EE416: Introduction to Image Processing and Computer Vision

Il Yong Chun
Department of Electrical and Computer Engineering, the University of Hawaiʻi, Mānoa

October 1, 2021

2 Discrete "space" signals and systems

2.1 2D convolution

By the sifting property of the Kronecker impulse function (see §2.2.3), we can represent any 2D discrete-space (DS) signal as a sum:

$$f[m,n] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\delta[m-k,n-l].$$

For a LSI system, we can use the superposition property to see that

$$f[m,n] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\delta[m-k,n-l] \to \boxed{\text{LSI } \mathcal{S}} \to \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]h[m-k,n-l].$$

Thus we have derived the following general input-output relationship for 2D DS LSI system:

$$f[m,n] \to \boxed{\text{LSI } h[m,n]} \to g[m,n] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]h[m-k,n-l] = f[m,n] * h[m,n]$$

which is called the **2D convolution (sum)**.

Note: An LSI system is completely specified by its impulse response.

2D convolution recipe.

- Mirror $h[k,l]$ about origin to form $h[-k,-l]$
- Shift the mirrored signal by m,n to form $h[m-k,n-l]$
- Multiply $h[m-k,n-l]$ by $f[k,l]$
- Sum over all k,l.
- Repeat for every m,n.

Simply put, move a mirrored kernel and apply weighted sum with weights in a mirrored kernel!
Ex. Find $g = f * h$.

$$h[k,l] = \begin{bmatrix} 2 & 4 & 6 \\ 1 & 2 & 3 \end{bmatrix} \quad h[-k,-l] = \boxed{?} \qquad f[k,l] = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad g[m,n] = \boxed{?}$$

Ex. Find the response of the 3 by 3 moving average system ($\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$) to the 2D input signal

$$f[m,n] = \cos(\pi m)\cos(\pi n) = (-1)^{m+n}.$$

What does this signal look like? checkerboard with ± 1: $\begin{bmatrix} 1 & -1 & 1 & \\ -1 & 1 & 1 & \cdots \\ 1 & -1 & 1 & \end{bmatrix}$.

The output is $g[m,n] = \boxed{?}$.

2.1.1 Computation and separability

The computational cost of 2D convolution of a $N_1 \times M_1$ image with a $N_2 \times M_2$ image is approximately $(N_1 + N_2 - 1) \cdot (M_1 + M_2 - 1) \cdot \min\{N_1 M_1, N_2 M_2\}$ adds/multipliers, or about $N^2 M^2$ if $N_1 = M_1 = N \gg N_2 = M_2 = M$.

Using a separable filter can greatly reduce computation. If $h[m,n] = h_1[m]h_2[n]$, then we can rewrite the convolutional sum as follows:

$$\sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] h_1[m-k] h_2[n-l] = \sum_{k=-\infty}^{\infty} h_1[m-k] \left(\sum_{l=-\infty}^{\infty} f[k,l] h_2[n-l] \right).$$

- The inner sum is simply 1D convolution of each column of the image with the 1D kernel h_2.
- The outer sum is then the 1D convolution of each row of that intermediate result with the 1D kernel h_1.
- For convolving a large $N \times N$ image with a small separable $M \times M$ kernel, the computational requirement is about $N^2(M + M) = 2N^2 M$,[1] so we have about a factor of $M/2$ computation.
- Because M is at least 3 usually, that means at least a $\boxed{?}$% reduction in computation, and often much more.

To read: [T2, §3.4.Separable Filter Kernels]

2.1.2 2D convolution properties

- Commutative property: $f * h = h * f$
- Associative property: $(f * h_1) * h_2 = f * (h_1 * h_2)$
- Distributive property: $f * (h_1 + h_2) = (f * h_1) + (f * h_2)$
- Shift-invariance: If $g[m,n] = f[m,n] * h[m,n]$, then $f[m - k_1, n - l_1] * h[m - k_2, n - l_2] = g[m - k_1 - k_2, n - l_1 - l_2]$
- Separability: $(f_1[m] f_2[n]) * (h_1[m] h_2[n]) = (f_1[m] * h_1[m]) \cdot (f_2[n] * h_2[n])$
- Identity: $f[m,n] * \delta[m,n] = f[m,n]$
- Shifting property: $f[m,n] * \delta[m - m_0, n - n_0] = f[m - m_0, n - n_0]$

2.2 Discrete-space Fourier transform (DSFT)

2.2.1 Discrete-time Fourier transform (DTFT) and inverse DTFT

$$F(e^{\mathrm{i}\omega}) = \sum_{n=-\infty}^{\infty} f[n] e^{-\mathrm{i}\omega n} \qquad \text{(forward)}$$

$$f[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(e^{\mathrm{i}\omega}) e^{\mathrm{i}\omega n} \, \mathrm{d}\omega \qquad \text{(inverse)}$$

- The DTFT is periodic with period 2π: $F(e^{\mathrm{i}(\omega + 2\pi)}) = f(e^{\mathrm{i}\omega})$.
- Thus, functions such as $\mathrm{rect}(\omega)$ are not valid DTFTs.

[1] To be more precise, 1D convolution (if implemented carefully) of a N-point signal with a M-point signal, with $N \geq M$, requires MN multiplications, and nearly the same number of additions.

2.2.2 DSFT and inverse DSFT

$$F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} f[m,n] e^{-\mathrm{i}(\mu m + \nu n)} \qquad \text{(forward)}$$

$$f[m,n] = \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) e^{\mathrm{i}(\mu m + \nu n)} \, d\mu \, d\nu \qquad \text{(inverse)}$$

- The DSFT is a 2D *periodic* function with period 2π in both the μ and ν dimensions:

$$F(e^{\mathrm{i}(\mu+2\pi)}, e^{\mathrm{i}(\nu+2\pi)}) = X(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}).$$

2.2.3 Useful discrete-space functions

Discrete-space delta function: 2D Kronecker impulse.

- The **2D Kronecker delta** or **impulse** function is defined by

$$\delta[m,n] \triangleq \begin{cases} 1, & m,n = 0, \\ 0, & \text{otherwise.} \end{cases}$$

  Unlike the Dirac impluse, which is not a function, the Kronecker impulse is a (discrete-space) function.
- Properties:
  - Unity sum: $\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \delta[m,n] = 1$
    (analogous to the unity integral property of the Dirac impulse).
  - Sampling property: $\delta[m-m_0, n-n_0] f[m,n] = \delta[m-m_0, n-n_0] f[m_0, n_0]$
  - Sifting property: $\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \delta[m-m_0, n-n_0] f[m,n] = f[m_0, n_0]$
  - Symmetric property: $\delta[-m,-n] = \delta[m,n]$
  - Separability: $\delta[m,n] = \delta[m]\delta[n]$, where $\delta[n] \triangleq \begin{cases} 1, & n = 0, \\ 0, & n \neq 0. \end{cases}$

2D unit-step function.

- Definition: $\mathrm{step}[m,n] \triangleq \begin{cases} 1, & m,n \geq 0, \\ 0, & \text{otherwise.} \end{cases}$
- Separability holds: $\mathrm{step}[m,n] = \mathrm{step}[m]\mathrm{step}[n]$, where $\mathrm{step}[n] \triangleq \begin{cases} 1, & n \geq 0, \\ 0, & \text{otherwise.} \end{cases}$
- In 1D, $\delta[n] = \mathrm{step}[n] - \mathrm{step}[n-1]$. Then, $\delta[m,n] = \boxed{?}$
- $\delta[m,n] = (\mathrm{step}[m,n] - \mathrm{step}[m-1,n]) \, (\mathrm{step}[m,n] - \mathrm{step}[m,n-1])$.

2.2.4 DSFT properties

- Multiplication (windowing): Because the 2D DSFT is periodic, the space-domain multiplication property is a little more complicated than simply "convolution" in the frequency domain:

$$f[m,n]g[m,n] \overset{\text{DSFT}}{\Longleftrightarrow} \frac{1}{(2\pi)^2} F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) \boxed{*}_{2\pi} G(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})$$

$$\triangleq \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F(e^{\mathrm{i}\mu'}, e^{\mathrm{i}\nu'}) G(e^{\mathrm{i}(\mu-\mu')}, e^{\mathrm{i}(\nu-\nu')}) \, d\mu' \, d\nu'$$

  which is (2π) periodic convolution. (Notice that the integrand is periodic with period 2π.)
  Ex. Derive this result (Hint: substitute the summations for $F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})$ and $G(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})$ into the

Property	Space domain function	DSFT
Linearity	$af[m,n] + bg[m,n]$	$aF(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) + bG(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})$
Conjugation	$f^*[m,n]$	$F^*(e^{-\mathrm{i}\mu}, e^{-\mathrm{i}\nu})$
Shifting	$f[m-m_0, n-n_0]$	$e^{-\mathrm{i}(\mu m_0 + \nu n_0)} F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})$
Modulation	$e^{\mathrm{i}(\mu_0 m + \nu_0 n)} f[m,n]$	$F\left(e^{\mathrm{i}(\mu-\mu_0)}, e^{\mathrm{i}(\nu-\nu_0)}\right)$
Convolution	$f[m,n] * g[m,n]$	$F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) G(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})$
Cross correlation	$f[m,n] \star g[m,n]$	$F^*(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) G(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})$
Multiplication (windowing)	$f[m,n] g[m,n]$	$\frac{1}{(2\pi)^2} F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) \boxed{*}_{2\pi} G(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})$
Separability	$f[m] g[n]$	$F(e^{\mathrm{i}\mu}) G(e^{\mathrm{i}\nu})$
Parseval's Theorem	$\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n] g^*[m,n]$ $= \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) G^*(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) \, \mathrm{d}\mu \, \mathrm{d}\nu$	

convolution sum and reduce the expressionto an integral that can be recognized as the DSFT of $f[m,n]g[m,n]$.)

$\boxed{?}$

- Duality? Is there a duality property for the DSFT? No, because the arguments $[m,n]$ and (μ,ν) are fundamentally different.
- Scaling property? In continuous space we have the simple scaling property, $f(ax, by) \overset{\text{CSFT}}{\Longleftrightarrow} \frac{1}{|ab|} F(u/a, v/b)$. *The corresponding properties in discrete space are more complicated!*
  Ex. Find a FT property for downsampling: $g[m,n] = f[2m, 2n]$. (Hint: Due to possible aliasing, the answer is *not* simply $G(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \frac{1}{4} F(e^{\mathrm{i}\mu/2}, e^{\mathrm{i}\nu/2})$.)

  $\boxed{?}$

  Ex. Find a FT property for upsampling: $g[m,n] = \begin{cases} f[m/2, n/2], & n \text{ even and } m \text{ even} \\ 0, & \text{otherwise.} \end{cases}$

  $\boxed{?}$

  Up-sampling and down-sampling are very common operations in image processing and computer vision, so their effects on the spectrum are important.
- Notice that there exists no scaling or rotation properties for the DSFT.

Ex. The frequency response of the (separable!) 3×3 moving average filter is given by

$$H(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} h[m,n] e^{-\mathrm{i}(\mu m + \nu n)} = \frac{1}{9} \sum_{m=-1}^{1} \sum_{n=-1}^{1} e^{-\mathrm{i}\mu m} e^{-\mathrm{i}\nu n} = \frac{1}{9}(1 + 2\cos\mu)(1 + 2\cos\nu)$$

Is this a good low-pass filter? $\boxed{?}$

2.2.5 Elementary DSFT relations

Space domain function	DSFT
$\delta[m,n]$	1
1	$(2\pi)^2 \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \delta(\mu - 2\pi k, \nu - 2\pi l)$
$e^{\mathrm{i}(am+bn)}, \quad a, b \in (-\pi, \pi]$	$(2\pi)^2 \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \delta(\mu - a - 2\pi k, \nu - b - 2\pi l)$
$\begin{cases} 1, & \lvert m\rvert \le M, \lvert n\rvert \le N, \\ 0, & \text{otherwise,} \end{cases} \quad M, N \in \{0, 1, 2, \ldots\}$	$\dfrac{\sin(\mu(M+1/2))}{\sin(\mu/2)} \dfrac{\sin(\nu(N+1/2))}{\sin(\nu/2)}$

- In CS, we have the FT pair $e^{\mathrm{i}2\pi(ax+by)} \overset{\text{CSFT}}{\Longleftrightarrow} \delta(u-a, v-b)$. What about in DS? Can the specturm of a DS signal consist of a single impulse? $\boxed{?}$

  Ex. Show the DSFT relation $e^{\mathrm{i}(am+bn)} \overset{\text{DSFT}}{\Longleftrightarrow} (2\pi)^2 \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \delta(\mu - a - 2\pi k, \nu - b - 2\pi l)$. (Hint: Use the inverse DSFT definition assuming $a, b \in (-\pi, \pi]$.)

  $\boxed{?}$

- The function $\frac{\sin(\mu(M+1/2))}{\sin(\mu/2)}$ is sometimes called Dirichlet kernel or the periodic sinc function. (At $\mu = 0$, its value is $2N+1$.)
- This DS table is noticeably shorter than our table of 2D CSFT pairs. In DS we usually work numerically (using fast Fourier transform) or focus on simple functions (typically the impulse response of a 2D filter) as follows:

$$h[m,n] = \sum_l \alpha_l \delta[m - m_l, n - n_l] \overset{\text{DSFT}}{\Longleftrightarrow} H(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \sum_l \alpha_l e^{-\mathrm{i}(\mu m_l + \nu n_l)}.$$

2.2.6 DSFT revisited with DSFT properties

Engineer's proof of the inverse 2D DSFT relationship:

$$\begin{aligned}
\frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) e^{\mathrm{i}(\mu m + \nu n)} \,\mathrm{d}\mu \,\mathrm{d}\nu &= \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} \left\{ \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] e^{-\mathrm{i}(\mu k + \nu l)} \right\} e^{\mathrm{i}(\mu m + \nu n)} \,\mathrm{d}\mu \,\mathrm{d}\nu \\
&= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \left\{ \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} e^{\mathrm{i}\{\mu(m-k) + \nu(n-l)\}} \,\mathrm{d}\mu \,\mathrm{d}\nu \right\} \\
&= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l] \delta[m-k, n-l] = f[m,n].
\end{aligned}$$

where the last equality holds by the modulation property.

2.3 2D sampling and aliasing

This section requires both continuous- and discrete-space, so we include the subscripts for clarity, e.g., $g_a(x,y)$ and $g_d[m,n]$.

Examples of 2D sampling:

- In some digital mammography systems, the intensity of a transmitted X-ray beam is recorded in "analog" format on some continuous detector (e.g., photographic film, or a rewritable equivalent). The intensity that is recorded is our $g_a(x, y)$. Then a digitization device (such as a laser-scanning system or CCD camera) samples $g_a(x, y)$ over a regular grid, and sends (quantized) values $g_d[m, n]$ to the digital computer.
- An optical scanner is another device that converts a continuous, analog image (picture) into a sampled, digital form.

2.3.1 2D sampling with the DSFT

- Consider ideal rectilinear point sampling model:

$$g_d[m, n] = g_a(m\Delta_x, n\Delta_y),$$

where Δ_x and Δ_y are the **sampling interval** in the x and y directions, respectively. (The ratios $1/\Delta_x$ and $1/\Delta_y$ can be called the **sampling rates** in the x and y directions, respectively.)
- Using the inverse CSFT, we have

$$\begin{aligned}
g_d[m, n] &= g_a(m\Delta_x, n\Delta_y) \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G_a(u, v) e^{\mathrm{i}2\pi(m\Delta_x u + n\Delta_y v)} \, \mathrm{d}u \, \mathrm{d}v \\
&= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \int_{(k-1/2)/\Delta_x}^{(k+1/2)/\Delta_x} \int_{(l-1/2)/\Delta_y}^{(l+1/2)/\Delta_y} G_a(u, v) e^{\mathrm{i}2\pi(m\Delta_x u + n\Delta_y v)} \, \mathrm{d}u \, \mathrm{d}v \\
&\text{let } \mu = 2\pi(u\Delta_x + k) \text{ and } \nu = 2\pi(v\Delta_y + l) \\
&= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} G_a\left(\frac{\mu/(2\pi) - k}{\Delta_x}, \frac{\nu/(2\pi) - l}{\Delta_y}\right) e^{\mathrm{i}\{(\mu - 2\pi k)m + (\nu - 2\pi l)n\}} \frac{\mathrm{d}\mu \, \mathrm{d}\nu}{(2\pi\Delta_x)(2\pi\Delta_y)} \\
&= \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} \left\{ \frac{1}{\Delta_x \Delta_y} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} G_a\left(\frac{\mu - 2\pi k}{2\pi\Delta_x}, \frac{\nu - 2\pi l}{2\pi\Delta_y}\right) \right\} e^{\mathrm{i}(\mu m + \nu n)} \, \mathrm{d}\mu \, \mathrm{d}\nu.
\end{aligned}$$

This final expression is in the form of the inverse DSFT. Thus, the braced expression must be the DSFT of $g_d[n, m]$.
Ex. Explain why the last equality holds.
$\boxed{?}$

- In other words, we have the following relationship between the spectrum of sampled signal $g_d[m, n]$ the spectrum of original CS signal $g_a(x, y)$:

$$\boxed{G_d(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \frac{1}{\Delta_x \Delta_y} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} G_a\left(\frac{\mu - 2\pi k}{2\pi\Delta_x}, \frac{\nu - 2\pi l}{2\pi\Delta_y}\right).}$$

- Intuition

- Scale frequencies:

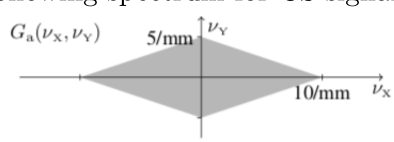
$$(u,v) = (0,0) \Leftrightarrow (\mu,\nu) = (0,0)$$
$$(u,v) = \left(\frac{1}{2\Delta_x}, 0\right) \Leftrightarrow (\mu,\nu) = (\pi,0)$$
$$(u,v) = \left(0, \frac{1}{2\Delta_y}\right) \Leftrightarrow (\mu,\nu) = (0,\pi)$$
$$(u,v) = \left(\frac{1}{2\Delta_x}, \frac{1}{2\Delta_y}\right) \Leftrightarrow (\mu,\nu) = (\pi,\pi)$$

- Replicate along both μ and ν with period 2π.
- Apply gain factor of $\frac{1}{\Delta_x \Delta_y}$.

2.3.2 Aliasing

- A CS signal, $g_a(x,y)$, may be uniquely reconstructed from its sampled version, $g_d[m,n]$, if $G_a(u,v) = 0$ for all $|u| > \frac{1}{2\Delta_x}$ and $|v| > \frac{1}{2\Delta_y}$.
- Spectral overlap is called **aliasing** (higher spatial frequencies will "alias" as lower spatial frequencies).
- This condition is sufficient, but not necessary.
  Example $\boxed{?}$

- **Ex.** Suppose that we have the following spectrum for CS signal $g_a(x,y)$:



  1) To avoid aliasing, what is maximum sample spacing? $\boxed{?}$
  2) Suppose we sample with $\Delta_x = \Delta_y = 1/30$ mm. Find spectrum of $g_d[m,n]$:
  $\boxed{?}$

To read: [T1, §3-5], [T2, §4.5.2-D Sampling and the 2-D Sampling Theorem and §4.5.Aliasing]

2.4 From charged coupled devices to image display: A practical example of 2D sampling and reconstruction

Focal plain arrays: Charged coupled devices (CCD) imaging array

- Solid state device used in video and still cameras.
- Each cell collects photons in a square $\Delta \times \Delta$ region.
- Response of each cell is linear with energy (photons).
- Cell should be large for best sensitivity.
- Finite cell size violates sampling assumptions.

2.4.1 CCD models

- Mathematical model for CCD
  - Let $g_d[m,n]$ be the output of cell (m,n), then

$$g_d[m,n] = \int_{\mathbb{R}^2} h(x - m\Delta, y - n\Delta) g_a(x,y)\, \mathrm{d}x\, \mathrm{d}y$$

    where $h(x,y)$ is the rectangular window for each cell, $h(x,y) = \frac{1}{\Delta^2}\mathrm{rect}(x/\Delta, y/\Delta)$.[2]
  - Define $\tilde{g}_a(x,y)$ so that

$$\begin{aligned}\tilde{g}_a(\xi,\eta) &= \int_{\mathbb{R}^2} h(x-\xi, y-\eta) g_a(x,y)\, \mathrm{d}x\, \mathrm{d}y \\ &= h(-x,-y) * g_a(x,y)\end{aligned}$$

    and then we have

$$g_d[m,n] = \tilde{g}_a(m\Delta, n\Delta).$$

- CCD model in space domain:
  - Filter signal with space reversed cell profile: $\tilde{g}_a(x,y) = h(-x,-y) * g_a(x,y) = \frac{1}{\Delta^2}\mathrm{rect}(\frac{x}{\Delta}, \frac{y}{\Delta}) * g_a(x,y)$
  - Sample filtered image $g_d[m,n] = \tilde{g}_a(m\Delta, n\Delta)$
  - Cell aperture blurs image.
- CCD model in frequency domain:
  - Filter signal with cell profile: $\tilde{G}_a(u,v) = H^*(u,v)G_a(u,v) = \mathrm{sinc}(u\Delta, v\Delta)G_a(u,v)$
  - Sample filtered image: $G_d(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \dfrac{1}{\Delta^2}\displaystyle\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} \tilde{G}_a\left(\frac{\mu - 2\pi k}{2\pi\Delta}, \frac{\nu - 2\pi l}{2\pi\Delta}\right)$
  - Complete model: $G_d(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \dfrac{1}{\Delta^2}\displaystyle\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} \mathrm{sinc}\left(\frac{\mu - 2\pi k}{2\pi\Delta}, \frac{\nu - 2\pi l}{2\pi\Delta}\right) G_a\left(\frac{\mu - 2\pi k}{2\pi\Delta}, \frac{\nu - 2\pi l}{2\pi\Delta}\right)$
  - Sinc function filters image.

2.4.2 Reconstruction: Sampled image display or rendering

- CRT and LCD displays convert discrete-space images to continuous-space images.
- First, let's get some intuitions. How can we "recover" a (band-limited) image $g_a(x,y)$ from its samples $\{g_d[m,n]\}$?
  - If G_a is band-limited to $(1/(2\Delta), 1/(2\Delta))$ (and hence adequately sampled), then

$$G_a(u,v) \approx G_d(e^{\mathrm{i}2\pi\Delta u}, e^{\mathrm{i}2\pi\Delta v}) \cdot \mathrm{rect}(u\Delta, v\Delta),$$

    using $\mu = 2\pi\Delta u$, $\nu = 2\pi\Delta v$. The ideal lowpass filter $\mathrm{rect}(u\Delta, v\Delta)$ selects out the central replicate. This replicate is a weighted spectrum of $G_a(u,v)$, $\mathrm{sinc}(u,v)G_a(u,v)$. This is easily seen in the spectrum, but is perhaps puzzling in the space domain!

[2] For analysis of sampling, it is convenient to use a grid of Dirac impulses ("bed of nails") to synthesize the following continuous-space "function" $g_s(x,y)$ from the samples $\{g_d[m,n]\}$. This manipulation is purely analytical – no such impulses exist in practice. The hypothetical "sample carrying" continuous-space image is defined by

$$g_s(x,y) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} g_d[m,n]\Delta^2\delta(x - m\Delta, y - n\Delta).$$

- Taking the inverse CSFT of the key property above yields:

$$\begin{aligned} g_a(x,y) &\approx g_d(x,y) * \left(\frac{1}{\Delta^2}\operatorname{sinc}\left(\frac{x}{\Delta},\frac{y}{\Delta}\right)\right) \\ &= \left(\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} g_d[m,n]\Delta^2\delta(x-m\Delta,y-n\Delta)\right) * \left(\frac{1}{\Delta^2}\operatorname{sinc}\left(\frac{x}{\Delta},\frac{y}{\Delta}\right)\right) \\ &= \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} g_d[m,n]\operatorname{sinc}\left(\frac{x-m\Delta}{\Delta},\frac{y-m\Delta}{\Delta}\right), \end{aligned}$$

  using the shift property for convolution with a Dirac impulse. Thus we can approximately recover $g_a(x,y)$ by interpolating the samples $g_d[m,n]$ using sinc functions. This is called **sinc interpolation**. The sinc function has infinite extent, so in practice approximate interpolators are used more frequently (such as B-splines). (See Matlab's `interp2` routine.)

- Model for reconstruction
  - $g_d[m,n]$: sampled image, $r(x,y)$: displayed image,
    $p(x,y)$: PSF of display that *approximates* the ideal lowpass filter (i.e., the transfer function $\operatorname{rect}(u\Delta, v\Delta)$) with finite extent in (x,y)
  - In space domain, $r(x,y) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} g_d[m,n]p(x-m\Delta,y-n\Delta)$.
    In other words, we display $r(x,y)$ by interpolating the samples $g_d[m,n]$ using practical PSF function $p(x,y)$.
  - In frequency domain, $R(u,v) = P(u,v)G_d(e^{i2\pi\Delta u}, e^{i2\pi\Delta v})$ (notice $\mu = 2\pi\Delta u$, $\nu = 2\pi\Delta v$).
  - Monitor PSF further "softens" image.

- Model for sampling and reconstruction
  - $R(u,v) = \dfrac{P(u,v)}{\Delta^2}\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} H^*\left(u-\frac{k}{\Delta}, v-\frac{l}{\Delta}\right) G_a\left(u-\frac{k}{\Delta}, v-\frac{l}{\Delta}\right)$
  - When no aliasing occurs, this reduces to

$$\begin{aligned} R(u,v) &= \frac{P(u,v)H^*(u,v)}{\Delta^2}G_a(u,v) \\ &= \frac{P(u,v)\operatorname{sinc}(u\Delta,v\Delta)}{\Delta^2}G_a(u,v), \end{aligned}$$

    assuming that the transfer function $P(u,v)$ selects out the central replicate.

- Effects of sampling and reconstruction
  - The image is filtered by the transfer function $\frac{1}{\Delta^2}P(u,v)H^*(u,v) = \frac{1}{\Delta^2}P(u,v)\operatorname{sinc}(u\Delta,v\Delta)$.
  - Scanned image often must be "sharpened" to remove the effects of softening produced in the sampling and display processes.

2.5 Filtering 2D signals, i.e., images

Filters are used extensively in image processing and computer vision for tasks such as reducing noise, reducing blur caused by the imperfect PSF or an image formation system, enhancing certain features, etc.

2.5.1 Introduction

Implementation: One can implement filters using

- direct convolution (practical for FIR filters where double sum is finite):

$$g[m,n] = \sum_{k,l} h[k,l] f[m-k, n-l]$$

- difference equations (very popular for 1D IIR filters; far less common in 2D)

$$g[m,n] = \sum_{k,l} a[k,l] g[m-k, n-l] + \sum_{k,l} b[k,l] f[m-k, n-l]$$

- fast Fourier transform (FFT)

$$x[m,n] \to \boxed{\text{FFT}} \to X[k,l] \to \bigotimes \to Y[k,l] \to \boxed{\text{iFFT}} \to y[m,n]$$

$$\begin{array}{c} \uparrow \\ H[k,l] \end{array}$$

- DSFT, which looks fine on paper but is impractical. Why? $\boxed{?}$

$$f[m,n] \to \boxed{\text{DSFT}} \to F(e^{\imath\mu}, e^{\imath\nu}) \to \bigotimes \to G(e^{\imath\mu}, e^{\imath\nu}) \to \boxed{\text{inverse DSFT}} \to g[m,n]$$

$$\begin{array}{c} \uparrow \\ H(e^{\imath\mu}, e^{\imath\nu}) \end{array}$$

FIR vs IIR tradeoffs: Filters can be finite impulse response (FIR) or infinite impulse response (IIR).

- FIR filters are always (BIBO) stable.
- FIR filters are easily made zero phase simply by designing them to be symmetric: $h[m,n] = h[-m,-n]$. Such designs will be "noncausal" but that property is acceptable for image processing and computer vision.
- Many techniques for 1D FIR filter design can be extended to 2D FIR filter design, including the windowing method and the frequency sampling method.
- Extending the Remez algorithm for min-max filter design is less straightforward.

Practical IIR filters are implemented using simple recursions (*not* by convolution). For given specifications on the frequency response (passband ripple, etc.), an efficiently implemented IIR filter requires fewer operations (additions and multiplications) than the corresponding FIR filter. However, zero phase design of IIR filters is more challenging, and testing for stability is quite complicated in 2D.

In EE416, we will mainly focus on FIR filters (IIR filters may be introduced in its graduate level course, EE616).

2.5.2 FIR filters

- Difference equation:

$$y[m,n] = \sum_{k=-N}^{N} \sum_{l=-N}^{N} h[k,l] x[m-k, n-l]$$

- For $N = 2$,

$$\begin{array}{ccccc} \circ & \circ & \circ & \circ & \circ \\ \circ & \circ & \circ & \circ & \circ \\ \circ & \circ & \otimes & \circ & \circ \\ \circ & \circ & \circ & \circ & \circ \\ \circ & \circ & \circ & \circ & \circ \end{array}$$

  where $\circ$ denotes an input point and $\times$ denotes an output point.
- Number of multipliers per output point $= (2N+1)^2$
- Transfer function:

$$H(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = \sum_{k=-N}^{N} \sum_{l=-N}^{N} h[k,l] e^{-\mathrm{i}(k\mu + l\nu)}$$

Ex. (Smoothing filter).

- PSF (or impulse response) of filter:

$$\frac{1}{16} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 2 & \boxed{4} & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

  where the box indicates the center element of the filter.
- Applying filters with the zero boundary condition (pixels outside the image are 0) gives

$$\underbrace{\begin{array}{ccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \end{array}}_{\text{Input image}} \Rightarrow \underbrace{\begin{array}{ccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 3 & 4 & 4 & 3 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ 0 & 0 & 4 & 12 & 16 & 16 & 12 \\ 0 & 0 & 4 & 12 & 16 & 16 & 12 \\ 0 & 0 & 3 & 9 & 12 & 12 & 9 \end{array}}_{\text{Output image}}$$

- Frequency response:
  - Observe that its impulse response is separable, i.e., $h[m,n] = h_1[m]h_1[n]$, where

$$\begin{aligned} 4h_1[n] &= [\ldots, 0, 1, 2, 1, 0, \ldots], \\ h_1[n] &= \frac{\delta[n+1] + 2\delta[n] + \delta[n-1]}{4}. \end{aligned}$$

  - The DTFT of $h_1[n]$ is $H_1(e^{\mathrm{i}w}) = \frac{1}{4}(e^{\mathrm{i}w} + 2 + e^{-\mathrm{i}w}) = \frac{1}{2}(1 + \cos(w))$.
  - Using the separability, the DSFT of $h[m,n]$ is

$$H(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = H_1(e^{\mathrm{i}\mu}) H_1(e^{\mathrm{i}\nu}) = \frac{1}{4}(1 + \cos(\mu))(1 + \cos(\nu)).$$

  - This is a *low pass* filter with $H(e^{\mathrm{i}0}, e^{\mathrm{i}0}) = 1$.
    Sketch: $\boxed{?}$

Ex. (Horizontal derivative filter).

- PSF (or impulse response) of filter:

$$\frac{1}{16}\cdot\begin{bmatrix} 2 & 0 & -2 \\ 4 & \boxed{0} & -4 \\ 2 & 0 & -2 \end{bmatrix}$$

where the box indicates the center element of the filter.
- Applying filters with the zero boundary condition gives

$$\underbrace{\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \end{matrix}}_{\text{Input image}} \Rightarrow \underbrace{\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ 0 & 0 & 6 & 6 & 0 & 0 & -6 \\ 0 & 0 & 8 & 8 & 0 & 0 & -8 \\ 0 & 0 & 8 & 8 & 0 & 0 & -8 \\ 0 & 0 & 6 & 6 & 0 & 0 & -6 \end{matrix}}_{\text{Output image}}$$

Ex. (Vertical derivative filter).

- PSF (or impulse response) of filter:

$$\frac{1}{16}\cdot\begin{bmatrix} 2 & 4 & 2 \\ 0 & \boxed{0} & 0 \\ -2 & -4 & -2 \end{bmatrix}$$

where the box indicates the center element of the filter.
- Applying filters with the zero boundary condition gives

$$\underbrace{\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \\ 0 & 0 & 0 & 16 & 16 & 16 & 16 \end{matrix}}_{\text{Input image}} \Rightarrow \underbrace{\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ 0 & 0 & 2 & 6 & 8 & 8 & 6 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & -6 & -8 & -8 & -6 \end{matrix}}_{\text{Output image}}$$

Ex. (High pass filter).

- Construct the filter impulse response $h[m,n] = h_1[m]h_1[n]$ with

$$\begin{aligned} 4h_1[n] &= [\ldots, 0, 1, -2, 1, 0, \ldots] \\ h_1[n] &= \frac{\delta[n+1] - 2\delta[n] + \delta[n-1]}{4}. \end{aligned}$$

Then, $h[m,n]$ is a separable function with

$$\frac{1}{16}\cdot\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}.$$

- The DTFT of $h_1[n]$ is $H_1(e^{\mathrm{i}w}) = \frac{1}{4}(e^{\mathrm{i}w} - 2 + e^{-\mathrm{i}w}) = \boxed{?}$
- Using the separability, the DSFT of $h[m,n]$ is

$$H(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu}) = H_1(e^{\mathrm{i}\mu})H_1(e^{\mathrm{i}\nu}) = \boxed{?}$$

- This is a *high pass* filter with $H(e^{\mathrm{i}0}, e^{\mathrm{i}0}) = 0$.
  Sketch: $\boxed{?}$

2.5.3 From discrete Fourier transform (DFT) to Fast Fourier transform (FFT) in 2D

2D MN-point discrete Fourier transform (DFT).

$$F[k,l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f[m,n]e^{-\mathrm{i}2\pi(km/M+ln/N)},\ k = 1,\ldots,M-1, l = 1,\ldots,N-1. \qquad \text{(forward)}$$

$$f[m,n] = \frac{1}{MN}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} F[k,l]e^{\mathrm{i}2\pi(km/M+ln/N)},\ m = 1,\ldots,M-1, n = 1,\ldots,N-1. \qquad \text{(inverse)}$$

- (M,N)-point circular extension of $x[m,n]$:

$$\tilde{x}[m,n] \triangleq x[m \mod M, n \mod N].$$

  where $m \mod M$ denotes the remainder of the division of m by M. Notice that this circular extension depends only on the values of $x[m,n]$ for $m = 1,\ldots,M-1$ and $n = 1,\ldots,N-1$. The circular boundary condition is essential in DFT properties.

Relationship between 2D DFT and DSFT. We can relate the 2D DFT to the DSFT as follows:

$$\begin{aligned}
X[k,l] &= \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n]e^{-\mathrm{i}2\pi(km/M+ln/N)} \\
&= \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} x[m,n]e^{-\mathrm{i}2\pi(km/M+ln/N)} \\
&= X(e^{\mathrm{i}\mu}, e^{\mathrm{i}\nu})|_{\mu=2\pi k/M, \nu=2\pi l/N} = X\left(e^{\frac{2\pi k}{M}}, e^{\frac{2\pi l}{N}}\right), \quad \begin{array}{c} k = 0,\ldots,M-1, \\ l = 0,\ldots,N-1. \end{array}
\end{aligned}$$

In other words, DFT coefficients are samples of the DSFT.

2D DFT properties.

Property	Space domain function	2D DFT
Linearity	$ax[m,n] + by[m,n]$	$aX[k,l] + bY[k,l]$
Separability	$x[m,n] = x_1[m]x_2[n]$	$X_1[k]X_2[l]$
Average value	$\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n]$	$\frac{1}{MN}X[0,0]$
Complex conjugate	$x^*[m,n]$	$X^*[-k \mod M, -l \mod N]$
Duality	$X[m,n]$	$MNx^*[-k \mod M, -l \mod N]$
Modulation	$e^{\mathrm{i}2\pi(k_0 m/M + l_0 n/N)}x[m,n]$	$X[(k-k_0) \mod N, (l-l_0) \mod M]$
Circular shift	$x[(m-m_0) \mod M, (n-n_0) \mod N]$	$e^{-\mathrm{i}2\pi(km_0/M + ln_0/N)}X[k,l]$
Circular space reversal	$x[-m \mod M, -n \mod N]$	$X[-k \mod M, -l \mod N]$
Convolution	$h[m,n] \circledast_{M,N} x[m,n]$	$H[k,l]X[k,l]$
Multiplication	$x[m,n]y[m,n]$	$\frac{1}{MN}X[k,l] \circledast_{M,N} Y[k,l]$
Parseval's Theorem	$\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x[m,n]y^*[m,n] = \frac{1}{MN}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} X[k,l]Y^*[k,l]$	

- Average value: Caution: $X[0,0]$ is often called the DC value, but it is the sum, not the average value!
- Convolution property: One of the main uses of the DFT is to implement fast convolution via the FFT. So the convolution property is particularly important. In fact, this property is probably the primary reason why the DFT is used so much more frequently than the many alternative orthogonal transforms.
  What happens if we take two sets of (M,N)-point DFT coefficients, multiply them, and take the inverse DFT? Clearly the answer cannot in general match linear convolution, because linear convolution of two signals results in a signal with larger support.
  The convolution property in 2D DFT is given with the circular convolution:

$$\begin{aligned} y[m,n] &= \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} h[k,l]x[(m-k) \mod M, (n-1) \mod N] \\ &\triangleq x[m,n] \circledast_{M,N} h[m,n]. \end{aligned}$$

- Many properties such as complex conjugate, duality, modulation, circular shift, circular space reversal, and complex conjugate, depend on $x[n \mod N, m \mod M]$, the (M,N)-point circular extension of a signal or the (M,N)-point circular extension of the DFT coefficients $X[k,l]$.

To read: [T1, §3-8.1, §3-8.2], [T2, §4.6.Some Properties of the 2-D DFT and IDFT].

2D fast Fourier transform (FFT). Brute-force evaluation of the 2D DFT or its inverse would require $\mathcal{O}((MN)^2)$ flops. By recognizing that the 2D DFT is a separable operation, we can reduce greatly the computation.

We can rewrite the DFT expression as follows:

$$F[k,l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f[m,n]e^{-\mathrm{i}2\pi(km/M + ln/N)} = \sum_{m=0}^{M-1} e^{-\mathrm{i}2\pi km/M}\left\{\sum_{n=0}^{N-1} f[m,n]e^{-\mathrm{i}2\pi ln/N}\right\}.$$

This is called the row-column decomposition. Apply the 1D DFT to each column of the image, and then apply the 1D DFT to each row of the result. Naturally we will want to use the FFT for these 1D DFTs, which thus reduces the computation to $\mathcal{O}(M \cdot N\log N)$ for the inner set of 1D FFTs, and then $\mathcal{O}(N \cdot M\log M)$ for the outer set of 1D FFTs, for a total of $\mathcal{O}(MN\log MN)$ flops. For a 512×512

image, the savings in using the row-column with 1D FFTs is about a factor of $15,000$ relative to the brute-force 2D DFT!

To read: [T1, §3-9], [T2, §4.11]

Homework (due by 10/8, 11:55 PM; Upload your solution and Matlab codes to Laulima/Assignments)

Prob. 1. Consider the following 2D system with input $x[m,n]$ and output $y[m,n]$:

$$y[m,n] = x[m,n] + \lambda\left(x[m,n] - \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} x[m-k, n-l]\right).$$

(a) Is this a linear system? Is this a space invariant system?
(b) What is the 2D impulse response of this system, $h[m,n]$?
(c) Derive the frequency response, $H(e^{i\mu}, e^{i\nu})$. (Hint: Use Euler's formula.)
(d) Describe how the filter behaves when $\lambda > 0$ and large.
(e) Describe how the filter behaves when $-1 < \lambda < 0$.

Prob. 2 (Computer project). Consider the following low pass filter:

$$h[m,n] = \begin{cases} \frac{1}{25}, & |m| \le 2, |n| \le 2 \\ 0, & \text{otherwise.} \end{cases}$$

The unsharp mask filter is then given by

$$g[m,n] = \delta[m,n] + \lambda(\delta[m,n] - h[m,n]).$$

where $\lambda > 0$.
(a) Calculate an analytical expression for $H(e^{i\mu}, e^{i\nu})$, the DSFT of $h[m,n]$. Use Matlab to plot the magnitude of the frequency response $|H(e^{i\mu}, e^{i\nu})|$. (Make sure to label the axes properly and plot on over the region $[-\pi, \pi] \times [-\pi, \pi]$.)
(b) Calculate an analytical expression for $G(e^{i\mu}, e^{i\nu})$, the DSFT of $g[m,n]$. Use Matlab to plot the magnitude of the frequency response $|G(e^{i\mu}, e^{i\nu})|$ with $\lambda = 1.5$. (Make sure to label the axes properly and plot on over the region $[-\pi, \pi] \times [-\pi, \pi]$.)
(c) Writing your own codes for 2D filtering/convolution, apply your sharpening filter to blurry-moon.tif for $\lambda = 0.2, 0.8, 1.5$. In your report, include blurry-moon.tif and the output image for $\lambda = 1.5$, and also your Matlab code used for the filtering. In particular, all filters will be implemented using zero boundary conditions along edges, and pixel values will be clipped to a range of $[0, 255]$.

- Clipping: Image processing operations will sometimes cause a pixel to exceed the value 255 or go below the value 0. In these cases, you should clip the pixel's value before displaying or exporting the image.

$$y[m,n] = \begin{cases} 0, & x[m,n] < 0, \\ 255, & x[m,n] > 255, \\ x[m,n], & 0 \le x[m,n] \le 255. \end{cases}$$

In addition, use the following Matlab functions to read, display, and write images:

- Reading images: Read an image file, img.tif, into the Matlab workspace using the command
  `x=imread('img.tif');`
  This will produce an image matrix x of data type *uint8*.

- Displaying images: You can display the image array x with the following commands:

  ```
  image(x); colormap(gray(256)); truesize;
  ```

  If x is a gray scale image, the colormap function is needed to tell Matlab which color to display for each possible pixel value. The `truesize` command, which is included in the image processing toolbox, maps each image pixel to a single display pixel to avoid any interpolation on the display.
- Writing images: When producing a lab report document, you should strive to present the best representation of your output images. Therefore it is best to export your images to a lossless file format, such as TIFF or BMP, which can then be imported into your lab report document. You can write the image array x to a file using the `imwrite` function:

  ```
  imwrite(x, 'output.tif');
  ```

  Note that if x is of type *uint8*, the `imwrite` function assumes a dynamic range of $[0, 255]$, and will clip any values outside that range. If x is of type *double*, then `imwrite` assumes a dynamic range of $[0, 1]$, and will linearly scale to the range $[0, 255]$, clipping values outside that range, before writing out the image to a file. To convert the image to type *uint8* before writing, you can use

  ```
  imwrite(unit8(x), 'output.tif');
  ```