Universitatea Tehnica din Cluj-Napoca
Departamentul de Calculatoare

# Procesarea imaginilor

**Digitalizarea grafurilor**

*Proiect 2018-2019*

Nume:Grigor Sonia Eufrosina Maria
Email:sonia.grigor@gmail.com

Nume:Lăzăreanu Sabina Ștefana
Email:lazareanusabina@gmail.com

Grupa:30233

As. dr. ing. Diana Borza
Diana.Borza@cs.utcluj.ro

# Cuprins

# Abstract

Scopul proiectului "Digitalizarea grafurilor" este realizarea unui produs software care își propune detectarea grafurilor și identificarea elementelor componente precum și a informațiilor din interiorul nodurilor. Proiectul are la baza procesarea imaginilor care conțin reprezentări ale grafurilor neorientate.

Propunerea de rezolvare a acestui proiect consta în identificarea fiecărui element ( nod, muchie, informație din nod), apoi maparea la o anumita structura de date.

# Chapter 1

# Introducere

Proiectul "Digitalizarea grafurilor" presupune procesarea imaginilor în care sunt reprezentate vizual (desenate) grafuri și recunoașterea nodurile și muchiile. Rezultatul algoritmului va fi graful reprezentat sub forma de listă de adiacență.

Etapele care au fost urmate în vederea rezolvării proiectului sunt preprocesare imagine, detecție linii (muchii), detecție cercuri (noduri), recunoașterea cifrelor (id-ul fiecărui nod).

În ceea ce privește baza de date: se va crea o bază de date prin scanarea/fotografierea mai multor desene cu grafuri.

Alegerea acestui proiect a avut la baza solide argumente în ceea ce privește acumularea unor noi cunoștințe, aprofundarea cunoștințelor dobândite în anii anteriori, precum și interconectarea lor. Aceasta aplicație este foarte folositoare pentru digitalizarea grafurilor în scopul ușurării manipulării datelor și de asemenea aplicarea unor algoritmi pe rezultatele obținute.

# Chapter 2

# Studiu bibliografic

## 2.1 Metode de rezolvare

Un urma studiului "Optical Graph Recognition"[6] realizat de cei de la University of Passau,Passau, Germany care se bazează pe recunoașterea optica a grafurilor,au descoperit ca acest proces consta de fapt în 4 faze care trebuie urmate, în urma cărora, graful optic va fi digitalizat.Cele 4 faze sunt prezentate mai jos.

Procesarea: scopul aceste prime faze este acela de a separa pixelii background de pixelii grafului, de aceea se folosește bina rizarea. Cantitatea de informatie filtrata depinde atât de calitate imaginii cat si de performanta etapelor următoare. Apoi etichetele sunt eliminate din imagine, astfel încât acestea sa nu fie ulterior identificate eronat de etapele următoare ale algoritmului. În prima faza se folosește binarizarea bazata pe un trashold global. După binarizare se aplica o metoda de reducere a zgomotelor care depinde de calitate imaginii.

Segmentarea: aceasta faza are ca si input imaginea binari zara rezultata în urma procesului de procesare. Mai precis, pentru fiecare pixel obiect, este determinat în ce categorie se încadrează acesta (muchie sau nod). Ca și ieșire vom avea o imagine formata din trei culori, cate una pentru fiecare element (background, muchie sau nod). În cazul în care, cercurile au fost desenate uniform,acestea pot fi determinate folosind algoritmu Hough.

Recunoașterea topologiei: aceasta faza primește o imagine ca și input unde toți pixelii sunt clasificați ca și background, noduri sau linii. În imagine, regiunile de pixeli care reprezinta doua noduri, sunt conectate printr-o muchie. O regiune continua de pixeli muchie poate corespunde mai multor muchii, dacă muchiile se intersectează, așadar, astfel de situații sunt inabordabile.

Postprocesarea: în faza de postprocesare se utilizează structura topologica de intrare. Sarcinile posibile sunt atribuirea coordonatelor la Vârfurile obținute în faza de segmentare, atașarea etichetelor obținute în faza de procesare, recunoașterea direcțiilor de margine, asignarea culorilor, și transformare în formate de fișiere.

## 2.2 Avantaje și dezavantaje

Ca orice alta metoda de dezvoltare a unui algoritm, acesta metoda de recunoaștere a grafurilor are avantaje și dezavantaje.

Un prim avantaj, foarte important, ar fi acela ca, datorita pașilor foarte preciși și generici pe care algoritmul ii urmează, detectarea corecta a nodurilor, muchiilor și a relațiilor dintre ele este desul de precisa, precizia depinzând de corectitudinea desenării grafului. Totodată acest algoritm recunoaște atât grafurile proiectate pe calculator, dar și grafurile desenate de mana pe hârtie ceea ce reprezinta un avantaj major în ceea ce privește flexibilitate acestuia. Independenta de dimensiunea și culoarea nodurilor și a muchiilor reprezinta un alt avantaj major.

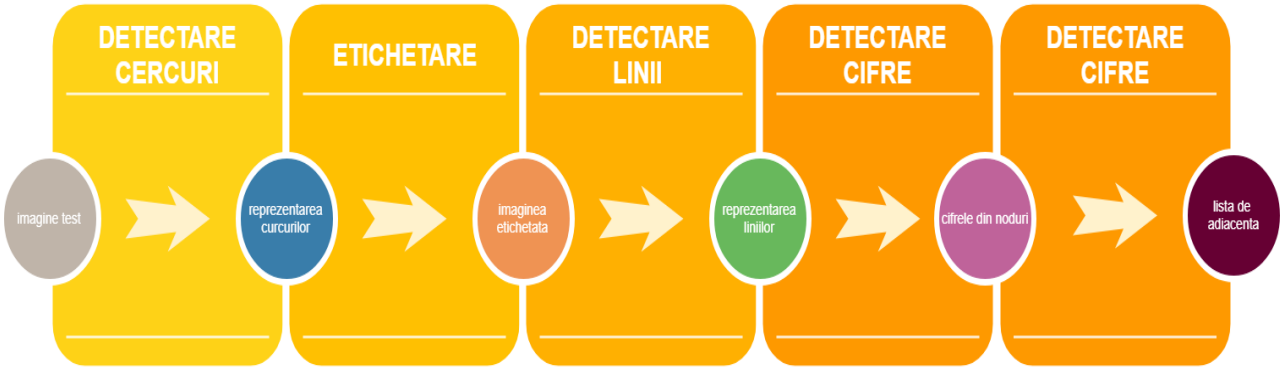
Printre dezavantajele acestui algoritm se număra faptul ca nu poate detecta muchiile care se încrucișează dar nici muchiile care nu sunt drepte. Totodată, dacă se va utiliza o metoda de proiectare a grafului care presupune utilizarea unor margini prea groase, algoritmul va da greș. Nici detectarea informațiilor din interiorul fiecărui nod nu este prevazuta în rezolvarea acestei probleme ceea ce reprezinta un punct slab al acestei abordări însa poate fi ușor îmbunătățit.

# Chapter 3

# Metoda propusa

## 3.1 Schema bloc

O schema bloc care sugerează modul în care se face prelucrarea imaginilor este prezentata în figura de mai jos.




## 3.2 Descrierea soluției

Soluția propusa este una complexa care trece imaginea prin mai multe etape. O scurta prezentare a metodei utilizate se poate deduce din următoarea secventa de cod surprinsa din cadrul funcției main:

```
while (openFileDlg(fname))
        {
                Mat src, binarizedImg, edgeImg, circleImg, linesImage, imagineEtichetata
                src = imread(fname, CV_LOAD_IMAGE_GRAYSCALE);
                binarizedImg = binaryImage(src);
                circleImg = houghCirclesFuncionImage(src.clone());
                imshow("Real Circles", circleImg);
                imagineEtichetata = etichetare(binarizedImg.clone());
                imshow("Labeled Image", imagineEtichetata);
                vector<Mat>::iterator it;
                vector<Vec4i> edges = houghLinesFuncion(binarizedImg.clone());
                setEdges(edges);
                auto objects = objectsFromImage(src.clone());
```

```
            printf("Nr Objects: %d\n", objects.size());
            auto digits = detectDigitImages(objects, 50);

            bindCircleInfo();
            digitalizeGraph(binarizedImg.clone());
            printAdjacencyMatrix();
            convertAdjacencyMatrixToAdjacencyList();

            waitKey();
    }
```

### 3.2.1 Detectarea cercurilor

Etapa de detectare a cercurilor are la baza algoritmul Hough de detecție a cercurilor. Pe lângă aplicarea algoritmului, tot în aceasta etapa se va popula vectorul care păstrează nodurile. Prin aceasta populare a nodurilor se înțelege crearea unei structuri de date care conține un punct. Acest punct este centrul cercului.

Înainte de aplicarea algoritmului Hough de detecție a cercurilor, a fost nevoie de o prelucrare a imaginii prin aplicarea unui filtru Gaussian cu scopul de a blura imaginea.

### 3.2.2 Etichetarea

Etichetarea este o etapa importanta din flow-ul proiectului deoarece face distincția intre elementele grafice(cercuri, linii) și informațiile din fiecare cerc. Am ales aceasta metoda pentru a face un triaj din numărul total de obiecte aflate în imagine.

### 3.2.3 Detectarea liniilor

Precum detecția cercurilor, detecția liniilor se face utilizând algoritmul de detecție a liniilor HoughLineP implementat în OpenCV. O etapa post-mergătoare a fost eliminarea liniilor care erau duplicate. Din varii motive, algoritmul detecta mai multe linii care erau apropiate intre ele. Pentru a alege muchia reala am apelat la calcularea centrului fiecărei linii. După compararea centrelor, am șters liniile duplicate, astfel realizându-se detecția și procesarea muchiilor.

### 3.2.4 Detectarea cifrelor

Etapa de detectare a cifrelor a avut o etapa premergătoare de a extrage obiectele din imagine. Am ales aceasta abordare deoarece astfel s-a redus semnificativ regiunea noastră de interes.

După aceasta pre-procesare a imaginii am obținut obiectele asupra cărora vom aplica algoritmul de identificare a cifrelor. Pentru a eficientiza cat mai mult procesul de character recognision, am ales sa binarizam imaginea și apoi sa număram numărul de pixeli negrii. Am aplicat algoritmul KNN doar asupra acelor obiecte care aveau un numai mai mare de pixel decât un prag stabilit de noi.

Detecția caracterelor și implicit a cifrelor are la baza algoritmul K-Nearest Neighbour.

Pe lângă detectarea numerelor, tot la acest pas am decis sa facem corelația intre cerc și informație. Practic avem o structura de date compusa dintr-un punct și o valoare care reprezinta structura definitorie pentru nod.

### 3.2.5 Digitalizare

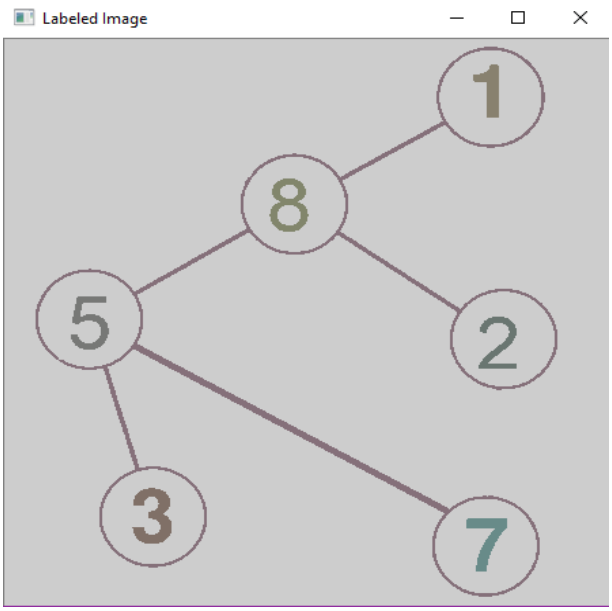
Procesul denumit generic digitalizare are mai multe etape. Însa cea mai importanta este crearea legăturii dintre liniile găsite la pasul anterior și cercurile. Astfel, după aceasta etapa, programul identifica toate elementele și creaza legăturile- muchiile intre noduri. Etapele ulterioare constau în crearea matricei de adiacenta, iar mai apoi crearea listei de adiacenta a grafului în discuție.

## 3.3 Imagini intermediare

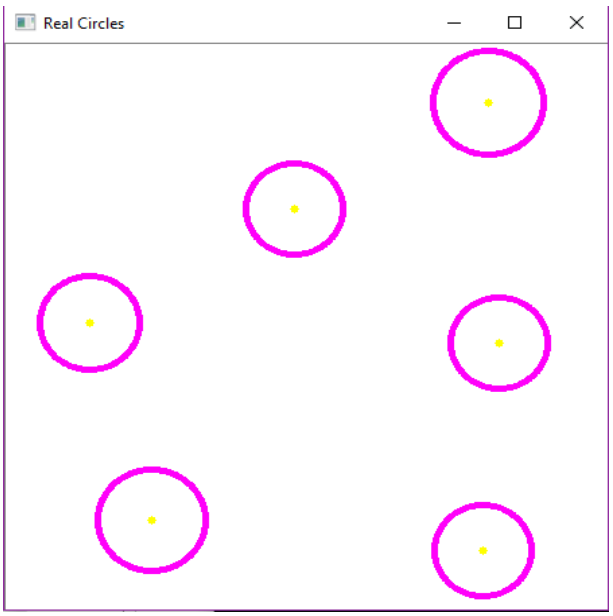
În dezvoltarea acestui algoritm am fost nevoiți sa folosim o mulțime de pași intermediare de testare și verificare. Pentru a reuși sa realizam proiectul în stadiul în care este acum, am folosit diferite metode de detectare a erorilor: debugger, afișarea valorilor în consola, tool-ul de ImageWatch precum și afișarea imaginilor intermediare.

Dintre imaginile afișare în stadiile intermediare fac parte

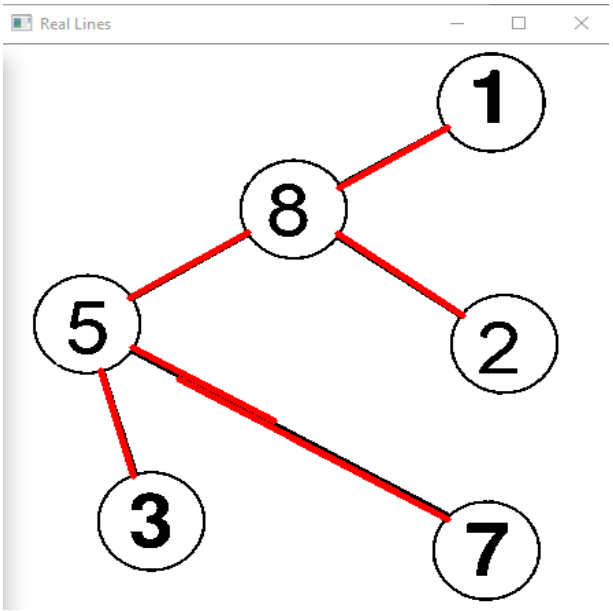
- Afișarea imaginii etichetate. Etichetarea a făcut distingerea intre cifrele din noduri și celelalte elemente: linii, cercuri.




- Afișarea imaginii care reprezinta identificarea cercurilor și centrele lor.




- Afișarea imaginii care reprezinta identificarea liniilor.



- Afișarea imaginii care reprezinta identificarea informațiilor din noduri.

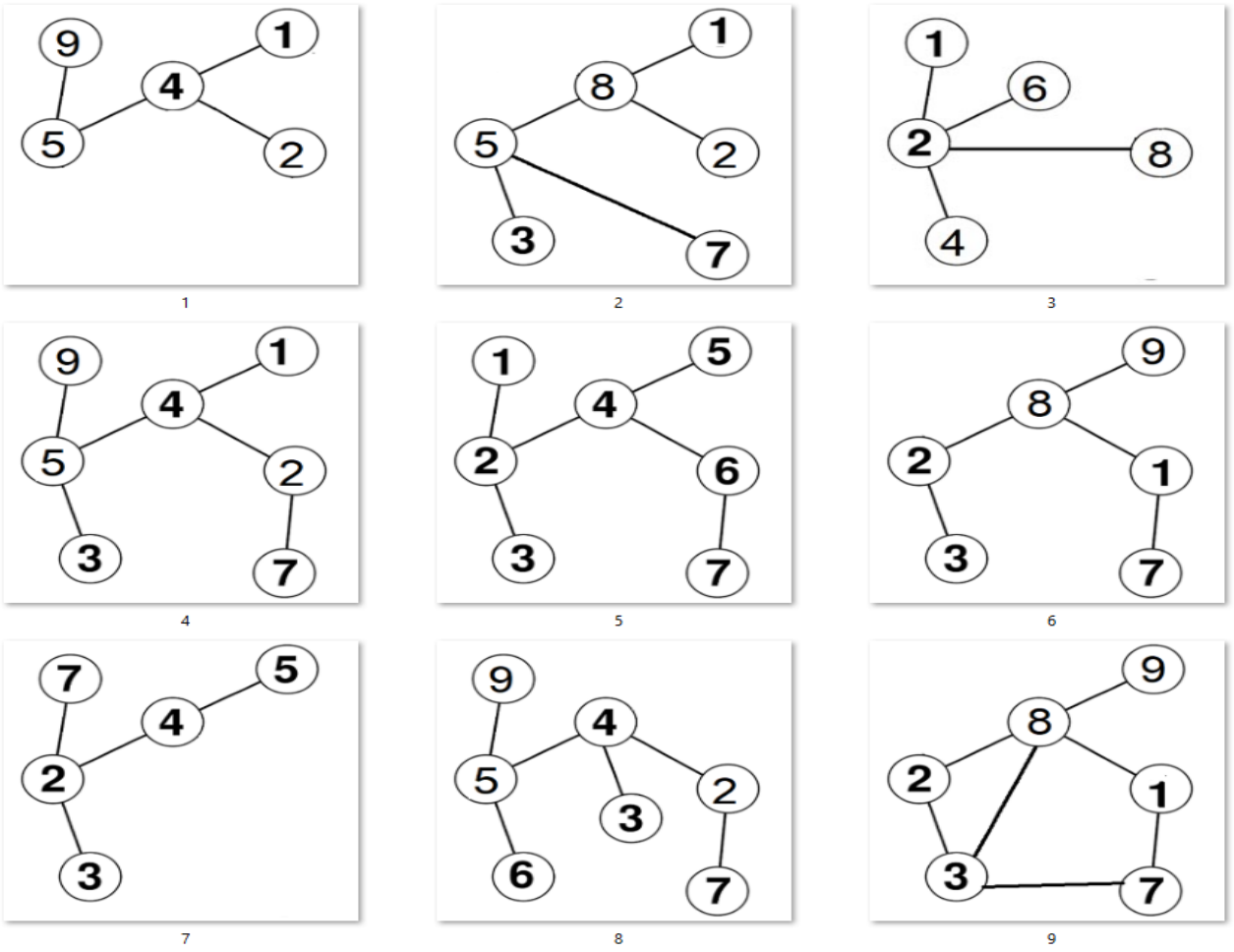
# Chapter 4

# Rezultate experimentale

## 4.1 Imagini

Imaginile pe care le-am folosit ca baza de date pentru test în acest proiect sunt imagini de dimensiuni egale care conțin diferite reprezentări ale unor grafuri conexe. Aceste grafuri conțin un diferit număr de noduri și muchii, unele conțin cicluri sau sunt simple lanțuri. Prin utilizarea acestor imagini am încercat sa surprindem cat mai bine eficienta acestui proiect de digitalizare a grafurilor.

Baza de date a fost compusa din 10 imagini ale căror rezultate sunt corecte și verificate. Câteva exemple din cele 10 imagini găsiți în figura de mai jos:

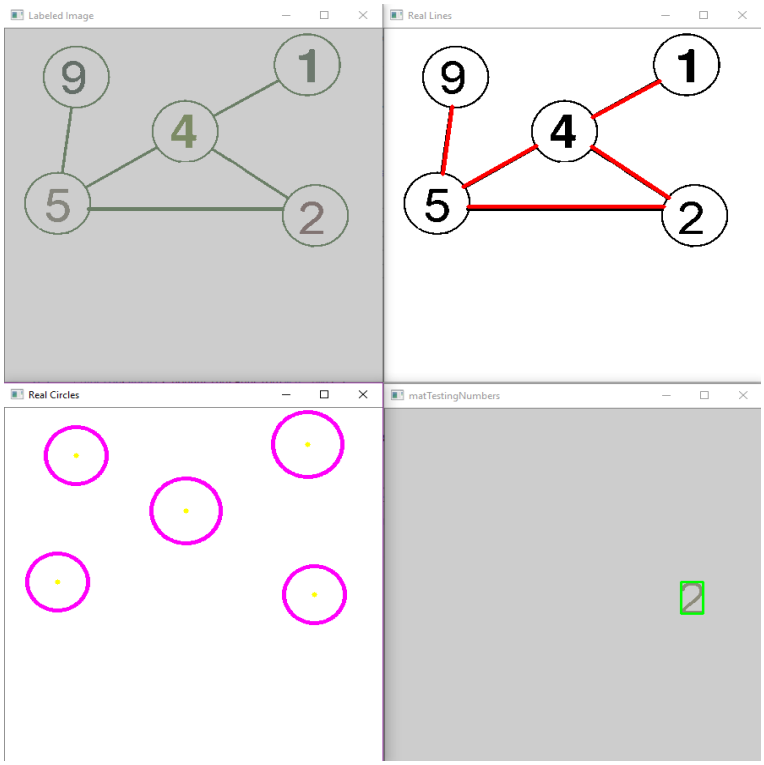


Pentru o buna funcționare a proiectului și obținerea rezultatelor așteptate, este nevoie de respectarea unor parametrii ai imaginilor, și anume: imaginile de test sa aibă dimensiunea de 447px x 447px , iar pentru o buna recunoaștere a cifrelor, acestea trebuie sa aibă dimensiunea aproximativa a fontului Arial, dimensiune 45, fără a fi bold sau italic.

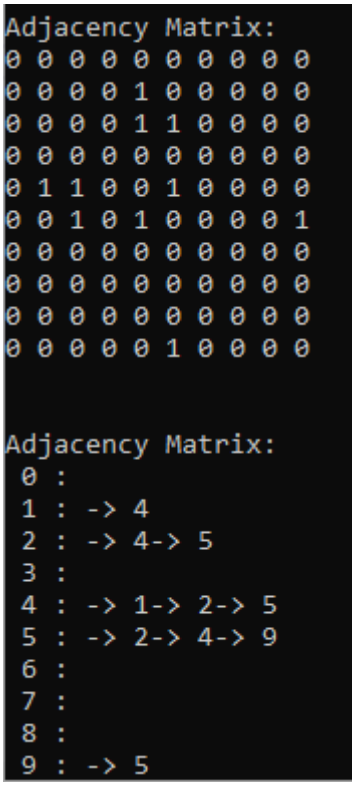
Soluția funcționează corect pe baza imaginilor de test puse la dispoziție. Cu toate acestea am observat ca poziția elementelor în imagine este importanta. Calculându-se densitatea de pixeli, algoritmul va fi sensibil în ceea ce privește detecția muchiile. Estimativ, dintr-o imagine nou introdusa de utilizator, care sa respecte toate cerințele, rata de detecție a muchiilor va fi de 80%. Cifrele sunt identificare corect și nu exista confuzii intre ele. Detecția cercurilor are rezultatele așteptate în proporție de 97%.

Rezultatele afișate vor conține rezultate intermediare sub forma imaginilor și rezultate finale sub forma de text afișate în consola

1. imaginea cu etichetarea fiecărui obiect în culori din aceeași gama
2. imaginea cu liniile detectate de către algoritm. Aceasta imagine va avea liniile detectate colorate cu rosu
3. imaginea cu cercurile detectate. Se va prezenta conturul lor împreuna cu centrul
4. imaginea care prezintă gradual concomitent cu afișarea în consola a detecției caracterelor din fiecare cerc.
5. matrice de adiacenta pe baza grafului ales
6. lista de adiacenta a nodurilor din graf




Rezultate intermediare




Rezultate finale

## 4.2 Evaluare numerica

În ceea ce privește evaluarea numerica, metoda propusa de noi funcționează conform așteptărilor utilizând imaginile de test puse la dispoziție în folderul Images. În mod normal, dacă se respecta indicațiile cu privire la dimensiunea imaginii, grosimea liniilor, forma cercurilor și fontul cifrelor, proiectul ar trebui sa aibă un rezultat favorabil, lista de adiacenta rezultata fiind în concordanta cu imaginea gradului inițial.

## 4.3 Parametrii metodelor

Referitor la parametrii metodelor, aceștia nu ar trebui modificați deoarece sunt aleși în asa fel încât sa funcționeze corect. A fost nevoie de ajustarea lor pentru a ajunge la rezultatele așteptate. Acesta parametrii se refera de parametrii metodelor predefinite din OpenCV cum ar fi
*void HoughLinesP(InputArray image, OutputArray lines, double rho, double theta, int threshold, double minLineLength=0, double maxLineGap=0 )*
sau
*void Canny(InputArray image, OutputArray edges, double threshold1, double threshold2, int apertureSize=3, bool L2gradient=false ).*

Daca vorbim despre parametrii algoritmului sub forma unor date de intrare, nu este nevoie de asa ceva. Singura informație pe care trebuie sa o furnizeze utilizatorul este o imagine de test.

# Chapter 5

# Concluzii și dezvoltări ulterioare

Proiectul realizat și-a atins scopul, acela de a digitaliza grafurile. Astfel, având la dispoziție o imagine cu un graf neorientat am reușit crearea unui algoritm pe baza căruia se poate digitaliza graful din imaginea inițiala. Digitalizarea unei astfel de imagini înseamna crearea și popularea unei structuri de date cu datele din imagine. Structura pe care am ales sa o folosim este o lista de adiacenta, aceasta fiind specificarea proiectului. Totuși, în crearea unei liste de adiacenta, am creat mai întâi o matrice de adiacenta. Dezavantajul matricii este faptul ca trebuie alocata memoria în prealabil, pe când, în cazul listei se va aloca dinamic.

În ceea ce privește posibilele dezvoltări ulterioare acestea se pot materializa pe mai multe direcții:

- extinderea proiectului actual pentru a fi capabil sa proceseze o gama mai larga a informațiilor din noduri. În momentul de fata, se face procesare doar pentru cifre din intervalul 1-9.
- realizarea posibilității de digitalizarea a grafurilor orientate.
- implementarea proiectului în vederea procesării imaginilor de dimensiuni diferite, fonturi diferite. Practic posibilitatea de a putea procesa orice tip de imagine.
- atașarea unui modul ce permite digitalizarea automata prin conectarea unei camere și obținerea rezultatului în timp real.

# Bibliografie


1. P. Hough, “Method and means for recognizing complex patterns”, US patent 3,069,654, 1962.

2. R. O. Duda and P. E. Hart, ”Use of the Hough Transformation to Detect Lines and Curves in Pictures,” Comm. ACM, Vol. 15, pp. 11–15, 1972.

3. https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html?highlight=edge

4. http://users.utcluj.ro/~rdanescu/srf/lab_11r.pdf

5. https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/hough_circle/hough_circle.html

6. https://www.infosun.fim.uni-passau.de/~chris/down/OpticalGraphRecognition.pdf?fbclid=IwAR31jJ6oNmsoeoQEoP6PgT66z8TdGZ5AJ8bV1OIA148yHp2NmUE9MUNQqWo

7. https://sci-hub.tw/10.1007/s001380050054?fbclid=IwAR0G_F-heHfJkyycsU-JDAzGHZbr_tDAYjsiXrUlEGh149mWTww0QhhXBaM