Universidade de Aveiro

Adriana Rocha, Joana Silva

deti
universidade de aveiro
departamento de eletrónica,
telecomunicações e informática

Projeto de Algoritmo e Estruturas de Dados

DETI

Universidade de Aveiro

Adriana Rocha, Joana Silva
108002, 100006

Licenciatura em Engenharia de Computadores e Informática
Licenciatura em Engenharia Computacional
23 de novembro de 2023

Conteúdo

Capítulo 1

Introdução/Planeamento do Trabalho

Neste trabalho, o nosso objetivo é desenvolver um algoritmo que opera com imagens a preto e branco, onde cada píxel pode tomar um valor de intensidade entre 0 e 255. Para tal, tivemos que desenvolver duas grandes funções.

A função ImageLocateSubImage() permite determinar a localização de uma sub imagem numa imagem fornecida (caso exista), ou seja, tendo como exemplo a imagem1 e a imagem2, sempre que o utilizador chamar esta função, irá perceber se os pixeis da imagem2 são iguais aos pixeis da imagem1, se forem, a função guarda os valores dos pixeis e retorna *true*.
A função ImageBlur() desfoca uma imagem, pois ao percorrer cada pixel da imagem calcula a média dos pixeis à sua volta, e atribui esse valor ao pixel em questão.

Capítulo 2

Análise

2.1 Análise da complexidade da função ImageLocateSubImage()

2.1.1 Análise Formal

Esta função procura por uma sub imagem (img2) dentro de outra imagem (img1).
Primeiramente, confirmamos se os ponteiros img1 e img2 não são *NULL*. Para isso fazemos "assert()". Antes de prosseguir com a execução da função, verifica-se se os ponteiros img1 e img2 diferem de *NULL*. Essas asserções são importantes para garantir que os ponteiros de imagem sejam válidos e para evitar comportamento indefinido.
De seguida, usamos dois ciclos *for* que percorrem todas as posições da sub imagem na imagem. O primeiro *for* tem o intervalo [0; img1-width - img2-width] e o segundo ciclo *for* tem como intervalo $[0; img1 - height - img2 - height]$. Estes limites garantem que img2 não ultrapassa os limites de img1.
Imediatamente após, temos um ciclo *if* que usa a função ImageMatchSubImage(), vamos explorar esta função.
ImageMatchSubImage() compara os píxeis um a um de img2 com os píxeis de img1. Ela irá retornar *true* se todos os píxeis corresponderem aos píxeis de img1, mas basta um único píxel não corresponder para a função retornar *false*.
Após recebermos o feedback desta função, se tivermos um *return true*, as variáveis *px* e *py* são atualizadas para a posição atual de img2.
Se a correspondência for encontrada, ImageLocateSubImage() irá retornar *true*, caso contrário, irá retornar *false*.

2.2 Análise da complexidade da função ImageBlur()

2.2.1 Análise Formal

Esta função serve para desfocar uma imagem. Para fazermos isso, cada píxel será substituído pela média dos píxeis no retângulo com largura $[x - dx, x + dx]$ e comprimento $[y - dy, y + dy]$.
Quando o utilizador chama esta função, ela irá verificar se a imagem existe (utilizando a função *assert()*), se $dx = 0$, e $dy = 0$, pois as variáveis não podem ser negativas.
No desenvolvimento desta função começamos por criar uma cópia da imagem original, depois percorremos Píxel a Píxel da imagem, e para cada Pixel, percorremos os Pixeis que estavam no retângulo anteriormente referido.
Por cada ciclo /textitfor, verificamos se os Pixeis do retângulo efetivamente existiam, pois, na primeira linha de Pixeis, se o $dy! = 0$, os Pixeis acima da primeira linha não existem. Logo, aplicamos um /textitif, com a condição $(ImageGetPixel(img, k, l)! = 0)$, esta condição verifica se os Pixeis existem, se sim, somamos o valor do Pixel à variável sum, e incrementamos a variável NPíxel, para no fim dos ciclos facilitar o cálculo da média dos valores dos Pixeis.
Por fim, utilizamos a função ImageSetPixel(), para trocar o valor do Pixel em questão na réplica da imagem Original.
A função retorna uma Imagem, portanto é do tipo Image, tivemos que alterar isso no ficheiro image8bit.h, e no código fornecido anteriormente pelo professor.

Capítulo 3

Conclusão

Este Projeto contribuiu para melhorar os nossos conhecimentos nesta disciplina. Não conseguimos obter resultados, pois ao executar o código "make test1", aparece um /textitwarning na função /textitImageValidPos, sendo que esta função é fornecida pelo professor, decidimos não mexer nesse mesmo código.