Image Smoothing (convolution mask)

## Introducere

In cadrul acestui proiect am avut de realizat netezirea unei imagini tip 24bit BMP – RGB. Imaginea rezultat se va obtine in urma unei convolutii, iar fiecare pixel de ieșire este modificat de contribuțiile unui număr de pixeli de intrare adiacenți.

Definitia procedurii de Image Smoothing înseamnă că fiecare pixel din imaginea sursă este răspândit și amestecat în pixeli înconjurători. Mai obiectiv, avem o matrice de numere peste imaginea noastra, care se numește convolution kernel.

Pentru fiecare pixel din imagine, luăm numerele corespunzătoare din kernel și pixelii peste care se află, îi înmulțim și adăugăm toate rezultatele împreună pentru a face noul pixel. Am efectuat o estompare 2D făcând două estompări 1D, o dată în direcția orizontală și o dată în direcția verticală.

Acest cod este o implementare a unui filtru de estompare a imaginilor, care utilizează multithreading pentru a accelera procesarea. Clasa Main este punctul de intrare în aplicație, care citeste primul argument din linia de comanda care este calea catre imaginea sursa, apoi citeste din consola raza pentru convolutie pe orizontala, verticala si numarul de iteratii.

Apoi, se creaza un obiect de tipul BlurFilter care primeste ca parametri raza de convolutie pe orizontala, verticala si numarul de iteratii, apoi se aplica filtrul (metoda filter) asupra imaginii sursa, si se scrie imaginea in fisierul specificat ca al doilea argument din linia de comanda.

In final, se calculeaza si se afiseaza timpul de citire, procesare si scriere si timpul total de executie.

In plus, se apeleaza 3 metode diferite din clasa BlurFilter, cu diferite parametrii sau fara parametri.

Modul de apelare : C:\Users\maria\workspace\proiect\bin>java package1.Main otter.bmp out.bmp

## Arhitectura

In proiect, am implementat dupa cum urmeaza:

Un prim pachet numit package1, ce contine modulul principal Main.java, si cel de-al doilea pachet, numit package2 ce contine modulele Math.java, AbstractBufferedImage.java, BlurFilter.java, BlurSubClass.java si Interface.java .

Package1:

* Main.java

Această clasă conține funcția de main în care un obiect de tip BufferedImage va conține imaginea de prelucrat și un obiect File pentru specificarea căii de acces la folderul dorit.

Citim fișierul imaginii sursă în format BMP cu primul argument conținut în linia de comandă care este calea de acces la fișier. După citirea efectuata corect, se calculează și se afișează timpul de execuție. Parametrii sunt introduși de la tastatură de către utilizator, aceștia fiind: raza pentru convoluția pe orizontală, raza pentru convoluția pe verticală și numărul de iterații al metodei de estompare. Se creează un obiect de tip BlurFilter cu , apoi aplicăm metoda filter din cadrul clasei BlurFilter obiectului de tip BufferedImage pentru modificarea imaginii. Apoi, se introduc informațiile în fișierul destinație, a cărui cale de acces este specificată de al doilea argument din linia de comandă. La final, se calculează și se afișează timpul total de execuție al programului.

Package2:

* Math.java

Clasa Math cuprinde două funcții matematice utilizate în clasa BoxBlurFilter și anume o funcție float și una int pentru pozitionarea unui număr o valoare minima si maxima.

* AbstractBufferedImage.java

Această clasă este părintele clasei BlurFilter și este o clasă abstractă, din care clasa BlurFilter utilizează metodele de obținere și setare a valorilor RGB a pixelilor dintr-o imagine. Clasa AbstactBufferedImage este o clasă abstractă și conține metode abstracte( ex: public int comp() {})

* BlurFilter.java

Procesam imaginea prin aplicarea metodei blur de estompare în interiorul metodei filter. Are ca atribute raza pentru convoluția pe orizontală, raza pentru convoluția pe verticală și numărul de iterații al metodei de estompare. In metoda filter obtinem valorile RGB ale pixelilor din imagine, apoi se efectueaza metoda blur si setează din nou valorile RGB obținute ale pixelilor.

* BlurSubClass.java

Clasa AbstactBufferedImage este părintele clasei BlurFilter iar clasa BlurSubClass este clasa copil a clasei BlurFilter, și reutilizează constructorul din clasa părinte.

* Interface.java

Clasa Interface cuprinde trei metode și este interfață pentru clasa BoxBlurFilter, aceasta din urmă utilizând metodele pentru afișarea atributelor prin overriding.

## Concluzii si observatii

Daca introducem in program parametrii pentru raza pe orizontala = raza pe verticala = 5 și numar de iteratii = 1, obținem o imagine blurata, fiind destul de clar totusi modelul din imaginea anterioara. Cu cat crestem valorile razelor si al numarului de iteratii, cu atat aceasta devine si mai netezita, si creste si timpul de executie.

In concluzie, din acest proiect am invatat sa netezesc o imagine, prin liniarizarea acesteia sub forma unei matrici, si modificarea acesteia cu ajutorul a două estompări 1D, o dată pe orizontală și o dată pe verticală.

## Bibliografie

<https://courses.cs.duke.edu/spring06/cps296.1/handouts/Image%20Processing.pdf>

[Convolution Kernel Mask Operation - Java Tutorial | Olympus LS (olympus-lifescience.com)](https://www.olympus-lifescience.com/en/microscope-resource/primer/java/digitalimaging/processing/kernelmaskoperation/)

<https://www.geeksforgeeks.org/image-processing-java-set-1-read-write/>

<https://en.wikipedia.org/wiki/Kernel_(image_processing)>