

Proiect AWJ

Edge Extraction (Detection)

Introducere

În acest proiect, am implementat o metodă de detectare a marginilor pentru imagini digitale. Metoda se bazează pe filtrarea imaginii cu un kernel de detectare a marginilor. Proiectul este realizat în IDE Eclipse Mars 2, folosind limbajul de programare Java (versiunea Java 8). Scopul proiectului este de a aplica metoda de Edge Extraction asupra unor imagini. Metoda a fost testată pe o varietate de imagini, inclusiv imagini simple, imagini cu text și imagini cu obiecte complexe. Rezultatele au fost satisfăcătoare, iar metoda a reușit să detecteze cu succes marginile în toate imaginile testate.

Partea teoretică

Metoda Sobel este o tehnică larg utilizată pentru detectarea marginilor în imagini digitale. Este un algoritm de detectare a marginilor bazat pe gradient, ceea ce înseamnă că identifică marginile prin calcularea magnitudinii gradientului intensității imaginii. Gradientul este o măsură a cât de rapid intensitatea imaginii se schimbă de la un pixel la altul.

Cum funcționează metoda Sobel pentru extragerea marginilor?

Metoda Sobel utilizează două kerneluri de 3x3 pentru a calcula gradientele orizontale și verticale ale intensității imaginii. Aceste kerneluri sunt aplicate fiecărui pixel din imagine, iar rezultatele sunt utilizate pentru a calcula magnitudinea și direcția gradientului la acel pixel.

Magnitude: Magnitudinea gradientului la un pixel este o măsură a intensității generale a marginii. Se calculează luând rădăcina pătrată a sumei pătratelor componentelor gradientului orizontal și vertical.

Pașii metodei Sobel:

- Conversia imaginii în tonuri de gri.

- Aplicarea kernelurilor Sobel la imagine.
- Aplicarea unui prag imaginii magnitudinii gradientului.
- Dilatarea și eroziunea imaginii marginilor.

Outputul algoritmului de detectare a marginilor Sobel este o imagine a marginilor, în care marginile sunt reprezentate de pixeli albi. Marginile din imagine pot fi utilizate pentru diverse sarcini, cum ar fi detectarea obiectelor, segmentarea imaginilor și extragerea caracteristicilor.

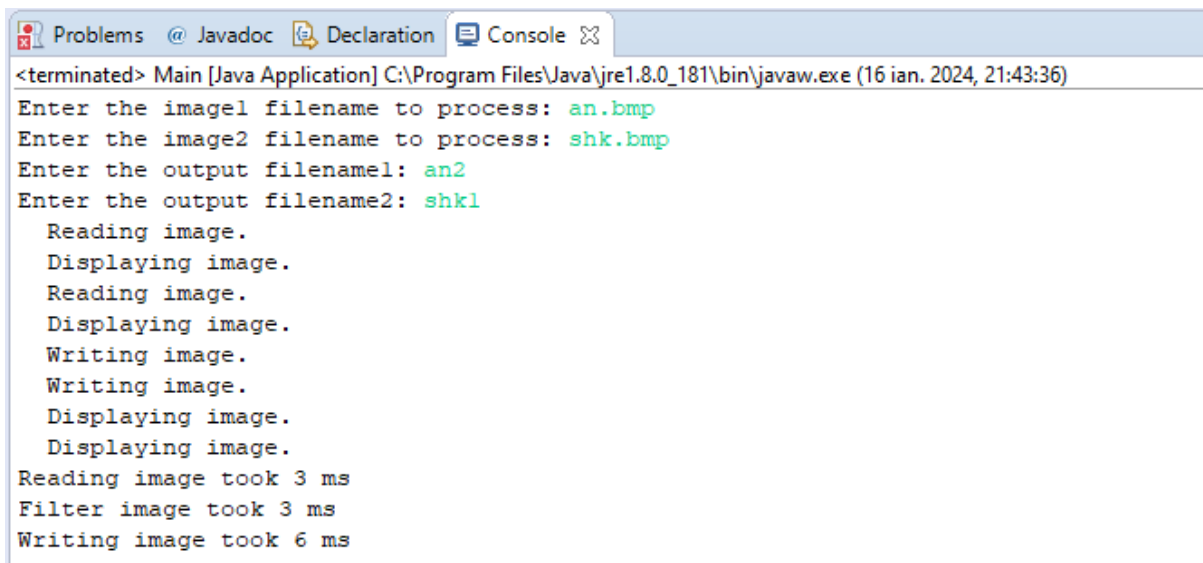
Descrierea Functionala a Aplicatiei

Algoritmul de detectare a marginilor

Acest algoritm detectează marginile dintr-o imagine folosind o tehnică numită gradientul de intensitate. Acest algoritm funcționează prin calcularea gradientului de intensitate pentru fiecare pixel din imagine. Gradientul de intensitate este o măsură a schimbării de intensitate a culorii de la un pixel la altul. Pentru a calcula gradientul de intensitate, algoritmul folosește două filtre: un filtru vertical și un filtru orizontal. Filtrele sunt matrici de numere care sunt folosite pentru a înmulți pixelii din imagine cu valori specifice.

Magnitudinea gradientului este o măsură a intensității generale a marginii. Este calculată folosind următoarea formulă:

$$|G| = \sqrt{G_x^2 + G_y^2}$$



```
Problems @ Javadoc Declaration Console
<terminated> Main [Java Application] C:\Program Files\Java\jre1.8.0_181\bin\javaw.exe (16 ian. 2024, 21:43:36)
Enter the image1 filename to process: an.bmp
Enter the image2 filename to process: shk.bmp
Enter the output filename1: an2
Enter the output filename2: shk1
  Reading image.
  Displaying image.
  Reading image.
  Displaying image.
  Writing image.
  Writing image.
  Displaying image.
  Displaying image.
Reading image took 3 ms
Filter image took 3 ms
Writing image took 6 ms
```



Descrierea Arhitecturala a Aplicatiei

Aplicatia este format din 3 foldere: images, packTest, packWork

In folderul images se afla imaginile date spre a fi procesate

In pachetul packTest este clasa Main, iar in pachetul packWork se afla urmatoarele clase: DisplayImg, ImageFilter, ReadImg, Time, WriteImg.

Main.java reprezinta principala clasa din care se face posibila citirea de la tastatura a numelor fisierelor(atat cele de input, cat si cele de output). Sunt apelate si folosite functiile de: citire/read, scriere/write, afisare/display, *detectEdges + cei 3 timpi de executie;*

DisplayImg.java – este clasa care implementeaza interfata Time si este clasa in care se afiseaza imaginea.

ImageFilter.java – reprezinta clasa care extinde clasa ReadImg si totodata clasa in care se foloseste metoda de procesare a imaginii prin Edge Detection

Time.java- aceasta interfata defineste metode pentru calcularea timpului necesar pentru diferite operatiuni în fluxul de prelucrare a imaginilor.

ReadImg.java – reprezinta clasa care da extend la DisplayImg si foloseste metoda de citire a imaginii.

WriteImg.java reprezinta clasa care da extend la ImageFilter si foloseste metoda de scriere a imaginii

Descrierea modulelor

import java.awt.image.BufferedImage - Importă clasa BufferedImage din pachetul java.awt.image. Această clasă reprezintă o imagine care poate fi manipulată de biblioteca de grafică AWT Java.

import java.io.File - Importă clasa File din pachetul java.io. Această clasă reprezintă un fișier care poate fi citit sau scris.

import java.io.IOException - Importă clasa IOException din pachetul java.io. Această clasă reprezintă o excepție care poate fi aruncată atunci când există o eroare la citirea sau scrierea unui fișier.

import javax.imageio.ImageIO - Importă clasa ImageIO din pachetul javax.imageio. Această clasă oferă metode pentru citirea, scrierea și manipularea imaginilor.

import javax.swing.ImageIcon - Importă clasa ImageIcon din pachetul javax.swing. Această clasă creează o pictogramă dintr-o imagine.

import javax.swing.JFrame - Importă clasa JFrame din pachetul javax.swing. Această clasă reprezintă o fereastră într-o aplicație Java Swing.

import javax.swing.JLabel - Importă clasa JLabel din pachetul javax.swing. Această clasă afișează un text sau o imagine.

import javax.swing.WindowConstants - Importă clasa WindowConstants din pachetul javax.swing. Această clasă oferă constante pentru proprietățile ferestrei, cum ar fi dacă o fereastră este redimensionabilă sau nu.

import java.awt.BorderLayout - Importă clasa BorderLayout din pachetul java.awt. Această clasă definește un manager de aspect care împarte un container în cinci regiuni: nord, sud, est, vest și centru.

import java.util.Scanner - Importă clasa Scanner din pachetul java.util. Această clasă oferă un mod de a citi intrare dintr-o sursă, cum ar fi un fișier sau consola.

import packWork.ImageFilter - Importă clasa ImageFilter din pachetul packWork. Această clasă conține algoritmul pentru detectarea marginilor într-o imagine.

import packWork.ReadImg - Importă clasa ReadImg din pachetul packWork. Această clasă citește un fișier imagine și îl stochează într-un obiect ***BufferedImage***

import packWork.WriteImg - Importă clasa WriteImg din pachetul packWork. Această clasă salvează un obiect BufferedImage într-un fișier imagine.

Performante

Performantele codului si a algoritmului de procesare al imaginii folosit masoara diferiti timpi in functie de imaginile pe care le introduc.



13. Decrease color depth Gray-Scale Image
14. Rotate Image (90, 180, 270)
15. Translate Image (X – Horizontal, Y – Vertical – prescribed by user)
16. Edge Extraction (Detection)
17. Converting Color Image to Gray-Scale Image – Weighted method (luminosity method)
18. Image resizing (Zooming +/-) – keeping aspect ratio. Pixel replication method
19. Image resizing (Zooming +/-) – keeping aspect ratio. Zero order hold method
20. Image resizing (Zooming +/-) – keeping aspect ratio. Zooming K times method
21. Image Brightness modification
22. Image Contrast modification
23. Gray Level Histogram Sliding (+/-)
24. Gray Level Histogram Stretching (+/-)
25. Linear Gray Level Transform
26. Logarithmic Gray Level Transform
27. Power-Law Gray Level Transform
28. Laplacian Operator (Positive/Negative)

Timpii de executare pentru cele 2 imagini de mai sus:

```
Reading image took 2 ms  
Filter image took 2 ms  
Writing image took 8 ms
```

Concluzie

Metoda implementată în acest proiect este o metodă eficientă și precisă de detectare a marginilor pentru imagini digitale. Metoda poate fi utilizată în diverse aplicații, cum ar fi recunoașterea imaginilor, segmentarea imaginilor și analiza imaginilor.

Bibliografie

1. https://en.wikipedia.org/wiki/Edge_detection
2. <https://www.analyticsvidhya.com/blog/2022/08/comprehensive-guide-to-edge-detection-algorithms/>
3. <https://vincmazet.github.io/bip/detection/edges.html>
4. <https://www.mygreatlearning.com/blog/introduction-to-edge-detection/#methods-of-edge-detection>
5. <https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm>
6. https://www.researchgate.net/profile/Djemel-Ziou/publication/312890367_'Edge_detection_techniques_An_overview'/links/59d3c61c0f7e9b4fd7ffbfb/Edge-detection-techniques-An-overview.pdf
7. https://d1wqtxts1xzle7.cloudfront.net/31159717/Maini-libre.pdf?1392250617=&response-content-disposition=inline%3B+filename%3DStudy_and_comparison_of_various_image_ed.pdf&Expires=1705433271&Signature=elhttpLPETxUmLTiexf~BTj5cx7j3Rx9ixaD3OzEjG8aEDVI3RXEgJ1o-5j~iYT9u1lnz5GrAAXOL8G-Z5eU~G4QHp4YDW-~K7WXVUKQkwXbAx3EoE-E4N5V799CUt~W6zoiRczKVpQuFnNR16yUGClooTC-5pp29gU5xVT6khzgkOpS7CSBK6liiL2NrYI012GXH9XTzRDL0fhsgUG09BHo~I4c2C-86qtg7j-dq6sl71pkbSQzWTFWAAAtOfzM-9HoOO5O7Wc4APzryH5qPSpKA2-83M-cdmTJQLP-4laTI9wYfBpoCCs9tIXhFN9ZPymxAzpsetuMcgGVfOdMiw_&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA