Quantization

Non-uniform scalar quantizer

과목명	디지털신호처리
담당교수	심동규 교수님
학과	컴퓨터정보공학부
학년	3학년
학번	2019202009
이름	서여지
제출일	2021.11.2 (화)

KWANGWOON UNIVERSITY

1. 과제 개요

저번 과제에 이어서 C++ 언어를 이용하여 양자화기를 구현한다. 이번에 구현하는 양자화기와 역양자화기는 non-uniform scalar 방식의 양자화와 역양자화를 수행한다. 테스트 이미지는 512 * 512 해상도를 갖고, R, G, B 채널로 이루어져 있다. 각각의 채널에 대해 한 픽셀에는 8bit의 색상 정보가 들어있다. 8bit보다 적은 수의 bit를 이용하여 이미지를 저장하는 양자화기와, 저장한 이미지를 다시 복원하는 역양자화기를 설계한다.

2. 과제 수행 방법 (이론과 구현)

- Non-uniform scalar Quantization

  Uniform scalar Quantization은 다수의 샘플을 모두 독립적으로, 균일하지 않은 간격으로 나누어 양자화 한 것을 의미한다. training data의 값에 따라 양자화하기 위한 구간의 크기와 구간에 매핑되는 값이 변화한다. 이번 과제에서는 Lloyd-Max quantizer를 사용하여 구간을 계산하였다.

- Lloyd-Max quantizer

  Lloyd-Max quantizer는 양자화 오차를 최소화하는 decision level과 representation level을 결정한다.

$$E\{(s-r_i)^2\} = \sum_{i=1}^{L} \int_{d_{i-1}}^{d_i} (s-r_i)^2 p(s) ds$$

  위 양자화 오차 식에서 reconstruction level (r)과 decision level(d)를 얻는다.

$$r_i = \frac{\int_{d_{i-1}}^{d_i} sp(s)ds}{\int_{d_{i-1}}^{d_i} p(s)ds}, (1 \le i \le L) \qquad d_i = \frac{r_i + r_{i+1}}{2}, (1 \le i \le L-1)$$

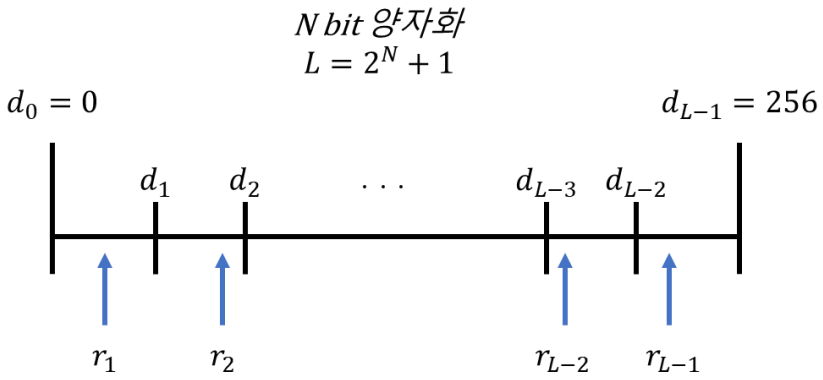
  이때 s는 sample의 값, p(s)는 해당 sample의 확률, L은 전체 구간의 수이다.

  Lloyd-Max quantizer는 r과 d를 번갈아가며 여러 번 반복하여 계산하는 과정을 통해 두 값을 최적화한다.

- 구현방법

  - Lloyd-Max quantizer

    d와 r값을 $2^N + 1$크기의 int형 배열로 선언하여 사용하였다. d와 r배열이 나타내는 값은 다음과 같다.



r_0 사용하지 않음 (총 2^N 개의 r값)
d: min(0)과 max(256)을 모두 포함 ($2^N + 1$개의 값)

d[0]은 sample의 최소 값인 0을 의미하고, d[L-1]은 256을 의미한다. 이때 d[L-1]을 255가 아닌 256으로 지정하는 이유는 반복문 계산의 편의성을 위함이다. r[0]의 값은 사용하지 않으며, d와 r배열 사이에 $d[i-1] \le r[i] \le d[i]$의 관계가 성립한다. 양자화 과정에서 sample의 값이 d[i-1]보다 크고, d[i]보다 작다면 i로 매핑하고, 이를 역양자화 했을 때 복원되는 값은 r[i]이다.

또한 초기 d배열의 값은 전체 구간을 균등한 크기로 나눈 값을 사용하였다.

  - 프로그램 구조

    - 양자화기

      양자화기는 input 디렉토리의 Lenna_512x512_original.raw를 읽어서 양자화한 뒤, output 디렉토리를 생성하여 Lenna_512x512_code와 Lenna_512x512_codebook을 출력한다.

      프로그램은 원본 파일을 읽는 부분, 값의 등장 빈도를 계산하고 초기 di 값을 구하는 부분, ri와 di를 번갈아 반복하며 계산하여 구간을 구하는 부분, 구한 구간에 따라 이미지 값을 양자화 하는 부분, 결과를 저장하는 부분으로 나누어진다. 동적

할당한 unsigned char 배열을 이용하여 원본 파일을 모두 읽은 뒤, 0에서 255까지 값의 빈도를 계산하여 p배열에 저장한다. 초기 d 배열의 값은 전체 구간을 균등한 크기로 나눈 값을 이용하므로 구간의 크기를 구한 뒤 0에서부터 누적하여 각 구간의 끝 값을 구하였다. 이후 반복문 내부에서 r과 d의 값을 반복하여 수정한다. r배열의 값은 현재 구간 내에 존재하는 샘플의 평균값으로 수정되며, d배열의 값은 r[i]와 r[i+1]의 평균 값으로 수정된다. 이 과정은 d배열의 값이 더 이상 수정되지 않을 때까지 반복한다. 이후 이미지 값에 해당하는 구간의 index값(u)을 code파일에 출력한다. 결과를 code파일에 저장하는 과정은 저번 과제 구현에서와 동일하게 unsigned int형의 버퍼를 이용하여 수행된다. 하위 Nbit를 채워넣고, 상위 1byte를 쓰는 과정을 반복한다. codebook에는 r배열의 값을 순서대로 쓴다.

◆ 역양자화기

역양자화기는 input 디렉토리의 Lenna_512x512_code과 Lenna_512x512_codebook을 읽어서 역양자화 한 뒤, output 디렉토리를 생성하여 Lenna_512x512_reconstruct.raw로 출력한다.

프로그램은 code와 codebook 파일을 읽는 부분, 역양자화 하는 부분, 결과를 저장하는 부분으로 나누어진다. code파일은 저번 과제와 동일하게 파일의 값을 1byte씩 읽어 버퍼에 저장한 뒤, 상위 N 비트를 image 배열에 저장하여 양자화된 코드를 읽는다. codebook파일의 내용은 순서대로 int형 배열 r에 저장한다. image[i][j]의 값(u)과 대응하는 r값(r[u])을 이용하여 역양자화한 뒤 결과를 파일에 쓰고 종료한다.

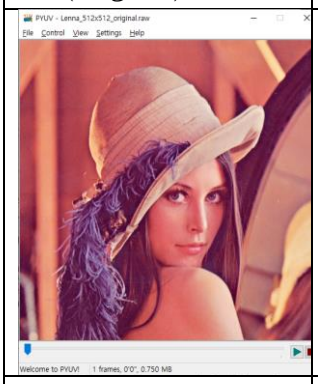
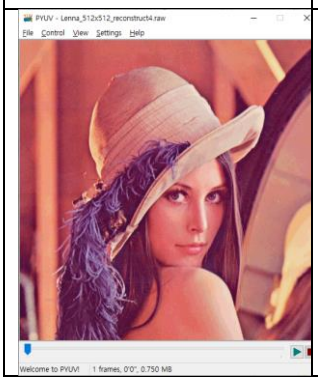
3. 결과

d배열과 r배열의 값을 출력하면 양자화 구간이 나눠진 것을 확인할 수 있다. 다음은 5bit를 이용한 경우의 결과의 일부이다.

	d[i]	r[i[]
2	0	0
3	9	7
4	13	11
5	17	15

d[0]~d[1]구간은 0에서 8까지 9개의 값을 갖으며 해당 구간의 값은 7로 역양자화된다.

반면 d[1]~d[2] 구간은 9에서 12까지 4개의 값을 갖으며 이 구간의 값은 11로 역양자화된다.

8bit(original)	7bit	6bit	5bit
			
4bit	3bit	2bit	1bit
			

bit	codesize(bit)	bit per pixel	MSE	Loss
8	6291456	24	0	24
7	5505024	21	0.483655	22.9346
6	4718592	18	1.42706	23.7083
5	3932160	15	5.25129	36.0052
4	3145728	12	19.0716	88.2865
3	2359296	9	65.3006	270.201
2	1572864	6	227.406	1115.62
1	786432	3	926.297	3708.19

제시된 이미지에 대해 1~7bit를 이용해 양자화 한 뒤, 다시 복원한 결과는 위와 같이 나타났다. Loss 평가가 가장 뛰어난 것은 7bit를 이용하여 양자화 한 경우였으며, 6bit를 활용한 경우와 함께 원본 영상보다도 효과적인 것으로 나타났다. 사용하는 bit수가 줄어들며 MSE의 값이 증가하며, 그로 인해 Loss값이 크게 증가하는 것은 저번 과제에서 구현한 scalar uniform quantizer와 일치하는 결과를 보였다.

4. 고찰

저번 과제에서 구현한 scalar uniform quantizer와 결과를 비교하면 다음과 같다.

	scalar uniform quantizer		scalar non-uniform quantizer	
bit	MSE	Loss	MSE	Loss
8	0	24	0	24
7	0.492086	22.9683	0.483655	22.9346
6	1.50674	24.027	1.42706	23.7083
5	5.54247	37.1699	5.25129	36.0052
4	22.0863	100.345	19.0716	88.2865
3	88.8075	364.23	65.3006	270.201
2	330.474	1327.9	227.406	1115.62
1	1219.01	4879.06	926.297	3708.19

위 결과에서 같은 bit를 사용한 경우 non-uniform 양자화기가 더 뛰어난 성능을 보인 것을 확인할 수 있다. 두 양자화기의 성능차이는 사용하는 bit의 수가 적을수록 크게 나타났다. non-uniform 양자화 결과가 uniform양자화 결과보다 뛰어난 것을 통해 양자화에 사용한 이미지의 전체 pixel값의 빈도가 균일하지 않음을 알 수 있다.