Projeto Final - Video Encoding

Trabalho Realizado por: Pedro Oliveira - nº mec 89156
Diogo Silva - nº mec 89348

Data de Entrega: Aveiro, 6 de Janeiro de 2020

Cadeira: Complementos Sobre Linguagens de Programação

universidade de aveiro deti departamento de electrónica, telecomunicações e informática

Índice

1. Introdução **3**

2. Objetivos Atingidos **4**

BitStream **4**

Golomb **4**

Video Player **4**

Lossless Intra-Frame Video Encoder **4**

3. Como Correr **5**

4. O que podia ter corrido melhor **6**

5. Conclusão **6**

1. Introdução

Este projeto foi desenhado como projeto final de Complementos de Linguagem e Programação, tendo como objetivo o ensino a aprendizagem de técnicas de compressão de vídeo e tratamento de bits no paradigma de Programação Orientada a Objetos para Python e C++.

A entrega inicial deste projeto tinha sido pensada para o final do 1o semestre, porém foi adiada para o início da Época de Exames

O projeto pode ser encontrado no repositório:

https://github.com/HerouFenix/data-compression-and-video-coding

2. Objetivos Atingidos

BitStream

Foi o primeiro módulo que implementa-mos, tanto em Python com C++

Implementada nos ficheiros **Bitstream.py** e **Bitstream.cpp**

Golomb

Depois do Bitstream partimos logo para o Golomb, tendo implementado este em Python e C++.

Deve-se notar que tivemos que realizar várias alterações à versão do C++ quando começamos a trabalhar no Lossless Intra-Frame Video Encoder

Está implementada nos ficheiros **Golomb.py** e **Golomb.cpp**

Video Player

O Video-Player está implementado tanto em Python como em C++

Em C++ dependemos muito mais das funções do OpenCV do que em Python, devido às restrições de tempo e conhecimento sobre tratamento matricial em C++ que possuíamos.

Tanto em Python como em C++ a função **play_video** da Classe **VideoCodec** presente nos ficheiros **VideoCodec.py** e **VideoPlayer.cpp**.

Lossless Intra-Frame Video Encoder

A compressão e decompressão de um ficheiro de vídeo utilizando Lossless Intra-Frame encoding foi implementado tanto em Python como em C++.

São permitidos todos os modos de compressão lecionados (JPEG 1 até 7 e JPEG-LS)

As frames dos vídeos são representadas pela nossa superclass **Frame** (e, dependendo do formato do vídeo, 4:2:0, 4:2:2, 4:4:4, pelas subclasses Frame<Format>, p.ex frames de vídeos 4:2:0 são representadas pela classe Frame420).

Os ficheiros responsáveis pela codificação do vídeo são então o **VideoCodec.py/VideoPlayer.cpp** e o **Frame.py/Frame.cpp** (sendo que, claro, são também usados os módulos de BItsTream e Golomb criados anteriormente)

3. Como Correr

GitHub: https://github.com/HerouFenix/data-compression-and-video-coding

Python:

Para testar o Video Player, Compressor e Descompressor basta correr o VideoCodec:

python3 VideoCodec.py

Para testar o Bitstream e o Golomb basta correr o Bitstream ou Golomb, correspondientemente:

python3 Bitstream.py

python3 Golomb.py

C++:

Para testar o Video Player, Compressor e Descompressor basta correr o nosso script run.sh:

./run.sh

Para testar o Bitstream e o Golomb basta correr o Bitstream ou Golomb, correspondientemente:

python3 Bitstream.py

python3 Golomb.py

4. O que podia ter corrido melhor

Existem vários módulos e features que nós não pudemos completar devido às constrições de tempo. Se estas restrições não fossem um problema, nós poderíamos melhorar a performance dos módulos, implementando uma melhor alocação de memória para as estruturas de dados implementadas; implementar compressão interframe, tirando partido de codificação de vetores de movimento e diferença de matrizes; e melhorar a documentação do projeto com Doxyfile, que foi iniciada no início do projeto, mas que não a pudemos completar.

5. Conclusão

Chegando então ao término deste trabalho, consideramos que o que foi pedido de nós foi demasiado ambicioso tendo em conta a consolidação com os projetos e trabalhos de outras Cadeiras. Despendemos de bastante carga horária tanto durante a época de aulas como durante a época de exames, mas infelizmente não conseguimos passar para além do 4o exercício (Video Coding Ex 2) do guião, tanto por causa de nunca termos trabalhado com a linguagem C++, módulos OpenCV e Numpy e pela dificuldade na resolução dos vários bugs que foram surgidos ao longo do projeto.