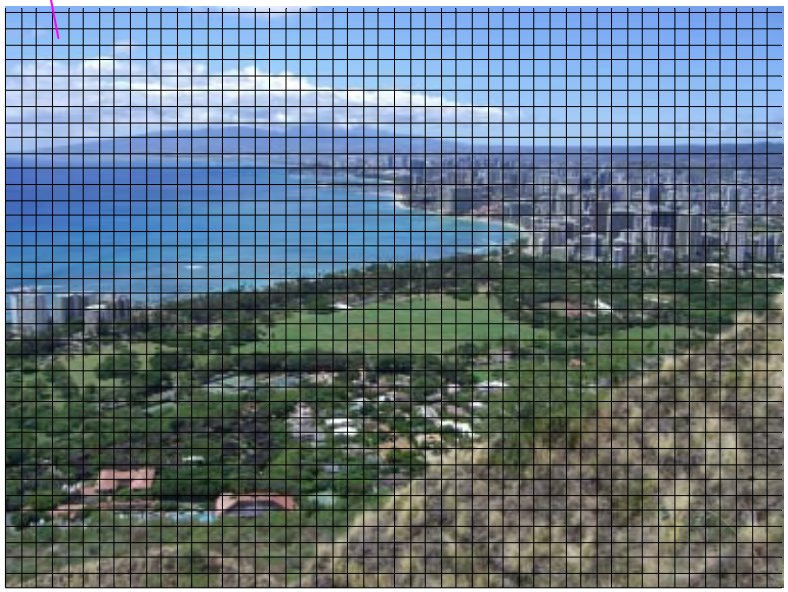
JPEGとMPEGのアルゴリズム

Takuichi Hirano

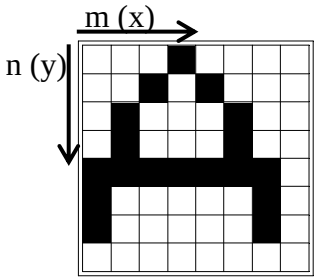
Tokyo Institute of Technology

JPEG Algorithm

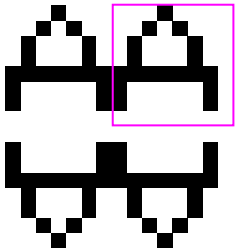
- RGB→YCbCr (YUV)
- DCT for All 8 x 8 Sub Picture
- Quantize by Quantization Table
- Zigzag Scan
- Entropy Encoding (Huffman, Arithmetic, etc.)
- Create JPEG File



DCT for 8dots x 8dots Picture



8 x 8 Picture $f(m,n)$



$N=8$

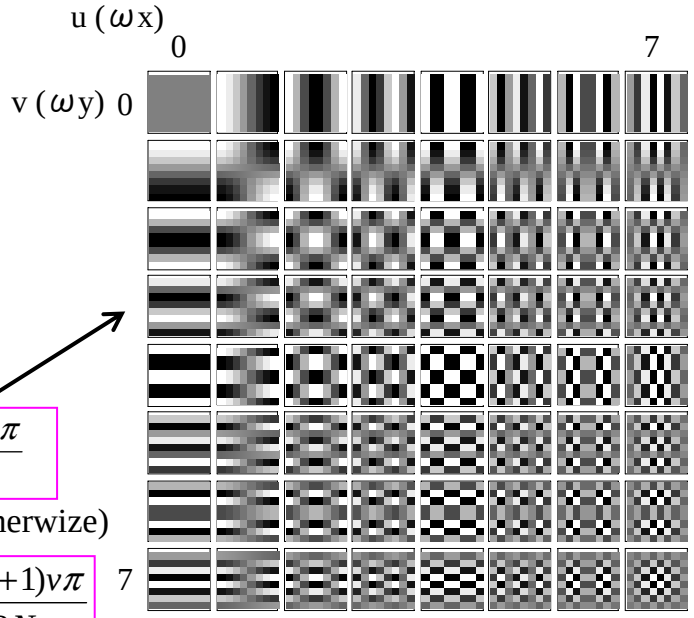
Discrete Cosine Transform (DCT)

$$F(u,v)=\frac{2}{N}c_u c_v\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}f(m,n)\cos\frac{(2m+1)u\pi}{2N}\cos\frac{(2n+1)v\pi}{2N}$$

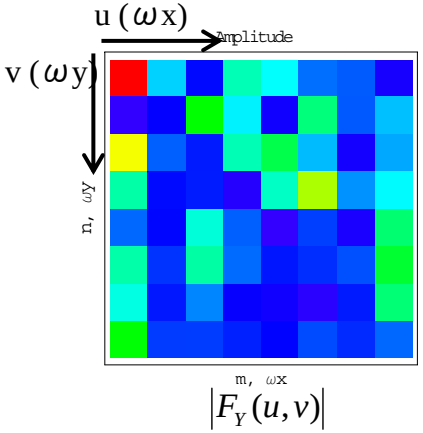
$$c_u=1/2\ (u=0),1\ (\text{otherwize})$$

Inverse DCT

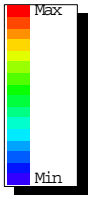
$$f(m,n)=\frac{2}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}c_u c_v F(u,v)\cos\frac{(2m+1)u\pi}{2N}\cos\frac{(2n+1)v\pi}{2N}$$



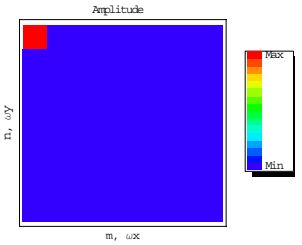
8 x 8=64 Bases



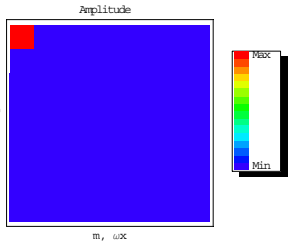
$|F_Y(u,v)|$



442	−108	24	−157	128	−66	59	11
0	19	−231	−121	−15	−181	57	−100
333	60	32	158	−201	−99	−13	−90
163	25	32	5	151	302	−81	125
−64	−23	142	61	0	47	10	185
−162	−42	163	66	−30	40	−54	−212
138	31	−77	−18	14	−4	32	183
230	46	−46	35	−22	52	−38	−64



$|F_{C_b}(u,v)|$



$|F_{C_r}(u,v)|$

Quantization

Quantization Table

どうやってこれを決めたのか？→人間が見て綺麗なように試行錯誤

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

$$Q_{C_bC_r} = \begin{pmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{pmatrix}$$

For Cb and Cr components

$$F_{YQ} = Round\ (F_Y / (Q_Y / s)) = \lfloor Round\ (F_{Y\,ij} / (Q_{Y\,ij} / s)) \rfloor =$$

Quality

Large: High quality
Small: Low quality

$$\begin{pmatrix} 28 & -10 & 2 & -10 & 5 & -2 & 1 & 0 \\ 0 & 2 & -17 & -6 & -1 & -3 & 1 & -2 \\ 24 & 5 & 2 & 7 & -5 & -2 & 0 & -2 \\ 12 & 1 & 1 & 0 & 3 & 3 & -1 & 2 \\ -4 & -1 & 4 & 1 & 0 & 0 & 0 & 2 \\ -7 & -1 & 3 & 1 & 0 & 0 & 0 & -2 \\ 3 & 0 & -1 & 0 & 0 & 0 & 0 & 2 \\ 3 & 0 & 0 & 0 & 0 & 1 & 0 & -1 \end{pmatrix}$$

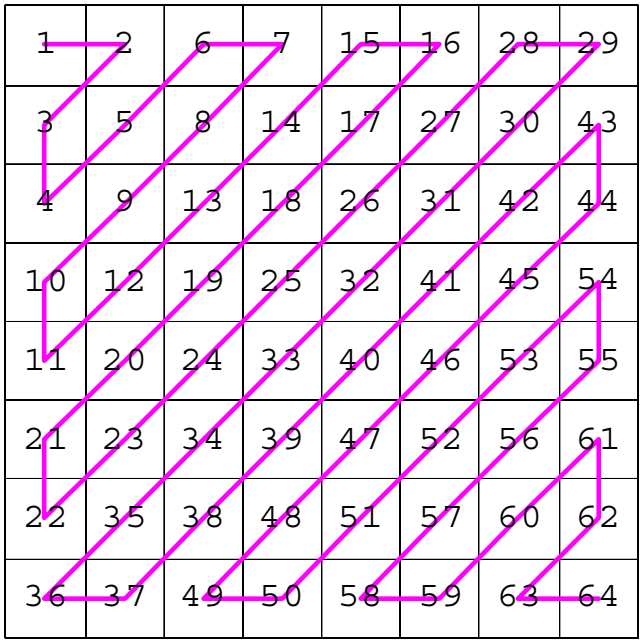
$$F_{C_bQ} = \begin{pmatrix} 60 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$F_{C_rQ} = \begin{pmatrix} 60 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Quantization

$$F_{YQ} = \begin{pmatrix} 28 & -10 & 2 & -10 & 5 & -2 & 1 & 0 \\ 0 & 2 & -17 & -6 & -1 & -3 & 1 & -2 \\ 24 & 5 & 2 & 7 & -5 & -2 & 0 & -2 \\ 12 & 1 & 1 & 0 & 3 & 3 & -1 & 2 \\ -4 & -1 & 4 & 1 & 0 & 0 & 0 & 2 \\ -7 & -1 & 3 & 1 & 0 & 0 & 0 & -2 \\ 3 & 0 & -1 & 0 & 0 & 0 & 0 & 2 \\ 3 & 0 & 0 & 0 & 0 & 1 & 0 & -1 \end{pmatrix}$$

$$F_{C_bQ} = \begin{pmatrix} 60 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad F_{C_rQ} = \begin{pmatrix} 60 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$



- Zigzag Scan
- Entropy Encoding (Huffman Enc., Arithmetic, etc.)
- Create JPEG File

JPEG Decoding

$$F_Y \cong F_{YQ} \times (Q_Y / s) = \lfloor F_{YQ\,ij} \times (Q_{Y\,ij} / s) \rfloor$$

$$\begin{pmatrix} 448 & -110 & 20 & -160 & 120 & -80 & 51 & 0 \\ 0 & 24 & -238 & -114 & -26 & -174 & 60 & -110 \\ 336 & 65 & 32 & 168 & -200 & -114 & 0 & -112 \\ 168 & 17 & 22 & 0 & 153 & 261 & -80 & 124 \\ -72 & -22 & 148 & 56 & 0 & 0 & 0 & 154 \\ -168 & -35 & 165 & 64 & 0 & 0 & 0 & -184 \\ 147 & 0 & -78 & 0 & 0 & 0 & 0 & 202 \\ 216 & 0 & 0 & 0 & 0 & 100 & 0 & -99 \end{pmatrix}$$

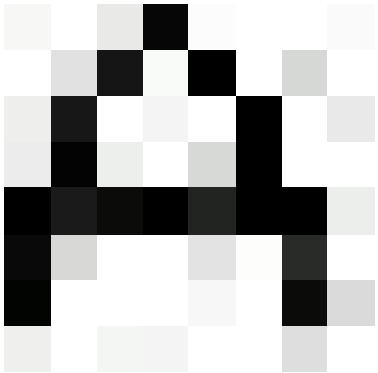
(a) Decoded (s=1)

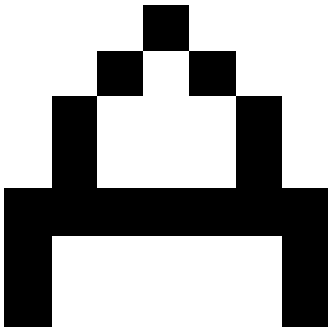
$$\begin{pmatrix} 442 & -108 & 24 & -157 & 128 & -66 & 59 & 11 \\ 0 & 19 & -231 & -121 & -15 & -181 & 57 & -100 \\ 333 & 60 & 32 & 158 & -201 & -99 & -13 & -90 \\ 163 & 25 & 32 & 5 & 151 & 302 & -81 & 125 \\ -64 & -23 & 142 & 61 & 0 & 47 & 10 & 185 \\ -162 & -42 & 163 & 66 & -30 & 40 & -54 & -212 \\ 138 & 31 & -77 & -18 & 14 & -4 & 32 & 183 \\ 230 & 46 & -46 & 35 & -22 & 52 & -38 & -64 \end{pmatrix}$$

(b) Original

Y component

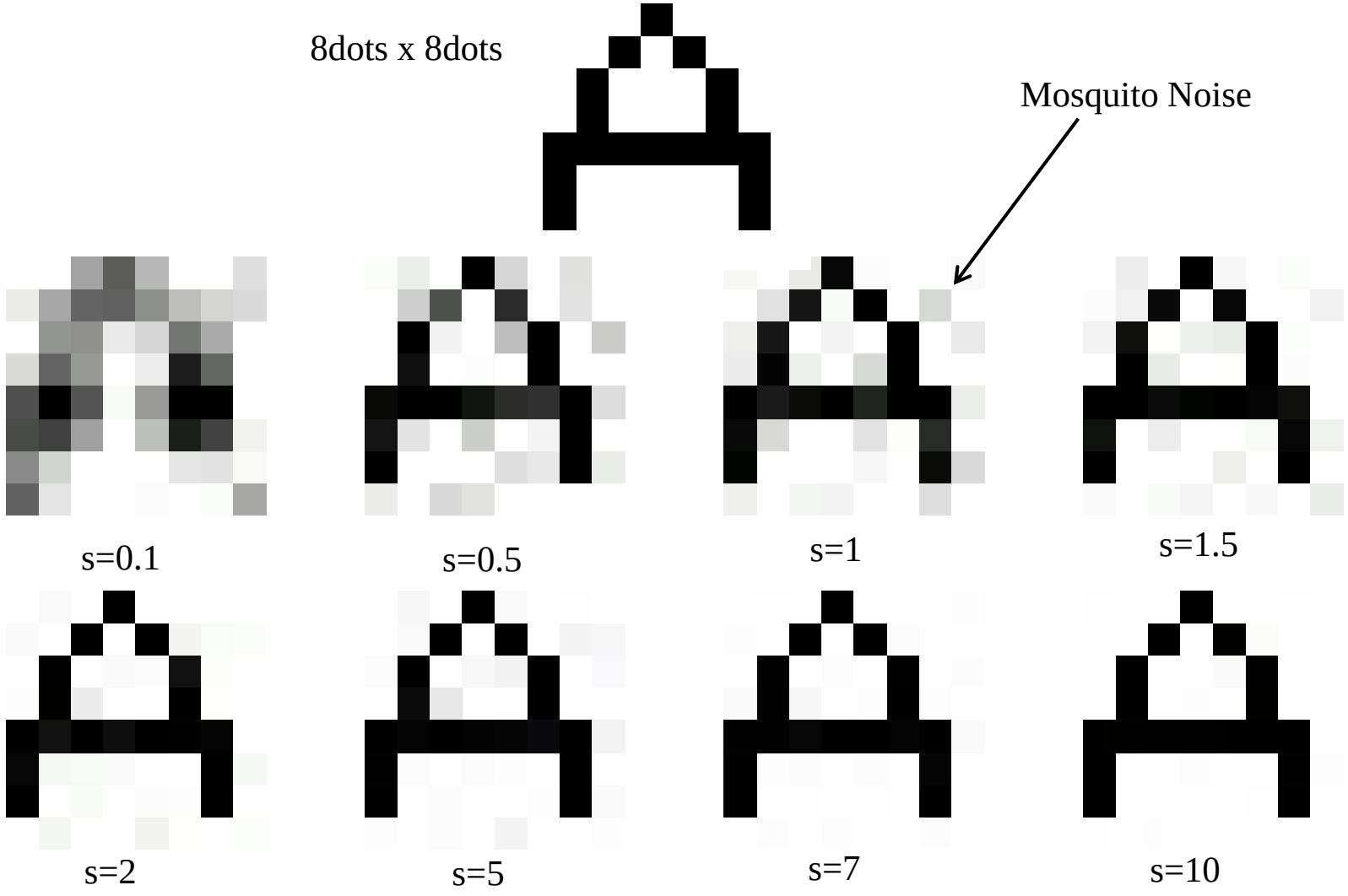
- IDCT for All 8 x 8 Sub Picture
- YCbCr (YUV)→RGB



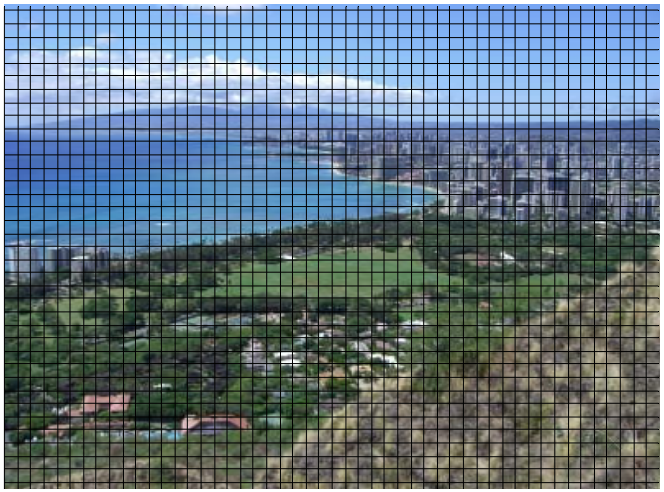


BMP
GIF (LZW)
PNG
PCX (RLE)
TGA (RLE, 非圧縮)
TIFF
JPEG

JPEG Quality (Example1)



JPEG Quality (Example2)



Original (400dots x 300dots)

s=1

Block Noise

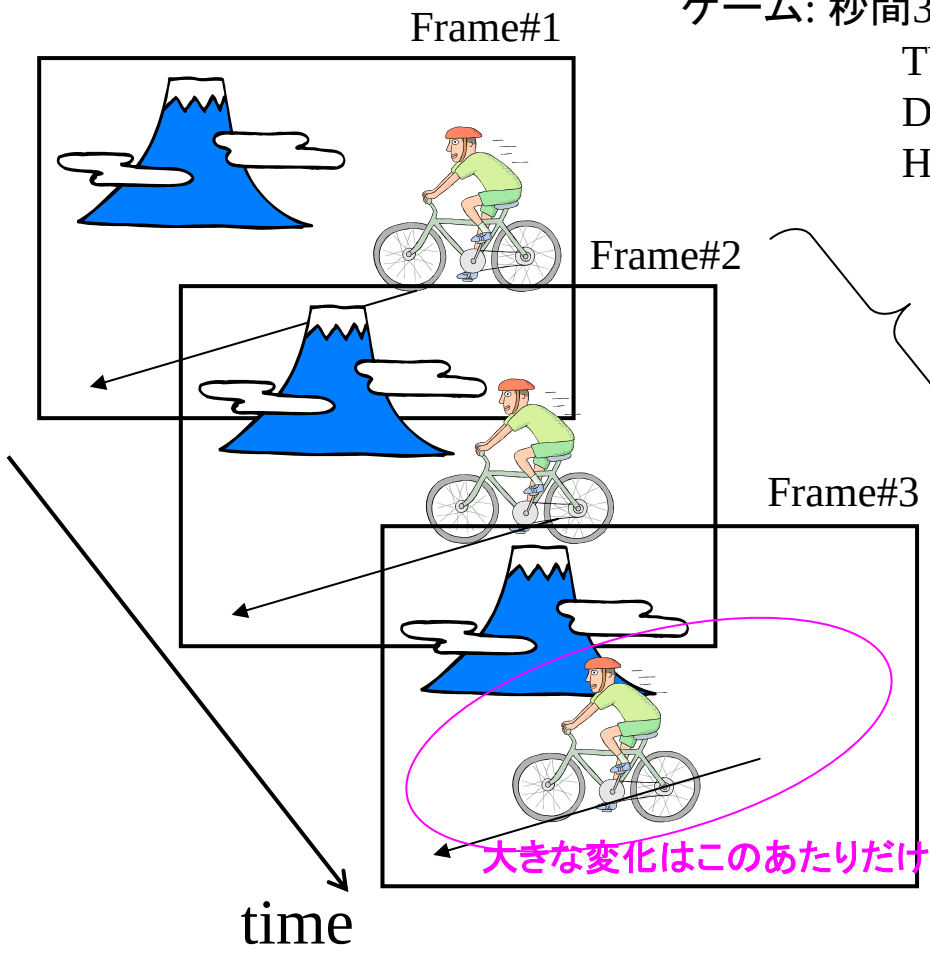
s=0.1

MPEG

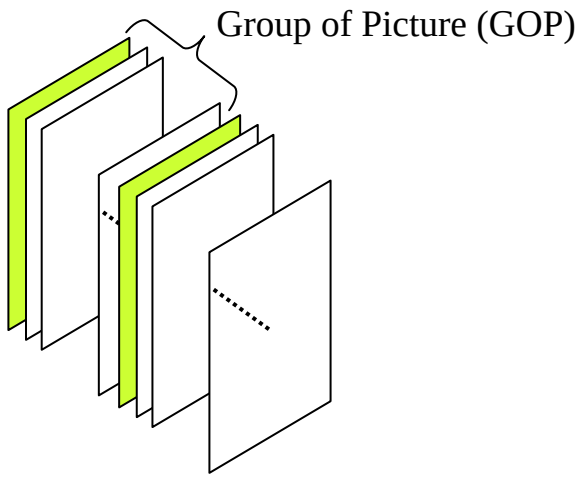
TV, DVD: 秒間30フレーム・秒間60フィールド(インターレース)
ゲーム: 秒間30 or 60フレーム

TV: Analog x 525本
DVD: 720 x 480 (9.8Mbps)
HDTV: 1280 x 720 (40Mbps)　HDMI: 4.5Gbps

1/30 ⇄ 1280 x 720 x 24 x 60 / (1024)^3=1.2Gbps



Frame#i – Frame#(i-1)は滑らかな画像
(JPEGで高圧縮となる)



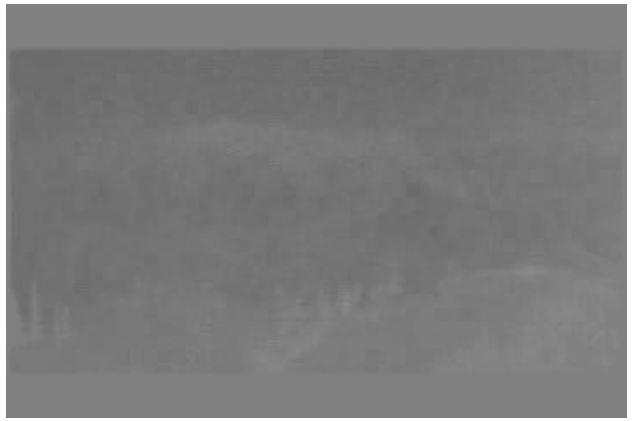
DVDでは最大18フレームでフレーム間
予測はやめる。

フレーム間予測 (1)

Frame 1 (Original)

Y

Cb



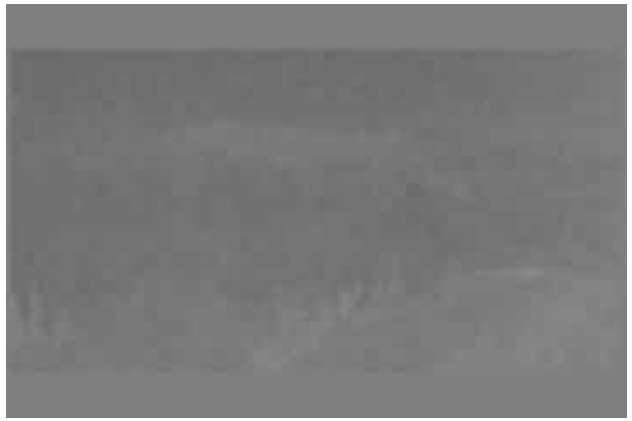
Cr

フレーム間予測 (2)

Frame 1 (Coded + Decoded) 360 x 240; s=1

Y

Cb



Cr

フレーム間予測 (3)

Frame 2 (Original)

Y

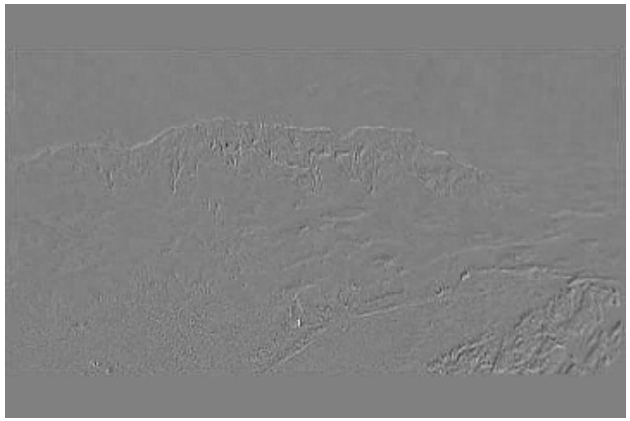
Cb

Cr

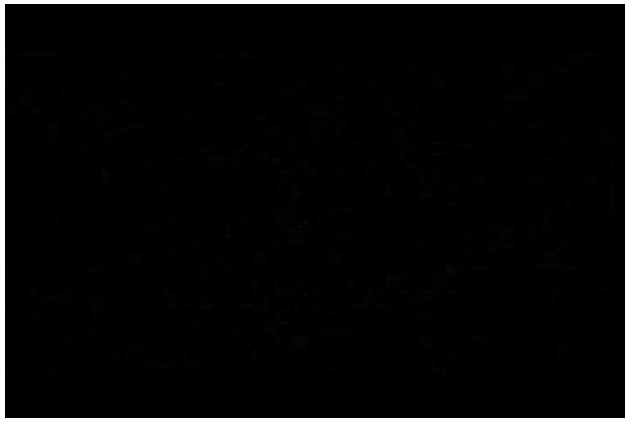
フレーム間予測 (4)

Frame 2 (Original) – Frame 1 (Coded + Decoded)

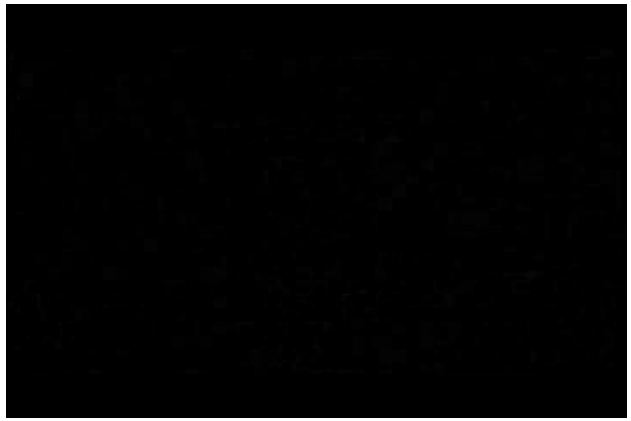
これらの成分のJPEG圧縮率は高い！



Y



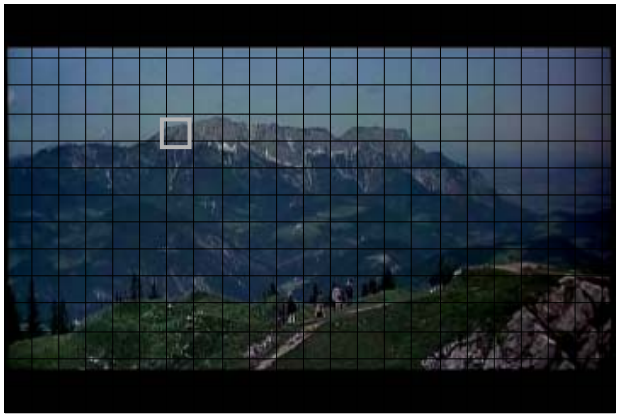
Cb

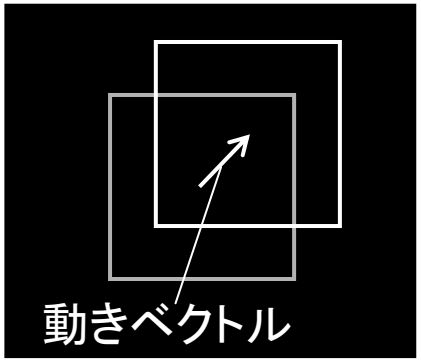


Cr

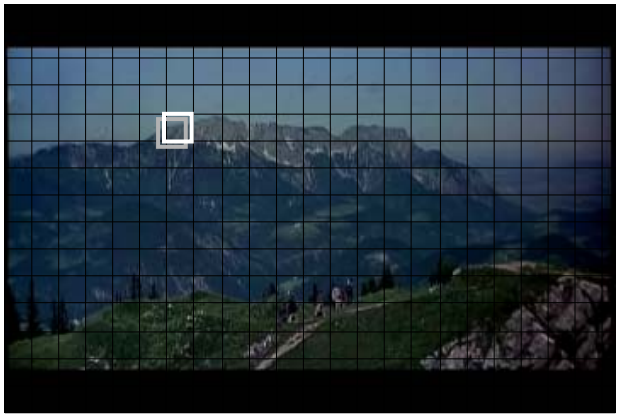
フレーム間予測 (動きベクトルについて)

Frame1





Frame2



1フレーム前の相関が最も高い部分からの動きベクトルを保存し、そことの差の画像を保存する方式にすると、差の画像はより滑らかになり、より圧縮率が高くなる。