

Module 4

2D Fourier Transform

- Fourier Transform Basics
- Sinusoidal Image
- 2D Discrete Fourier Transform
- Meaning of Image Frequencies
- Sampling Theorem
- Some Important 2D DFT Pairs

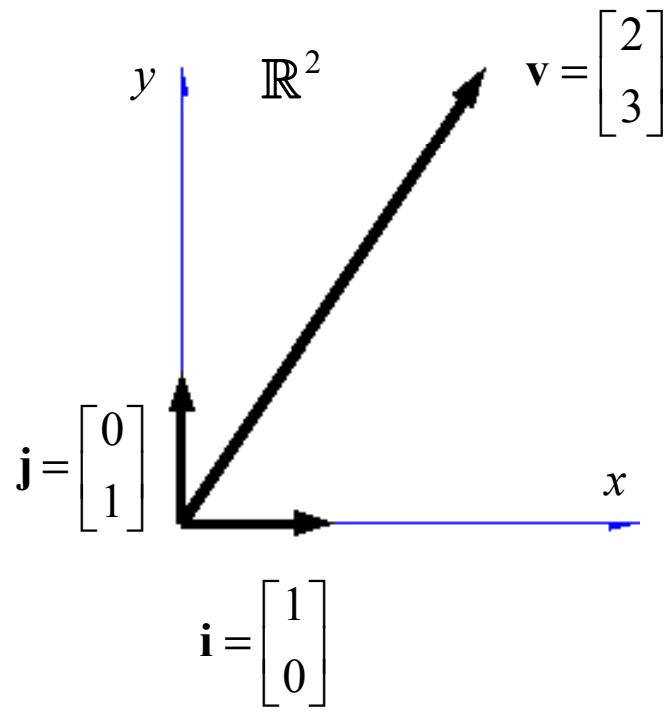
Joseph Fourier



*Yesterday was my 21st birthday, at that age
Newton and Pascal had already acquired many
claims to immortality. - In a letter by Fourier*

- Fourier was a math professor. He studied heat conduction. His PhD advisor was Lagrange.
- His most important contribution was development of the idea that an arbitrary function could be written as an infinite sum (series) of sines and cosines.
- Ironically, this work was bitterly criticized by both Laplace and Lagrange at the time!

Vectors in \mathbb{R}^2



- The natural basis: $\{\mathbf{i}, \mathbf{j}\}$
- Representation: write an arbitrary vector \mathbf{v} as a linear combination of the basis.
- **Question:** how do we do this?
- **Answer:** use the **dot product** (inner product).
- For two vectors $\mathbf{v} = [v_1 \quad v_2]^T$ and $\mathbf{w} = [w_1 \quad w_2]^T$, the dot product is often written using angle brackets:
$$\langle \mathbf{v}, \mathbf{w} \rangle = v_1 w_1 + v_2 w_2$$
- Recall that the dot product is a **scalar**.
- Any arbitrary $\mathbf{v} \in \mathbb{R}^2$ can be written as $\mathbf{v} = \langle \mathbf{v}, \mathbf{i} \rangle \mathbf{i} + \langle \mathbf{v}, \mathbf{j} \rangle \mathbf{j}$.

- For example, on the last page we had $\mathbf{v} = [2 \ 3]^T$. Using the dot product,

$$\begin{aligned}
\mathbf{v} &= \langle \mathbf{v}, \mathbf{i} \rangle \mathbf{i} + \langle \mathbf{v}, \mathbf{j} \rangle \mathbf{j} \\
&= \left\langle \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right\rangle \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \left\langle \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right\rangle \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\
&= (2 \cdot 1 + 3 \cdot 0) \begin{bmatrix} 1 \\ 0 \end{bmatrix} + (2 \cdot 0 + 3 \cdot 1) \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \quad \checkmark
\end{aligned}$$

- This is a powerful way to think about things because it works for **any** orthonormal basis.
- **Fact:** another orthonormal basis for \mathbb{R}^2 is given by the vectors

$$\mathbf{e}_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}^T \quad \mathbf{e}_2 = \begin{bmatrix} \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}^T$$

- For $\mathbf{v} = [2 \ 3]^T$, you can easily check that

$$\langle \mathbf{v}, \mathbf{e}_1 \rangle \mathbf{e}_1 + \langle \mathbf{v}, \mathbf{e}_2 \rangle \mathbf{e}_2 = \left\langle \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \right\rangle \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} + \left\langle \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \right\rangle \begin{bmatrix} \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \quad \checkmark$$

Some Important Things About the Dot Product

- In higher dimensional spaces like \mathbb{R}^{100} , it becomes convenient to use “Capital Sigma” notation to write the dot product:

$$\langle \mathbf{v}, \mathbf{w} \rangle = v_1 w_1 + v_2 w_2 + \cdots + v_{100} w_{100} = \sum_{k=1}^{100} v_k w_k$$

- But the whole procedure still works exactly like before in \mathbb{R}^2 .
- For an orthonormal basis $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_{100}\}$, the representation of \mathbf{v} is

$$\mathbf{v} = \langle \mathbf{v}, \mathbf{e}_1 \rangle \mathbf{e}_1 + \langle \mathbf{v}, \mathbf{e}_2 \rangle \mathbf{e}_2 + \cdots + \langle \mathbf{v}, \mathbf{e}_{100} \rangle \mathbf{e}_{100} = \sum_{k=1}^{100} \langle \mathbf{v}, \mathbf{e}_k \rangle \mathbf{e}_k$$

- The operation of computing the dot products $\langle \mathbf{v}, \mathbf{e}_k \rangle$ defines a **transform**. It gives the representation (or coordinates) of \mathbf{v} relative to the basis $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_{100}\}$.
- The operation of writing \mathbf{v} as sum of the basis vectors defines an **inverse transform**.

Important Things About Dot Products

- One thing you may not have been taught about the dot product is: if the vectors have **complex** entries, then you must **conjugate** one of the vectors in the dot product.
- In engineering, we always conjugate the **second** vector.
- The dot product then becomes

$$\langle \mathbf{v}, \mathbf{w} \rangle = \sum_k v_k w_k^*$$

- To write a vector \mathbf{v} as a sum of the basis, you still add up the dot products times the basis vectors just like before:

$$\mathbf{v} = \sum_k \langle \mathbf{v}, \mathbf{e}_k \rangle \mathbf{e}_k$$

- **Note:** sometimes in math and physics, it is the **first** vector that gets conjugated in the dot product. But in engineering, we **always** conjugate the **second** vector.

Infinite Dimensional Vector Spaces

- These ideas extend easily to infinite dimensional vector spaces.
- In your undergraduate Signals & Systems course, you dealt with discrete-time signals like $x[n]$.
- You can consider these signals to be vectors in \mathbb{C}^∞ or \mathbb{R}^∞ .
- The dot product in these spaces works just like you think it should:

$$\langle x[n], y[n] \rangle = \sum_{n=-\infty}^{\infty} x[n] y^*[n].$$

- For an orthonormal basis $\{\dots, \mathbf{e}_{-1}, \mathbf{e}_0, \mathbf{e}_1, \dots\}$, you can write $x[n]$ as

$$x[n] = \sum_{k=-\infty}^{\infty} \langle x[n], \mathbf{e}_k \rangle \mathbf{e}_k$$

Natural Basis in \mathbb{C}^∞ or \mathbb{R}^∞

- Think of the signals $\delta[n]$, $\delta[n-1]$, $\delta[n+1]$, and $\delta[n-k]$ from your undergraduate Signals & systems course.
- Each one of them is “turned on” in exactly one place where it is equal to one.
- For vector spaces of discrete-time signals $x[n]$, the set of **translates** $\delta[n-k]$ of the Kronecker delta for $k \in \mathbb{Z}$ plays the **same role** that the basis $\{\mathbf{i}, \mathbf{j}\}$ plays in \mathbb{R}^2 ; i.e., it is the **natural basis**.
- The dot product of $x[n]$ with the basis vector $\delta[n-k]$ is the number

$$\langle x[n], \delta[n-k] \rangle = \sum_{n=-\infty}^{\infty} x[n] \delta[n-k] = x[k].$$

- The signal $x[n]$ can be written as a sum of the basis just like before:

$$x[n] = \cdots + x[-1]\delta[n+1] + x[0]\delta[n] + x[1]\delta[n-1] + \cdots$$

$$= \sum_{k=-\infty}^{\infty} x[k]\delta[n-k] = \sum_{k=-\infty}^{\infty} \langle x[n], \delta[n-k] \rangle \delta[n-k].$$

Uncountably Infinite Dimensions

- We can let the number of dimensions grow even larger, so that it is equal to the number of real numbers in \mathbb{R} .
- The vectors in these spaces are exactly the continuous-time signals $x(t)$ from your undergraduate Signals & Systems course.
- Everything still works just like it did before, but now you have to use “Capital S” adding instead of “Capital Sigma” adding.
- The dot product in these spaces is given by

$$\langle x(t), y(t) \rangle = \int_{-\infty}^{\infty} x(t) y^*(t) dt.$$

- For an orthonormal basis $\{ \mathbf{e}_\tau \}_{\tau \in \mathbb{R}}$, you can write $x(t)$ just like before:

$$x(t) = \int_{-\infty}^{\infty} \langle x(\tau), \mathbf{e}_\tau \rangle \mathbf{e}_\tau d\tau$$

Uncountably Infinite Dimensions

- The natural basis is a little bit harder to talk about in this case, because **distribution theory** is required to treat the Dirac delta $\delta(t)$ rigorously.
- Here, “distribution theory” means the theory of generalized functions. It has nothing to do with “probability distributions.”
- We won’t go into the details (take ECE 5213, DSP, for that).
- Nevertheless, it should make intuitive sense to you at this point that the natural basis is given by the translates of the Dirac delta:

$$\{ \delta(t - \tau) \}_{\tau \in \mathbb{R}}$$

- The dot product of $x(t)$ with a basis function is the number

$$\langle x(t), \delta(t - \tau) \rangle = \int_{-\infty}^{\infty} x(t) \delta(t - \tau) dt = x(\tau).$$

- You write $x(t)$ as a sum of the basis just like before, by adding up the dot products times the basis vectors:

$$x(t) = \int_{-\infty}^{\infty} x(\tau) \delta(t - \tau) d\tau.$$

Change of Basis

- Recall from calculus that changing coordinate systems can sometimes turn a hard problem into an easier one.
 - For example, when computing a solid of revolution, switching from rectangular coordinates to cylindrical coordinates often makes the problem easier.
- Similarly, when working on signals or images, it is sometimes very useful to change from the natural basis to some other basis.
- This can let you look at the signal in a different way where it's easier to see different information that might be important for a particular problem.
- It can also turn some “hard” problems into easy ones. For example, an appropriate change of basis turns convolution into multiplication (more on this later).

Spectral Basis for Discrete-Time Signals

- **Fact:** the set of signals $\{e^{j2\pi fn}\}$ for $f \in (-\frac{1}{2}, \frac{1}{2}]$ is an orthonormal basis for the vector space of signals $x[n]$ that you dealt with in your undergraduate Signals & Systems course.
- Because the basis vectors (basis signals) have **complex** entries, it is important for you to remember to conjugate when you take dot products!
- How do we write $x[n]$ as a sum of this basis?
 1. Take the dot product of $x[n]$ with each of the basis vectors. This gives us a (complex) number for each basis vector.
 2. Add up the dot products times the basis vectors to get $x[n]$.
- Since there are an infinite number of basis vectors, we will have a lot of dot products to keep track of... one for each $f \in (-\frac{1}{2}, \frac{1}{2}]$.
- Notice that this defines a **function**. The domain is $f \in (-\frac{1}{2}, \frac{1}{2}]$ and the range is complex numbers (the dot products).
- We will call it $X(e^{j2\pi f})$ and use it to keep track of the dot products.

1D Discrete-Time Fourier Transform

- **Step 1:** take the dot product of $x[n]$ with a basis vector (don't forget to conjugate the entries of the second vector!):

$$X(e^{j2\pi f}) = \left\langle x[n], e^{j2\pi fn} \right\rangle = \sum_{n=-\infty}^{\infty} x[n] e^{-j2\pi fn}$$

- This is called the **discrete-time Fourier transform (DTFT)**.
- **Step 2:** add up the dot products times the basis functions to get $x[n]$. Because the number of basis functions is uncountable, we have to use “Capital S” adding:

$$x[n] = \int_{-\frac{1}{2}}^{\frac{1}{2}} \left\langle x[n], e^{j2\pi fn} \right\rangle e^{j2\pi fn} df = \int_{-\frac{1}{2}}^{\frac{1}{2}} X(e^{j2\pi f}) e^{j2\pi fn} df$$

- This is called the **inverse DTFT**.

Why Use the DTFT?

- Looking at the graph of $X(e^{j2\pi f})$ can make it easy to see certain things that are hard to see from the graph of $x[n]$.
 - For example, if $x[n]$ is a digital audio signal, changing to the spectral basis and looking at the graph of $X(e^{j2\pi f})$ may make it easy to see if there is too much bass or not enough midrange.
- Let H be a discrete-time LTI system with impulse response $h[n]$ and frequency response $H(e^{j2\pi f})$.
- For an arbitrary input signal $x[n]$, the output signal is given by the convolution $y[n] = x[n] * h[n]$ which can take a lot of work to compute.
- But if $x[n]$ is an **eigenfunction** of the system, then the output is easy to compute. It is given by $y[n] = \lambda x[n]$, where λ is an eigenvalue that is complex-valued in general.
- For an arbitrary input signal $x[n]$, the output will be easy to compute if we can write $x[n]$ as a sum of eigenfunctions. Each term of the sum will just get multiplied by a complex eigenvalue.

- **Fact:** the DTFT basis signals are all eigenfunctions of **any** discrete-time LTI system. The eigenvalues are given by the system frequency response.
- **Proof:** let the system input be $x[n] = e^{j2\pi f n}$. The output is given by

$$\begin{aligned}
 y[n] &= x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[n-k]h[k] \\
 &= \sum_{k=-\infty}^{\infty} e^{j2\pi f(n-k)}h[k] \\
 &= e^{j2\pi f n} \left[\sum_{k=-\infty}^{\infty} h[k]e^{-j2\pi fk} \right] = H(e^{j2\pi f})x[n]. \quad \blacksquare
 \end{aligned}$$

- Use the DTFT to write an arbitrary input $x[n]$ as a sum of eigenfunctions. The system output is given by

$$\begin{aligned}
 y[n] &= H\{x[n]\} = H\left\{\int_{-\frac{1}{2}}^{\frac{1}{2}} X(e^{j2\pi f})e^{j2\pi fn} df\right\} \\
 &= \int_{-\frac{1}{2}}^{\frac{1}{2}} X(e^{j2\pi f})H\{e^{j2\pi fn}\} df \\
 &= \int_{-\frac{1}{2}}^{\frac{1}{2}} X(e^{j2\pi f})H(e^{j2\pi f})e^{j2\pi fn} df = \int_{-\frac{1}{2}}^{\frac{1}{2}} Y(e^{j2\pi f})e^{j2\pi fn} df.
 \end{aligned}$$

- In other words, for an arbitrary input signal $x[n]$, the output of the LTI system H is given by

$$y[n] = \text{IDTFT} \left\{ X(e^{j2\pi f})H(e^{j2\pi f}) \right\}.$$

- If we represent the input signal using the natural basis, then convolution is required to compute the system output.
- But if we use the DTFT to change to the spectral basis, then the output is given by multiplication.

1D Fourier Transform

- **Fact:** the set of signals $\{e^{j2\pi ft}\}$ for $f \in \mathbb{R}$ is an orthonormal basis for the vector space of signals $x(t)$ that you dealt with in your undergraduate Signals & Systems course.
- Let's write $x(t)$ as a sum of this basis.
- **Step 1:** take the dot product of $x(t)$ with a basis vector (don't forget to conjugate the entries of the second vector!):

$$X(f) = \langle x(t), e^{j2\pi ft} \rangle = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft} dt$$

- This is called the **Fourier transform (FT)**.
- **Step 2:** add up the dot products times the basis functions to get $x(t)$:

$$x(t) = \int_{-\infty}^{\infty} \langle x(t), e^{j2\pi ft} \rangle e^{j2\pi ft} df = \int_{-\infty}^{\infty} X(f) e^{j2\pi ft} df$$

- This is called the **inverse Fourier Transform (IFT)**.

Orthogonal Basis

- Going back to \mathbb{R}^2 , consider the basis

$$\mathbf{e}_1 = \begin{bmatrix} 3 & 0 \end{bmatrix}^T \quad \mathbf{e}_2 = \begin{bmatrix} 0 & 3 \end{bmatrix}^T$$

- This basis is **orthogonal**, but it is **not orthonormal** since each basis vector has length **three** instead of one.
- If we try to write the vector $\mathbf{v} = \begin{bmatrix} 2 & 3 \end{bmatrix}^T$ as a sum of this basis using our method, we get

$$\begin{aligned}\langle \mathbf{v}, \mathbf{e}_1 \rangle \mathbf{e}_1 + \langle \mathbf{v}, \mathbf{e}_2 \rangle \mathbf{e}_2 &= \left\langle \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 3 \\ 0 \end{bmatrix} \right\rangle \begin{bmatrix} 3 \\ 0 \end{bmatrix} + \left\langle \begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 0 \\ 3 \end{bmatrix} \right\rangle \begin{bmatrix} 0 \\ 3 \end{bmatrix} \\ &= (2 \cdot 3 + 3 \cdot 0) \begin{bmatrix} 3 \\ 0 \end{bmatrix} + (2 \cdot 0 + 3 \cdot 3) \begin{bmatrix} 0 \\ 3 \end{bmatrix} \\ &= 6 \begin{bmatrix} 3 \\ 0 \end{bmatrix} + 9 \begin{bmatrix} 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 18 \\ 27 \end{bmatrix} = 9 \begin{bmatrix} 2 \\ 3 \end{bmatrix} \quad \times\end{aligned}$$

- We get the **wrong** answer.
 - Because the basis was not orthonormal, the dot products were all too big by a factor of 3.
 - We multiplied these dot products that were all too big times basis vectors that were all too long by a factor of 3.
- All together, our answer is **too big** by a factor of $3 \times 3 = 9$, which is the length of a basis vector squared.
- When the basis is orthogonal, but **not** orthonormal, our method needs some “fixing up.”
 - We can divide by the squared length of a basis vector when we compute the dot products (**fix up on the transform**),
 - or we can divide by the squared length of a basis vector when we add up the dot products times the basis vectors (**fix up on the inverse transform**).
 - or we can divide by the length of a basis vector in both places (**fix up on both the transform and the inverse transform**).

- You will often see the DTFT basis written in terms of radian frequency instead of Hertzian frequency.
- The basis is then $\{e^{j\omega n}\}$ for $\omega \in (-\pi, \pi]$.
- This basis is **orthogonal**, but it is **not** orthonormal.
 - Using the Riemann-Lebesgue lemma of distribution theory, one can show that the length of each basis vector is $\sqrt{2\pi}$, not one (take ECE 5213, DSP, for the details).
 - Because of this, we have to divide by 2π somewhere to make our method work.
- In engineering, the most common convention is to divide by 2π on the inverse transform. This gives the alternate DTFT/IDTFT definition

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$$

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(e^{j\omega}) e^{j\omega n} d\omega$$

which is found in many textbooks.

- Likewise, you will often see the FT basis written in terms of radian frequency instead of Hertzian frequency.
- The basis is then given by $\{e^{j\omega t}\}$ for $\omega \in \mathbb{R}$.
- This basis is again **orthogonal** but **not** orthonormal. Each basis vector has length $\sqrt{2\pi}$ instead of one.
- This gives the alternate FT/IFT definition

$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt$$

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t} d\omega$$

1D Discrete Fourier Transform

- Consider a finite-length discrete-time signal $x[n]$ defined for $0 \leq n \leq N-1$.
- Such signals can be considered as vectors in \mathbb{R}^N or \mathbb{C}^N .
- Since the space is N -dimensional, there are N vectors in a basis.
- The spectral (or Fourier) basis is given by

$$\left\{ e^{j \frac{2\pi k}{N} n} \right\}_{0 \leq k \leq N-1}, \quad 0 \leq n \leq N-1$$

- The **discrete Fourier transform or DFT** is defined (as always) by taking the dot product of the signal with a basis vector:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j \frac{2\pi k}{N} n}, \quad 0 \leq k \leq N-1$$

- The **inverse DFT (IDFT)** is then obtained by adding up the dot products times the basis vectors and dividing out the squared length of a basis vector:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j \frac{2\pi k}{N} n}, \quad 0 \leq n \leq N-1$$

- It is customary to write the DFT and IDFT using the shorthand notation

$$W_N = e^{-j2\pi/N}.$$

- NOTE: W_N is a complex **number**... a **constant** (for any fixed choice of N).
- The forward and reverse transforms then become

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, \quad 0 \leq k \leq N-1$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn}, \quad 0 \leq n \leq N-1$$

- Notice that the conjugation operation is **built in** to the symbol W_N .
 - This makes the “–” sign in the exponent appear on the inverse transform instead of the forward transform when the “ W_N ” notation is used.
- **Note:** it's important that the DFT math starts with $n = 0$ and $k = 0$.
- In a Matlab DFT array, you have to remember that the first element has array index 1, but it's for $k = 0$.

n D Sinusoids – Continuous Case

- The 1D cosine: $\cos(2\pi ft)$.
- Instead of the scalar t , in n D we have a position **vector** $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_n]^T$.
- Instead of the scalar f , we have a frequency **vector** $\mathbf{u} = [u_1 \ u_2 \ \dots \ u_n]^T$.
- But the argument of cos must still be **scalar**.
- It is the dot product of the frequency vector and the position vector:

$$\mathbf{u}^T \mathbf{x} = u_1 x_1 + u_2 x_2 + \dots + u_n x_n$$

- The n D cosine: $\cos(2\pi \mathbf{u}^T \mathbf{x})$.
- The n D sine: $\sin(2\pi \mathbf{u}^T \mathbf{x})$.
- The n D complex exponential: $e^{j2\pi \mathbf{u}^T \mathbf{x}} = \cos(2\pi \mathbf{u}^T \mathbf{x}) + j \sin(2\pi \mathbf{u}^T \mathbf{x})$.
- Note: if $n = 1$, these reduce to the usual 1D definitions.
- We'll develop intuition about them later; for now we focus on the math.

nD Fourier Transform

- For the n D continuous case, the spectral basis is $\left\{e^{j2\pi \mathbf{u}^T \mathbf{x}}\right\}_{\mathbf{u} \in \mathbb{R}^n}$.
- Let $s(\mathbf{x})$ be an n D signal.
- For the FT, you simply take the dot product of $s(\mathbf{x})$ with a basis signal (as always) – don't forget to conjugate the entries of the second vector:

$$S(\mathbf{u}) = \int_{\mathbb{R}^n} s(\mathbf{x}) e^{-j2\pi \mathbf{u}^T \mathbf{x}} d\mathbf{x}$$

- You can “write out” the n D integral like this:

$$S(\mathbf{u}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} s(\mathbf{x}) e^{-j2\pi(u_1 x_1 + u_2 x_2 + \cdots + u_n x_n)} dx_1 dx_2 \cdots dx_n$$

- The n D inverse Fourier Transform is just what you would expect:

$$s(\mathbf{x}) = \int_{\mathbb{R}^n} S(\mathbf{u}) e^{j2\pi \mathbf{u}^T \mathbf{x}} d\mathbf{u} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} S(\mathbf{u}) e^{j2\pi(u_1 x_1 + u_2 x_2 + \cdots + u_n x_n)} du_1 du_2 \cdots du_n$$

nD Fourier Transform

- If you want to use a radian frequency vector $\omega = 2\pi\mathbf{u}$, then you have to divide out a “ 2π ” for **each** dimension.
- In engineering, we usually do it on the inverse transform.
- So in terms of radian frequency, the continuous n D FT and IFT become:

$$S(\boldsymbol{\omega}) = \int_{\mathbb{R}^n} s(\mathbf{x}) e^{-j\boldsymbol{\omega}^T \mathbf{x}} d\mathbf{x}$$

$$s(\mathbf{x}) = \frac{1}{(2\pi)^n} \int_{\mathbb{R}^n} S(\boldsymbol{\omega}) e^{j\boldsymbol{\omega}^T \mathbf{x}} d\boldsymbol{\omega}$$

Important Notation Changes

- For the rest of the module, we will write (m, n) for image spatial coordinates. For discussing Fourier transforms, this almost always means *(column, row)*.
 - Up to now we have thought of (m, n) as (row, col) , consistent with Matlab and usual matrix “row-col” addressing.
 - But for the FT we think of 2D functions $f(x, y)$, where x is horizontal.
- We will also usually think of the frequency in cycles per image (cpi) instead of cycles per sample (pixel), since this is easier to interpret visually.
- For the case of 2D digital images, we will write the 2D frequency vector as $\mathbf{u} = [u \quad v]^T$, where u = “horizontal frequency” and v = “vertical frequency.”

Sinusoidal Images

- We shall make frequent discussion in this module of image **frequency content**.
- The image having the **simplest** frequency content is the **sinusoidal image**.

Sinusoidal Images

- A **discrete sine image** I is given by

$$I(m, n) = \sin\left[2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right]$$

and a **discrete cosine image** is given by

$$I(m, n) = \cos\left[2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right]$$

for $0 \leq m \leq M-1$, $0 \leq n \leq N-1$

- Here, m is the column and n is the row.

Important Note

- For the mathematics of the 2D Discrete Fourier Transform (DFT) to work out, it is **very important** that
 - The first row in the image is row ZERO, **not** row one!
 - The first column is column ZERO, **not** column one!
- If the image is in a Matlab array, you must remember that the first row has index 1, but in terms of the mathematics, it is row 0!
- Similarly, in a Matlab array, the first column has index 1, but in terms of the mathematics it is column 0!

Allowable Values for the Frequencies

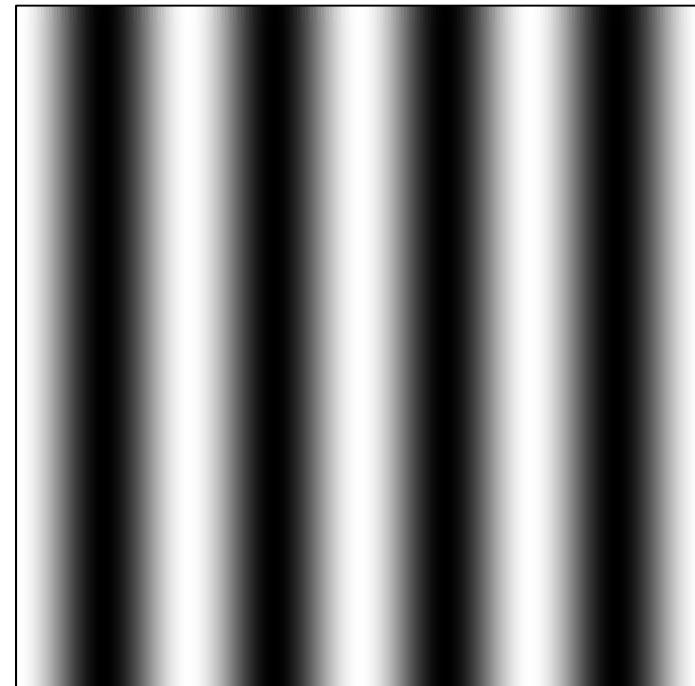
- The horizontal frequency u and vertical frequency v can be any **real** numbers.
- However, for constructing the 2D DFT basis for an $M \times N$ digital image, we will only need
 - u to be integers between 0 and $M-1$.
 - v to be integers between 0 and $N-1$.
- For this reason, u and v will often show up as integers.

Cosine Images

- If $M = N = 256$, $u = 4$, and $v = 0$, then the image is

$$I(m, n) = \cos [2\pi \frac{4}{256} m]$$

- Since the image does not depend on n , **every row is the same**.
- Every row is a cosine with 4 cycles per image:



- If $M = N = 256$, $u = 0$, and $v = 6$, then the image is

$$I(m, n) = \cos [2\pi \frac{6}{256} n]$$

- Since the image does not depend on m , **every column is the same**.
- Every column is a cosine with 6 cycles per image:



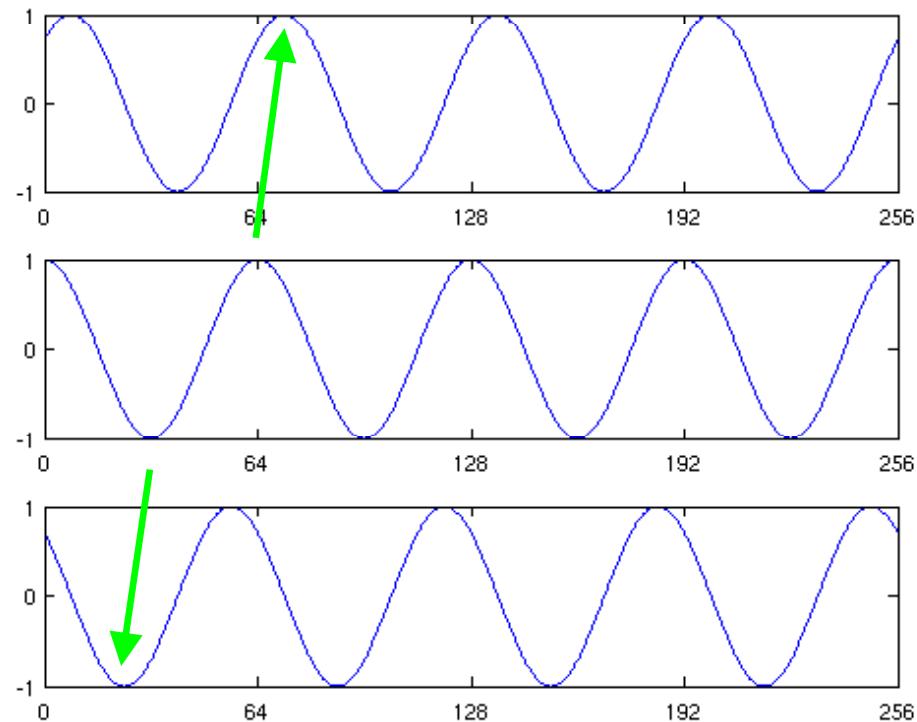
- For a moment, think of the 1D signal $\cos(2\pi ft)$.
- The graph of $\cos(2\pi ft - \phi)$ is the same, but shifted right by an amount $\phi/(2\pi f)$ on the t axis, since

$$\cos(2\pi ft - \phi) = \cos\left[2\pi f\left(t - \frac{\phi}{2\pi f}\right)\right].$$

$$\cos\left[\frac{2\pi 4}{256}t - \frac{\pi}{4}\right]$$

$$\cos\left[\frac{2\pi 4}{256}t\right]$$

$$\cos\left[\frac{2\pi 4}{256}t + \frac{\pi}{4}\right]$$



- Now, let $M = N = 256$, $u = 4$, and $v = 6$. The image is

$$I(m, n) = \cos [2\pi(\frac{4}{256}m + \frac{6}{256}n)]$$

- On the first row, $n=0$, so

$$I(m, n) = \cos [2\pi \frac{4}{256}m]$$

- On the 2nd row, $n=1$, so

$$I(m, n) = \cos [2\pi \frac{4}{256}m + \frac{2\pi 6}{256}]$$

- On the 3rd row, $n=2$, so

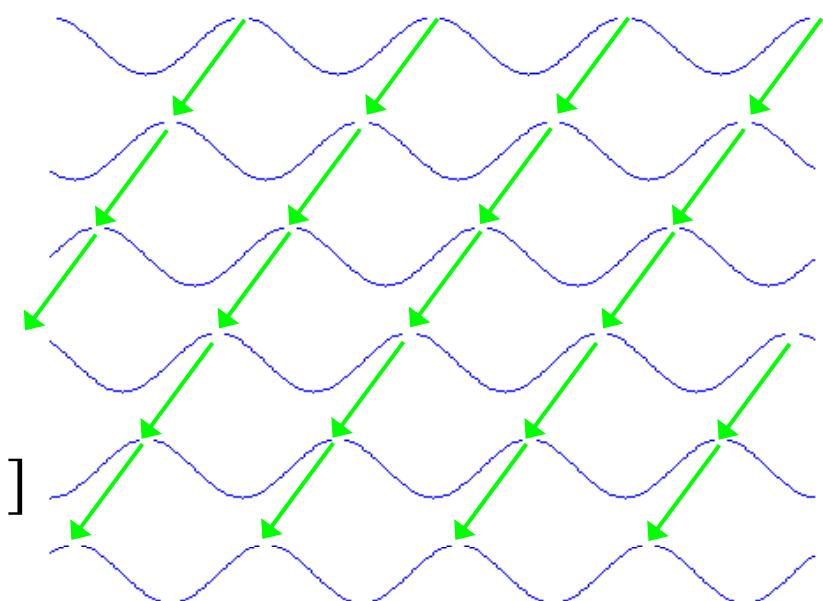
$$I(m, n) = \cos [2\pi \frac{4}{256}m + \frac{2\pi 12}{256}]$$

- On the k^{th} row, $n=k-1$, so

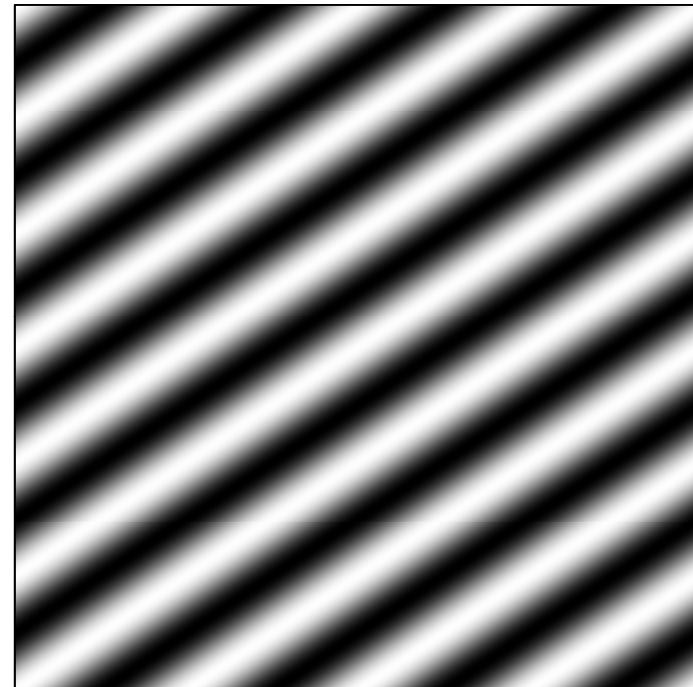
$$I(m, n) = \cos [2\pi \frac{4}{256}m + \frac{2\pi 6(k-1)}{256}]$$

- They are all cosines with horizontal frequency 4 cpi.

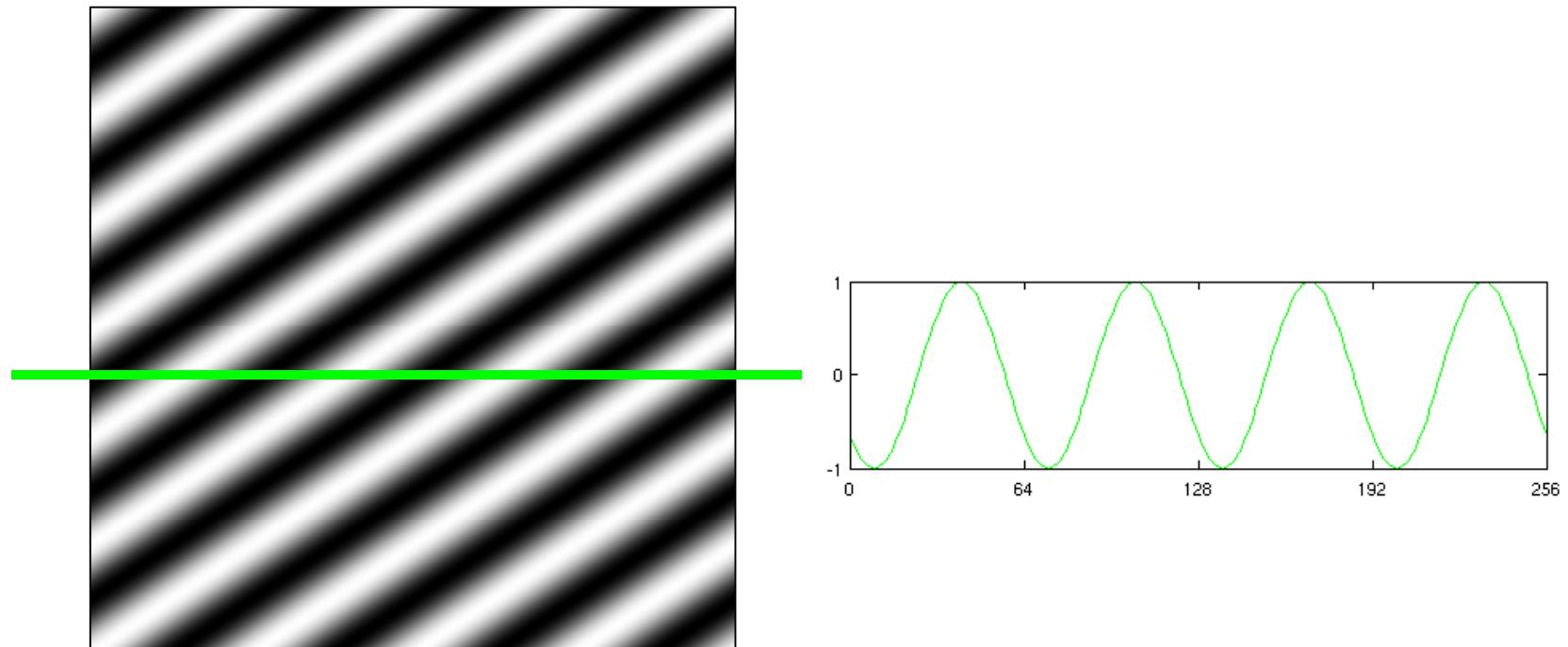
- But each row is shifted by $\phi = \frac{2\pi 6}{256}$ compared to the row above.



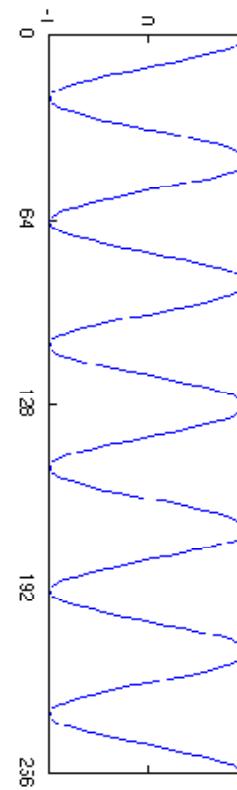
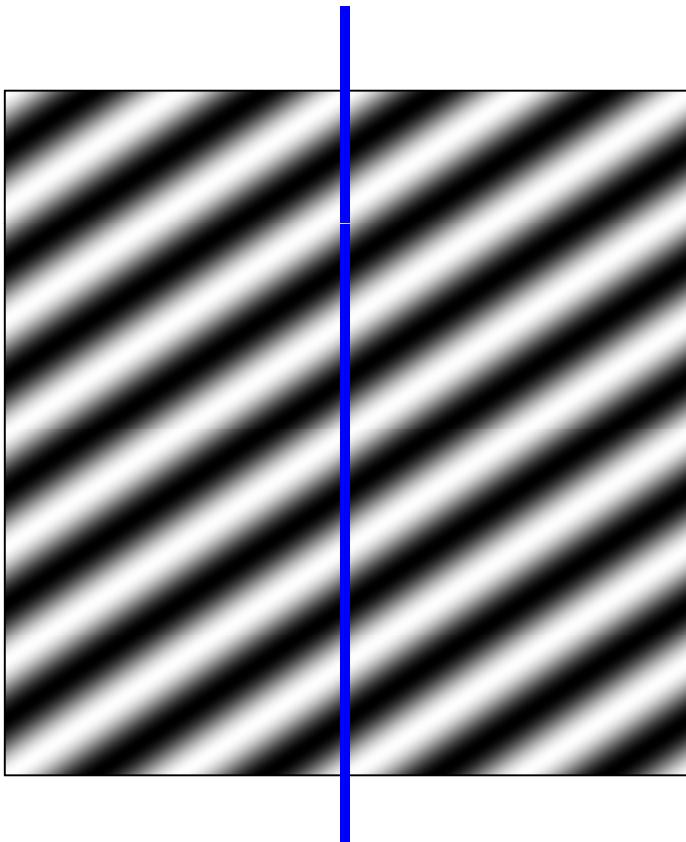
- As an image, this looks like:
 $(M = N = 256, u = 4, v = 6)$
- Horizontally, every row goes through $u=4$ cycles across the image (but they start at different phase offsets).
- Vertically, every column goes through $v=6$ cycles down the image (but starting at different phase offsets).



- If you take any horizontal slice through the image, you get a 1D cosine with frequency $u=4$ cpi.

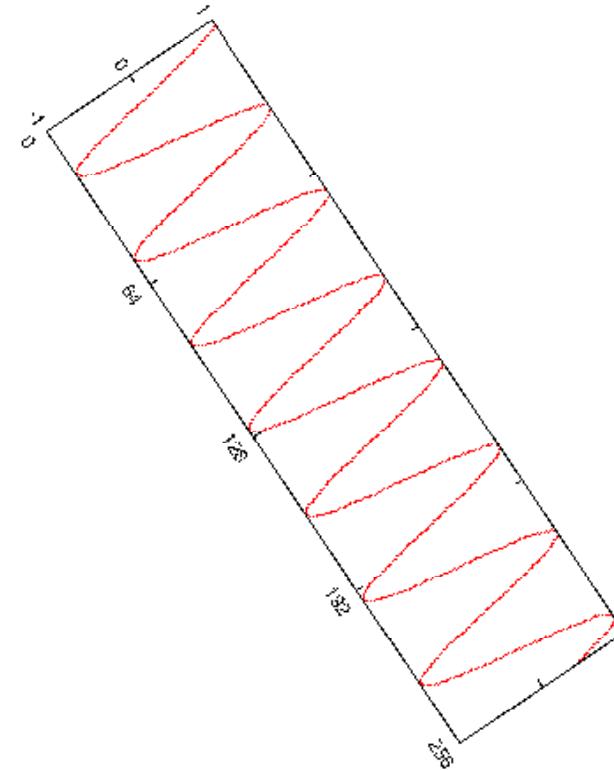
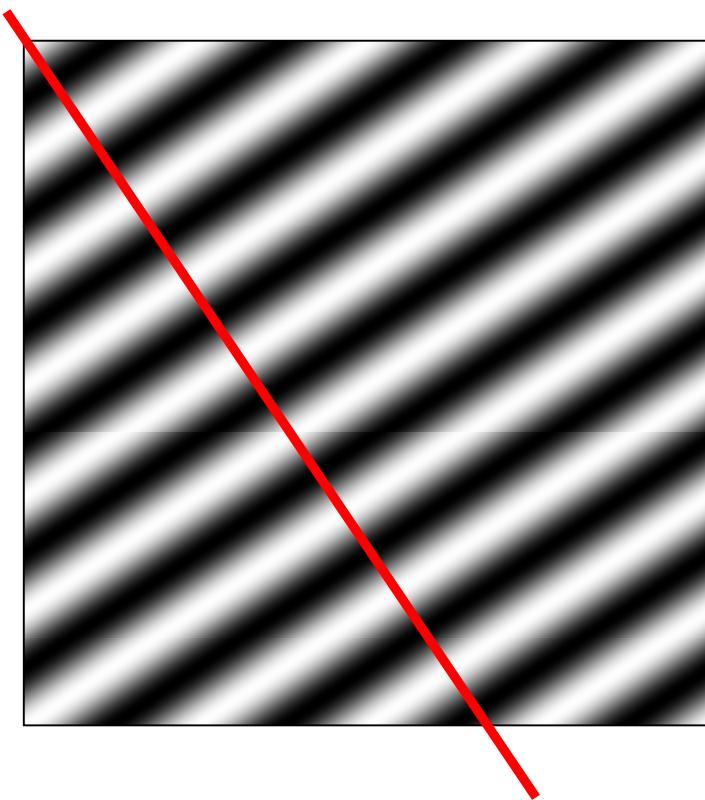


- If you take any vertical slice through the image, you get a 1D cosine with frequency $\nu=6$ cpi.



- If you take a slice through the image in a direction normal to the wavefronts, you get a 1D cosine with frequency

$$\sqrt{u^2 + v^2} = \sqrt{4^2 + 6^2} = 7.21 \text{ cpi}$$

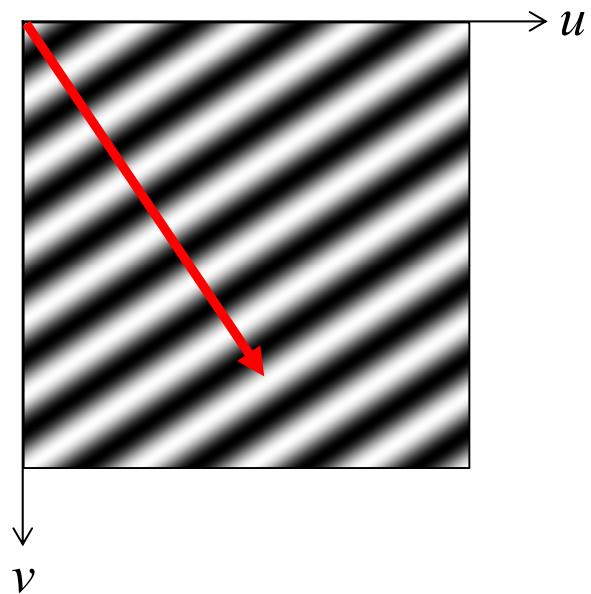


Note: the main diagonal of the image is longer than one “image.” It has a length of $\sqrt{1^2 + 1^2} = 1.41$ images. So along a line the length of the main diagonal, we expect to see about $1.41 \times 7.21 = 10.17$ cycles.

Interpreting the 2D Frequency

- The 2D frequency vector $[u \ v]^T$ has a direction $\theta = \tan^{-1}(v/u)$.
- This is usually called the **frequency orientation**.
- If you draw u and v axes that agree with the sense of the image coordinates, then the frequency vector points in a direction normal to the wavefronts; it points in the direction of “propagation.”
- The magnitude of the frequency vector is

$$\Omega = \sqrt{u^2 + v^2}$$

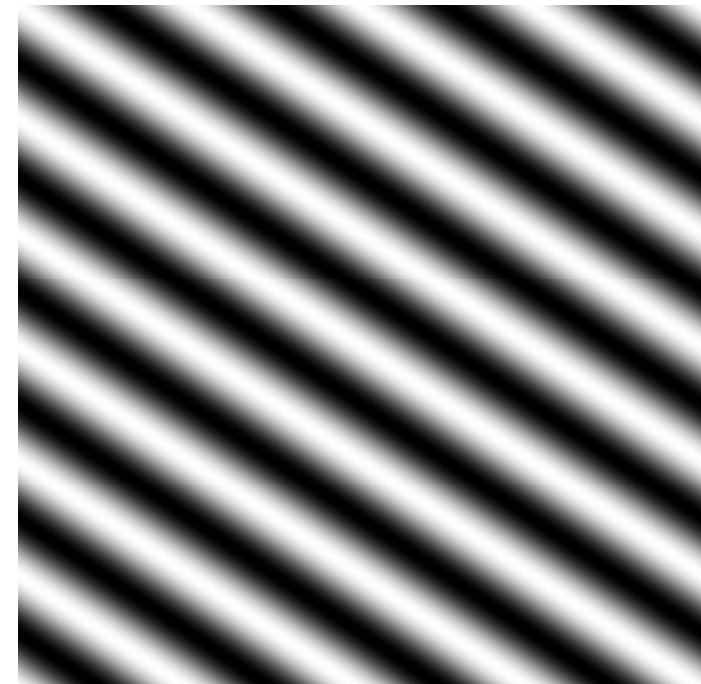


- This is called the **radial frequency** or **magnitude frequency**. It is also often denoted by R .
- In the direction θ , the frequency of oscillation is Ω cpi.

More on Cosine Images

- If $u = -4$ and $v = -6$, we get the same image over again since cosine is **even**.
- However, the relative signs of u and v matter.
- If $u = -4$ and $v = 6$, it changes the orientation and we get a different image.

$$u = -4; v = 6$$



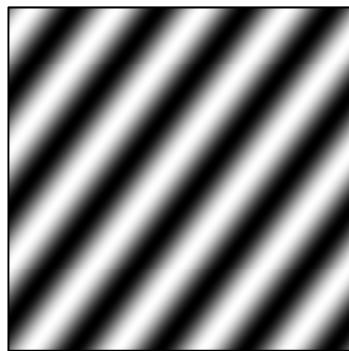
Sine Images

- Everything works pretty much the same way for sine images.
- Except: since sine is **odd**, if you negate both u and v it multiplies the image by -1.
- Also, for a sine image

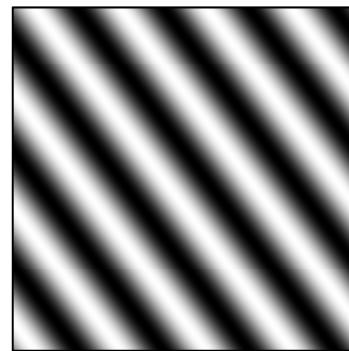
$$I(m, n) = \sin\left[2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right]$$

the pixel at $m=0, n=0$ is gray instead of white.

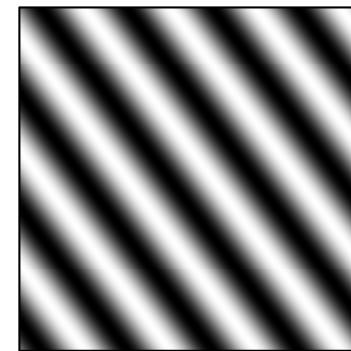
$u=4, v=3$



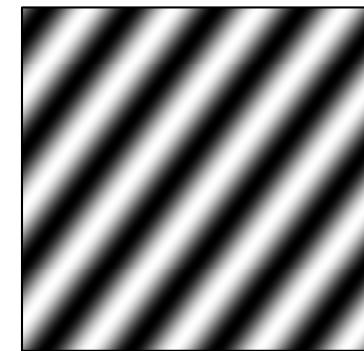
$u=4, v=-3$



$u=-4, v=3$



$u=-4, v=-3$

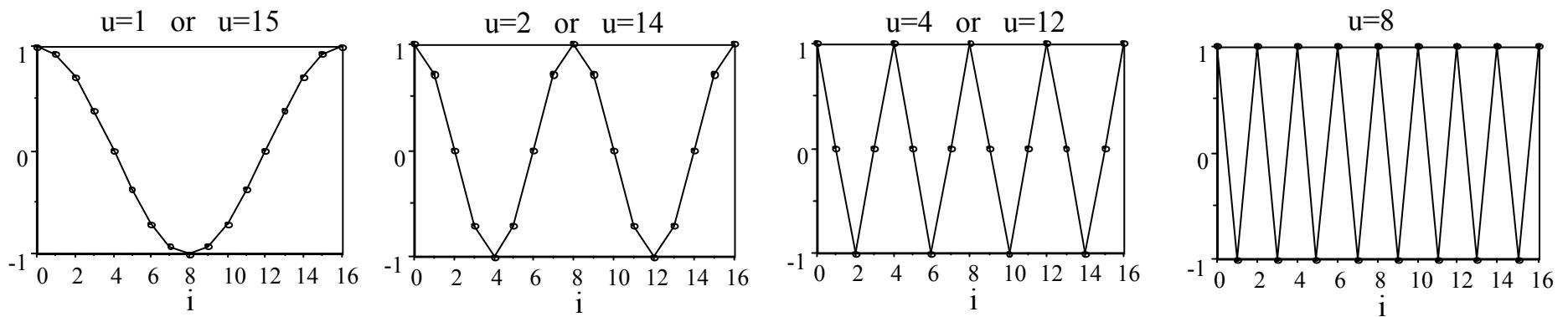




Spatial Frequencies

Digital Sinusoid Example

- Let $M = 16$, $v = 0$: $I(m) = \cos(2\pi um/16)$: a cosine wave oriented in m -direction with frequency u . One row:



- Note that $I(m) = \cos(2\pi um/16) = \cos[2\pi(16-u)m/16]$.
- Thus the **highest frequency wave occurs at $u = M/2$** (M is even here). **This will be important later.**

Explanation

- Why does $\cos(2\pi um/16) = \cos[2\pi(16-u)m/16]$?
- The reason is that adding any integer multiple of 2π to the argument of cos does not change the value.

$$\begin{aligned}\cos\left[\frac{2\pi(M-u)}{M}m\right] &= \cos\left[\frac{2\pi M}{M}m - \frac{2\pi u}{M}m\right] \\ &= \cos\left[-\frac{2\pi u}{M}m + 2\pi m\right] \\ &= \cos\left[-\frac{2\pi u}{M}m\right] = \cos\left[\frac{2\pi u}{M}m\right].\end{aligned}$$

Complex Exponential Image

- We'll use **complex exponential functions** to define the Discrete Fourier Transform.
- Define the **2-D complex exponential**:

$$\exp\left[-j2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] \text{ for } 0 \leq m \leq M-1, 0 \leq n \leq N-1$$

- Here, we put the “-” sign in front of the $j2\pi$ for compatibility with the W_N notation.

- Intuitively, the complex exponential image can be understood in terms of sine and cosine images, since

$$\begin{aligned}\exp\left[-j2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] &= \cos\left[2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] - j\sin\left[2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] \\ &= \cos\left[2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] + j\sin\left[2\pi\left(\frac{-u}{M}m + \frac{-v}{N}n\right)\right].\end{aligned}$$

- Also note that the complex exponential image is **separable**:

$$e^{-j2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)} = e^{-j2\pi\frac{u}{M}m} e^{-j2\pi\frac{v}{N}n}$$

Properties of Complex Exponential

- Recall from page 4.23:

$$W_N = \exp\left(-j \frac{2\pi}{N}\right)$$

(N = image dimension).

- Hence, for a square image with $M=N$,

$$\exp\left[-j2\pi\left(\frac{u}{N}m + \frac{v}{N}n\right)\right] = W_N^{(um + vn)} = W_N^{um} W_N^{vn}$$

- More generally, when the image is **not** square, we have

$$\begin{aligned}\exp\left[-j2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] &= \exp\left[-j2\pi\frac{u}{M}m\right]\exp\left[-j2\pi\frac{v}{N}n\right] \\ &= W_M^{um} W_N^{vn}\end{aligned}$$

- The magnitude and phase of the complex exponential image are

$$\left| \exp\left[-j2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] \right| = 1$$

$$\angle \exp\left[-j2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] = -2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)$$

Complex Exponential Image

- From Euler's identity:

$$W_M = \cos\left(\frac{2\pi}{M}\right) - j \sin\left(\frac{2\pi}{M}\right)$$

and

$$W_M^{um} = \cos\left(2\pi \frac{u}{M} m\right) - j \sin\left(2\pi \frac{u}{M} m\right)$$

- The u and v **powers** of W_M and W_N index the frequencies of the horizontal and vertical sinusoids.

Simple Properties of W_M and W_N

$$\cos\left[2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] = \frac{1}{2}\left(W_M^{um}W_N^{vn} + W_M^{-um}W_N^{-vn}\right)$$

$$\sin\left[2\pi\left(\frac{u}{M}m + \frac{v}{N}n\right)\right] = j\frac{1}{2}\left(W_M^{um}W_N^{vn} - W_M^{-um}W_N^{-vn}\right)$$

$$|W_M^{um}| = \left\{ \cos^2\left(2\pi\frac{u}{M}m\right) + \sin^2\left(2\pi\frac{u}{M}m\right) \right\} = 1$$

$$\angle W_M^{um} = \tan^{-1} \left\{ \sin\left(2\pi\frac{u}{M}m\right) / \cos\left(2\pi\frac{u}{M}m\right) \right\} = 2\pi\frac{u}{M}m$$

Comments

- It's possible to develop frequency domain concepts without complex numbers - but the math is much lengthier.
- Using $W_M^{um}W_N^{vn}$ to represent a **frequency component** oscillating horizontally at u cpi and vertically at v cpi simplifies things considerably.
- It is useful to think of $W_M^{um}W_N^{vn}$ as a **representation** of a **direction and frequency of oscillation**.

Max/Min Values for u and v

- The complex exponential

$$W_M^{um} = \exp\left(\frac{-j2\pi}{M} um\right)$$

is a sinusoid with frequency indexed by exponent u .

- **Minimum physical frequencies:** $u = kM, k \in \mathbb{Z}$

$$W_M^{0m} = W_M^{kMm} = 1 \quad \forall m, k \in \mathbb{Z}$$

- **Maximum physical frequencies:** $u = (k+1/2)M$ (period 2)

$$W_M^{(kM+M/2)m} = 1 \cdot W_M^{(M/2)m} = (-1)^m \quad (M \text{ even})$$

2D DISCRETE FOURIER TRANSFORM

- Any $M \times N$ image \mathbf{I} can be uniquely written as a **weighted sum** of MN **complex exponential** basis images:

$$I(m, n) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \tilde{I}(u, v) W_M^{-um} W_N^{-vn} \quad (\text{IDFT})$$

- This is called the **2D Inverse Discrete Fourier Transform** or **IDFT**.
- The integers $0 \leq u \leq M-1$ and $0 \leq v \leq N-1$ index the basis images and give their frequencies in cpi.
- The weights \tilde{I} are the **2D DFT coefficients** of I .

Forward Transform (DFT)

- The forward transform is given by the dot product of the image \mathbf{I} with a basis vector:

$$\tilde{\mathbf{I}}(u, v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \mathbf{I}(m, n) W_M^{um} W_N^{vn} \quad (\text{DFT})$$

- Remember that (m, n) are **space indices**, while (u, v) are **spatial frequency indices**.

2D DFT as an Array

- The 2D DFT is an array with the **same dimensions** ($M \times N$) as the image \mathbf{I} :

$$\tilde{\mathbf{I}} = [\tilde{\mathbf{I}}(u, v); 0 \leq u \leq M - 1, 0 \leq v \leq N - 1]$$

- You can think of it as an image (complex-valued).
- Note that the DFT is a **linear transformation**, so it commutes with linear combinations:

$$\text{DFT}[a_1\mathbf{I}_1 + a_2\mathbf{I}_2 + \cdots + a_L\mathbf{I}_L] = a_1\tilde{\mathbf{I}}_1 + a_2\tilde{\mathbf{I}}_2 + \cdots + a_L\tilde{\mathbf{I}}_L$$

DFT Array Properties

- The DFT of an image \mathbf{I} is generally **complex**:

$$\tilde{\mathbf{I}} = \tilde{\mathbf{I}}_{\text{Real}} + j \tilde{\mathbf{I}}_{\text{Imag}}$$

where

$$\tilde{\mathbf{I}}_{\text{Real}}(u, v) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m, n) \cos \left[2\pi \left(\frac{u}{M} m + \frac{v}{N} n \right) \right]$$

$$\tilde{\mathbf{I}}_{\text{Imag}}(u, v) = - \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m, n) \sin \left[2\pi \left(\frac{u}{M} m + \frac{v}{N} n \right) \right]$$

Magnitude and Phase of the DFT

- It is usually more intuitive to look at the **magnitude** and **phase** of the DFT rather than the real and imaginary parts:

$$|\tilde{I}| = \left[|\tilde{I}(u, v)|; 0 \leq u \leq M - 1, 0 \leq v \leq N - 1 \right]$$

$$\angle \tilde{I} = \left[\angle \tilde{I}(u, v); 0 \leq u \leq M - 1, 0 \leq v \leq N - 1 \right]$$

where

$$|\tilde{I}(u, v)| = \sqrt{\tilde{I}_{\text{Real}}^2(u, v) + \tilde{I}_{\text{Imag}}^2(u, v)}$$

$$\angle \tilde{I}(u, v) = \tan^{-1} \left[\tilde{I}_{\text{Imag}}(u, v) / \tilde{I}_{\text{Real}}(u, v) \right]$$

- In polar form, the 2D DFT is

$$\tilde{I}(u, v) = |\tilde{I}(u, v)| \exp(j \angle \tilde{I}(u, v))$$

Symmetry of the DFT

- For a real image \mathbf{I} , the DFT is **conjugate symmetric**:

$$\tilde{\mathbf{I}}(M-u, N-v) = \tilde{\mathbf{I}}^*(u, v) ; \quad 0 \leq u \leq M-1, \quad 0 \leq v \leq N-1$$

Proof:

$$\begin{aligned}\tilde{\mathbf{I}}(M-u, N-v) &= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \mathbf{I}(m, n) W_M^{(M-u)m} W_N^{(N-v)n} \\ &= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \mathbf{I}(m, n) W_M^{Mm} W_M^{-um} W_N^{Nn} W_N^{-vn} \\ &= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \mathbf{I}(m, n) \left[W_M^{um} W_N^{vn} \right]^* = \mathbf{I}^*(u, v)\end{aligned}$$

More Symmetry Properties

- The symmetry of the DFT array for a real \mathbf{I} implies that it is **redundant**. If you know half of the coefficients, you can figure out the rest from symmetry.
- We also have

$$\tilde{I}_{\text{Real}}(M-u, N-v) = \tilde{I}_{\text{Real}}(u, v)$$

$$\tilde{I}_{\text{Imag}}(M-u, N-v) = -\tilde{I}_{\text{Imag}}(u, v)$$

and

$$|\tilde{I}(M-u, N-v)| = |\tilde{I}(u, v)|$$

$$\angle \tilde{I}(M-u, N-v) = -\angle \tilde{I}(u, v)$$

for $0 \leq u \leq M-1$, $0 \leq v \leq N-1$

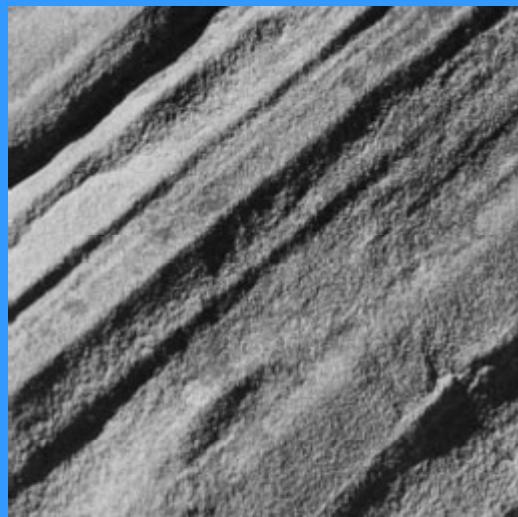
Viewing the DFT

- To view the DFT, we usually display the **magnitude and phase as images**.
- In most cases, the phase is very difficult to interpret visually.
- Hence, it is common to look at the magnitude **only**.
- Logarithmic compression and a full-scale stretch are almost always applied to the magnitude for display:

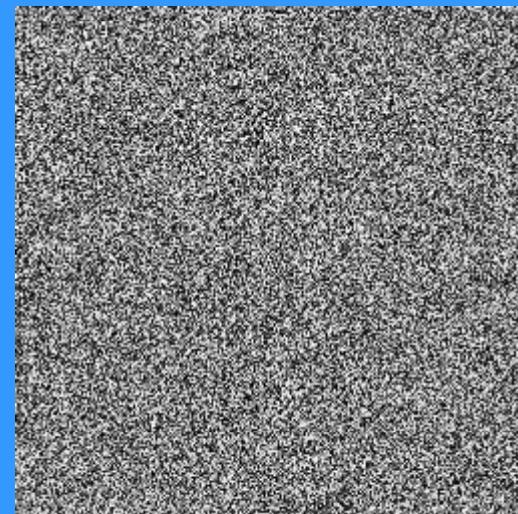
$$\log \left[1 + |\tilde{I}(u, v)| \right]$$

- This is the **log-magnitude spectrum** of the image I .

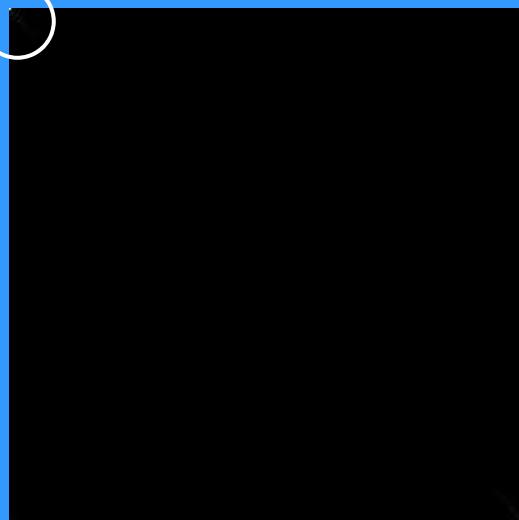
I



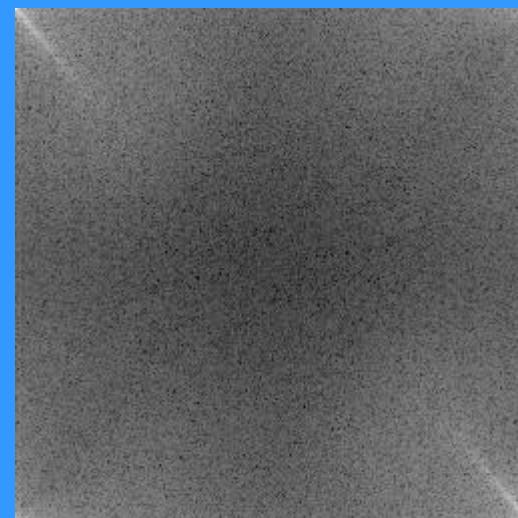
$\angle \tilde{I}$



DC



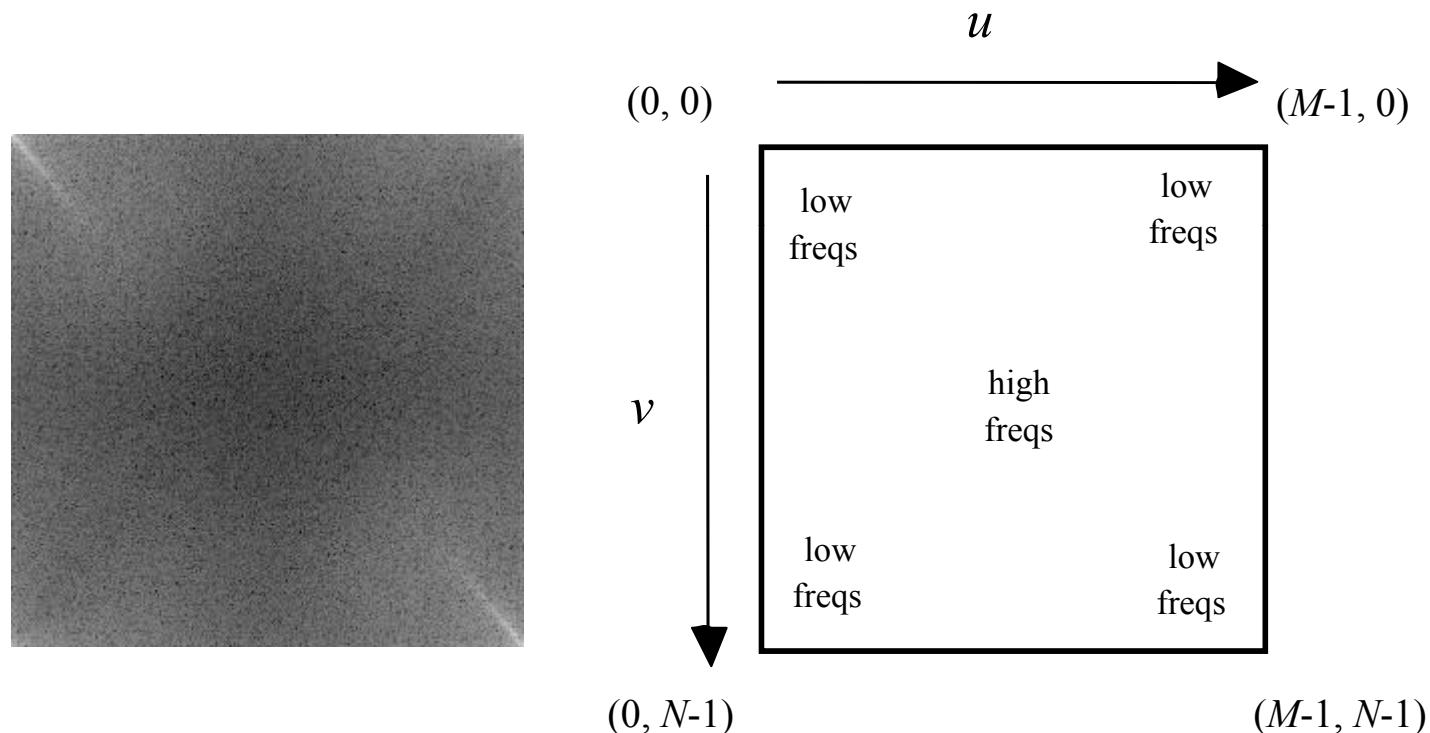
$|\tilde{I}|$



$\log(1 + |\tilde{I}|)$

62

- Note that the coefficients of the highest physical frequencies are **located near the center** of the DFT array: near $(u, v) = (M/2, N/2)$.



Periodicity of the DFT

- In the computer, the DFT coefficients are an array with a **finite** size the same as the image ($M \times N$):

$$\tilde{\mathbf{I}} = [\tilde{I}(u, v)] \quad 0 \leq u \leq M-1, 0 \leq v \leq N-1$$

- However, for values of u, v **outside** this range, the mathematics of the DFT is **periodic**.
- Conceptually, the DFT coefficient array is thus **infinite in size and periodic** with

$$\tilde{I}(u + pM, v + qN) = \tilde{I}(u, v) \quad \forall p, q \in \mathbb{Z}.$$

- Mathematically, the image \mathbf{I} can be recovered by applying the IDFT equation to **any one period** of $\tilde{\mathbf{I}}$.

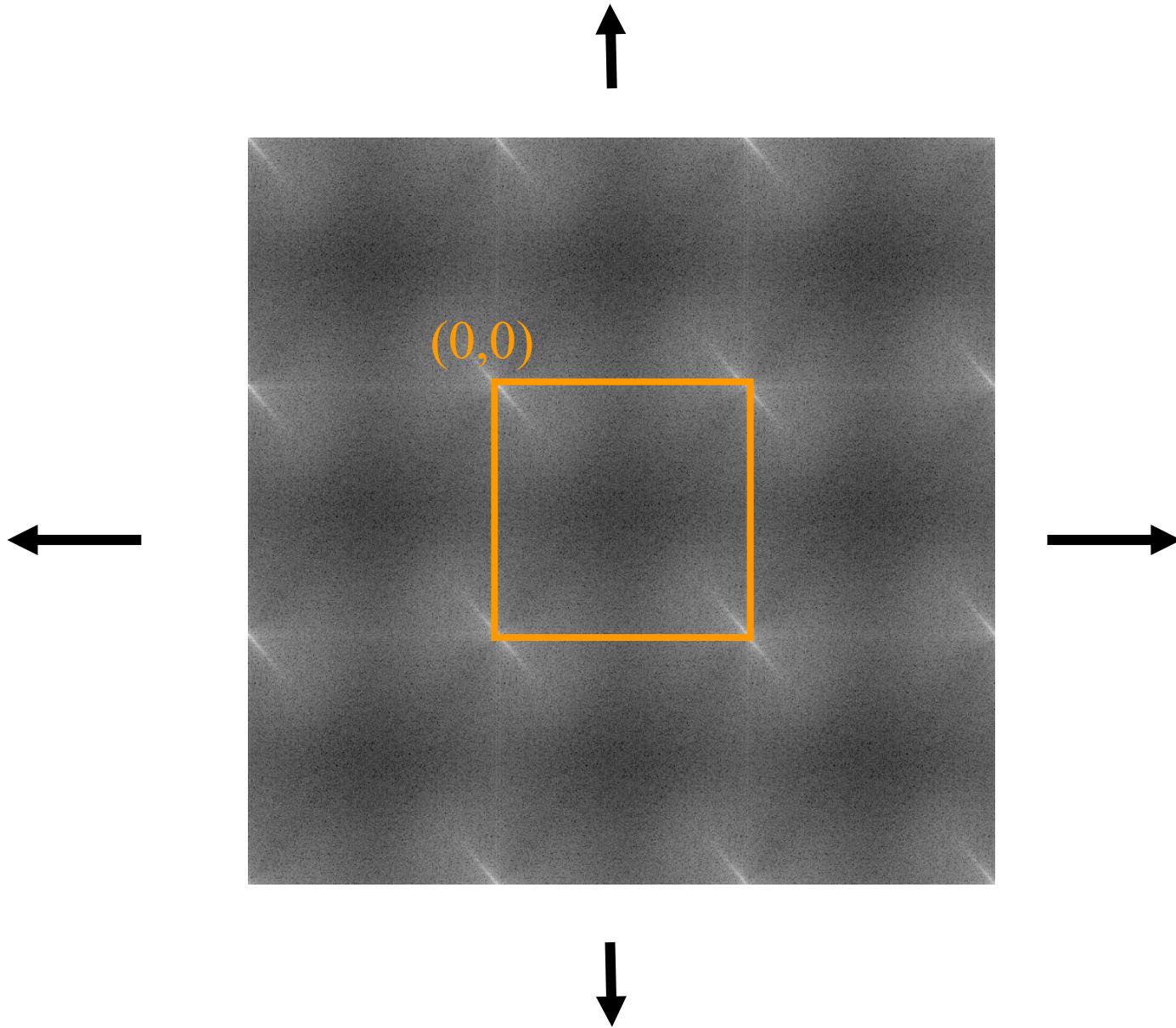
Proof of DFT Periodicity

- Let u, v be in the range $0 \leq u \leq M-1$ and $0 \leq v \leq N-1$.
- Let p, q be any integers. Then

$$\begin{aligned}\tilde{I}(u + pM, v + qN) &= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m, n) W_M^{(u+pM)m} W_N^{(v+qN)n} \\ &= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m, n) W_M^{um} W_N^{vn} W_M^{pMm} W_N^{qNn} \\ &= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m, n) W_M^{um} W_N^{vn} = \tilde{I}(u, v)\end{aligned}$$

This follows because $W_M^{qMm} = e^{-j2\pi \frac{pMm}{M}} = e^{-j2\pi(\text{integer})} = 1.$ 65

Periodically extended DFT array



Periodic Extension of Image

- Mathematically, the DFT basis functions W_M^{-um} W_N^{-vn} are defined for all integers m, n . Conceptually, they are **infinite in size and periodic**.
- When we write \mathbf{I} as a sum of these basis functions

$$\mathbf{I}(m, n) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \tilde{\mathbf{I}}(u, v) W_M^{-um} W_N^{-vn},$$

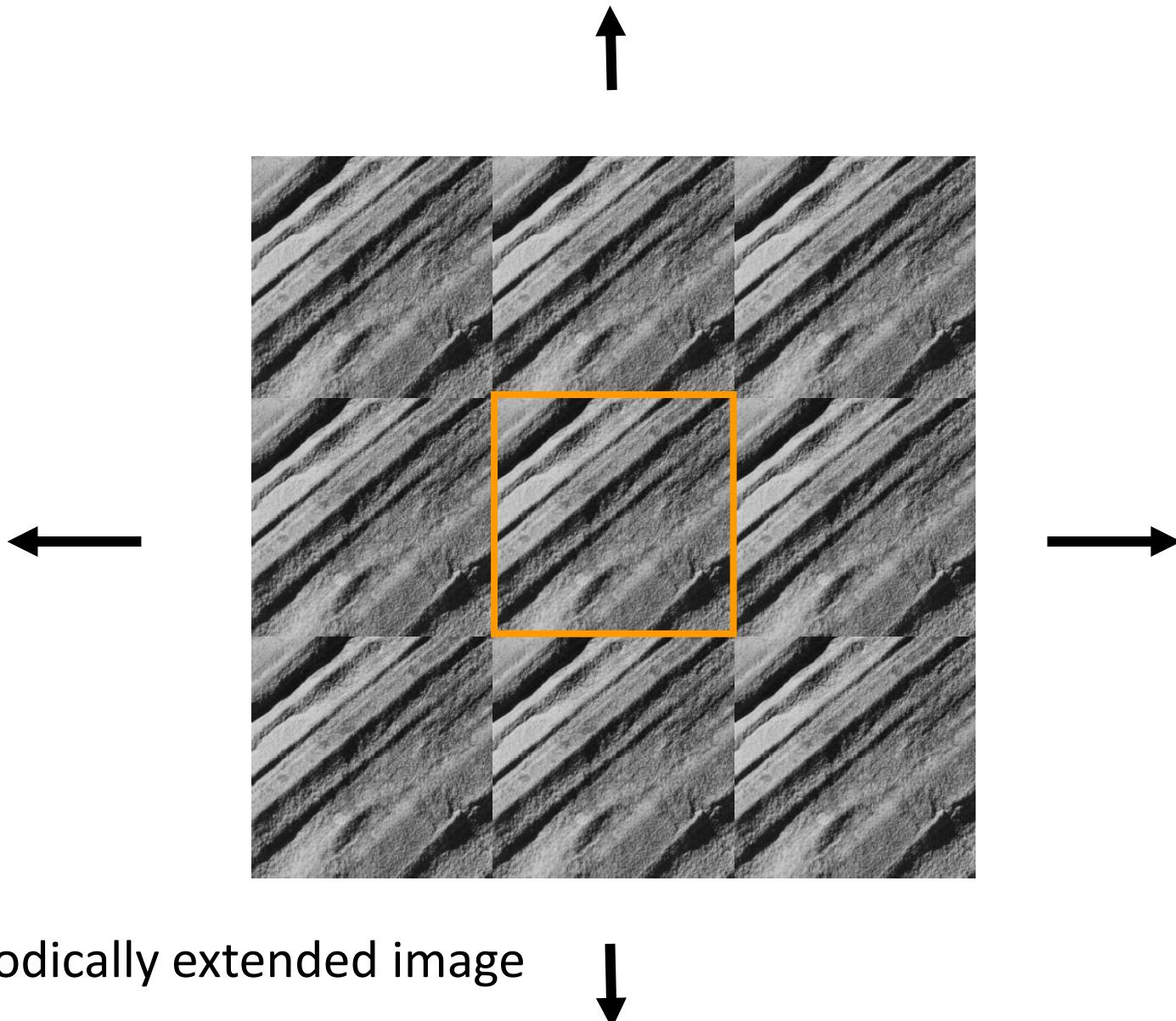
it defines an image that is also **infinite in size and periodic**.

- The DFT does not understand the concept of a finite-sized image.
- To the mathematics of the DFT, **every** image is **infinite in size and periodic** and **every** image has a DFT coefficient array that is also **infinite in size and periodic**.

Proof that the Image is Periodic

- Let m, n be in the range $0 \leq m \leq M-1$ and $0 \leq n \leq N-1$.
- Let p, q be any integers. Then

$$\begin{aligned} I(m + pM, n + qM) &= \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \tilde{I}(u, v) W_M^{-u(m+pM)} W_N^{-v(n+qN)} \\ &= \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \tilde{I}(u, v) W_M^{-um} W_N^{-vn} W_M^{-upM} W_N^{-vqN} \\ &= \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \tilde{I}(u, v) W_M^{-Mm} W_N^{-Nn} = I(m, n) \end{aligned}$$



Displaying the Centered DFT

- For interpretation, it's usually desirable to display the log-magnitude spectrum with all of the low frequencies together in the center (instead of at the corners – see page 4.63).
- This can be accomplished using the frequency shift property of the DFT if we multiply the image \mathbf{I} times an alternating image at the Nyquist rate:

$$(-1)^{m+n} \mathbf{I}(m, n)$$

- Taking the DFT of this modified image then produces the **centered log-magnitude spectrum**.
- For images, this is how the DFT is normally viewed.

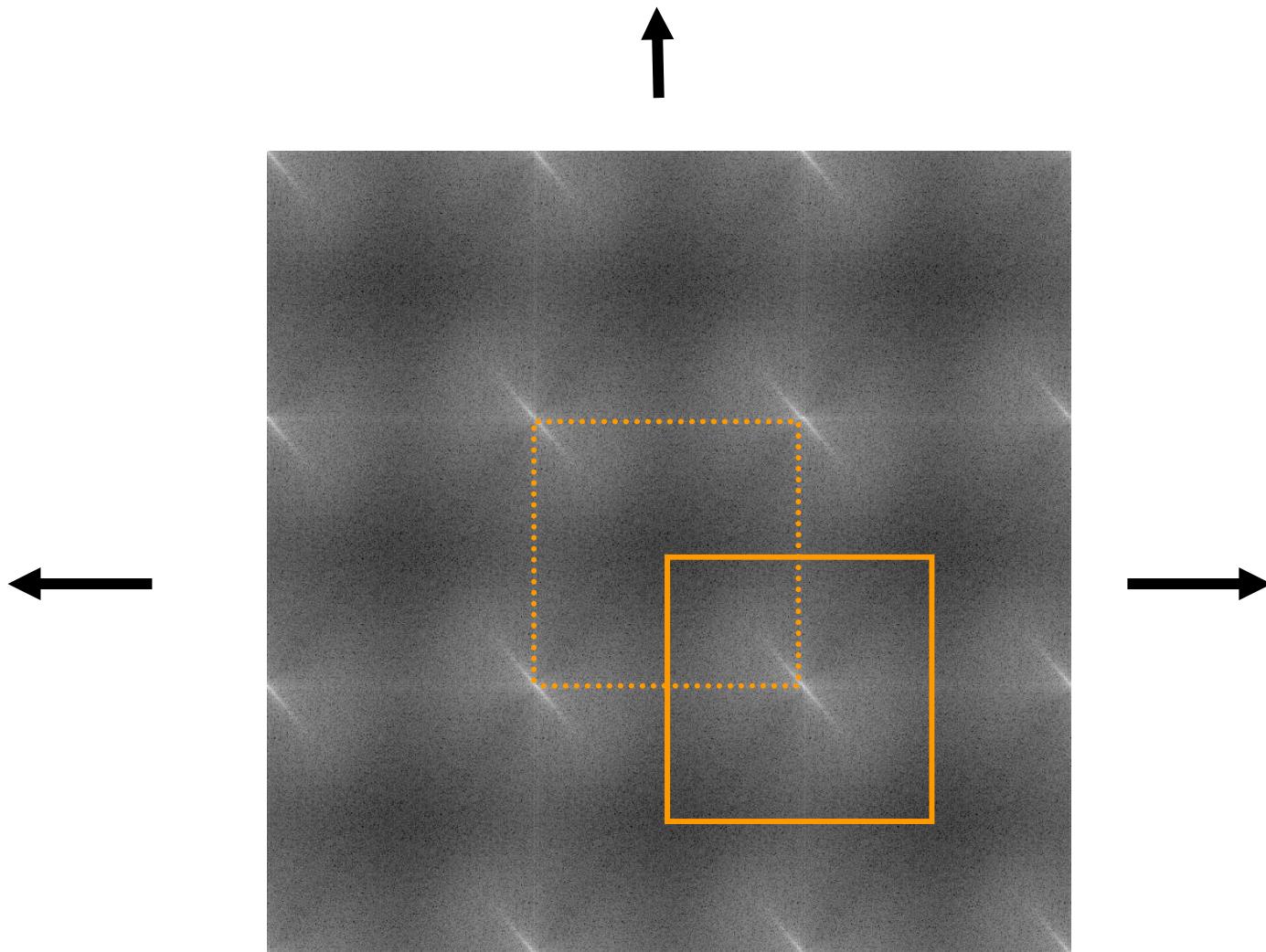
Centering the DFT

- How it works: note that

$$(-1)^{m+n} = (-1)^m (-1)^n = W_M^{-mM/2} W_N^{-nN/2},$$

so

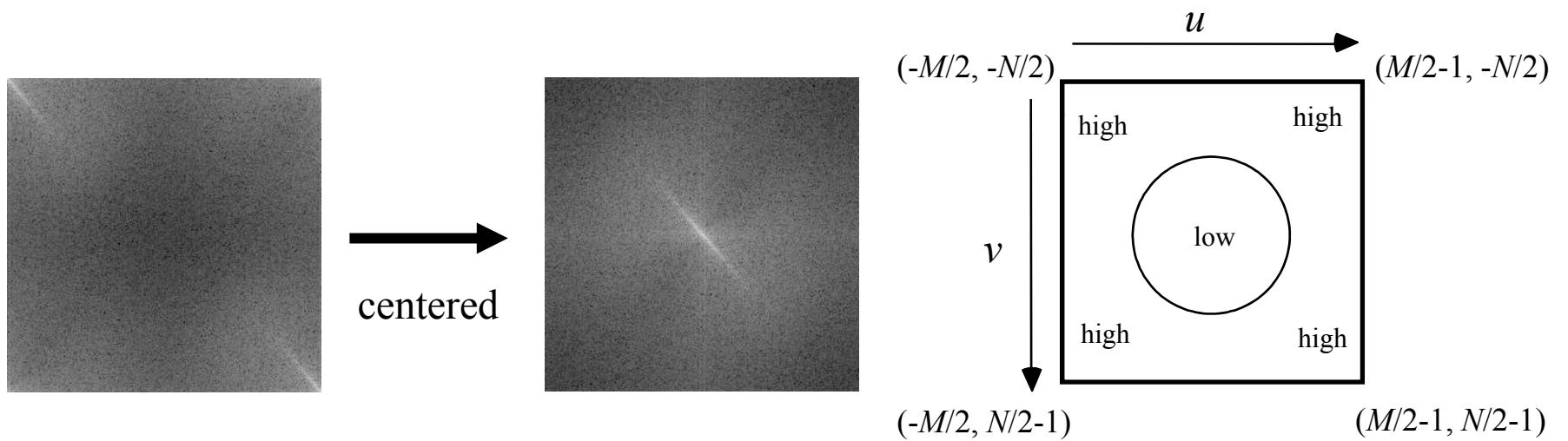
$$\begin{aligned}\text{DFT}\left[(-1)^{m+n} \mathbf{I}(m, n)\right] &= \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \mathbf{I}(m, n) (-1)^{m+n} W_M^{um} W_N^{vn} \\ &= \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \mathbf{I}(m, n) W_M^{um} W_N^{vn} W_M^{-(M/2)m} W_N^{-(N/2)n} \\ &= \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \mathbf{I}(m, n) W_M^{[u-(M/2)]m} W_N^{[v-(N/2)]n} \\ &= \tilde{\mathbf{I}}[u - (M/2), v - (N/2)]\end{aligned}$$



Shifted (centered) DFT
from periodic extension



Centered DFT



Understanding the Centered DFT

III	$u=-4$	$u=-3$	$u=-2$	$u=-1$	$u=0$	$u=1$	$u=2$	$u=3$	IV
$v=-4$	i=0, j=0 k=0	i=1, j=0 k=1	i=2, j=0 k=2	i=3, j=0 k=3	i=4, j=0 k=4	i=5, j=0 k=5	i=6, j=0 k=6	i=7, j=0 k=7	$v=-4$
$v=-3$	i=0, j=1 k=8	i=1, j=1 k=9	i=2, j=1 k=10	i=3, j=1 k=11	i=4, j=1 k=12	i=5, j=1 k=13	i=6, j=1 k=14	i=7, j=1 k=15	$v=-3$
$v=-2$	i=0, j=2 k=16	i=1, j=2 k=17	i=2, j=2 k=18	i=3, j=2 k=19	i=4, j=2 k=20	i=5, j=2 k=21	i=6, j=2 k=22	i=7, j=2 k=23	$v=-2$
$v=-1$	i=0, j=3 k=24	i=1, j=3 k=25	i=2, j=3 k=26	i=3, j=3 k=27	i=4, j=3 k=28	i=5, j=3 k=29	i=6, j=3 k=30	i=7, j=3 k=31	$v=-1$
<hr/>									
$v=0$	i=0, j=4 k=32	i=1, j=4 k=33	i=2, j=4 k=34	i=3, j=4 k=35	i=4, j=4 k=36	i=5, j=4 k=37	i=6, j=4 k=38	i=7, j=4 k=39	$v=0$
$v=1$	i=0, j=5 k=40	i=1, j=5 k=41	i=2, j=5 k=42	i=3, j=5 k=43	i=4, j=5 k=44	i=5, j=5 k=45	i=6, j=5 k=46	i=7, j=5 k=47	$v=1$
$v=2$	i=0, j=6 k=48	i=1, j=6 k=49	i=2, j=6 k=50	i=3, j=6 k=51	i=4, j=6 k=52	i=5, j=6 k=53	i=6, j=6 k=54	i=7, j=6 k=55	$v=2$
$v=3$	i=0, j=7 k=56	i=1, j=7 k=57	i=2, j=7 k=58	i=3, j=7 k=59	i=4, j=7 k=60	i=5, j=7 k=61	i=6, j=7 k=62	i=7, j=7 k=63	$v=3$
II	$u=-4$	$u=-3$	$u=-2$	$u=-1$	$u=0$	$u=1$	$u=2$	$u=3$	I

- Suppose the image \mathbf{I} is 8×8 .
- This picture shows the frequencies u, v for every coefficient in the centered 2D DFT array.
- i, j are the col/row indices for a C array. For Matlab, add 1.
- k is the offset from the base of the array in memory. For a 1D C array, it is the index. For a 1D Matlab array, add 1.
- We think of the 2D frequency plane as being divided into four quadrants:
 - Quadrant I: $u \geq 0, v \geq 0$
 - Quadrant II: $u < 0, v \geq 0$
 - Quadrant III: $u < 0, v < 0$
 - Quadrant IV: $u \geq 0, v < 0$

Understanding the Centered DFT

- Matlab provides builtin functions `fft2` and `ifft2` to compute the 2D DFT equations on pages 4.54-4.55.
- Matlab also provides a builtin function `fftshift` to center the DFT as shown on pages 4.70-4.73.
 - If the length of the DFT array is even, then applying `fftshift` a second time will undo the shift.
 - Matlab also provides a function `ifftshift` to undo the shift. If the length of the DFT array is odd, then you *must* use `ifftshift`.
- The Matlab FFT implementation calls an open source package named FFTW3.
- If you use another language, you should also install FFTW3 and call it. There is a link to it on the course web site.

Example

- Let \mathbf{X}_1 be a cosine image with $M=N=64$ and $u=v=8$:

$$\mathbf{X}_1(m, n) = \cos\left[\frac{2\pi}{64}(8m + 8n)\right] = \frac{1}{2} W_{64}^{-(8m - 8n)} + \frac{1}{2} W_{64}^{-(8m + 8n)}$$

- Note that $MN=4096$.
- Following the pattern on page 4.74, the DC coefficient of the DFT array ($u=v=0$) will be at (row,col) = (33,33) in a Matlab array or (32,32) in a C array.
- The real part of the DFT array $\tilde{\mathbf{X}}_1$ will be equal to $4096/2 = 2048$ at $(u,v) = (-8,-8)$ and $(8,8)$.
 - This is (row,col) = (33-8, 33-8) = (25,25) and (33+8, 33+8) = (41,41) in a Matlab array.
 - Or (row,col) = (24,24) and (40,40) in a C array.
 - Everywhere else, the real part of the DFT array will be zero.
- Because cosine is **real and even**, the imaginary part of the DFT array is all zero.

Example

- Let \mathbf{X}_2 be a sine image with $M=N=64$, $u = -4$, and $v = 7$:

$$\mathbf{X}_2(m, n) = \sin\left[\frac{2\pi}{64}(-4m + 7n)\right] = \frac{-j}{2} W_{64}^{-(4m + 7n)} + \frac{j}{2} W_{64}^{-(4m - 7n)}$$

- We have again that $MN=4096$.
- Because sine is **pure imaginary and odd**, the real part of the DFT array is all zero.
- The imaginary part of the DFT array $\tilde{\mathbf{X}}_2$ will be equal to $-4096/2 = -2048$ at $(u,v) = (-4,7)$.
 - This is $(\text{row},\text{col}) = (33+7, 33-4) = (40,29)$ in a Matlab array.
 - Or $(\text{row},\text{col}) = (39,28)$ in a C array.
- The imaginary part will be $4096/2 = 2048$ at $(u,v) = (4,-7)$.
 - This is $(\text{row},\text{col}) = (33-7, 33+4) = (26,37)$ in a Matlab array.
 - Or $(\text{row},\text{col}) = (25,36)$ in a C array.
- Everywhere else, the imaginary part will be zero.

Example

- Let $\mathbf{X}_3 = \mathbf{X}_1 + \mathbf{X}_2$.
- Then $\tilde{\mathbf{X}}_3 = \tilde{\mathbf{X}}_1 + \tilde{\mathbf{X}}_2$.
- Because $\tilde{\mathbf{X}}_1$ is real and $\tilde{\mathbf{X}}_2$ is pure imaginary, we have

$$\operatorname{Re}\{\tilde{\mathbf{X}}_3\} = \tilde{\mathbf{X}}_1 \quad j \operatorname{Im}\{\tilde{\mathbf{X}}_3\} = \tilde{\mathbf{X}}_2$$

$$\tilde{\mathbf{X}}_3 = \operatorname{Re}\{\tilde{\mathbf{X}}_3\} + j \operatorname{Im}\{\tilde{\mathbf{X}}_3\} = \tilde{\mathbf{X}}_1 + \tilde{\mathbf{X}}_2$$

- In simple words:
 - The real part of $\tilde{\mathbf{X}}_3$ is equal to $\tilde{\mathbf{X}}_1$.
 - The imaginary part of $\tilde{\mathbf{X}}_3$ (times j) is equal to $\tilde{\mathbf{X}}_2$.
- Matlab code to implement this example:

```
[COLS,ROWS] = meshgrid(0:63,0:63);  
X1 = cos(2*pi/64 * (8*COLS + 8*ROWS));  
X2 = sin(2*pi/64 * (-4*COLS + 7*ROWS));  
X3 = X1 + X2;  
X3tilde = fftshift(fft2(X3));
```

```

>>real(X3tilde(25,25))

ans =
2.0480e+03

>>real(X3tilde(41,41))

ans =
2.0480e+03

>>imag(X3tilde(26,37))

ans =
2048

>>imag(X3tilde(40,29))

ans =
-2048

```

```

>> imshow(255*((X1+1)/2), [0 255])
>> imshow(255*((X2+1)/2), [0 255])
>> imshow(255*((X3+2)/4), [0 255])

```



```

>> imshow(real(X3tilde), [0 2048])
>> imshow(imag(X3tilde), [-2048 2048])

```



Example in C

- On the course web site, there is a program “DoFFT.c” that implements this example in C.
- You can find it under the “code” link on the web site.
- It contains a function fft2d that calls FFTW3 to compute the 2D DFT and then centers it.
 - Although only the forward transform is used in this example, fft2d can do both forward and inverse transforms.
 - There is a parameter “direction.” If direction=1, then it does the forward transform. If direction=-1, then it does the inverse transform.
- The comment block at the top of the program explains how to compile it and link it to the FFTW3 library.
- You need to install FFTW3 **before** you compile the program.

Complex Exponential Matrix

- We define a special matrix of complex numbers:

$$\mathbf{W}_N = \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & W_N^2 & W_N^3 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & W_N^4 & W_N^6 & \cdots & W_N^{2(N-1)} \\ 1 & W_N^3 & W_N^6 & W_N^9 & \cdots & W_N^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & W_N^{3(N-1)} & \cdots & W_N^{(N-1)^2} \end{bmatrix}$$

- Note that it is a **symmetric** matrix

Inverse Matrix

$$\mathbf{W}_N^{-1} = \frac{1}{N} \mathbf{W}_N^* = \frac{1}{N} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^{-1} & W_N^{-2} & W_N^{-3} & \cdots & W_N^{1-N} \\ 1 & W_N^{-2} & W_N^{-4} & W_N^{-6} & \cdots & W_N^{-2(N-1)} \\ 1 & W_N^{-3} & W_N^{-6} & W_N^{-9} & \cdots & W_N^{-3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{1-N} & W_N^{-2(N-1)} & W_N^{-3(N-1)} & \cdots & W_N^{-(N-1)^2} \end{bmatrix}$$

Matrix-Product Form of DFT

- For a square image \mathbf{I} , we can then concisely write the DFT and IDFT as **matrix products**:

$$\tilde{\mathbf{I}} = \mathbf{W}_N \mathbf{I} \mathbf{W}_N \quad (\text{DFT})$$

$$\mathbf{I} = \frac{1}{N^2} \mathbf{W}_N^* \tilde{\mathbf{I}} \mathbf{W}_N^* \quad (\text{IDFT})$$

- This form is often useful for analysis and proofs and for programming a computer implementation.

Computation of the DFT

- Fast DFT/IDFT algorithms are collectively referred to as the **Fast Fourier Transform** (FFT).
- Matlab provides routines `fft2`, `ifft2`, and `fftshift` (for centering).
- For C, C++, and other languages, the FFTW package is strongly recommended.
- The speedup is the same as in 1D: from $\mathcal{O}(M^2N^2)$ to $\mathcal{O}[MN\log(MN)]$.

Interpreting Spatial Frequencies in the Image

- It's easy to lose track of the meaning of the DFT and the notion of **frequency content** in all the math.
- By examining the DFT or **spectrum** of an image (especially its magnitude), we can often deduce much about the image.

Qualitative Properties of the DFT

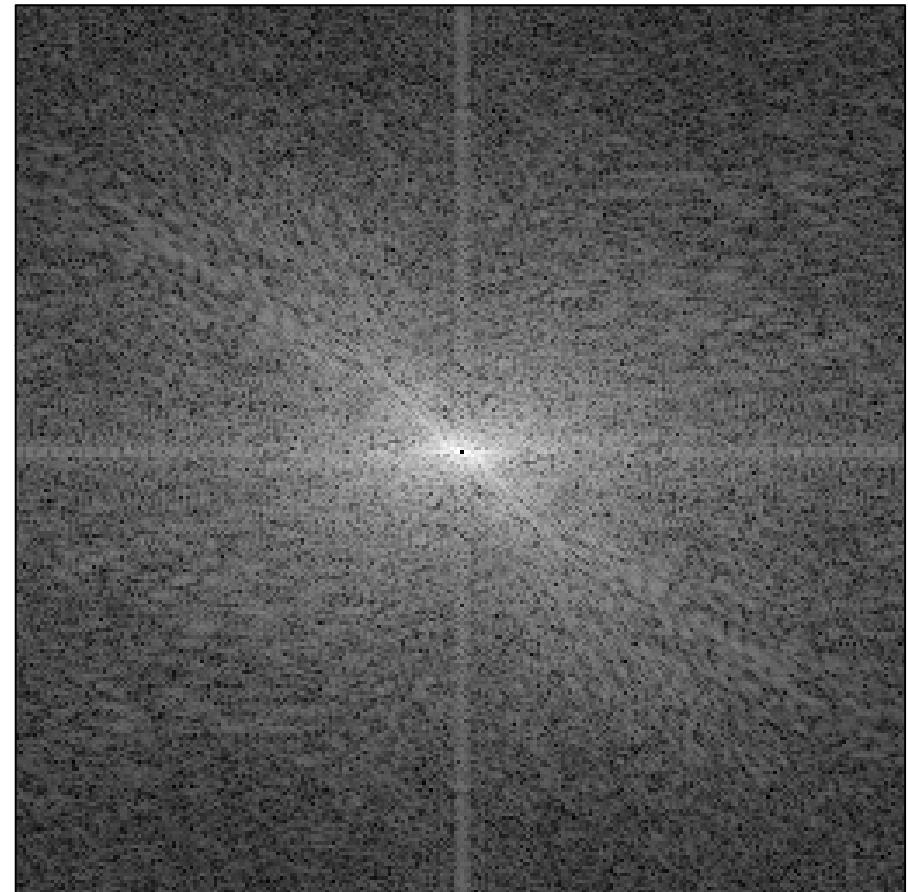
- We may regard the DFT magnitude as an **image of frequency content**.
- Bright regions in the DFT magnitude "image" correspond to frequencies having **large magnitudes** in the actual image.
- It is intuitive to think of image frequency content in terms of **granularity** (distribution of radial frequencies) and **orientation** (distribution of frequency vector directions).

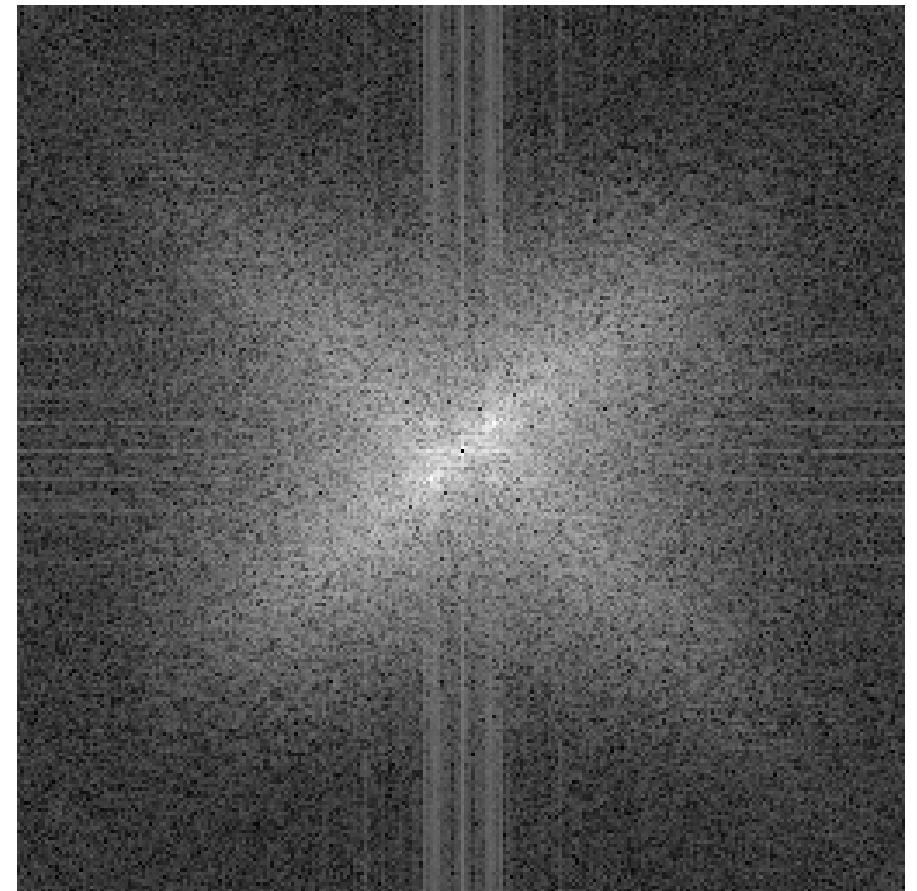
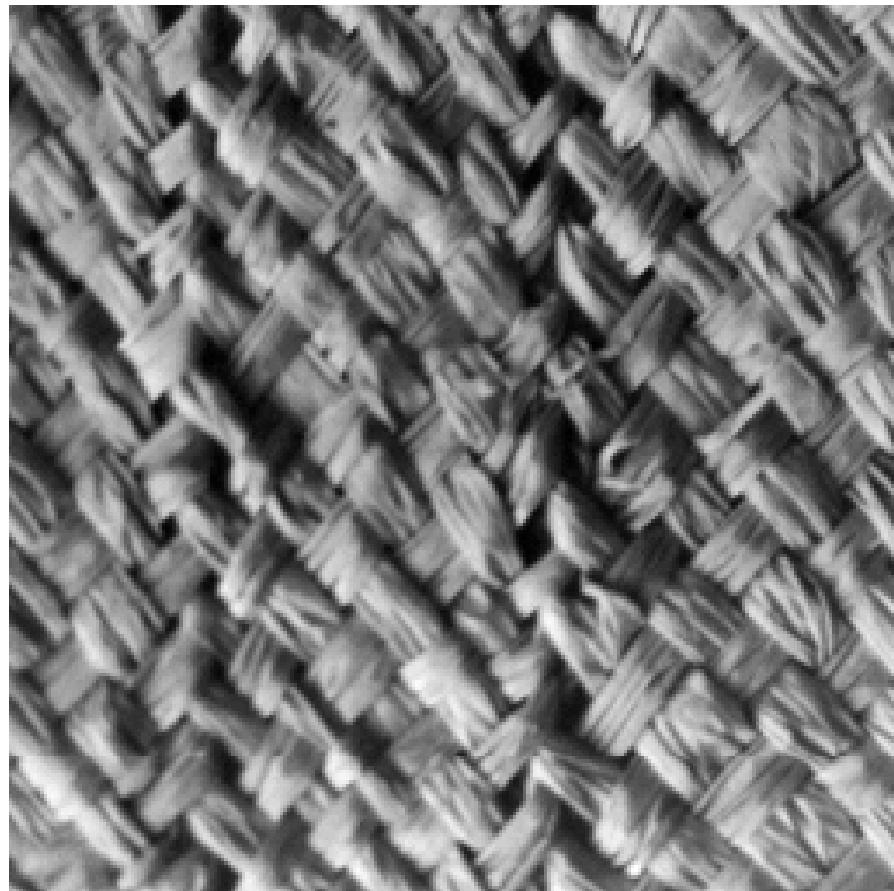
Image Granularity

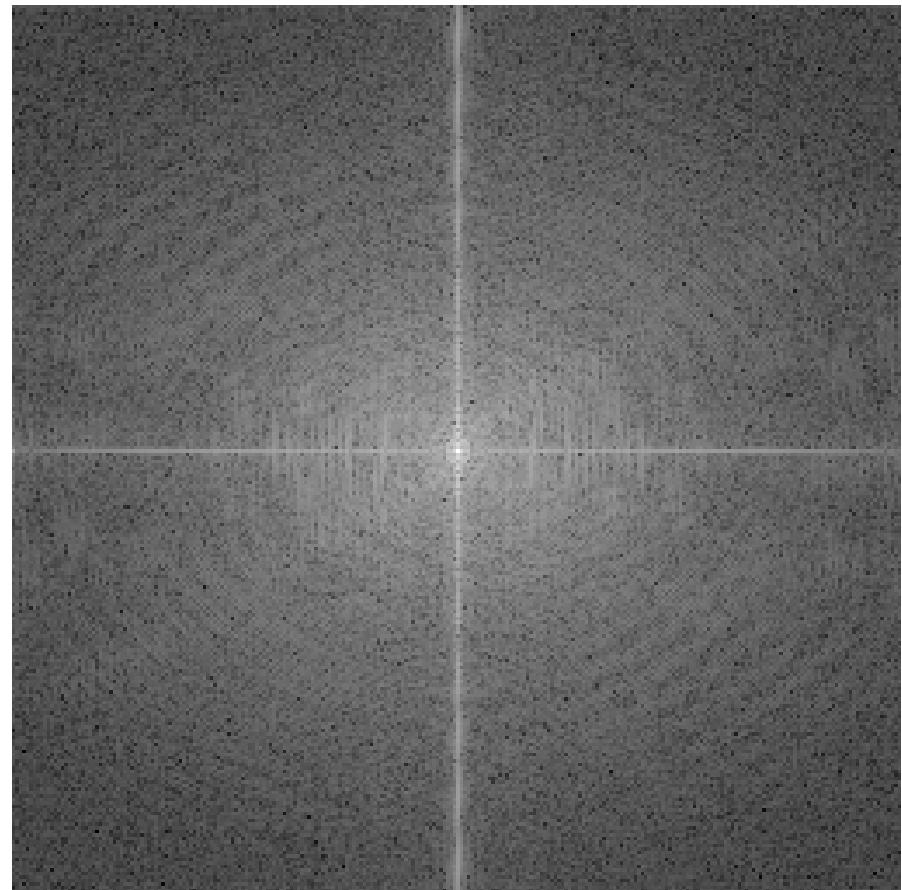
- DFT coefficients near the origin correspond to large, **smooth structure in the image** - often background.
- Since most images are non-negative, the DFT usually shows a strong peak at $(u, v) = (0, 0)$.
- Low frequency = long wavelength = large-scale structure in the image.
- High frequency = short wavelength = small-scale structure in the image.

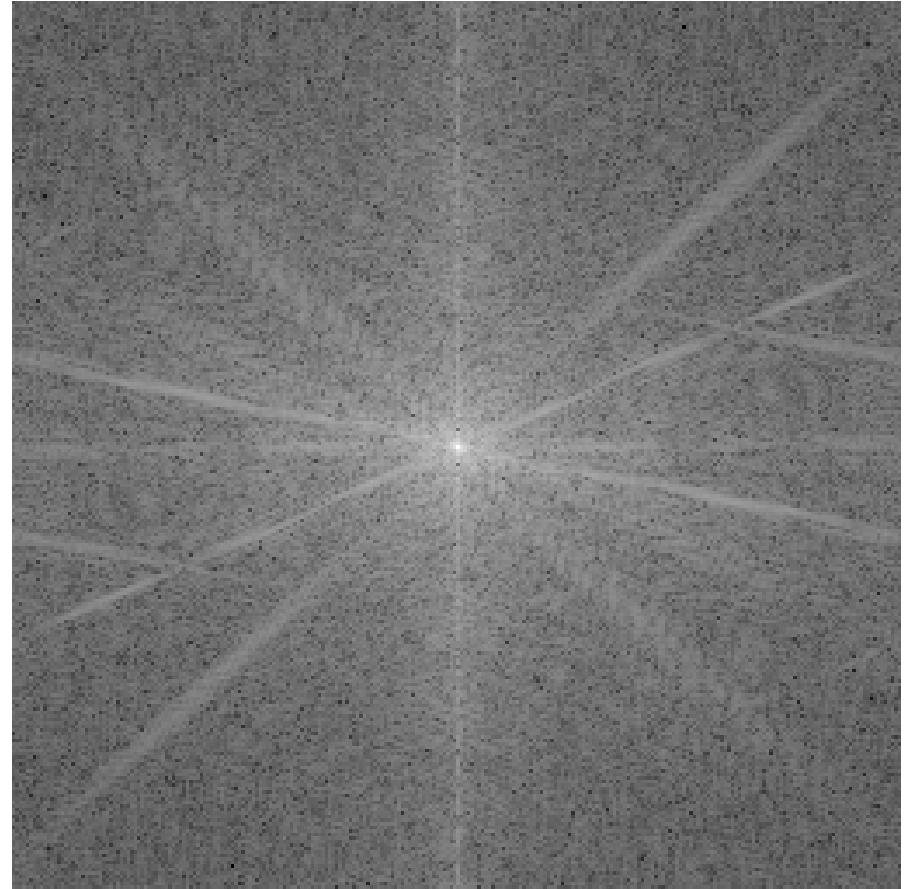
Image Directionality

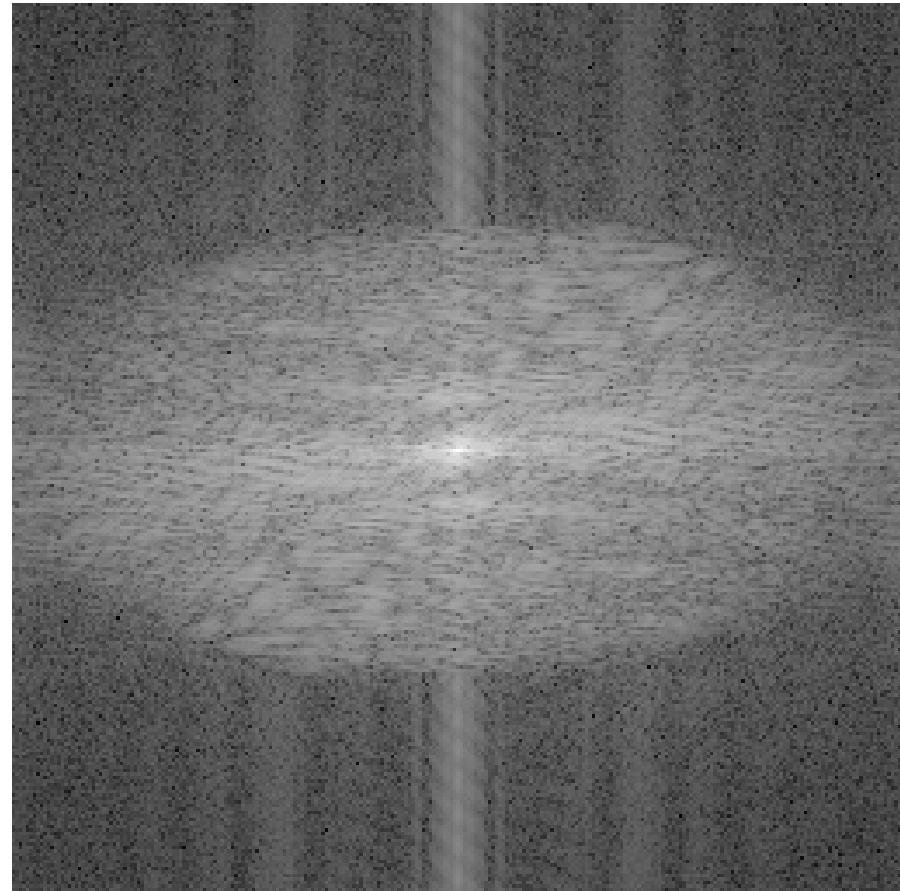
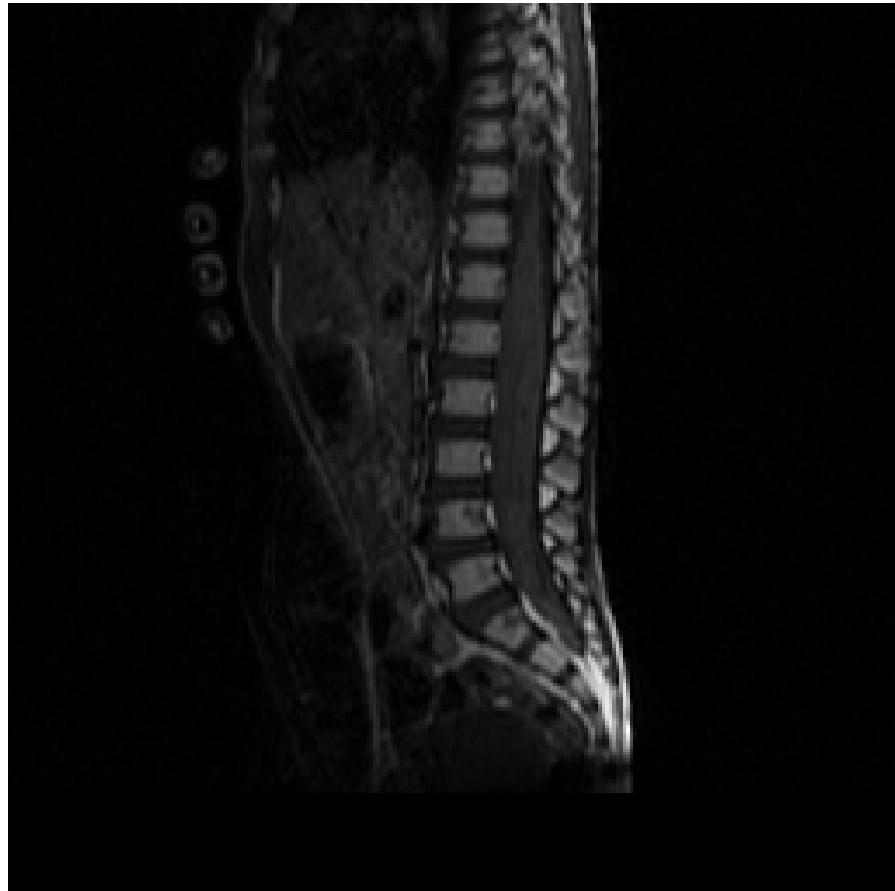
- Large DFT coefficients along certain orientations correspond to **directional** structure in the image.
- When there are strong, oriented structures in the DFT, strongly directional structures will be visible in the image.
- Visible structure in the centered log-magnitude spectrum is **orthogonal** to the structure in the image.
- This is because the direction of the frequency vector $[u \ v]^T$ is normal to the wavefronts in the image (see page 4.40).





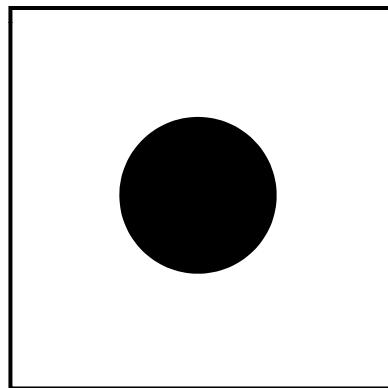




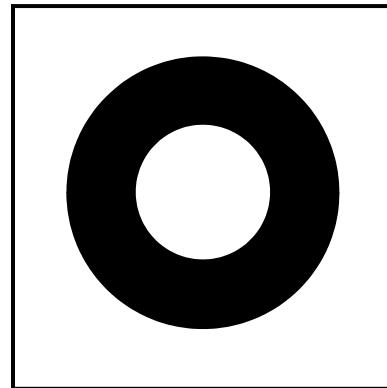


Granularity Selective Masking

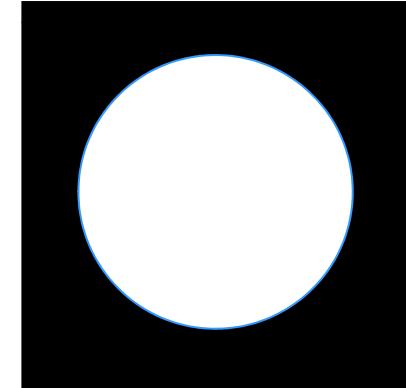
- Define zero-one masks (black = 1)



low-frequency mask



mid-frequency mask

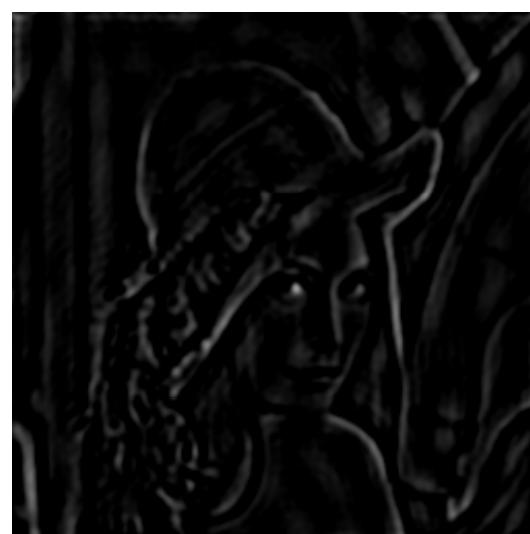
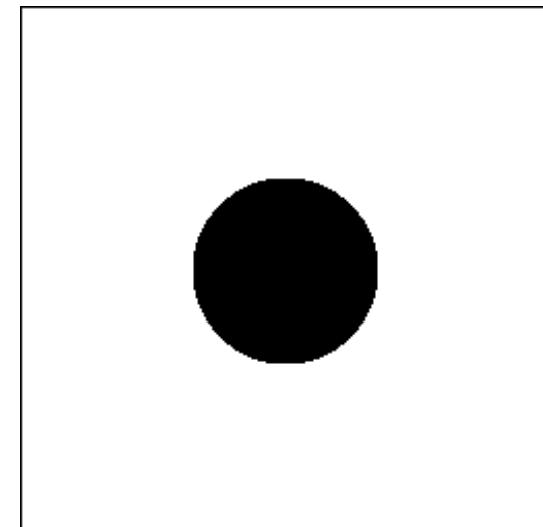
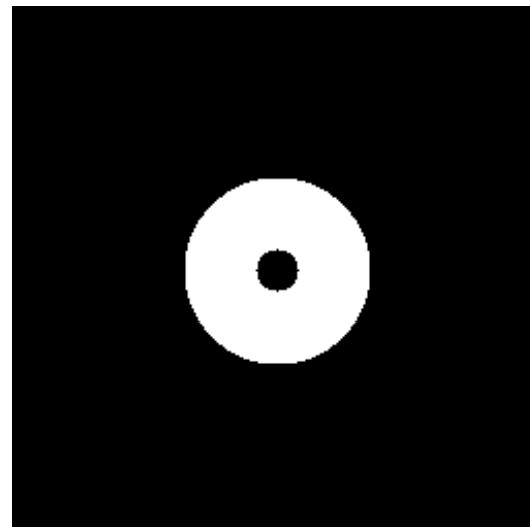
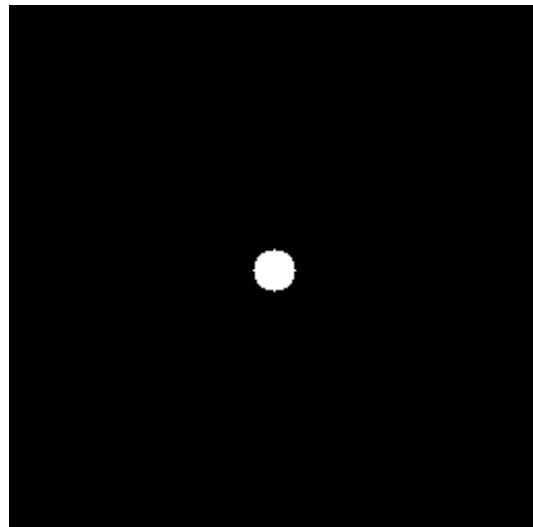


high-frequency mask

- **Masking** (multiplying) a DFT with these will produce IDFT images retaining only low, middle, or high radial frequencies.

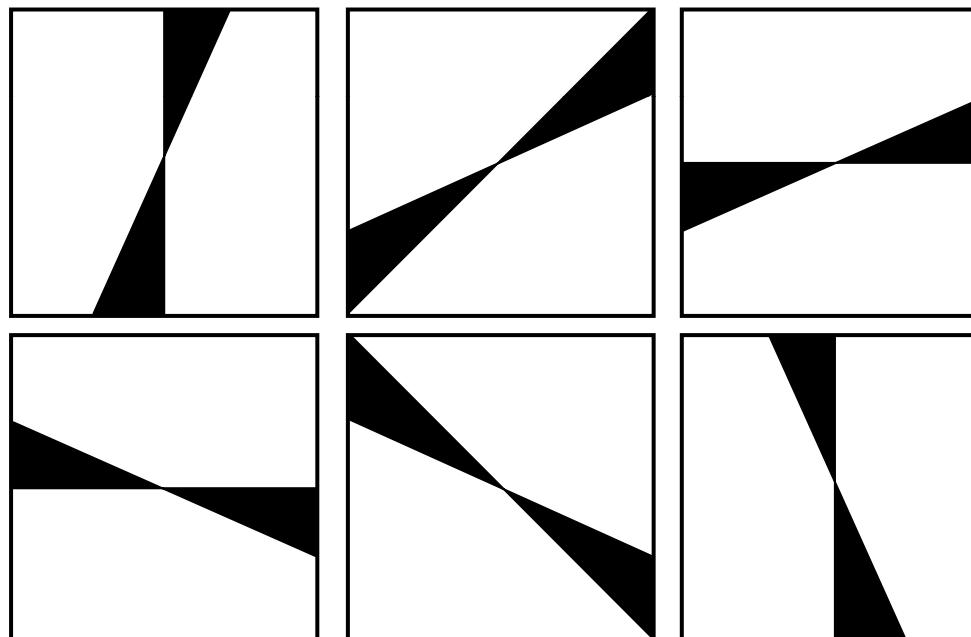
Granularity Selective Masking

(white = 1)



Orientation Selective Masking

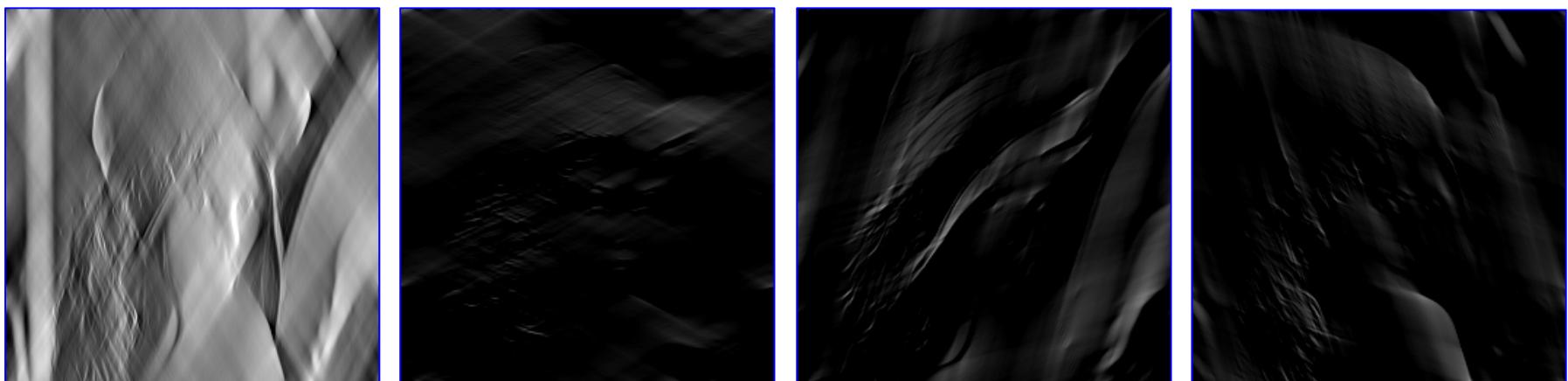
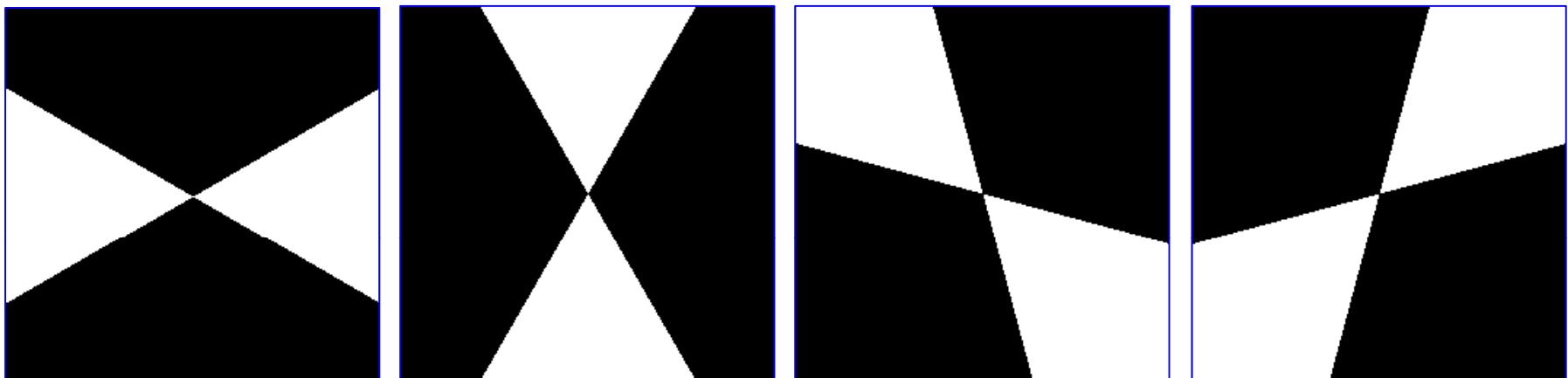
- Define **oriented**, angular **zero-one masks**:



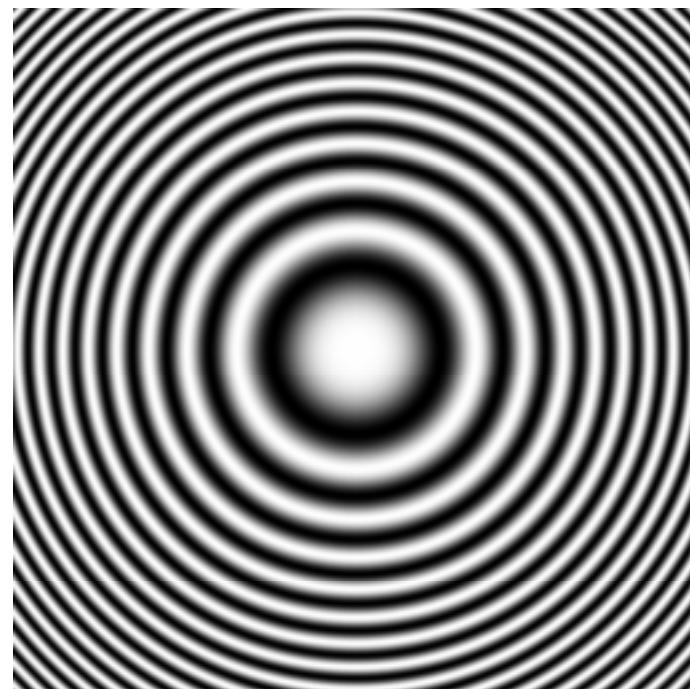
- Multiplying these times the DFT and inverting will retain **only the selected** orientations.

Orientation Selective Masking

(white = 1)

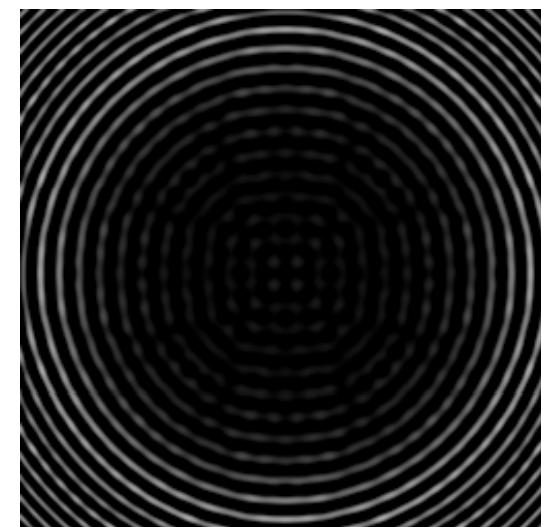
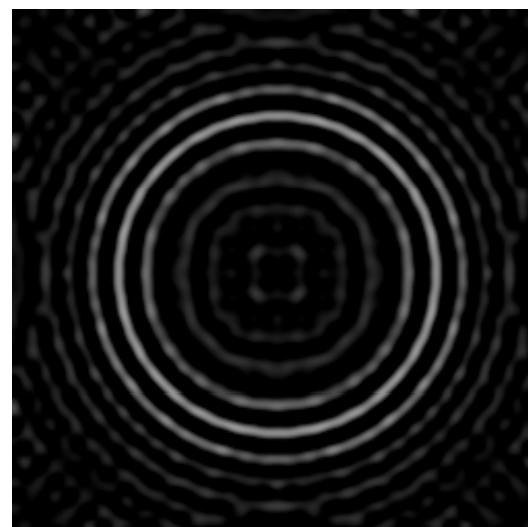
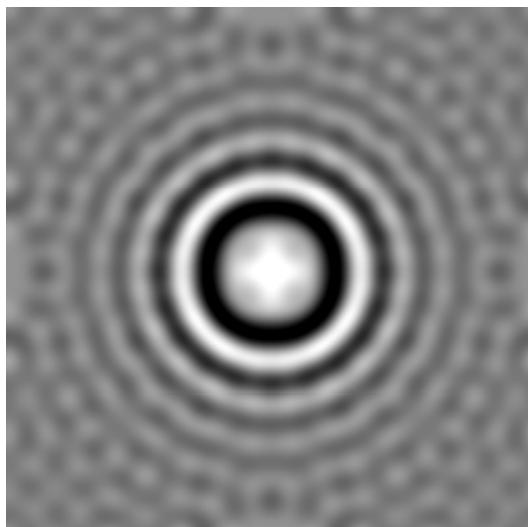
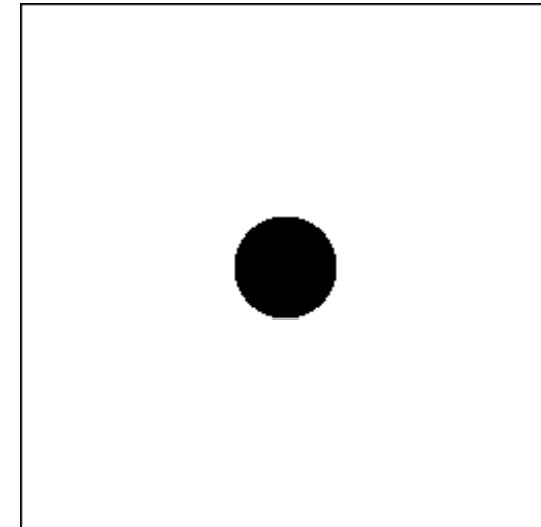
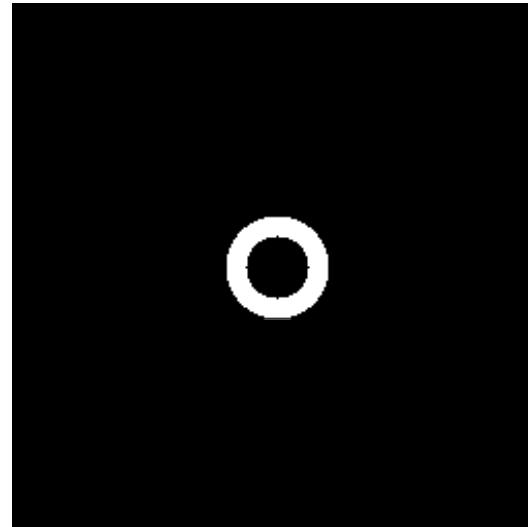
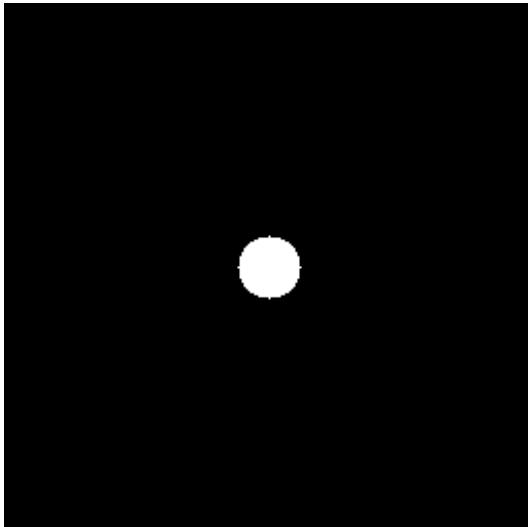


Chirp Image



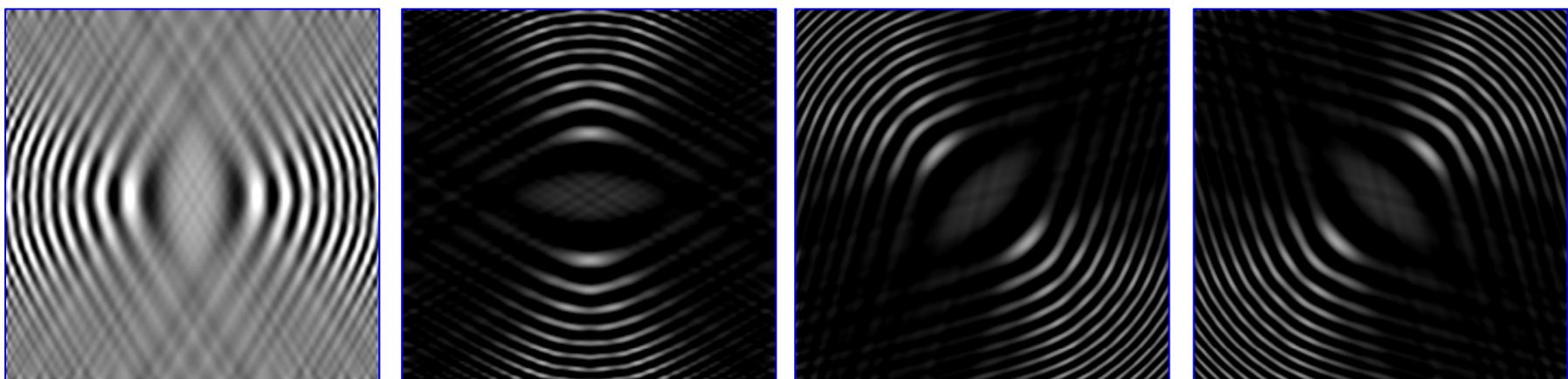
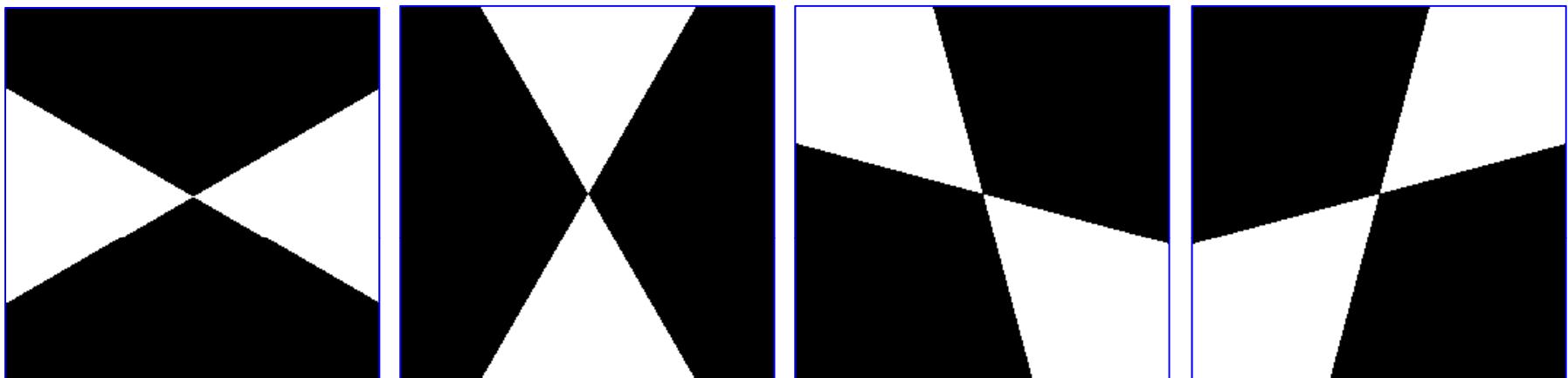
Granularity Selective Masking

(white = 1)



Orientation Selective Masking

(white = 1)



Granularity Selective Masking

```
fidx = fopen('/home/joebob/Images/lena.bin','r');
[x,junk] = fread(fidx,[256,256], 'uchar');
x = x';
[COLS,ROWS] = meshgrid(0:255,0:255);
Lcutoff = 10;
LMask = (sqrt((ROWS-128).^ 2 + (COLS-128).^ 2) <= Lcutoff);
imshow(255*LMask, [0 255]);
imwrite(255*LMask, 'LMask.tif', 'tif');
xLow = ifft2(fftshift(fftshift(fft2(x)) .* LMask));
imshow(real(xLow), [0 255]);
imwrite(uint8(real(xLow)), 'LenaLow.tif', 'tif');

Hcutoff = 45;
HMask = (sqrt((ROWS-128).^ 2 + (COLS-128).^ 2) >= Hcutoff);
xHi = ifft2(fftshift(fftshift(fft2(x)) .* HMask));
imshow(real(xHi), [0 255]);
imwrite(uint8(real(xHi)), 'LenaHi.tif', 'tif');

MMask = (1 - HMask) - LMask;
imshow(255*MMask, [0 255]);
imwrite(255*MMask, 'MMask.tif', 'tif');
xMid = ifft2(fftshift(fftshift(fft2(x)) .* MMask));
imshow(real(xMid), [0 255]);
imwrite(uint8(real(xMid)), 'LenaMid.tif', 'tif');
```

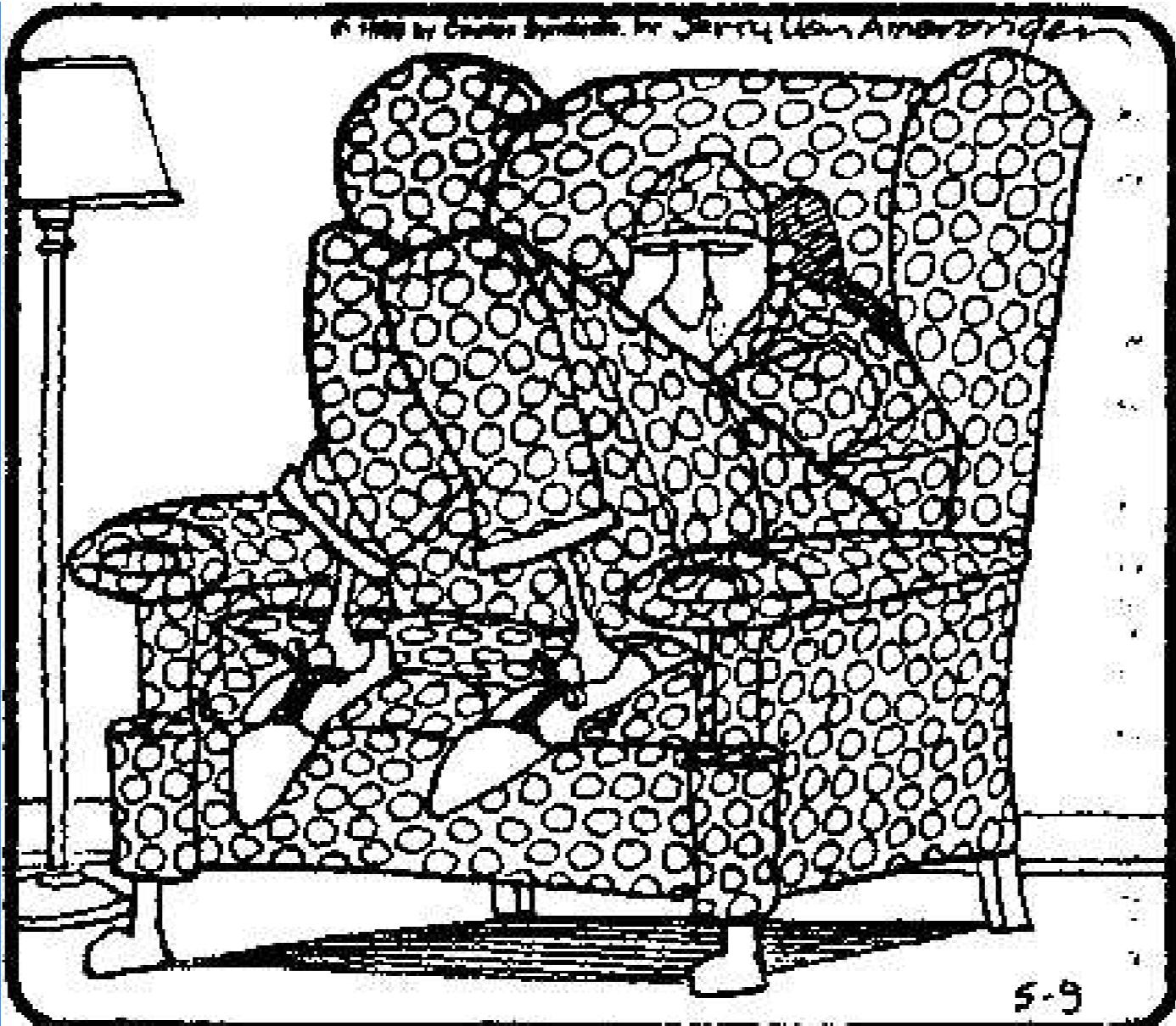
Orientation Selective Masking

```
fidx = fopen('/home/joebob/Images/lena.bin','r');
[x,junk] = fread(fidx,[256,256], 'uchar');
x = x';
[COLS,ROWS] = meshgrid(0:255,0:255);

Cp = 3*pi/4;           % center angle (plus)
Cm = Cp+pi;           % center angle (opposite)
W = pi/6;              % angular half-bandwidth
U = COLS-128;
V = ROWS-128;
Theta = atan2(V,U);
Mask = ((Theta >= Cp-W) .* (Theta <= Cp+W)) + ...
       ((Theta+2*pi >= Cp-W) .* (Theta+2*pi <= Cp+W)) + ...
       ((Theta-2*pi >= Cp-W) .* (Theta-2*pi <= Cp+W)) + ...
       ((Theta >= Cm-W) .* (Theta <= Cm+W)) + ...
       ((Theta+2*pi >= Cm-W) .* (Theta+2*pi <= Cm+W)) + ...
       ((Theta-2*pi >= Cm-W) .* (Theta-2*pi <= Cm+W));
imshow(255*Mask, [0 255]);
imwrite(255*Mask,'Mask.tif','tif');
y = ifft2(fftshift(fftshift(fft2(x)) .* Mask));
%
imshow(real(y), [0 255]);
title('Orientation Masked Image');
imwrite(uint8(real(y)), 'y.tif','tif');
```



When the monster came, Lola, like the
peppered moth and the arctic hare,
remained motionless and undetected.
Harold, of course, was immediately devoured.



5-9

You get the feeling Bob's not going out
and grabbing life by the throat anytime
soon.

DISCRETE-SPACE FOURIER TRANSFORM (DSFT)

- We have been looking at the 2D DFT. It's the 2D version of the 1D DFT on page 4.22.
- In 1D, we usually use n for the time variable (or index) and k for the frequency variable (which you can think of as indexing the basis signals).
- In a general N D formulation, you could use $n = (n_1 \ n_2 \ \dots \ n_N)$ for the "time" index and $k = (k_1 \ k_2 \ \dots \ k_N)$ for the frequency index.
- For the 2D case, we used (m, n) for the space index and (u, v) for the frequency index. We did this to minimize the use of subscripts. This is a **widely** used convention.
- Now we are going to look at the 2D version of the 1D DTFT on page 4.13. It is called the 2D **Discrete-Space Fourier Transform (DSFT)**.
- For the 2D DSFT, we will use (m, n) for the space variables and (U, V) for the frequency variables.

THE DISCRETE-SPACE FOURIER TRANSFORM (DSFT)

- Let the image \mathbf{I} be defined on *all* of \mathbb{Z}^2 .
- The **2D Discrete-Space Fourier Transform** is given by:

$$\tilde{\mathbf{I}}_D(U, V) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \mathbf{I}(m, n) e^{-j2\pi(Um+Vn)} \quad (\text{DSFT})$$

$$\mathbf{I}(m, n) = \int_{-\frac{1}{2}}^{\frac{1}{2}} \int_{-\frac{1}{2}}^{\frac{1}{2}} \tilde{\mathbf{I}}_D(U, V) e^{j2\pi(Um+Vn)} dU dV \quad (\text{IDSFT})$$

Discrete-Space Fourier Transform

- The units of U and V are cycles per pixel.
 - We could alternatively use units of radians per pixel. The limits of integration in the IDSFT would then be $-\pi$ to π and a factor of $1/(2\pi)^2$ would be required on the IDSFT.
- The **DSFT is periodic** with period ONE along each axis.
- The DSFT is **continuous** in spatial frequency.
- In most practical cases, the image I has a **finite** size.
- For (m, n) **outside** the range $0 \leq m \leq M-1$, $0 \leq n \leq N-1$, we set $I(m, n) = 0$ for the DSFT.

Relating the DSFT and DFT

- In the theoretical case, it's possible to have an image \mathbf{I} of infinite size. Such an image has a DSFT $\tilde{\mathbf{I}}_D$, but it does not have a DFT $\tilde{\mathbf{I}}$.
- Any practical image \mathbf{I} is finite in size. It has a DFT $\tilde{\mathbf{I}}$. By zero padding the image to all of \mathbb{Z}^2 , we can also give it a DSFT $\tilde{\mathbf{I}}_D$.
- For a finite sized image, there is a relationship between the DSFT and the DFT. The DFT is given by **samples** of the DSFT:

$$\tilde{\mathbf{I}}(u, v) = \tilde{\mathbf{I}}_D(U, V) \Big|_{U=\frac{u}{M}, V=\frac{v}{N}} ; u=0, \dots, M-1, \quad v=0, \dots, N-1$$

DSFT

- It is more usual to introduce the DSFT first, followed by the DFT.
- However, our emphasis is on **algorithms** and **applications**, which generally use the **DFT**.
- We will find the **DSFT** to be a **valuable analytic tool**.

Sampling

- We will now study the relationships between the DFT/DSFT and the continuous 2D Fourier transform of the **original, unsampled optical image** that exists inside the camera.
- The digital image \mathbf{I} is a sampled version of the **continuous optical intensity image** $I_C(x, y)$ that is focused through the lens system to become incident upon a sensor located on the camera focal plane.

Notation and Units

- For the n D continuous Fourier transform on page 4.25, we used a vector \mathbf{x} for the spatial variable and a vector \mathbf{u} for the frequency variable.
- For 2D continuous optical images, we will use $\mathbf{x} = (x, y)$ to avoid subscripts. Thus, we will write $I_C(x, y)$.
- The spatial coordinates (x, y) can be expressed in mm, inches, or any other appropriate unit of length.
- Often, it is convenient to express them in units given by the horizontal and vertical detector spacings on the focal plane, i.e., in units that are given by the physical size of a pixel in the camera.
- Again to avoid subscripts, we will write the 2D continuous Fourier transform frequency variable as $\mathbf{u} = (\Omega, \Lambda)$, where Ω = horizontal frequency and Λ = vertical frequency.

Continuous Fourier Transform

- The continuous image $I_C(x, y)$ has a **Continuous Fourier Transform (CFT)** $\tilde{I}_C(\Omega, \Lambda)$ where (x, y) are space coordinates and (Ω, Λ) are Hertzian spatial frequencies:

$$\tilde{I}_C(\Omega, \Lambda) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I_C(x, y) e^{-j2\pi(\Omega x + \Lambda y)} dx dy \quad (\text{CFT})$$

$$I_C(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \tilde{I}_C(\Omega, \Lambda) e^{j2\pi(\Omega x + \Lambda y)} d\Omega d\Lambda \quad (\text{ICFT})$$

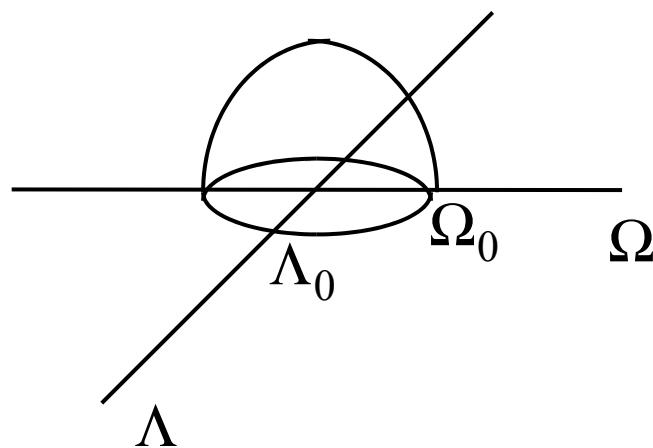
Continuous Fourier Transform

- The CFT is **not periodic** in general.
- The horizontal frequency Ω and vertical frequency Λ are continuous and expressed in cycles per unit of length, i.e. per unit of (x,y) .
- The unit of length must be specified. For convenience, we will usually assume it is the height/width of a pixel on the detector.

Relating the CFT and DSFT/DFT

- Assume $\tilde{I}_C(\Omega, \Lambda)$ is **bandlimited**, or **zero** outside a certain range of frequencies:

$$\tilde{I}_C(\Omega, \Lambda) = 0 \text{ for } |\Omega| > \Omega_0, |\Lambda| > \Lambda_0$$



On Bandlimitedness

- Any real-world image is **approximately bandlimited**: its CFT becomes vanishingly small for large Ω, Λ .
- If it were not so the image would contain infinite energy since (Parseval)

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |I_C(x, y)|^2 dx dy = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |\tilde{I}_C(\Omega, \Lambda)|^2 d\Omega d\Lambda$$

- Ex: a Gaussian image has a Gaussian CFT which is not bandlimited. But the integrals converge because the CFT vanishes as $\Omega, \Lambda \rightarrow \infty$.

Image Sampling

- The detector creates $I(m, n)$ by sampling the optical image with spacings X, Y in the x - & y -directions:

$$I(m, n) = I_C(mX, nY); m = 0, \dots, M-1, \quad n = 0, \dots, N-1$$

- The DSFT and CFT are related by:

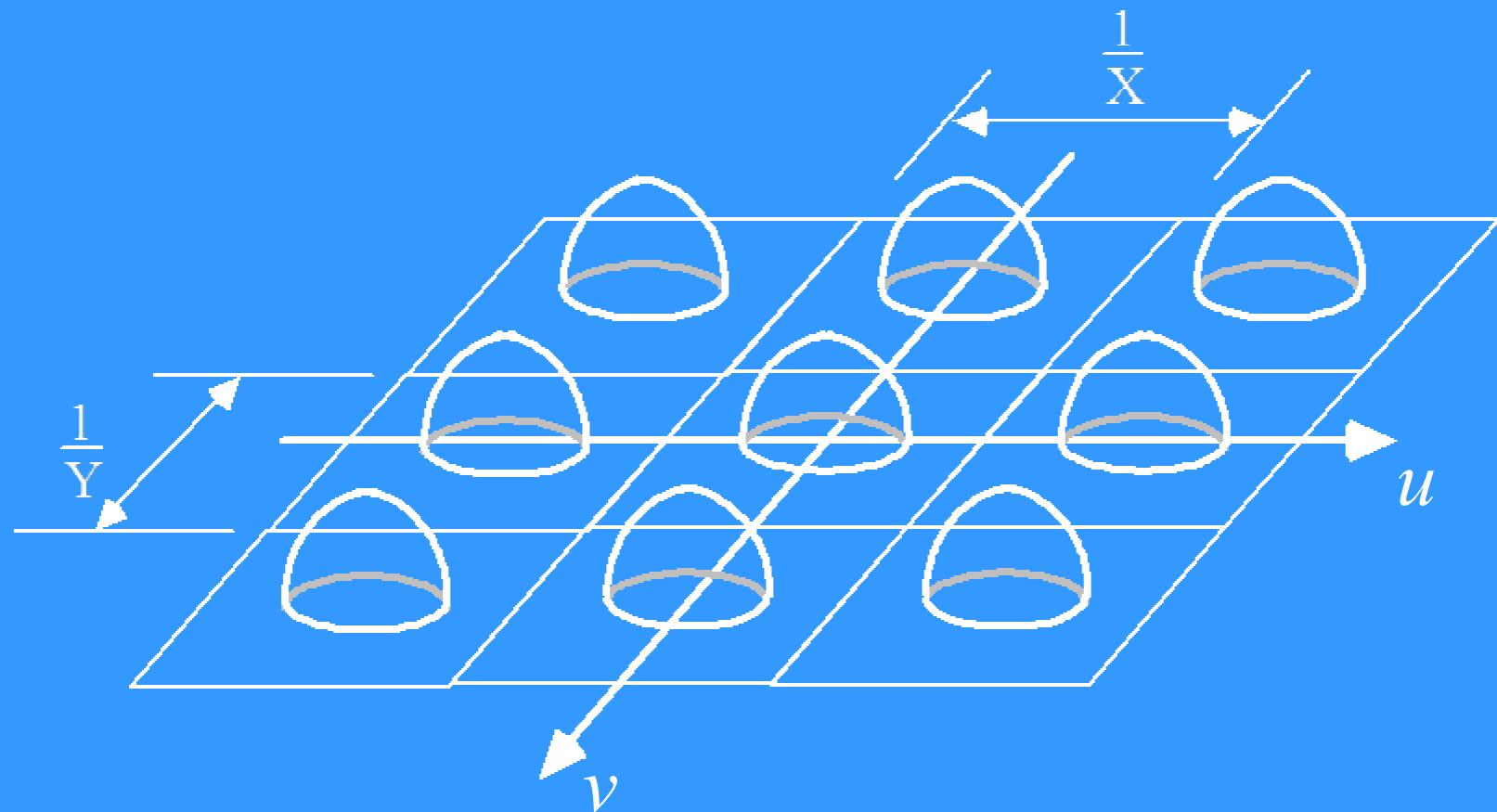
$$\begin{aligned}\tilde{I}_D(U, V) &= \frac{1}{XY} \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} \tilde{I}_C\left(\Omega - \frac{p}{X}, \Lambda - \frac{q}{Y}\right) \Big|_{(\Omega, \Lambda) = \left(\frac{U}{X}, \frac{V}{Y}\right)} \\ &= \frac{1}{XY} \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} \tilde{I}_C\left[\frac{1}{X}(U-p), \frac{1}{Y}(V-q)\right]\end{aligned}$$

Relating DFT to CFT

- We have:

$$\begin{aligned}\tilde{I}(u, v) &= \tilde{I}_D(U, V) \Big|_{U=\frac{u}{M}, V=\frac{v}{N}} \\ &= \frac{1}{XY} \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} \tilde{I}_C \left[\frac{1}{X} \left(\frac{u}{M} - p \right), \frac{1}{Y} \left(\frac{v}{N} - q \right) \right]\end{aligned}$$

- A sum of shifted versions of the CFT. It is periodic in the u - and v -directions with periods $1/X$ and $1/Y$, respectively.



Unit-Period Case

- We can always set $X = Y = 1$.
- This simply says that we measure length and area in units that are given by the physical size of a pixel on the detector array (usually on the order of microns).
- Then

$$\tilde{I}(u, v) = \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} \tilde{I}_C \left[\left(\frac{u}{M} - p \right), \left(\frac{v}{N} - q \right) \right]$$

Sampling Theorem

- If $\Omega_0 > \frac{1}{2X}$ or $\Lambda_0 > \frac{1}{2Y}$, the replicas of the CFT will **overlap** (sum up), **distorting them**. This is called **aliasing**.
- The digital image can be severely **distorted**.
- To avoid aliasing, the **sampling frequencies** $1/X$ and $1/Y$ must be at least **twice** the highest frequencies Ω_0 and Λ_0 in the continuous image.

Comments on Sampling

- This is a **mathematical** reason why images must be sampled sufficiently densely. If violated, image distortion can be **visually severe**.
- If the Sampling Theorem is **satisfied**, then the DFT is periodic (sampled) replicas of the CFT.

Aliased Chirp Image

- A chirp image

$$I_C(x, y) = A \cos[\varphi(x, y)] = A \cos(ax^2 + by^2)$$

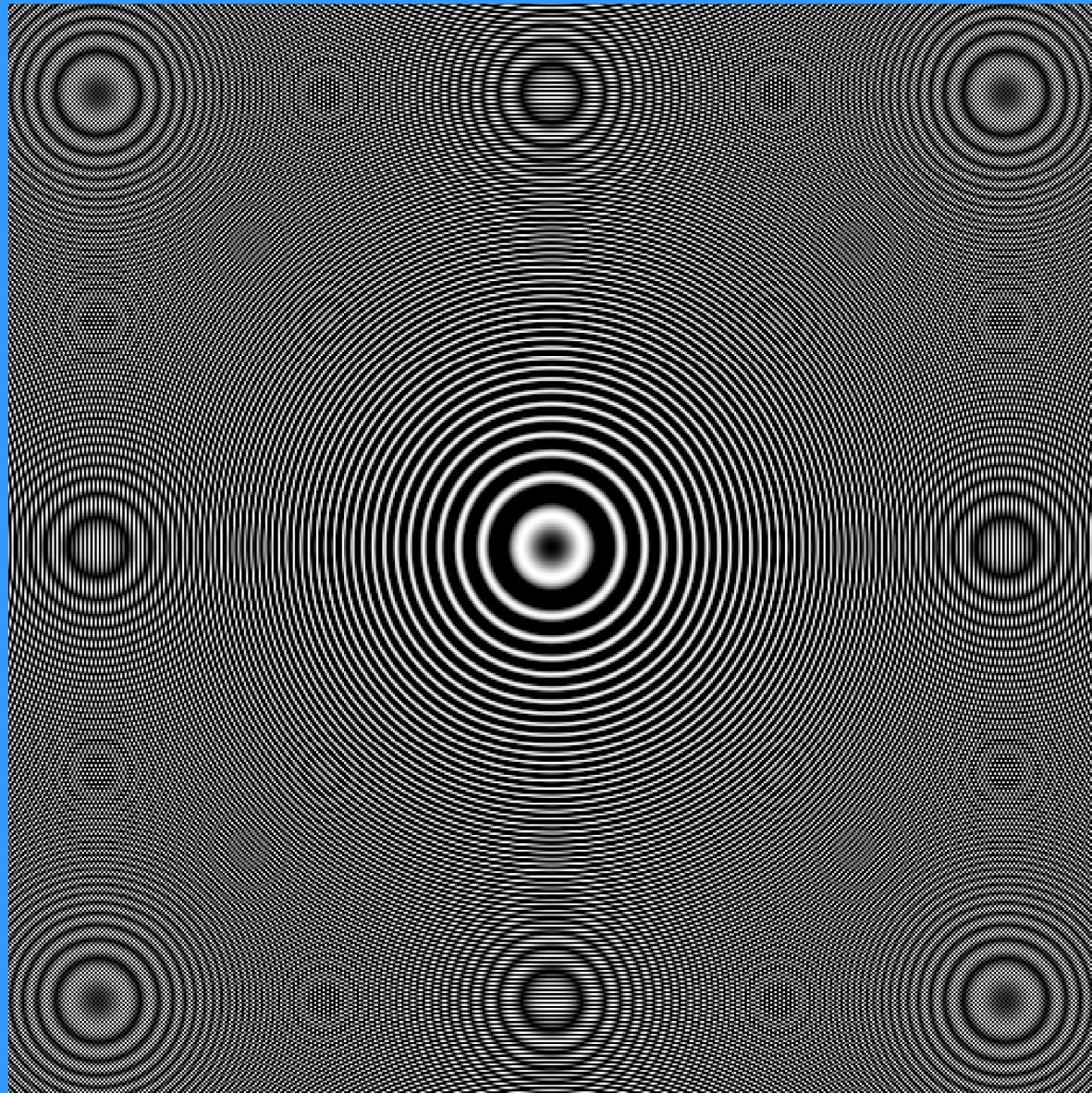
has **instantaneous spatial frequencies**

$$(\Omega_{\text{inst}}, \Lambda_{\text{inst}}) = (\varphi_x(x, y), \varphi_y(x, y)) = (2ax, 2by)$$

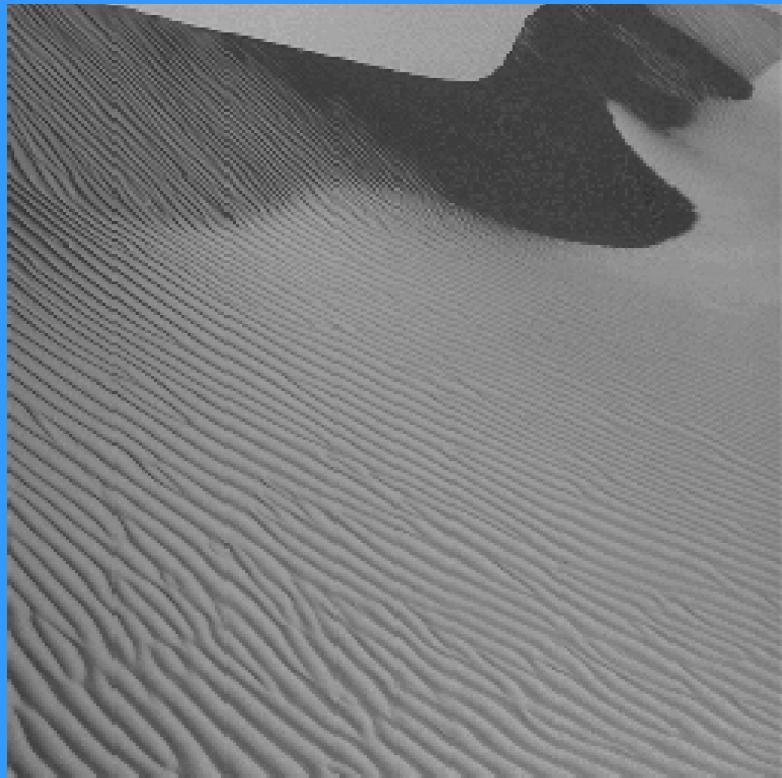
which **increase linearly** away from the origin.

- For such an image, the origin $(x, y) = (0, 0)$ is usually taken at the center of the image.

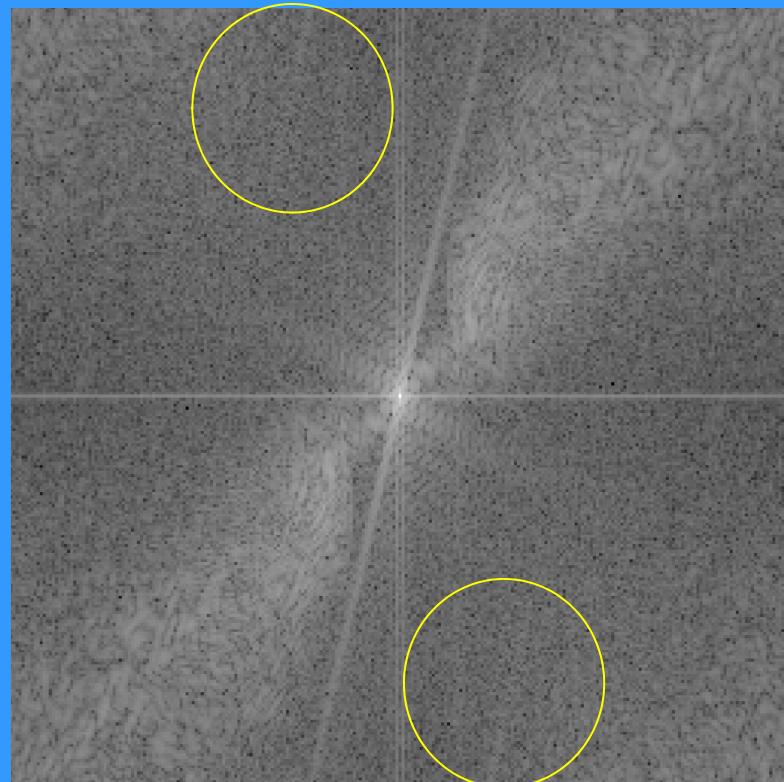
Sampled Chirp



Aliased Image



Sand Dune Image



Centered DFT Showing Aliasing

IMPORTANT 2-D FUNCTIONS

AND THEIR DFTS

- It is worthwhile to examine the DFTs of some **specific images**. This is usually hard to do explicitly by hand for the DFT/DSFT.
- So we'll give some simple ones.
- Then state some others as **CFT transform pairs**.

Constant Image

- Let $I(m, n) = c \quad \text{for } 0 \leq m \leq M-1, 0 \leq n \leq N-1$
- Then

$$\tilde{I}(u, v) = c MN \delta(u, v)$$

where

$$\delta(u, v) = \begin{cases} 1 & ; u=v=0 \\ 0 & ; \text{else} \end{cases} = \text{unit impulse}$$

2-D Unit Pulse Image

- Let $I(m, n) = c\delta(m, n)$
- Then

$$\begin{aligned}\tilde{I}(u, v) &= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} c\delta(m, n) W_M^{um} W_N^{vn} \\ &= cW_N^0 W_M^0 = c \quad (\text{constant DFT})\end{aligned}$$

Cosine Wave Image

- Let

$$I(m, n) = d \cos \left[2\pi \left(\frac{b}{M}m + \frac{c}{N}n \right) \right] = \left(\frac{d}{2} \right) \left[W_M^{bm} W_N^{cn} + W_M^{-bm} W_N^{-cn} \right]$$

- Then

$$\begin{aligned}\tilde{I}(u, v) &= \left(\frac{d}{2} \right) \sum_{m=0}^{N-1} \sum_{n=0}^{M-1} \left[W_M^{bm} W_N^{cn} + W_M^{-bm} W_N^{-cn} \right] W_M^{um} W_N^{vn} \\ &= \left(\frac{d}{2} \right) \sum_{m=0}^{N-1} \sum_{n=0}^{M-1} \left[W_M^{(u+b)m} W_N^{(v+c)n} + W_M^{(u-b)m} W_N^{(v-c)n} \right] \\ &= \left(\frac{d}{2} \right) MN [\delta(u-b, v-c) + \delta(u+b, v+c)]\end{aligned}$$

Sine Wave Image

- Likewise, if

$$I(m, n) = d \sin\left[2\pi\left(\frac{b}{M}m + \frac{c}{N}n\right)\right]$$

then

$$\tilde{I}(u, v) = j\left(\frac{d}{2}\right)MN[\delta(u+b, v+c) - \delta(u-b, v-c)]$$

- The Fourier representations of pure sinusoids are concentrated **single frequencies**

DFT AS A SAMPLED CFT

- Now some CFT pairs that are either **difficult to express** or are too lengthy to do by hand as DFT (or even as DSFT) pairs.
- If **sampled adequately**, then
 - the DSFT is given by appropriately scaled periodic repetitions of the CFT.
 - the DFT is given by samples of the DSFT.

Rectangle Function

- Let

$$I_C(x, y) = c \cdot \text{rect}\left(\frac{x}{A}\right) \cdot \text{rect}\left(\frac{y}{B}\right) = \begin{cases} c ; & |x| \leq \frac{A}{2}, |y| \leq \frac{B}{2} \\ 0 ; & \text{else} \end{cases}$$

- Then

$$\tilde{I}_C(\Omega, \Lambda) = c \cdot A \cdot B \cdot \text{sinc}(A\Omega) \cdot \text{sinc}(B\Lambda)$$

where

$$\text{rect}(x) = \begin{cases} 1 ; & |x| \leq \frac{1}{2} \\ 0 ; & \text{else} \end{cases} \quad \text{and} \quad \text{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$$

Sinc Function

- Let

$$I_C(x, y) = c \cdot \text{sinc}(ax) \cdot \text{sinc}(by)$$

then

$$\tilde{I}_C(\Omega, \Lambda) = \frac{c}{ab} \cdot \text{rect}\left(\frac{\Omega}{a}\right) \cdot \text{rect}\left(\frac{\Lambda}{b}\right) = \begin{cases} \frac{c}{ab}; & |\Omega| \leq \frac{a}{2}, |\Lambda| \leq \frac{b}{2} \\ 0; & \text{else} \end{cases}$$

Gaussian Function

- Let

$$I_C(x, y) = \exp\left[-\frac{(x^2 + y^2)}{\sigma^2}\right]$$

then

$$\tilde{I}_C(\Omega, \Lambda) = \pi\sigma^2 \exp\left[-\pi^2\sigma^2(\Omega^2 + \Lambda^2)\right]$$

- The Fourier transform of a Gaussian is also Gaussian – an **unusual property**.
- Note: since $\tilde{I}_C(\Omega, \Lambda)$ is **not** bandlimited, the digital image **will** be aliased.

Comments

- We now have a basic understanding of **frequency-domain concepts** in 2D.
- Let's put them to use in **linear filtering applications**... onward to **Module 5**.