Assignment 1: Transform Coding

CS 4600 Computer Graphics Fall 2018

Rui Ying u1234364

1. Audio Coding
   a. DCT

   According to Fig. 1, we can write code for DCT transform. Use for loop to sum.

$$F(u) = c(u) \sum_{i=0}^{N-1} cos\left(\frac{(2i+1)u\pi}{2N}\right) f(i),$$

Fig. 1

Code snippet

```
for (int u = 0; u < size; u++) {
    float sum = 0;
    for (int i = 0; i < size; i++) {
        sum += std::cos((2 * i + 1) * u * M_PI / 2 / size) * A[i];
    }
    C[u] = (u == 0 ? sqrtf(2) / 4 : 0.5) * sum;
}
```

Bug solved

Use 0.5 instead of ½ because integer arithmetic in C++ produces an integer which in this case would be 0.

   b. Compress

Because we do not really throw away data, so the last m data would only be wiped to zero.

Code snippet

```
for (int i = size - m; i < size; ++i) {
    C[i] = 0;
}
```

   c. Inverse DCT

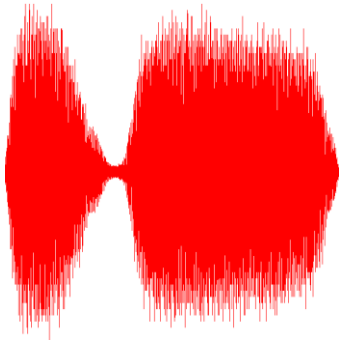
Same as DCT, we just write code according to the formula (Fig. 2).

$$f(i) = \sum_{u=0}^{N-1} c(u)\, cos\left(\frac{(2i+1)u\pi}{2N}\right) F(u)$$

Fig. 2

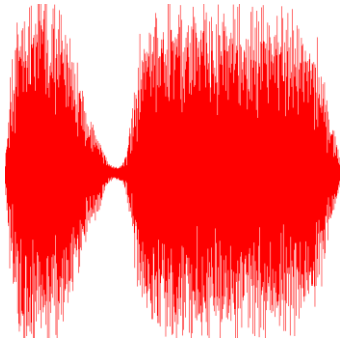
   d. Result

Original sound (m=0)



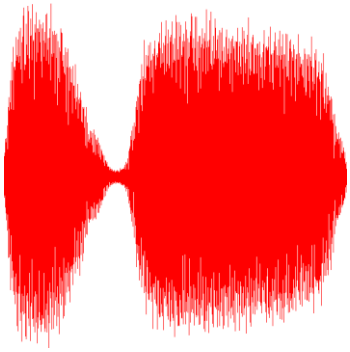
m=8 (max compression)

m=5 (required compression)



Wave shrinks. Sound is slightly distorted.

m=1 (min compression)



2. Image Coding

a. 2D DCT

Following the formula in Fig. 3, use for loop to sum the value.

$$F(u,v) = c(u)\,c(v) \sum_{x,y=0}^{N-1} \cos\left(\frac{(2x+1)u\pi}{2N}\right) \cos\left(\frac{(2y+1)v\pi}{2N}\right) f(x,y)$$

Fig. 3

Because the input array is not 2D, when using i and j to index elements, we need to convert 2D index to 1D index, as is shown in code.

Code snippet

```
for (int v = 0; v < blockSize; v++) {
    for (int u = 0; u < blockSize; u++) {
        int indexUV = v * blockSize + u;
        float sum = 0;
        for (int y = 0; y < blockSize; y++) {
            for (int x = 0; x < blockSize; x++) {
                int indexXY = y * blockSize + x;
                sum += std::cos((2 * x + 1) * u * M_PI / 2 / blockSize) * std::cos((2 * y + 1) * v * M_PI / 2 / blockSize)*A[indexXY];
            }
        }
        C[indexUV] = (u == 0 ? sqrtf(2) / 4 : 0.5f) * (v == 0 ? sqrtf(2) / 4 : 0.5f) * sum;
    }
}
```

b. Compress

Similar as 1D DCT but we will set some matrix elements to zero.

Code snippet

```
for (int i = 0; i < blockSize; i++) {
    for (int j = 0; j < blockSize; j++) {
        if (i + j ≤ m)
        {
            continue;
        }
        int index = i * blockSize + j;
        C[index] = 0;
    }
}
```

c. Inverse 2D DCT

Use the formula in Fig. 4.

$$f(x,y) = \sum_{u,v=0}^{N-1} c(u)\,c(v) \cos\left(\frac{(2x+1)u\pi}{2N}\right) \cos\left(\frac{(2y+1)v\pi}{2N}\right) F(u,v)$$

Fig. 4

Code snippet

```
for (int y = 0; y < blockSize; y++) {
    for (int x = 0; x < blockSize; x++) {
        int indexXY = y * blockSize + x;
        float sum = 0;
        for (int v = 0; v < blockSize; v++) {
            for (int u = 0; u < blockSize; u++) {
                int indexUV = v * blockSize + u;
                sum += (u == 0 ? sqrtf(2) / 4 : 0.5f) * (v == 0 ? sqrtf(2) / 4 : 0.5f)
                    * std::cos((2 * x + 1) * u * M_PI / 2 / blockSize)
                    * std::cos((2 * y + 1) * v * M_PI / 2 / blockSize)
                    *C[indexUV];
            }
        }
        B[indexXY] = sum;
    }
}
```

d. Result

Original photo

m = 1

m = 3

m=15

3. Reference
   CS 4600 Assignment 1.pdf