图像处理与可视化 Homework 03

学号: 21307140051
姓名: 雍崔扬

Problem 1

编程实现图像域基于空间滤波器的平滑和锐化算法.

(1) 平滑

最简单的可分离低通滤波器核是**盒式核**，其系数的值相同 (通常为 1) (因此其秩为 1，故而可分离)
其前面还有一个归一化系数 (通常为 $\frac{1}{n^2}$, 如果尺寸为 $n \times n$ 的话)

Gauss 核 $w(s,t) = G(s,t) = K \exp\{-\frac{s^2+t^2}{2\sigma^2}\}$ 是唯一可分离的各向同性核.
因此 Gauss 核的计算速度不仅可与盒式滤波器媲美，而且还具有很多适合于图像处理的有用性质.
记 $r = \sqrt{s^2+t^2}$，则 $G(s,t) = K \exp\{-\frac{s^2+t^2}{2\sigma^2}\}$ 可改写为 $G(r) = K \exp\{-\frac{r^2}{2\sigma^2}\}$

可分离核 (separable kernel) $w \in \mathbb{R}^{n\times m}$ 即一个可表示为两个向量 $w_1 \in \mathbb{R}^n$ 和 $w_2 \in \mathbb{R}^m$ 外积 $w_1 w_2^T$ 的矩阵.
事实上，w_1, w_2 可以看作两个核，而 w 可以看作这两个核的二维卷积.
此时 $w \star f = (w_1 \star w_2^T) \star f = (w_2^T \star w_1) \star f = w_2^T \star (w_1 \star f) = (w_1 \star f) \star w_2^T$
这表明一幅图像与一个可分离核 $w = (w_1 \star w_2^T)$ 的卷积，等于先用 f 与 w_1 卷积，再用 w_2^T 对结果进行卷积.
这样的分解有助于减少计算复杂度:

- 一方面，$w \star f$ 的计算复杂度为 $O(MNmn)$，结果的尺寸为 $M \times N$
- 另一方面，第一次卷积 $w_1 \star f$ 的计算复杂度为 $O(MNn)$，中间结果的尺寸为 $M \times N$
  第二次卷积 $(w_1 \star f) \star w_2^T$ 的计算复杂度为 $O(MNm)$，结果的尺寸为 $M \times N$
  总计算复杂度为 $O(MN(m+n))$

生成平滑核的函数 `smooth_filter`:

```
def smooth_filter(n, type="box", sigma=1, scale=1):
    """生成平滑滤波器核。

    参数:
        n (int): 核的大小，必须是奇数。
        type (str): 滤波器类型，可以是 "box" 或 "gaussian"。
        sigma (float): 高斯滤波器的标准差，仅在类型为 "gaussian" 时使用。
        scale (float): 高斯核的缩放因子，仅在类型为 "gaussian" 时使用。

    返回:
        tuple: 包含一维权重 w1, w2 和二维滤波器核。
    """
    assert n % 2 == 1, "n 必须是奇数"

    if type == "box":
        # 盒式滤波器
        w1 = np.ones(n).reshape(-1, 1) / n
        w2 = w1.copy()
        kernel = np.outer(w1, w2)  # 计算外积，形成二维核
    else:
        if n > np.ceil(6 * np.sqrt(sigma)):
            print("Warning: A width of ceil(6 * sigma)^2 may be too large.")
            print("For Gaussian kernels, we generally use a size of ⌈6σ⌉ × ⌈6σ⌉.")
            print("This is because the value of the Gaussian function becomes negligible for r ≥ 3σ.")
            print("Therefore, a Gaussian kernel of size ⌈6σ⌉ × ⌈6σ⌉ has a similar effect to a larger-sized kernel.")
            print("Since we typically work with odd-sized kernels, we select the smallest odd number greater than 6σ.")
            print("(For example, when σ = 7, we use the smallest odd number greater than 6σ = 42, which is 43.)")
```

```
        # Gauss 滤波器
        w1 = np.zeros(n).reshape(-1, 1)  # 初始化一维 Gauss 核
        for x in range(n):
            x_coord = x - (n - 1) / 2  # 中心化坐标
            w1[x] = np.exp(-(x_coord**2) / (2 * sigma**2))
        kernel = np.outer(w1, w1)
        w1 = np.sqrt(scale) * w1 / np.sqrt(np.sum(kernel))
        w2 = w1.copy()
        kernel = kernel / np.sum(kernel)  # 计算外积，形成二维核

    return w1, w2, kernel
```

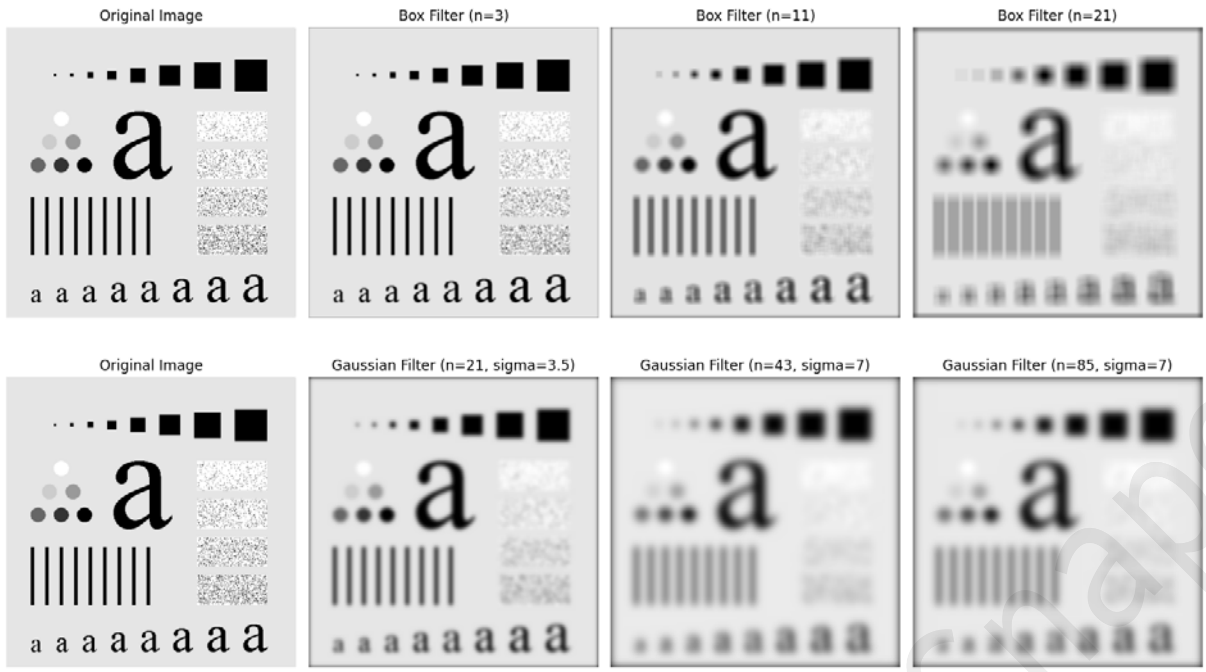
应用平滑核并展示平滑图像的函数 `apply_and_plot`：

```
def apply_and_plot(image_array, filter_sizes, filter_type, sigma=None):
    """应用平滑滤波器并绘制结果图像。"""
    plt.figure(figsize=(15, 10))
    plt.subplot(1, len(filter_sizes) + 1, 1)
    plt.title("Original Image")
    plt.imshow(image_array, cmap='gray')
    plt.axis('off')

    for i, n in enumerate(filter_sizes):
        if filter_type == "box":
            w1, w2, _ = smooth_filter(n, type="box")
        else:
            # 确保 sigma 是一个向量，并引用相应的值
            current_sigma = sigma[i] if sigma is not None else 1
            w1, w2, _ = smooth_filter(n, type="gaussian", sigma=current_sigma)

        # 应用卷积
        intermediate_image = convolve(image_array, w1, mode='constant', cval=0)
        smoothed_image = convolve(intermediate_image, w2.T, mode='constant', cval=0)

        plt.subplot(1, len(filter_sizes) + 1, i + 2)
        if filter_type == "box":
            plt.title(f"{filter_type.capitalize()} Filter (n={n})")
        else:
            plt.title(f"{filter_type.capitalize()} Filter (n={n}, sigma={current_sigma})")
        plt.imshow(smoothed_image, cmap='gray')
        plt.axis('off')

    plt.tight_layout()
    plt.show()
```

函数调用:

```
if __name__ == "__main__":
    # 读取图像
    image_path = 'Fig 03.33(a) (test_pattern_blurring_orig).tif'
    img = Image.open(image_path).convert('L')  # 转换为灰度图像
    image_array = np.array(img)

    # 盒式核平滑
    box_filter_sizes = [3, 11, 21]
    apply_and_plot(image_array, box_filter_sizes, filter_type="box")

    # 高斯核平滑
    gaussian_filter_sizes = [21, 43, 85]
    gaussian_sigmas = [3.5, 7, 7]
    apply_and_plot(image_array, gaussian_filter_sizes, filter_type="gaussian",
sigma=gaussian_sigmas)
```

运行结果:



(2) 锐化

引入对角方向的 Laplace 算子为:

$$\begin{aligned}\nabla^2 f(x,y) := & f(x+1,y)+f(x-1,y)+f(x,y+1)+f(x,y-1)\\ & +f(x-1,y-1)+f(x-1,y+1)+f(x+1,y-1)+f(x+1,y+1)\\ & -8f(x,y)\end{aligned}$$

实现上述公式的 Laplace 核为:

1	1	1
1	−8	1
1	1	1

b

−1	−1	−1
−1	8	−1
−1	−1	−1

d

FIGURE 3.45 (b) Kernel used to implement an extension of this equation that includes the diagonal terms.

符号取反的 Laplace 核会产生等效的结果,
但在组合 Laplace 滤波后的图像与另一幅图像时，必须考虑符号的差异.
当使用具有**负中心系数**的 Laplace 核时，原图像**减去** Laplace 滤波后的图像就可 "恢复" 背景特征，同时保留 Laplace 的锐化效果.
当使用具有**正中心系数**的 Laplace 核时，原图像**加上** Laplace 滤波后的图像就可 "恢复" 背景特征，同时保留 Laplace 的锐化效果.

$$g(x,y) = f(x,y) + \mathrm{sgn}(\nabla^2_{(2,2)}) \cdot [\nabla^2 f(x,y)]$$

其中 $\mathrm{sgn}(\nabla^2_{(2,2)})$ 代表 Laplace 算子 ∇^2 的表示矩阵的中心位置 (即 $(2,2)$ 位置) 系数的符号.

实现 Laplace 锐化的函数 `laplace_sharpening`:

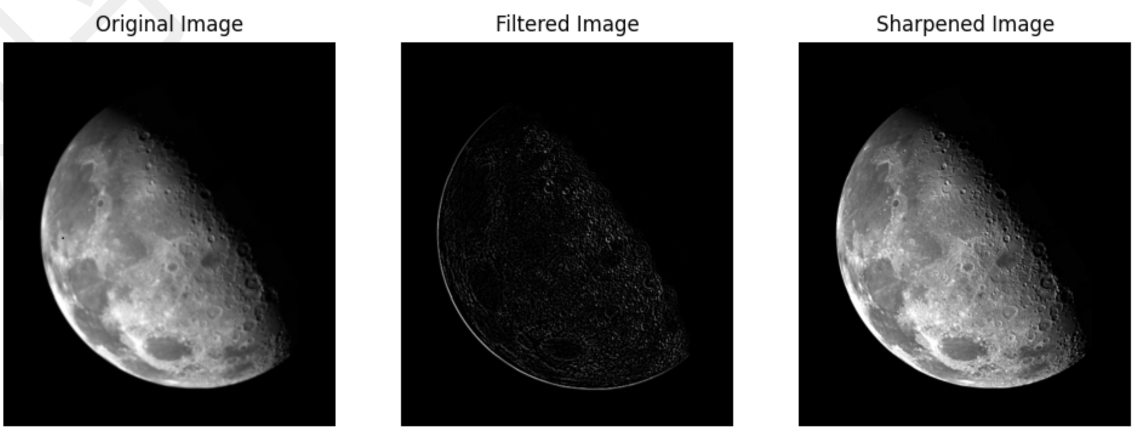
```
def laplace_sharpening(image_array):
    # 定义 Laplace 核
```

```
    laplace_kernel = np.array([[0, 1, 0],
                               [1,-4, 1],
                               [0, 1, 0]], dtype=np.float32)

    # 应用 Laplace 核
    laplace_filtered  = convolve(image_array.astype(float), laplace_kernel, mode='reflect')

    # Laplace 核具有负中心系数, 故原图像应当减去拉伸后的滤波图像得到锐化图像
    sharpened_image = image_array.astype(float) - laplace_filtered

    # 确保像素值在有效范围内
    laplace_filtered= np.clip(laplace_filtered, 0, 255).astype(np.uint8)
    sharpened_image = np.clip(sharpened_image, 0, 255).astype(np.uint8)

    return laplace_filtered, sharpened_image
```

函数调用:

```
if __name__ == "__main__":
    # 读取图像
    image_path = 'Fig 03.38(a)(blurry_moon).tif'
    img = Image.open(image_path).convert('L')  # 转换为灰度图像
    image_array = np.array(img)

    # 应用锐化
    laplace_filtered, sharpened_image = laplace_sharpening(image_array)

    # 显示结果
    plt.figure(figsize=(12, 6))
    plt.subplot(1, 3, 1)
    plt.title('Original Image')
    plt.imshow(image_array, cmap='gray')
    plt.axis('off')

    plt.subplot(1, 3, 2)
    plt.title('Filtered Image')
    plt.imshow(laplace_filtered, cmap='gray')
    plt.axis('off')

    plt.subplot(1, 3, 3)
    plt.title('Sharpened Image')
    plt.imshow(sharpened_image, cmap='gray')
    plt.axis('off')

    plt.show()
```

运行结果:



Problem 2

Part (1)

试证明冲击串 (impulse train) 的傅里叶变换后的频域表达式也是一个冲击串.

Proof:

- ① 首先考虑 $t = t_0$ 处的 Dirac 函数 $\delta(t-t_0)$ 的 Fourier 变换:

$$\begin{aligned}\tilde{\delta}(\mu) &= \int_{-\infty}^{\infty} \delta(t-t_0)\exp\{-i(2\pi\mu)t\}dt \\ &= \exp\{-i(2\pi\mu)t_0\}\end{aligned}$$

  上述过程用到了 Dirac 函数 $\delta(\cdot)$ 的取样性质.

- ② 其次考虑 $e(t) = \exp\{i\omega_0 t\} = \exp\left(i(2\pi\mu_0)t\right)$ 的 Fourier 变换:

$$\begin{aligned}\tilde{e}(\mu) &= \int_{-\infty}^{\infty} e(t)\exp\{-i(2\pi\mu)t\}dt \\ &= \int_{-\infty}^{\infty} \exp\{i(2\pi\mu_0)t\}\exp\{-i(2\pi\mu)t\}dt \\ &= \int_{-\infty}^{\infty} \exp\{i\cdot 2\pi(\mu_0-\mu)t\}dt \quad (\text{note that } \delta(\mu) := \int_{-\infty}^{\infty} \exp\left(i(2\pi\mu)t\right)dt) \\ &= \delta(\mu-\mu_0)\end{aligned}$$

- ③ 现考虑时间间隔 ΔT 的冲激串 $s_{\Delta T}(t) := \sum_{k=-\infty}^{\infty}\delta(t-k\Delta T)$ 的 Fourier 变换.
  由于 $s_{\Delta T}(t)$ 是一个周期为 ΔT 的周期函数，故它可表示为一个 Fourier 级数:

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} c_n \exp\{i\frac{2\pi n}{\Delta T}t\}$$

$$\begin{aligned}c_n &= \frac{1}{\Delta T}\int_{-\frac{\Delta T}{2}}^{\frac{\Delta T}{2}} s_{\Delta T}(t)\exp\{-i\frac{2\pi n}{\Delta T}t\}dt \\ &= \frac{1}{\Delta T}\int_{-\frac{\Delta T}{2}}^{\frac{\Delta T}{2}} \sum_{k=-\infty}^{\infty}\delta(t-k\Delta T)\exp\{-i\frac{2\pi n}{\Delta T}t\}dt \quad (\text{note that } k\Delta T \in (-\frac{\Delta T}{2},\frac{\Delta T}{2}) \text{ iff } k=0) \\ &= \frac{1}{\Delta T}\int_{-\frac{\Delta T}{2}}^{\frac{\Delta T}{2}} \delta(t-0)\exp\{-i\frac{2\pi n}{\Delta T}t\}dt \\ &= \frac{1}{\Delta T}\exp\{-i\frac{2\pi n}{\Delta T}\cdot 0\} \\ &= \frac{1}{\Delta T}\end{aligned}$$

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} c_n \exp\{i\frac{2\pi n}{\Delta T}t\} = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\exp\{i\frac{2\pi n}{\Delta T}t\}$$

  因此 $s_{\Delta T}(t) := \sum_{k=-\infty}^{\infty}\delta(t-k\Delta T)$ 的 Fourier 级数表示为 $s_{\Delta T}(t) = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\exp\{i\frac{2\pi n}{\Delta T}t\}$
  于是我们有:

$$\begin{aligned}\tilde{s}_{\Delta T}(\mu) &:= \int_{-\infty}^{\infty} s_{\Delta T}(t)\exp\left(-i(2\pi\mu)t\right)dt \\ &= \int_{-\infty}^{\infty}\frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\exp\{i\frac{2\pi n}{\Delta T}t\}\exp\left(-i(2\pi\mu)t\right)dt \\ &= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\int_{-\infty}^{\infty}\exp\{i(\frac{2\pi n}{\Delta T}-2\pi\mu)t\}dt \quad (\text{note that } \delta(\mu) := \int_{-\infty}^{\infty}\exp\left(i(2\pi\mu)t\right)dt) \\ &= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\delta(2\pi\mu-\frac{2\pi n}{\Delta T}) \\ &= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\delta(\mu-\frac{n}{\Delta T})\end{aligned}$$

Part (2)

试证明实信号 $f(x)$ 的离散频域变换结果是共轭对称的.

Proof:
定义时间间隔 ΔT 的冲激串 $s_{\Delta T}(t) := \sum_{k=-\infty}^{\infty} \delta(t - k\Delta T)$ (其中 $\delta(\cdot)$ 为 Dirac 函数)
取样函数 $f_s(t) := f(t) s_{\Delta T}(t) = \sum_{k=-\infty}^{\infty} f(t)\delta(t - k\Delta T) = \sum_{k=-\infty}^{\infty} f(k\Delta T)\mathbb{1}(t = k\Delta T)$
取样点为 $k\Delta T$ $(k = 0, \pm 1, \ldots)$

注意到 $\tilde{f}_s(\mu)$ 可以直接写成 $f_s(t)$ 的 Fourier 变换:

$$\begin{aligned}
\tilde{f}_s(\mu) &= \int_{-\infty}^{\infty} f_s(t) \exp\{-i(2\pi\mu)t\} dt \quad (\text{note that } f_s(t) := \sum_{k=-\infty}^{\infty} f(t)\delta(t - k\Delta T)) \\
&= \int_{-\infty}^{\infty} \sum_{k=-\infty}^{\infty} f(t)\delta(t - k\Delta T) \exp\{-i(2\pi\mu)t\} dt \\
&= \sum_{k=-\infty}^{\infty} \int_{-\infty}^{\infty} f(t) \exp\{-i(2\pi\mu)t\}\delta(t - k\Delta T) dt \quad (\text{Sampling property of the Dirac delta function}) \\
&= \sum_{k=-\infty}^{\infty} f(k\Delta T) \exp\{-i(2\pi\mu)(k\Delta T)\}
\end{aligned}$$

假设我们在从周期 $\mu \in [0, \frac{1}{\Delta T})$ 内等间隔地取 $\tilde{f}_s(\mu)$ 的 N 个样本:

$$\mu = \frac{n}{N\Delta T} \ (n = 0, 1, \ldots, N-1)$$

在已知 $f(k\Delta T)$ $(k = 0, 1, \ldots, N-1)$ 的情况下，我们便可得到离散 Fourier 变换:

$$\left.\begin{aligned}
\tilde{f}_{\text{DFT}}(\frac{n}{N\Delta T}) &:= \sum_{k=0}^{N-1} f(k\Delta T) \exp\{-i(2\pi\frac{n}{N\Delta T})(k\Delta T)\} \\
&= \sum_{k=0}^{N-1} f(k\Delta T) \exp\{-i(2\pi\frac{n}{N})k\}
\end{aligned}\right. \quad (n = 0, 1, \ldots, N-1)$$

使用 x 表示图像坐标变量，并使用 μ 表示频率变量更为直观，它们都可视为整数.
则我们有:

$$\tilde{f}_{\text{DFT}}(\mu) = \sum_{x=0}^{N-1} f(x) \exp\{-i(2\pi\frac{\mu}{N})x\} \ \ (\mu = 0, 1, \ldots, N-1)$$

其中 $f(x)$ 为实信号.

于是我们有:

$$\begin{aligned}
\overline{\tilde{f}_{\text{DFT}}(\mu)} &= \overline{\sum_{x=0}^{N-1} f(x) \exp\{-i(2\pi\frac{\mu}{N})x\}} \quad (\text{note that } f(x) \in \mathbb{R} \text{ for all } x = 0, 1, \ldots, N-1) \\
&= \sum_{x=0}^{N-1} f(x)\overline{\exp\{-i(2\pi\frac{\mu}{N})x\}} \\
&= \sum_{x=0}^{N-1} f(x) \exp\{i(2\pi\frac{\mu}{N})x\} \\
&= \sum_{x=0}^{N-1} f(x) \exp\{-i(2\pi\frac{(-\mu)}{N})x\} \\
&= \tilde{f}_{\text{DFT}}(-\mu)
\end{aligned}$$

这表明实信号 $f(x)$ 的离散频域变换结果 $\tilde{f}_{\text{DFT}}(\mu)$ 是共轭对称的.

Part (3)

试证明二维变量的离散 Fourier 变换的卷积定理.

Proof:
考虑关于连续空间变量 t, s 的连续函数 $f(t,s), g(t,s)$，其二维连续 Fourier 变换为:

$$\begin{aligned}\tilde{f}(\mu,\nu) &:= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t,s)\exp\{-i2\pi(\mu t+\nu s)\}dtds \\ \tilde{g}(\mu,\nu) &:= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(t,s)\exp\{-i2\pi(\mu t+\nu s)\}dtds\end{aligned}$$

其中 μ, ν 为频率变量.

一方面，我们有:

$$\begin{aligned}&\widetilde{(f\star g)}(\mu,\nu) \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(f\star g)(t,s)\exp\{-i2\pi(\mu t+\nu s)\}dtds \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\tau,\sigma)g(t-\tau,s-\sigma)d\tau d\sigma\right)\exp\{-i2\pi(\mu t+\nu s)\}dtds \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\tau,\sigma)\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(t-\tau,s-\sigma)\exp\{-i2\pi(\mu t+\nu s)\}dtds\right)d\tau d\sigma \\ &= \left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\tau,\sigma)\exp\{-i2\pi(\mu\tau+\nu\sigma)\}d\tau d\sigma\right)\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(t-\tau,s-\sigma)\exp\{-i2\pi(\mu(t-\tau)+\nu(s-\sigma))\}dtds\right) \\ &= \tilde{f}(\mu,\nu)\cdot\tilde{g}(\mu,\nu) \\ &= (\tilde{f}\cdot\tilde{g})(\mu,\nu)\end{aligned}$$

另一方面，我们有:

$$\begin{aligned}\widetilde{(f\cdot g)}(\mu,\nu) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(f\cdot g)(t,s)\exp\{-i2\pi(\mu t+\nu s)\}dtds \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t,s)g(t,s)\exp\{-i2\pi(\mu t+\nu s)\}dtds \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\tilde{f}(\tau,\sigma)\exp\{i2\pi(\tau t+\sigma s)\}d\tau d\sigma\right)g(t,s)\exp\{-i2\pi(\mu t+\nu s)\}dtds \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\tilde{f}(\tau,\sigma)\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(t,s)\exp\{i2\pi(\tau t+\sigma s)\}\exp\{-i2\pi(\mu t+\nu s)\}dtds\right)d\tau d\sigma \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\tilde{f}(\tau,\sigma)\left(\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(t,s)\exp\{-i2\pi((\mu-\tau)t+(\nu-\sigma)s)\}\}dtds\right)d\tau d\sigma \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\tilde{f}(\tau,\sigma)\tilde{g}(\mu-\tau,\nu-\sigma)d\tau d\sigma \\ &= (\tilde{f}\star\tilde{g})(\mu,\nu)\end{aligned}$$

(可恶，看错题目了)
设 $f(x,y)$ 为 $M\times N$ 的数字图像.
其二维离散 Fourier 变换和二维离散 Fourier 逆变换为:

$$\begin{aligned}\tilde{f}_{\text{DFT}}(\mu,\nu) &:= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp\{-i2\pi(\frac{\mu x}{M}+\frac{\nu y}{N})\}\ \ (\begin{cases}\mu=0,1,\dots,M-1\\ \nu=0,1,\dots,N-1\end{cases}) \\ f(x,y) &= \frac{1}{MN}\sum_{\mu=0}^{M-1}\sum_{\nu=0}^{N-1}\tilde{f}_{\text{DFT}}(\mu,\nu)\exp\{i2\pi(\frac{x\mu}{M}+\frac{y\nu}{N})\}\ \ (\begin{cases}x=0,1,\dots,M-1\\ y=0,1,\dots,N-1\end{cases})\end{aligned}$$

其中 μ, ν 为频率变量.

现考虑两幅 $M\times N$ 的数字图像 $f(x,y)$ 和 $g(x,y)$
假设其在 x 轴方向以周期 M 进行拓展，在 y 轴方向以周期 N 进行拓展
(因为离散卷积是循环卷积，所以这样假设是等价的，而且用起来会比较方便)
一方面，我们有:

$$
\begin{aligned}
&(\widetilde{f_{\mathrm{DFT}} \star g_{\mathrm{DFT}}})(\mu, \nu) \\
&= \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (f \star g)(x, y) \exp\{-i2\pi(\frac{\mu x}{M} + \frac{\nu y}{N})\} \\
&= \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left(\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) g(x-m, y-n) \right) \exp\{-i2\pi(\frac{\mu x}{M} + \frac{\nu y}{N})\} \\
&= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left\{ f(m, n) \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} g(x-m, y-n) \exp\{-i2\pi(\frac{\mu x}{M} + \frac{\nu y}{N})\} \right\} \\
&= \left(\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) \exp\{i2\pi(\frac{\mu m}{M} + \frac{\nu n}{N})\} \right) \left(\sum_{x-m=0}^{M-1} \sum_{y-n=0}^{N-1} g(x-m, y-n) \exp\{-i2\pi(\frac{\mu(x-m)}{M} + \frac{\nu(y-n)}{N})\} \right) \\
&= \tilde{f}_{\mathrm{DFT}}(\mu, \nu) \cdot \tilde{g}_{\mathrm{DFT}}(\mu, \nu) \\
&= (\tilde{f}_{\mathrm{DFT}} \cdot \tilde{g}_{\mathrm{DFT}})(\mu, \nu)
\end{aligned}
$$

另一方面，我们有:

$$
\begin{aligned}
(\widetilde{f \cdot g})(\mu, \nu) &= \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (f \cdot g)(x, y) \exp\{-i2\pi(\frac{\mu x}{M} + \frac{\nu y}{N})\} \\
&= \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) g(x, y) \exp\{-i2\pi(\frac{\mu x}{M} + \frac{\nu y}{N})\} \\
&= \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left(\frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \tilde{f}(m, n) \exp\{i2\pi(\frac{mx}{M} + \frac{ny}{N})\} \right) g(x, y) \exp\{-i2\pi(\frac{\mu x}{M} + \frac{\nu y}{N})\} \\
&= \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \tilde{f}(m, n) \left(\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} g(x, y) \exp\{i2\pi(\frac{mx}{M} + \frac{ny}{N})\} \exp\{-i2\pi(\frac{\mu x}{M} + \frac{\nu y}{N})\} \right) \\
&= \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \tilde{f}(m, n) \left(\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} g(x, y) \exp\{-i2\pi(\frac{(\mu-m)x}{M} + \frac{(\nu-n)y}{N})\} \right) \\
&= \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \tilde{f}(m, n) \tilde{g}(\mu-m, \nu-n) \\
&= \frac{1}{MN} (\tilde{f}_{\mathrm{DFT}} \star \tilde{g}_{\mathrm{DFT}})(\mu, \nu)
\end{aligned}
$$

The End