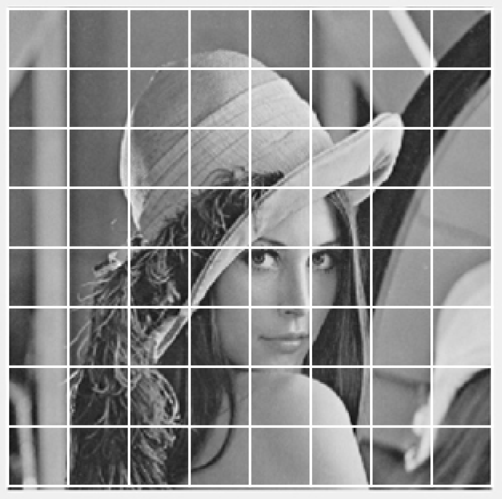
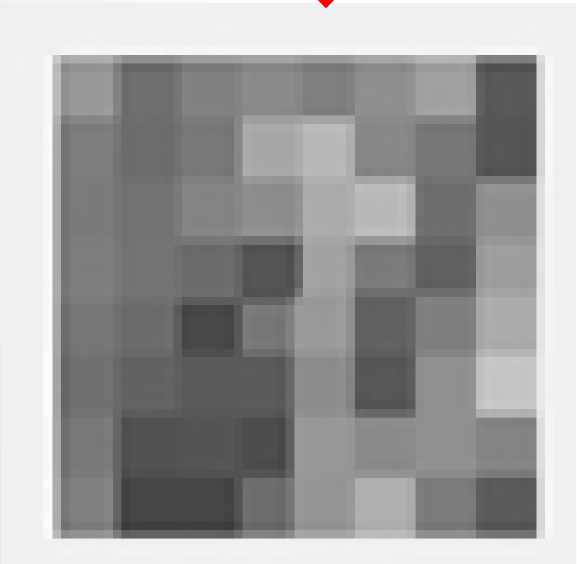
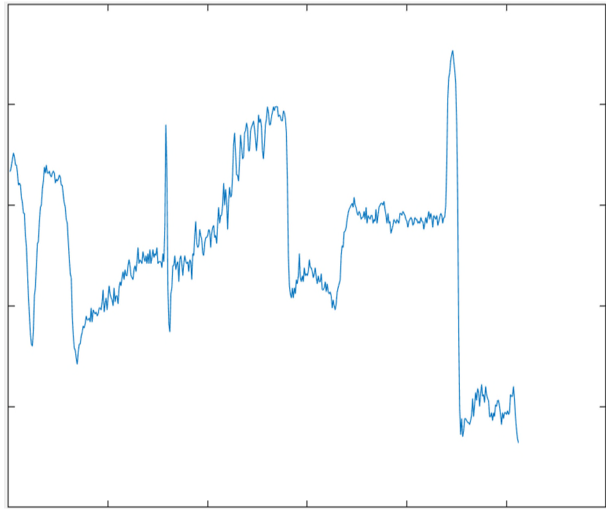
Image raw format

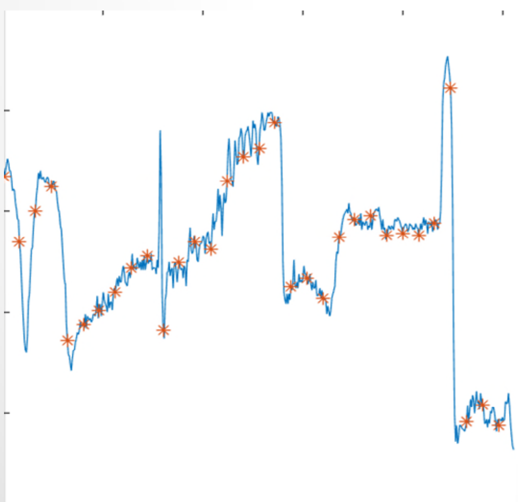




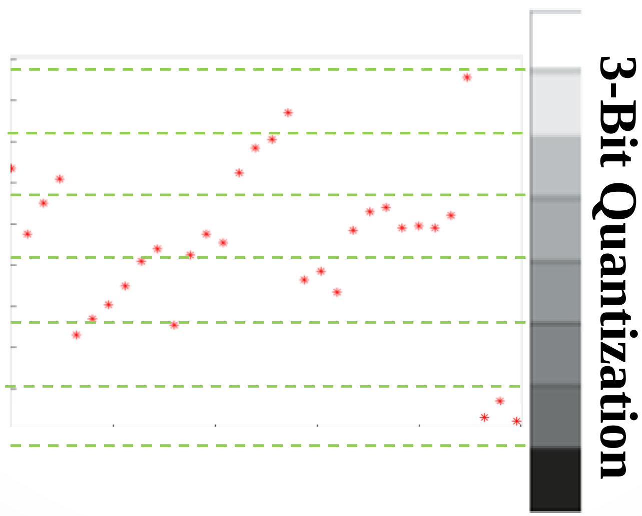
151	111	129	138	127	143	160	89
125	108	120	167	182	136	118	86
122	114	132	141	171	184	110	143
125	115	107	85	161	122	97	157
118	107	73	118	153	97	126	171
112	98	90	91	140	87	144	196
120	81	84	77	151	138	144	132
128	71	70	108	149	175	123	91

Sampling





Sampling



Top-left: Image.

Top-right: A scan line showing intensity variations along red line in the image.

Bottom-left: Apply sampling on X axis.

Bottom-right: Apply quantization on Y axis.

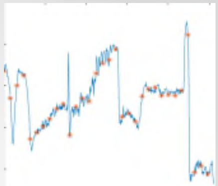
How about Nyquist frequency?

= the highest waveform frequency

Sampling rate
= 2 x highest frequency

Aliasing distortion?

Sampling effect



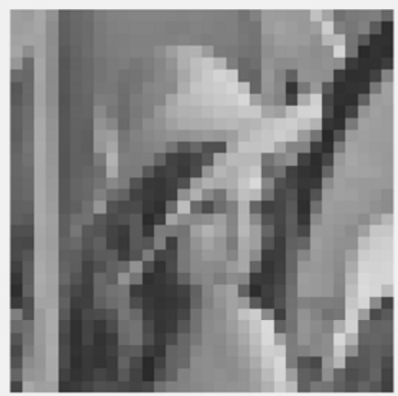
Sampling

512x512

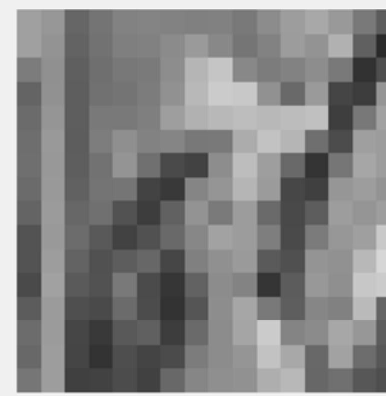
256x256

128x128

64x64



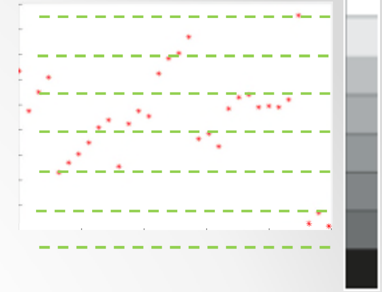
32x32



16x16

Sampling at certain resolutions

Sampling – Quantization Effect



8-bit (0-255)

7-bit (0-127)

6-bit (0-63)

5-bit (0-31)

4-bit (0-15)

3-bit (0-7)

2-bit (0-3)

1-bit (0-1)

Image Quantization

Image sensor

- Pixel is square because squares concatenate together without leaving gaps.
- Each pixel has two Green (G) sensors and one Red (R) or one Blue (B) to mimic the physiology of the human eye. The luminance perception of the human retina during daylight vision are most sensitive to green light.

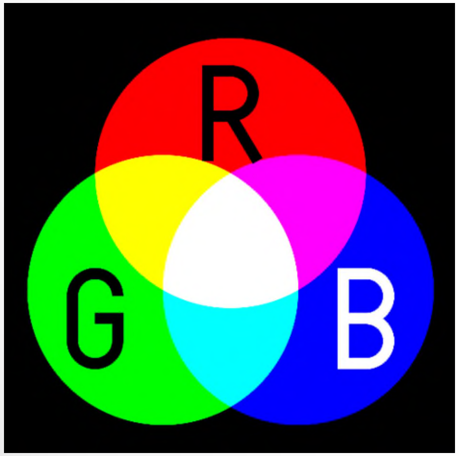
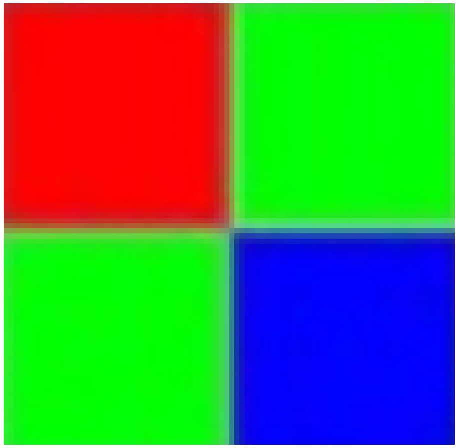


Image is obtained from
https://en.wikipedia.org/wiki/Color_theory



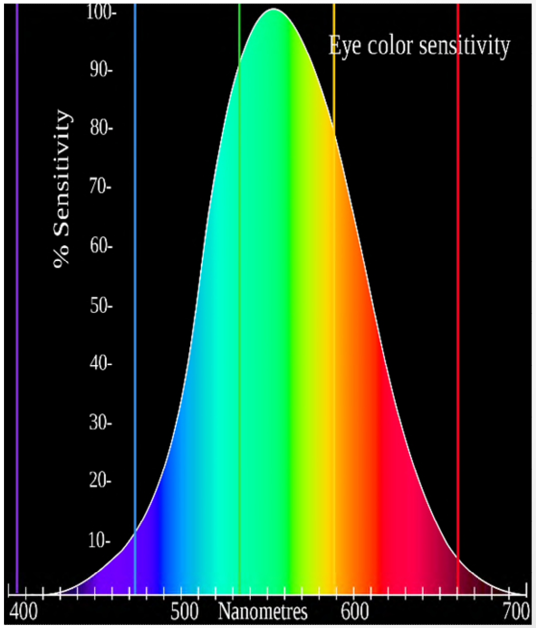
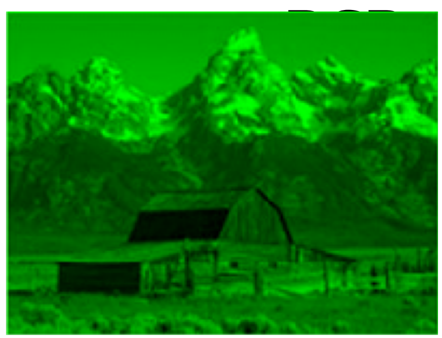


Image is obtained from
https://en.wikipedia.org/wiki/Color_vision

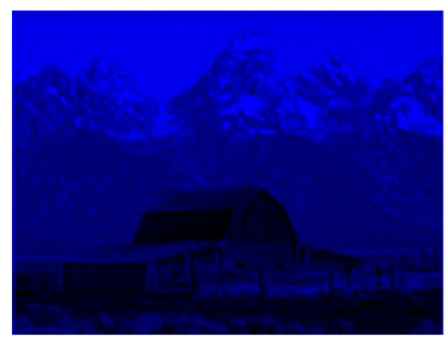
RGB to YUV

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

R



G



B

Y

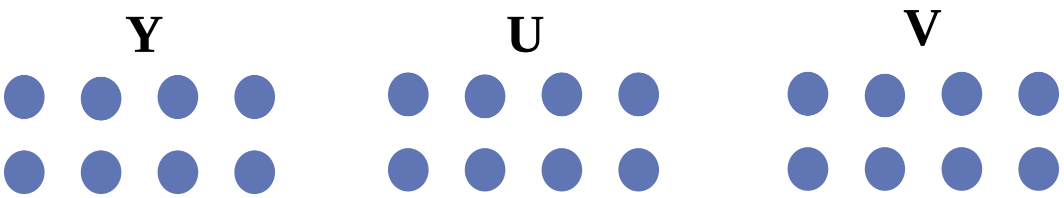
U

V

Image is obtained from https://en.wikipedia.org/wiki/YUV

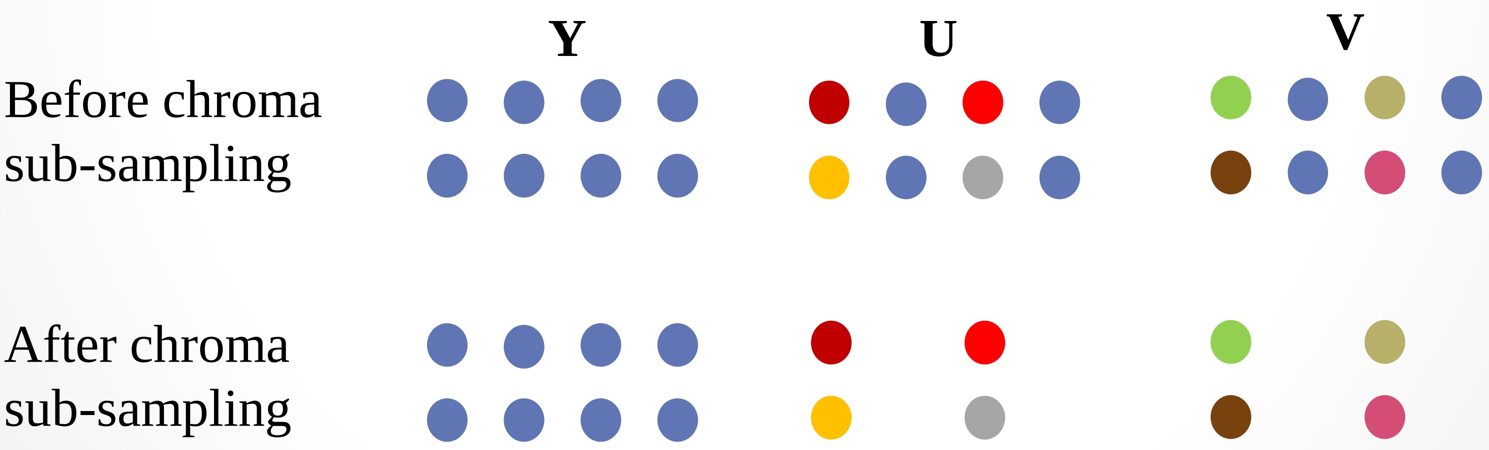
Sampling Pattern Definitions

- **4:4:4 Sampling --** Some of the diagrams indicate 4:4:4 sampling.



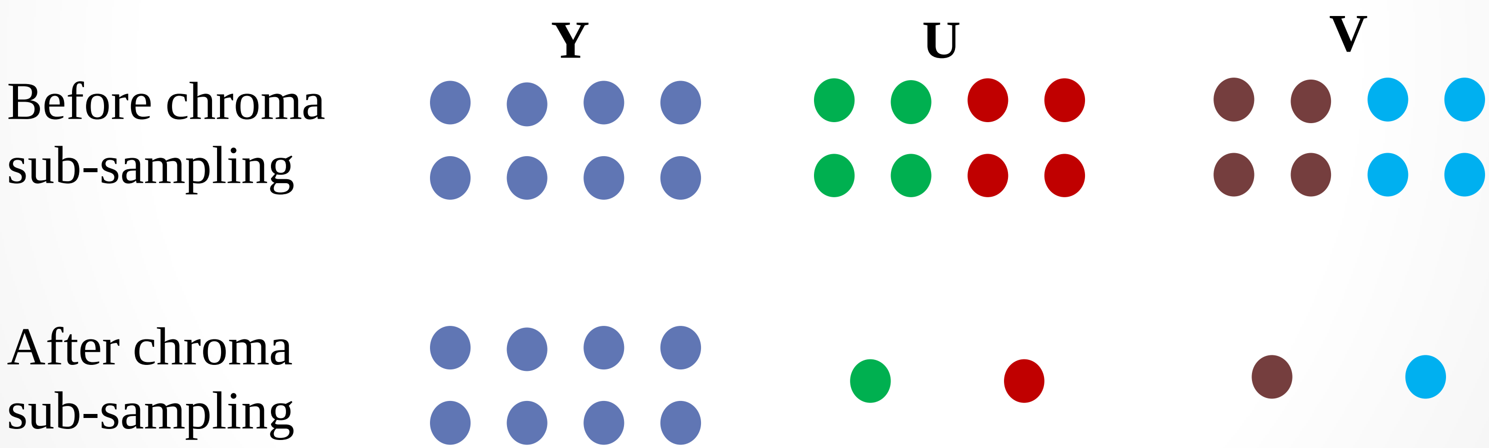
Sampling Pattern Definitions

- **4:2:2 Sampling --** 4:2:2 sampling is described by ITU-R BT.601-4 (Rec. 601). It means that for every four pixels, we get four luma samples (four Y's) but only two chroma samples (two samples of Cr and Cb respectively, which together determine the chroma), like this:



Sampling Pattern Definitions

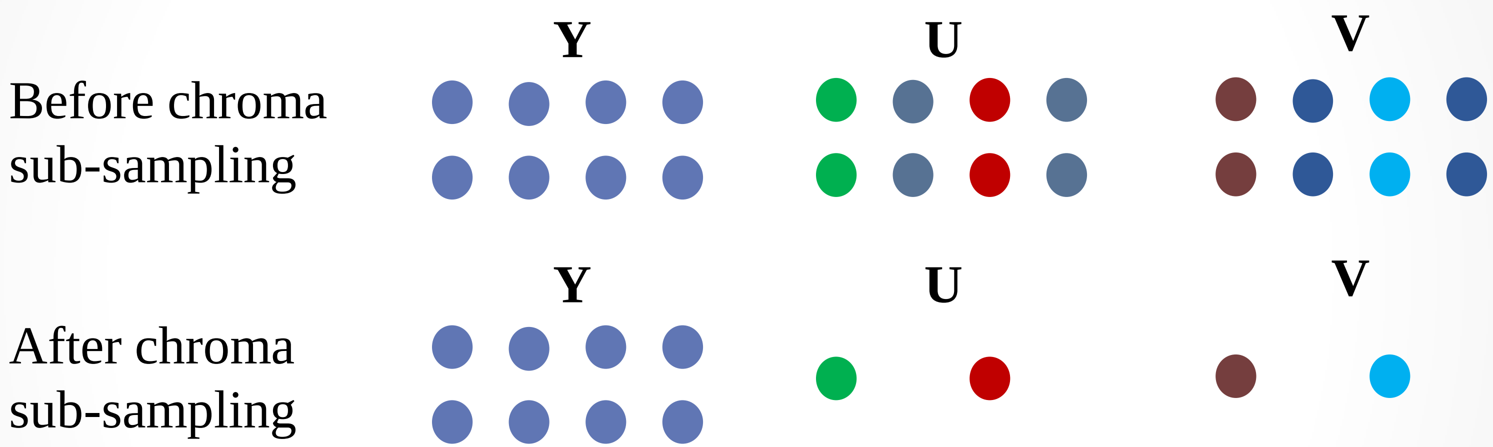
- **4:2:0 Sampling --** In this case, there is one chroma sample (one pair of Cr and Cb) for every four luminance samples (Y). The MPEG-1 and H.261 video compression standards use this 4:2:0 sampling pattern:



A new pixel is an average of 4 neighbouring pixels

Sampling Pattern Definitions

- In addition to supporting 4:4:4 and Rec.601 4:2:2, the MPEG–2 video compression standard supports this 4:2:0 sampling pattern:



A new pixel is an average of 2 neighbouring pixels

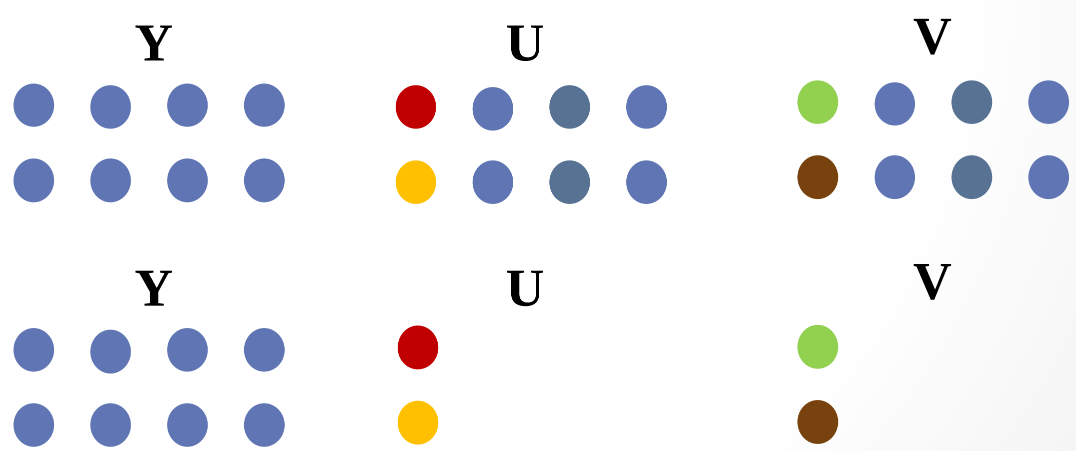
- Why the MPEG committee changed it? Perhaps they wanted to make conversion from Rec.601 4:2:2 to MPEG-2 4:2:0 less computationally expensive.

Sampling Pattern Definitions

- **4:1:1 Sampling --** Another sub-sampling method with similar "compression" as 4:2:0 is 4:1:1.

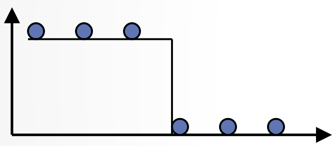
Before chroma sub-sampling

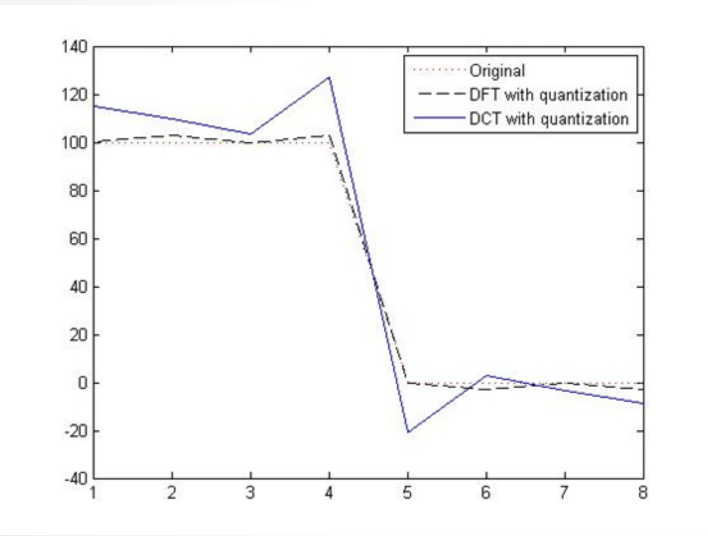
After chroma sub-sampling

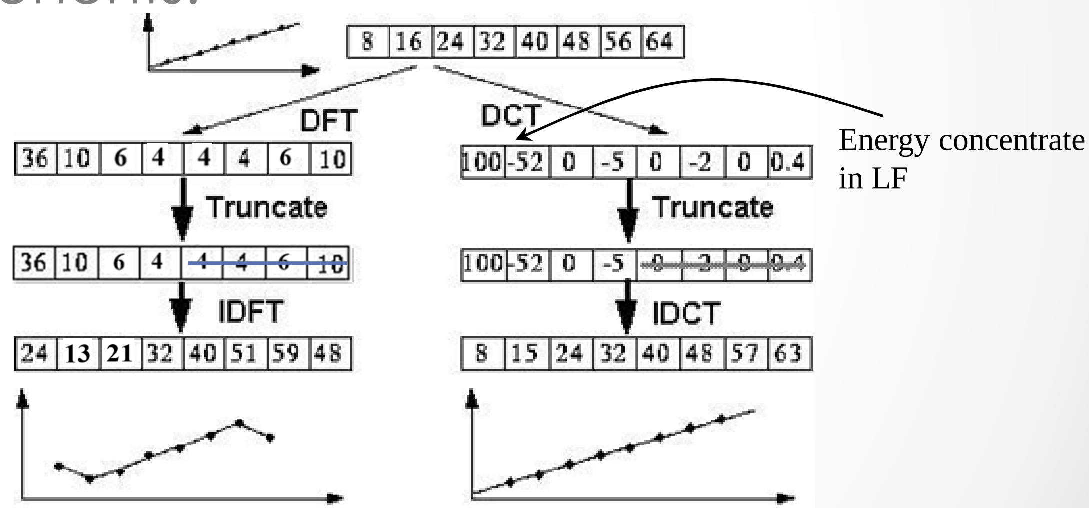


Discrete Cosine Transform

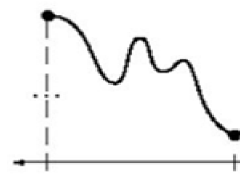
- Why DCT not DFT? -- DCT is like DFT, but can approximate linear signals well with few coefficients. DFT use to model the discontinuity of the samples, energy will be located at high frequency components.



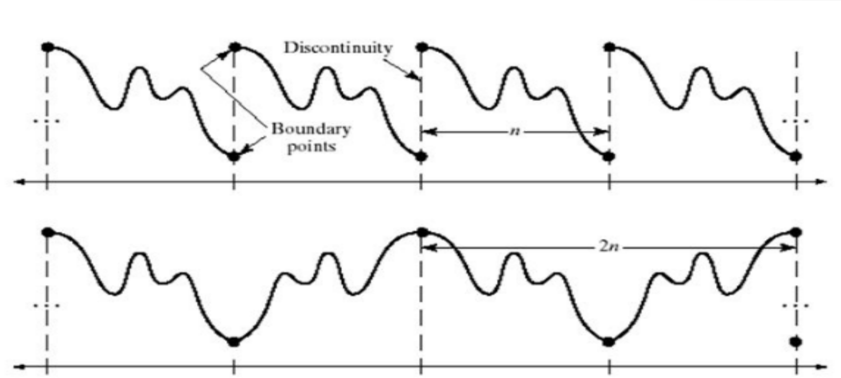




Discrete Cosine Transform



Original signal



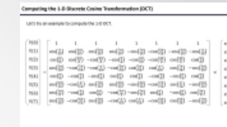
DFT

DCT

- DFT assumes that each block is repeated with periodicity N.
- DCT assumes that each block is mirror symmetry with periodicity 2N.

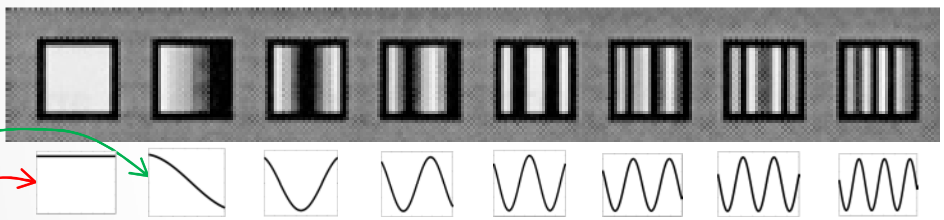
- Implicit periodicity of the DFT means that discontinuities usually occur at the boundary. These end-head discontinuities cause a high-frequency distribution in the corresponding DFT.
- On other hand, DCT computes DFT of back-to-back periodic sequence. As a result, there is no high-frequency component corresponding to the <u>boundary pixels discontinuities</u>. Hence, the DCT possesses better energy compaction in the low frequencies than the DFT.

Computing the 1-D DCT



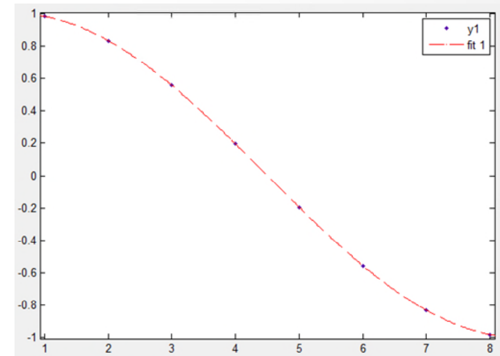
$$Y(k) = \sum_{i=0}^{7} x(i)\cos\left(\frac{(2i+1)k\pi}{16}\right) \qquad k=0,..,7$$

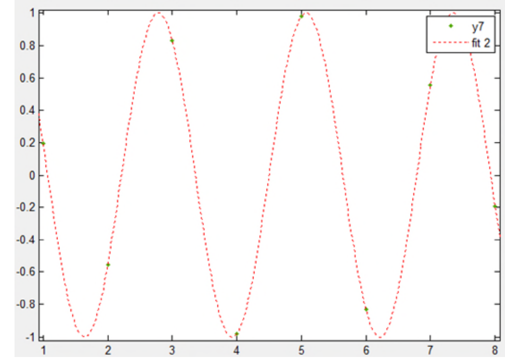
- Y(0) takes the average of the pixels.
- For Y(1), calculate low frequency. Y1 will be large if signal gradually change.

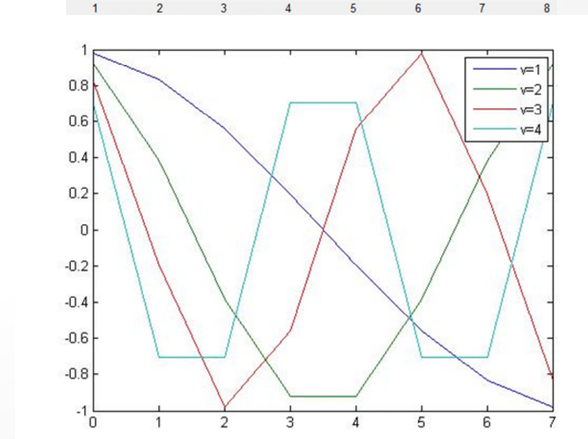


- Y(7) tends to emphasize the high frequency elements of the pixels (difference between pixels).

$$\begin{pmatrix} Y(0)\\ Y(1)\\ Y(2)\\ Y(3)\\ Y(4)\\ Y(5)\\ Y(6)\\ Y(7) \end{pmatrix} = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\\
\cos(\frac{\pi}{16}) & \cos(\frac{3\pi}{16}) & \cos(\frac{5\pi}{16}) & \cos(\frac{7\pi}{16}) & -\cos(\frac{7\pi}{16}) & -\cos(\frac{5\pi}{16}) & -\cos(\frac{3\pi}{16}) & -\cos(\frac{\pi}{16})\\
\cos(\frac{\pi}{8}) & \cos(\frac{3\pi}{8}) & -\cos(\frac{3\pi}{8}) & -\cos(\frac{\pi}{8}) & -\cos(\frac{\pi}{8}) & -\cos(\frac{3\pi}{8}) & \cos(\frac{3\pi}{8}) & \cos(\frac{\pi}{8})\\
\cos(\frac{3\pi}{16}) & -\cos(\frac{7\pi}{16}) & -\cos(\frac{\pi}{16}) & -\cos(\frac{5\pi}{16}) & \cos(\frac{5\pi}{16}) & \cos(\frac{\pi}{16}) & \cos(\frac{7\pi}{16}) & -\cos(\frac{3\pi}{16})\\
\cos(\frac{\pi}{4}) & -\cos(\frac{\pi}{4}) & -\cos(\frac{\pi}{4}) & \cos(\frac{\pi}{4}) & \cos(\frac{\pi}{4}) & -\cos(\frac{\pi}{4}) & -\cos(\frac{\pi}{4}) & \cos(\frac{\pi}{4})\\
\cos(\frac{5\pi}{16}) & -\cos(\frac{\pi}{16}) & \cos(\frac{7\pi}{16}) & \cos(\frac{3\pi}{16}) & -\cos(\frac{3\pi}{16}) & -\cos(\frac{7\pi}{16}) & \cos(\frac{\pi}{16}) & -\cos(\frac{5\pi}{16})\\
\cos(\frac{3\pi}{8}) & -\cos(\frac{\pi}{8}) & \cos(\frac{\pi}{8}) & -\cos(\frac{3\pi}{8}) & -\cos(\frac{3\pi}{8}) & \cos(\frac{\pi}{8}) & -\cos(\frac{\pi}{8}) & \cos(\frac{3\pi}{8})\\
\cos(\frac{7\pi}{16}) & -\cos(\frac{5\pi}{16}) & \cos(\frac{3\pi}{16}) & -\cos(\frac{\pi}{16}) & \cos(\frac{\pi}{16}) & -\cos(\frac{3\pi}{16}) & \cos(\frac{5\pi}{16}) & -\cos(\frac{7\pi}{16})
\end{pmatrix} \times \begin{pmatrix} x(0)\\ x(1)\\ x(2)\\ x(3)\\ x(4)\\ x(5)\\ x(6)\\ x(7) \end{pmatrix}$$





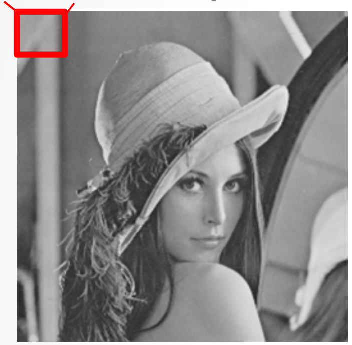


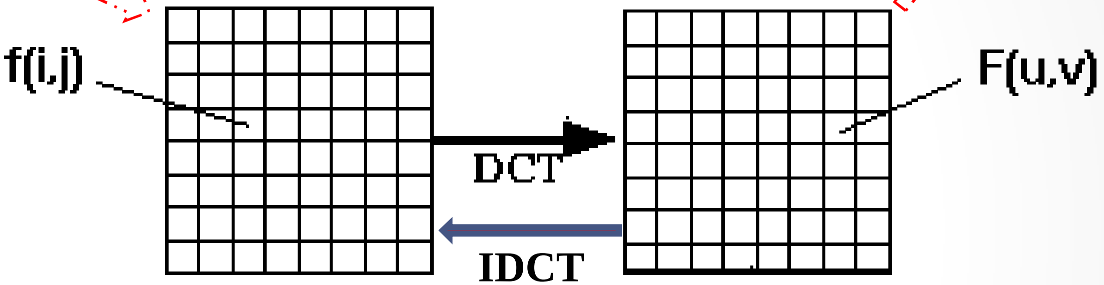
2-D DCT

76	68	65	65	65	65	68	73
76	68	66	66	64	65	68	73
75	67	66	66	64	65	68	72
74	68	65	65	64	65	68	73
73	67	65	65	64	65	68	73
73	67	65	65	64	65	67	73
73	70	66	63	63	66	70	73
73	70	66	63	63	66	70	73

-483.1250	1.7102	25.5989	-0.2148	11.3750	3.1852	3.3324	-0.4426
3.5185	2.2448	1.1681	1.8343	0.1998	0.6538	-0.3247	0.2546
-0.2590	0.4080	0.3384	-0.1283	0.8562	0.1920	0.2866	0.3167
0.2695	-0.3552	0.2529	0.6294	0.3285	-1.0440	-0.1421	0.1350
-0.3750	-0.7855	0.4339	0.1022	-0.3750	-0.6576	-0.5856	-0.0507
-0.1464	0.0721	0.0876	0.4864	0.3698	-0.3669	-0.2240	0.3948
-0.2986	0.0187	-0.4634	0.2122	-0.2194	0.0268	0.1616	-0.0938
-0.2979	-0.2150	-0.5027	0.0962	0.1672	0.6272	0.1019	0.4927

- From spatial domain to frequency domain:





- A reversible, linear transform maps the image f(i,j) into transform coefficients F(u,v), then quantized & coded
- For most natural images, a significant number of coefficients have small magnitudes and can be coarsely quantized or discarded with little distortion ---> compression
- Hence: DCT --> good compromise between information packing and computational complexity

2-D DCT

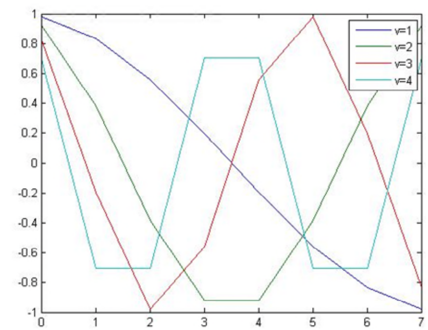
2-D Discrete Cosine Transform (DCT)

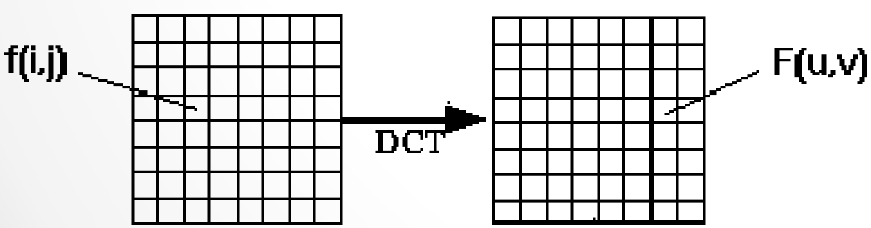
$$F(u,v) = c(u)c(v)\sum_{j=0}^{7}\sum_{i=0}^{7} f(i,j)\cos\left[\frac{(2i+1)u\pi}{16}\right]\cos\left[\frac{(2j+1)v\pi}{16}\right]$$

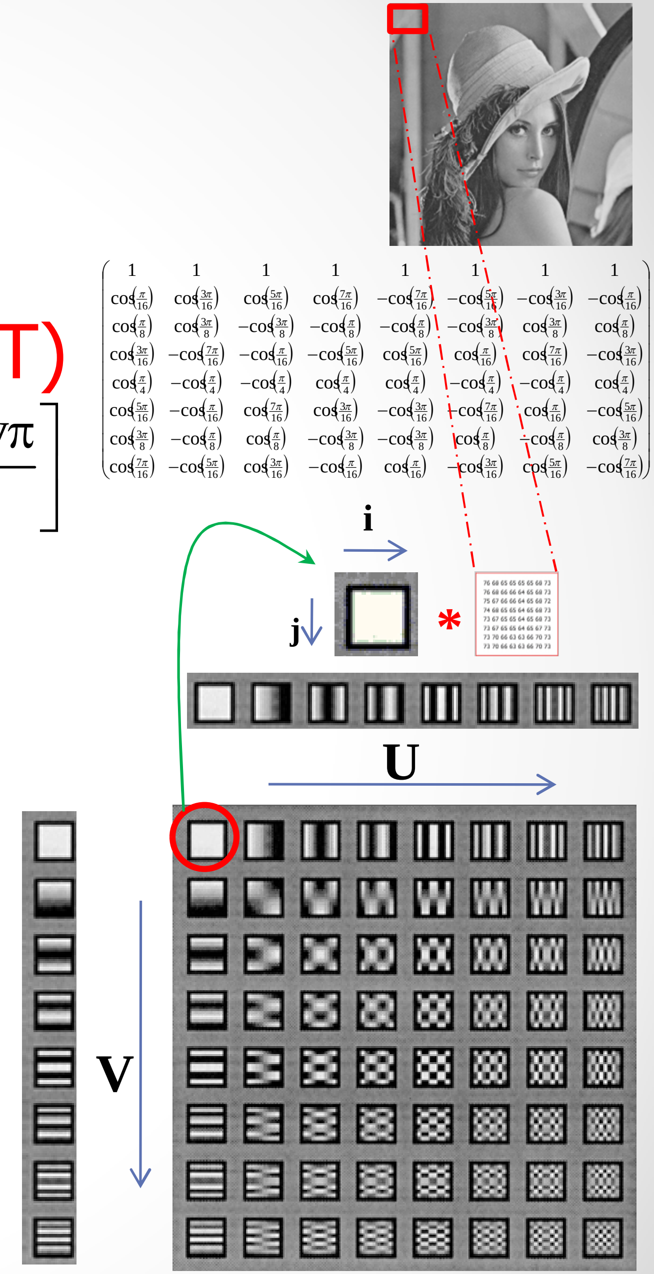
$$u, v = 0, 1, 2, \ldots, 7$$

$$c(u) = \sqrt{\frac{1}{8}} \ldots\ldots\text{for..}u = 0$$

$$c(u) = \sqrt{\frac{1}{4}} \ldots\ldots\text{for..}u = 1, 2, \ldots, 7$$







(At receiver: 2-D IDCT performed by decoder)

Computing the 2-D DCT by 1-D DCT

- Row-Column Decomposition. The 2-D DCT is calculated using row-column decomposition. First, the 1-D DCT of each row is computed. Then, the 1-D DCT of each of the resulting columns is computed, which yields the 2-D transform. The transformation of the columns cannot begin until all the rows have been transfc

$$F[u,v] = \frac{1}{2}\sum_{i} A(u)\cos\frac{(2i+1)u\pi}{16}G[i,v]$$

$$G[i,v] = \frac{1}{2}\sum_{j} A(v)\cos\frac{(2j+1)v\pi}{16}f[i,j]$$