Обробка зображень і мультимедіа

Олег Гутік

Лекція 14: Стиснення зображень, VIII. Дискретне косинус-перетворення

Перш за все ми розглянемо одновимірне (векторне) перетворення DCT (у додатках використовується *двовимірне* (*матричне*) косинус-перетворення, але векторне DCT простіше зрозуміти, і воно ґрунтується на тих же принципах). На рис. зображено вісім хвиль косинуса, $w(f) = \cos(f\theta)$, при $0 \leqslant \theta \leqslant \pi$, з частотами $f = 0, 1, \ldots, 7$. На кожному графіку зазначено вісім значень функції $w(f)$ з абсцисами

$$\theta = \frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}, \tag{1}$$

які формують базовий вектор $\vec{\boldsymbol{v}}_f$.

Перш за все ми розглянемо одновимірне (векторне) перетворення DCT (у додатках використовується *двовимірне* (*матричне*) косинус-перетворення, але векторне DCT простіше зрозуміти, і воно ґрунтується на тих же принципах). На рис. зображено вісім хвиль косинуса, $w(f) = \cos(f\theta)$, при $0 \leqslant \theta \leqslant \pi$, з частотами $f = 0, 1, \ldots, 7$. На кожному графіку зазначено вісім значень функції $w(f)$ з абсцисами

$$\theta = \frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}, \tag{1}$$

які формують базовий вектор $\vec{v}_f$.

Перш за все ми розглянемо одновимірне (векторне) перетворення DCT (у додатках використовується *двовимірне* (*матричне*) косинус-перетворення, але векторне DCT простіше зрозуміти, і воно ґрунтується на тих же принципах). На рис. зображено вісім хвиль косинуса, $w(f) = \cos(f\theta)$, при $0 \leqslant \theta \leqslant \pi$, з частотами $f = 0, 1, \ldots, 7$. На кожному графіку зазначено вісім значень функції $w(f)$ з абсцисами

$$\theta = \frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}, \qquad (1)$$

які формують базовий вектор $\vec{v}_f$.

Перш за все ми розглянемо одновимірне (векторне) перетворення DCT (у додатках використовується *двовимірне* (*матричне*) косинус-перетворення, але векторне DCT простіше зрозуміти, і воно ґрунтується на тих же принципах). На рис. зображено вісім хвиль косинуса, $w(f) = \cos(f\theta)$, при $0 \leqslant \theta \leqslant \pi$, з частотами $f = 0, 1, \ldots, 7$. На кожному графіку зазначено вісім значень функції $w(f)$ з абсцисами

$$\theta = \frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}, \qquad (1)$$

які формують базовий вектор $\vec{v}_f$.

Перш за все ми розглянемо одновимірне (векторне) перетворення DCT (у додатках використовується *двовимірне* (*матричне*) косинус-перетворення, але векторне DCT простіше зрозуміти, і воно ґрунтується на тих же принципах). На рис. зображено вісім хвиль косинуса, $w(f) = \cos(f\theta)$, при $0 \leqslant \theta \leqslant \pi$, з частотами $f = 0, 1, \dots, 7$. На кожному графіку зазначено вісім значень функції $w(f)$ з абсцисами

$$\theta = \frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}, \qquad (1)$$

які формують базовий вектор $\vec{v}_f$.

Перш за все ми розглянемо одновимірне (векторне) перетворення DCT (у додатках використовується *двовимірне* (*матричне*) косинус-перетворення, але векторне DCT простіше зрозуміти, і воно ґрунтується на тих же принципах). На рис. зображено вісім хвиль косинуса, $w(f) = \cos(f\theta)$, при $0 \leqslant \theta \leqslant \pi$, з частотами $f = 0, 1, \ldots, 7$. На кожному графіку зазначено вісім значень функції $w(f)$ з абсцисами

$$\theta = \frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}, \qquad (1)$$

які формують базовий вектор $\vec{v}_f$.

Перш за все ми розглянемо одновимірне (векторне) перетворення DCT (у додатках використовується *двовимірне* (*матричне*) косинус-перетворення, але векторне DCT простіше зрозуміти, і воно ґрунтується на тих же принципах). На рис. зображено вісім хвиль косинуса, $w(f) = \cos(f\theta)$, при $0 \leqslant \theta \leqslant \pi$, з частотами $f = 0, 1, \ldots, 7$. На кожному графіку зазначено вісім значень функції $w(f)$ з абсцисами

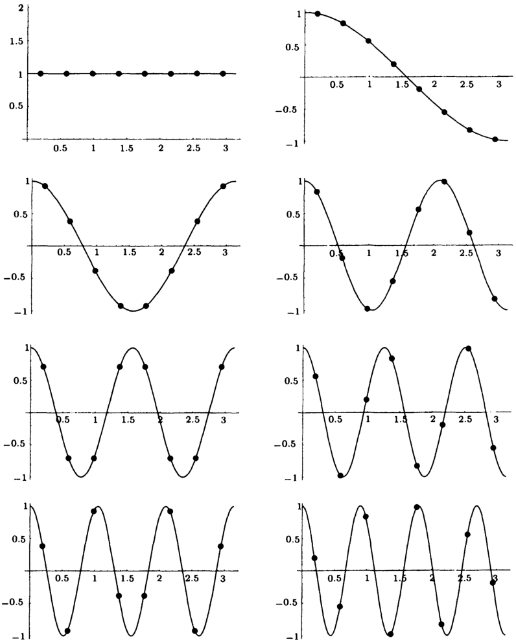
$$\theta = \frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}, \qquad (1)$$

які формують базовий вектор $\vec{v}_f$.

Перш за все ми розглянемо одновимірне (векторне) перетворення DCT (у додатках використовується *двовимірне* (*матричне*) косинус-перетворення, але векторне DCT простіше зрозуміти, і воно ґрунтується на тих же принципах). На рис. зображено вісім хвиль косинуса, $w(f) = \cos(f\theta)$, при $0 \leqslant \theta \leqslant \pi$, з частотами $f = 0, 1, \ldots, 7$. На кожному графіку зазначено вісім значень функції $w(f)$ з абсцисами

$$\theta = \frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}, \qquad (1)$$

які формують базовий вектор $\vec{\boldsymbol{v}}_f$.



Стиснення зображень. Дискретне косинус-перетворення

θ	0.196	0.589	0.982	1.374	1.767	2.160	2.553	2.945
$\cos 0\theta$	1.	1.	1.	1.	1.	1.	1.	1.
$\cos 1\theta$	0.981	0.831	0.556	0.195	-0.195	-0.556	-0.831	-0.981
$\cos 2\theta$	0.924	0.383	-0.383	-0.924	-0.924	-0.383	0.383	0.924
$\cos 3\theta$	0.831	-0.195	-0.981	-0.556	0.556	0.981	0.195	-0.831
$\cos 4\theta$	0.707	-0.707	-0.707	0.707	0.707	-0.707	-0.707	0.707
$\cos 5\theta$	0.556	-0.981	0.195	0.831	-0.831	-0.195	0.981	-0.556
$\cos 6\theta$	0.383	-0.924	0.924	-0.383	-0.383	0.924	-0.924	0.383
$\cos 7\theta$	0.195	-0.556	0.831	-0.981	0.981	-0.831	0.556	-0.195

У результаті вийде вісім векторів $\vec{v}_f$, $f = 0, 1, \ldots, 7$ (всього 64 числа), які представлені в табл. Вони є базисом одновимірного косинус-перетворення. Зазначимо схожість цієї таблиці з матрицею W із рівняння (2).

$$W = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}. \tag{2}$$

В обох випадках частота зміни знаків зростає рядками.

θ	0.196	0.589	0.982	1.374	1.767	2.160	2.553	2.945
$\cos 0\theta$	1.	1.	1.	1.	1.	1.	1.	1.
$\cos 1\theta$	0.981	0.831	0.556	0.195	-0.195	-0.556	-0.831	-0.981
$\cos 2\theta$	0.924	0.383	-0.383	-0.924	-0.924	-0.383	0.383	0.924
$\cos 3\theta$	0.831	-0.195	-0.981	-0.556	0.556	0.981	0.195	-0.831
$\cos 4\theta$	0.707	-0.707	-0.707	0.707	0.707	-0.707	-0.707	0.707
$\cos 5\theta$	0.556	-0.981	0.195	0.831	-0.831	-0.195	0.981	-0.556
$\cos 6\theta$	0.383	-0.924	0.924	-0.383	-0.383	0.924	-0.924	0.383
$\cos 7\theta$	0.195	-0.556	0.831	-0.981	0.981	-0.831	0.556	-0.195

У результаті вийде вісім векторів $\vec{v}_f$, $f = 0, 1, \ldots, 7$ (всього 64 числа), які представлені в табл. Вони є базисом одновимірного косинус-перетворення. Зазначимо схожість цієї таблиці з матрицею W із рівняння (2).

$$W = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}. \tag{2}$$

В обох випадках частота зміни знаків зростає рядками.

θ	0.196	0.589	0.982	1.374	1.767	2.160	2.553	2.945
$\cos 0\theta$	1.	1.	1.	1.	1.	1.	1.	1.
$\cos 1\theta$	0.981	0.831	0.556	0.195	-0.195	-0.556	-0.831	-0.981
$\cos 2\theta$	0.924	0.383	-0.383	-0.924	-0.924	-0.383	0.383	0.924
$\cos 3\theta$	0.831	-0.195	-0.981	-0.556	0.556	0.981	0.195	-0.831
$\cos 4\theta$	0.707	-0.707	-0.707	0.707	0.707	-0.707	-0.707	0.707
$\cos 5\theta$	0.556	-0.981	0.195	0.831	-0.831	-0.195	0.981	-0.556
$\cos 6\theta$	0.383	-0.924	0.924	-0.383	-0.383	0.924	-0.924	0.383
$\cos 7\theta$	0.195	-0.556	0.831	-0.981	0.981	-0.831	0.556	-0.195

У результаті вийде вісім векторів $\vec{v}_f$, $f = 0, 1, \ldots, 7$ (всього 64 числа), які представлені в табл. Вони є базисом одновимірного косинус-перетворення. Зазначимо схожість цієї таблиці з матрицею W із рівняння (2).

$$W = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}. \tag{2}$$

В обох випадках частота зміни знаків зростає рядками.

θ	0.196	0.589	0.982	1.374	1.767	2.160	2.553	2.945
$\cos 0\theta$	1.	1.	1.	1.	1.	1.	1.	1.
$\cos 1\theta$	0.981	0.831	0.556	0.195	-0.195	-0.556	-0.831	-0.981
$\cos 2\theta$	0.924	0.383	-0.383	-0.924	-0.924	-0.383	0.383	0.924
$\cos 3\theta$	0.831	-0.195	-0.981	-0.556	0.556	0.981	0.195	-0.831
$\cos 4\theta$	0.707	-0.707	-0.707	0.707	0.707	-0.707	-0.707	0.707
$\cos 5\theta$	0.556	-0.981	0.195	0.831	-0.831	-0.195	0.981	-0.556
$\cos 6\theta$	0.383	-0.924	0.924	-0.383	-0.383	0.924	-0.924	0.383
$\cos 7\theta$	0.195	-0.556	0.831	-0.981	0.981	-0.831	0.556	-0.195

У результаті вийде вісім векторів $\vec{v}_f$, $f = 0, 1, \ldots, 7$ (всього 64 числа), які представлені в табл. Вони є базисом одновимірного косинус-перетворення. Зазначимо схожість цієї таблиці з матрицею W із рівняння (2).

$$W = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}. \tag{2}$$

В обох випадках частота зміни знаків зростає рядками.

θ	0.196	0.589	0.982	1.374	1.767	2.160	2.553	2.945
$\cos 0\theta$	1.	1.	1.	1.	1.	1.	1.	1.
$\cos 1\theta$	0.981	0.831	0.556	0.195	-0.195	-0.556	-0.831	-0.981
$\cos 2\theta$	0.924	0.383	-0.383	-0.924	-0.924	-0.383	0.383	0.924
$\cos 3\theta$	0.831	-0.195	-0.981	-0.556	0.556	0.981	0.195	-0.831
$\cos 4\theta$	0.707	-0.707	-0.707	0.707	0.707	-0.707	-0.707	0.707
$\cos 5\theta$	0.556	-0.981	0.195	0.831	-0.831	-0.195	0.981	-0.556
$\cos 6\theta$	0.383	-0.924	0.924	-0.383	-0.383	0.924	-0.924	0.383
$\cos 7\theta$	0.195	-0.556	0.831	-0.981	0.981	-0.831	0.556	-0.195

У результаті вийде вісім векторів $\vec{v}_f$, $f = 0, 1, \ldots, 7$ (всього 64 числа), які представлені в табл. Вони є базисом одновимірного косинус-перетворення. Зазначимо схожість цієї таблиці з матрицею W із рівняння (2).

$$W = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}. \qquad (2)$$

В обох випадках частота зміни знаків зростає рядками.

Можна довести, що всі вектори $\vec{v}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{v}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{v}_i$.

Можна довести, що всі вектори $\vec{v}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{v}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{v}_i$.

Можна довести, що всі вектори $\vec{v}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{v}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{v}_i$.

Можна довести, що всі вектори $\vec{v}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{v}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{v}_i$.

Можна довести, що всі вектори $\vec{v}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{v}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{v}_i$.

Можна довести, що всі вектори $\vec{\boldsymbol{v}}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{\boldsymbol{v}}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{\boldsymbol{v}}_i$.

Можна довести, що всі вектори $\vec{v}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{v}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{v}_i$.

Можна довести, що всі вектори $\vec{\boldsymbol{v}}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{\boldsymbol{v}}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{\boldsymbol{v}}_i$.

Можна довести, що всі вектори $\vec{v}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{v}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{v}_i$.

Можна довести, що всі вектори $\vec{\boldsymbol{v}}_i$ ортогональні між собою (а саме через спеціальний вибір восьми точок відліку θ). Те саме можна виявити прямим обчисленням за допомогою відповідної математичної програми. Отже, ці вісім векторів можна помістити в матрицю розміром 8×8 і розглянути відповідне їй ортогональне перетворення — обертання у восьмивимірному просторі, яке називається *одновимірним дискретним косинус-перетворенням* (DCT). Двовимірне DCT можна також інтерпретувати як подвійне обертання.

Одномірне дискретне косинус-перетворення має іншу інтерпретацію. Можна розглянути векторний простір, базисом якого служать вектори $\vec{\boldsymbol{v}}_i$, і виразити будь-який вектор цього простору у вигляді лінійної комбінації векторів $\vec{\boldsymbol{v}}_i$.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{p} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{p}$ як суму $\vec{p} = \sum_i w_i \vec{v}_i$ восьми векторів $\vec{v}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{p}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{p}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{v}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{p}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{p} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{p}$ як суму $\vec{p} = \sum_i w_i \vec{v}_i$ восьми векторів $\vec{v}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{p}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{p}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{v}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{p}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{p} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{p}$ як суму $\vec{p} = \sum_i w_i \vec{v}_i$ восьми векторів $\vec{v}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{p}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{p}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{v}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{p}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{aligned} &w_0 = 0.506, && w_1 = 0.0143, && w_2 = 0.0115, && w_3 = 0.0439, \\ &w_4 = 0.0795, && w_5 = -0.0432, && w_6 = 0.00478, && w_7 = -0.0077. \end{aligned}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{aligned} &w_0 = 0.506, && w_1 = 0.0143, && w_2 = 0.0115, && w_3 = 0.0439, \\ &w_4 = 0.0795, && w_5 = -0.0432, && w_6 = 0.00478, && w_7 = -0.0077. \end{aligned}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{aligned} w_0 &= 0.506, & w_1 &= 0.0143, & w_2 &= 0.0115, & w_3 &= 0.0439, \\ w_4 &= 0.0795, & w_5 &= -0.0432, & w_6 &= 0.00478, & w_7 &= -0.0077. \end{aligned}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{aligned} &w_0 = 0.506, && w_1 = 0.0143, && w_2 = 0.0115, && w_3 = 0.0439, \\ &w_4 = 0.0795, && w_5 = -0.0432, && w_6 = 0.00478, && w_7 = -0.0077. \end{aligned}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{aligned} &w_0 = 0.506, \quad &&w_1 = 0.0143, \quad &&w_2 = 0.0115, \quad &&w_3 = 0.0439, \\ &w_4 = 0.0795, \quad &&w_5 = -0.0432, \quad &&w_6 = 0.00478, \quad &&w_7 = -0.0077. \end{aligned}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum\limits_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{p} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{p}$ як суму $\vec{p} = \sum_i w_i \vec{v}_i$ восьми векторів $\vec{v}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{p}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{p}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{v}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{p}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Наприклад, виберемо 8 (корельованих) чисел

$$\vec{\boldsymbol{p}} = (0.6, 0.5, 0.4, 0.5, 0.6, 0.5, 0.4, 0.55)$$

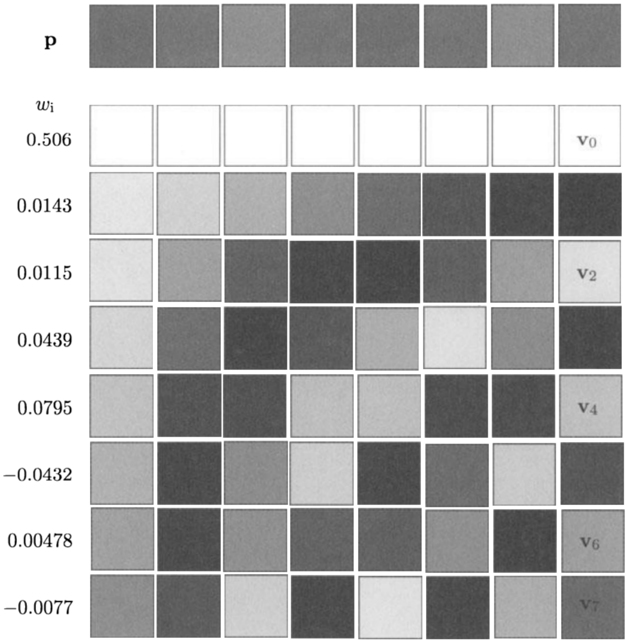
в якості тестових даних. Виразимо вектор $\vec{\boldsymbol{p}}$ як суму $\vec{\boldsymbol{p}} = \sum_i w_i \vec{\boldsymbol{v}}_i$ восьми векторів $\vec{\boldsymbol{v}}_i$. Розв'язавши цю систему з 8 лінійних рівнянь, знаходимо вісім ваг

$$\begin{array}{llll} w_0 = 0.506, & w_1 = 0.0143, & w_2 = 0.0115, & w_3 = 0.0439, \\ w_4 = 0.0795, & w_5 = -0.0432, & w_6 = 0.00478, & w_7 = -0.0077. \end{array}$$

Вага w_0 не сильно відрізняється від елементів вектора $\vec{\boldsymbol{p}}$, але решта сім ваг набагато менша. Це показує, як DCT або будь-яке інше ортогональне перетворення) робить стиснення. Тепер можна просто записати ці вісім ваг у стислий файл, де вони займатимуть менше місця, ніж вісім компонентів вихідного вектора $\vec{\boldsymbol{p}}$. Квантування ваг w_i може суттєво підвищити фактор стиснення, причому при дуже малій втраті даних.

Рис. ілюструє цю лінійну комбінацію графічно. Всі вісім векторів $\vec{\boldsymbol{v}}_i$ зображені у вигляді ряду з восьми маленьких сірих квадратиків, причому значення $+1$ представлене білим кольором, а значення -1 пофарбовано в чорний колір. Кожна із восьми компонентів вектора $\vec{\boldsymbol{p}}$ виражена у вигляді виваженої суми восьми сірих відтінків.

Стиснення зображень. Дискретне косинус-перетворення



На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \tag{3}$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \tag{4}$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2}\, C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \dfrac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \dots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \dots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \tag{3}$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \tag{4}$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \dfrac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2j+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів).

Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

На практиці одномірне DCT простіше всього обчислювати за формулою

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right), \qquad (3)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Тут вихідними даними (пікселями, фрагментами звуку чи іншими елементами) є величини p_t, а їм відповідними коефіцієнтами DCT є числа G_f. Формула (3) дуже проста, але процес обчислення нею повільний. Декодер отримує на вході коефіцієнти DCT, ділить їх на вісімки та застосовує до них *обернене перетворення* DCT (*inverse* DCT, IDCT) для відновлення вихідних даних (теж у вигляді груп по 8 елементів). Найпростіша формула для обчислення IDCT має вигляд

$$p_t = \frac{1}{2} \sum_{j=0}^{7} C_j G_j \cos\left(\frac{(2t+1)j\pi}{16}\right), \qquad \text{для} \quad t = 0, 1, \ldots, 7. \qquad (4)$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{p} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{p} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{p} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{\boldsymbol{p}} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{\boldsymbol{p}} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{\boldsymbol{p}} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{p} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{\boldsymbol{p}} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{\boldsymbol{p}} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{\boldsymbol{p}} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{p} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Стиснення зображень. Дискретне косинус-перетворення

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{p} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Наступний приклад демонструє переваги методу DCT. Розглянемо множину, яка складається з 8 величин (вихідних даних)

$$\vec{\boldsymbol{p}} = (12, 10, 8, 10, 12, 10, 8, 11),$$

застосуємо до них DCT і отримаємо вісім коефіцієнтів

$$28.6375, 0.571202, 0.46194, 1.757, 3.18198, -1.72956, 0.191342, -0.308709.$$

Ці числа можна використовувати для точного відновлення вихідних даних (з невеликою помилкою, викликаною обмеженням на точність комп'ютерних обчислень). Наша мета, однак, покращити стиснення за допомогою відповідного квантування коефіцієнтів. Округлюємо (квантуємо) їх до

$$28.6, 0.6, 0.5, 1.8, 3.2, -1.8, 0.2, -0.3,$$

застосовуємо IDCT та отримуємо

$$12.0254, 10.0233, 7.96054, 9.93097, 12.0164, 9.99321, 7.94354, 10.9989.$$

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ще раз квантуємо коефіцієнти:

$$28, 1, 1, 2, 3, -2, 0, 0,$$

і знову отримуємо за допомогою IDCT такий результат:

$$12.1883, 10.2315, 7.74931, 9.20863, 11.7876, 9.54549, 7.82865, 10.6557.$$

Нарешті квантуємо коефіцієнти до

$$28, 0, 0, 2, 3, -2, 0, 0,$$

і отримуємо за допомогою IDCT послідовність

$$11.236, 9.62443, 7.66286, 9.57302, 12.3471, 10.0146, 8.05304, 10.6842,$$

у якій найбільша різниця між вихідним значенням (12) та реконструйованим (11.236) дорівнює 0.764 (або 6.4% від 12).

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,
(2) тільки чотири з них не дорівнюють нулю,
(3) нульові коефіцієнти утворюють серії,
(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Ці прості приклади показують переваги методу DCT. Множина

$$28, 0, 0, 2, 3, -2, 0, 0$$

грубо квантованих коефіцієнтів DCT має чотири властивості, які роблять його ідеальним для стиснення, причому із чудовою декомпресією за малої втрати даних. Ось ці чотири властивості:

(1) множина складається тільки з цілих чисел,

(2) тільки чотири з них не дорівнюють нулю,

(3) нульові коефіцієнти утворюють серії,

(4) серед ненульових коефіцієнтів тільки перший має велику величину; решта менше вихідних чисел.

Ці властивості можна використовувати при реалізації схеми RLE, методу Гаффмана або будь-якої іншої техніки.

Стиснення зображень. Дискретне косинус-перетворення

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{i} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{i} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад

Одновимірне DCT (рівняння (3))

$$G_f = \frac{1}{2} C_f \sum_{t=0}^{7} p_t \cos\left(\frac{(2t+1)f\pi}{16}\right),$$

восьми корельованих величин

$$11, 22, 33, 44, 55, 66, 77 \quad \text{і} \quad 88$$

зробить вісім коефіцієнтів

$$140, -71, 0, -7, 0, -2, 0, 0.$$

Після квантування отримуємо множину

$$140, -71, 0, 0, 0, 0, 0, 0$$

і застосовуємо IDCT. В результаті отримуємо:

$$15, 20, 30, 43, 56, 69, 79 \quad \text{і} \quad 84.$$

Ці числа дуже близькі до вихідних. Найбільша розбіжність дорівнює 4.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: "Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!" Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: “Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!” Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: "Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!" Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: “Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!” Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: “Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!” Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: “Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!” Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \dots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: "Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!" Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: “Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!” Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: "Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!" Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: “Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!” Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: "Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!" Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: "Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!" Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: “Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!” Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корельованих даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Приклад (продовження)

Дійшовши до цього місця, натхненний слухач може вигукнути: “Дивно! Вісім вихідних даних відновлюються лише за допомогою двох чисел. Дива якісь!” Однак ті, хто зрозумів властивості перетворень, можуть дати просте пояснення. Відновлення даних відбувається не лише за двома числами 140 і -71, але також за їхнім розташуванням у послідовності з 8 коефіцієнтів. Крім того, вихідні величини відновлюються з високою точністю завдяки присутності надмірності.

Граничним випадком надлишкових даних є послідовність однакових величин. Вони, звичайно, мають досконалу кореляцію, і ми відчуваємо інтуїтивно, що одного числа буде достатньо для їхнього відновлення. Реконструкція послідовності високо корелюваних даних, такої, як $20, 31, 42, 53, \ldots$ вимагатиме всього двох чисел. Ними можуть бути початкове значення (20) та крок (11) (різниця цієї арифметичної прогресії), але можуть бути й інші числа. У разі, що менше корельовані дані, тим більше чисел знадобиться їхнє відновлення.

Стиснення зображень. Двовимірне (матричне) DCT

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \dfrac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Стиснення зображень. Двовимірне (матричне) DCT

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Стиснення зображень. Двовимірне (матричне) DCT

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Стиснення зображень. Двовимірне (матричне) DCT

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \tag{5}$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \tag{6}$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Стиснення зображень. Двовимірне (матричне) DCT

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Стиснення зображень. Двовимірне (матричне) DCT

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (пікселі корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \dfrac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Стиснення зображень. Двовимірне (матричне) DCT

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються.

Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Стиснення зображень. Двовимірне (матричне) DCT

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

З досвіду добре відомо, що пікселі зображення мають кореляцію за двома напрямками, а не лише по одному (піксели корелюють зі своїми сусідами ліворуч, праворуч, а також зверху та знизу). Тому методи стиснення зображень використовують двовимірне DCT, яке визначається за формулою

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (5)$$

при $0 \leqslant i, j \leqslant n-1$. Зображення розбивається на блоки пікселів p_{xy} розміру $n \times n$ (у нашому прикладі $n = 8$), і рівняння (5) використовуються для знаходження коефіцієнтів G_{ij} для кожного блоку пікселів. Якщо допускається часткова втрата інформації, то коефіцієнти квантуються. Декодер відновлює стислий блок даних (точно або приблизно), обчислюючи обернене DCT (IDCT) за формулою

$$p_{xy} = \frac{1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right), \qquad (6)$$

де

$$C_f = \begin{cases} \frac{1}{\sqrt{2}}, & \text{якщо } f = 0; \\ 1, & \text{якщо } f = 1, 2, \ldots, 7. \end{cases}$$

Стиснення зображень. Двовимірне (матричне) DCT

Двовимірне DCT можна інтерпретувати двома способами: за допомогою обертання (насправді композиції двох обертань), і за допомогою базису в n-вимірному векторному просторі. У першій інтерпретації використовується блок $n \times n$ пікселів (рис. де елементи позначені літерою “L”).

$$
\begin{array}{cccccccc}
\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}\\
\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}\\
\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}\\
\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}\\
\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}\\
\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}\\
\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}\\
\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}&\mathrm{L}
\end{array}
\qquad
\begin{array}{cccccccc}
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}\\
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}\\
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}\\
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}\\
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}\\
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}\\
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}\\
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}
\end{array}
$$

(a) (b)

$$
\begin{array}{cccccccc}
\mathrm{L}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}&\mathrm{S}\\
\mathrm{S}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}\\
\mathrm{S}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}\\
\mathrm{S}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}\\
\mathrm{S}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}\\
\mathrm{S}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}\\
\mathrm{S}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}\\
\mathrm{S}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}&\mathrm{s}
\end{array}
$$

(c)

Двовимірне DCT та подвійне обертання

Стиснення зображень. Двовимірне (матричне) DCT

Двовимірне DCT можна інтерпретувати двома способами: за допомогою обертання (насправді композиції двох обертань), і за допомогою базису в n-вимірному векторному просторі. У першій інтерпретації використовується блок $n \times n$ пікселів (рис. де елементи позначені літерою “L”).

L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L

(a)

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Двовимірне DCT та подвійне обертання

Стиснення зображень. Двовимірне (матричне) DCT

Двовимірне DCT можна інтерпретувати двома способами: за допомогою обертання (насправді композиції двох обертань), і за допомогою базису в n-вимірному векторному просторі. У першій інтерпретації використовується блок $n \times n$ пікселів (рис. де елементи позначені літерою "L").

L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L

(a)

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Двовимірне DCT та подвійне обертання

Стиснення зображень. Двовимірне (матричне) DCT

Двовимірне DCT можна інтерпретувати двома способами: за допомогою обертання (насправді композиції двох обертань), і за допомогою базису в n-вимірному векторному просторі. У першій інтерпретації використовується блок $n \times n$ пікселів (рис. де елементи позначені літерою “L”).

L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L

(a)

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Двовимірне DCT та подвійне обертання

Стиснення зображень. Двовимірне (матричне) DCT

Двовимірне DCT можна інтерпретувати двома способами: за допомогою обертання (насправді композиції двох обертань), і за допомогою базису в n-вимірному векторному просторі. У першій інтерпретації використовується блок $n \times n$ пікселів (рис. де елементи позначені літерою “L”).

L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L

(a)

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Двовимірне DCT та подвійне обертання

Стиснення зображень. Двовимірне (матричне) DCT

Двовимірне DCT можна інтерпретувати двома способами: за допомогою обертання (насправді композиції двох обертань), і за допомогою базису в n-вимірному векторному просторі. У першій інтерпретації використовується блок $n \times n$ пікселів (рис. де елементи позначені літерою “L”).

L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L
L	L	L	L	L	L	L	L

(a)

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Двовимірне DCT та подвійне обертання

Спочатку розглядаються рядки цього блоку як точки

$$(p_{x,0}, p_{x,1}, \dots p_{x,n-1})$$

у n-вимірному просторі, які повертаються в цьому просторі за допомогою перетворення, що задається внутрішньою сумою

$$G1_{x,j} = C_j \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right)$$

з рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right). \qquad (5)$$

Спочатку розглядаються рядки цього блоку як точки

$$(p_{x,0}, p_{x,1}, \ldots p_{x,n-1})$$

у n-вимірному просторі, які повертаються в цьому просторі за допомогою перетворення, що задається внутрішньою сумою

$$G1_{x,j} = C_j \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right)$$

з рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right). \qquad (5)$$

Спочатку розглядаються рядки цього блоку як точки

$$(p_{x,0}, p_{x,1}, \dots p_{x,n-1})$$

у n-вимірному просторі, які повертаються в цьому просторі за допомогою перетворення, що задається внутрішньою сумою

$$G1_{x,j} = C_j \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right)$$

з рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right). \qquad (5)$$

Стиснення зображень. Двовимірне (матричне) DCT

Спочатку розглядаються рядки цього блоку як точки

$$(p_{x,0}, p_{x,1}, \dots p_{x,n-1})$$

у n-вимірному просторі, які повертаються в цьому просторі за допомогою перетворення, що задається внутрішньою сумою

$$G1_{x,j} = C_j \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right)$$

з рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right). \qquad (5)$$

Спочатку розглядаються рядки цього блоку як точки

$$(p_{x,0}, p_{x,1}, \ldots p_{x,n-1})$$

у n-вимірному просторі, які повертаються в цьому просторі за допомогою перетворення, що задається внутрішньою сумою

$$G1_{x,j} = C_j \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right)$$

з рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right). \qquad (5)$$

Спочатку розглядаються рядки цього блоку як точки

$$(p_{x,0}, p_{x,1}, \dots p_{x,n-1})$$

у n-вимірному просторі, які повертаються в цьому просторі за допомогою перетворення, що задається внутрішньою сумою

$$G1_{x,j} = C_j \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right)$$

з рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right). \quad (5)$$

Спочатку розглядаються рядки цього блоку як точки

$$(p_{x,0}, p_{x,1}, \ldots p_{x,n-1})$$

у n-вимірному просторі, які повертаються в цьому просторі за допомогою перетворення, що задається внутрішньою сумою

$$G1_{x,j} = C_j \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right)$$

з рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right). \qquad (5)$$

Результатом цього обертання служить блок $G1_{x,j}$ з $n \times n$ коефіцієнтів, у якому рядках домінують перші елементи (позначені як "L" рис. (b)), а всі інші елементи малі (вони позначені цьому малюнку як "S").

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

Зовнішня сума рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

дорівнює

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i \sum_{x=0}^{n-1} p_{xy} G1_{x,j} \cos\left(\frac{(2x+1)i\pi}{2n}\right).$$

Стиснення зображень. Двовимірне (матричне) DCT

Результатом цього обертання служить блок $G1_{x,j}$ з $n \times n$ коефіцієнтів, у якому рядках домінують перші елементи (позначені як "L" рис. (b)), а всі інші елементи малі (вони позначені цьому малюнку як "S").

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

Зовнішня сума рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

дорівнює

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i \sum_{x=0}^{n-1} p_{xy} G1_{x,j} \cos\left(\frac{(2x+1)i\pi}{2n}\right).$$

Стиснення зображень. Двовимірне (матричне) DCT

Результатом цього обертання служить блок $G1_{x,j}$ з $n \times n$ коефіцієнтів, у якому рядках домінують перші елементи (позначені як "L" рис. (b)), а всі інші елементи малі (вони позначені цьому малюнку як "S").

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

Зовнішня сума рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

дорівнює

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i \sum_{x=0}^{n-1} p_{xy} G1_{x,j} \cos\left(\frac{(2x+1)i\pi}{2n}\right).$$

Результатом цього обертання служить блок $G1_{x,j}$ з $n \times n$ коефіцієнтів, у якому рядках домінують перші елементи (позначені як “L” рис. (b)), а всі інші елементи малі (вони позначені цьому малюнку як “S”).

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

Зовнішня сума рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

дорівнює

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i \sum_{x=0}^{n-1} p_{xy} G1_{x,j} \cos\left(\frac{(2x+1)i\pi}{2n}\right).$$

Результатом цього обертання служить блок $G1_{x,j}$ з $n \times n$ коефіцієнтів, у якому рядках домінують перші елементи (позначені як "L" рис. (b)), а всі інші елементи малі (вони позначені цьому малюнку як "S").

$$\begin{matrix} \mathsf{L} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} \\ \mathsf{L} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} \\ \mathsf{L} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} \\ \mathsf{L} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} \\ \mathsf{L} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} \\ \mathsf{L} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} \\ \mathsf{L} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} \\ \mathsf{L} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} & \mathsf{S} \end{matrix}$$

(b)

Зовнішня сума рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

дорівнює

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i \sum_{x=0}^{n-1} p_{xy} G1_{x,j} \cos\left(\frac{(2x+1)i\pi}{2n}\right).$$

Результатом цього обертання служить блок $G1_{x,j}$ з $n \times n$ коефіцієнтів, у якому рядках домінують перші елементи (позначені як "L" рис. (b)), а всі інші елементи малі (вони позначені цьому малюнку як "S").

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

Зовнішня сума рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \tag{5}$$

дорівнює

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i \sum_{x=0}^{n-1} p_{xy} G1_{x,j} \cos\left(\frac{(2x+1)i\pi}{2n}\right).$$

Результатом цього обертання служить блок $G1_{x,j}$ з $n \times n$ коефіцієнтів, у якому рядках домінують перші елементи (позначені як “L” рис. (b)), а всі інші елементи малі (вони позначені цьому малюнку як “S”).

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

Зовнішня сума рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

дорівнює

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i \sum_{x=0}^{n-1} p_{xy} G1_{x,j} \cos\left(\frac{(2x+1)i\pi}{2n}\right).$$

Стиснення зображень. Двовимірне (матричне) DCT

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це “L”) і $n-1$ маленьких коефіцієнтів в інших місцях (“S” і “s” на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

Стиснення зображень. Двовимірне (матричне) DCT

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це “L”) і $n-1$ маленьких коефіцієнтів в інших місцях (“S” і “s” на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

Стиснення зображень. Двовимірне (матричне) DCT

$$\begin{array}{cccccccc} L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \end{array} \qquad \begin{array}{cccccccc} L & S & S & S & S & S & S & S \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \end{array}$$

(b) (c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це “L”) і $n-1$ маленьких коефіцієнтів в інших місцях (“S” і “s” на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

Стиснення зображень. Двовимірне (матричне) DCT

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це "L") і $n-1$ маленьких коефіцієнтів в інших місцях ("S" і "s" на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

Стиснення зображень. Двовимірне (матричне) DCT

$$
\begin{array}{cccccccc}
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} \\
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} \\
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} \\
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} \\
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} \\
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} \\
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} \\
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S}
\end{array}
$$

(b)

$$
\begin{array}{cccccccc}
\mathrm{L} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} & \mathrm{S} \\
\mathrm{S} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} \\
\mathrm{S} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} \\
\mathrm{S} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} \\
\mathrm{S} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} \\
\mathrm{S} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} \\
\mathrm{S} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} \\
\mathrm{S} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s} & \mathrm{s}
\end{array}
$$

(c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це “L”) і $n-1$ маленьких коефіцієнтів в інших місцях (“S” і “s” на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

Стиснення зображень. Двовимірне (матричне) DCT

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це “L”) і $n-1$ маленьких коефіцієнтів в інших місцях (“S” і “s” на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це “L”) і $n-1$ маленьких коефіцієнтів в інших місцях (“S” і “s” на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S

(b)

L	S	S	S	S	S	S	S
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s
S	s	s	s	s	s	s	s

(c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це “L”) і $n-1$ маленьких коефіцієнтів в інших місцях (“S” і “s” на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

$$\begin{array}{cccccccc} L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \\ L & S & S & S & S & S & S & S \end{array} \qquad \begin{array}{cccccccc} L & S & S & S & S & S & S & S \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \\ S & s & s & s & s & s & s & s \end{array}$$

(b) (c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це “L”) і $n-1$ маленьких коефіцієнтів в інших місцях (“S” і “s” на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

L	S	S	S	S	S	S	S	L	S	S	S	S	S	S	S
L	S	S	S	S	S	S	S	S	s	s	s	s	s	s	s
L	S	S	S	S	S	S	S	S	s	s	s	s	s	s	s
L	S	S	S	S	S	S	S	S	s	s	s	s	s	s	s
L	S	S	S	S	S	S	S	S	s	s	s	s	s	s	s
L	S	S	S	S	S	S	S	S	s	s	s	s	s	s	s
L	S	S	S	S	S	S	S	S	s	s	s	s	s	s	s
L	S	S	S	S	S	S	S	S	s	s	s	s	s	s	s

(b) (c)

Тут вже стовпці матриці $G1_{x,j}$ розглядаються як точки n-мірного векторного простору, над якими відбувається перетворення обертання. В результаті виходить один великий коефіцієнт у верхньому лівому куті блоку (на рис. (b) — це "L") і $n-1$ маленьких коефіцієнтів в інших місцях ("S" і "s" на рис. (c)). Ця інтерпретація розглядає двовимірне DCT у вигляді двох різних обертань розмірності n. Цікаво відзначити, що два обертання розмірності n обчислюються швидше, ніж одне обертання розмірності n, оскільки друге вимагає матрицю разміру $n^2 \times n^2$.

Друга інтерпретація (при $n = 8$) використовує рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

до створення 64 блоків по 8×8 величин у кожному. Всі 64 блоки розглядаються як базис 64-мірного векторного простору (це базисні зображення). Будь-який блок B з 8×8 пікселів можна виразити як лінійну комбінацію цих базисних зображень, і всі 64 ваги цієї лінійної комбінації утворюють коефіцієнти DCT блоку.

На рис. зображено графічне уявлення 64 базисних образів двовимірного DCT при $n = 8$.

Друга інтерпретація (при $n = 8$) використовує рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \tag{5}$$

до створення 64 блоків по 8×8 величин у кожному. Всі 64 блоки розглядаються як базис 64-мірного векторного простору (це базисні зображення). Будь-який блок B з 8×8 пікселів можна виразити як лінійну комбінацію цих базисних зображень, і всі 64 ваги цієї лінійної комбінації утворюють коефіцієнти DCT блоку.

На рис. зображено графічне уявлення 64 базисних образів двовимірного DCT при $n = 8$.

Друга інтерпретація (при $n = 8$) використовує рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

до створення 64 блоків по 8×8 величин у кожному. Всі 64 блоки розглядаються як базис 64-мірного векторного простору (це базисні зображення). Будь-який блок B з 8×8 пікселів можна виразити як лінійну комбінацію цих базисних зображень, і всі 64 ваги цієї лінійної комбінації утворюють коефіцієнти DCT блоку.

На рис. зображено графічне уявлення 64 базисних образів двовимірного DCT при $n = 8$.

Друга інтерпретація (при $n = 8$) використовує рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

до створення 64 блоків по 8×8 величин у кожному. Всі 64 блоки розглядаються як базис 64-мірного векторного простору (це базисні зображення). Будь-який блок B з 8×8 пікселів можна виразити як лінійну комбінацію цих базисних зображень, і всі 64 ваги цієї лінійної комбінації утворюють коефіцієнти DCT блоку.

На рис. зображено графічне уявлення 64 базисних образів двовимірного DCT при $n = 8$.

Друга інтерпретація (при $n = 8$) використовує рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

до створення 64 блоків по 8×8 величин у кожному. Всі 64 блоки розглядаються як базис 64-мірного векторного простору (це базисні зображення). Будь-який блок B з 8×8 пікселів можна виразити як лінійну комбінацію цих базисних зображень, і всі 64 ваги цієї лінійної комбінації утворюють коефіцієнти DCT блоку.

На рис. зображено графічне уявлення 64 базисних образів двовимірного DCT при $n = 8$.

Друга інтерпретація (при $n = 8$) використовує рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

до створення 64 блоків по 8×8 величин у кожному. Всі 64 блоки розглядаються як базис 64-мірного векторного простору (це базисні зображення). Будь-який блок B з 8×8 пікселів можна виразити як лінійну комбінацію цих базисних зображень, і всі 64 ваги цієї лінійної комбінації утворюють коефіцієнти DCT блоку.

На рис. зображено графічне уявлення 64 базисних образів двовимірного DCT при $n = 8$.

Стиснення зображень. Двовимірне (матричне) DCT

Друга інтерпретація (при $n = 8$) використовує рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

до створення 64 блоків по 8×8 величин у кожному. Всі 64 блоки розглядаються як базис 64-мірного векторного простору (це базисні зображення). Будь-який блок B з 8×8 пікселів можна виразити як лінійну комбінацію цих базисних зображень, і всі 64 ваги цієї лінійної комбінації утворюють коефіцієнти DCT блоку.

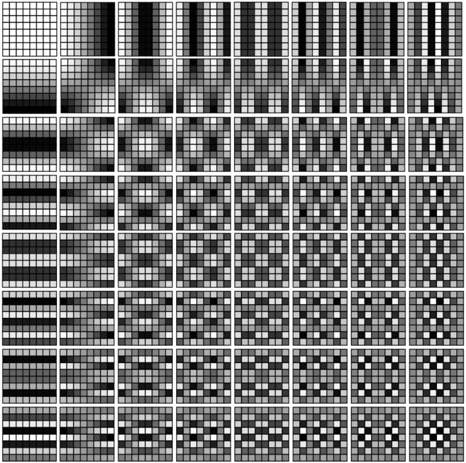
На рис. зображено графічне уявлення 64 базисних образів двовимірного DCT при $n = 8$.

Друга інтерпретація (при $n = 8$) використовує рівняння (5)

$$G_{ij} = \frac{1}{\sqrt{2n}} C_i C_j \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} p_{xy} \cos\left(\frac{(2y+1)j\pi}{2n}\right) \cos\left(\frac{(2x+1)i\pi}{2n}\right) \qquad (5)$$

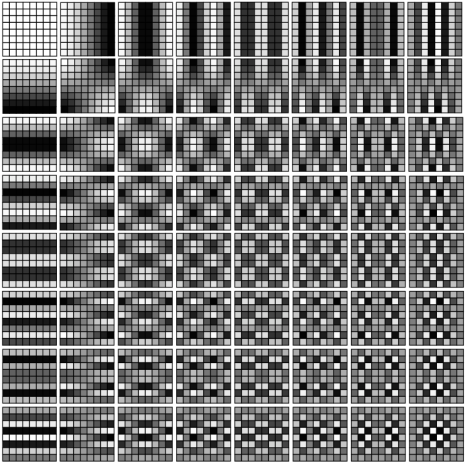
до створення 64 блоків по 8×8 величин у кожному. Всі 64 блоки розглядаються як базис 64-мірного векторного простору (це базисні зображення). Будь-який блок B з 8×8 пікселів можна виразити як лінійну комбінацію цих базисних зображень, і всі 64 ваги цієї лінійної комбінації утворюють коефіцієнти DCT блоку.
На рис. зображено графічне уявлення 64 базисних образів двовимірного DCT при $n = 8$.

Стиснення зображень. Двовимірне (матричне) DCT



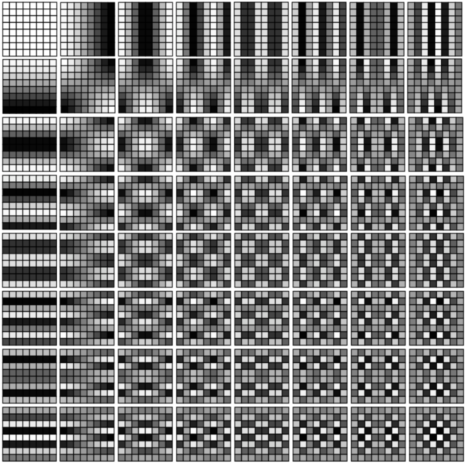
Кожен елемент (i, j) на цьому рисунку є блоком розміру 8×8, який отриманий добутком $\cos(i \cdot s)\cos(j \cdot t)$, де s і t — змінюються незалежно в межах, зазначених у рівнянні (1)

$$\frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}. \tag{1}$$



Кожен елемент (i, j) на цьому рисунку є блоком розміру 8×8, який отриманий добутком $\cos(i \cdot s)\cos(j \cdot t)$, де s і t — змінюються незалежно в межах, зазначених у рівнянні (1)

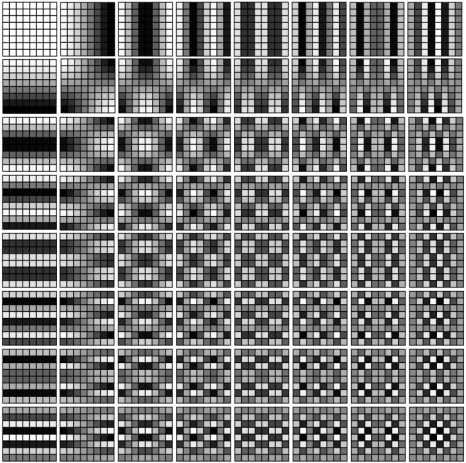
$$\frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}. \tag{1}$$



Кожен елемент (i, j) на цьому рисунку є блоком розміру 8×8, який отриманий добутком $\cos(i \cdot s)\cos(j \cdot t)$, де s і t — змінюються незалежно в межах, зазначених у рівнянні (1)

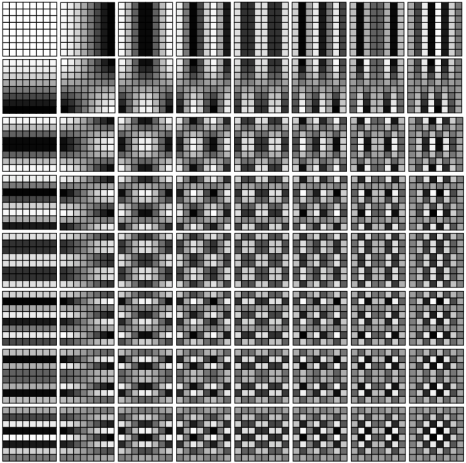
$$\frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}. \tag{1}$$

Стиснення зображень. Двовимірне (матричне) DCT



Кожен елемент (i, j) на цьому рисунку є блоком розміру 8×8, який отриманий добутком $\cos(i \cdot s)\cos(j \cdot t)$, де s і t — змінюються незалежно в межах, зазначених у рівнянні (1)

$$\frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}. \qquad (1)$$



Кожен елемент (i, j) на цьому рисунку є блоком розміру 8×8, який отриманий добутком $\cos(i \cdot s)\cos(j \cdot t)$, де s і t — змінюються незалежно в межах, зазначених у рівнянні (1)

$$\frac{\pi}{16}, \frac{3\pi}{16}, \frac{5\pi}{16}, \frac{7\pi}{16}, \frac{9\pi}{16}, \frac{11\pi}{16}, \frac{13\pi}{16}, \frac{15\pi}{16}. \qquad (1)$$

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

Використовуючи відповідне програмне забезпечення, легко виконати обчислення DCT та відобразити результати графічно. На рис. (a) наведена випадкова матриця 8×8 з нулів та одиниць. Ця матриця зображено на рис. (b) за допомогою білих та чорних квадратиків, що позначають 1 та 0, відповідно. На рис. (c) зображено чисельні значення ваги, на які слід помножити кожен з 64 коефіцієнтів DCT для того, щоб відновити вихідну матрицю.

Стиснення зображень. Двовимірне (матричне) DCT

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

Використовуючи відповідне програмне забезпечення, легко виконати обчислення DCT та відобразити результати графічно. На рис. (a) наведена випадкова матриця 8×8 з нулів та одиниць. Ця матриця зображено на рис. (b) за допомогою білих та чорних квадратиків, що позначають 1 та 0, відповідно. На рис. (c) зображено чисельні значення ваги, на які слід помножити кожен з 64 коефіцієнтів DCT для того, щоб відновити вихідну матрицю.

Стиснення зображень. Двовимірне (матричне) DCT

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

Використовуючи відповідне програмне забезпечення, легко виконати обчислення DCT та відобразити результати графічно. На рис. (a) наведена випадкова матриця 8×8 з нулів та одиниць. Ця матриця зображено на рис. (b) за допомогою білих та чорних квадратиків, що позначають 1 та 0, відповідно. На рис. (c) зображено чисельні значення ваги, на які слід помножити кожен з 64 коефіцієнтів DCT для того, щоб відновити вихідну матрицю.

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

Використовуючи відповідне програмне забезпечення, легко виконати обчислення DCT та відобразити результати графічно. На рис. (a) наведена випадкова матриця 8×8 з нулів та одиниць. Ця матриця зображено на рис. (b) за допомогою білих та чорних квадратиків, що позначають 1 та 0, відповідно. На рис. (c) зображено чисельні значення ваги, на які слід помножити кожен з 64 коефіцієнтів DCT для того, щоб відновити вихідну матрицю.

Стиснення зображень. Двовимірне (матричне) DCT

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

Використовуючи відповідне програмне забезпечення, легко виконати обчислення DCT та відобразити результати графічно. На рис. (a) наведена випадкова матриця 8×8 з нулів та одиниць. Ця матриця зображено на рис. (b) за допомогою білих та чорних квадратиків, що позначають 1 та 0, відповідно. На рис. (c) зображено чисельні значення ваги, на які слід помножити кожен з 64 коефіцієнтів DCT для того, щоб відновити вихідну матрицю.

Стиснення зображень. Двовимірне (матричне) DCT

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

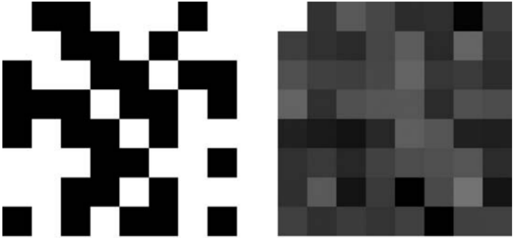
4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

Використовуючи відповідне програмне забезпечення, легко виконати обчислення DCT та відобразити результати графічно. На рис. (a) наведена випадкова матриця 8×8 з нулів та одиниць. Ця матриця зображено на рис. (b) за допомогою білих та чорних квадратиків, що позначають 1 та 0, відповідно. На рис. (c) зображено чисельні значення ваги, на які слід помножити кожен з 64 коефіцієнтів DCT для того, щоб відновити вихідну матрицю.

Стиснення зображень. Двовимірне (матричне) DCT

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010



(a) (b) (c)

4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

На цьому рисунку нуль показаний нейтрально-сірим кольором, додатні числа — світло-сірим, а від'ємні — темним. На рис. (d) викладено чисельні значення цих ваг.

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

На цьому рисунку нуль показаний нейтрально-сірим кольором, додатні числа — світло-сірим, а від'ємні — темним. На рис. (d) викладено чисельні значення цих ваг.

Стиснення зображень. Двовимірне (матричне) DCT

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

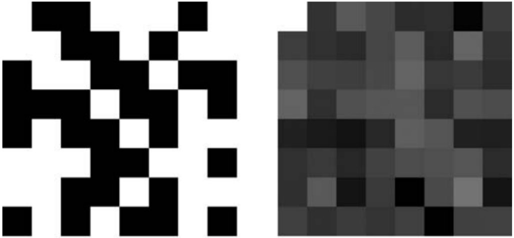
4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

На цьому рисунку нуль показаний нейтрально-сірим кольором, додатні числа — світло-сірим, а від'ємні — темним. На рис. (d) викладено чисельні значення цих ваг.

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)



(b) (c)

4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

На цьому рисунку нуль показаний нейтрально-сірим кольором, додатні числа — світло-сірим, а від'ємні — темним. На рис. (d) викладено чисельні значення цих ваг.

10011101
11001011
01100100
00010010
01001011
11100110
11001011
01010010

(a)

(b)

(c)

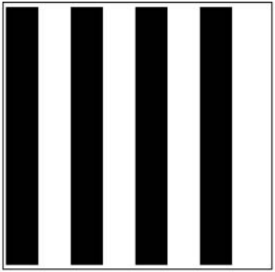
4.000	−0.133	0.637	0.272	−0.250	−0.181	−1.076	0.026
0.081	−0.178	−0.300	0.230	0.694	−0.309	0.875	−0.127
0.462	0.125	0.095	0.291	0.868	−0.070	0.021	−0.280
0.837	−0.194	0.455	0.583	0.588	−0.281	0.448	0.383
−0.500	−0.635	−0.749	−0.346	0.750	0.557	−0.502	−0.540
−0.167	0	−0.366	0.146	0.393	0.448	0.577	−0.268
−0.191	0.648	−0.729	−0.008	−1.171	0.306	1.155	−0.744
0.122	−0.200	0.038	−0.118	0.138	−1.154	0.134	0.148

(d)

На цьому рисунку нуль показаний нейтрально-сірим кольором, додатні числа — світло-сірим, а від'ємні — темним. На рис. (d) викладено чисельні значення цих ваг.

01010101
01010101
01010101
01010101
01010101
01010101
01010101
01010101

(a)



(b)

(c)

$$
\begin{array}{cccccccc}
4.000 & -0.721 & 0 & -0.850 & 0 & -1.273 & 0 & -3.625 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
$$

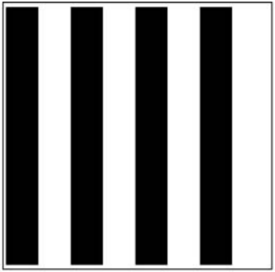
(d)

На рис. зроблені самі побудови, але для застосування у випадку більш регулярних вихідних даних.

Стиснення зображень. Двовимірне (матричне) DCT

01010101
01010101
01010101
01010101
01010101
01010101
01010101
01010101

(a)



(b)

(c)

$$
\begin{array}{cccccccc}
4.000 & -0.721 & 0 & -0.850 & 0 & -1.273 & 0 & -3.625 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
$$

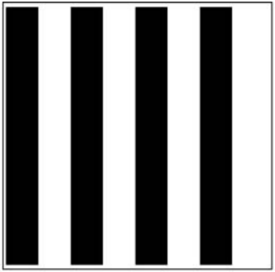
(d)

На рис. зроблені самі побудови, але для застосування у випадку більш регулярних вихідних даних.

Стиснення зображень. Двовимірне (матричне) DCT

01010101
01010101
01010101
01010101
01010101
01010101
01010101
01010101

(a)



(b)

(c)

4.000	−0.721	0	−0.850	0	−1.273	0	−3.625
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(d)

На рис. зроблені самі побудови, але для застосування у випадку більш регулярних вихідних даних.

Стиснення зображень. Двовимірне (матричне) DCT

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

Стиснення зображень. Двовимірне (матричне) DCT

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

Стиснення зображень. Двовимірне (матричне) DCT

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

Стиснення зображень. Двовимірне (матричне) DCT

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

Стиснення зображень. Двовимірне (матричне) DCT

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

Стиснення зображень. Двовимірне (матричне) DCT

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

Стиснення зображень. Двовимірне (матричне) DCT

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

Стиснення зображень. Двовимірне (матричне) DCT

Тепер продемонструємо переваги двовимірного DCT стосовно двох блоків чисел. Перший блок (табл. ліворуч) складається з сильно корелюваних цілих чисел в інтервалі $[8, 12]$,

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

а другий (табл. ліворуч) утворений випадковими числами того ж інтервалу.

8	10	9	11	11	9	9	12	79.12	0.98	0.64	-1.51	-0.62	-0.86	1.22	0.32
11	8	12	8	11	10	11	10	0.15	-1.64	-0.09	1.23	0.10	3.29	1.08	-2.97
9	11	9	10	12	9	9	8	-1.26	-0.29	-3.27	1.69	-0.51	1.13	1.52	1.33
9	12	10	8	8	9	8	9	-1.27	-0.25	-0.67	-0.15	1.63	-1.94	0.47	-1.30
12	8	9	9	12	10	8	11	-2.12	-0.67	-0.07	-0.79	0.13	-1.40	0.16	-0.15
8	11	10	12	9	12	12	10	-2.68	1.08	-1.99	-1.93	-1.77	-0.35	0	-0.80
10	10	12	10	12	10	10	12	1.20	2.10	-0.98	0.87	-1.55	-0.59	-0.98	2.76
12	9	11	11	9	8	8	12	-2.24	0.55	0.29	0.75	-2.40	-0.05	0.06	1.14

Двовимірне DCT блоку випадкових величин

Перший блок породжує один великий коефіцієнт DC, за яким йдуть маленькі (включаючи 20 нульових) коефіцієнтів AC. Серед коефіцієнтів DCT другого, випадкового, блоку є лише один нуль.

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

Видно, чому пікселі у табл. корельовані. Усі вісім чисел верхнього рядка таблиці близькі один до одного (відстань між ними дорівнюють 2 або 3). Всі інші рядки виходять циклічним зсувом праворуч попереднього рядка.

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

Видно, чому пікселі у табл. корельовані. Усі вісім чисел верхнього рядка таблиці близькі один до одного (відстань між ними дорівнюють 2 або 3). Всі інші рядки виходять циклічним зсувом праворуч попереднього рядка.

Стиснення зображень. Двовимірне (матричне) DCT

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

Видно, чому пікселі у табл. корельовані. Усі вісім чисел верхнього рядка таблиці близькі один до одного (відстань між ними дорівнюють 2 або 3). Всі інші рядки виходять циклічним зсувом праворуч попереднього рядка.

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

Видно, чому пікселі у табл. корельовані. Усі вісім чисел верхнього рядка таблиці близькі один до одного (відстань між ними дорівнюють 2 або 3). Всі інші рядки виходять циклічним зсувом праворуч попереднього рядка.

12	10	8	10	12	10	8	11	81	0	0	0	0	0	0	0
11	12	10	8	10	12	10	8	0	1.57	0.61	1.90	0.38	-1.81	0.20	-0.32
8	11	12	10	8	10	12	10	0	-0.61	0.71	0.35	0	0.07	0	0.02
10	8	11	12	10	8	10	12	0	1.90	-0.35	4.76	0.77	-3.39	0.25	-0.54
12	10	8	11	12	10	8	10	0	-0.38	0	-0.77	8.00	0.51	0	0.07
10	12	10	8	11	12	10	8	0	-1.81	-0.07	-3.39	-0.51	1.57	0.56	0.25
8	10	12	10	8	11	12	10	0	-0.20	0	-0.25	0	-0.56	-0.71	0.29
10	8	10	12	10	8	11	12	0	-0.32	-0.02	-0.54	-0.07	0.25	-0.29	-0.90

Двовимірне DCT блоку корельованих величин

Видно, чому пікселі у табл. корельовані. Усі вісім чисел верхнього рядка таблиці близькі один до одного (відстань між ними дорівнюють 2 або 3). Всі інші рядки виходять циклічним зсувом праворуч попереднього рядка.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг $w_j^{(i)}$, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)}{}_j, w^{(2)}{}_j, \ldots, w^{(k)}{}_j$. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стиснений файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення зображень. Двовимірне (матричне) DCT

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64 ваг $w_j^{(i)}$, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)}{}_j, w^{(2)}{}_j, \ldots, w^{(k)}{}_j$. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)_j}, w^{(2)_j}, \ldots, w^{(k)_j},$. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)_j}, w^{(2)_j}, \ldots, w^{(k)_j}$,. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \dots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \dots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)_j}, w^{(2)_j}, \dots, w^{(k)_j},$. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)_j}, w^{(2)_j}, \ldots, w^{(k)_j},$. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)_j}, w^{(2)_j}, \ldots, w^{(k)_j}$,. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)}{}_j, w^{(2)}{}_j, \ldots, w^{(k)}{}_j$,. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення зображень. Двовимірне (матричне) DCT

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)}{}_j, w^{(2)}{}_j, \ldots, w^{(k)}{}_j$,. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)j}, w^{(2)j}, \ldots, w^{(k)j},$. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)}{}_j, w^{(2)}{}_j, \ldots, w^{(k)}{}_j$,. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення зображень. Двовимірне (матричне) DCT

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \dots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \dots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)}{}_j, w^{(2)}{}_j, \dots, w^{(k)}{}_j$,. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i.
Зазначимо, що метод JPEG працює дещо інакше.

Стиснення зображень. Двовимірне (матричне) DCT

Стиснення будь-якого зображення за допомогою DCT тепер можна зробити нижче описаним чином.

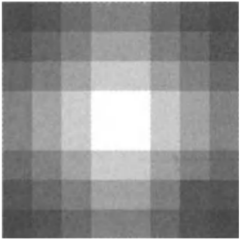
1. Розділити його на блоки пікселів розміру $n \times n$ (зазвичай 8×8).
2. Застосувати DCT до кожного блоку B_i, тобто, зобразити кожен блок у вигляді лінійної комбінації 64 базисних блоків. Результатом стануть блоки (ми називатимемо їх векторами) $W^{(i)}$ з 64-х ваг w_j^i, де $j = 0, 1, \ldots, 63$.
3. Усі k векторів $W^{(i)}$ $(i = 1, 2, \ldots, k)$ розділити на 64 вектори коефіцієнтів C^j з k компонентами $w^{(1)_j}, w^{(2)_j}, \ldots, w^{(k)_j},$. Вектор перших компонентів C^0 складається з k коефіцієнтів DC.
4. Зробити квантування кожного вектора коефіцієнтів C^j незалежно від інших. Отриманий квантований вектор Q^j записати (після додаткової компресії за методом RLE, Гаффмана чи іншого методу) у стислий файл.

Декодер читає 64 квантованих вектора коефіцієнтів Q^j використовує їх для побудови до k вагових векторів W^j і застосовує IDCT до кожного вагового вектора для (наближеної) реконструкції 64 пікселів блоку B_i. Зазначимо, що метод JPEG працює дещо інакше.

Стиснення зображень. Приклад

Цей приклад демонструє різницю у продуктивності методу DCT при стисненні неперервно тонового зображення та дискретно-тонового зображення. Ми виходимо із сильно корелюваного зразка, наведеного в табл.

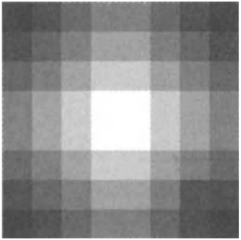
00	10	20	30	30	20	10	00
10	20	30	40	40	30	20	10
20	30	40	50	50	40	30	20
30	40	50	60	60	50	40	30
30	40	50	60	60	50	40	30
20	30	40	50	50	40	30	20
10	20	30	40	40	30	12	10
00	10	20	30	30	20	10	00



Зразок із високою кореляцією

Цей приклад демонструє різницю у продуктивності методу DCT при стисненні неперервно тонового зображення та дискретно-тонового зображення. Ми виходимо із сильно корелюваного зразка, наведеного в табл.

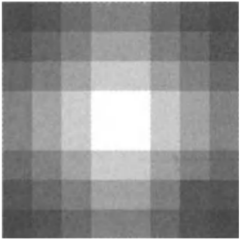
00	10	20	30	30	20	10	00
10	20	30	40	40	30	20	10
20	30	40	50	50	40	30	20
30	40	50	60	60	50	40	30
30	40	50	60	60	50	40	30
20	30	40	50	50	40	30	20
10	20	30	40	40	30	12	10
00	10	20	30	30	20	10	00



Зразок із високою кореляцією

Цей приклад демонструє різницю у продуктивності методу DCT при стисненні неперервно тонового зображення та дискретно-тонового зображення. Ми виходимо із сильно корелюваного зразка, наведеного в табл.

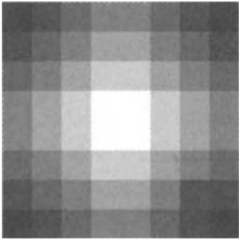
00	10	20	30	30	20	10	00
10	20	30	40	40	30	20	10
20	30	40	50	50	40	30	20
30	40	50	60	60	50	40	30
30	40	50	60	60	50	40	30
20	30	40	50	50	40	30	20
10	20	30	40	40	30	12	10
00	10	20	30	30	20	10	00



Зразок із високою кореляцією

Цей приклад демонструє різницю у продуктивності методу DCT при стисненні неперервно тонового зображення та дискретно-тонового зображення. Ми виходимо із сильно корелюваного зразка, наведеного в табл.

00	10	20	30	30	20	10	00
10	20	30	40	40	30	20	10
20	30	40	50	50	40	30	20
30	40	50	60	60	50	40	30
30	40	50	60	60	50	40	30
20	30	40	50	50	40	30	20
10	20	30	40	40	30	12	10
00	10	20	30	30	20	10	00



Зразок із високою кореляцією

Стиснення зображень. Приклад

Це буде ідеалізована модель неперервно тонового зображення, оскільки сусідні пікселі відрізняються на постійну величину. Усі 64 коефіцієнти DCT наведено у табл.

239	1.19	–89.76	–0.28	1.00	–1.39	–5.03	–0.79
1.18	–1.39	0.64	0.32	–1.18	1.63	–1.54	0.92
–89.76	0.64	–0.29	–0.15	0.54	–0.75	0.71	–0.43
–0.28	0.32	–0.15	–0.08	0.28	–0.38	0.36	–0.22
1.00	–1.18	0.54	0.28	–1.00	1.39	–1.31	0.79
–1.39	1.63	–0.75	–0.38	1.39	–1.92	1.81	–1.09
–5.03	–1.54	0.71	0.36	–1.31	1.81	–1.71	1.03
–0.79	0.92	–0.43	–0.22	0.79	–1.09	1.03	–0.62

DCT коефіцієнти зразка

Це буде ідеалізована модель неперервно тонового зображення, оскільки сусідні пікселі відрізняються на постійну величину. Усі 64 коефіцієнти DCT наведено у табл.

239	1.19	–89.76	–0.28	1.00	–1.39	–5.03	–0.79
1.18	–1.39	0.64	0.32	–1.18	1.63	–1.54	0.92
–89.76	0.64	–0.29	–0.15	0.54	–0.75	0.71	–0.43
–0.28	0.32	–0.15	–0.08	0.28	–0.38	0.36	–0.22
1.00	–1.18	0.54	0.28	–1.00	1.39	–1.31	0.79
–1.39	1.63	–0.75	–0.38	1.39	–1.92	1.81	–1.09
–5.03	–1.54	0.71	0.36	–1.31	1.81	–1.71	1.03
–0.79	0.92	–0.43	–0.22	0.79	–1.09	1.03	–0.62

DCT коефіцієнти зразка

Це буде ідеалізована модель неперервно тонового зображення, оскільки сусідні пікселі відрізняються на постійну величину. Усі 64 коефіцієнти DCT наведено у табл.

239	1.19	–89.76	–0.28	1.00	–1.39	–5.03	–0.79
1.18	–1.39	0.64	0.32	–1.18	1.63	–1.54	0.92
–89.76	0.64	–0.29	–0.15	0.54	–0.75	0.71	–0.43
–0.28	0.32	–0.15	–0.08	0.28	–0.38	0.36	–0.22
1.00	–1.18	0.54	0.28	–1.00	1.39	–1.31	0.79
–1.39	1.63	–0.75	–0.38	1.39	–1.92	1.81	–1.09
–5.03	–1.54	0.71	0.36	–1.31	1.81	–1.71	1.03
–0.79	0.92	–0.43	–0.22	0.79	–1.09	1.03	–0.62

DCT коефіцієнти зразка

Це буде ідеалізована модель неперервно тонового зображення, оскільки сусідні пікселі відрізняються на постійну величину. Усі 64 коефіцієнти DCT наведено у табл.

239	1.19	−89.76	−0.28	1.00	−1.39	−5.03	−0.79
1.18	−1.39	0.64	0.32	−1.18	1.63	−1.54	0.92
−89.76	0.64	−0.29	−0.15	0.54	−0.75	0.71	−0.43
−0.28	0.32	−0.15	−0.08	0.28	−0.38	0.36	−0.22
1.00	−1.18	0.54	0.28	−1.00	1.39	−1.31	0.79
−1.39	1.63	−0.75	−0.38	1.39	−1.92	1.81	−1.09
−5.03	−1.54	0.71	0.36	−1.31	1.81	−1.71	1.03
−0.79	0.92	−0.43	−0.22	0.79	−1.09	1.03	−0.62

DCT коефіцієнти зразка

Це буде ідеалізована модель неперервно тонового зображення, оскільки сусідні пікселі відрізняються на постійну величину. Усі 64 коефіцієнти DCT наведено у табл.

239	1.19	−89.76	−0.28	1.00	−1.39	−5.03	−0.79
1.18	−1.39	0.64	0.32	−1.18	1.63	−1.54	0.92
−89.76	0.64	−0.29	−0.15	0.54	−0.75	0.71	−0.43
−0.28	0.32	−0.15	−0.08	0.28	−0.38	0.36	−0.22
1.00	−1.18	0.54	0.28	−1.00	1.39	−1.31	0.79
−1.39	1.63	−0.75	−0.38	1.39	−1.92	1.81	−1.09
−5.03	−1.54	0.71	0.36	−1.31	1.81	−1.71	1.03
−0.79	0.92	−0.43	−0.22	0.79	−1.09	1.03	−0.62

DCT коефіцієнти зразка

Видно, що є лише кілька домінуючих коефіцієнтів. У табл. наведено результат деякого грубого квантування нашого зразка.

239	1	−90	0	0	0	0	0
0	0	0	0	0	0	0	0
−90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Грубе квантування з 4 ненульовими коефіцієнтами

У цій таблиці є всього чотири ненульові коефіцієнти.

Видно, що є лише кілька домінуючих коефіцієнтів. У табл. наведено результат деякого грубого квантування нашого зразка.

239	1	–90	0	0	0	0	0
0	0	0	0	0	0	0	0
–90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Грубе квантування з 4 ненульовими коефіцієнтами

У цій таблиці є всього чотири ненульові коефіцієнти.

Видно, що є лише кілька домінуючих коефіцієнтів. У табл. наведено результат деякого грубого квантування нашого зразка.

239	1	–90	0	0	0	0	0
0	0	0	0	0	0	0	0
–90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Грубе квантування з 4 ненульовими коефіцієнтами

У цій таблиці є всього чотири ненульові коефіцієнти.

Стиснення зображень. Приклад

Видно, що є лише кілька домінуючих коефіцієнтів. У табл. наведено результат деякого грубого квантування нашого зразка.

239	1	−90	0	0	0	0	0
0	0	0	0	0	0	0	0
−90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Грубе квантування з 4 ненульовими коефіцієнтами

У цій таблиці є всього чотири ненульові коефіцієнти.

Видно, що є лише кілька домінуючих коефіцієнтів. У табл. наведено результат деякого грубого квантування нашого зразка.

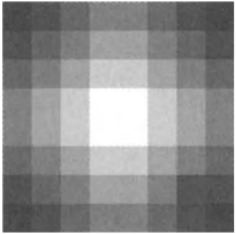
239	1	–90	0	0	0	0	0
0	0	0	0	0	0	0	0
–90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Грубе квантування з 4 ненульовими коефіцієнтами

У цій таблиці є всього чотири ненульові коефіцієнти.

Результат застосування IDCT до цих коефіцієнтів наведено в табл.

0.65	9.23	21.36	29.91	29.84	21.17	8.94	0.30
9.26	17.85	29.97	38.52	38.45	29.78	17.55	8.91
21.44	30.02	42.15	50.70	50.63	41.95	29.73	21.09
30.05	38.63	50.76	59.31	59.24	50.56	38.34	29.70
30.05	38.63	50.76	59.31	59.24	50.56	38.34	29.70
21.44	30.02	42.15	50.70	50.63	41.95	29.73	21.09
9.26	17.85	29.97	38.52	38.45	29.78	17.55	8.91
0.65	9.23	21.36	29.91	29.84	21.17	8.94	0.30

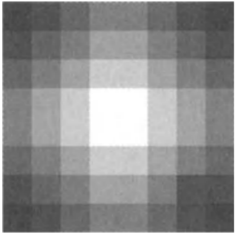


Результат IDTC

Очевидно, що ці чотири коефіцієнти дозволяють відновити зразок із високим ступенем точності.

Результат застосування IDCT до цих коефіцієнтів наведено в табл.

0.65	9.23	21.36	29.91	29.84	21.17	8.94	0.30
9.26	17.85	29.97	38.52	38.45	29.78	17.55	8.91
21.44	30.02	42.15	50.70	50.63	41.95	29.73	21.09
30.05	38.63	50.76	59.31	59.24	50.56	38.34	29.70
30.05	38.63	50.76	59.31	59.24	50.56	38.34	29.70
21.44	30.02	42.15	50.70	50.63	41.95	29.73	21.09
9.26	17.85	29.97	38.52	38.45	29.78	17.55	8.91
0.65	9.23	21.36	29.91	29.84	21.17	8.94	0.30

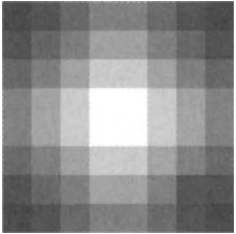


Результат IDTC

Очевидно, що ці чотири коефіцієнти дозволяють відновити зразок із високим ступенем точності.

Результат застосування IDCT до цих коефіцієнтів наведено в табл.

0.65	9.23	21.36	29.91	29.84	21.17	8.94	0.30
9.26	17.85	29.97	38.52	38.45	29.78	17.55	8.91
21.44	30.02	42.15	50.70	50.63	41.95	29.73	21.09
30.05	38.63	50.76	59.31	59.24	50.56	38.34	29.70
30.05	38.63	50.76	59.31	59.24	50.56	38.34	29.70
21.44	30.02	42.15	50.70	50.63	41.95	29.73	21.09
9.26	17.85	29.97	38.52	38.45	29.78	17.55	8.91
0.65	9.23	21.36	29.91	29.84	21.17	8.94	0.30

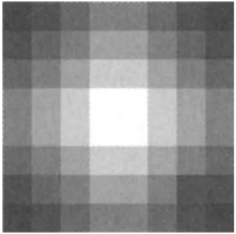


Результат IDTC

Очевидно, що ці чотири коефіцієнти дозволяють відновити зразок із високим ступенем точності.

Результат застосування IDCT до цих коефіцієнтів наведено в табл.

0.65	9.23	21.36	29.91	29.84	21.17	8.94	0.30
9.26	17.85	29.97	38.52	38.45	29.78	17.55	8.91
21.44	30.02	42.15	50.70	50.63	41.95	29.73	21.09
30.05	38.63	50.76	59.31	59.24	50.56	38.34	29.70
30.05	38.63	50.76	59.31	59.24	50.56	38.34	29.70
21.44	30.02	42.15	50.70	50.63	41.95	29.73	21.09
9.26	17.85	29.97	38.52	38.45	29.78	17.55	8.91
0.65	9.23	21.36	29.91	29.84	21.17	8.94	0.30



Результат IDTC

Очевидно, що ці чотири коефіцієнти дозволяють відновити зразок із високим ступенем точності.

Стиснення зображень. Приклад

У наступних таблицях повторений той же процес стосовно Y-подібного блоку даних, типового для дискретно-тонового зображення.

00	10	00	00	00	00	00	10
00	00	10	00	00	00	10	00
00	00	00	10	00	10	00	00
00	00	00	00	10	00	00	00
00	00	00	00	10	00	00	00
00	00	00	00	10	00	00	00
00	00	00	00	10	00	00	00
00	00	00	00	10	00	00	00

Y-подібний блок

13.75	–3.11	–8.17	2.46	3.75	–6.86	–3.38	6.59
4.19	–0.29	6.86	–6.85	–7.13	4.48	1.69	–7.28
1.63	0.19	6.40	–4.81	–2.99	–1.11	–0.88	–0.94
–0.61	0.54	5.12	–2.31	1.30	–6.04	–2.78	3.05
–1.25	0.52	2.99	–0.20	3.75	–7.39	–2.59	1.16
–0.41	0.18	0.65	1.03	3.87	–5.19	–0.71	–4.76
0.68	–0.15	–0.88	1.28	2.59	–1.92	1.10	–9.05
0.83	–0.21	–0.99	0.82	1.13	–0.08	1.31	–7.21

DCT коефіцієнти блоку

Стиснення зображень. Приклад

У наступних таблицях повторений той же процес стосовно Y-подібного блоку даних, типового для дискретно-тонового зображення.

00	10	00	00	00	00	00	10
00	00	10	00	00	00	10	00
00	00	00	10	00	10	00	00
00	00	00	00	10	00	00	00
00	00	00	00	10	00	00	00
00	00	00	00	10	00	00	00
00	00	00	00	10	00	00	00
00	00	00	00	10	00	00	00

Y-подібний блок

13.75	–3.11	–8.17	2.46	3.75	–6.86	–3.38	6.59
4.19	–0.29	6.86	–6.85	–7.13	4.48	1.69	–7.28
1.63	0.19	6.40	–4.81	–2.99	–1.11	–0.88	–0.94
–0.61	0.54	5.12	–2.31	1.30	–6.04	–2.78	3.05
–1.25	0.52	2.99	–0.20	3.75	–7.39	–2.59	1.16
–0.41	0.18	0.65	1.03	3.87	–5.19	–0.71	–4.76
0.68	–0.15	–0.88	1.28	2.59	–1.92	1.10	–9.05
0.83	–0.21	–0.99	0.82	1.13	–0.08	1.31	–7.21

DCT коефіцієнти блоку

При цьому використовувалося досить легке квантування (див. табл.).

13.75	–3	–8	2	3	–6	–3	6
4	–0	6	–6	–7	4	1	–7
1	0	6	–4	–2	–1	–0	–0
–0	0	5	–2	1	–6	–2	3
–1	0	2	–0	3	–7	–2	1
–0	0	0	1	3	–5	–0	–4
0	–0	–0	1	2	–1	1	–9
0	–0	–0	0	1	–0	1	–7

Слабке квантування округленням до найближчого цілого

При цьому використовувалося досить легке квантування (див. табл.).

13.75	−3	−8	2	3	−6	−3	6
4	−0	6	−6	−7	4	1	−7
1	0	6	−4	−2	−1	−0	−0
−0	0	5	−2	1	−6	−2	3
−1	0	2	−0	3	−7	−2	1
−0	0	0	1	3	−5	−0	−4
0	−0	−0	1	2	−1	1	−9
0	−0	−0	0	1	−0	1	−7

Слабке квантування округленням до найближчого цілого

При цьому використовувалося досить легке квантування (див. табл.).

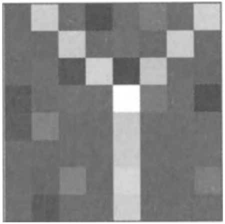
13.75	−3	−8	2	3	−6	−3	6
4	−0	6	−6	−7	4	1	−7
1	0	6	−4	−2	−1	−0	−0
−0	0	5	−2	1	−6	−2	3
−1	0	2	−0	3	−7	−2	1
−0	0	0	1	3	−5	−0	−4
0	−0	−0	1	2	−1	1	−9
0	−0	−0	0	1	−0	1	−7

Слабке квантування округленням до найближчого цілого

Стиснення зображень. Приклад

Видно, що результат реконструкції цього образу (див. табл.) не такий гарний, як у попередньому випадку.

-0.13	8.96	0.55	-0.27	0.27	0.86	0.15	9.22
0.32	0.22	9.10	0.40	0.84	-0.11	9.36	-0.14
0.00	0.62	-0.20	9.71	-1.30	8.57	0.28	-0.33
-0.58	0.44	0.78	0.71	10.11	1.14	0.44	-0.49
-0.39	0.67	0.07	0.38	8.82	0.09	0.28	0.41
0.34	0.11	0.26	0.18	8.93	0.41	0.47	0.37
0.09	-0.32	0.78	-0.20	9.78	0.05	-0.09	0.49
0.16	-0.83	0.09	0.12	9.15	-0.11	-0.08	0.01

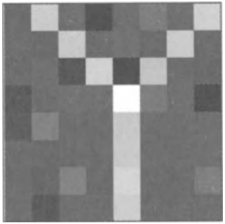


Результат IDTC. Погана якість

Величини, які дорівнювали 10, тепер розташовані в інтервалі від 8.96 до 10.11, а ті, що були нулями, виросли до 0.86.

Видно, що результат реконструкції цього образу (див. табл.) не такий гарний, як у попередньому випадку.

-0.13	8.96	0.55	-0.27	0.27	0.86	0.15	9.22
0.32	0.22	9.10	0.40	0.84	-0.11	9.36	-0.14
0.00	0.62	-0.20	9.71	-1.30	8.57	0.28	-0.33
-0.58	0.44	0.78	0.71	10.11	1.14	0.44	-0.49
-0.39	0.67	0.07	0.38	8.82	0.09	0.28	0.41
0.34	0.11	0.26	0.18	8.93	0.41	0.47	0.37
0.09	-0.32	0.78	-0.20	9.78	0.05	-0.09	0.49
0.16	-0.83	0.09	0.12	9.15	-0.11	-0.08	0.01

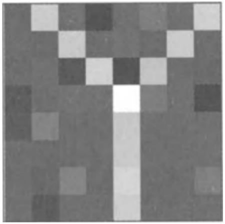


Результат IDTC. Погана якість

Величини, які дорівнювали 10, тепер розташовані в інтервалі від 8.96 до 10.11, а ті, що були нулями, виросли до 0.86.

Видно, що результат реконструкції цього образу (див. табл.) не такий гарний, як у попередньому випадку.

-0.13	8.96	0.55	-0.27	0.27	0.86	0.15	9.22
0.32	0.22	9.10	0.40	0.84	-0.11	9.36	-0.14
0.00	0.62	-0.20	9.71	-1.30	8.57	0.28	-0.33
-0.58	0.44	0.78	0.71	10.11	1.14	0.44	-0.49
-0.39	0.67	0.07	0.38	8.82	0.09	0.28	0.41
0.34	0.11	0.26	0.18	8.93	0.41	0.47	0.37
0.09	-0.32	0.78	-0.20	9.78	0.05	-0.09	0.49
0.16	-0.83	0.09	0.12	9.15	-0.11	-0.08	0.01



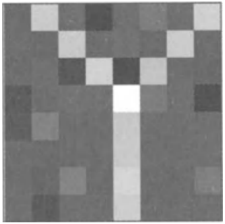
Результат IDTC. Погана якість

Величини, які дорівнювали 10, тепер розташовані в інтервалі від 8.96 до 10.11, а ті, що були нулями, виросли до 0.86.

Стиснення зображень. Приклад

Видно, що результат реконструкції цього образу (див. табл.) не такий гарний, як у попередньому випадку.

-0.13	8.96	0.55	-0.27	0.27	0.86	0.15	9.22
0.32	0.22	9.10	0.40	0.84	-0.11	9.36	-0.14
0.00	0.62	-0.20	9.71	-1.30	8.57	0.28	-0.33
-0.58	0.44	0.78	0.71	10.11	1.14	0.44	-0.49
-0.39	0.67	0.07	0.38	8.82	0.09	0.28	0.41
0.34	0.11	0.26	0.18	8.93	0.41	0.47	0.37
0.09	-0.32	0.78	-0.20	9.78	0.05	-0.09	0.49
0.16	-0.83	0.09	0.12	9.15	-0.11	-0.08	0.01

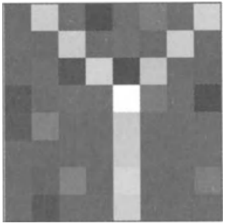


Результат IDTC. Погана якість

Величини, які дорівнювали 10, тепер розташовані в інтервалі від 8.96 до 10.11, а ті, що були нулями, виросли до 0.86.

Видно, що результат реконструкції цього образу (див. табл.) не такий гарний, як у попередньому випадку.

-0.13	8.96	0.55	-0.27	0.27	0.86	0.15	9.22
0.32	0.22	9.10	0.40	0.84	-0.11	9.36	-0.14
0.00	0.62	-0.20	9.71	-1.30	8.57	0.28	-0.33
-0.58	0.44	0.78	0.71	10.11	1.14	0.44	-0.49
-0.39	0.67	0.07	0.38	8.82	0.09	0.28	0.41
0.34	0.11	0.26	0.18	8.93	0.41	0.47	0.37
0.09	-0.32	0.78	-0.20	9.78	0.05	-0.09	0.49
0.16	-0.83	0.09	0.12	9.15	-0.11	-0.08	0.01



Результат IDTC. Погана якість

Величини, які дорівнювали 10, тепер розташовані в інтервалі від 8.96 до 10.11, а ті, що були нулями, виросли до 0.86.

Дякую за увагу!