

# Computer Vision

## Computer Science Tripos Part II

Dr Christopher Town

3. Mathematical operations for extracting structure from images.



Dr Chris Town

## Fourier Analysis

Any image can be represented by a linear combination of basis functions:

$$f(x, y) = \sum_k a_k \Psi_k(x, y)$$

In the case of 2D Fourier analysis:

$$\Psi_k(x, y) = \exp(i(\mu_k x + \nu_k y))$$

$$\exp(i\theta) = \cos(\theta) + i \sin(\theta)$$

Dr Chris Town

## Fourier Analysis

$$f(x, y) = \sum_k a_k \exp(i(\mu_k x + \nu_k y))$$

The transform finds a set of coefficients  $a_k$  for every spatial frequency and orientation in the 2D Fourier domain spanned by the 2D frequency variables  $(\mu_k, \nu_k)$ . These coefficients may be computed by the **Fourier Transform**:

$$a_k = \int_X \int_Y \exp(-i(\mu_k x + \nu_k y)) f(x, y) dx dy$$

$$F(\mu, \nu) = \int_X \int_Y \exp(-i(\mu x + \nu y)) f(x, y) dx dy$$

Each  $F(\mu, \nu)$  is a complex coefficient which defines the magnitude and phase of a sinusoid basis function.

Dr Chris Town

## Fourier Analysis

$$\exp(i\theta) = \cos(\theta) + i \sin(\theta)$$

Real part - cosine wave, Imaginary part - sine wave

Complex exponentials are the Eigenfunctions of linear systems

$$\exp(i\mu_k t) \rightarrow \boxed{h(t)} \rightarrow A \exp(i\mu_k t)$$

The Fourier transform is a linear operation:

$$\mathcal{F}(af(x) + bg(x)) = a\mathcal{F}(f(x)) + b\mathcal{F}(g(x))$$

Dr Chris Town

## Series expansion of transcendental functions

$$\exp(\theta) = 1 + \frac{\theta}{1!} + \frac{\theta^2}{2!} + \frac{\theta^3}{3!} + \dots + \frac{\theta^n}{n!} + \dots,$$

$$\cos(\theta) = 1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \frac{\theta^6}{6!} + \dots,$$

$$\sin(\theta) = \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \frac{\theta^7}{7!} + \dots,$$

$$e^{i\pi} + 1 = 0$$

{0,1} represent arithmetic,  $\pi$  represents geometry,  $i$  represents algebra, and  $e = 2.718\dots$  represents analysis, since one way to define  $e$  is to compute the limit of  $(1 + \frac{1}{n})^n$  as  $n \rightarrow \infty$ .

Dr Chris Town

$$f(x, y) = \sum_k a_k \exp(i(\mu_k x + \nu_k y))$$

where the parameters  $\mu_k$  and  $\nu_k$  define the coordinates of the 2D Fourier domain. These  $(\mu_k, \nu_k)$  coordinates are called vector spatial frequencies, and the array of them must span the  $(\mu, \nu)$  Fourier plane in a uniform Cartesian lattice.

It is often useful to think of the  $(\mu, \nu)$  Fourier plane as resolved into polar coordinates, where  $\omega = \sqrt{\mu^2 + \nu^2}$  is (scalar) spatial frequency and  $\phi = \tan^{-1}(\nu/\mu)$  is (scalar) orientation.

Dr Chris Town

## The Discrete Fourier Transform

Fourier Transform

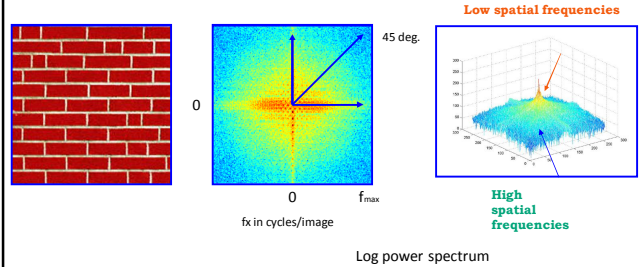
$$F[u, v] = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f[x, y] e^{-i\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)}$$

Inverse Fourier Transform (reconstruction)

$$f[x, y] = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} F[m, n] e^{+i\pi \left( \frac{mx}{M} + \frac{ny}{N} \right)}$$

Dr Chris Town

## How to interpret a Fourier Spectrum



A.Torrallba

Dr Chris Town

The Fourier Transform coefficients are complex valued:

$$F[u, v] = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f[x, y] e^{-i\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)}$$

$$= R[u, v] + iJ[u, v]$$

$$e^{-i\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)} = \cos\left(\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)\right) - i \sin\left(\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)\right)$$

$$\cos(-\theta) = \cos(\theta) \quad \sin(-\theta) = -\sin(\theta)$$

Dr Chris Town

The Fourier Transform coefficients are complex:

$$F[u, v] = R[u, v] + iJ[u, v]$$

Real component:

$$R[u, v] = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f[x, y] \cos\left(\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)\right)$$

Imaginary component:

$$J[u, v] = -\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f[x, y] \sin\left(\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)\right)$$

Dr Chris Town

Representation in terms of **magnitude** and **phase**

$$F[u, v] = R[u, v] + iJ[u, v]$$

$$= |F(u, v)| e^{i\Phi(u, v)}$$

Magnitude spectrum

$$|F(u, v)| = \sqrt{R(u, v)^2 + I(u, v)^2} = \sqrt{F(u, v)F^*(u, v)}$$

$$F(u, v)F^*(u, v) \quad \text{Power spectrum}$$

Phase spectrum

$$\Phi(u, v) = \tan^{-1} \left( \frac{I(u, v)}{R(u, v)} \right)$$

Dr Chris Town

## The Discrete Fourier Transform

The Fourier Transform of a real-valued image is conjugate-symmetric

$$F[u, v] = F^*(-u, -v)$$

Therefore the magnitude spectrum is even symmetric

$$|F(u, v)| = |F(-u, -v)|$$

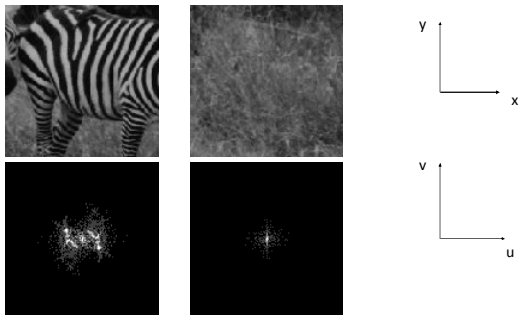
and the phase spectrum is odd symmetric

$$\Phi(u, v) = -\Phi(-u, -v)$$

Note that the Fourier spectrum is often re-arranged for display such that the zero-frequency component is in the centre.

Dr Chris Town

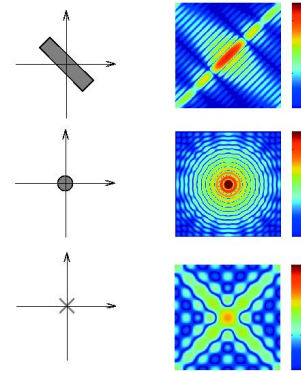
Stripes of the zebra create high energy waves generally along the u-axis; grass pattern is fairly random causing scattered low frequency energy



Computer Vision - A Modern Approach - Set: Pyramids and Texture - Slides by D.A. Forsyth

Dr Chris Town

Fourier Transforms of Some Simple Shapes



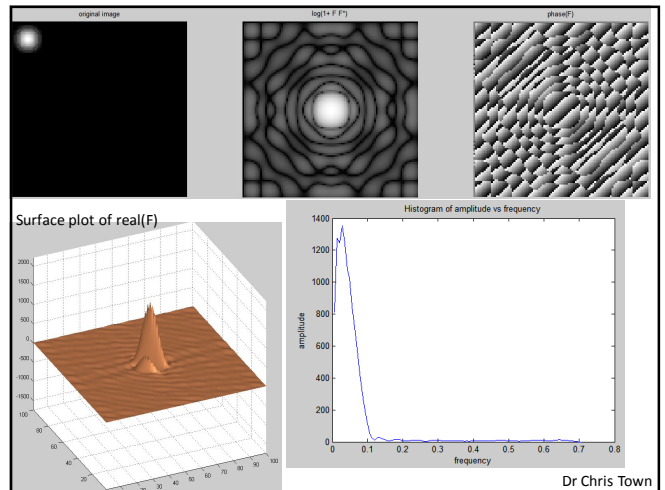
Source: Matlab 7 Documentation

Dr Chris Town

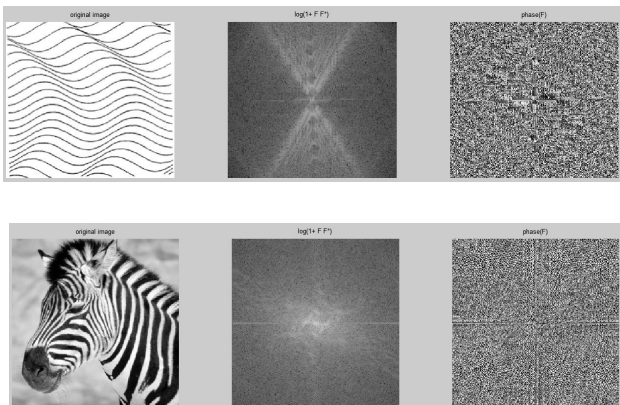
### Demo Matlab code

```
im = imread('waves.bmp');
F = fft2(im);
%fft2(X) returns the two-dimensional discrete Fourier transform (DFT) of X,
computed with a fast Fourier transform (FFT) algorithm. The result Y is the
same size as X.
F = fftshift(F);
%fftshift(X) rearranges the outputs of fft, fft2, and fftn by moving the zero-
frequency component to the center of the array. It is useful for visualizing a
%Fourier transform with the zero-frequency component in the middle of the
spectrum
%angle(X): Phase angle
figure;
subplot(1,3,1), imagesc(im); colormap gray, axis image, axis off,
title('original image');
subplot(1,3,2), imshow(log(1+F.*conj(F)), []); title('log(1+ F F*)');
subplot(1,3,3), imshow(angle(F), []); title('phase(F)');
inspect(im); %IMSPECT - Plots image amplitude spectrum averaged over all
orientations.
showsurf(real(F)); %SHOWSURF - shows parametric surface in a convenient way
```

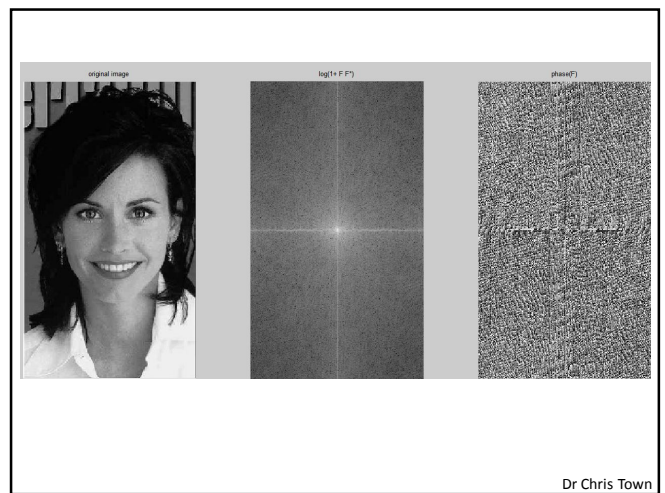
Dr Chris Town



Dr Chris Town

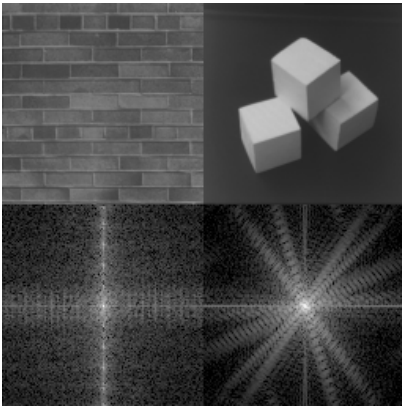


Dr Chris Town



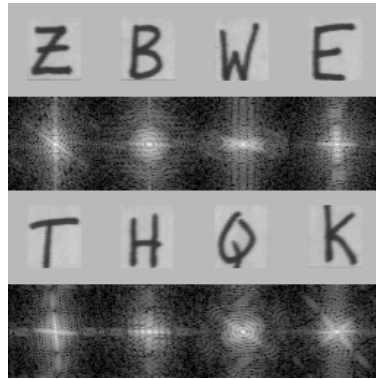
Dr Chris Town

DFT captures periodicity and directionality



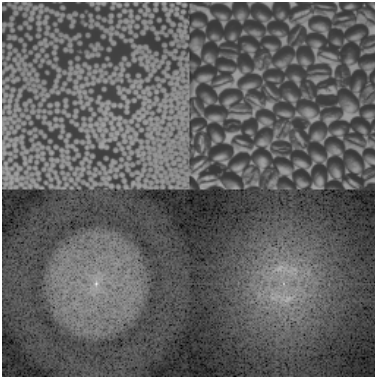
<http://www.cs.unm.edu/~brayer/vision/fourier.html> Dr Chris Town

DFT captures periodicity and directionality



<http://www.cs.unm.edu/~brayer/vision/fourier.html> Dr Chris Town

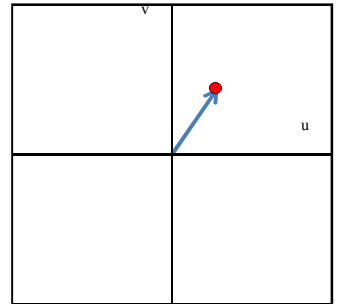
DFT captures periodicity and directionality



<http://www.cs.unm.edu/~brayer/vision/fourier.html> Dr Chris Town

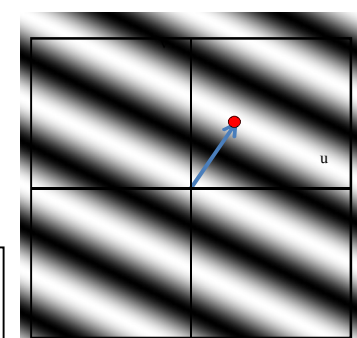
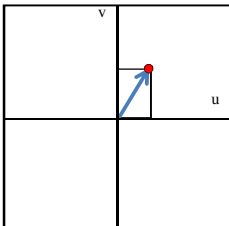
Visualising individual Fourier components

To get some sense of what basis elements look like, we plot a **basis element** (or rather, its real part) as a function of  $x, y$  for some fixed  $u, v$ .



Dr Chris Town

We get a function that is constant when  $(ux+vy)$  is constant. The **magnitude** of the vector  $(u, v)$  gives a **frequency**, and its **direction** gives an **orientation**. The function is a **sinusoid** with this frequency along this direction, and it is constant perpendicular to this direction.

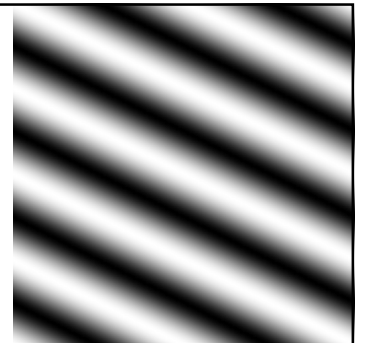
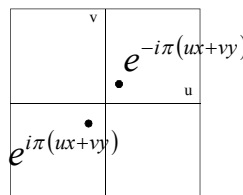


Length of  $(u,v)$  is proportional to frequency and inversely proportional to wavelength

Dr Chris Town

The Fourier coefficients represent the original image as a linear combination of such sinusoids. The coefficients are complex valued.

We can add a conjugate pair of complex exponentials to obtain a real-valued cosine function.



Dr Chris Town

Adding a conjugate pair of complex exponentials yields a real-valued cosine

$$\exp(i\theta) = \cos(\theta) + i \sin(\theta)$$

$$\cos(\theta) = \frac{\exp(i\theta) + \exp(-i\theta)}{2}$$

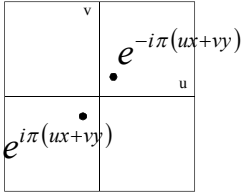
$$\sin(\theta) = \frac{\exp(i\theta) - \exp(-i\theta)}{2i}$$

$$f(t) = \frac{1}{5}e^{i2\pi st} + \frac{1}{2}e^{-i2\pi st}$$

$$= \frac{1}{2}[\cos(2\pi st) + i \sin(2\pi st)] + \frac{1}{2}[\cos(-2\pi st) + i \sin(-2\pi st)]$$

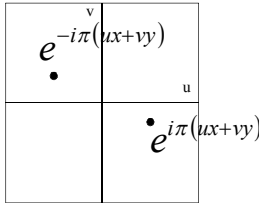
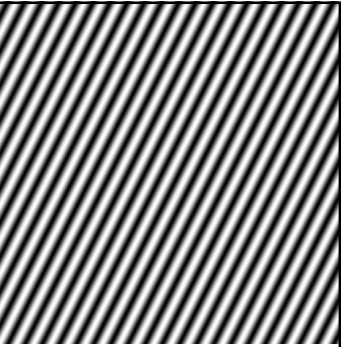
$$= \frac{1}{2}[\cos(2\pi st) + i \sin(2\pi st)] + \frac{1}{2}[\cos(2\pi st) - i \sin(2\pi st)]$$

$$= \frac{1}{2}[\cos(2\pi st)] + \frac{1}{2}[\cos(2\pi st)]$$

$$= \cos(2\pi st)$$


Dr Chris Town

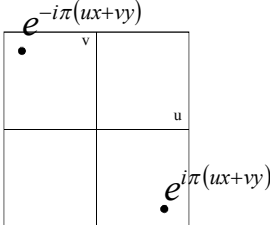
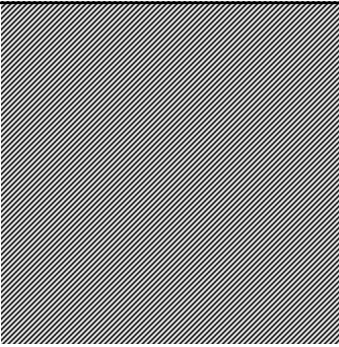
Here u and v are larger than in the previous slide.

A.Torralba

Dr Chris Town

And larger still...

Dr Chris Town

Matlab demo...

```

im = imread('pat0.png');
%fourier reconstruction
freqcomp(im, 50);
freqcomp(im, 500, 0.01);

% freqcomp displays:
% * The image.
% * The Fourier transform (spectrum) of the image with a conjugate
%   pair of Fourier components marked with red dots.
% * The sine wave basis function that corresponds to the Fourier
%   transform pair marked in the image above.
% * The reconstruction of the image generated from the sum of the sine
%   wave basis functions considered so far.

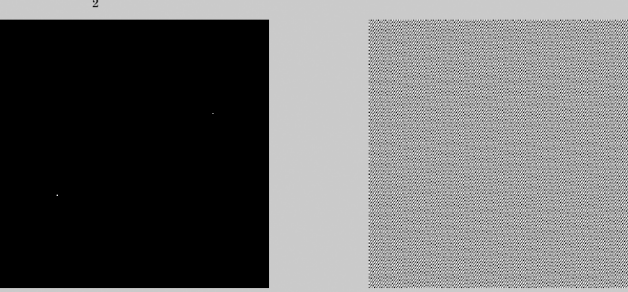
figure; inspect(im, 500);

%% INSPECT - Plots image amplitude spectrum averaged over all orientations.

```

Dr Chris Town

2



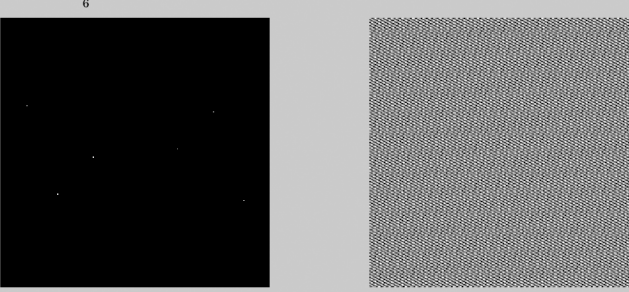
#1: Range [0, 1] Dims [256, 256]

#2: Range [0.000109, 0.0267] Dims [256, 256]

A.Torralba

Dr Chris Town

6



#1: Range [0, 1] Dims [256, 256]

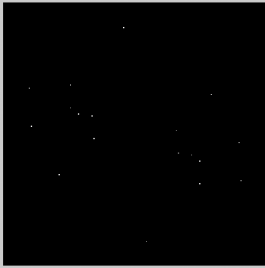
#2: Range [1.89e-007, 0.226] Dims [256, 256]

A.Torralba

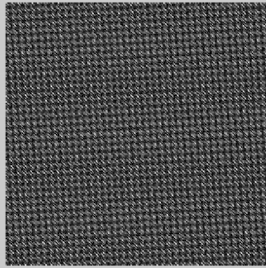
Dr Chris Town

18

18



#1: Range [0, 1]  
Dims [256, 256]



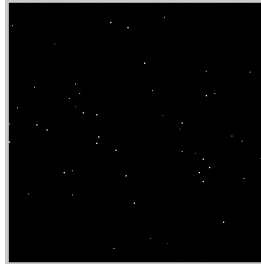
#2: Range [4.79e-007, 8.503]  
Dims [256, 256]

A.Torralba

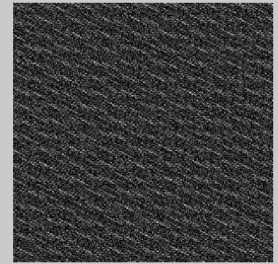
Dr Chris Town

50

50



#1: Range [0, 1]  
Dims [256, 256]



#2: Range [0.6e-005, 1.7]  
Dims [256, 256]

A.Torralba

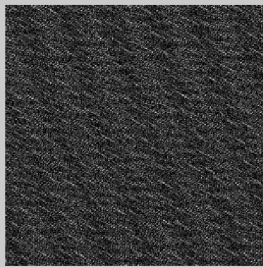
Dr Chris Town

82

82



#1: Range [0, 1]  
Dims [256, 256]



#2: Range [3.85e-007, 2.21]  
Dims [256, 256]

A.Torralba

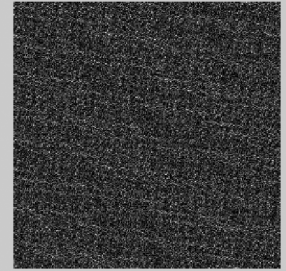
Dr Chris Town

136

136



#1: Range [0, 1]  
Dims [256, 256]



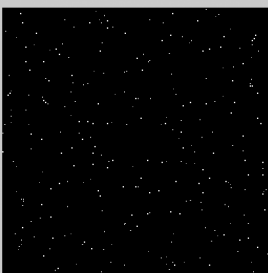
#2: Range [0.25e-006, 3.48]  
Dims [256, 256]

A.Torralba

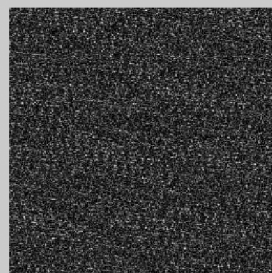
Dr Chris Town

282

282



#1: Range [0, 1]  
Dims [256, 256]



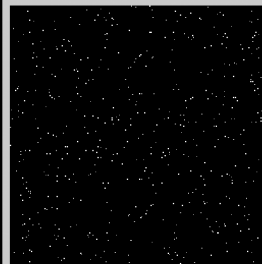
#2: Range [1.29e-005, 5.88]  
Dims [256, 256]

A.Torralba

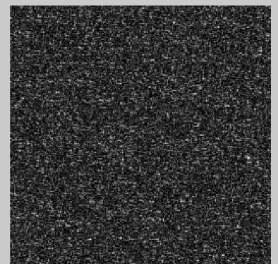
Dr Chris Town

538

538



#1: Range [0, 1]  
Dims [256, 256]



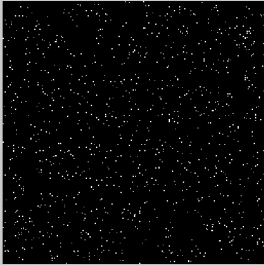
#2: Range [0.17e-005, 8.4]  
Dims [256, 256]

A.Torralba

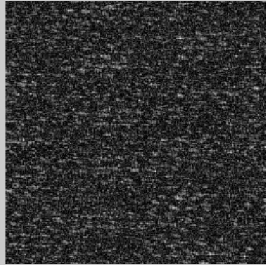
Dr Chris Town

1088

1088



#1: Range [0, 1]  
Dims [256, 256]



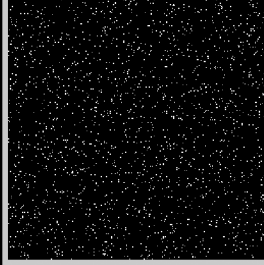
#2: Range [9.99e-005, 15]  
Dims [256, 256]

A.Torralba

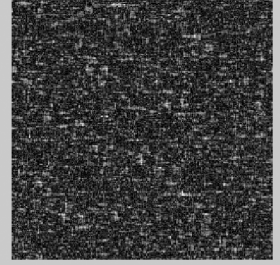
Dr Chris Town

2094

2094



#1: Range [0, 1]  
Dims [256, 256]



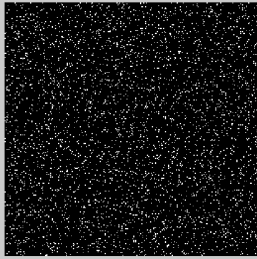
#2: Range [7e-005, 18]  
Dims [256, 256]

A.Torralba

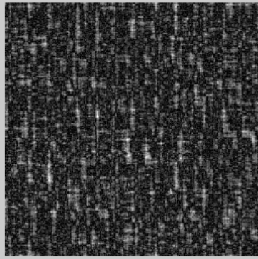
Dr Chris Town

4052.

4052



#1: Range [0, 1]  
Dims [256, 256]



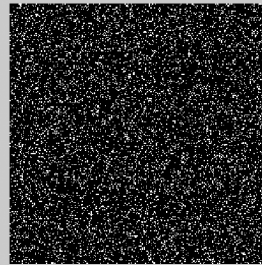
#2: Range [0.000558, 37.7]  
Dims [256, 256]

A.Torralba

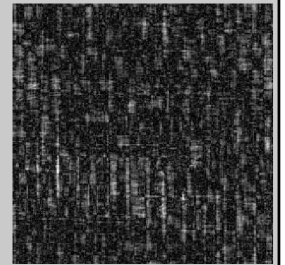
Dr Chris Town

8056.

8056



#1: Range [0, 1]  
Dims [256, 256]



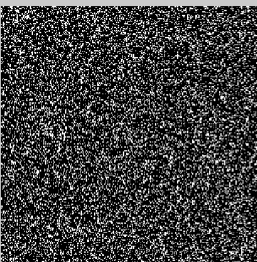
#2: Range [0.00032, 64.5]  
Dims [256, 256]

A.Torralba

Dr Chris Town

15366

15366



#1: Range [0, 1]  
Dims [256, 256]



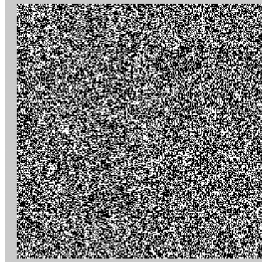
#2: Range [0.000231, 81.1]  
Dims [256, 256]

A.Torralba

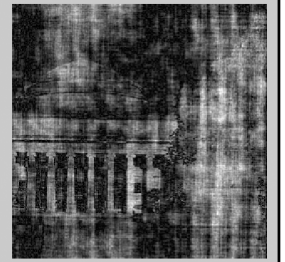
Dr Chris Town

28743

28743



#1: Range [0, 1]  
Dims [256, 256]

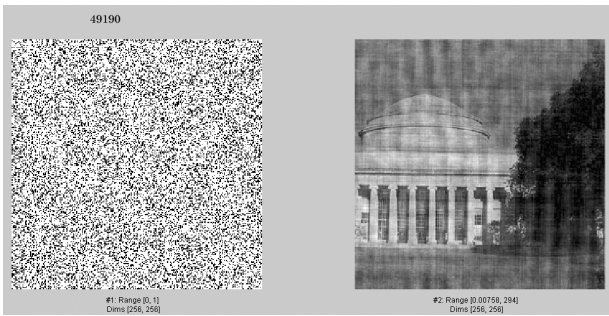


#2: Range [0.00109, 148]  
Dims [256, 256]

A.Torralba

Dr Chris Town

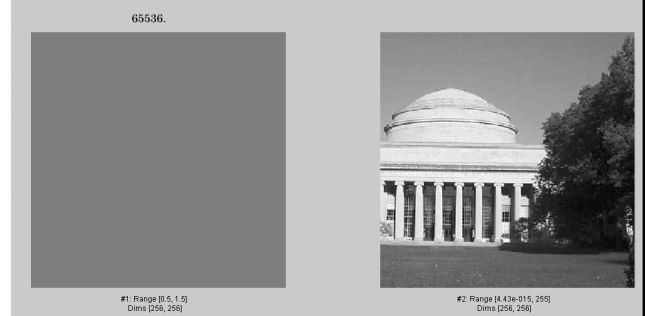
49190.



A.Torralba

Dr Chris Town

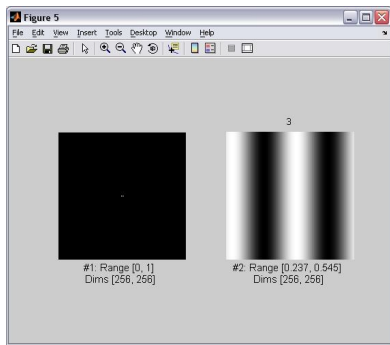
65536.



A.Torralba

Dr Chris Town

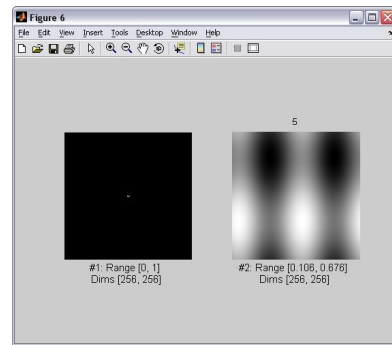
Now, an analogous sequence of images, but selecting Fourier components in descending order of magnitude.



A.Torralba

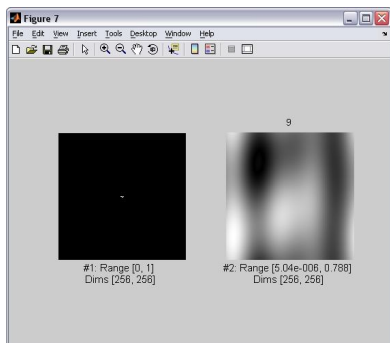
Dr Chris Town

5



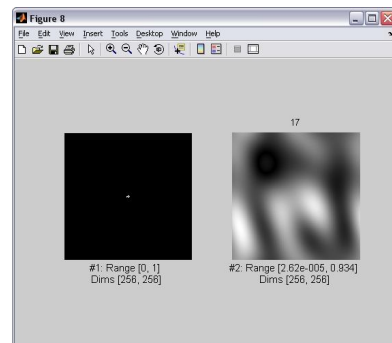
Dr Chris Town

9



Dr Chris Town

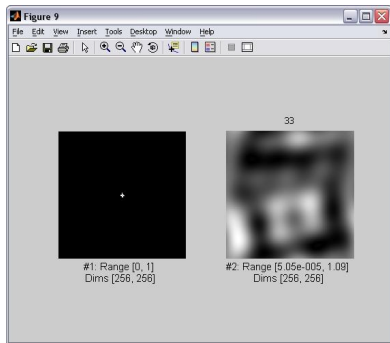
17



Dr Chris Town

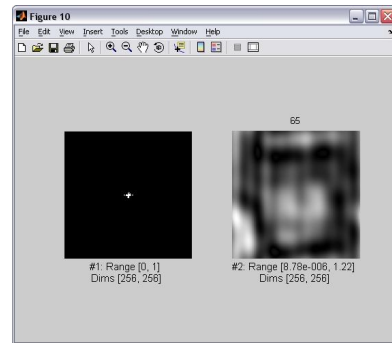


33



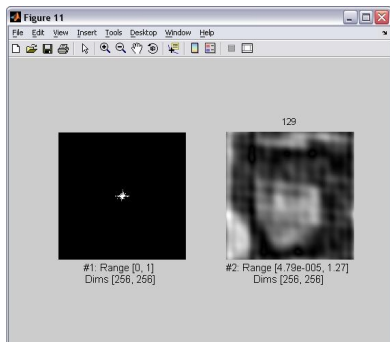
Dr Chris Town

65



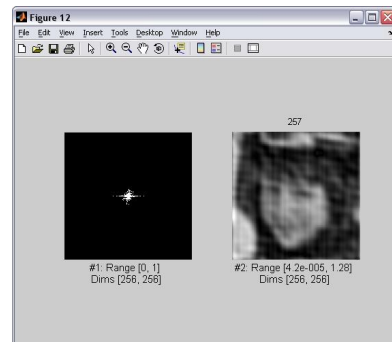
Dr Chris Town

129



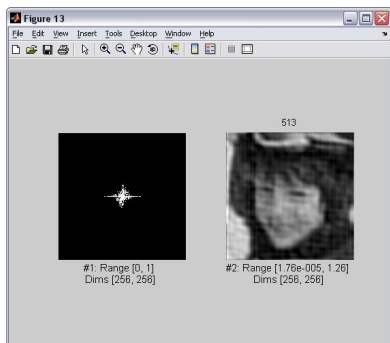
Dr Chris Town

257



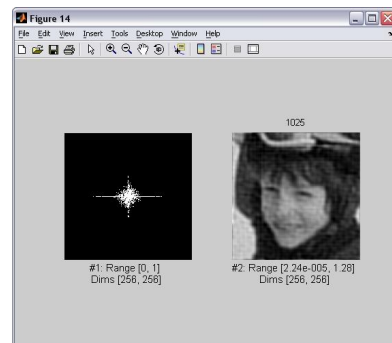
Dr Chris Town

513



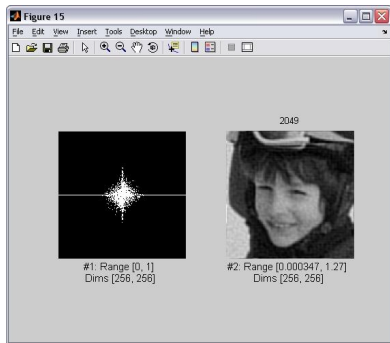
Dr Chris Town

1025



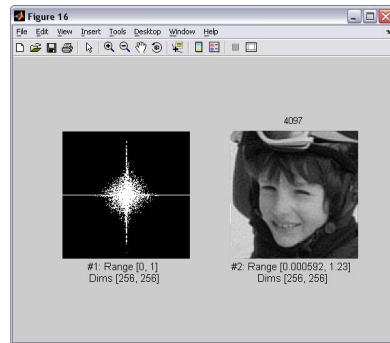
Dr Chris Town

2049



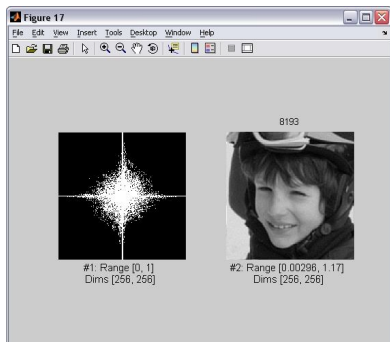
Dr Chris Town

4097



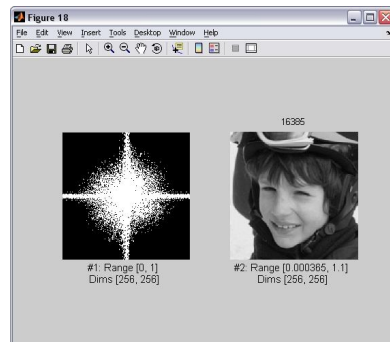
Dr Chris Town

8193



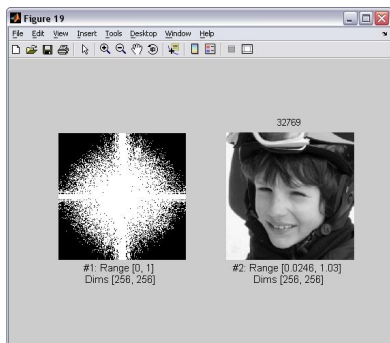
Dr Chris Town

16385



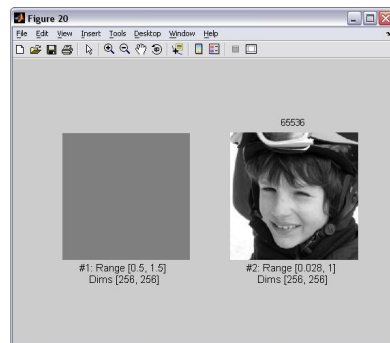
Dr Chris Town

32769



Dr Chris Town

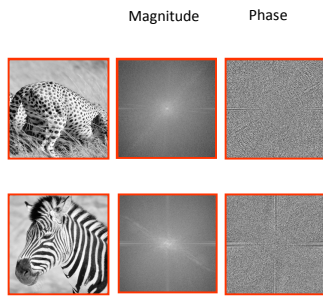
65536



Dr Chris Town

# Fourier Transform

- Fourier transform of a real function is complex
  - difficult to plot, visualize
  - instead, we can think of the phase and magnitude of the transform



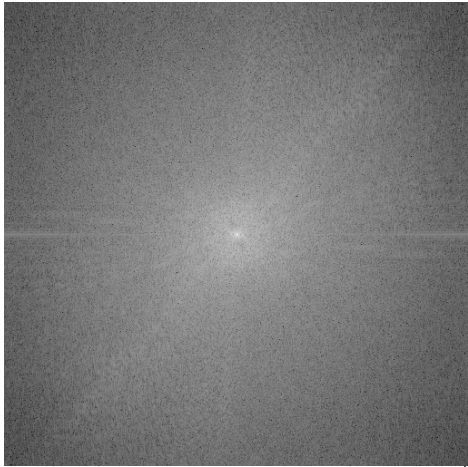
Computer Vision - A Modern Approach - Set: Pyramids and Texture - Slides by D.A. Forsyth  
Dr Chris Town



A.Torralba

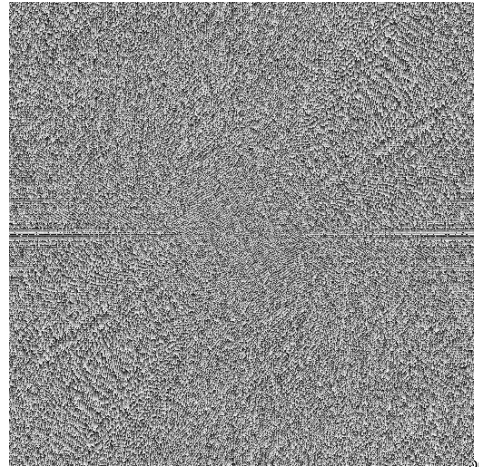
Dr Chris Town

This is the magnitude transform of the cheetah pic



Dr Chris Town

This is the phase transform of the cheetah pic

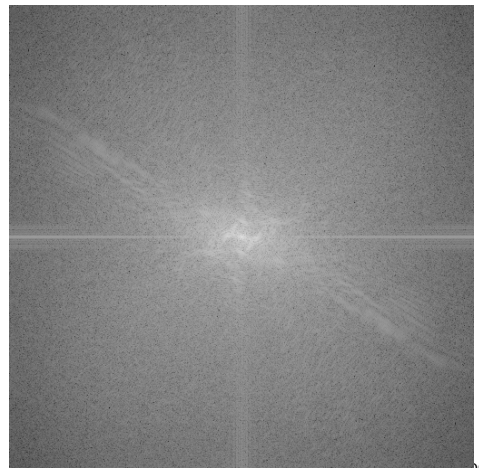


Dr Chris Town



Dr Chris Town

This is the magnitude transform of the zebra pic



Dr Chris Town

This is the phase transform of the zebra pic

Dr Chris Town

Reconstruction with zebra phase, cheetah magnitude

A.Torralba Dr Chris Town

Reconstruction with cheetah phase, zebra magnitude

A.Torralba Dr Chris Town

### Phase and Magnitude

Image with cheetah phase (and zebra magnitude)

Image with zebra phase (and cheetah magnitude)

Computer Vision - A Modern Approach - Set: Pyramids and Texture - Slides by D.A. Forsyth Dr Chris Town

### Randomizing the phase

A.Torralba Dr Chris Town

### A simple texture descriptor

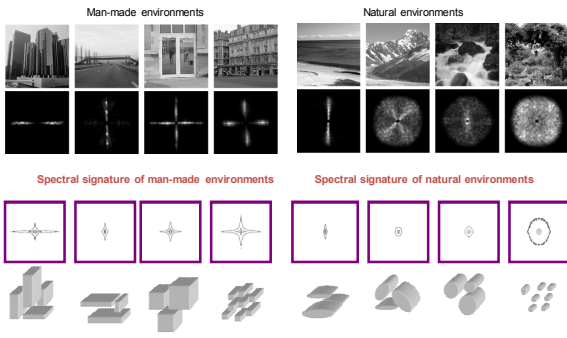
#### Magnitude of the Fourier Transform

$$A(f_x, f_y) = \left| \sum_{x,y} i(x,y) e^{-2\pi i(f_x x + f_y y)} \right|$$

Magnitude of the Fourier Transform encodes unlocalised information about dominant orientations and scales in the image.

A.Torralba Dr Chris Town

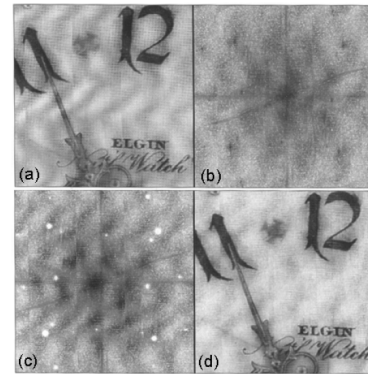
# Statistics of Scene Categories



Oliva et al (99), Oliva & Torralba (01)

Look at Mumford's work [on this slide](#)

# Fourier image enhancement



The Image Processing Handbook, chapter 4. CRC Press, 1992.

Dr Chris Town

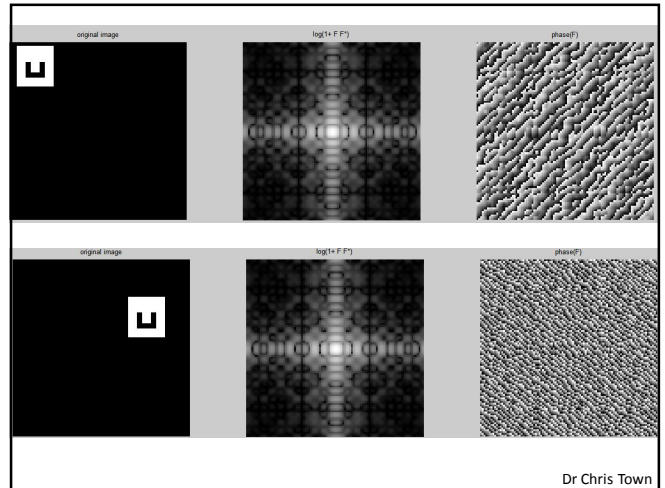
**Shift Theorem:** Shifting the original pattern in  $(x, y)$  by some 2D displacement  $(\alpha, \beta)$  merely multiplies its 2DFT by  $\exp(-i(\alpha\mu + \beta\nu))$ . Thus the 2DFT of the shifted pattern  $f(x - \alpha, y - \beta)$  is:  $F(\mu, \nu) \exp(-i(\alpha\mu + \beta\nu))$ .

**Practical Application:** The power spectrum of any isolated pattern is thus translation-invariant: it does not depend on where the pattern is located within the image, and so you don't have to find it first. The power spectrum is defined as the product of the pattern's 2DFT,  $F(\mu, \nu)$ , times its complex conjugate,  $F^*(\mu, \nu)$ , which just requires that the sign  $(-)$  of the imaginary part of  $F(\mu, \nu)$  gets reversed. You can easily see that the power spectrum of the shifted pattern  $f(x - \alpha, y - \beta)$ , namely:

$$\exp(-i(\alpha\mu + \beta\nu))F(\mu, \nu) \exp(i(\alpha\mu + \beta\nu))F^*(\mu, \nu)$$

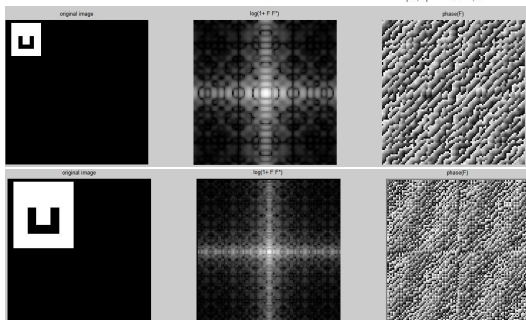
is equal to the power spectrum of the original unshifted pattern, namely:  $F(\mu, \nu)F^*(\mu, \nu)$ . Thus the power spectrum is translation-invariant.

Dr Chris Town



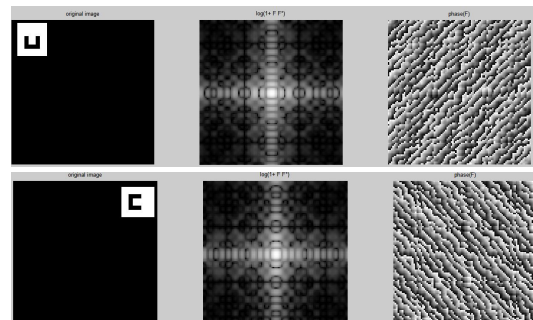
Dr Chris Town

**Similarity Theorem:** If the size of the original pattern  $f(x, y)$  changes (shrinks/expands), say by a factor  $\alpha$  in the  $x$ -direction, and by a factor  $\beta$  in the  $y$ -direction, becoming  $f(\alpha x, \beta y)$ , then the 2DFT of the pattern,  $F(\mu, \nu)$ , also changes (expands/shrinks) by the reciprocal of those factors and with similarly scaled amplitude. It becomes:  $\frac{1}{|\alpha\beta|}F(\frac{\mu}{\alpha}, \frac{\nu}{\beta})$ .



Dr Chris Town

**Rotation Theorem:** If the original pattern  $f(x, y)$  rotates through some angle  $\theta$ , becoming  $f(x \cos(\theta) + y \sin(\theta), -x \sin(\theta) + y \cos(\theta))$ , then its 2DFT  $F(\mu, \nu)$  also just rotates through the same angle. It becomes:  $F(\mu \cos(\theta) + \nu \sin(\theta), -\mu \sin(\theta) + \nu \cos(\theta))$ .



Dr Chris Town

**Practical Application:** Size- and orientation-invariant pattern representations can be constructed by these relationships. Specifically, if the Fourier domain  $(\mu, \nu)$  is now mapped into log-polar coordinates  $(r, \theta)$  where  $r = \log(\sqrt{\mu^2 + \nu^2})$  and  $\theta = \tan^{-1}(\nu/\mu)$ , then any dilation (size change) in the original pattern becomes simply a translation along the  $r$ -coordinate; and any rotation of the original pattern becomes simply a translation along the orthogonal  $\theta$ -coordinate in this log-polar Fourier domain. But we saw earlier that translations are made immaterial by taking a power spectrum, and so these effects of dilation and rotation of the pattern are eliminated in such a representation.

Combined with the translation-invariant property of the power spectrum, we now see how it becomes possible to represent patterns in a manner that is independent of their position in the image, their orientation, and their size (i.e. the Poincaré group of transformations).

Dr Chris Town

**Convolution Theorem:** Let function  $f(x, y)$  have 2DFT  $F(\mu, \nu)$ , and let function  $g(x, y)$  have 2DFT  $G(\mu, \nu)$ . The convolution of  $f(x, y)$  with  $g(x, y)$ , which is denoted  $f * g$ , combines these two functions to generate a third function  $h(x, y)$ , whose value at location  $(x, y)$  is equal to the integral of the product of functions  $f$  and  $g$  after one is flipped and undergoes a relative shift by amount  $(x, y)$ :

$$h(x, y) = \int_{\alpha} \int_{\beta} f(\alpha, \beta)g(x - \alpha, y - \beta)d\alpha d\beta \quad (9)$$

The Convolution Theorem states that convolving two functions  $f(x, y)$  and  $g(x, y)$  together in the image domain, simply multiplies their two 2DFT's together in the 2D Fourier domain:

$$H(\mu, \nu) = F(\mu, \nu)G(\mu, \nu) \quad (10)$$

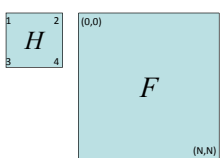
where  $H(\mu, \nu)$  is the 2DFT of the desired result  $h(x, y)$ .

Dr Chris Town

### Correlation Filtering

$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v]F[i + u, j + v]$$

- This is called cross-correlation, denoted  $G = H \otimes F$
- Filtering an image
  - Replace each pixel by a weighted combination of its neighbors.
  - The filter "kernel" or "mask" is the prescription for the weights in the linear combination.

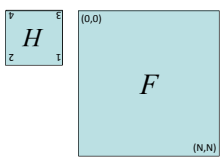


Dr Chris Town

### Convolution

$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v]F[i - u, j - v]$$

- Convolution:**
  - Flip the filter in both dimensions (bottom to top, right to left)
  - Then apply cross-correlation

$$G = H \star F$$


Dr Chris Town

### Convolution vs. Correlation

- Correlation**

$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v]F[i + u, j + v]$$

Matlab: filter2

$$G = H \otimes F$$
- Convolution**

$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v]F[i - u, j - v]$$

Matlab: conv2

$$G = H \star F$$
- Note**
  - If  $H[-u, -v] = H[u, v]$ , then correlation  $\equiv$  convolution.

Note the difference!

Dr Chris Town

### Shift Invariant Linear System

- Shift invariant:**
  - Operator behaves the same everywhere, i.e. the value of the output depends on the pattern in the image neighborhood, not the position of the neighborhood.
- Linear:**
  - Superposition:  $h * (f1 + f2) = (h * f1) + (h * f2)$
  - Scaling:  $h * (kf) = k (h * f)$

Dr Chris Town

## Properties of convolution

- Linear & shift invariant
- Commutative:  
 $f * g = g * f$
- Associative  
 $(f * g) * h = f * (g * h)$
- Identity:  
unit impulse  $e = [\dots, 0, 0, 1, 0, 0, \dots]$ .  $f * e = f$
- Differentiation:  
 $\frac{\partial}{\partial x}(f * g) = \frac{\partial f}{\partial x} * g$

Dr Chris Town

$$\text{result}(i, j) = \sum_m \sum_n \text{kernel}(m, n) \cdot \text{image}(i - m, j - n)$$

```
int i, j, m, n, sum, image[iend][jend],
    kernel[mend][nend], result [iend][jend];

for (i = mend; i < iend; i++) {
  for (j = nend; j < jend; j++) {
    sum = 0;
    for ( m = 0; m < mend; m++) {
      for ( n = 0; n < nend; n++) {
        sum += kernel[m][n] * image[i-m][j-n];
      }
    }
    result[i][j] = sum/(mend*nend);
  }
}
```

Dr Chris Town

**Differentiation Theorem:** Computing the derivatives of an image  $f(x, y)$  is equivalent to multiplying its 2DFT,  $F(\mu, \nu)$ , by the corresponding frequency coordinate raised to a power equal to the order of differentiation:

$$\left(\frac{\partial}{\partial x}\right)^m \left(\frac{\partial}{\partial y}\right)^n f(x, y) \xrightarrow{2DFT} (i\mu)^m (i\nu)^n F(\mu, \nu) \quad (11)$$

A particularly useful implication of this theorem is that isotropic differentiation, which treats all directions equally (for which the lowest possible order of differentiation is 2nd-order, known as the Laplacian operator  $\nabla^2$ ) is equivalent simply to multiplying the 2DFT of the image by a paraboloid:

$$\nabla^2 f(x, y) \equiv \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) f(x, y) \xrightarrow{2DFT} -(\mu^2 + \nu^2)F(\mu, \nu) \quad (12)$$

**Practical Application: Multi-Resolution Edge Detection.**

Dr Chris Town

For kernels smaller or equal to 5x5, FFT followed by multiplication is generally faster than explicit convolution

Dr Chris Town