UNIVERSITATEA ALEXANDRU IOAN CUZA IAŞI
FACULTATEA DE INFORMATICĂ

LUCRARE DE LICENŢĂ

Recunoaștere de semnături

propusă de
Cosmin Salca

Sesiunea: *Iulie, 2017*

Coordonator ştiinţific
Lect. Univ. Dr. Anca Ignat

DECLARAŢIE PRIVIND ORIGINALITATE ŞI RESPECTAREA DREPTURILOR DE AUTOR

Prin prezenta declar că Lucrarea de licenţă cu titlul „*Recunoaștere de semnături*" este scrisă de mine şi nu a mai fost prezentată niciodată la o altă facultate sau instituţie de învăţământ superior din ţară sau străinătate. De asemenea, declar că toate sursele utilizate, inclusiv cele preluate de pe Internet, sunt indicate în lucrare, cu respectarea regulilor de evitare a plagiatului:

- toate fragmentele de text reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi deţin referinţa precisă a sursei;
- reformularea în cuvinte proprii a textelor scrise de către alţi autori deţine referinţa precisă;
- codul sursă, imagini etc. preluate din proiecte *open-source* sau alte surse sunt utilizate cu respectarea drepturilor de autor şi deţin referinţe precise;
- rezumarea ideilor altor autori precizează referinţa precisă la textul original.

Iaşi, *03.07.2017*

Absolvent *Cosmin Salca*

_____

(semnătura în original)

DECLARAŢIE DE CONSIMŢĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licenţă cu titlul „*Recunoaștere de semnături*”, codul sursă al programelor şi celelalte conţinuturi (grafice, multimedia, date de test etc.) care însoţesc această lucrare să fie utilizate în cadrul Facultăţii de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea Alexandru Ioan Cuza Iaşi să utilizeze, modifice, reproducă şi să distribuie în scopuri necomerciale programele-calculator, format executabil şi sursă, realizate de mine în cadrul prezentei lucrări de licenţă.

Iaşi, *03.07.2017*

Absolvent *Cosmin Salca*

_____

(semnătura în original)

Acord privind proprietatea dreptului de autor

Facultatea de Informatică este de acord ca drepturile de autor asupra programele-calculator, format executabil şi sursă, să aparţină autorului prezentei lucrări, *Cosmin Salca.*

Încheierea acestui acord este necesară din următoarele motive:

[Se explică de ce este necesar un acord, se descriu originile resurselor utilizate în realizarea produsului-program (personal, tehnologii, fonduri) şi aportul adus de fiecare resursă.]

Iaşi, *03.07.2017*

Decan *Adrian Iftene*

_____

(semnătura în original)

Absolvent *Cosmin Salca*

_____

(semnătura în original)

Cuprins

Introducere

Tehnologiile biometrice sunt foarte folosite în ziua de astăzi ca o metodă de securitate pentru aplicații. Aceste sisteme doresc să ajungă să recunoască o persoană după trăsăturile ei fiziologice sau comportamentale. Recunoașterea trăsăturilor fiziologice se face pe baza măsurării trăsăturilor biologice, precum amprenta, irisul sau fața, sau a celor comportamentale, precum vocea sau a semnătura.

Semnătura este un tip de trăsătură comportamentală foarte importantă din cauză că este folosită în multe domenii, precum cel financiar, administrativ sau legal. În domeniul recunoașterii semnăturii s-au făcut multe cercetări în ultimele decenii. Aceasta poate să fie online sau offline. Sistemele online folosesc informațiile semnăturii dinamic, verificând semnătura când aceasta a fost facută. Cele offline verifică imaginile scanate ale semnăturilor.

Această lucrare vorbește despre recunoașterea offline a unei semnături și scopul ei este de a prezenta noțiunile teoretice despre recunoașterea unei semnături și algortimii implementați, metodele de îmbunătățire a unei imagini pentru a fi mai ușor de manipulat și statisticile pentru rezultatele obținute.

Contribuțiile personale în realizarea acestei lucrări constau în studierea unui domeniu care nu face parte din programa studiată în anii de facultate și implementarea unei aplicații care are rolul de a recunoaște o semnătură.

În Capitolul 1, intitulat „Noțiuni introductive, se prezintă informații de bază care au fost folosite pe întreaga lucrare. Se definește imaginea și imaginea digitală, prezentându-se diferite tipuri de imagini digitale.

În Capitolul 2, intitulat „Îmbun ătățire”, se prezintă modul în care se procesează o imagine ca aceasta să fie mai clară și cu care se poate lucra mai ușor. Se arată mai multe metode de filtrare a unei imagini și diferite detectetoare de colțuri sau muchii. Cele mai importante detectoare, care au fost folosite și in aplicația din această lucrare, sunt cele ale lui Sobel și Harris, dar se prezintă și detectoare înrudite cu cele două, precum Moravec, Roberts sau Prewitt.

În Capitolul 3, intitulat „Lucrări în domeniu”, se prezintă ce metode au mai fost folosite pentru recunoașterea unei semnături. Prezintă provocări care au fost întâlnite, seturi de date folosite, metode de preprocesare și extragere de caracteristici.

În Capitolul 4, intitular „Descrierea aplicației”, se prezintă cum aplicația din această a fost realizată, pașii pe care îi urmează pentru a ajunge la rezultat și tabele cu statistici pe rezultate obținute.

1. Noțiuni introductive

1.1 Definiție imagine și imagine digitală

O imagine este o proiecție a unei scene 3D într-un plan 2D. Imaginea este reprezentată ca o funcție bidimensională f(x, y), unde x și y sunt coordonatele orizontale și verticale, iar valoarea funcției în poziția (x, y) reprezintă intensitatea luminoasă în acel punct.[1]

O imagine digitală este o reprezentare a unei imagini bidimensionale ca un set finit de valori digitale, denumiți pixeli. Un pixel este cea mai mică componentă a unei imaginidigitale.[2]

1.2 Tipuri de imagini digitale

1. Imagini binare

Imaginea binară este cel mai simplu tip de imagine și poate lua doar două valori, alb și negru sau 0 și 1. Imaginea binarș folosește doar un singur bit ca să reprezinte un pixel. Aceste tipuri de imagini sunt folosite în aplicațiile unde se folosește doar conturul sau forma lucrurilor din imagine.

2. Imagini grayscale

Imaginea grayscale este o imagine monocromă care conține doar nivele de gri. Numărul de biți folosiți pentru un pixel determină numărul de nivele de gri disponibile. O imagine tipică grayscale conține 8 biți per pixel și permite 256 de nivele diferite de gri.

3. Imagini color

Imaginile color sunt, de obicei, reprezentate cu ajutorul culorilor roșu, verde și albastru (imagini RGB). Fiecare pixel are câte 8 biți pentru fiecare culoare dintre cele trei menționate, deci câte 24 de biți pentru fiecare pixel, iar fiecare culoare poate să aibă valori cuprinse între 0 și 255.

4.Imagini multispectrale

Imaginile multispectrale conțin informații care nu se pot vedea cu ochiul uman. Printre acestea se numără cele infraroșu, ultraviolete sau cele cu raze X. Sunt imagini diferite de cele obișnuite deoarece nu sunt vizibile pentru sistemul vizual uman, dar informațiile din ele pot fi convertite în imagini color, formatul RGB.[3]

2. Îmbunătățire

2.1 Rolul îmbunătățirii

Scopul îmbunătățirii imaginilor este de a procesa imaginea pentru a îi fi văzute și evaluate informațiile vizuale pe care le conține cu mai multă claritate. Îmbunătățirea imaginilor este subiectivă pentru că depinde foarte tare de informația specifică pe care utilizatorul speră să o obțină din imagine.

Condițiile principale pentru îmbunătățirea imaginilor este ca informația care se dorește a fi extrasă trebuie să existe în imagine, informația să nu fie acoperită complet de zgomotul din imagine, iar imaginea procesată trebuie să fie mai potrivită decât cea originală pentru sarcina propusă.

2.1.1 Îmbunătățire prin filtrarea imaginilor

Printre cele mai populare tehnici de îmbunătățire se numără eliminarea zgomotului și accentuarea contururilor imaginilor, dar există și alte operațiuni care pot fi realizate prin procesul de filtrare în domeniul spațial. Filtrarea în domeniul spațial indică faptul că procesul de filtrare are loc direct pe pixelii din imagine. Filtrările acționează asupra unei imgini pentru a schimba valorile pixelilor într-un mod specific și sunt clasificate în două tipuri: liniare și neliniar.

Indiferent de filtrarea folosită, toate abordările de filtrare în domeniul spațial acționează în același fel. Fiecare pixel din imagine – un pixel la un moment ales se numește pixelul țintă- este accesat succesiv. Valoarea pixelului țintă este apoi înlocuită de o nouă valoare care depinde numai de valoarea pixelilor aflați în vecinătatea acestuia.

2.2 Vecinătățile pixelului

O proprietate importantă în procesarea de imagini este conceptul de conectivitate. Multe operațiuni din procesarea de imagini folosesc conceptul unei vecinătăți pentru a defini o zonă de interes. În centrul acestei teme în definirea vecinătății se află noțiunea de conectivitate a pixelilor, unde se alege conexiunea fiecărui pixel cu ceilalți. În 4-conectivitate, numai pixelii care se află in nordul, vestul, estul sau sudul pixelului sunt conectați cu el. Dacă se iau in considerare și pixelii de pe diagonale, atunci există 8-conectivitate.

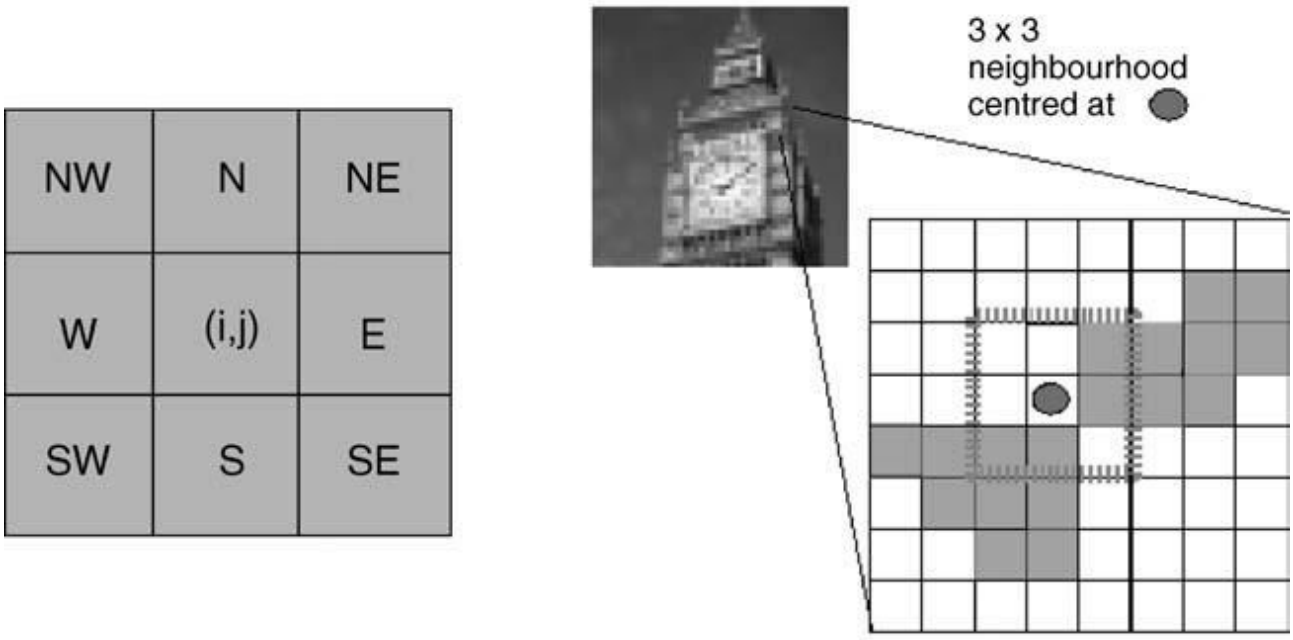

Figura 1. Conectivitate (stânga) și exemplu de vecinătate 3x3 la o locație specifică a unui pixel

În Figura 1 (drepta) se verifică daca regiunile A și B sunt conectate și se folosește un model de conectivitate (NxN = 3x3) pentru a determina dacă au aceeași caracteristici. Operațiile, precum filtrarea și detecția de muchii, calculează noi valori ale pixelilor în funcție de o vecinătate prestabilită și niște ponderi aplicate pixelilor definiți de vecinătatea aleasă.Dimensiunea vecinătății poate fi controlată prin modificarea parametrului N după care este calculată deplasarea k. În general, vecinătatea poate fi și NxM, unde N este diferit de M, dar cea mai folosită este NxN. Majoritatea tehnicilor de procesare de imagini folosesc acum 8-conectivitate, care, pentru o dimensiune rezonabilă a vecinătății, este realizată în timp real de procesoarele moderne pentru majoritatea operațiilor. Operațiile de filtrare asupra întregii suprafețe a unei imagini folosesc un principiu în care fiecare pixel din imagine este procesat pe baza unei operții efectuată vecinătății sale NxN.

2.3 Filtrare liniară și moduri de aplicare

În filtrările liniare spațiale valoarea nouă sau filtrată a pixelului țintă este determinată ca o combinație liniară a valorilor pixelilor din vecinătatea sa. Orice alt tip de filtrare este, prin definiție, una neliniară. Combinația liniară a pixelilor vecini este determinată de către nucleul de filtrare (denumit și mască). Acesta este o subimagine care are aceeași dimensiune ca vecinătatea care conține ponderele ce urmează să fie asignate tuturor pixelilor corespunzători din vecinătatea pixelului țintă. Filtrarea continuă prin poziționarea cu succes a nucleului, astel încât locația pixelului central coincide cu locația fiecărui pixel țintă de fiecare dată când valoarea filtrată este

calculată de către combinația ponderată aleasă. Această procedură de filtrare poate fi astfel vizualizată ca glisarea nucleului deasupra tuturor locațiilor de interes din imaginea originală (i, j), multiplicând pixelii de sub nucleu cu ponderea w, calculând noua valoare însumând totalul și copiindu-le în aceeași locație în noua imagine filtrată.

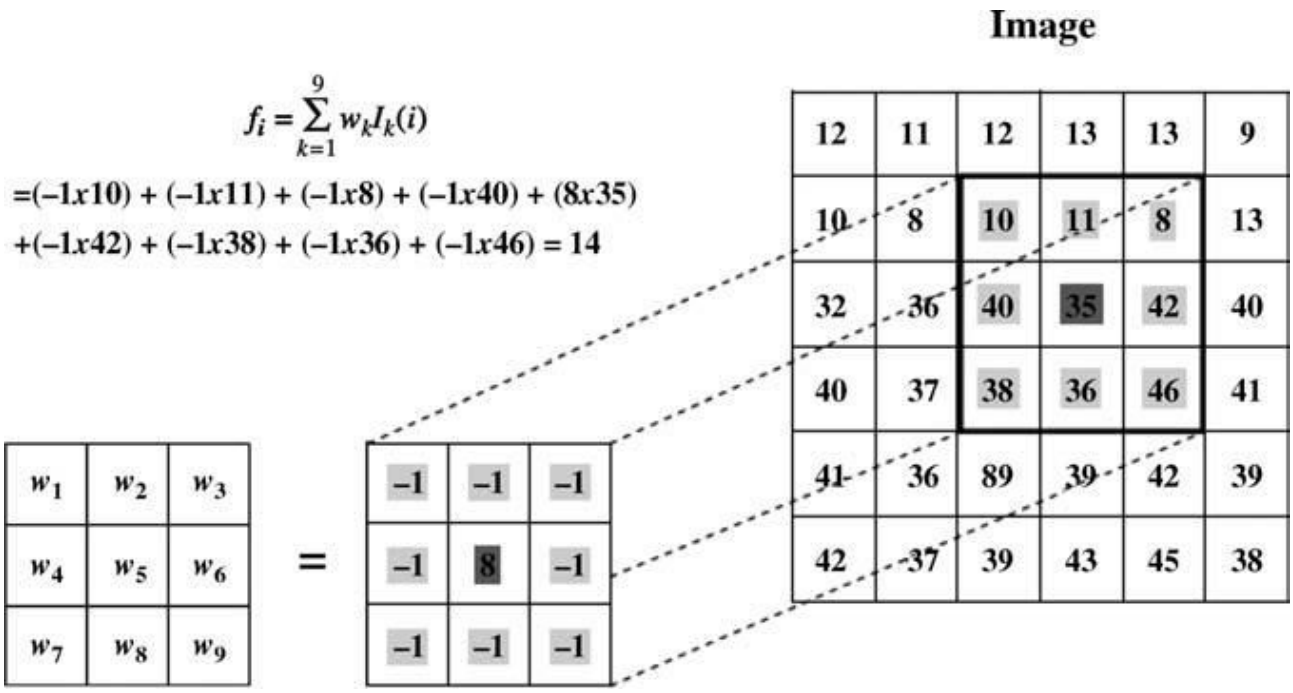

Figura 2. Mecanica filtrării liniare cu o filtrare de nucleu, unde NxN = 3x3

Modul de aplicare a filtrelor spațiale liniare exprimă, de fapt, în formă discretă un proces denumit convoluție. Din această cauză, multe filtrări de nucleu sunt descrise ca nuclee de convoluție, se întelege implicit că sunt aplicate imaginii în modul liniar descris mai sus. Formal, se poate exprima acțiunea de convoluție dintre un nucleu și o imagine in două moduri. Prima se adresează indicilor rândului și coloanei imaginii și nucleului:

$$f(x,y) = \sum_{i=I_{min}}^{I_{max}} \sum_{j=J_{min}}^{J_{max}} w\,(i,j) I(x+i, y+j)$$

Aici, indicii i=0, j=0 corespund pixelului central al nucleului care are o dimensiune $(I_{max} - I_{min} + 1, J_{max} - J_{min} + 1)$.A doua abordare este echivalentă și folosește indici liniari:

$$f_i = \sum_{k=1}^{N} w_k\, I_k(i)$$

În acest caz, $I_k(i)$reprezintă pixelii vecini ai pixelului pe poziția i, unde k este un index liniar care rulează peste regiunea vecină orizontal sau vertical. $w_k$este valoarea nucleului și

$f_i$reprezintă valoarea filtrată rezultată din valoarea originală de intrare $I_k(i)$. Figura 2 ilustrează procedura de bază, unde pixelul central al nucleului și cel țintă din imagine au un fundal gri închis. Nucleul este plasat în imagine în așa fel încât pixelii centrali și cei țintă să se potrivească. Valoarea filtrată a pixelului țintă $f_i$este dată de către combinația liniară a valorilor pixelilor vecini, ponderele specifice fiind determinate de către valorile nucleelor w_k. În acest caz, valoarea pixelului țintă de 35 este filtrată către o valoare de ieșire 14.

Filtrarea liniară (convoluția) poate fi rezumată în urmatorii pași:

(1) Definirea nucleului de filtrare.

(2) Glisarea nucleului deasupra imaginii astfel încât pixelii centrali să coincidă cu fiecare pixel țintă din imagine.

(3) Înmulțirea pixelilor de sub nucleu cu valorile ponderelor din nucleul de deasupra și sumarea totalului.

(4) Copierea valorii rezultate în aceeași locație în o nouă imagine.

Aplicarea operației de filtrare pentru pixelii aflați în aproprierea marginilor imaginii poate crea probleme, deoarece nucleul de convoluție are elemente plasate în afara imaginii. În general, sunt trei abordări principale pentru rezolvarea acestei situații:

(1) Se lasă neschimbați acei pixeli țintă care ar crea probleme în aplicarea nucluelui de filtrare.

(2) Se aplică filtrarea doar asupra acelor pixeli care se află în interiorul imaginii.

(3) Completarea pixelilor lipsă din operațiunea de filtrare prin duplicarea valorilor din afara imaginii.

Cele mai folosite sunt metodele (2) și (3). În anumite situații, este acceptat ca imaginea să fie tăiată, asta înseamnă că se extrage doar partea din imagine în care toți pixelii din aproprierea frontierei imaginii care nu au fost filtrați bine sunt îndepărtați complet.

2.4 Filtrarea pentru detecția muchiilor

O muchie poate fi considerată o discontinuitate în valorile de intensitate ale pixelilor din interiorul imaginii. De aceea, filtrările care folosesc derivatele imaginii sunt importante pentru detecția de muchii în procesarea de imagini.

2.4.1 Filtrări de derivare pentru discontinuități

Filtrările normale sumeaza pixelii din o vecinătate, iar acest lucrupoate să afecteze prin netezirea sau pierderea calității imaginii. De fapt, este o integrare în formă discretă. Pe de altă parte, filtrările care utilizează operația de derivare pot fi folosite pentru detecția de discontinuități

dintr-o imagine și joacă un rol important în îmbunătățirea constrastului unei imagini. Filtrările derivate sunt făcute să răspundă la punctele de discontinuitate dintr-o imagine și să le ignore în zomele netede, adică detectează muchiile.

Unul dintre cele mai importante aspecte în sistemul vizual uman este modul în care se folosește de contururile sau muchiile obiectelor pentru recunoașterea și percepția distanței și orientării. Această caracteristică a condus la o teorie pentru sistemul vizual uman bazată pe ideea conform căreia cortexul vizual conține un complex de detectoare de caracteristici care sunt îndreptate către muchiile și segmentele de diferite lățimi și orientări. Caracteristicile muchiilor, prin urmare, pot juca un rol important în analiza imaginilor.

Detecția de muchii este, în principiu, o metodă de segmentare a imaginii în regiuni bazată pe discontinuități. Discontinuitatea îi permite utilizatorului să observe caracteristicile dintr-o imagine unde este mai mult sau mai puțin o schimbare în nivelul de gri sau textură, indicând sfârșitul unei regiuni în imagine și începutul alteia. Îmbunătățirea acestor discontinuități din imagine permite îmbunătățirea calității imaginii sub anumite condiții. Cu toate acestea, ca și alte metode de analiză a imaginilor, detecția de muchii este sensibilă la zgomot.

Detecția de muchii se folosește de operatorii diferențiali ca să detecteze schimbările de gradient ale nivelelor de gri sau culoare din imagine. Această detecție este divizată în două categorii principale: detecția muchiilor de gradul întâi și detecția muchiilor de gradul doi. Detecția de gradul întâi se bazează pe folosirea derivatelelorde ordinul întâi ale imaginii, iar cea de gradul doi se bazează pe folosirea derivatelor de oridnul doi. Tabelul 1 conține definițiile continue și aproximările discrete pentru o imagine 2-D f(x, y)ale acestor derivate.

Derivată 2 D	Caz Continuu	Caz discret
$\frac{\partial f}{\partial x}$	$\lim_{\Delta x \to 0} \frac{f(x+\Delta x, y) - f(x,y)}{\Delta x}$	$f(x+1, y) - f(x, y)$
$\frac{\partial f}{\partial y}$	$\lim_{\Delta y \to 0} \frac{f(x, y+\Delta y) - f(x,y)}{\Delta y}$	$f(x, y+1) - f(x, y)$
$\nabla f(x, y)$	$\left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$	$[f(x+1, y) f(x, y), f(x, y+1) f(x, y)]$
$\frac{\partial^2 f}{\partial x^2}$	$\lim_{\Delta x \to 0} \frac{(\partial f / \partial x)(x+\Delta x, y) - (\partial f / \partial x) f(x, y)}{\Delta x}$	$f(x+1, y) - 2f(x, y) + f(x-1, y)$

$\frac{\partial^2 f}{\partial y^2}$	$\lim_{\Delta y \to 0} \frac{(\partial f/\partial x)(x, y+\Delta y) - (\partial f/\partial x) f(x,y)}{\Delta y}$	$f(x, y+1) - 2f(x,y) + f(x, y-1)$
$\nabla^2 f(x,y)$	$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$	$f(x+1, y) + f(x-1, y) - 4f(x,y) + f(x, y+1) + f(x, y-1)$

Tabelul 1. Operatori derivați: definițiile continue și aproximarile discrete

Deși sunt ușor de implementat, reprezentările discrete din Tabelul 1 nu sunt, în general, alese în practică. Aceasta este din cauză că detecția de muchii, de obicei, este precedată de operația de suprimare a zgomotului care se realizează prin aplicarea unui filtru de netezire, de obicei un filtru Gaussian. Se aplică filtrele de netezire (de eliminare a zgomotului) pentru că filtrele de derivare îl accentuează, ceea ce face mai dificilă găsirea muchiilor.

2.4.2 Detecția de muchii de gradul întâi

Mai multe nuclee de filtrare au fost propuse pentru aproximarea primei derivate a gradientului imaginii. Trei dintre cele mai cunoscute (numele lor fiind luate de la autorii acestora) sunt prezentate în Figura 3, unde se află detectoarele de muchii ale lui Roberts, Prewitt și Sobel. Toate trei sunt implementate ca o combinație de doua nuclee: unul pentru derivata x și unul pentru derivata y.

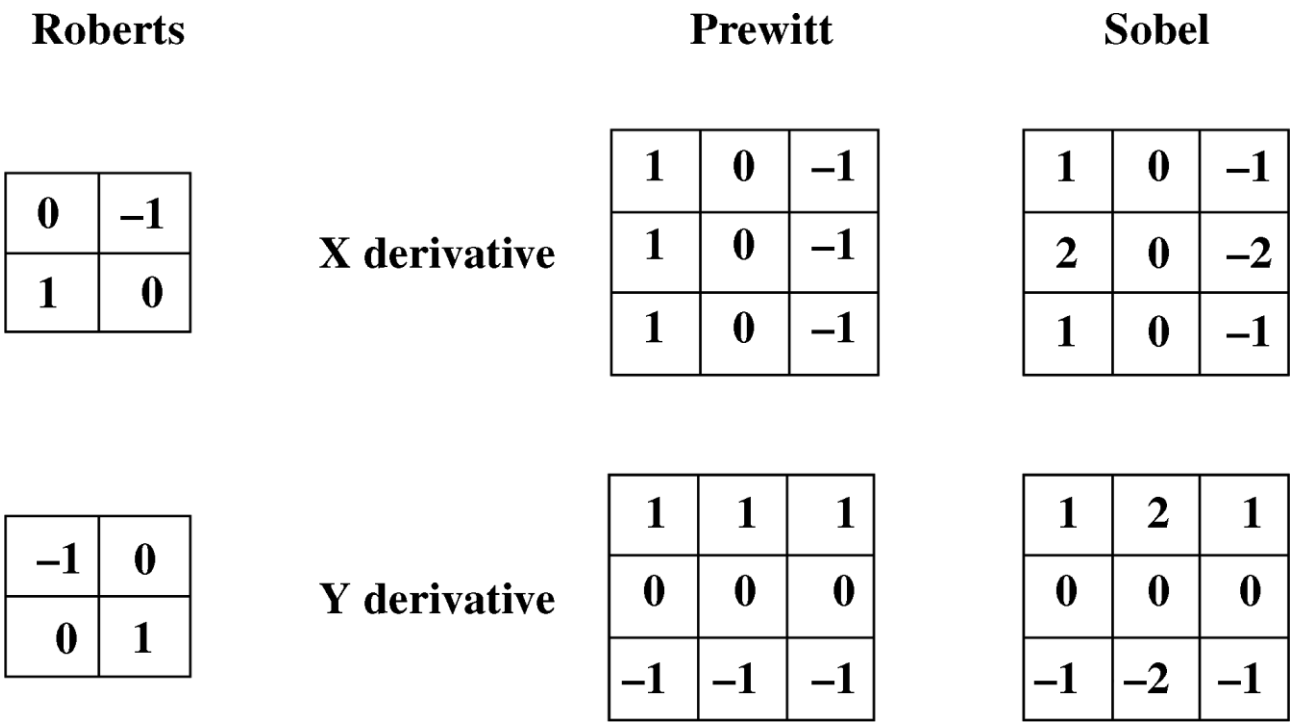

Figura 3. Detectoare de muchii de gradul întâi

Operatorii simplii 2x2 ale lui Roberts sunt printre primele metode cu rolul de a detecta muchii. Operatorul lui Robers calculează o aproximarea gradientului spațial 2-D, simplă și eficientă, asupra regiunilor puse în evidență pe o imagine. Acesta este implementat utilizând două nuclee de convoluție, fiecare proiectat să răspundă la muchii care au înclinația de maxim$\pm 45^{\circ}$, nuclee care returnează derivatele x și y ale imaginii, $G_x şi G_y$. Magnitudinea $|G|$și orientarea θ ale gradientului imaginii sunt date de formulele:

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

$$\theta = arctan\left(\frac{G_y}{G_x}\right) + \frac{\pi}{4}$$

Operatorul lui Roberts calculează repede (datorită dimensiunii mici a nucleelor) imaginea filtrată, dar este foarte sensibil la zgomot. Detectoarele Prewitt și Sobel depășesc aceste limitări, dar folosesc nuclee de convoluție mai complexe (Figura 3 în centru și dreapta).

Nucleele Prewitt sau Sobel sunt mai folosite decât cel a lui Roberts pentru că gradientul nu este mutat cu jumătate de pixel în ambele direcții, iar extinderea către dimensiuni mai mari (pentru vecinătăți mai mari de 3x3) nu este posibilă cu operatorul Roberts. Cea mai importantă diferență dintre operatorii Sobel și Prewitt este că nucleul Sobel implementează diferențierea

într-o direcție și media Gaussian în cealaltă. Avantajul acestuia este că netezește regiunea muchiei, reducând șansele ca pixelii zgomotoși sau izolați să acopere răspunsul filtrat.

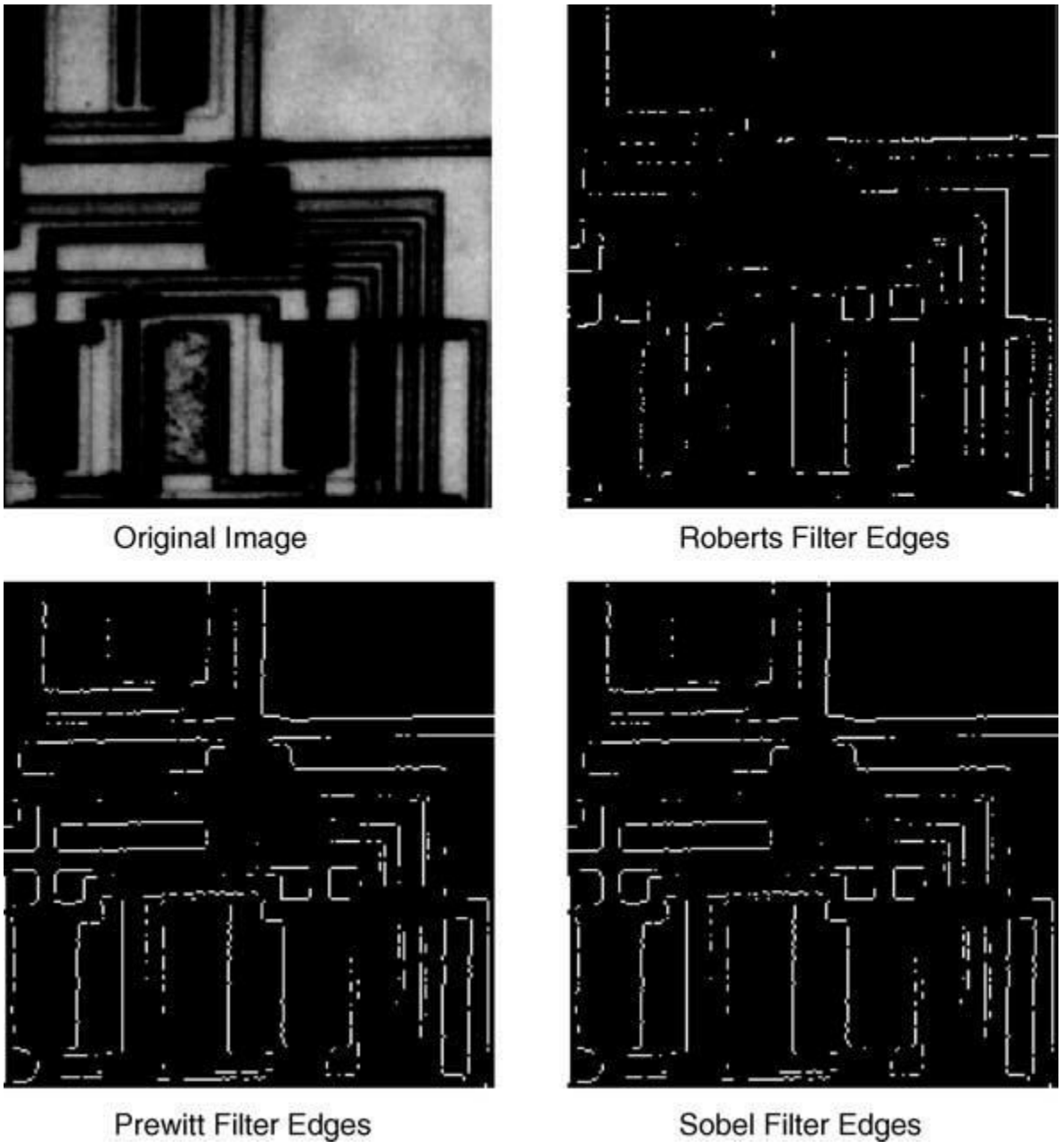

Figura 4. Răspunsul la magnitudine ale lui Roberts, Prewitt și Sobel

2.4.2.1 Filtrare liniară separabilă

Filtrările Sobel și Prewitt sunt exemple de filtrări liniare separabile. Asta înseamnă că nucleul de filtrare poate fi exprimat ca produsul de matrice dintre un vector coloană și un vector linie. Prin urmare, nucleele de filtrare din Figura 3 pot fi scrise:

$$\begin{bmatrix}10-1\\10-1\\10-1\end{bmatrix} = \begin{bmatrix}1\\1\\1\end{bmatrix}[10-1] \text{și} \begin{bmatrix}10-1\\20-2\\10-1\end{bmatrix} = \begin{bmatrix}1\\2\\1\end{bmatrix}[10-1]$$

O consecință importante este că procesul de filtrare 2-D poate fi realizat de două operații de filtrare 1-D secvențiale. Prin urmare, rândurile imaginii sunt prima dată filtrate cu o filtrare de linie 1-D și imaginea rezultată este apoi filtrată pe verticală cu o filtrare de coloane 1-D. Acest lucru ajută la reducerea numărului de operații aritmetice necesare pentru o convoluție. Economia de operații aritmetice este modestă pentru cazul 3x3 (o reducere la șase înmulțiri în comparație cu nouă pentru versiunea 2-D), dar este mai mare pentru cazul nucleelor cu dimensiuni mai mari. În general, filtrările liniare separabile au o complexitate de calcul de ordinul 2N, iar pentru cele care nu sunt separabile, sunt de ordinul N^2.

2.4.3 Detecția de muchii de gradul doi

2.4.3.1 Detectorul de muchii Laplacian

Un operator de derivate de ordinul doi foarte popular este cel a lui Laplacian:

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^{2f}}{\partial y^2}$$

Forma discretă este luată din Tabelul 1 ca:

$$\nabla^2 f = f(x+1,y) + f(x-1,y) - 4f(x,y) + f(x,y+1) + f(x,y-1)$$

Acesta poate fi implementată cu ușurință pe o filtrare de nucleu 3x3, cum se poate observa în Figura 5 A. Dacă se analizează o imagine când este aplicat local acest operator, se așteaptă ca răspunsul să fie mai mare (Laplacian fiind un operator de derivare de gradul doi) la acele puncte din imagine unde gradientul local se schimbă cel mai repede. Una dintre potențialele dezavantaje pentru aplicarea măștii în forma din Figura 5 A, este lipsa de răspuns la caracteristici aflate în direcțiile diagonale față de axele imaginii. Dacă axele sunt rotite cu 45° și este suprapus Laplacianul rotit pe original, atunci se poate construi o filtrare care este neschimbată la multiple rotații de 45° (Figura 5 B).

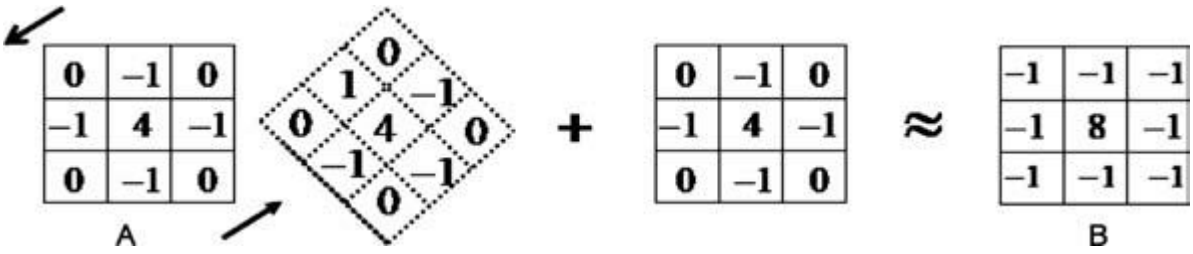

Figura 5. Construcția nucleului discret Laplacian

Figura 6 compară răspunsul dintre filtrările lui Sobel și Laplacian. Se observă cum operatorul Sobel tinde să producă „muchii groase”, iar Laplacianul tinde să producă muchii mai fine. Cu toate acestea, deoarece nucleele Laplacianului aproximează a doua derivată a imaginii, acestea sunt foarte sensibile la zgomot.

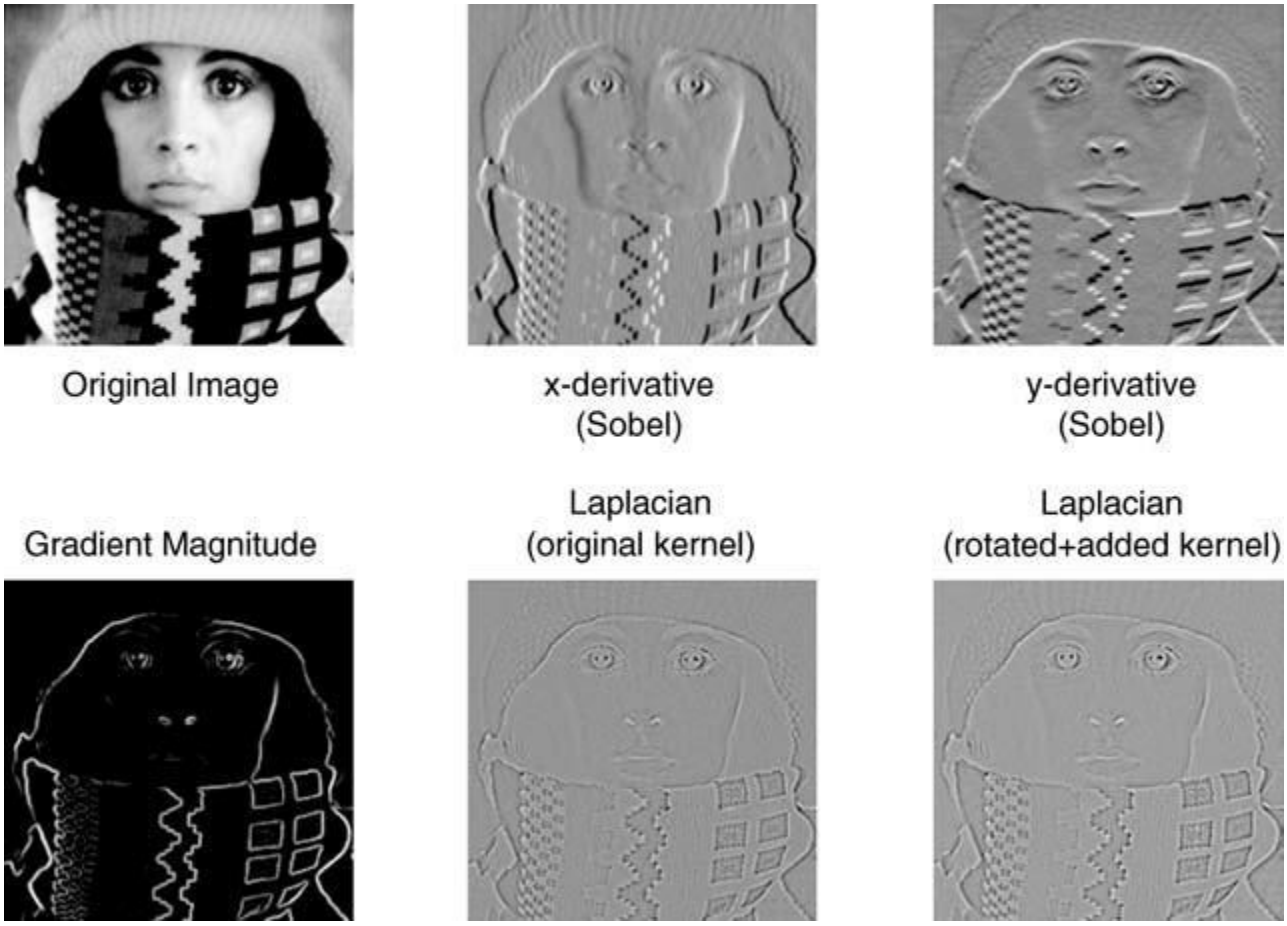

Figura 6. Comparație între filtrările derivate de gradul întâi (Sobel) și cele de gradul doi (Laplacian)

2.4.3.2 „Laplacian of Gaussian”

Pentru a combate această sensibilitate la zgomot a filtrării folosind operatorul Laplacian, nucleul Laplacian standard (Figura 5) este, în mod obișnuit, combinat cu nucleul Gaussian pentru a produce o metodă de filtrare mai robustă. Aceste două nuclee pot fi aplicate una după alta imaginii ca două operații de convoluție separate – prima dată se netezește cu nucleul Gaussian, apoi se accentuează muchiile cu cel Laplacian. Cu toate acestea, cum convoluția este asociativă, putem combina operatorul de netezire Gaussian cu operatorul Laplacian ca să obținem un singur nucleu: filtrarea „Laplacian of Gaussian” (LoG). Acest nucleu este aplicat imaginii într-un singur

pas. Aceasta oferă o reducere semnificativă de calcule, aproximativ jumătate din calculul necesar.

Răspunsul fitrării va fi zero în zone cu intensitate uniformă, în timp ce în zonele de tranziție va fi diferit de zero. Pentru o muchie operatorul o să returneze un răspuns pozitiv pentru partea închisă și unul negativ pentru partea mai dechisă.

2.4.3.3 Detectorul „zero-crossing”

Se folosește detectorul „zero-crossing” pentru a localiza pixelii a căror valoare a Laplacianului trece prin zero, adică punctele unde Laplacianul își schimbă valoarea. Aceasta se întâmplă la muchiile din imagine unde intensitatea se schimbă rapid (sau în zone unde intensitatea se schimbă din cauza zgomotului). Se poate spune despre detectorul „zero-crossing” că este mai mult un detector de caracteristici decât unul specific de muchii. Rezultatul detectorului „zero-crossing” este, de obicei, o imagine binară cu linii groase cât un pixel care arată pozițiile punctelor „zero-crossing”.

Punctul de start pentru detectorul „zero-crossing” este o imagine care a fost filtrată folosind filtrarea LoG (pentru a combate efectul de zgomot). Dimensiunea Gaussianului folosit pentru etapa de netezire influențează foarte mult rezultatele filtrării „zero-crossing”. Cu cât netezirea crește (dimensiunea mare a filtrului de netezire), cu atât sunt din ce în ce mai puține contururi „zero-crossing”.

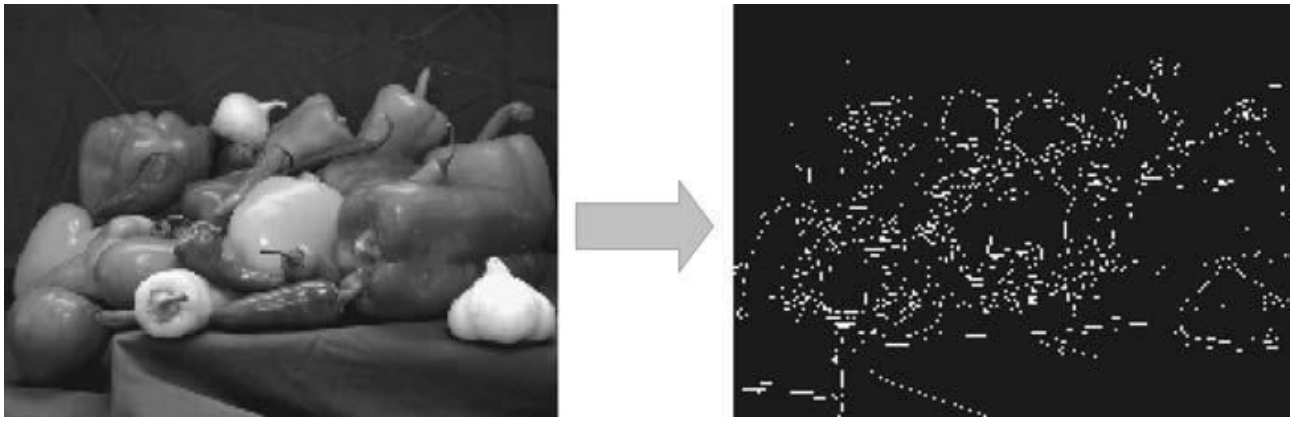

Figura 7. Detecție de muchii cu „zero-crossing”

Filtrarea LoG este predispusă la zgomot dacă deviația standard a netezirii Gaussiene este mică. O soluție la această problemă este ca aceasta să fie mărită pentru a păstra doar muchiile mai proeminente. O alternativă ar fi ca să se păstreze „zero-crossing” doar dacă gradientul

acesteia este peste un anumit prag. Aceasta va face să se păstreze doar muchiile mai proeminente, dar derivata a treia este și ea sensibilă la zgomot.

Noțiunile legate de îmbunatățirea imaginilor au fost adaptate după cartea [4]

2.5 Detecție de colțuri

Detecția de colțuri este o tehnică de procesare de imagini utilizată în sistemele de Computer Vision pentru a extrage anumite caracteristici din conținutul unei imagini.

Un colț poate fi definit ca o intersecție de două muchii. Un colț mai poate fi definit ca un punct pentru care există două direcții dominante și diferite într-o vecinătate locală a punctului.

Un punct de interes este un punct într-o imagine care are o poziție bine definită și poate fi detectat ușor. Asta înseamnă că un punct de interes poate fi un colț.

Detectoarele de colțuri nu sunt, de obicei, foarte robuste și de multe ori necesită introducerea de multe informații redundante pentru a preveni efectul erorilor individuale. Ceea ce determină calitatea unui detector de colțuri este abilitatea de a detecta același colț in mai multe imagini diferite, unde diferă condițiile de lumină, rotație și alte transformări.

O metodă de detectare a colțurilor în imagini a fost propusă de către Harris, care este o îmbunatațire a metodei lui Moravec.

2.5.1 Detectorul de colțuri Moravec

Detectorul de colțuri Moravec analizează variația locală din jurul unui punct prin compararea zonelor imaginilor locale și calculând autocorelarea locală care nu este normalizată între acestea. Pentru fiecare pixel compară o zonă centrată pe acel pixel cu opt zone locale care sunt mutate puțin (câte un pixel pentru fiecare dintre cele opt direcții posibile) față de zona curentă. Aceste zone sunt comparatefolosind formula sumei diferențelor pătrate și se păstrează minimul dintre aceste opt valori. Această valoare minimă caracterizează posibilitatea de a fi colț pentru pixel.

$$V_{u,v}(i,j) = \sum_{\forall a,b \in Window} \big(f(i+u+a, j+v+b) - f(i+a, j+b)\big)^2$$

unde $(u, v) \in \{(-1,-1), (-1,0), (-1,1), (0,-1), (0,1), (1,-1), (1,0), (1,1)\}$și Window este, de obicei, 3x3, 5x5 sau 7x7.

Figura 8. Două imagini simple (stânga) împreună cu harta colțurilor Moravec (centru-stânga), colțurile detectate de Moravec (centru-dreapta) și colțurile detectate de Harris (drepta)

Detectorul de colțuri Moravec are două probleme care au dus la dezvoltarea de alte detectoare:

- Răspuns anizotrop[1]. În semnul rutier din Figura 8 se observă că detectorul de colțuri Moravec pune mai mult accent pe liniile diagonale, iar mai puțin pe cele verticale și orizontale. Prin urmare, răspunsul operatorului nu este izotropic[2]. Răspunsul anizotrop poate fi redus prin netezirea imaginiiînainte aplicăriidetectorului de colțuri.
- Răspunsul la zgomot. Detectorul Moravec este și destul de sensibil la zgomot. Răspunsul la zgomot poate fi diminuat prin folosirea unei suprafețe mai mari sau prin aplicarea unui filtru de netezire înainte de aplicarea detectorului de colțuri.

2.5.2 Detectorul de colțuri Harris

Detectorul de colțuri Harris diferetă de cel a lui Moravec prin modul în care determină caracteristica de a fi colț a unui pixel. În loc de suma diferențelor pătrate, se folosește de derivatele parțiale, o funcție pondertă Gaussian și valorile proprii ale unei reprezentări matriciale a ecuației.

Variația de intensitate (suma de diferețe pătrate) a zonei unei imagini W pentru un schimb mic (Δi, Δj) este:

$$SSD_W(\Delta i, \Delta j) = \sum_{(i,j)\in W}\big(f(i,j) - f(i-\Delta i, j-\Delta j)\big)^2$$

Al doilea termen poate fi aproximat ca:

[1]Anizotrop = care nu are aceeași proprietăți în toate direcțiile
[2]Izotrop = care are aceeași proprietăți în toate direcțiile

$$f(i-\Delta i, j-\Delta j) \approx f(i,j) + \begin{bmatrix} \frac{\delta f(i,j)}{\delta i} & \frac{\delta f(i,j)}{\delta j} \end{bmatrix} \begin{bmatrix} \Delta i \\ \Delta j \end{bmatrix}$$

Prin urmare, ecuația poate fi rescrisă astfel:

$$SSD_W(\Delta i, \Delta j) = \sum_{(i,j)\in W} \left(f(i,j) - f(i,j) - \begin{bmatrix} \frac{\delta f(i,j)}{\delta i} & \frac{\delta f(i,j)}{\delta j} \end{bmatrix} \begin{bmatrix} \Delta i \\ \Delta j \end{bmatrix} \right)^2$$

$$SSD_W(\Delta i, \Delta j) = \sum_{(i,j)\in W} \left(\begin{bmatrix} \frac{\delta f(i,j)}{\delta i} & \frac{\delta f(i,j)}{\delta j} \end{bmatrix} \begin{bmatrix} \Delta i \\ \Delta j \end{bmatrix} \right)^2$$

$$SSD_W(\Delta i, \Delta j) = \sum_{(i,j)\in W} \left([\Delta i \Delta j] \left(\begin{bmatrix} \frac{\delta f(i,j)}{\delta i} \\ \frac{\delta f(i,j)}{\delta j} \end{bmatrix} \begin{bmatrix} \frac{\delta f(i,j)}{\delta i} & \frac{\delta f(i,j)}{\delta j} \end{bmatrix} \right) \begin{bmatrix} \Delta i \\ \Delta j \end{bmatrix} \right)$$

$$SSD_W(\Delta i, \Delta j) = [\Delta i \Delta j] \left(\sum_{(i,j)\in W} \begin{bmatrix} \frac{\delta f(i,j)}{\delta i} \\ \frac{\delta f(i,j)}{\delta j} \end{bmatrix} \begin{bmatrix} \frac{\delta f(i,j)}{\delta i} & \frac{\delta f(i,j)}{\delta j} \end{bmatrix} \right) \begin{bmatrix} \Delta i \\ \Delta j \end{bmatrix}$$

$$SSD_W(\Delta i, \Delta j) = [\Delta i \Delta j] \begin{bmatrix} \sum_{(i,j)\in W} \left(\frac{\delta f(i,j)}{\delta i} \right)^2 & \sum_{(i,j)\in W} \frac{\delta f(i,j)}{\delta i} \frac{\delta f(i,j)}{\delta j} \\ \sum_{(i,j)\in W} \frac{\delta f(i,j)}{\delta i} \frac{\delta f(i,j)}{\delta j} & \sum_{(i,j)\in W} \left(\frac{\delta f(i,j)}{\delta j} \right)^2 \end{bmatrix} \begin{bmatrix} \Delta i \\ \Delta j \end{bmatrix}$$

Această ecuație permite calcularea valorilor proprii (λ_1, λ_2)ale matricei centrale. Dacă ambele valori proprii sunt mari, atunci este un colț. Dacă doar una dintre ele este mare, atunci este o muchie, altfel este o regiune constantă. Ținând cont de aceste proprietăți, Harris a propus următoarea măsură a posibilitații de a fi colț:

$$C(i,j) = det(M) - k\big(trace(M)\big)^2$$

unde k este o constantă determinată empiric cu valori între 0.04 și 0.06, iar:

$$M = \begin{bmatrix} \sum_{(i,j)\in W} \left(\frac{\delta f(i,j)}{\delta i} \right)^2 & \sum_{(i,j)\in W} \frac{\delta f(i,j)}{\delta i} \frac{\delta f(i,j)}{\delta j} \\ \sum_{(i,j)\in W} \frac{\delta f(i,j)}{\delta i} \frac{\delta f(i,j)}{\delta j} & \sum_{(i,j)\in W} \left(\frac{\delta f(i,j)}{\delta j} \right)^2 \end{bmatrix} = \begin{bmatrix} A & B \\ B & C \end{bmatrix}$$

$$det(M) = \lambda_1 \lambda_2 = AC + B^2$$

$$trace(M) = \lambda_1 + \lambda_2 = A + C$$

Însumarea asupra zonei de imagine W este pondertă cu scopul de a pune mai mult accent pe măsurătorile care sunt făcute mai aproape de zona centrală a ferestrei. Se calculează folosind funcția Gaussian.

0.04	0.12	0.04
0.12	0.36	0.12
0.04	0.12	0.04

Figura 9. Ponderea Gaussiană pentru fereastră de 3x3 folosită in detectorul de colțuri Harris

Operatorul Harris este semnificativ mai costisitor computațional decât cel a lui Moravec, este sensibil la zgomot și răspunsul se schimbă in funcție de orientare. Cu toate acestea, detectorul Harris este unul dintre cele mai folosite detectoare de colțuri, datorită urmatorilor factori:

a. Are un răspuns repetabil.

b. Are o rată mai bună de detecție decât cea a lui Moravec.

Figura 10. Detectorul de colțuri Harris. Imaginea grayscale (stânga) a fost prelucrată ca să se obțină harta de colțuri din centru care a fost folostă pentru a extrage colțurile (dreapta)

Noțiunile legate de îmbunatățirea imaginilor au fost adaptate după cartea [6]

3. Lucrări in domeniu

3.1 Noțiuni introductive

Sistemele biometrice folosesc două metode: verificare sau identificare. Prin verificare se controlează dacă o persoană este cine pretinde să fie, iar prin identificare se caută într-o bază de date o persoană și se spune cine este această persoană.Sistemele de verificare a semnăturii clasifică semnătura în două categorii: autentice și falsuri.

Falsurile sunt de trei tipuri: aleatorii, simple și intenționate. Pentru cele aleatorii, falsificatorul nu are informații despre utilizator sau despre semnătura acestuia și își folosește semnătura proprie pentru verificare. Cele două semnături au semnificații semnatice diferite, prezentând forme diferite per total. În cazul falsurilor simple, falsificatorul cunoaște numele utilizatorului, dar nu îi știe semnătura. Semnătura falsificatorul ar putea prezenta mai multe similarități cu semnătura autentică, mai ales dacă semnătura utilizatorului conține și numele acestuia. Pentru falsurile intenționate, falsificatorul are acces la numele și semnătura utilizatorului, iar acesta, de obicei, exersează imitarea semnăturii utilizatorului. Aceste falsuri au similarități mai mari cu cea autentică și sunt mai greu de detectat.

Verificarea semnăturilor, în funcție de metoda folosită, este de două feluri: online (dinamică) și offline (statică). Verificarea online se face în timp ce aceasta este făcută de către utilizator. Verificarea se face pe baza poziției instrumentului de scris, se pot adăuga și alte informații, precum înclinația instrumentului sau presiunea pe care o exercită. În cazul verificării offline, semnătura se verifică după ce a fost făcută. Semnătura este reprezentată ca o imagine digitală, de obicei formatul grayscale.

3.2 Provocări

Cea mai mare provocare pentru verificarea semnăturilor este variabilitatea acestora. În comparație cu trăsături biometrice, precum amprenta sau irisul, semnăturile de la același utilizator pot avea poziții sau forme diferite una față de cealaltă (se poate observa în Figura 11).

Figura 11. Mai multe semnături suprapuse de la același utilizator

Această provocare devine mai dificilă atunci când variabilitatea este mică între semnăturile autentice și cele ale unui falsificator. În Figura 12 se poate observa că se poate ca semnături autentice să difere între ele, iar unele falsuri intenționate pot semnăna foarte tare cu anumite semnături autentice.

O altă provocare pentru verificarea automată de semnături este prezența doar de informații parțiale pe parcursul antrenamentului. În realitate, în timpul antrenamentului există acces doar la semnăturile autentice pentru utilizatorii din baza de date. Când se fac operații, în schimb, nu se vrea doar ca sistemul să accepte semnături autentice, ci să le respingă pe cele false. Această provocare este dificil de combătut deoarece un clasificator în timpul antrenamentului nu știe cum să facă diferența dintre o semnătură autentică sau una falsă.

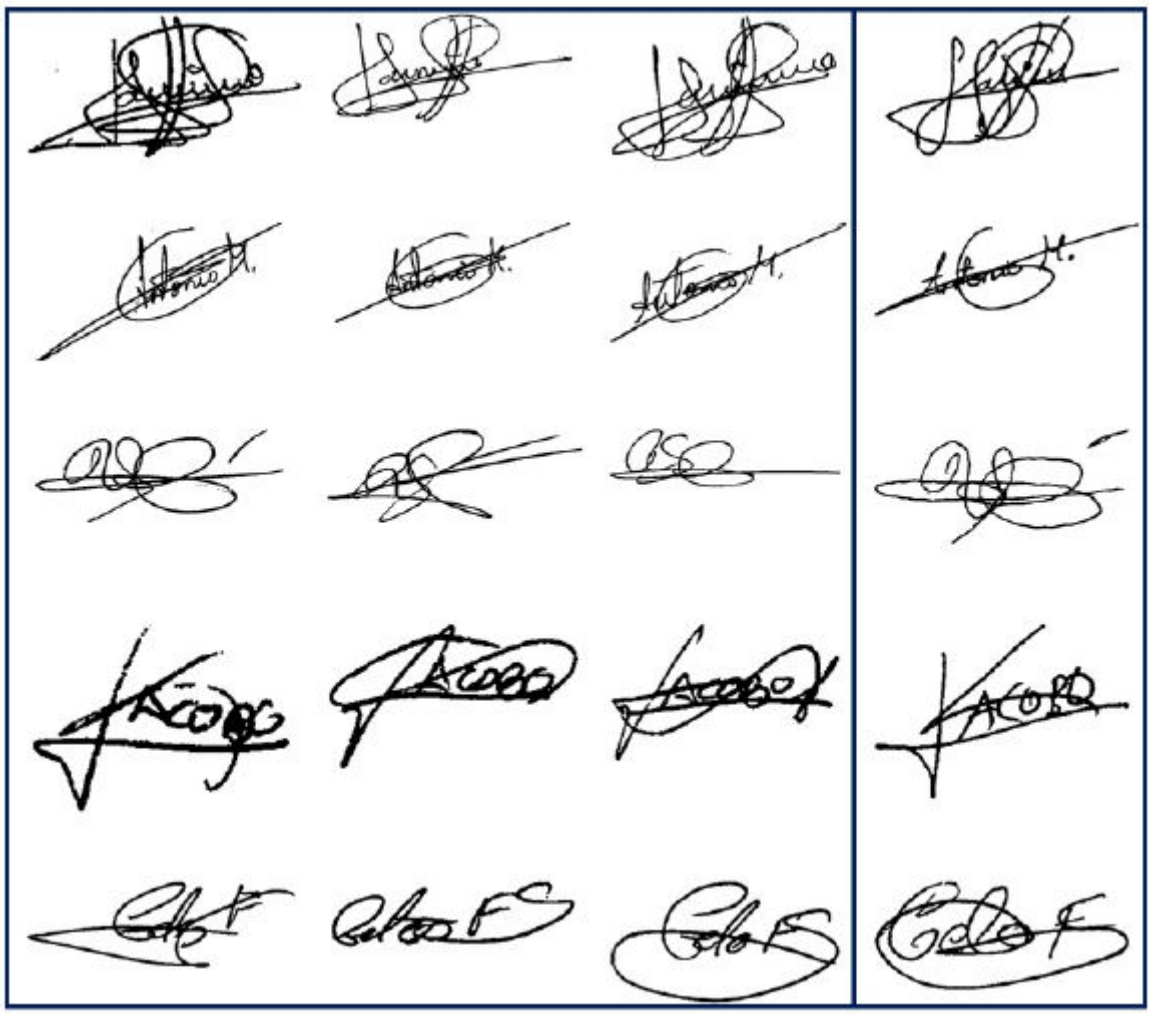

Figura 12. Exemplu de semnături autentice (stânga) și falsuri intenționate (dreapta) din setul de date GPDS-160

3.3 Seturi de date

Multe cercetări pentru verificarea semnăturii automat au fost făcute cu seturi de date private. Acest lucru face dificilă compararea cercetărilor între ele, deoarece nu se știe dacă îmbunătățirea a fost datorită unei metode mai bune de verificare sau unui set de date mai clar. În ultimul deceniu, mai multe seturi de date cu semnături au fost făcute publice.

Procesul de obținere a imaginilor cu semnături are cam aceiași pași pentru majoritatea seturilor de date. Semnăturile autentice sunt obținute în una sau mai multe sesiuni și utilizatorul trebuie să ofere mai multe exemplare pentru aceeași semnătură. Utilizatorul primește un formular cu mai multe celule și trebuie să completeze fiecare celulă cu câte un exemplar din semnătura lui. Colectarea falsurilor folosește alt proces: utilizatorul primește mai multe exemplare de semnături autentice și i se cere să încerce să imite semnăturile. Pentru unele seturi de date, utilizatorii care se folosesc pentru obținerea semnăturilor autentice, se folosesc și pentru obținerea falsurilor (un utilizator falsifică semnăturile altor utilizatori), iar pentru alte seturi de date, falsurile au fost create cu ajutorul altor utilizatori. Figura 13 arată formularul de colectare a semnăturilor pentru un set de date. După ce formularele sunt colectate, sunt scanate și prepocesate. Preprocesarea constă în separarea semnăturilor în imagini individuale și, pentru unele seturi de date, se fac și alte acțiuni, precum eliminarea zgomotului.

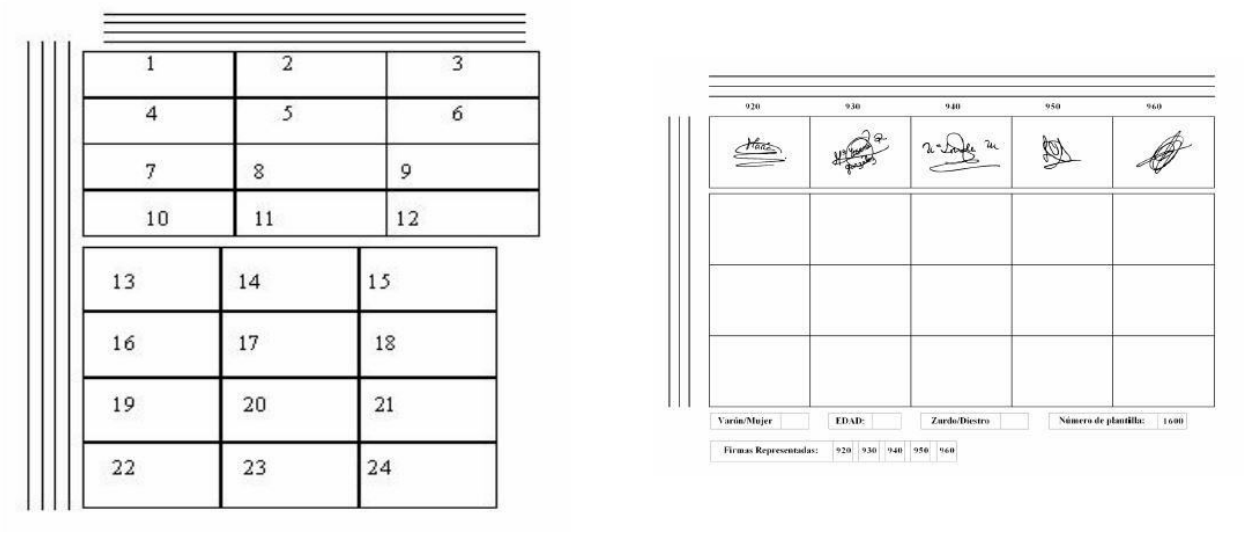

Figura 13. Formulare pentru colectarea semnăturilor pentru setul de date GPDS-960. În stânga se află un formular pentru colectarea semnăturilor autentice, iar în dreapta un formular pentru colectarea falsurilor

Următorul tabel conține cele mai utilizate seturi de date.

Numele setului de date	Numărul utilizatorilor	Numărul semnăturilor autentice	Numărul falsurilor
Brazilian (PUC-PR)	60+108	40	10 simple, 10 intenționate
CEDAR	55	24	24
MCYT-75	75	15	15
MCYT-100	100	25	25
GPDS-160	160	24	30
GPDS-960	960	24	30
GPDS-960 Grayscale	881	24	30

Tabelul 2. Exemple de seturi de date

3.4 Preprocesarea

Preprocesarea joacă un rol important în procesul de verificare a semnăturii. Semnăturile pot conține variație în grosime, dimensiune sau rotație.

Cele mai folosite tehnici de preprocesare sunt: extragerea de semnături, eliminarea zgomotului, normalizarea dimensiunii și centrarea semnăturii.

- Extragerea semnăturii – această tehnică constă în găsirea și extragerea unei semnături dintr-un document. Extragerea poate să devină dificilă când vine vorba de cecuri bancare, unde semnătura se poate afla deasupra unui fundal complex. Însă, acest caz nu este prea cercetat deoarece studiile folosesc semnături deja extrase din documente, care sunt separate câte una și pe un fundal alb.
- Eliminarea zgomotului – imaginile digitale ale semnăturilor sunt obținute, de obicei, prin scanare, iar acest proces poate produce zgomot pe semnătură, un pixel negru se poate afla pe un fundal alb sau un pixel alb pe un fundal negru. O strategie foarte folosită este aplicarea unui filtru de eliminare a zgomotului din imagine, cum ar fi filtrul median din Figura 14.

Figura 14. Aplicarea unui filtru pe median pe o imagine cu zgomot

- Normalizarea dimensiunii și centrarea semnăturii – cea mai simplă strategie de normalizarea este tăierea semnăturii, astfel încât marginile să atingă semnătura în patru direcții (sus, jos, stânga și dreapta). O altă strategie este tăierea părților din imagine care sunt la o distanță mare de centroidul acesteia. Se mai poate seta o înălțime, după care semnătura să fie aproximată sau se pot fixa marginile, unde semnătura să fie centrată: se calculează centroidul semnăturii și se folosește ca să se centreze imaginea, adăugând părți albe pentru a umple restul imaginii.
- Reprezentarea semnăturii – pe lângă folosirea de imagini de intrare grayscale, mai există și alte reprezentări pentru semnături. În Figura 15 se pot observa câteva dintre aceste reprezentări

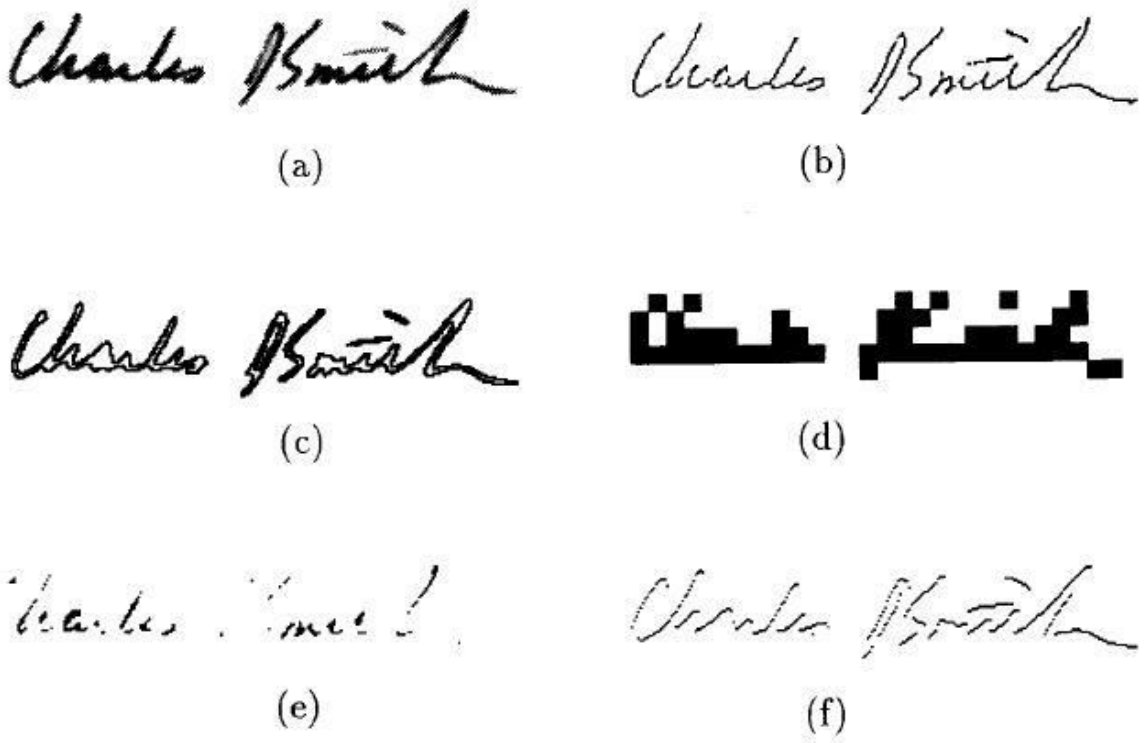

Figura 15. Reprezentări ale unei semnături: a) originala, b) schelet, c) contur, d) distribuția cernelii, e) regiunile cu intensitate mai mare, f) direcția frontierei.

3.5 Extragerea de caracteristici

După ce semnăturile au fost preprocesate, următorul pas este de a extrage caracteristicile distincte pentru fiecare semnătură. Metodele de extragere pot folosi caracteristici globale sau locale. La cele globale se reprezintă semnătura ca un întreg, sunt caracteristici precum înălțimea, lățimea, și se aplică metodele de extragere pe întreaga imaginea cu semnătura. În schimb, la cele locale se reprezintă doar părți din semnătură sau se împarte toată imaginea în subimagini și se aplică metoda de extragere pentru fiecare parte sau subimagine în parte.

3.5.1 Caracteristici geometrice

Caracteristicile geometrice reprezintă forma semnăturii ca un întreg. Se folosesc descriptori de bază, precum înălțimea, lățimea sau raportul înălțime/lățime. Pe lângă folosirea de caracteristici globale, se mai folosesc și caracteristici locale prin divizarea semnăturii în mai multe subimagini și extragerea caracteristicilor pentru fiecare subimagine în parte, de exemplu, calcularea densității pixelilor pentru o subimagine. O metodă de a folosi aceste caracteristici este de a utiliza diferite reprezentări ale semnăturii (contur sau schelet) ca imagine de intrare,după

care se împarte fiecare reprezentare în mai multe subimagini şi se foloseşte densitatea pixelilor pentru fiecare subimagine ca şi caracteristică pentru semnătură.

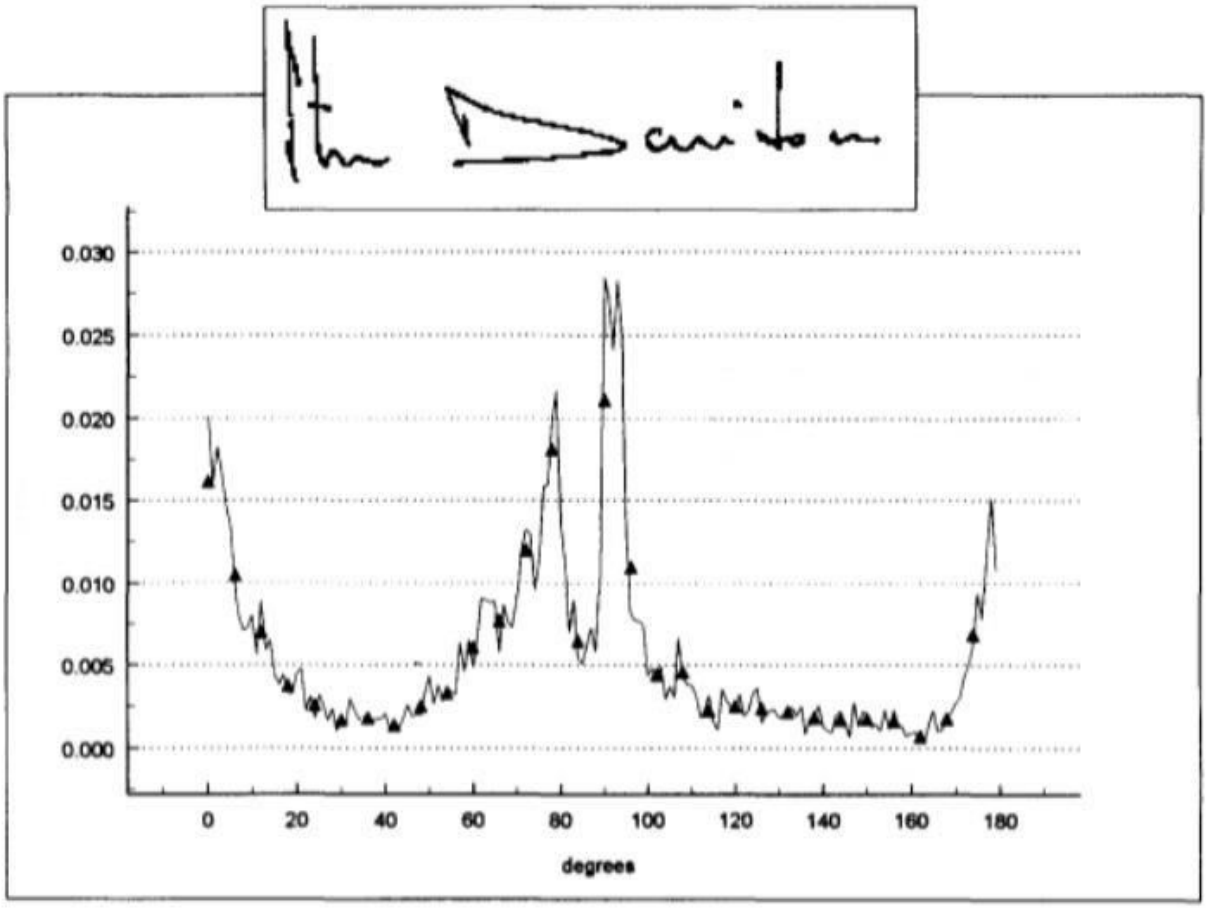

Figura 16. Exemplu caracteristici de direcție într-o semnătură

3.5.2 Caracteristici direcționale

Caracteristicile imaginii reprezintă o imagine în funcție de nivelele de apăsare (strokes) din semnătură. În Figura 16 se poate observa un exemplu de extragere a direcțiilor din conturul unei semnături. Recent, această metodă a fost îmbunătățită, folosindu-se diferite dimensiuni ale gridului.

O altă metodă este folosirea de histograme piramidale de orientări a gradientului (PHOG). Această metodă reprezintă forme din imagine printr-o histogramă de orientări ale muchiilor pentru diferite dimensiuni. Această metodă a obținut rezultate foarte bune pentru un experiment care folosea falsuri intenționate la antrenament.

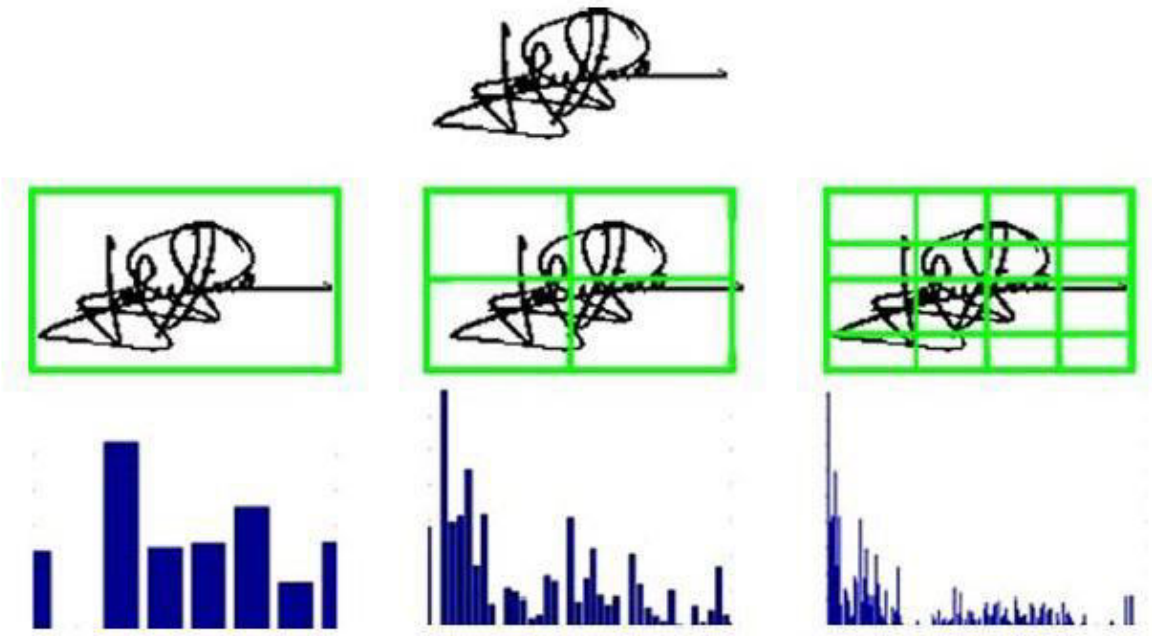

Figura 17. Caracteristici PHOG

3.5.3 Shadow code

Pentru această metodă se folosește un grid deasupra semnăturii. Gridul conține bare verticale, orizontale și diagonale, fiecare bară conține un număr exact de segmente. Se calculează pentru fiecare pixel din imagine cea mai apropiată bară și se colorează segmentul din bară care este cel mai aproape de acel pixel. După ce se colorează, se numără segmentele pentru fiecare bară, iar aceste numere sunt folosite ca și caracteristici pentru semnătură.

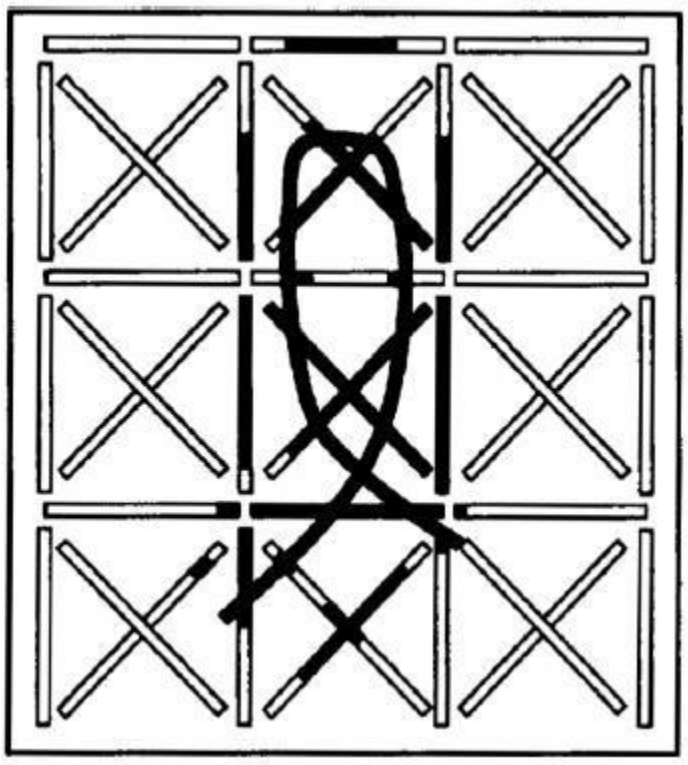

Figura 18. Exemplu de Shadow code

3.5.4 Caracteristici de textură

Pentru extragerea caracteristicelor de textură se folosesc diferite variante de modele binare locale (LBP). Primul operator LBP creat determină care sunt modelele pentru vecinătatea fiecărui pixel. Pentru un pixel se determină cei opt vecini, dacă valoarea pixelului central este mai mare decât cea a unui vecin, acel vecin ia valoarea 0, dacă este mai mică, ia valoarea 1. Pentru fiecare dintre cele opt direcții posibile, este asociată o valoare între2^0și 2^7și se păstrează valoarea doar a acelor vecini care aveau înainte valoarea 1, ceilalți rămânând cu valoare 0. Au fost create și extensii, precum tLBP, mLBP, vLBP și OCLBP, cel din urmă aducând îmbunătățiri pentru recunoașterea fețelor.

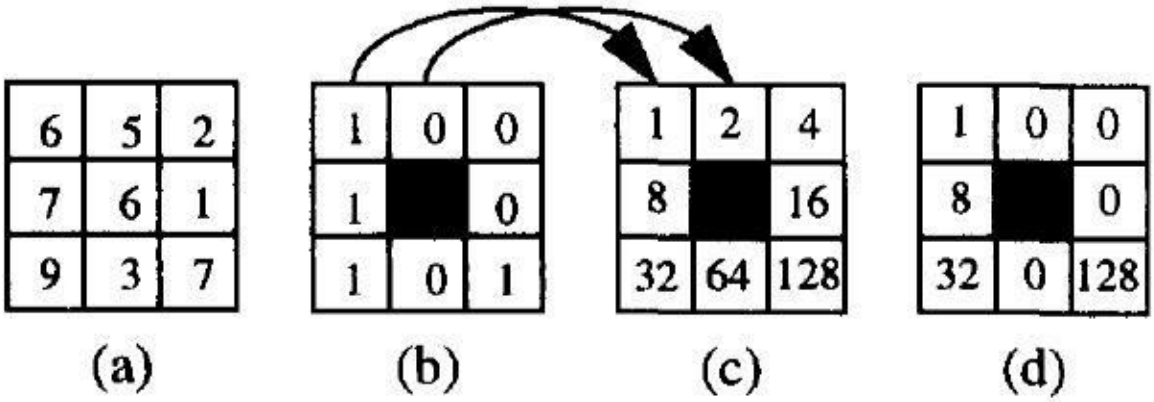

Figura 19. Exemplu pentru primul operator LBP creat

3.6 Concluzie

În ultimul deceniu, au fost propuse mai multe metode pentru verificarea offline a semnăturilor. În ciuda progreselor, rezultatele tot au erori mari când vine vorba de diferențierea semnăturilor autenticele de falsuri pentru seturi de date mari, precum GPDS. Nevoia de îmbunătățire a rezultatelor crește deoarece verificarea semnăturilor este folosită în practică în domenii unde rata erorilor trebuie sa fie cât mai mică.

Noțiunile legate de lucrările din domeniu au fost adaptate după cartea [7]

4. Descrierea aplicației

„Recunoașterea semnăturii” este o aplicație care identifică cele mai asemănătoare semnături dintr-o bază de date pentru o semnătură aleasă de utilizator.

În aplicație se folosesc diferite procedee, precum împărțirea unei imagini în subimagini, aplicarea algoritmului Sobel sau calcularea de norme euclidiene.

Recunoașterea se face în mai mulți pași:

(1) Se deschide o imagine, care conține o semnătură, dorită din baza de date.

(2) Această imagine este împărțită în 36 de subimagini, formatul 6x6, și se creează un vector de trăsături cu 36 de valori, fiecare valoare fiind numărul de pixeli de culoare neagră dintr-o subimagine.

(3) Se aplică detectorul de muchii Sobel pe imaginea de la pasul (1), se împarte imaginea în 36 de imagini, formatul 6x6, și se creează un vector de trăsături cu 36 de valori, fiecare valoare fiind numărul de pixeli de culoare albă dintr-o subimagine.

(4) Se creează un vector de trăsături care concatenează vectorii de trăsături de la pașii (2) și (3).

(5) Se creează un fișier text. Acest fișier conține pe fiecare linie câte un vector de trăsături concatenat cu lungimea și înălțimea imaginii specifice acelui vector (lungimea și înălțimea sunt folosite la pasul (6)). Vectorii de trăsături sunt obținuti prin aplicarea pasului (4) asupra tuturor imaginilor din baza de date. Acest pas este necesar doar o singură dată, fișierul, odată creat, poate fi folosit atâta timp cât nu se schimbă baza de date, formatul de împărțire a imaginii sau algoritmii folosiți.

(6) Se calculează norma euclidiană dintre vectorul de trăsături rezultat la pasul (4) și fiecare vector de trăsături din fișierul de la pasul (5), iar lungimea și înălțimea unei imagini ajută la calcularea normei.

(7) Sunt afișate primele trei imagini cu norma cea mai mică.

Aplicația este una desktop, scrisă în limbajul C#, interfața grafică este facută cu Windows Presentation Foundation, folosește librăria Emgu CV pentru diferite funcționalități și Linq pentru ordonarea vectorilor de trăsături.

Baza de date este formată din 912 semnături. Există 76 de persoane, fiecare persoană are câte 12 semnături.[8]

Interfața grafică este compusă din patru butoane cu diferite funcționalități, patru imagini, patru Label-uri cu denumiri descriptive și trei TextBlock-uri cu valori specifice.

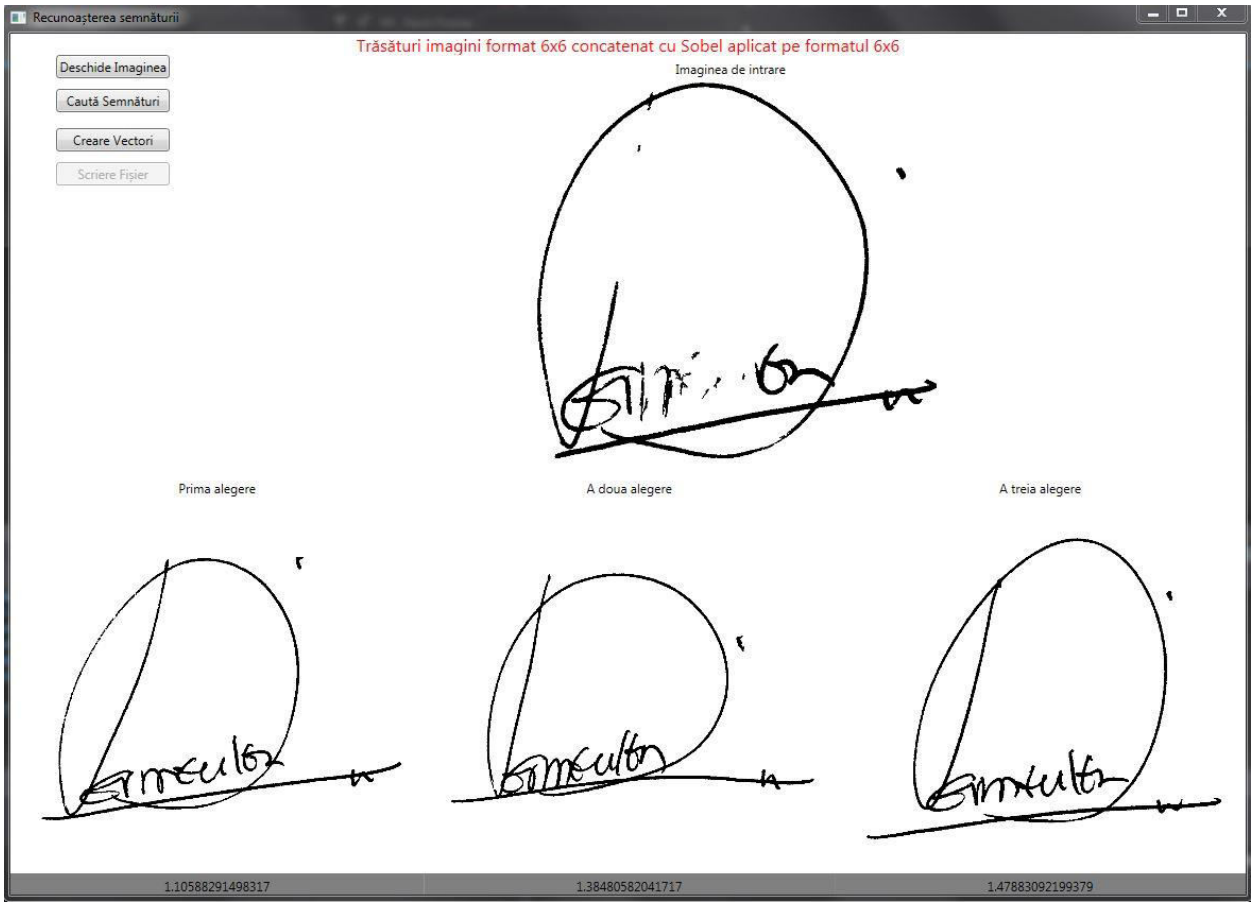

Figura 20. Interfața grafică – descrierea fișierelor de lucru

4.1 Componentele aplicației

1. Butonul „Deschide Imaginea”

Acest buton permite alegerea unei imagini din baza de date. Se folosește un OpenDialog care are ca filtru imaginile cu formatul JPEG și Portable Network Graphic (PNG). După ce este aleasă o imagine, aceasta apare în interfață împreună cu un Label care are mesajul „Imaginea de intrare”.

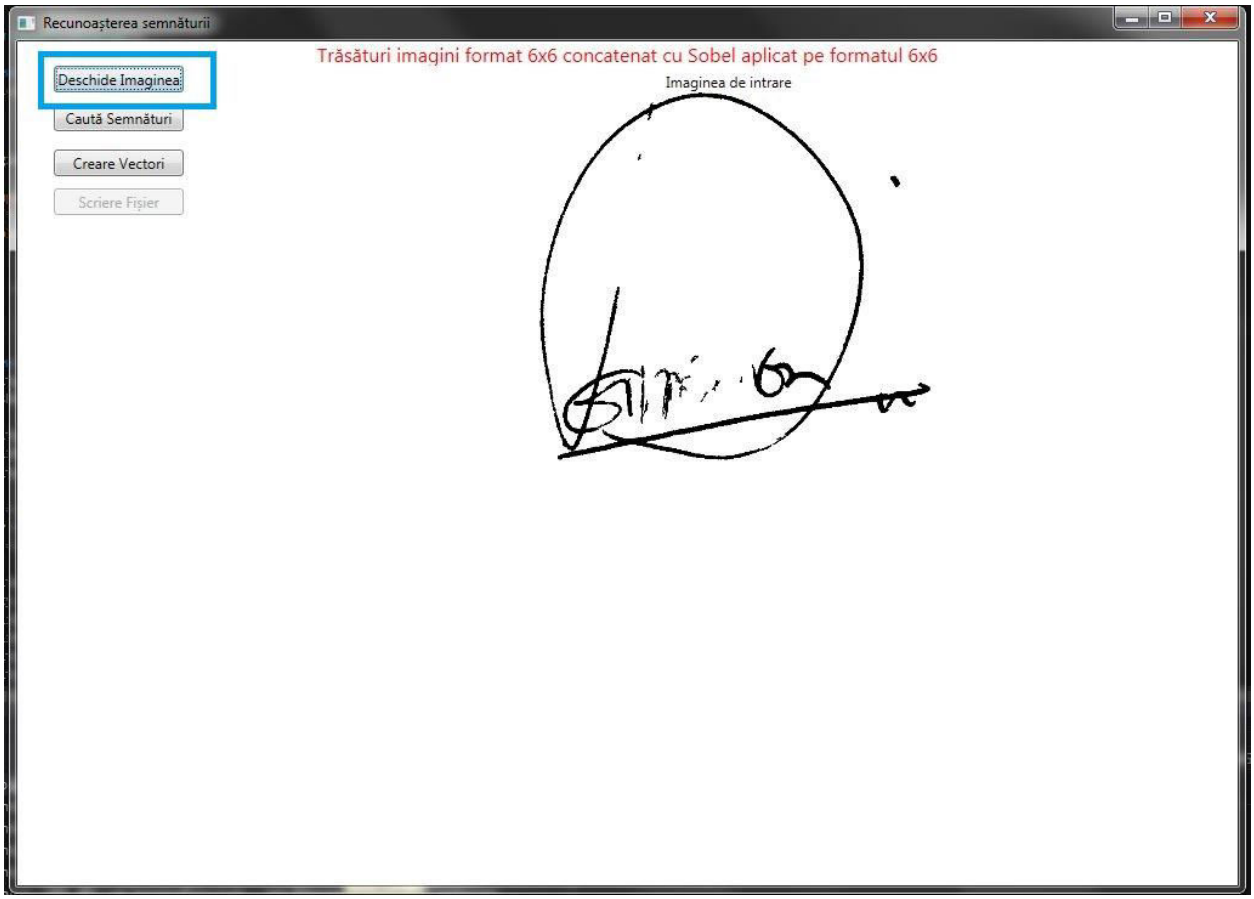

Figura 21. Interfața grafică – deschiderea imaginii

2. Butonul „Caută Semnături”

Butonul acesta devine activ doar după ce este aleasă o imagine prin butonul „Deschide Imaginea”. Rolul lui este să creeze vectorul de trăsături al imaginii alese și să calculeze normele euclidiene dintre acest vector și vectorii din fișierul text creat în prealabil. După carese afișează în interfață următoarele:

- primele trei imagini care au norma euclidină cea mai mică

- Labeluri specifice pentru fiecare imagine cu mesajele: „Prima alegere”, „A doua alegere” și „A treia alegere”

- trei TextBlock-uri cu valorile normelor euclidiene pe un fundal gri

Dacă s-a apăsat butonul „Deschide imaginea” din nou, imaginile, Labelurile și TextBlock-urile menționate mai sus dispar, ele apar iar doar dacă este reapăsat butonul „Caută Semnături”.

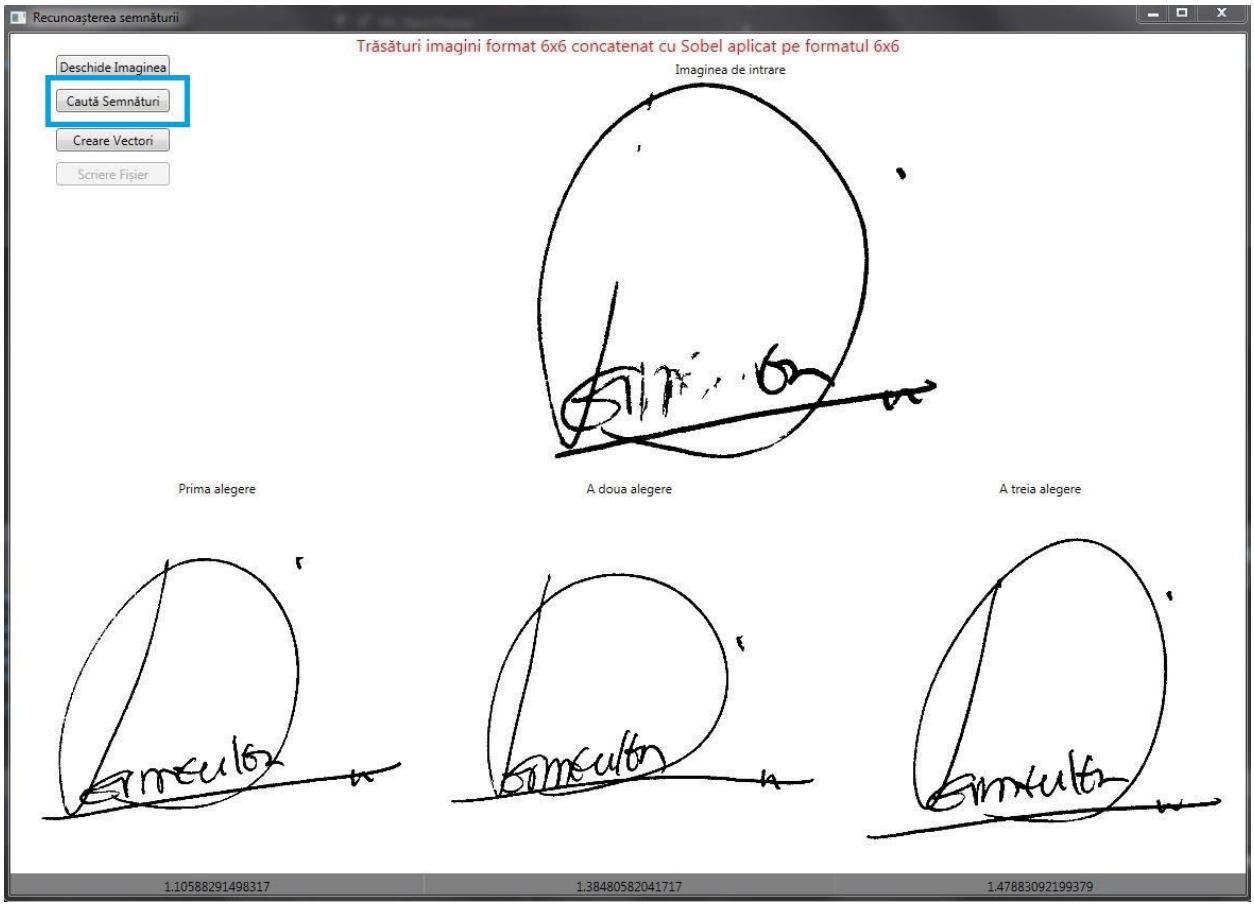

Figura 22. Interfața grafică – căutarea semnăturilor

3. Butonul „Creare Vectori”

Crearea vectorilor de trăsături se face cu ajutorul acestui buton. Toate imaginile din baza de date sunt accesate și li se creează vectorii de trăsături în paralel cu ajutorul firelor de execuție. S-a folosit pentru firele de execuție o listă de Task-uri din clasa System.Threading.Tasks.

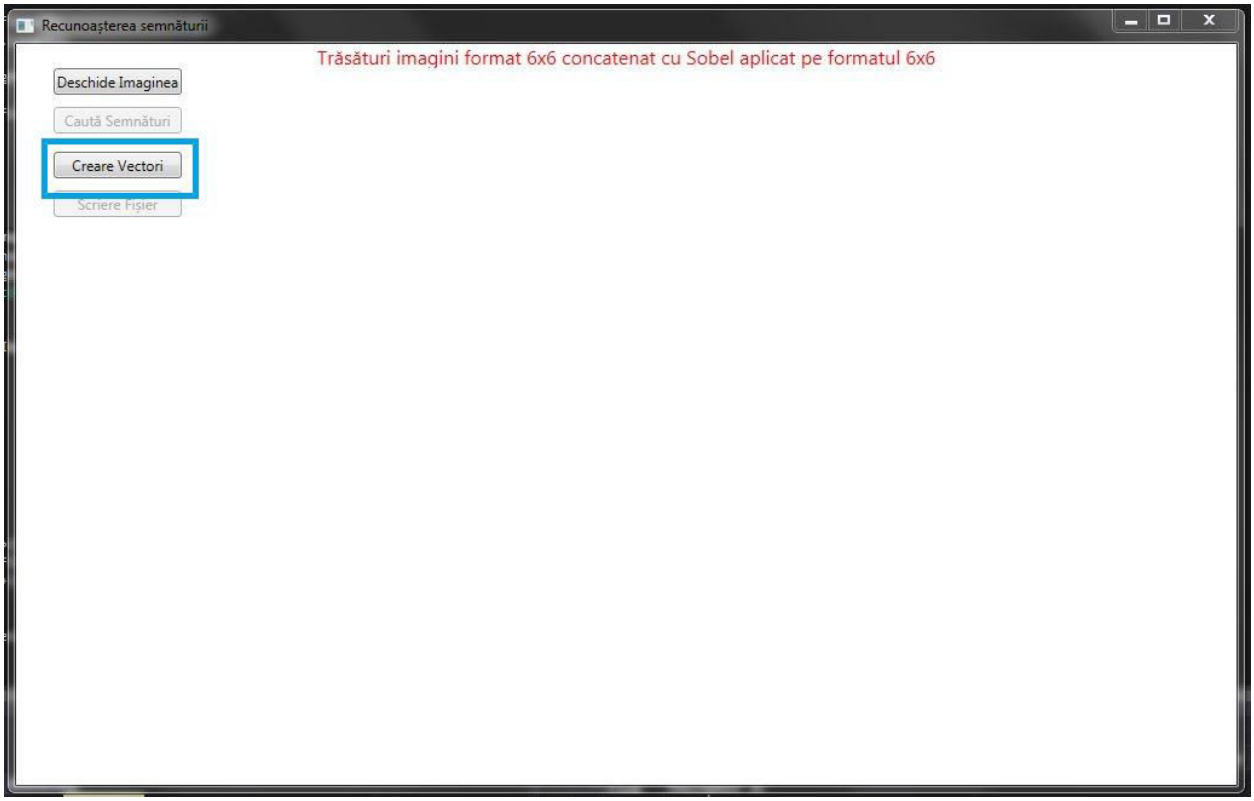

Figura 23. Interfața grafică – crearea vectorilor

4. Butonul „Scriere fișier”

Acest buton devine activ doar după ce este executat butonul „Creare Vectori”. Rolul lui este de a scrie într-un fișier text vectorii de trăsaturi creați cu ajutorul butonului menționat mai sus. Din cauză că firele de execuție se execută în paralel, vectorii de trăsături ai imaginilor își pierd ordinea din baza de date, de aceea, aceștia sunt ordonați în funcție de un index, după care sunt scrișI în fișier. Scrierea se face cu ajutorul metodei WriteLine din cadrul clasei StreamWriter, din System.IO, iar ordonarea se face cu ajutorul unui query din Linq.

Figura 24. Interfața grafică – scrierea fișierului

4.2 Statistici

Pentru aplicația lucrării s-a folosit cazul cu cel mai bun rezultat, adică vectorii de trăsaturi ai imaginilor cu formatul 6x6 concatenați cu vectorii de trăsături ai imaginilor cu formatul 6x6 pe care a fost aplicat operatorul Sobel. Însă, pentru această lucrare au fost încercate mai multe formate și mai mulți algoritmi, în diferite forme.

Pentru fiecare caz există 4 formate (5x5, 6x6, 5x7 și 6x8) și diferite concatenări cu sau fără algoritmi aplicați pe imagini. Pentru fiecare format se prezintă numărul de reușite, de eșecuri și procentajul reușitelor.

4.2.1 1-NN cu „leave-one-out”

Statisticile pentru cazurile care urmează sunt făcute cu ajutorul algoritmului k-Nearest Neightbor (k-NN), varianta 1-NN pe baza metodei de validare „cross-validation” de tipul „leave-one-out”.

Ca să se afle dacă este o reușită sau un eșec se ia o semnătură și se calculează normele euclidiene cu toate celelalte semnături. Dacă semnătura cu cea mai mică normă euclidiană și semnătura aleasă sunt de la aceeași persoană, atunci este o reușită, altfel, un eșec. Pentru

procentajul reușitelor se ia numărul total de reușite și se calculează cât la sută este din totalul imaginilor.

1. Imagini simple

Pentru acest caz imaginea cu semnătura este împarțită în formatele prezentate, respectiv 25, 36, 35 sau 48 de subimagini. Numărul de valori dintr-un vector de trăsături depinde de numărul de subimagini. O valoare este numărul de pixeli negri dintr-o subimagine.

	5x5	**6x6**	5x7	6x8
Reușite	804	**833**	810	823
Eșecuri	108	**79**	102	89
Procentaj reușite	88.15%	**91.33%**	88.81%	90.24%

Tabelul 3. Statistici 1-NN – rezultate pentru imagini simple

Rezultatele cele mai bune pentru acest caz le oferă formatul 6x6, care are procentajul reușitelor foarte bun și doar cinci reușite mai puțin față de cel mai bun rezultat dintre toate cazurile de la 1-NN.

2. Imagini pe care a fost aplicat algoritmul lui Harris

Se aplică algoritmul de detecție a colțurilor a lui Harris pe imagine, după care aceasta este împărțită în formatele prezentate. Numărul de valori dintr-un vector de trăsături depinde de numărul de subimagini. O valoare este numărul de pixeli negri dintr-o subimagine.

	5x5	6x6	5x7	**6x8**
Reușite	784	808	812	**818**
Eșecuri	128	104	100	**94**
Procentaj reușite	85.96%	88.59%	89.03%	**89.69%**

Tabelul 4. Statistici 1-NN – rezultate pentru imagini pe care a fost aplicat algoritmul lui Harris

În acest caz, aplicarea operatorului Harris face ca rezultatele să scadă pentru aproape toate formatele. Formatul 5x7 este singurul care are rezultate mai bune față de cazul precedent. Formatul 6x6, care avea cele mai bune rezultate la cazul anterior, scade cu cele mai multe reușite, iar cel 6x8, deși are rezultate mai slabe decât înainte, are cel mai bun procentaj de reușite pentru acest caz.

3. Imagini pe care a fost aplicat algoritmul lui Sobel

În acest caz se aplică algoritmul de detecție a muchiilor a lui Sobel pe imagine, după care aceasta este împărțită în subimagini. Numărul de valori dintr-un vector de trăsături depinde de numărul de subimagini. O valoare este numărul de pixeli albi dintr-o subimagine.

	5x5	6x6	5x7	**6x8**
Reușite	793	812	810	**821**
Eșecuri	119	100	102	**91**
Procentaj reușite	86.95%	89.03%	88.81%	**90.02%**

Tabelul 5. Statistici 1-NN – rezultate pentru imagini pe care a fost aplicat algoritmul lui Sobel

Acest caz are rezultatele mai bune pentru majoritatea cazurilor față de cazul precedent, dar mai slabe sau la fel față de primul caz. Formatul 6x8 crește față de cazul anterior, având iar cel mai mare procentaj al reușitelor. Formatul 5x7 are cele mai neașteptate rezultate de până acum, atunci când rezultatele de la celelalte formate scad, ale lui cresc, iar când rezultatele de la celelalte formate cresc, ale lui scad.

4. Imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Harris

Pentru acest caz se împarte imaginea cu semnătura în subimagini, după care se aplică operatoul Harris pe imaginea originală și se împarte la fel. Numărul de valori dintr-un vector de trăsături este numărul de subimagini de la imaginea originală adunat cu numărul de subimagini de la imaginea cu algoritmul lui Harris aplicat. O valoare este numărul de pixeli de culoare neagră dintr-o subimagine.

	5x5	**6x6**	5x7	6x8
Reușite	804	**837**	825	830
Eșecuri	108	**75**	87	82
Procentaj reușite	88.15%	**91.77%**	90.46%	91.01%

Tabelul 6. Statistici 1-NN – rezultate pentru imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Harris

Concatenarea vectorilor de trăsături aduce o îmbunătățire semnificativă asupra rezultatelor. Vectorii de trăsături se dublează și oferă mai multe detalii despre o semnătură. Formatul 5x5 este singurul care nu are cele mai bune rezultate ale lui de până acum, având același rezultat ca la primul caz (cel cu imagini simple). Deși are o creștere de aproape 1% pentru procentajul de reușite, formatul 6x8 nu este pe primul loc, fiind depășit de formatul 6x6, care are cele mai bune rezultate dintre toate formatele pentru acest caz, dintre cele mai bune rezultate

pentru toate cazurile 1-NN și cele ale formatului 6x6 pentru acest caz este diferență doar de o reușită.

5. Imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Sobel

Se împarte imaginea conform formatelor prezentate, după care se aplică algoritmul de detecție a muchiilor a lui Sobel pe imaginea originală și se împarte la rândul ei. Numărul de valori dintr-un vector de trăsături este numărul de subimagini de la imaginea originală adunat cu numărul de subimagini de la imaginea cu operatorul Sobel aplicat. Pentru prima jumătate a vectorului de trăsături, o valoare este numărul de pixeli albi din subimaginile rezultate din imaginea cu algoritmul lui Sobel aplicat, iar pentru a doua jumătate, o valoare este numărul de pixeli de culoare neagră din subimaginile rezultate din imaginea originală.

	5x5	**6x6**	5x7	6x8
Reușite	813	**838**	826	833
Eșecuri	99	**74**	86	79
Procentaj reușite	89.14%	**91.88%**	90.57%	91.33%

Tabelul 7. Statistici 1-NN – rezultate pentru imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Sobel

Dacă pentru cazul precedent concatenarea aduce îmbunătățiri mari asupra rezultatelor, pentru acest caz, le aduce pe cele mai bune pentru toate formatele dintre toate cazurile de la 1-NN. Formatul 6x6 are procentajul de reușite cel mai mare, fiind de aproape 92%. Deși are tot cele mai slabe rezultate, formatul 5x5 are rezulate rezonabile, depășind pragul de 89%. Formatul 6x8 reușește,alăuri de formatul 6x6 de la primul caz, să ajungă pe locul trei la cele mai bune rezultate dintre toate cazurile. Formatul 5x7 are cea mai mică creștere dintre toate, având doar cu o reușită mai mult decât la cazul anterior.

4.2.2 Top trei

Pentru cazurile următoare, statisticile sunt facută cu ajutorul celor mai mici norme euclidine rezultate.

O reușită este dacă printre primele trei cele mai mici norme euclidiene pentru o semnătură, se află cel puțin una de la aceeași persoană, altfel, este eșec. Pentru procentajul reușitelor se ia numărul total de reușite și se calculează cât la sută este din totalul imaginilor.

Cazurile care urmează sunt la fel ca cele de la 1-NN, diferă doar modul în care se calculează o reușită.

1. Imagini simple

	5x5	**6x6**	5x7	6x8
Reușite	838	**865**	847	859
Eșecuri	74	**47**	65	53
Procentaj reușite	91.88%	**94.84%**	92.87%	94.18%

Tabelul 8. Statistici Top trei – rezultate pentru imagini simple

Pentru primul caz de la top trei, rezultatele cele mai bune le are formatul 6x6, având al doilea cel mai bun procentaj al reușitelor, alături de încă două formate de la cazuri diferite (formatul 6x6 de la cazul al patrulea și formatul 6x8 de la ultimul caz), dintre toate cazurile de la top trei.

2. Imagini pe care a fost aplicat algoritmul lui Harris

	5x5	6x6	5x7	**6x8**
Reușite	835	845	840	**850**
Eșecuri	77	67	72	**62**
Procentaj reușite	91.55%	92.65%	92.10%	**93.20%**

Tabelul 9. Statistici Top trei – rezultate pentru imagini pe care a fost aplicat algoritmul lui Harris

Se poate observa că aplicarea algoritmului de detecție a colțurilor a lui Harris aduce o scădere a procentajului de reușite pentru toate formatele față de cazul precedent. Deși formatul 5x5 are cele mai slabe rezultate pentru ambele cazuri, acesta are cea mai mică scădere a procentajului de reușite. Formatul 6x8 scade cu aproape cele mai multe reușite, fiind întrecut doar de formatul 6x6 care are o scădere drastică, dar reușește să aibă cele mai bune rezultate pentru acest caz.

3. Imagini pe care a fost aplicat algoritmul lui Sobel

	5x5	**6x6**	5x7	6x8
Reușite	843	**852**	850	846
Eșecuri	69	**60**	62	66
Procentaj reușite	92.43%	**93.42%**	93.20%	92.76%

Tabelul 10. Statistici Top trei – rezultate pentru imagini pe care a fost aplicat algoritmul lui Sobel

Aplicarea operatorlui Sobel aduce o îmbunătățire pentru aproape toate formatele față de cazul precedent și o îmbunătățire pentru jumătate dintre formate față de primul caz. Formatele 5x5 și 5x7 reușesc să aibă cele mai bune rezultate ale lor de până acum, în timp ce pe formatele 6x6 și 6x8 doar cele de la primul caz le întrec pe cele de la cazul acesta. Formatul 6x6 are cele mai bune rezultate pentru acest caz, chiar dacă față de cazul anterior nu are cea mai mare îmbunătățire.

4. Imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Harris

	5x5	**6x6**	5x7	6x8
Reușite	856	**865**	860	863
Eșecuri	56	**47**	52	49
Procentaj reușite	93.85%	**94.84%**	94.29%	94.62%

Tabelul 11. Statistici Top trei – rezultate pentru imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Harris

Pentru acest caz dublarea vectorilor de trăsături face ca toate formatele să aibă cele mai bune rezultate ale lor de până acum. Cea mai bună îmbunătățire față de cazul anterior o are formatul 6x8, iar cea mai slabă o are formatul 5x7. Formatul 6x6 are cele mai bune rezultate, având același număr de reușite ca la primul caz. Diferența dintre primele două formate este foarte mică, doar trei reușite desparte formatul 6x6 de cel 6x8.

5. Imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Sobel

	5x5	6x6	5x7	6x8
Reușite	846	**870**	851	865
Eșecuri	66	**42**	61	47
Procentaj reușite	92.76%	**95.39%**	90.57%	94.84%

Tabelul 12. Statistici Top trei – rezultate pentru imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Sobel

Acest caz oferă cel mai mare procentaj de reușite dintre toate cazurile de la top trei, formatul 6x6 având un procentaj de peste 95%. În schimb, formatele 5x5 și 5x7 au rezultate mai slabe decât cazul precedent, formatul 5x5 având cea mai mare scădere în numărul de reușite. Formatul 6x8 reușește să atingă procentajul de reușite al celui mai bun format de la cazul precedent, crescând cu 0,24%.

4.2.3 Comparație statistici între 1-NN și Top trei

1. Asemănări

Cea mai important asemănare este faptul că pentru ambele metode procentajul de reușite cel mai mare se află la același format și caz (formatul 6x6 de la cazul „Imagini simple concatenate cu imagini pe care a fost aplicat algoritmul lui Sobel”). Cazul „Imagini pe care a fost aplicat algoritmul lui Harris” are cel mai mic număr de reușite pentru ambele metode, iar formatul 5x5 are cel mai mic procentaj de reușite pentru toate cazurile.

2. Diferențe

Cea mai mare diferență dintre ambele metode este faptul că Top trei are numărul de reușite mult mai mare decât 1-NN. Acest lucru se datorează faptului că 1-NN este inclus în Top trei. În timp ce 1-NN prezintă ca și reușite doar cazurile în care o semnătură are cea mai asemănătoare semnătură de la aceeași persoană, Top trei arată și cazurile unde existau semnături foarte aproape să devină cea mai asemănatoare. O altă diferență ar fi că formatul 6x6 are de mai multe ori cele mai bune rezultate pentru Top trei față de1-NN.

Concluzii

Recunoașterea de semnături rămâne o problemă foarte folosită și necesară pentru multe ramuri.

Această lucrare a prezentat noțiuni teoretice legate de cum se poate pregăti o imagine pentru a fi mai ușor de lucrat cu ea, cum se pot extrage caracteristici specifice din semnături și o aplicație care are rolul de a recunoaște semnături.

În această aplicație a fost folosit algoritmul lui Sobel, cel a lui Harris și k-NN. Pașii folosiți pot fi rezumați în câteva cuvinte: împărțirea imaginilor în diferite formate, aplicarea unor algoritmi cu rol în detecție de muchii sau colțuri și calcularea unor norme euclidiene, care au fost folosite de algoritmul k-NN ca să se obțină cele mai asemănătoare semnături. Din rezultatele obținute, am dedus că cea mai eficientă metodă dintre cele folosite este concaterea imaginilor care au formatul 6x6 cu imaginile pe care a fost aplicat detectorul de muchii a lui Sobel.

Chiar dacă recunoașterea de semnături este un domeniu cercetat în ultimele decenii, încă mai există modalități prin care se poate îmbunătății. Un subiect care se poate studia mai amănunțit este cel legat de variația care există între semnăturile de la aceeași persoană.

Pentru această lucrare, pentru rezultate mai bune se pot încerca mai mulți algoritmi pe mai multe formate, mai multe concatenări diferite de cele încercate sau se poate studia mai amănunțit motivul pentru care rata erorii nu este mai mică.

Bibliografie

1. S. Jayaraman, S. Esakkirajan, T. Veerajumar, *Digital Image Processing*

2. Muzamil Bhat, International Journal of Scientific & Technology Research Volume 3, 2014, *Digital Image Processing*

3. Wasseem Nahy Ibrahem, Image Processing - http://www.uotechnology.edu.iq

4. Chris Solomon, Toby Breckon, *Fundamentals of digital image processing*

5. https://en.wikipedia.org/wiki/Corner_detection

6. Kenneth Dawson-Howe, *A practical introduction to computer vision with Open CV*

7. Luiz G. Hafemann, Robert Sabourin, Luis S. Oliveira, *Offline handwritten signature verification*

8. A fost o bază de date de lucru pentru competiția de verificare de semnături SigComp, în cadrul ICDAR 2009

9. www.msdn.microsoft.com

10 . www.csharphelper.com