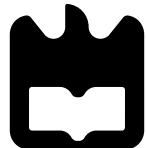


Relatório de IC

Projeto 2

Universidade de Aveiro



DETI

Tiago Pedrosa
93389

João Gonçalves
98287

2 de dezembro de 2022

Conteúdo

1 Parte I	2
1.1 Cópia de imagem e aplicação de efeitos	2
1.1.1 Versão negativa	2
1.1.2 Versão espelhada	3
1.1.3 Versão com rotação	3
1.1.4 Versão com alteração na saturação	4
2 Parte II	5
2.1 Golomb	5
3 Parte III	6
3.1 Codecs de audio	6
3.1.1 Lossless audio	6
4 Parte IV	7
4.1 Codec de imagem lossless	7
5 Contribuições dos Autores	8

Capítulo 1

Parte I

1.1 Cópia de imagem e aplicação de efeitos

Implementou-se um programa que através da manipulação de bits possibilitou a cópia de imagens pixel a pixel e posteriormente a aplicação de diversos efeitos, tais como:

1.1.1 Versão negativa

Começou-se pela criação de uma imagem com um efeito negativo aplicado à sua imagem original, para tal efetuou-se uma cópia da imagem original pixel a pixel como se fez anteriormente, mas durante essa cópia efetuou-se, em cada pixel, a subtração do valor 255 (valor máximo de rgb) pelo valor RGB de cada pixel, ou seja, pelo valor de R (vermelho), de G (verde) e de B (azul), ficando assim com o valor RGB oposto do da imagem original.

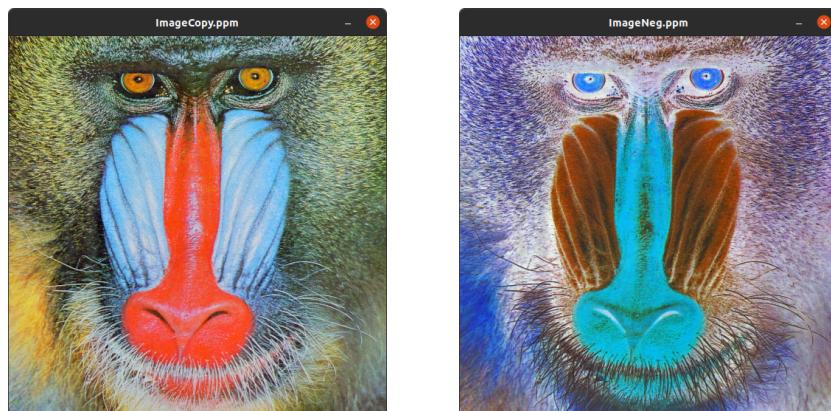


Figura 1.1: Figura original e figura com efeito negativo

1.1.2 Versão espelhada

Para se efetuar a reflexão da imagem, durante a cópia da imagem original, mantém-se os valores das linhas mas subtrai-se o valor total de colunas ao valor atual para efetuar a cópia da imagem e inverter o lado das colunas.

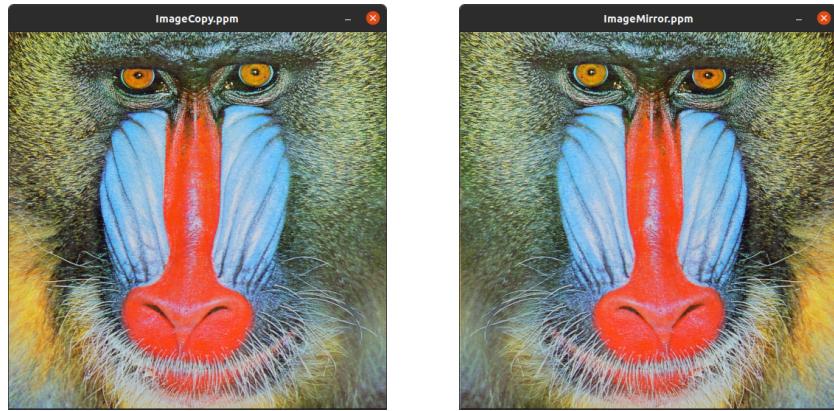


Figura 1.2: Figura original e figura refletida

1.1.3 Versão com rotação

Para efetuar a rotação da imagem, durante a cópia da imagem, efetua-se rotações de 90 em 90 graus. A rotação de 90 graus obteve-se com a cópia da imagem em que o valor da linha é a subtração do valor total das linhas pela coluna e a coluna corresponde à linha, a rotação de 180 copia-se os pixeis de cima para baixo ficando com as colunas iguais mas com as linhas invertidas, e a rotação de 270 substitui as linhas pelas colunas e os valores das colunas pelo máximo de colunas subtraído pelas linhas.

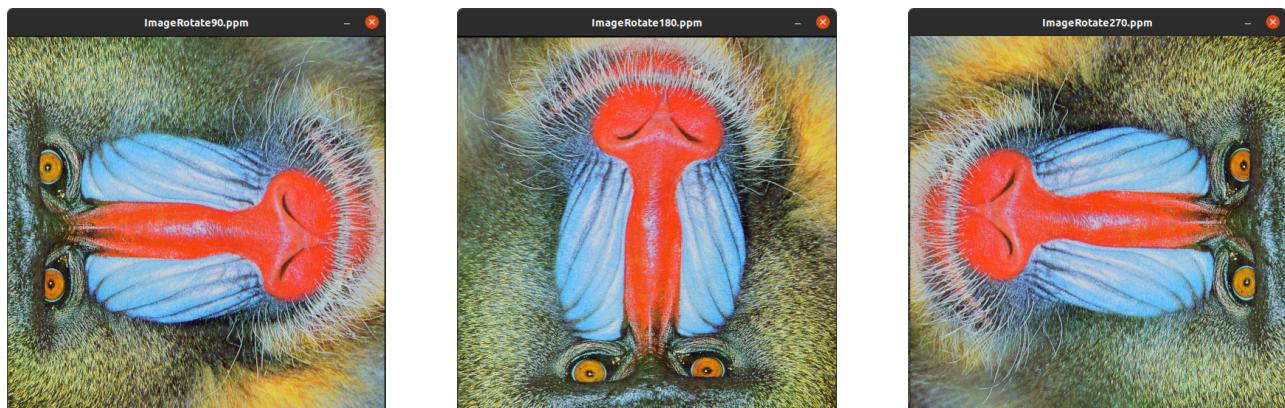


Figura 1.3: Figura com 90° , figura com 180° e figura com 270°

1.1.4 Versão com alteração na saturação

Em relação à saturação da imagem original, é possível alterar-la pegando nos valores **RGB** de um pixel e multiplicando os por uma escala à escolha, se esta escala for maior que 1 a imagem fica mais clara, se for menor que 1 mais escura. Tendo em conta que o valor de r,g e b não pode ser maior que 255 utilizou-se a função **saturate_cast** que impede que tal aconteça.

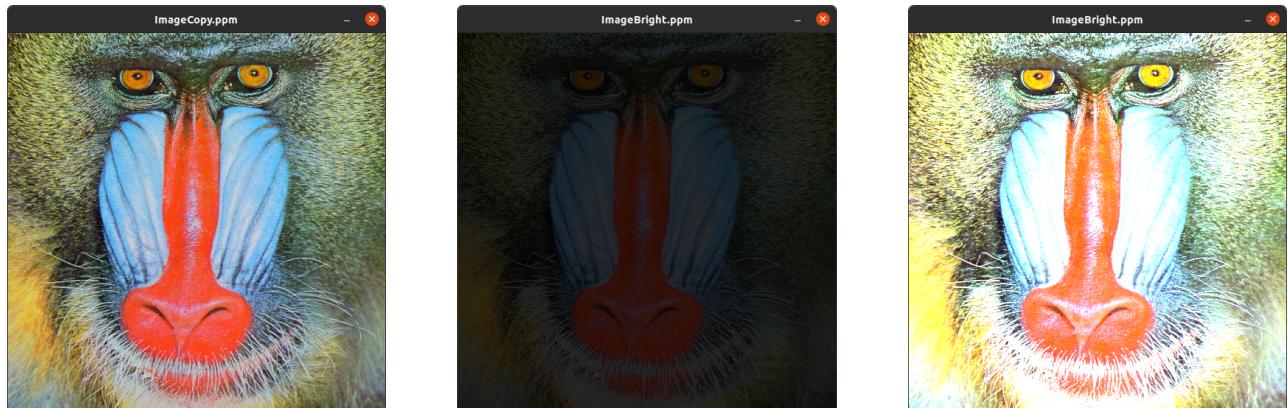


Figura 1.4: Figura original, figura com baixa saturação e figura com alta saturação

Capítulo 2

Parte II

2.1 Golomb

A codificação de Golomb permite gerar um conjunto de códigos de tamanho variável, esta é obtida ao separar um inteiro em duas partes a parte unário e a parte binária.

Para codificar um número inteiro i , faz -se o seguinte

- $q = \lfloor \frac{i}{m} \rfloor$
- $r = i - qm$

Sendo q o quociente, que pode ter os valores $0, 1, 2, \dots$ e será representado pelo respetivo código unário. Sendo r o resto da divisão que pode assumir os valores $0, 1, 2, \dots, m-1$ e irá ser representado pelo respetivo código binário, isto para quando m é potência de 2.

Se m não é potência de 2 a parte binária faz-se do seguinte modo:

- Definir $b = \lceil \log_2 m \rceil$
- $r < 2^b - m$, codificar os primeiros valores de r em binário usando $b-1$ bits
- Senão, $r + 2^b - m$, codificar r em binário com b bits

Para efetuar a descodificação, esta será feita através da leitura do valor escrito num ficheiro e contar o número de 1 que tem, o código unário. Ao obtermos o número de 1s, que representamos com U , conseguimos saber o tamanho do código unário que é $U + b + 1$ caso m seja uma potência de 2, os restantes $c + 1$ bits são o R e ao convertermos este número para decimal irá se obter o valor do código através de $mU + R$.

Caso m não seja uma potência de 2, após se contar o número de 1s obtemos R em decimal que é representado pelo valor de $c-1$ bits seguintes ao código unário, por ultimo para o $R < 2^c - m$ o valor do código é $mU + R$ senão tem de se considerar que os c bits seguintes ao código unário são R e o valor final obtem-se através de $mU + R(2^c - m)$.

Capítulo 3

Parte III

3.1 Codecs de audio

3.1.1 Lossless audio

Para conseguir uma melhor compressão dos dados de um ficheiro, é usado o conceito de preditores. Preditores vão utilizar valores anteriormente escritos durante o processo de escrita num ficheiro binário e utilizá-los de forma a representar próximos. É dado como possibilidade usar um de 3 preditores lossless, a diferença entre eles sendo o número de valores anteriores usados na representação de seguintes (1, 2 ou 3).

$$\begin{cases} \hat{x}_n^{(0)} = 0 \\ \hat{x}_n^{(1)} = x_{n-1} \\ \hat{x}_n^{(2)} = 2x_{n-1} - x_{n-2} \\ \hat{x}_n^{(3)} = 3x_{n-1} - 3x_{n-2} + x_{n-3} \end{cases}$$

Capítulo 4

Parte IV

4.1 Codec de imagem lossless

Capítulo 5

Contribuições dos Autores

O Trabalho foi analisado, discutido e realizado de modo homogéneo por:

- Tiago Pedrosa 93389
- João Gonçalves 98287