Video Compression HW 4－Entropy Coding

多工所 313553024 蘇柏叡

Github Link: https://github.com/rmd926/NYCU_VC/tree/main/Lab4

1. Introduction

本次作業實作一個JPEG影像壓縮流程，目的在於量化對壓縮率與重建畫質的影響。首先，以灰階Lena影像為例，將影像切分為8×8區塊，對每個區塊計算二維離散餘弦轉換(2D DCT)，將空間域像素轉換為頻率係數。接著分別套用題目提供的兩組8×8量化表，對各頻率係數作element-wise量化，模擬JPEG中對低頻與高頻採用不同量化程度的設計。在量化後，利用zigzag掃描將8x8區塊攤平成一維序列，讓低頻係數排在前段、高頻集中在後段，再進行Run-Length Encoding以壓縮連續零值。解碼時，先做RLE還原zigzag序列，再反zigzag、反量化與IDCT，最後拼回完整影像。透過比較兩組量化表產生的編碼檔案大小與峰值訊雜比(PSNR)，可以觀察到較粗量化雖能提升壓縮率，但會犧牲重建畫質，具體呈現壓縮效率與失真之間的取捨關係。

2. Methods and Implementations

(1) 8×8 block-based DCT coefficients of “lena.png.”

在讀入灰階影像後，利用雙層迴圈以8×8為步長做raster scan切成各個區塊，並且對每個區塊呼叫自寫的dct_2d()，依DCT-II公式逐一計算8×8的頻率係數矩陣。

(2) Quantize the coefficients with the two quantization tables.

先定義好題目提供的兩組8×8量化表Q1與Q2，隨後在quantize()中對每一個DCT區塊做element-wise相除並四捨五入得到量化係數，在解碼時再用dequantize()以相同量化表逐元素相乘進行反量化。

(3) Use a raster scan to visit all 8×8 blocks in these images.

在encode_decode_image()中使用雙層迴圈由上到下、由左到右依序處理所有8×8區塊，對每個區塊執行DCT、量化與後續編碼。

(4) Do the run length encoding by using a zigzag scan to visit all pixels in one block.

對量化後的8×8區塊，先利用預先定義的ZIGZAG_ORDER在zigzag()中依JPEG常用zigzag順序展平成64維序列，再於rle_encode()針對該序列進行零值導向的RLE，將連續的0壓縮為(0, count)，非零係數則記為(value, 1)。

(5) Do the run length decoding and IDCT to recover the image.

解碼時，對每個區塊先以rle_decode()還原zigzag序列，再用inv_zigzag()放回8×8位置，接著經dequantize()反量化後，呼叫idct_2d()得到空間域8×8區塊，最後依raster位置寫回輸出影像。

(6) Compare the encoded image sizes with the two quantization tables.

將所有區塊的RLE結果存入bitstream，分別以pickle.dump寫成qtable1.pkl與qtable2.pkl，利用os.path.getsize()取得檔案大小作為壓縮率指標，同時計算兩種量化表下重建影像的PSNR，用以比較壓縮效率與畫質之間的差異。

3. Experiment Results

3.3 Comparison of Full Search and Three Step Search

(1) PSNR

	PSNR	Run Time
Reconstruction (Q1)	**37.23 dB**	**6.390 s**
Reconstruction (Q2)	**34.33 dB**	**6.444 s**

(2) Execution result screenshots

```
PS C:\Users\user\Desktop\Video Compression\Lab4> &
==== DCT-based Image Compression Demo ====
> Quantization table 1...
> Quantization table 2...
[Q1] size=921347 bytes, time=6.390s, PSNR=37.23 dB
[Q2] size=537932 bytes, time=6.444s, PSNR=34.33 dB
```

(3) Visualization

4. Conclusion

實驗結果表明，在相同DCT與編碼流程下，量化表的選擇會明顯影響壓縮率與重建畫質：使用數值較小、較細緻的量化表時，重建影像的PSNR較高、視覺細節保留較好，但編碼後檔案尺寸也相對較大；相反地，較粗的量化表能有效減少非零係數與RLE後的資料長度，顯著縮小編碼檔案大小，代價則是PSNR降低與失真加劇。

5. Supplementary

(1) Command Line: python main.py

(2) Output Files:

(a) encoded_images:

- **(i) qtable1.pkl:** 使用Q1時，所有8×8區塊經zigzag + RLE之後的bitstream，以pickle序列化存成檔案，檔案大小用來估計壓縮後尺寸。
- **(ii) qtable2.pkl:** 使用Q2對應的RLE bitstream，格式同qtable1.pkl

(b) decoded_images:

- **(i) original.png:** 原始的灰階Lena影像
- **(ii) recon_q1.png:** 使用量化表Q1壓縮後，再經反量化與IDCT還原得到的重建影像。
- **(iii) recon_q2.png:** 使用量化表Q2壓縮後還原的重建影像。
- **(iv) comparison.png:** Original、Q1 reconstructed、Q2 reconstructed的對照圖