Scanned Pages

Bodea Adina Mariana1, Vasiu Andrei 2\*, Rosca David-Sorin 3

1 Documenter, Technical University of Cluj-Napoca, Strada Memorandumului 28, Cluj-Napoca, Romania

2 Scientist, Technical University of Cluj-Napoca, Strada Memorandumului 28, Cluj-Napoca, Romania

3 Programmer, Technical University of Cluj-Napoca, Strada Memorandumului 28, Cluj-Napoca, Romania

2\*Rosca.Se.David@student.utcluj.ro

Sinteza lucării: Lucrarea are la baza analizarea unei imagini ce contine o pagina dintr-o carte pozata in mod eronat si transpunerea ei in format pdf cu rectificarea erorilor atat de claritate cat si de incadrare prin delimitarea marginilor si cresterea contrastului asupra scrisului pentru o lizibilitate mult mai buna asupra continutului. Aplicatia are la baza transformata FOURIER pentru detectarea pixelilor de culoare neagra pe un fundal de culoare alba. Transformata se bazeaza pe detectarea amanuntita a esantioanelor de culoare neagra si crearea anvelopei semnalului pentru o detectie mai buna a spectrelor de culoare inchisa si transpunerea lor pe un fundal alb. Cu ajutorul limbajului de programare python am reusit sa cream o interfata grafica ce reuseste pe baza fotografiilor incarcate sa faca corectia pe imaginea inserata, urmand o previzualizare a inaginii noi create si convertirea ei in pdf pentru o mai buna calitate si ulitizare mai prietenoasa cu utilizator.

1. Rezumat într-o limbă de circulație internațională

The work is based on analyzing an image containing a page from a book, mistakenly photographed, and transposing it into a PDF format with the correction of both clarity and framing errors by cropping the margins and increasing the contrast for better legibility of the content.

The application relies on the Fourier transform to detect black ink strokes on a white background. The transform is based on detailed detection of black color samples and creating a signal envelope for better detection of dark color spectrum and transposing them onto a white background.

Using the Python programming language, we managed to create a graphical interface that, based on uploaded photos, corrects the inserted image, provides a preview of the newly created image and converts it into a PDF for improved quality and user-friendly utilization.

We have found that in order to filter and contrast the image we have to do some pre-processation of the image(Converting to gray scale, apply canny edge detector,find all the contours in the image,get the biggest corner points, wrap perspective transform, adaptive thresholding).

1. Stadiul actual (max 10%)

In stadiul actual pe piata sunt destul de multe aplicatii care fac conversia de la jpg/png la pdf cu tot cu ajustare a imaginii insa de cele mai multe ori este manuala in functie de preferinta utilizatorului. Un astfel de exemplu este Camscanner o aplicatie ce face acest lucru, ea lucreaza pe baza vectoriala si iti face automat detectia colturilor insa ai posibilitatea de a modifica aceste colturi in functie de nevoia utilizatorului. Un alt plus pe care aplicatia il are este acela de a modifica claritatea si dimensiunea pdf-ului si de a-l crea in functie de spatiul de stocare pe care utilizatorul doreste sa il ocupe insa asta modifica calitatea imaginii si a scrisului modificandu-se numarul de pixeli utilizat pentru aceeasi rezolutie.

Alte aplicatii asemanatoare sunt: Adobe Scan, Microsoft Office Lens, Scanner Pro, Office Lens by OneNote, Genius Scan, Scanbot, Tiny Scanner.

1. Fundamentare teoretică (max 30%)

Pentru a realiza scanner-ul a necesitat un algoritm pentru detectia marginilor si pentru aceasta am utilizat “Canny Edge Detection”. Cum imaginea este susceptibila la zgomot primul pas este eliminarea lui cu ajutorul unui filtru gausian 5X5. Imaginea netezită este apoi filtrată cu un nucleu Sobel atât în ​​direcția orizontală, cât și în cea verticală pentru a obține prima derivată în direcția orizontală (Gx) și în direcția verticală (Gy). Din aceste două imagini, putem găsi gradientul marginii și direcția pentru fiecare pixel. (site1)

Edge\_Gradient (G) = (3.1)

Angle () = tan-1  (3.2)

Direcția gradientului este întotdeauna perpendiculară pe margini si este rotunjit la unul dintre cele patru unghiuri reprezentând direcțiile verticale, orizontale și două diagonale. (site2)

In urma obtinerii marimii si directiei gradientului, se realizeaza o verificare complete a imaginii pentru a elimina orice pixel nedorit care ar putea sa nu constituie marginea. Pentru acest lucru, se verifica la fiecare pixel in vecinatatea sa daca este maxim local in directia gradientului.

C A B Gradient direction

Figura 3.1 Directia Gradientului

Punctul A este pe margine (în direcție verticală). Direcția gradientului este normală față de margine. Punctele B și C sunt în direcții de gradient. Deci punctul A este verificat cu punctele B și C pentru a vedea dacă formează un maxim local. Dacă da, este luat în considerare pentru următoarea etapă, în caz contrar, este suprimat (pus la zero). (site3)

Această etapă decide care sunt toate marginile sunt cu adevărat margini și care nu sunt. Pentru aceasta, avem nevoie de două valori de prag, minVal și maxVal. Orice muchii cu gradient de intensitate mai mare decât maxVal sunt cu siguranță margini, iar cele sub minVal sunt sigur non-margini, deci ignorate. Cei care se află între aceste două praguri sunt margini clasificate sau non-margini în funcție de conectivitatea lor. Dacă sunt conectați la pixeli „cu margini sigure”, ei sunt considerați a fi parte a marginilor. În caz contrar, sunt de asemenea ignorate.

A maxVal

C

B minVal

Figura 3.2 Margini, minVal-non-margini si maxVal margini

Muchia A este deasupra maxValului, deci considerată drept „margine sigură”. Deși muchia C este sub maxVal, este conectată la muchia A, astfel încât aceasta este considerată și ca muchie validă și obținem acea curbă completă. Dar muchia B, deși este deasupra minVal și este în aceeași regiune cu cea a muchiei C, nu este conectată la nicio „margine sigură”, așa că este aruncată. Deci, este foarte important să selectăm minVal și maxVal corespunzător pentru a obține rezultatul corect. [Doxygen]

OpenCV integrează toate aceste operații într-o singură funcție, respectiv cv.Canny().Această funcție acceptă o imagine de intrare și două valori, minVal și maxVal, care definesc intervalul de gradient. De asemenea, funcția utilizează un parametru numit aperture\_size pentru dimensiunea nucleului Sobel, implicit setat la 3. Parametrul final, denumit L2gradient, determină ecuația folosită pentru calculul mărimii gradientului. Dacă acesta este setat la True, se utilizează o ecuație mai precisă; în caz contrar, se alege o altă metodă pentru calcul:

Edge\_Gradiet (G) = |Gx| + |Gy| (Care este setat False implicit) (3.3)

## Operatii morfologice [site 4]

* Pe, scurt un set de operatii care proceseaza imaginile dupa forma acestora
* Cele mai cunoscute operatii morfologice sunt dilatarea si erodarea :
  + Reducerea zgomotului
  + Gasirea denivelarilor in imagine sau a unor “gauri” in aceasta
* Vom folosii o imagine pentru explicarea acestor doua operatii morfologice:

A white letter on a black background

Description automatically generated A white letter on a black background

Description automatically generated A white letter on a black background

Description automatically generated

### Figura 3.4 Dilatarea, Eroziunea

### Dilatarea a)

* Formula matematica a dilatarii dupa convolutie: dst(x,y)=max(x′,y′):element(x′,y′)≠0src(x+x′,y+y′)
* Pixelii albi din zona de fundal negru s-au “marit” cu ajutorul acestei operatii.

### Eroziunea b)

* Formula matematica a erodarii este : dst(x,y)=min(x′,y′):element(x′,y′)≠0src(x+x′,y+y′)
* Pixelii albi de pe fundalul negru s-au “micsorat” ceea ce ne duce la o concluzie clara ca dilatarea si erziunea sunt doua operatii inverse.

1. Implementarea soluției adoptate (30%-40%)

Am ales sa folosim strategia mentionata si mai sus.Vom avea in vedere dezvoltarea urmatorilor in ordinea cronologica propusa si gandita in cadrul echipei:

1. Convertirea pdf-ului de intrare in imagini(deoarece un pdf poate avea mai multe pagini) de tip .png/PNG [Site5]
2. Pre-procesarea imaginii care este realizata prin reducerea de zgomot. [Site6]
3. Detectia de margini/colturi pentru a crea noul centru de interes (ROI)
4. Gasirea conturului cel mai semnificativ din imagine
5. Crearea noii imagini care va avea colturile imaginii rezultate , colturile centrului de interes, eliminand astfel zonele de fundal care nu sunt relevante
6. Marirea contrastului pentru o vizualizare mult mai clara a textului final.
7. Converitirea imaginilor corectate intr-un pdf
8. Rezultate experimentale (min 10%)

Am observant ca rezultatele experimentale variaza de la o poza la alta si ca luminantele pixelilor dintr-o imagine nu pot fi prelucrate in acelasi fel pentru orice tip de poza.

Rezultate intermediare:

A paper with writing on it

Description automatically generatedA paper with writing on it

Description automatically generatedA black and white page

Description automatically generated

Figura 5.1 A-Imagine Originala, B-Gray, C-Adaptive threshold

A black and white photo of a piece of paper

Description automatically generatedA close-up of a paper

Description automatically generatedA paper with writing on it

Description automatically generatedA paper with writing on it

Description automatically generated

Figura 5.2 A-Adaptive threshold2, B-blur, C-Warp perpective, D-Gray warp perpective

An open book on a table

Description automatically generatedA page of a book

Description automatically generatedA close up of a book

Description automatically generatedA black and white image of a book

Description automatically generated

Figura 5.3 A-Original, B-Gray, C-Adaptive threshold, D-Adaptive threshold

Concluzii

Prin urmare aplicatia pe care noi am realizat-o va fi de folos in convertirea, respectiv adaptarea anumitor fisiere pdf astfel incat sa rezulte un fisier pdf cu contrast imbunatatit si bine centrat, punand in evidenta continutul cel mai semnificativ.

1. Bibliografie

13.1 Websites

[Site1] ‘OpenCV modules’, https://docs.opencv.org/4.x/, accessed November 9, 2021

[OpenCV1] OpenCV modules- Step-by-Step Lessons:

<https://docs.opencv.org/4.x/>

[Site2] ‘Contours in OpenCV’, https://docs.opencv.org/4.x/d3/d05/tutorial\_py\_table\_of\_contents\_contours.html, January 26, 2022

[Site3] ‘Histograms in OpenCV’, https://docs.opencv.org/4.x/de/db2/tutorial\_py\_table\_of\_contents\_histograms.html, May 17, 2022

[Site4] ‘Miscellaneous operating system interfaces, https://docs.python.org/3/library/os.html, November 1, 2012

[Site5] ‘User Interfaces for HumansTM, https://www.pysimplegui.org/en/latest/, August 6, 2022

[Site6] ‘PyMuPDF, https://pymupdf.readthedocs.io/en/latest/, March 24, 2018

13.2 Book, book chapter and manual

[Doxygen] Doxygen: ' Tesseract documentation ', in Various documents related to Tesseract OCR, May 14, 2015, pp 2-3

[Doxygen] Doxygen: 'Tesseract User Manual' in Various documents related to Tesseract OCR, April 13, 2020,pp 2-3

*13.3 Journal articles*

[Kenton O’Hara] University of Bristol: “Opportunities and barriers to portable document scanning”, pp 3-4

*13.4 Report*

[invicti] Detailed Scan Report,[Publies 2022], pp 1-6

*13.5 Patent*

[BIM] Scan-to-BIM Related Technology, [Publies 2020]