Proiect Aplicatii Web cu Suport Java

Converting Color Image to Gray-Scale Image – Weighted method (luminosity method)

Proiect realizat de: Georgescu Alexandru-Iulain

Grupa 333AB

Proiectul contine 2 pachete: packWork si packTest precum in laboratoare.

Pachetul packTest contine aplicatia de test(Main.java) si pachetul packWork continue restul claselor si anume:

* Buffer
* ConvertToGrayscale
* Image
* ImageConsumer
* ReadImageInterface
* ImageProducer
* ReadImage
* WriterResults

Clasa **Image** este o clasa abstracta, care continue numele fisierului .bmp si un obiect image de tip BufferedImage. Este o clasa abstracta deoarece contine o metoda abstracta(readImageQuarter(int quarter, String filename) care va fi implementata in copilul acestei clase ReadImage.

Clasa **ReadImage** mosteneste clasa Image si implementeaza metoda abstracta readImageQuarters, metoda ce are rolul de a citi imaginea pe sfertur-= si de asemenea implementeaza interfata **ReadImageInterface.**

Clasa **ConvertToGrayscale** extinde clasa ReadImage și este folosită pentru a converti o imagine în tonuri de gri. Clasa are mai multe câmpuri, inclusiv o variabilă fileName, un obiect ReadImage, un obiect BufferedImage numit grayscaleImage și un tip de variabilă statică care este setat la BufferedImage.TYPE\_INT\_RGB.

Constructorul ConvertToGrayscale preia un argument șir fileName și inițializează câmpul fileName. Clasa include, de asemenea, un bloc static de cod care setează câmpul de tip la BufferedImage.TYPE\_INT\_RGB.

Metoda reconstructImage(BufferedImage[] image) ia ca argument o matrice de obiecte BufferedImage și folosește clasa Graphics pentru a combina cele patru imagini într-o singură imagine mai mare. Acesta creează un nou obiect BufferedImage cu dimensiunile de 2 ori mai mari decât lățimea și înălțimea imaginilor originale și folosește metoda drawImage() pentru a desena fiecare dintre cele patru imagini în pozițiile corespunzătoare.

Metoda processImage(BufferedImage image) ia un obiect BufferedImage ca argument și parcurge toți pixelii din imagine. Pentru fiecare pixel, extrage valorile roșu, verde și albastru și le folosește pentru a calcula valoarea în tonuri de gri a pixelului folosind metoda luminozității. Apoi setează valoarea în tonuri de gri ca nouă valoare a pixelului din imaginea de intrare.

Include, de asemenea, un cronometru pentru a măsura timpul procesului de conversie.

In clasa **Buffer** regasim o matrice de obiecte BufferedImage pentru a stoca și a prelua date. Matricea, numită imageQuarters, este inițializată cu o dimensiune de 4 în constructor.

Metoda put() este folosită pentru a adăuga un obiect BufferedImage la matrice, iar metoda get() este folosită pentru a prelua un obiect BufferedImage dintr-un index specific din matrice. Ambele metode sunt marcate ca sincronizate, ceea ce înseamnă că un singur fir le poate executa o dată.

In clasa **ImageProducer,** se citesc sferturi de imagine, care sunt apoi transmise catre clasa **ImageConsumer** pentru a fi procesate. Clasa ImageConsumer proceseaza fiecare sfert de imagine si apoi le transmite catre clasa **WriterResults**, care se ocupa de scrierea imaginii finale in fisierul de iesire. Comunicarea dintre **ImageConsumer** si **WriterResults** se realizeaza prin intermediul pype-urilor. Dupa ce sunt primite toate sferturile, se reconstruieste imaginea finala, care va fi scrisa in fisierul de iesire.

Imaginea originala:

O imagine care conține iarbă, exterior, natură, apus

Descriere generată automat

Imaginea procesata:

O imagine care conține iarbă, exterior, cer, câmp

Descriere generată automat