Image compression and Fourier analysis

Vlad Shakhuro

2 October 2025

Outline

1. Lossless compression with PNG

2. Lossy compression with JPEG

3. Image analysis using Fourier transform

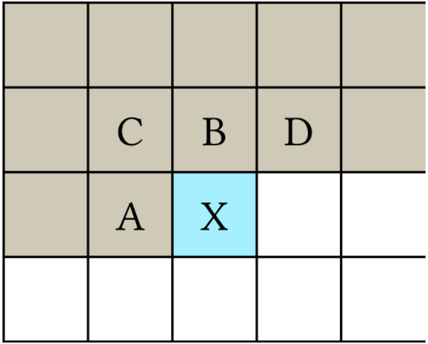
4. Image stitching with laplacian pyramid

Lossless compression with PNG

PNG — free alternative to GIF, proposed in 1995.

1. Supports only lossless compression.
2. RGBA, up to 16 bit per channel, supports palletes.
3. Simple cross-platform implementation.

Delta-coding



Coding methods:

0 Byte X is passed unaltered

1 Code $X - A$

2 Code $X - B$

3 Code $X - \lfloor \frac{A+B}{2} \rfloor$

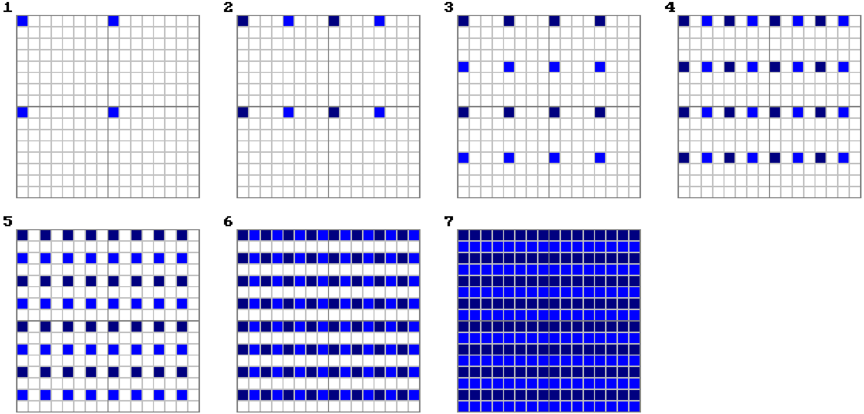
4 Code $X - F$, $F \in \{A, B, C\}$, nearest to $p = A + B - C$

Coding method is chosen per line

Obtained sequence of numbers is compressed using DEFLATE algorithm (used in ZIP)

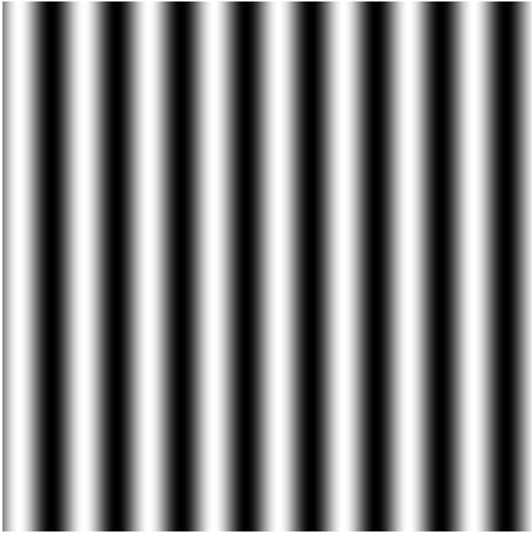
Image traversal (Adam7)

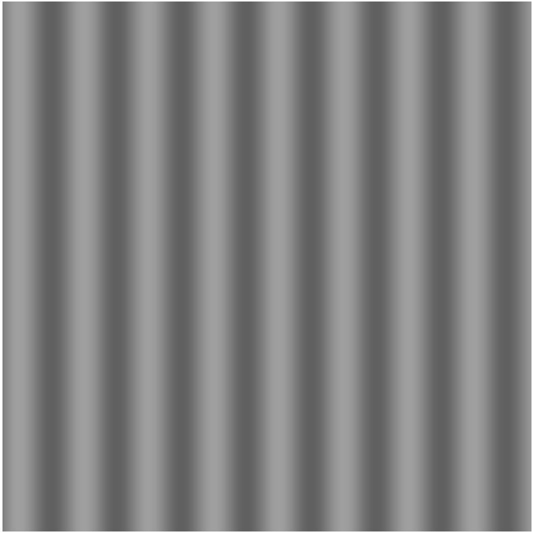
PNG can optionally use traversal in 7 steps for partial transmission of an image

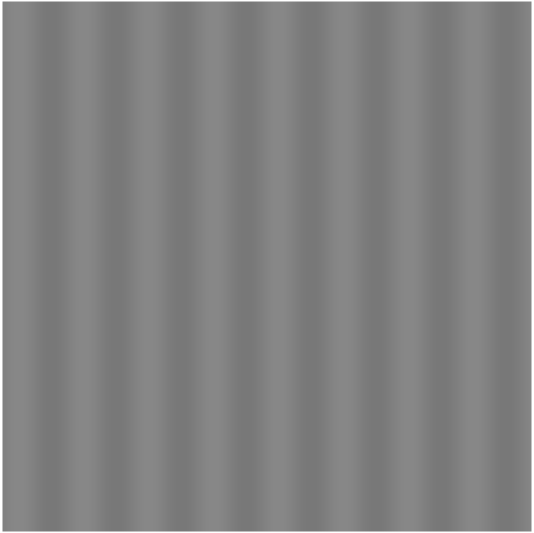


Outline

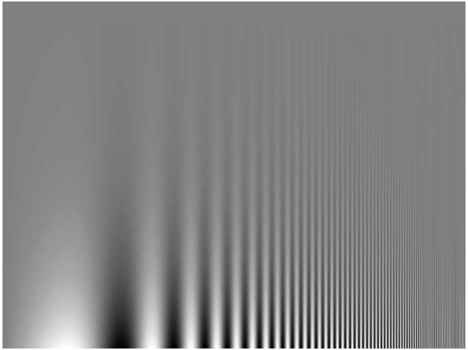
1. Lossless compression with PNG
2. Lossy compression with JPEG
3. Image analysis using Fourier transform
4. Image stitching with laplacian pyramid

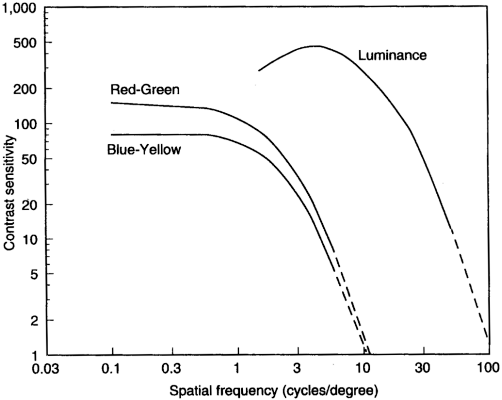






Contrast Sensitivity Function





Human perceive luminance and color with different resolutions. To make use of this fact, image transmission systems encode separate luminance component

Human sensitivity to luminance and color

RGB

luminance

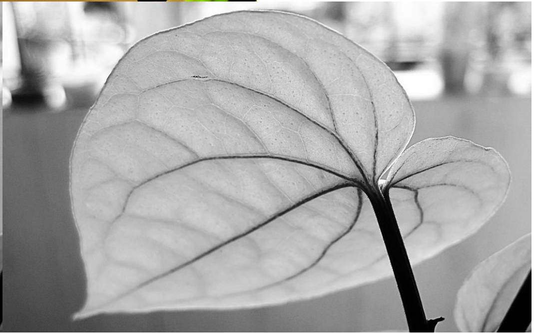
color

Human sensitivity to luminance and color

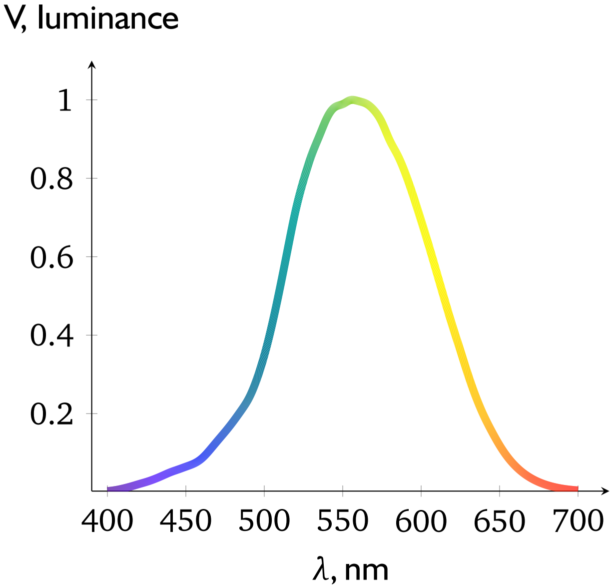
RGB

blurred color

blurred luminance



Photopic luminosity function



$$RGB \rightarrow YC_bC_r$$

$$\begin{aligned}
Y &= 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \\
C_b &= -0.169 \cdot R - 0.331 \cdot G + 0.500 \cdot B + 128 \\
C_r &= 0.500 \cdot R - 0.419 \cdot G - 0.081 \cdot B + 128
\end{aligned}$$

$$RGB \rightarrow YC_bC_r$$

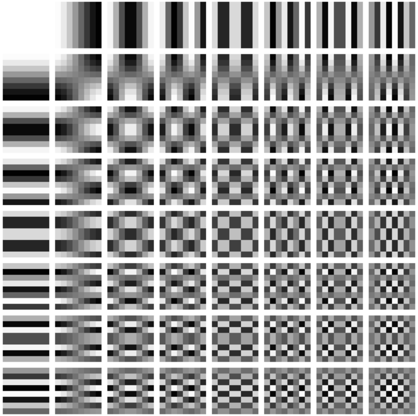
$$\begin{aligned}
Y &= 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \\
C_b &= -0.169 \cdot R - 0.331 \cdot G + 0.500 \cdot B + 128 \\
C_r &= 0.500 \cdot R - 0.419 \cdot G - 0.081 \cdot B + 128
\end{aligned}$$

Shift to 128

$$\begin{bmatrix} 139 & 144 & 149 & 153 & 155 & 155 & 155 & 155 \\ 144 & 151 & 153 & 156 & 159 & 156 & 156 & 156 \\ 150 & 155 & 160 & 163 & 158 & 156 & 156 & 156 \\ 159 & 161 & 162 & 160 & 160 & 159 & 159 & 159 \\ 159 & 160 & 161 & 162 & 162 & 155 & 155 & 155 \\ 161 & 161 & 161 & 161 & 160 & 157 & 157 & 157 \\ 162 & 162 & 161 & 163 & 162 & 157 & 157 & 157 \\ 162 & 162 & 162 & 161 & 163 & 158 & 158 & 158 \end{bmatrix} \begin{bmatrix} 11 & 16 & 21 & 25 & 27 & 27 & 27 & 27 \\ 16 & 23 & 25 & 28 & 31 & 28 & 28 & 28 \\ 22 & 27 & 32 & 35 & 30 & 28 & 28 & 28 \\ 31 & 33 & 34 & 32 & 32 & 31 & 31 & 31 \\ 31 & 32 & 33 & 34 & 34 & 27 & 27 & 27 \\ 33 & 33 & 33 & 33 & 32 & 29 & 29 & 29 \\ 34 & 34 & 33 & 35 & 34 & 29 & 29 & 29 \\ 34 & 34 & 34 & 33 & 35 & 30 & 30 & 30 \end{bmatrix}$$

source image　　　　　shifted image

Discrete cosine transform



$$
\begin{array}{cc}
\xrightarrow{x} & \\
\begin{bmatrix}
11 & 16 & 21 & 25 & 27 & 27 & 27 & 27 \\
16 & 23 & 25 & 28 & 31 & 28 & 28 & 28 \\
22 & 27 & 32 & 35 & 30 & 28 & 28 & 28 \\
31 & 33 & 34 & 32 & 32 & 31 & 31 & 31 \\
31 & 32 & 33 & 34 & 34 & 27 & 27 & 27 \\
33 & 33 & 33 & 33 & 32 & 29 & 29 & 29 \\
34 & 34 & 33 & 35 & 34 & 29 & 29 & 29 \\
34 & 34 & 34 & 33 & 35 & 30 & 30 & 30
\end{bmatrix} & \Big\Downarrow y
\end{array}
$$

$$
\begin{array}{cc}
\xrightarrow{u} & \\
\begin{bmatrix}
235.6 & -1.0 & -12.1 & -5.2 & 2.1 & -1.7 & -2.7 & 1.3 \\
-22.6 & -17.5 & -6.2 & -3.2 & -2.9 & -0.1 & 0.4 & -1.2 \\
-10.9 & -9.3 & -1.6 & 1.5 & 0.2 & -0.9 & -0.6 & -0.1 \\
-7.1 & -1.9 & 0.2 & 1.5 & 0.9 & -0.1 & 0.0 & 0.3 \\
-0.6 & -0.8 & 1.5 & 1.6 & -0.1 & -0.7 & 0.6 & 1.3 \\
1.8 & -0.2 & 1.6 & -0.3 & -0.8 & 1.5 & 1.0 & -1.0 \\
-1.3 & -0.4 & -0.3 & -1.5 & -0.5 & 1.7 & 1.1 & -0.8 \\
-2.6 & 1.6 & -3.8 & -1.8 & 1.9 & 1,2 & -0.6 & -0.4
\end{bmatrix} & \Big\Downarrow v
\end{array}
$$

amplitudes

Amplitude quantization

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

luminance quantization matrix

$$\begin{bmatrix} 15 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

quantized amplitudes

Quantization matrices

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \qquad \begin{bmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{bmatrix}$$

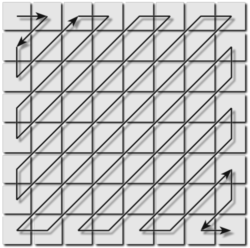
luminance quantization matrix   color quantization matrix

Quantization matrix depends on quality factor Q (from 1 to 100) set by user. Quantization matrices are constructed using:

1. Amplitude statistics of real images.
2. Model for human visual system.

Zigzag scanning

$$\begin{bmatrix} 15 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



15 0 -2 -1 -1 -1 0 0 -1 0 …0

Sequence of numbers is compressed using Zero Run-Length Encoding and common compression method (Huffman or arithmetic coding)

JPEG algorithm

1. Convert image in color space YC_bC_r.
2. Downsample color components.
3. Divide image into blocks of size 8×8 pixels, subtract 128.
4. For every block:
   4.1 Apply discrete cosine transform.
   4.2 Quantize coefficients (amplitudes).
   4.3 Compress quantized coefficients.

Algorithm specificity:

1. Designed for compressing images that are shown on a screen.
2. Images are compressed due to loss of high-frequency information.

Outline

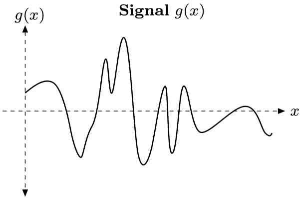
1. Lossless compression with PNG
2. Lossy compression with JPEG
3. Image analysis using Fourier transform
4. Image stitching with laplacian pyramid

Fourier transform

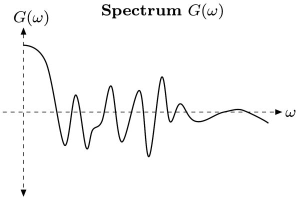
$$\underbrace{G(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} g(x) e^{-i\omega x} dx}_{\text{forward Fourier transform, } \mathscr{F}}$$

$$\underbrace{g(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} G(\omega) e^{i\omega x} d\omega}_{\text{inverse Fourier transform, } \mathscr{F}^{-1}}$$

Function and spectrum properties

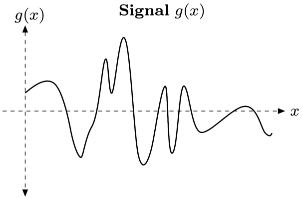


(a) Continuous nonperiodic signal.

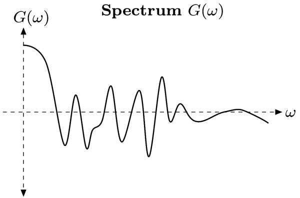


(b) Continuous nonperiodic spectrum.

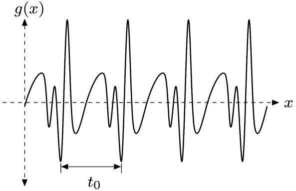
Function and spectrum properties



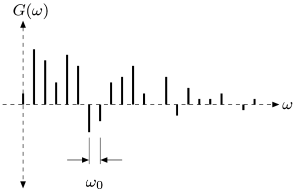
(a) Continuous nonperiodic signal.



(b) Continuous nonperiodic spectrum.

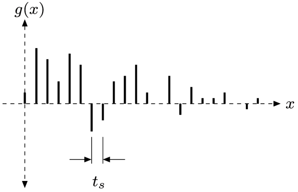


(c) Continuous periodic signal with period t_0.

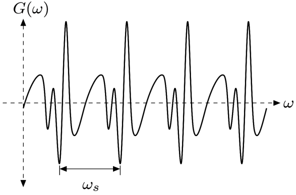


(d) Discrete nonperiodic spectrum with values spaced at $\omega_0 = 2\pi/t_0$.

Function and spectrum properties

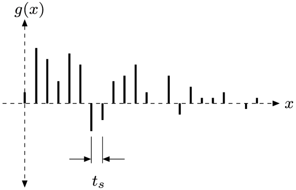


(e) Discrete nonperiodic signal with samples spaced at t_s.

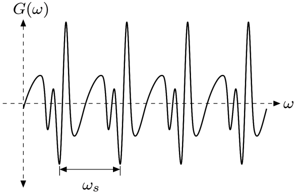


(f) Continuous periodic spectrum with period $\omega_s = 2\pi/t_s$.

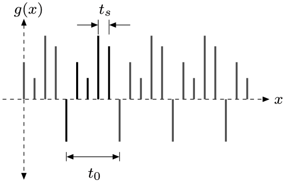
Function and spectrum properties



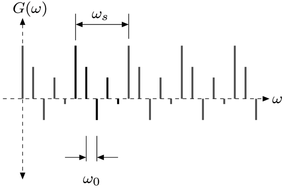
(e) Discrete nonperiodic signal with samples spaced at t_s.



(f) Continuous periodic spectrum with period $\omega_s = 2\pi/t_s$.



(g) Discrete periodic signal with samples spaced at t_s and period $t_0 = t_s M$.



(h) Discrete periodic spectrum with values spaced at $\omega_0 = 2\pi/t_0$ and period $\omega_s = 2\pi/t_s = \omega_0 M$.

Discrete Fourier transform

$$G(m) = \frac{1}{\sqrt{M}} \sum_{u=0}^{M-1} g(u) e^{-i2\pi \frac{mu}{M}}, \quad 0 \leqslant m < M$$

$$g(u) = \frac{1}{\sqrt{M}} \sum_{m=0}^{M-1} G(m) e^{i2\pi \frac{mu}{M}}, \quad 0 \leqslant u < M$$

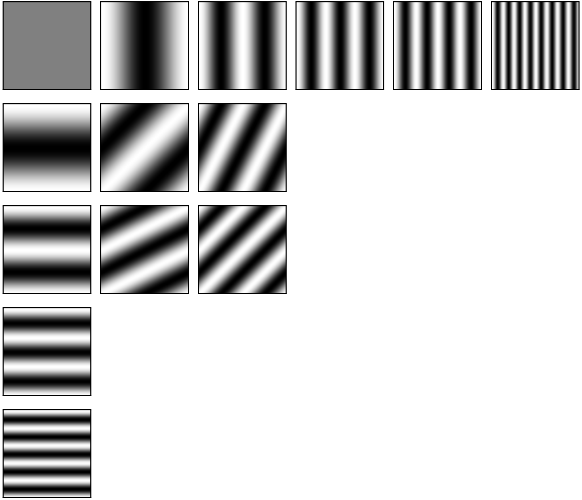
Discrete Fourier transform

$$G(m) = \frac{1}{\sqrt{M}} \sum_{u=0}^{M-1} g(u) e^{-i2\pi \frac{mu}{M}}, \quad 0 \leqslant m < M$$

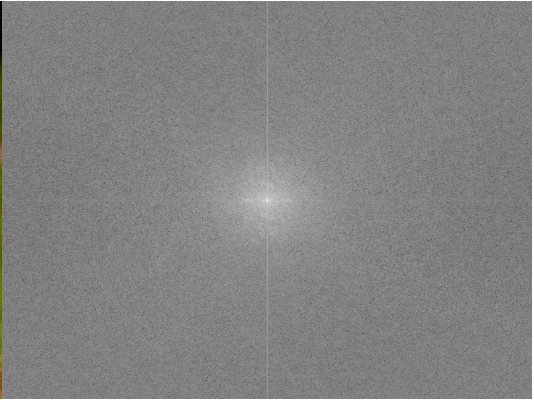
$$g(u) = \frac{1}{\sqrt{M}} \sum_{m=0}^{M-1} G(m) e^{i2\pi \frac{mu}{M}}, \quad 0 \leqslant u < M$$

Naive implementation: $O(M^2)$
Fast implementation: $O(M \log M)$

Fourier basis for images

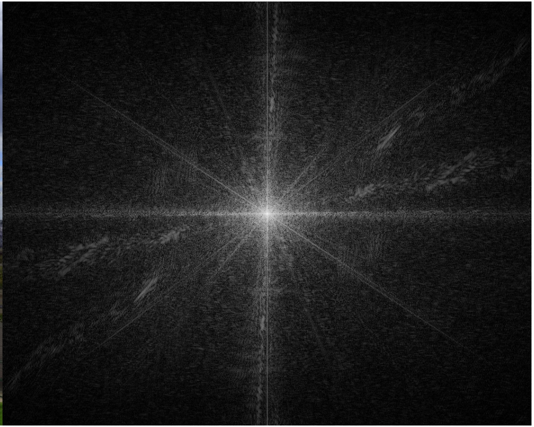


spatial domain



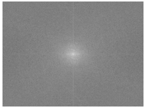
frequency domain

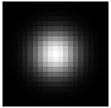
```
freq = log(1 + abs(fftshift(fft2(img))))
```

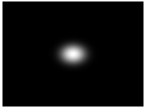
Convolution theorem

$\downarrow \mathscr{F}$



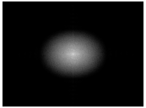


$\downarrow \mathscr{F}$



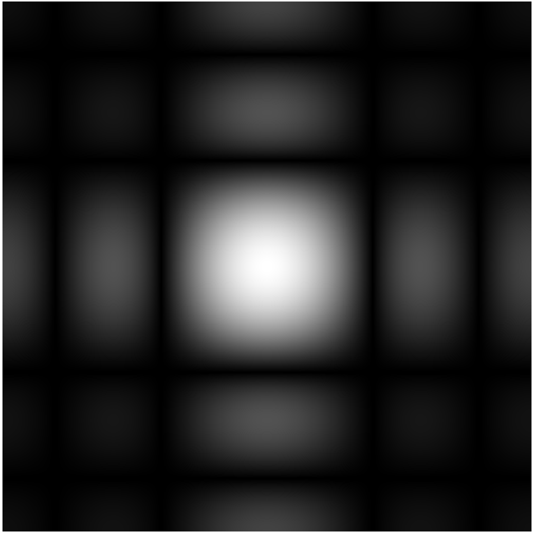
$*$

$\downarrow$



$\downarrow \mathscr{F}^{-1}$

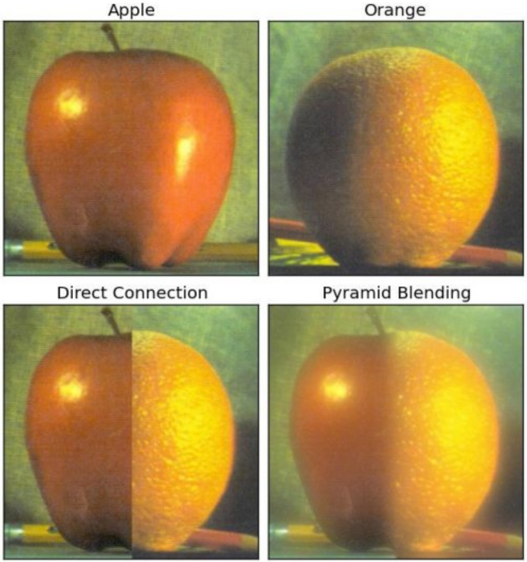
Fourier image of box filter



Outline

1. Lossless compression with PNG
2. Lossy compression with JPEG
3. Image analysis using Fourier transform
4. Image stitching with laplacian pyramid

Image stitching



Laplacian pyramid

Input:

Images A, B, mask M

Stitching algorithm:

1. Compute laplacian pyramids LA and LB for images A и B
2. Compute gaussian pyramid GM
3. Combine laplacian pyramids:
   $LS = GM * LA + (1 - GM) * LB$
4. Compute result image from pyramid LS

Conclusion

We reviewed following topics:

- PNG lossless image compression. It uses simple delta coding and DEFLATE compression for RGBA pixel values
- JPEG lossy image compression. It is based on discrete cosine transform, quantization and compression of amplitudes
- Fourier transform — a useful tool for analysis of images and linear filters
- image stitching via laplacian pyramid. We may justify its' applicability using Fourier analysis