CODECs não destrutivos para imagens monocromáticas

Guia de utilização de código

Trabalho prático nº2 da cadeira Teoria da Informação, 2º ano, 1º semestre

João Afonso Vieira de Sousa, 2019224599, uc2019224599@student.uc.pt

José Domingues da Silva, 2018296125, uc2018296125@student.uc.pt

Sancho Amaral Simões, 2019217590, uc2019217590@student.uc.pt

Tiago Filipe Santa Ventura, 2019243695, uc2019243695@student.uc.pt

20 de novembro de 2020

CMP

O *CMP* é um *CODEC* testável e manipulável desenvolvido por alunos do 2º ano da Licenciatura de Engenharia Informática que permite comprimir imagens *.bmp* de 8 *bits* em tons de cinzento. Recomenda-se que seja efetuada a análise da documentação do código, em especial da do ficheiro *BPMCodec.py*, para além da leitura deste documento. As instruções abaixo enumeradas assumem que o utilizador possui uma cópia do repositório criado no âmbito da realização deste trabalho prático.

1. Instalação / Configuração

a) Abra uma *IDE (Integrated Development Environment)* que suporte a linguagem *Python* (ex.: *PyCharm*, *WingIDE*, *Spyder*, etc.).

b) Abra o diretório *TI-TP2* na *IDE* escolhida no passo acima.

O diretório *TI-TP2* aparecerá na aba de navegação do seu *IDE*.

c) Aceda ao *script Python Main.py* localizado em *TI-TP2/source_code/cmp.*

d) Certifique-se que o caminho absoluto do *script* a executar especificado pela *IDE* coincide com o caminho absoluto do *script Main.py*.

e) Certifique-se que o caminho absoluto do diretório atual (*working directory*) especificado pela *IDE* coincide com o caminho absoluto do diretório onde o *script Main.py* se encontra.

f) Procure pela região (*region*) *Constants* no *script Main.py*.

As constantes definidas nessa região são *TO_COMPRESS_PATH*, *COMPRESSED_PATH*, *TO_DECOMPRESS_PATH, DECOMPRESSED_PATH* que indicam, respetivamente, o diretório de ficheiros alvo a comprimir, o diretório onde os ficheiros comprimidos serão colocados, o diretório de ficheiros alvo a descomprimir e o diretório dos ficheiros descomprimidos.

g) Altere os caminhos relativos especificados pelas constantes por outros, caso não deseje manter os valores predefinidos.

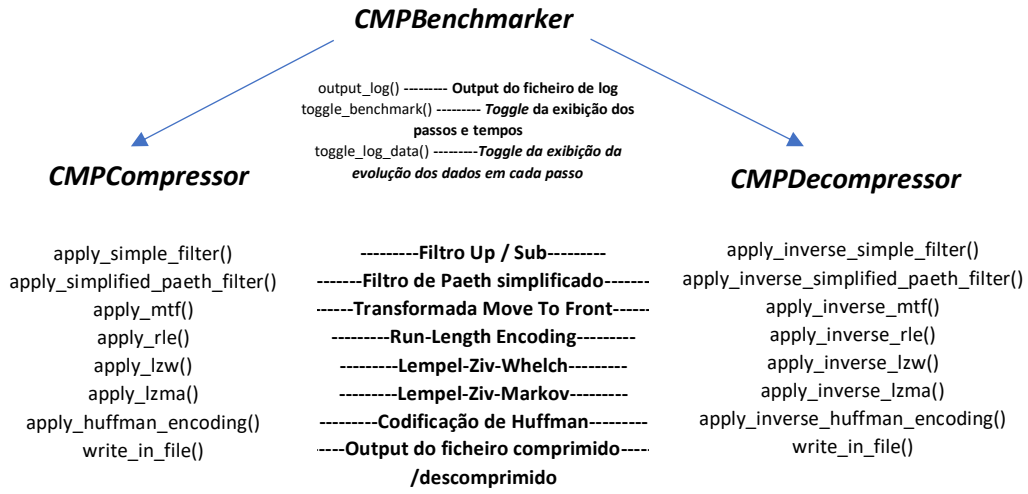
2. Modo de utilização

No *script Main.py* encontram-se as funções *compress_files()* e *decompress_files()*. A primeira efetua a compressão dos ficheiros de extensão *.bmp* dentro do diretório definido pela constante *TO_COMPRESS_PATH* e a segunda descomprime os ficheiros de extensão *.cmp* localizados no diretório especificado pela constante *TO_DECOMPRESS_PATH.* Em cada uma destas funções são instanciadas as classes *BMPCompressor* e *CMPDecompressor* que representam o compressor e descompressor, respetivamente, de modo a que o utilizador tenha acesso às diretivas de compressão/descompressão fornecidas pelo *CODEC CMP*.

2.1 – Diretivas e funcionalidades fornecidas pelas classes *CMPCompressor* e *CMPDecompressor*

Ambas as classes referidas estendem a classe *CMPBenchmarker* que, por sua vez, fornece funções e variáveis para a finalidade de *logging*/*benchmarking*.

Abaixo encontra-se um diagrama que explica sucintamente as funcionalidades disponíveis ao utilizador nas classes acima referidas.



Nota: Recomenda-se consulta da documentação das funções acima referidas para melhor entender o seu funcionamento e seus parâmetros.

O utilizador pode escolher várias combinações de transformações aplicadas aos dados alvo, a fim de conseguir apurar qual o algoritmo mais eficiente para a compressão de uma determinada imagem.

2.2 – *Compression* e *decompression stack*

Convém salientar que o utilizador deve certificar-se que a *compression stack* na função *compress_files()* é reversa e inversa da *decompression stack* da função *decompress_files()*, como abaixo ilustrado em pseudocódigo.

compress_files()	***decompress_files()***
t_0	$(t_n)^{-1}$
t_1	.
.	.
.	.
.	$(t_1)^{-1}$
t_n	$(t_0)^{-1}$

$$t_i, \forall i, 0 \leq i \leq n$$

é uma transformação qualquer invertível fornecida pelo *CMP* aplicada aos dados em questão.

2.3 – *Logging* e *Benchmarking*

Para uma recolha mais fácil de dados acerca da *performance* da *compression*/*uncompression stack* utilizada, como tempo de compressão/descompressão e taxa de compressão, o *CMP* dispõe de mecanismos de *logging* e *benchmarking*.

Um ficheiro *.txt* é criado no diretório especificado pelas constantes *COMPRESSED_PATH,* aquando da compressão de uma imagem, e *DECOMPRESSED_PATH*, aquando da descompressão de uma imagen, somente se a função *output_log()* for chamada.

Exemplos do formato dos ficheiros de *log*:

-------crop.bmp CMP COMPRESSION LOG-------

COMPRESSION STACK:

-> MTF

-> RLE

-> HUFFMAN ENCODING

TOTAL ELLAPSED COMPRESSION TIME: 0.57 sec.

INITIAL IMAGE SIZE: 117586 bytes

COMPRESSED IMAGE SIZE: 21117 bytes

COMPRESSION RATIO: 82.04%

-------crop.cmp CMP DECOMPRESSION LOG-------

DECOMPRESSION STACK:

-> INVERSE HUFFMAN ENCODING

-> INVERSE RLE

-> INVERSE MTF

TOTAL ELLAPSED UNCOMPRESSION TIME: 0.34 sec.

• Durante a compressão e descompressão das imagens, podem ser apresentados na consola os eventos que estão a ocorrer, as respetivas durações totais e os dados transformados. O *toggling* desta ação pode ser efetuado recorrendo à definição dos parâmetros opcionais *benchmark* e *log_data* dos construtores das classes *CMPCompressor* e *CMPDecompressor* ou através das funções *toggle_benchmark()* e *toggle_log_data()*.

Exemplos do formato do *output* de *logging/benchmarking* na consola:

```
--------------------
crop.bmp Compression
--------------------
Applying up filter...
Ellapsed up filtering time: 0.00 sec
Applying RLE encoding...
Ellapsed RLE encoding time: 0.05 sec
Applying Huffman encoding...
Ellapsed huffman encoding time: 0.02 sec
Total ellapsed compression time: 0.07 sec
Writing in file crop.cmp...
--------------------
```