IC Project #01
P1

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758

Relatório

Introdução

Neste Relatório são abordados todos os exercícios realizados do Projeto #01.
São utilizadas bibliotecas como AudioFile e OpenCV. Os exercícios que utilizam a biblioteca OpenCV necessitam que esta biblioteca esteja instalada bem como o CMake.
Código do Projeto: https://github.com/miguelf18/IC
Foram criadas duas pastas no repositório, uma com ficheiros de áudio e outra com imagens para que se possam testar os programas.

Parte A

Exercício 1 - Para o exercício 1 foram feitos dois tutoriais de C++, um sobre os aspetos mais básicos e gerais e outro sobre STL (containers, algoritmos e iteradores).

Parte B

Exercício 2 - No exercício 2 o objetivo é copiar o conteúdo de um ficheiro de texto, caracter a caracter, para outro ficheiro de texto.
Assim, o que o programa faz é, enquanto há linhas, copia caracter a caracter o conteúdo do ficheiro de entrada para o ficheiro de saída.

Exercício 3 - No exercício 3 o objetivo é copiar um ficheiro áudio no formato wav, sample a sample, para outro ficheiro com o mesmo formato.
O que o programa faz é, dado o número de Channels e de Samples do ficheiro áudio, para todas as samples de todos os canais, copia sample a sample todas as samples existentes para outro ficheiro.

Exercício 4 - No exercício 4 o objetivo é copiar uma imagem, pixel a pixel, para outro ficheiro.
O que o programa faz é, dada uma matriz com várias linhas e várias colunas, copia pixel a pixel, todas linhas e todas as colunas da matriz para outro ficheiro.

IC Project #01
P1

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758

Parte C

Exercício 5 - No exercício 5, o objetivo é calcular o histograma de letras existentes num ficheiro de texto, bem como a sua entropia total.

Para este efeito, foi criado um programa que lê uma amostra de texto, de um ficheiro fornecido, e que calcula o número de ocorrências de cada letra encontrada que, neste exemplo, pode ser visto no histograma da **Figura 1**. Sabendo o número de ocorrências de cada palavra, é possível calcular as respectivas probabilidades e finalmente a entropia total do ficheiro de texto, assumindo que os símbolos possíveis correspondem às próprias letras. Assim sendo, a entropia total é a seguinte:

$$H(letras) = -\sum_{i=1}^{n} P(letras(i)) * log_2(P(letras(i)))$$

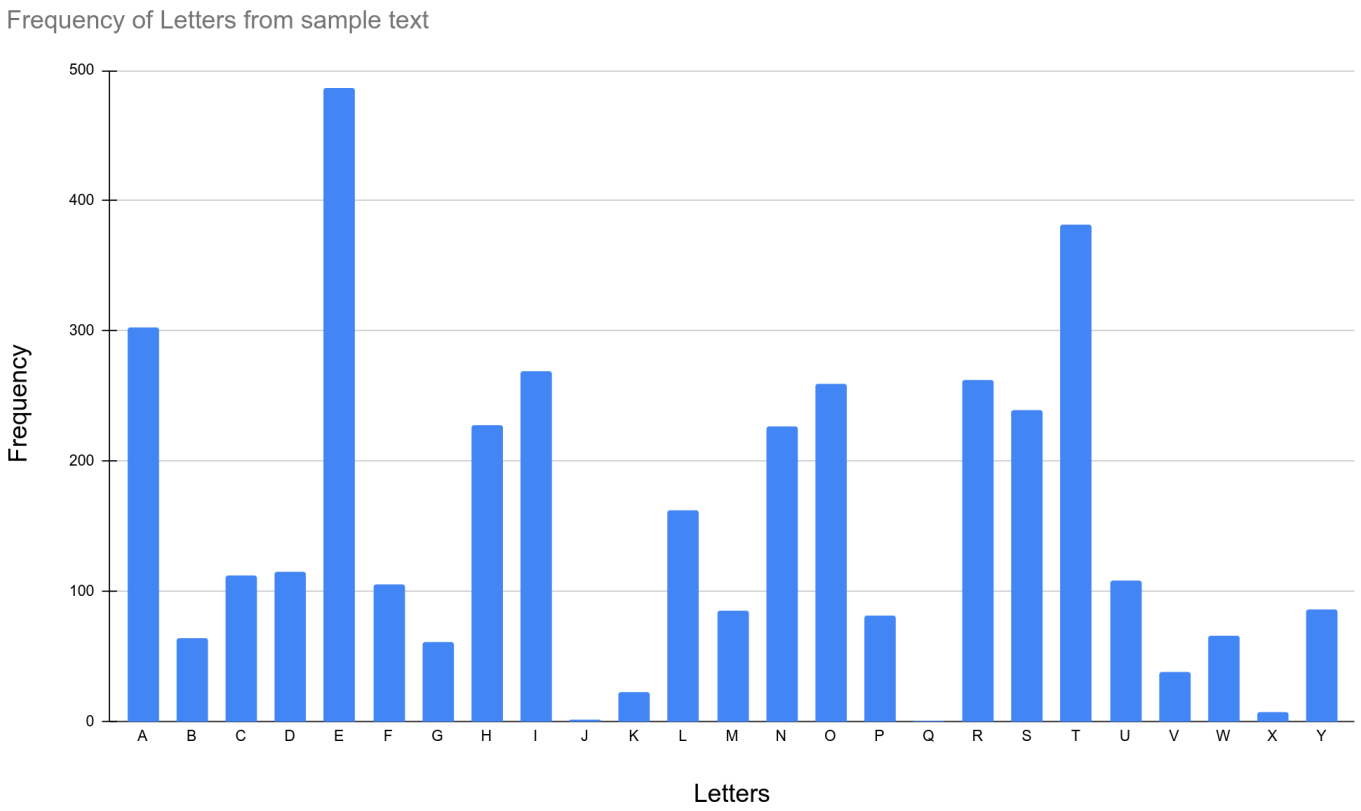

Figura 1. Histograma do ficheiro sample.txt

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758

Exercício 6 - No exercício 6 o objetivo é calcular o histograma de uma amostra de áudio e a entropia correspondente.
Para este exercício foram calculados o número de canais e o número de samples de cada canal da amostra de áudio. Depois de calculados, foi criado um mapa de ocorrências para determinar o número de vezes que cada sample aparece.
Sabendo o número de vezes que cada sample aparece, sabemos que a sua probabilidade de ocorrência é igual ao número de vezes que aparece a dividir pelo número total de canais vezes o número total de samples.

Ao descobrir as probabilidades da ocorrência de cada sample, é possível então calcular a entropia, que é dada por, $I(E) = -log_2(p(E))$, sendo p(E) a probabilidade de ocorrência de cada sample.
O programa ao ser executado é gerado um ficheiro .txt com o histograma.

ex06 > histograma.txt

```
20863   Sample 0.166321 aparece 22 vezes
20864   Sample 0.166351 aparece 34 vezes
20865   Sample 0.166382 aparece 35 vezes
20866   Sample 0.166412 aparece 21 vezes
20867   Sample 0.166443 aparece 37 vezes
20868   Sample 0.166473 aparece 30 vezes
20869   Sample 0.166504 aparece 15 vezes
20870   Sample 0.166534 aparece 39 vezes
20871   Sample 0.166565 aparece 44 vezes
20872   Sample 0.166595 aparece 30 vezes
20873   Sample 0.166626 aparece 15 vezes
20874   Sample 0.166656 aparece 28 vezes
20875   Sample 0.166687 aparece 26 vezes
```

Figura 2. Histograma do ficheiro sample05.wav

Exercício 7 - No exercício 7, o objetivo é calcular o histograma de uma imagem RGB e a sua versão Grayscale, seguido da sua entropia total.
De modo análogo aos exercícios anteriores, o programa criado calcula as ocorrências das intensidades de cada pixel para construir o histograma, tendo em conta que para uma imagem RGB é calculado os histogramas de cada canal (para os testes, utilizou-se um exemplo em formato .ppm). Além disso, o programa também converte uma imagem RGB na sua versão Grayscale utilizando os valores do Luma(video) coding. Os histogramas da imagem RGB podem ser vistos por canal na **Figura 3, Figura 4** e **Figura 5** e a versão Grayscale na **Figura 6**. No caso da entropia, existem 2 casos: imagem Grayscale e RGB. No caso da imagem Grayscale, assume-se que os símbolos possíveis correspondem à intensidade de cada pixel. Já na imagem RGB, assume-se que os símbolos possíveis correspondem aos trios de intensidade de cada pixel referentes a cada channel. Independentemente, a entropia será dada por:

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758

$$H(pixels) = -\sum_{i=1}^{n} P(pixels(i)) * log_2(P(pixels(i)))$$

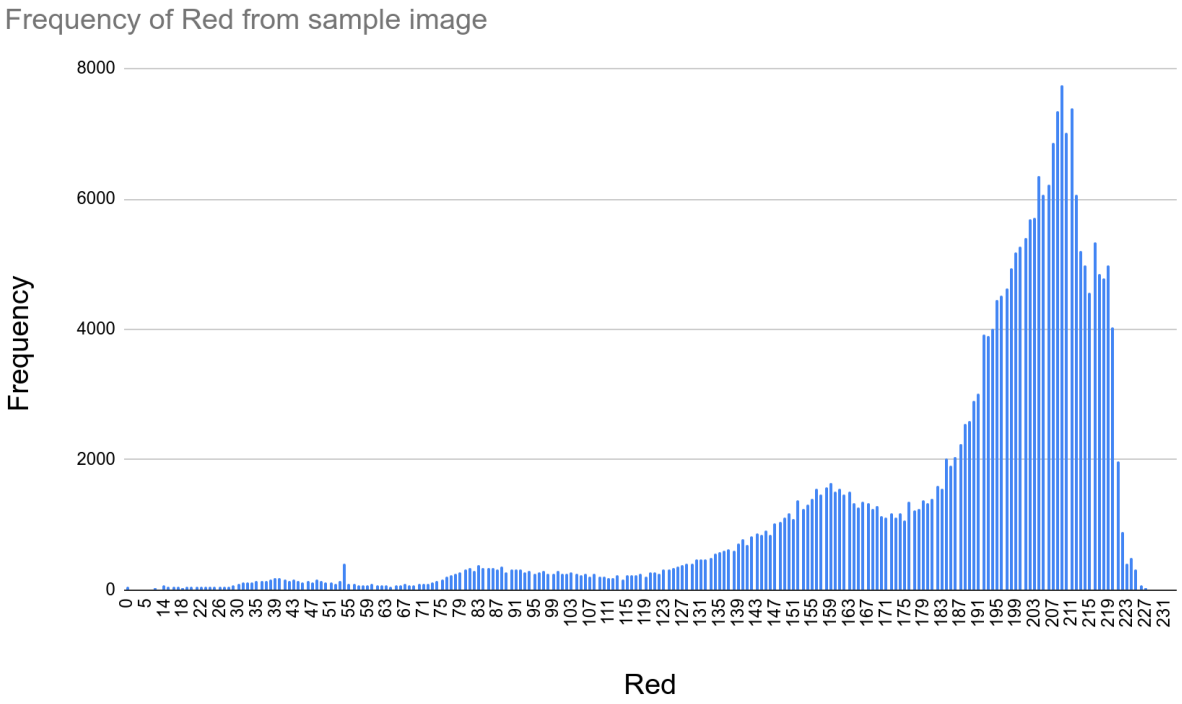

Figura 3. Histograma do ficheiro airplane.ppm (Canal vermelho)

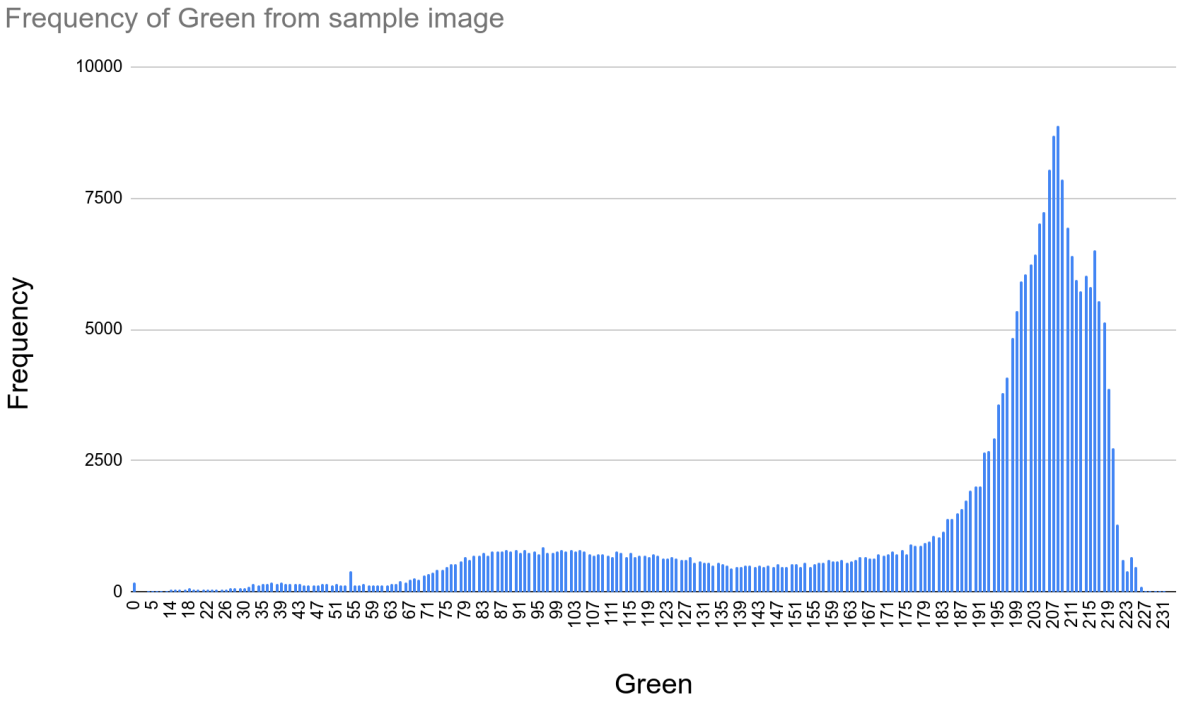

Figura 4. Histograma do ficheiro airplane.ppm (Canal verde)

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758

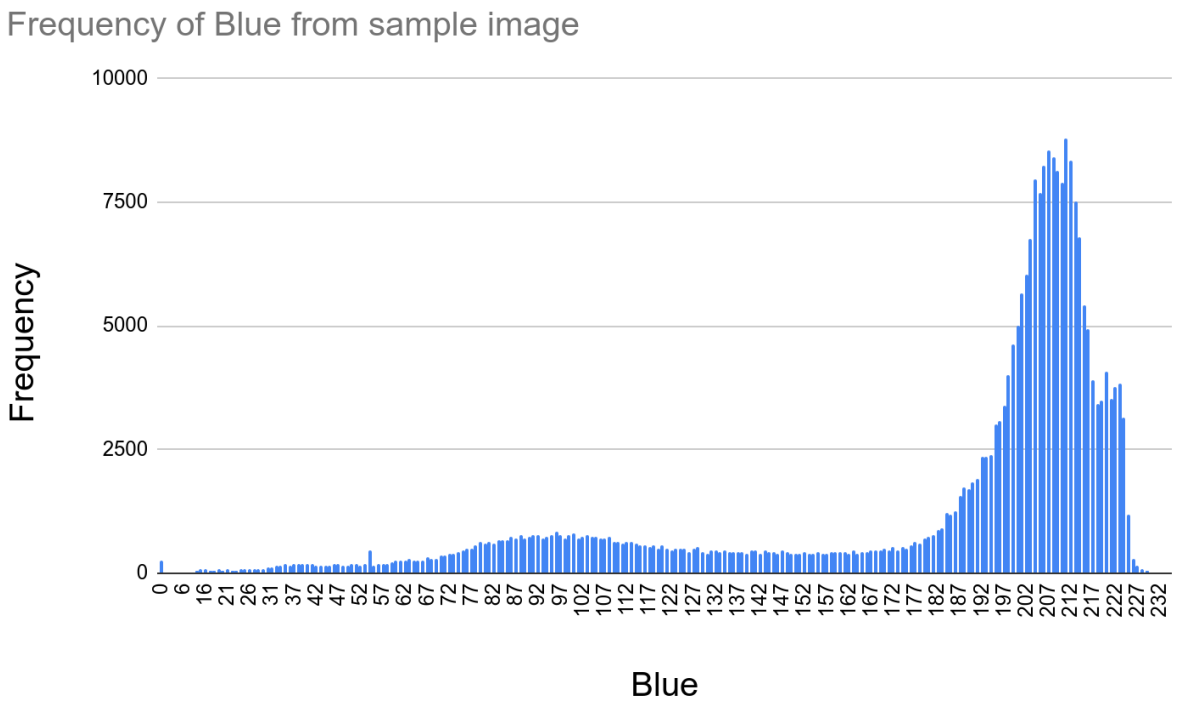

Figura 5. Histograma do ficheiro airplane.ppm (Canal azul)

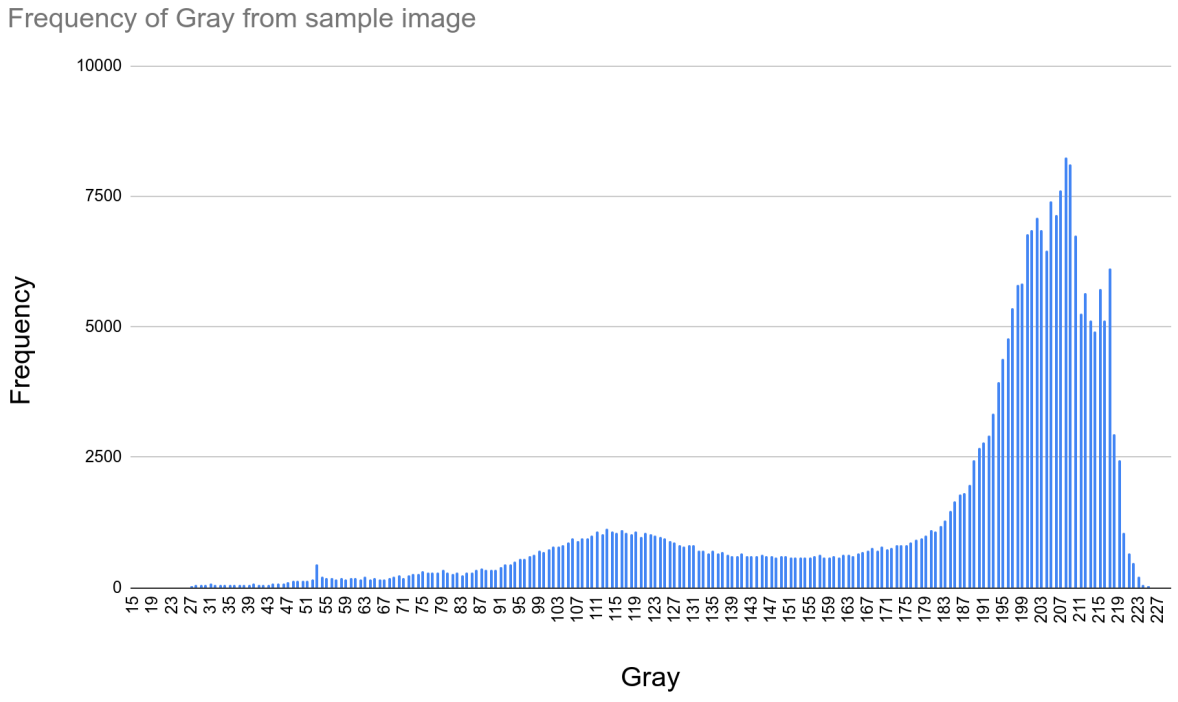

Figura 6. Histograma do ficheiro airplane.ppm (Grayscale)

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758

Parte D

Exercício 8 - No exercício 8 o objetivo é reduzir o número de bits utilizados para representar cada audio sample.
Para isso, manteve-se o número de bits e reduziu-se a informação presente através de um deslocamento à direita de 1 bit em cada sample, seguido de um deslocamento à esquerda do mesmo número de bits.

Exercício 9 - No exercício 9, o objetivo é reduzir o número de bits por pixel (bpp) e, consequentemente o número de cores utilizadas na imagem, através de quantização. Para tal foi utilizado OpenCV e também quantizadores uniformes mid-riser e mid-tread.
Assim, este programa recebe um ficheiro de imagem e o número de bpp como argumentos e o output são dois novos ficheiros de imagem, um com a imagem quantizada através de mid-riser e outro através de mid-tread.

O quantizador uniforme mid-riser implica truncação e é representado pela seguinte fórmula:

$$Q(x) = \Delta * \left(\lfloor \frac{x}{\Delta} \rfloor + \frac{1}{2} \right)$$

O quantizador uniforme mid-thread implica arredondamento e é representado pela seguinte fórmula:

$$Q(x) = \Delta * \lfloor \frac{x}{\Delta} + \frac{1}{2} \rfloor$$

Exemplo de quantização de imagem:

Figura 7. Imagem original

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758


Figuras 8 e 9. Imagens quantizadas com 4 bpp mid-riser e mid-thread respetivamente


Figuras 10 e 11. Imagens quantizadas com 2 bpp mid-riser e mid-thread respetivamente


Figuras 12 e 13. Imagens quantizadas com 1 bpp mid-riser e mid-thread respetivamente

Exercício 10 - No exercício 10, o objetivo é calcular o signal to noise ratio (**SNR**) de um ficheiro de áudio em relação ao ficheiro original e o seu erro máximo absoluto.
Para isso, foi utilizado um ficheiro sample em que o seu volume foi diminuído por um fator de 0.3. O programa determina o erro máximo absoluto analisando os erros entre o ficheiro original e o modificado e efetua os cálculos necessários para obter o SNR, seguindo a seguinte fórmula:

$$SNR = 10 * log_{10}(\frac{(sinal\ máximo)^2}{(e)^2})$$

em que

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758

$$(e)^{2} = \frac{1}{N*M} * \sum_{i=0}^{N} \sum_{k=0}^{M} [f(i,k) - f^{\neg}(i,k)]^{2}$$

onde, N = número de canais; M = número de samples for canal; $f(i,k)$ = dados do ficheiro original; $f^{\neg}(i,k)$ = dados do ficheiro modificado.

Exercício 11 - No exercício 11, o objetivo é calcular o signal to noise ratio (**SNR**) de um ficheiro de imagem em relação ao ficheiro original bem como o seu erro máximo absoluto. Para calcular o SNR é necessário saber o MSE(Mean Squad Error) que é dado por,

$$MSE=\frac{1}{c*i*j}\sum(I_1 - I_2)^{2}$$

sendo I_1 e I_2 duas imagens com duas dimensões de tamanho i e j , compostas por c canais.
sabendo o MSE, é possível então calcular o SNR, que é dado por

$$SNR= 10 * log_{10}(\frac{MAX_I^{2}}{MSE})$$

sendo MAX_I o valor máximo válido para um pixel. No caso de a imagem ter apenas um byte por pixel por canal, o valor de MAX_I é 255.

Conclusão

Com este projeto foram desenvolvidos conhecimentos acerca de manipulação de texto, áudio e imagem, bem como as bibliotecas utilizadas para tal.

Referências

- Documentation oficial de C++: https://en.cppreference.com/w/
- Tutorial básico C++: https://www.tutorialspoint.com/cplusplus/index.htm
- Tutoria STL C++: https://www.tutorialspoint.com/cplusplus/cpp_stl_tutorial.htm
- Biblioteca AudioFile: https://github.com/adamstark/AudioFile
- Biblioteca OpenCV: https://opencv.org/
- CMake: https://cmake.org/

João Oliveira 93295
Miguel Fernandes 93284
Rafael Oliveira 84758

- Fórmula PSNR: http://amroamroamro.github.io/mexopencv/opencv/image_similarity_demo.html#2
- Mapa de ocorrências: https://luksamuk.codes/posts/counting-occurencies.html
- Cálculo da entropia: https://en.wikipedia.org/wiki/Entropy_(information_theory)
- Algoritmos de quantização uniforme: https://en.wikipedia.org/wiki/Quantization_(signal_processing)