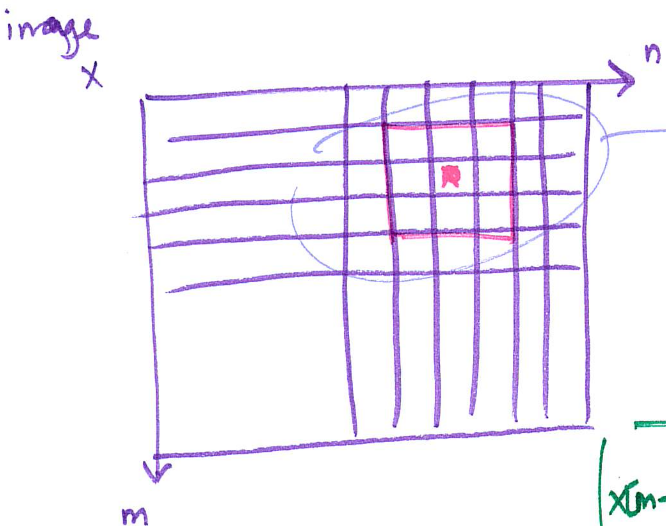
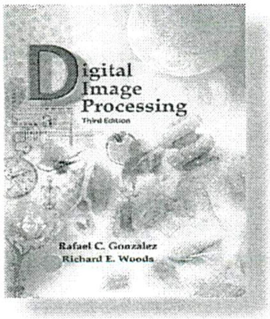
Lecture 23: Image Filtering and Compression.



h[-1,-1]	h[-1, 0]	h[-1, 1]
h[0,-1]	h[0,0]	h[0, 1]
h[1,-1]	h[1, 0]	h[1,1]

x[m-1,n-1]	x[m-1,n]	x[m-1,n+1]
x[m,n-1]	x[m,n]	x[m,n+1]
x[m+1,n-1]	x[m+1, n]	x[m+1,n+1]

$$y[m,n] = \sum_{s=-1}^{1} \sum_{t=-1}^{1} h[s,t]\, x[m-s,n-t]$$



Chapter 3
Intensity Transformations & Spatial Filtering

$X[m,n] = \delta[m]\,\delta[n]$

Origin $x[m,n]$

$$\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&0&1&0&0\\0&0&0&0&0\\0&0&0&0&0\end{matrix}\qquad h[m,n]\ \begin{matrix}1&2&3\\4&5&6\\7&8&9\end{matrix}$$

(a)

Padded X

$$\begin{matrix}0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\end{matrix}$$

(b)

Initial position for h

$$\begin{matrix}1&2&3&0&0&0&0&0&0\\4&5&6&0&0&0&0&0&0\\7&8&9&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\end{matrix}$$

(c)

Full correlation result

$$\begin{matrix}0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&9&8&7&0&0&0\\0&0&0&6&5&4&0&0&0\\0&0&0&3&2&1&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\end{matrix}$$

(d)

Cropped correlation result

$$\begin{matrix}0&0&0&0&0\\0&9&8&7&0\\0&6&5&4&0\\0&3&2&1&0\\0&0&0&0&0\end{matrix}$$

(e)

$\delta[m,n] = \delta[m]\cdot\delta[n]$

$\delta[m,n] * h[m,n] = h[m,n]$

Rotated h

$$\begin{matrix}9&8&7&0&0&0&0&0&0\\6&5&4&0&0&0&0&0&0\\3&2&1&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\end{matrix}$$

(f)

Full convolution result

$$\begin{matrix}0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&1&2&3&0&0&0\\0&0&0&4&5&6&0&0&0\\0&0&0&7&8&9&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0\end{matrix}$$

(g)

Cropped convolution result

$$\begin{matrix}0&0&0&0&0\\0&1&2&3&0\\0&4&5&6&0\\0&7&8&9&0\\0&0&0&0&0\end{matrix}$$

(h)

FIGURE 3.30 Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.

2

Recall

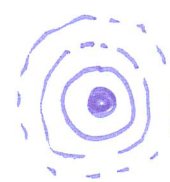
$$Y[k,l] = X[k,l] \times H[k,l]$$

$$\Rightarrow \texttt{ifft2}(Y) \rightarrow y$$

more precisely, to compute 2d convolution $x[m,n] * h[m,n]$:

```
1.  X = fft2(x);
2.  H = fft2(h);
3.  Y = X .* H;
4.  y = ifft2(Y);
```

Locations of large elements of H tells us what kinds of image data will be dampened or emphasized.



Transform coding (idea behind JPEG)
MP3
MPEG

Encoder

signal x → transform → encode large coefficients → bits

Decoder

bits → decode coeffs → inverse transform → image.

This gives great compression if x well-represented by small # of coefficients.

Ex: JPEG. transform = · divide image into 8x8 blocks
· compute Discrete Cosine Transform of each block

DCT is like DFT but does not have complex values:

$$x[m,n] = \frac{1}{MN}\Big[$$

DCT:

$$x[m,n] = \frac{1}{MN} \sum_{k} \sum_{l} \tilde{X}[k,l] \cos\left(\frac{\pi}{2M} k(2m+1)\right) \cos\left(\frac{\pi}{2N} l(2n+1)\right)$$

where

$$\tilde{X}[k,l] = w[k]\,w[l] \sum_{m} \sum_{n} x[m,n] \cos\left(\frac{\pi}{2M} k(2m+1)\right) \cos\left(\frac{\pi}{2N} l(2n+1)\right)$$

$$w[k] = \begin{cases} 1 & \text{if } k=0 \\ 2 & \cancel{k=1,\ldots,}\ k>0 \end{cases}$$

JPEG summary

1. divide image into 8x8 blocks
2. DCT of each block
3. Quantize each coefficient to integer value ~~that can be stored as bits~~ (small coefficients → 0)

~~4. Sm Store quantized values as bits using codes that give more bits to lower frequency~~

divide each coefficient by normalization factor (n. f.), then round to nearest integer.

have smaller n. f. for lower frequencies

⟹ more accuracy for low freq coeffs, and higher freq coeffs more likely to be rounded to zeros.