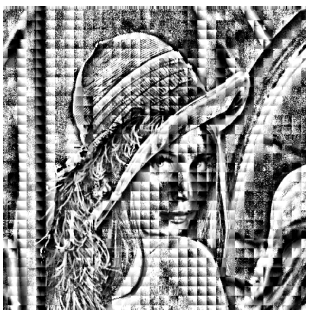

Figure 53: $block = 16 * 16$

Figure 54: $block = 32 * 32$

Figure 55: $block = 64 * 64$

2. 使用 night 图像进行展示，下面分别为 block 为 16*16，32*32，64*64 得到的局部直方图均衡化图像：

Figure 56: $block = 16 * 16$

Figure 57: $block = 32 * 32$

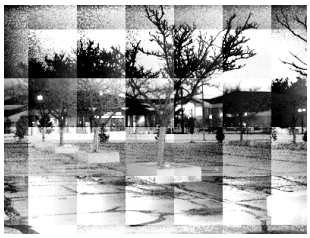

Figure 58: $block = 64 * 64$

3. 使用 car 图像进行展示，下面分别为 block 为 16*16，32*32，64*64 得到的局部直方图均衡化图像：

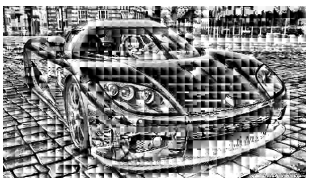

Figure 59: $block = 16 * 16$

Figure 60: $block = 32 * 32$

Figure 61: $block = 64 * 64$