Procesarea imaginilor

Imbunatatirea calitatii unei imagini

1. Introducere

Tema proiectului este o aplicație care permite încărcarea mai multor imagini, le analizează automat și identifică problemele existente pentru fiecare imagine. Pe baza acestei analize, aplicația aplică îmbunătățirile corespunzătoare. Pentru fiecare imagine se afișează rezultatul Before/After, precum și o secțiune care descrie deciziile luate. Rezultatele pot fi salvate în format color (24 biți) sau binar (8 biți).

2. Obiectivele

- afișarea comparativă a imaginilor înainte și după procesare, pentru o evaluare vizuală clară
- realizarea unui mecanism de analiză automată a imaginilor, care identifică caracteristici relevante privind calitatea acestora
- aplicarea unor operații de îmbunătățire automată a imaginilor, în funcție de rezultatele analizei
- gestionarea procesării imaginilor pe un thread separat, pentru a menține interfața grafică responsive
- afișarea detaliilor și deciziilor de procesare pentru fiecare imagine, într-o zonă dedicată de tip TextArea
- implementarea funcționalităților de salvare a imaginii curente sau a tuturor imaginilor procesate, în diferite formate
- implementarea functionalitatii de zoom pe poze, pentru a vizualiza mai clar diferentele

3. Fundamente teoretice

Aplicația a fost implementată în limbajul Java, utilizând framework-ul JavaFX pentru realizarea interfeței grafice.

Operațiile asupra imaginilor au fost realizate folosind biblioteca standard java.awt.image, precum și tool-ul utilizat în cadrul laboratorului.

Pentru manipularea pixelilor au fost folosite *BufferedImage* (pentru stocarea si manipularea pixelilor in memorie), *Raster* (pentru accesul la datele pixelior), *WritableRaster* (pentru modificarea pixelilor).

4. Solutia propusa

- Aplicația analizează automat calitatea unei imagini și aplică îmbunătățiri în funcție de rezultatele obținute.
- Evaluarea imaginii se bazează pe indicatori clasici din procesarea imaginilor. Se calculează media și deviația standard a luminanței pentru a aprecia luminozitatea și contrastul, iar analiza distribuției intensităților (histograma) ajută la identificarea imaginilor subexpuse sau supraexpuse și la aplicarea corecției de expunere (gamma).
- Zgomotul și claritatea sunt estimate folosind operatorul Laplacian: energia acestuia

indică nivelul de zgomot, iar metoda Variance of Laplacian permite detectarea imaginilor blurate. De asemenea, densitatea muchiilor și proporția pixelilor foarte albi sau foarte negri sunt folosite pentru a diferenția documentele de fotografii obișnuite.

- În funcție de aceste rezultate, aplicația aplică automat filtre și ajustări precum reducerea zgomotului (Gaussian sau median), corecții de luminozitate și contrast, echilibrare de culoare (Gray World), accentuare (sharpen) sau binarizare automată prin metoda Otsu pentru imaginile de tip document.

5. Implementarea solutiei si descrierea tehnica a tuturor functionalitatilor

Aplicația este organizată pe 6 componente principale:

A. MainApp

- Gestionează interfața grafică și fluxul aplicației.
- Încarcă imaginile
- Rulează procesarea în Task<Void> (pentru a nu bloca UI)
- Afișează imaginea înainte/după
- Afișează detaliile procesării

B. AnalizaImagine

Analizează calitatea imaginii (după redimensionare la max. 420px pentru viteză).
Calculează:

- luminozitate și contrast (meanLuma, stdLuma)
- nivel zgomot (noiseScore)
- claritate (blurVariance)
- densitate muchii (edgeDensity)
- proporții pixeli foarte albi/negri/luminoși
- detectează dacă este document sau fotografie

Rezultatele sunt salvate în obs[] și afișate în interfață.

C. ImbunatatireImagine

Aplică automat corecții pe baza analizei:

- Reducere zgomot (Gaussian/ Median)
- Corecție expunere și contrast
- Corecție gamma
- White balance (pentru fotografii)
- Sharpen (pentru fotografii)
- Binarizare Otsu (pentru documente)

Toate deciziile sunt salvate în detalii[] pentru afișare.

D. OperatiiImagine

Conține implementarea efectivă a algoritmilor:

- Conversii (ARGB, grayscale)
- Redimensionare
- Calcul metrici
- Filtre (Gaussian, Median, Sharpen)
- Ajustări luminozitate/contrast
- White balance (Gray-World)
- Binarizare Otsu

Este „motorul” de procesare al aplicației.

E. <u>ConversieImagine</u>

Transformă BufferedImage în WritableImage pentru afișarea în JavaFX.

F. <u>SalvareImagine</u>

Permite salvarea imaginii procesate: PNG sau JPG

6. Prezentarea aplicatiei

- Aplicatia este lansata -> se acceseaza meniul pentru a selecta fisierele/fisierul *(fig. 1)*
- Incarcarea si procesarea fisierelor este facuta pe task *(fig 2)*

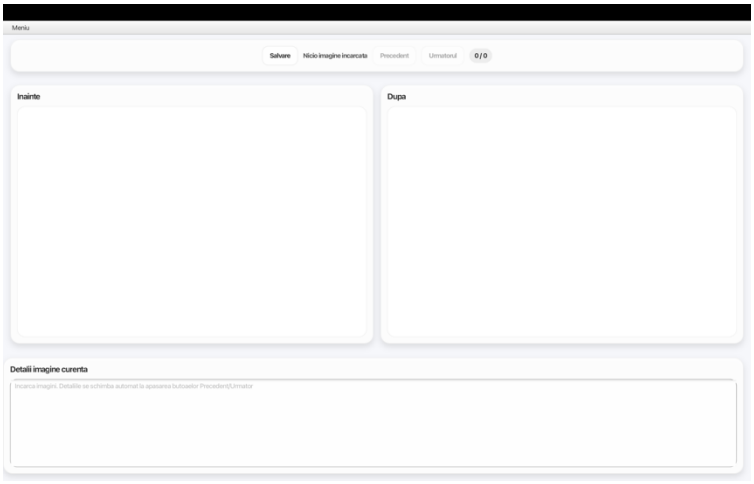

fig. 1

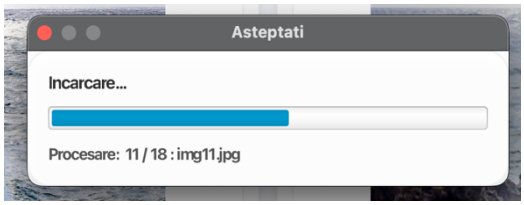

fig. 2

- Se afiseaza imaginile before/after, se parcurg prin intermediul butoanelor din partea de sus a aplicatiei ,iar in partea de jos a aplicatiei sunt scrise detalii despre imaginea curenta (fig. 3) + modificarile aduse (fig. 4)

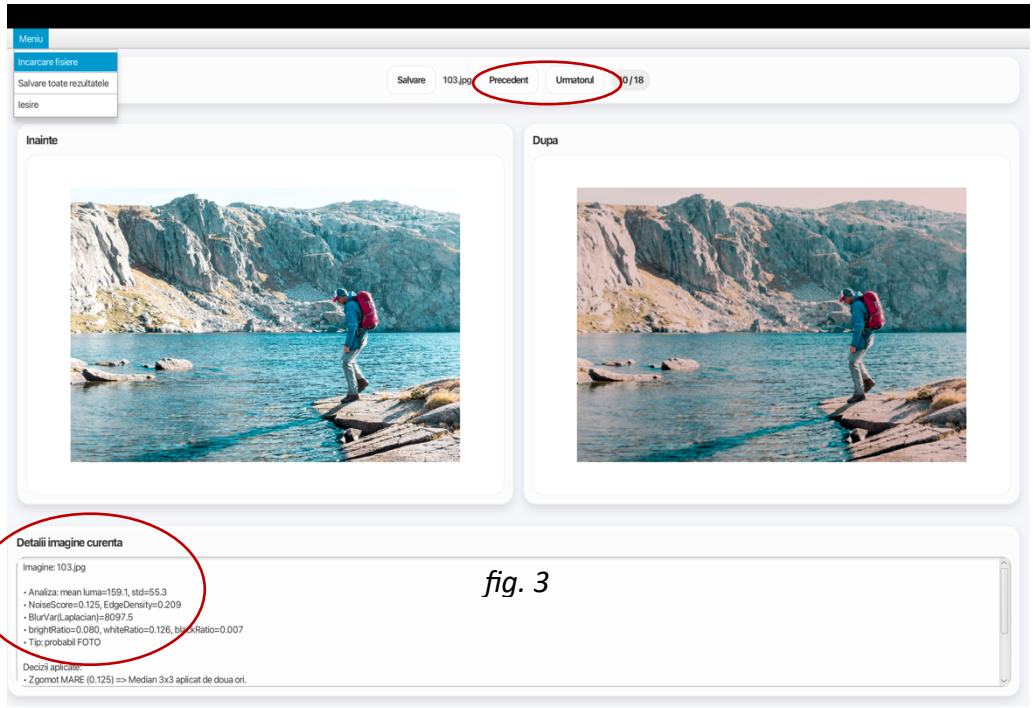

fig. 3

fig. 3

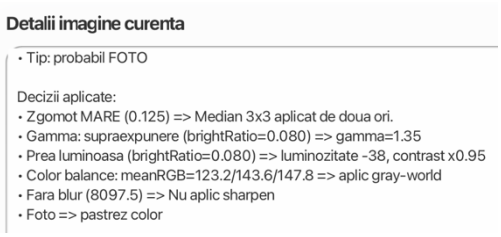

fig. 4

- Pentru a se analiza mai de aproape modificarile aduse, se poate da zoom pe poza folosind scroll ul. (fig 5)

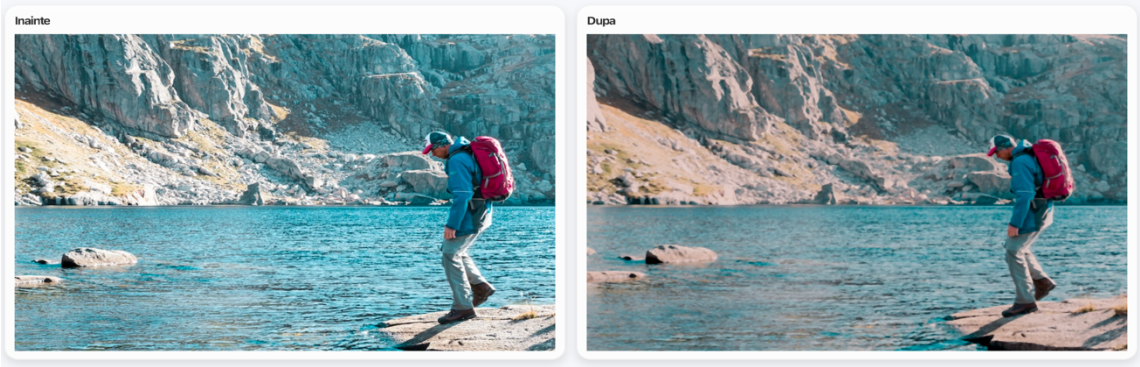

fig. 5

- Se poate salva doar imaginea curenta (se deschide un dialog de selectare a locatiei), si se pot salva toate imaginile modificate intr-un nou folder : se selecteaza/creeaza folderul (fig 6), dupa care se selecteaza tipul imaginilor (fig. 7).

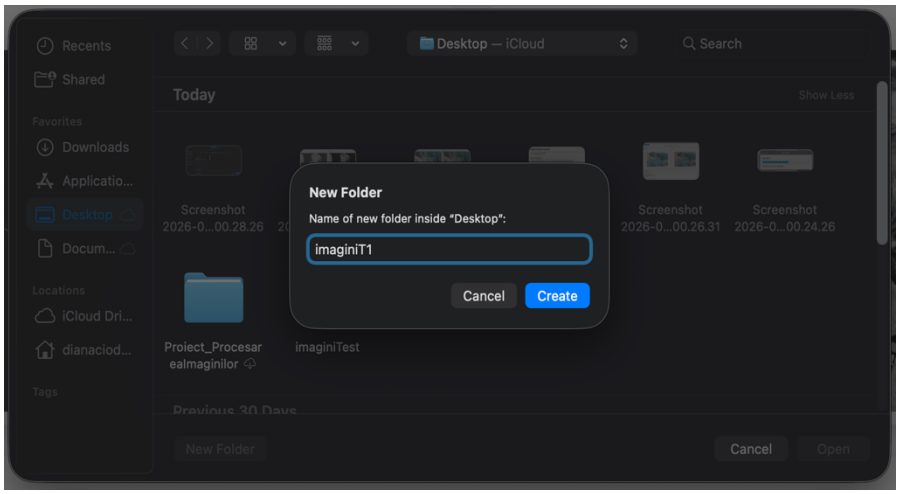

fig. 6

fig. 7

- In figura 8, in folderul “imaginiTest” sunt salvate imaginile inainte editari, iar in folderul “imaginiT1” sunt salvate imaginile dupa editari, diferentele fiind vizibile.

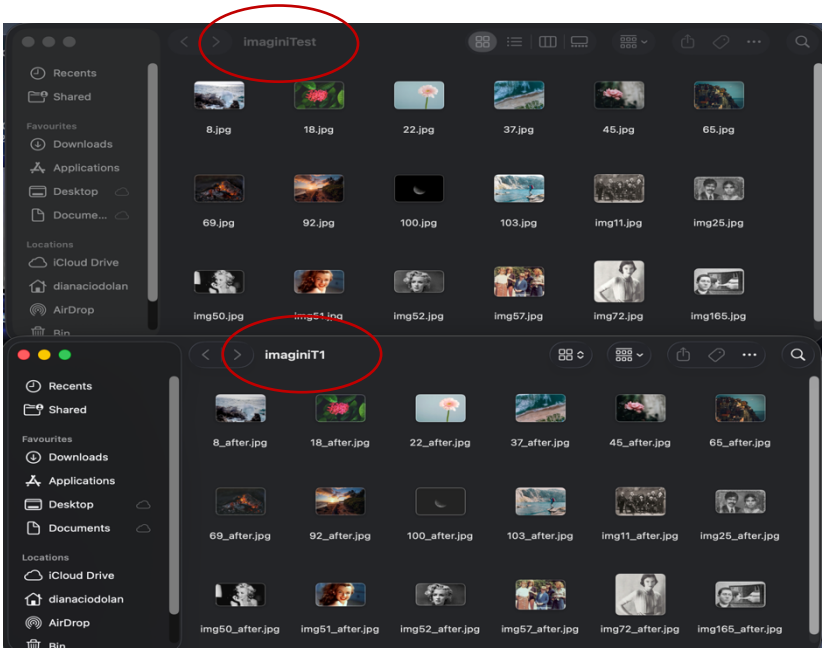

fig. 8

7. Concluzii

Prin realizarea acestui proiect am dezvoltat o aplicație care analizează automat o imagine și aplică corecții pentru a îmbunătăți calitatea acesteia. Pe baza unor indicatori precum luminozitatea, contrastul și nivelul de claritate, programul decide ce operații sunt necesare.

Proiectul m-a ajutat să înțeleg mai bine modul în care imaginile pot fi analizate numeric și cum pot fi aplicate algoritmi pentru îmbunătățirea lor.

Rezultatele obținute arată că procesarea automată poate oferi îmbunătățiri vizibile fără intervenție manuală.

8. Bibliografie

- Conversie bufferedImage to JavaFX image: https://blog.idrsolutions.com/convert-bufferedimage-to-javafx-image/
- Metode de imbunatatirea imaginilor in Java: https://www.geeksforgeeks.org/java/image-processing-in-java-contrast-enhancement/
- Ghid cu metodele de imbunatatire in general, pasi: https://www.pixartprinting.co.uk/blog/improve-photo-quality/?srsltid=AfmBOop-fgY59E4Z_ahnokkajns5wLXUuljliWmRDjOMa9x4VIXXIQpy
- Zoom pe un element de tip ImageView in Java: https://forums.oracle.com/ords/apexds/post/zooming-an-imageview-in-a-scrollpane-9044
- Metoda de salvare a unei imagini in javaFX pe disc: https://stackoverflow.com/questions/74777479/how-to-save-image-in-javafx
- Corectie Gamma - formule: https://www.sciencedirect.com/topics/engineering/gamma-correction