Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем

АНАЛІЗ ЗОБРАЖЕНЬ ЗА ДОПОМОГОЮ КЛІТИННИХ АВТОМАТІВ

Текстова частина до курсової роботи
за спеціальністю „Інженерія програмного забезпечення” 121

Керівник курсової роботи
к.ф.-м.н., доц. Жежерун О.П.

_____
(підпис)

“___“_____2020 р.

Виконав студент Кривошея М.І.

“___“_____2020 р.

Київ 2020

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ
Зав.кафедри мультимедійних
систем, к.ф.-м.н., доц.
_____О. П. Жежерун

„____”_____2019 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

студенту Кривошеї Михайлу Ігоровичу факультету інформатики 3-го курсу
ТЕМА Аналіз зображень за допомогою клітинних автоматів
Вихідні дані:

-
-

Зміст ТЧ до курсової роботи:

Індивідуальне завдання
Вступ
1 Огляд методів аналізу зображень
2 Розробка схеми алгоритму для фільтрації зображень на основі КА
3 Розробка програми реалізації алгоритму фільтрації зображень за допомогою КА та NVIDIA CUDA
Висновки
Список літератури
Додатки (за необхідністю)

Дата видачі „___” _____ 2019 р. Керівник _____
(підпис)
Завдання отримав _____
(підпис)

Календарний план виконання курсової роботи

Тема: Аналіз зображень за допомогою клітинних автоматів

Календарний план виконання роботи:

№ п/п	Назва етапу курсового проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	01.11.2019	
2.	Огляд технічної літератури за темою роботи.	До 15.11.2019	
3.	Виконати аналіз сучасних методів фільтрації зображень	До 25.11.2019	
4.	Розробка алгоритму для фільтрації зображень на основі КА	До 27.12.2019	
5.	Програмування розробленого алгоритму	До 15.01.2020	
6.	Застосування розробленого алгоритму до фільтрації шумів на зображеннях	До 14.02.2020	
7.	Виконання порівняльного аналізу результатів прогнозування, отриманих за допомогою розробленого алгоритму на основі нейромережі, та результатів, отриманих за допомогою регресійних моделей.	До 30.03.2020	
8.	Написання пояснювальної роботи.	До 21.04.2020	
9.	Створення слайдів для доповіді та написання доповіді.	До 23.04.2020	
10.	Аналіз отриманих результатів з керівником, написання доповіді та попередній захист курсової роботи.	До 27.04.2020	
11.	Корегування роботи за результатами попереднього захисту.	До 1.05.2020	
12.	Остаточне оформлення пояснювальної роботи та слайдів.	До 4.05.2020	
13.	Захист курсової роботи (проекту)		

Студент _____ М. І.Кривошея

Керівник _____ О.П.Жежерун

“_____”_____

ЗМІСТ

АНОТАЦІЯ

курсової роботи Кривошеї Михайла Ігоровича

на тему «Аналіз зображень за допомогою клітинних автоматів»

Метою даної роботи є розробити програмний продукт на основі клітинного автомата для фільтрації шумів - мета даної роботи.

У ході виконання роботи було здійснено дефінітивну характеристику основних положень поняття «Клітинний автомат», проведено огляд методів аналізу зображень, розглянуто приклади застосування КА для аналізу зображень, проведено аналіз методів фільтрації шумів на зображеннях, зроблено огляд переваг і недоліків INVIDIO CUDA та OPENCL при використанні.. На цьому підґрунті було розроблено алгоритм з використанням клітинного автомата й застосуванням INVIDIO CUDA для фільтрації шумів та створено програмний продукт, що реалізує цей алгоритм.

Ключові слова: клітинний автомат, аналіз зображень, шумозаглушення, INVIDIO CUDA.

Розмір текстової частини до курсової роботи - 35 аркушів, містить рисунків - 12, таблиць - 1, графіків -1.

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ ТА ТЕРМІНІВ

API – інтерфейс програмування застосунків (англ. Application Programming Interface);

CPU – центральний процесор (англ. Central Processing Unit);

CUDA – програмно-апаратна архітектура, що дозволяє виконувати обчислення з використанням графічних процесорів NVIDIA (англ. Compute Unified Device Architecture), що підтримують технологію GPGPU;

Geforce – бренд графічних процесорів і чіпсетів

GPGPU – техніка використання графічного процесора відеокарти для загальних обчислень, які зазвичай проводить центральний процесор (англ. General-purpose graphics processing units);

GPU – графічний процесор (англ. Graphics Processing Unit);

NVIDIA – американська технологічна компанія, розробник графічних процесорів

OPENCL – фреймворк для написання комп’ютерних програм (англ. Open Computing Language)

Quadro – бренд графічних карт NVIDIO

Tesla – американська компанія – виробник електромобілів

КА - клітинний автомат

ПП - програмний продукт

ВСТУП

Сьогодення розвитку засобів аналізу і обробки інформації характеризується масштабним впровадженням різних алгоритмів.

Аналіз зображень за допомогою клітинних автоматів – цікава задача. Водночас вона є перспективною, оскільки не існує досліджень, які б не можна було змоделювати за допомого клітинних автоматів [1].

Задачі аналізу і обробки зображень постають все в більш численних галузях науки. Цього потребують великі обсяги даних, що надійшли з найрізноманітніших джерел. Клітинні автомати поширені в багатьох сферах, у тому числі в економіці, фізиці, математиці, соціології. Їх застосовують як у моделюванні біологічної системи, так і в створенні віртуальної соціальної мережі [2].

Методи аналізу та обробки зображень є дуже різноманітними та залежать від завдання дослідження [3]. Немає чітких критеріїв, у якому випадку краще спрацює той чи інший метод, а в якому необхідно шукати нові підходи.

З огляду на вищесказане тема цієї роботи є актуальною.

Робота складається з чотирьох розділів.

Перший розділ присвячений дефінітивній характеристиці поняття КА. У ньому дано визначення клітинного автомата, наведено огляд видів КА та областей їх застосування, вказані переваги КА перед іншими методами обчислень.

У другому розділі наведено огляд методів аналізу зображень, розглянуто приклади застосування КА для аналізу й обробки зображень та приклади методів фільтрації шумів на зображеннях.

У третьому розділі здійснено порівняльну характеристику переваг та недоліків у використанні NVIDIA CUDA та OPENCL.

Четвертий розділ присвячений розробці алгоритму фільтрації шумів на основі КА з використанням NVIDIA CUDA та програмній реалізації цього алгоритму.

Метою даної роботи є розробити на основі КА з використанням NVIDIA CUDA програмний продукт для фільтрації шумів на зображеннях.

Об’єкт дослідження – КА.

Предмет дослідження – фільтрація шумів на зображеннях

Постановка задачі

1. Здійснити дефінітивну характеристику поняття «Клітинний автомат».
2. Зробити огляд методів аналізу зображень та розглянути приклади застосування КА для аналізу й обробки зображень.
3. На основі КА розробити алгоритм для фільтрації шумів на зображеннях.
4. Виконати програмну реалізацію розробленого алгоритму з використанням NVIDIA CUDA.

РОЗДІЛ 1. ДЕФІНІТИВНА ХАРАКТЕРИСТИКА ПОНЯТТЯ «КЛІТИННИЙ АВТОМАТ»

1.1 Визначення КА та основних понять, пов’язаних з ним

Вперше термін клітинний автомат був введений наприкінці 1940-х років Джоном фон Нейманом за пропозицією Станіслава Улама , проте стали популярними лише тоді, коли Джон Хортон Конвей розробив гру «Життя» Це був кінець 1960-х років [4].

На думку Т. Тоффолі та Н. Марголуса [5] клітинні автомати є стилізованими, синтетичними світами, які визначаються простими правилами, подібними до правил настільної гри. Вони мають свій власний вид матерії, який кружляє в своєму власному просторі та часі. Можна лише уявити вражаючу різноманітність цих світів. Є можливість дійсно побудувати їх і спостерігати за тим, як вони розвиваються. Машина клітинного автомата є синтезатором світів. Подібно до органу, він має клавіші та регістри, за допомогою яких можливості інструменту можна приводити до дії, комбінувати та перекомпоновувати, а його кольоровий екран є вікном, крізь яке можна спостерігати за світом, який весь час «грає» [5].

КА є прикладом розподілених систем, заснованих на простих правилах, які дозволяють реалізувати складну поведінку [1].

Клітинний автомат можна визначати, як множину кінцевих автоматів, кожен з яких може знаходитися в одному з можливих станів $\sigma \in Z$, де Z - множина станів кожної клітини [5, 6].

Зміна стану автоматів відбувається згідно правилу переходу f:

$$Z \times Z^{|N|} \to Z$$

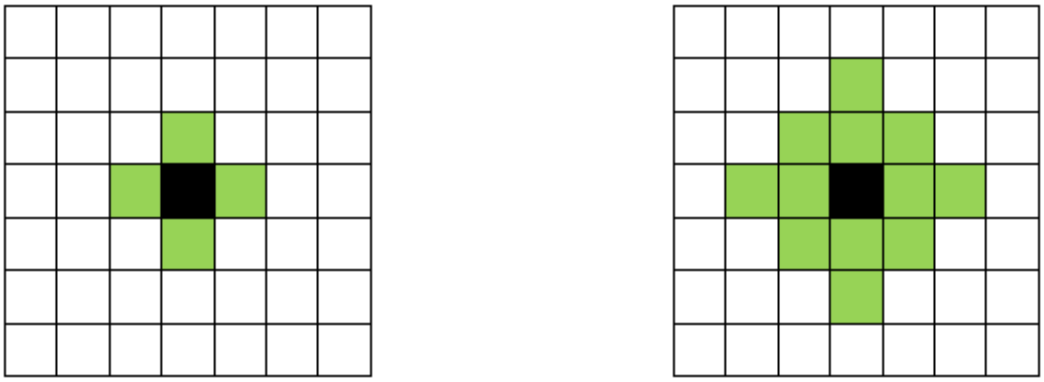
наступним чином:

$$\sigma_{i,j}(t+1) = f(\sigma_{k,l}(t), \sigma_{k,l}\ (t)\ \in N\ ), \qquad (1.1)$$

де N – множина автоматів, які складають окіл.

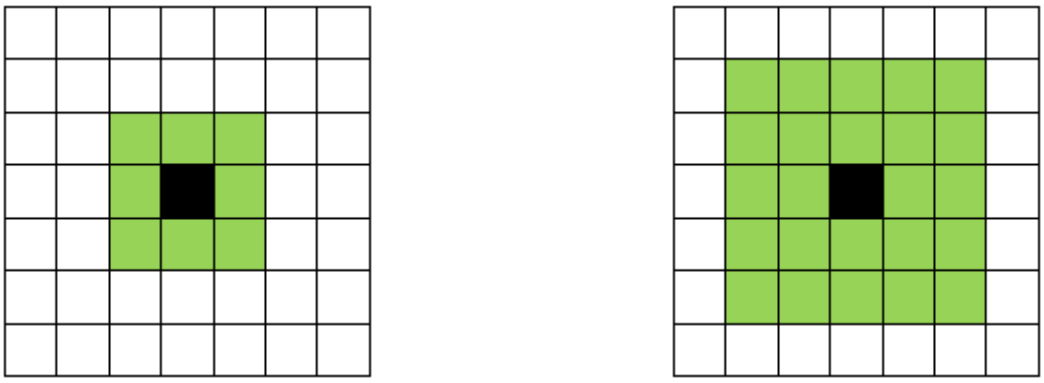
Окіл клітини складається з клітини ядра (центральної клітини) та оточуючих клітин, стани яких визначають її наступний стан. Радіус околу визначається як максимальна відстань від центральної клітини, горизонтально або вертикально, до сусідніх.

Для КА найчастіше використовують сусідство (окіл) Неймана та сусідство (окіл) Мура (див. рисунок 1.1 та рис. 1.2)



а) б)

Рисунок 1.1 - Приклади двовимірного околу фон Неймана: а) першого порядку; б) другого порядку [6]



а) б)

Рис. 1.2 Приклади двовимірного околу Мура: (а) – першого порядку; б) другого порядку [6]

Цифрове зображення розглядається як двовимірний масив пікселів $M \times N$ (див. рисунок 1.3). Кожен піксель може бути

охарактеризований впорядкованою трійкою $(i; j; k)$, де впорядкована пара $(i; j)$ представляє його положення в масиві, k являє собою колір, пов'язаний з цим пікселем. Зображення потім може розглядатися як особлива конфігурація КА, що займає клітинний простір масиву $M \times N$, який визначається зображенням. Кожен піксель зображення являє собою комірку КА і стан клітинки визначається величиною пікселя у зображенні [4].

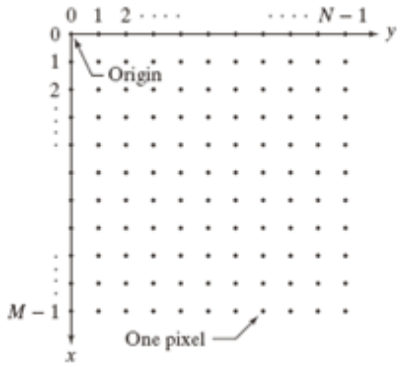


Рис. 1.3 - Піксельне зображення цифрового зображення [4]

У статті [4] вказано, що архітектура КА залежить від розміру моделі, кількості кінцевих станів клітини, кількості сусідніх клітин та їх радіусу, граничних умов і перехідних функцій або правил.

Граничними правилами називають значення, якими будуть заповнені клітини, котрі мають не повний набір сусідів.

У роботі [7] автор визначає чотири наступних типи граничних умов:

- фіксована (крайні клітини з’єднанні з логічним станом нуль\один;
- періодична (крайні клітини прилягають одна до одної);
- адіабатична (крайні клітини реплікують цей стан);
- рефлексивна (дзеркальні стани замінюють крайніми клітинами).

Розглянемо одновимірний КА, довжина якого п’ять клітин з наступними станами: x_1, x_2, x_3, x_4, x_5. Нехай сірі клітини – це крайні клітини сусідніх станів. Типи граничних умов будуть наступними (див. рисунок 1.4):

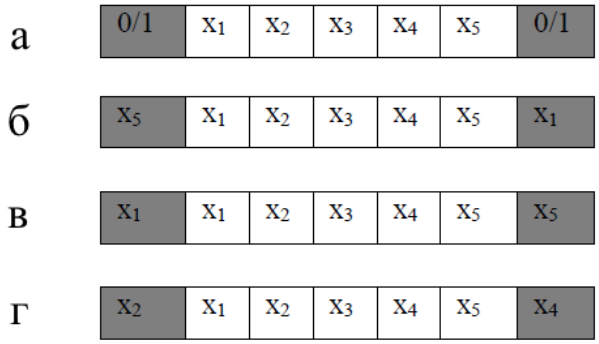


Рис. 1.4 – Типи граничних умов:

а – фіксована; б – періодична; в – адіабатична; г – рефлексивна [7]

Кількість усіх можливих правил переходу визначається числом станів σ та кількістю сусідів m:

$$n = \sigma^{\sigma^m} \qquad (1.2)$$

Зрозуміло, що загальна кількість правил переходу різко зростає навіть при досить малих σ та m (див. таблицю 1.1).

Таблиця 1.1 – Залежність кількості правил переходу від числа станів σ та кількості сусідів m [6]

Кількість станів σ	Кількість сусідів m	σ^{σ^m}	Кількість правил n
2	2	2^{2^2}	16
2	3	2^{2^3}	256
2	5	2^{2^5}	4 296 967 296
2	10	$2^{2^{10}}$	$1{,}797*10^{308}$
5	2	5^{5^2}	$2{,}98*10^{17}$
5	3	5^{5^3}	$2{,}35*10^{87}$
5	5	5^{5^5}	$1{,}91*10^{2184}$
10	2	10^{10^2}	10^{100}
10	3	10^{10^3}	10^{1000}
10	5	10^{10^5}	10^{100000}

Для двовимірного КА з двома станами й сусідством Мура існує 2^{512} правил і лише 2^9 серед них є лінійними, а віднявши базові та нульове правило, матимемо 502 лінійних правил [8].

1.2 Види КА

З огляду на те, що задача класифікації КА залишається відкритою, є різні підходи щодо їх класифікації, зокрема:

- у залежності від розмірності решітки, розрізняють одно-, дво-, тривимірні і т. д. КА;
- у залежності від кількості станів КА бувають бінарні, трійкові і т. д.;
- у залежності від синхронізації оновлення клітин КА бувають синхронні та асинхронні (в першому випадку клітини оновлюються одночасно, у другому – кожна клітина робить це незалежно від інших).

Серед найбільш поширених класифікацій КА є наступні:

- за основу класифікації КА беруть використання автоматизованого знаходження локальних структур, виділяють три області: впорядковані системи, хаотичні системи, системи зі складною поведінкою [9];
- застосовують ймовірнісний метод класифікації [10], виділяють відповідно наступні класи: строго сприятливі, сприятливі в середньому, нейтральні в середньому, несприятливі в середньому, строго несприятливі (класи виділяють на основі ймовірнісного переходу випадкової клітини у стан 1на першому кроці еволюції при випадкових початкових умовах;
- класифікація КА за типами їх еволюції, - Стівеном Вольфрамом виділено наступні чотири класи [1]:

– до першого класу віднесено КА, де еволюція системи закінчується переходом всіх клітин поля в однаковий стан. Результатом еволюції майже всіх початкових станів є швидка стабілізація стану та

його гомогенність, при цьому будь-які випадкові конструкції в таких правилах швидко зникають;

– до другого класу віднесено КА, де існує багато кінцевих станів, проте всі вони складаються з набору простих структур, які залишаються незмінними або повторюються через певну невелику кількість кроків. Більшість випадкових структур у початкових умовах швидко зникає і залишаються лише деякі. Локальні зміни в початкових умовах мають локальний характер на подальший хід еволюції системи;

– до третього класу віднесено КА, де поведінка виглядає хаотичною і є складною. Структури змінюються з певним періодом і залежать від початкової конфігурації. Будь-які стабільні структури, що виникають, майже одразу знищуються шумом, що їх оточує. Локальні зміни в початкових умовах надають широкий, невизначений вплив на хід еволюції усієї системи;

– до четвертого класу віднесено КА, де суміш хаосу і порядку: породжуються локальні структури, що переміщуються та взаємодіють між собою у дуже складний спосіб. Результатом еволюції майже всіх правил є структури, наслідком взаємодії яких є формування локальних стійких структур, які здатні виживати впродовж тривалого часу. В результаті еволюції правил цього класу можуть з’являтися деякі послідовності першого класу. Локальні зміни в початкових умовах мають широкий, невизначений вплив на хід усієї еволюції системи.

Досить складним є віднесення КА до якогось із класів, оскільки не вказано, за яких початкових умов очікується вказана вище поведінка. Здійснити класифікацію клітинного автомата за оптимальний проміжок часу практично неможливо у зв’язку з необхідністю виконання великого обсягу обчислень та а рахунок складного аналізу [1].

1.3 Переваги КА перед традиційними методами обчислень

Серед переваг клітинних автоматів перед традиційними методами обчислень є наступні [11]:

- простота реалізації (встановлено, що системи, основою яких є КА, можуть бути легко впроваджені, оскільки кожна клітина зазвичай працює за декількома простими правилами);
- КА є одночасно паралельними та просто обчислювальними;
- КА є розширюваними (можемо розширити прості правила, використовуючи деякі нові методи обчислення);
- підтримує категорії n-розмірів та m-міток, де кількість міток не збільшує час або складність обчислень;
- КА є більш інтерактивними (користувачі можуть вносити виправлення та модифікації в будь-коли під час операцій).

1.4. Області застосування клітинних автоматів та постановка задачі

Основним напрямок дослідження КА є алгоритмічна розв’язність окремих задач. У наш час сліди КА можна знайти в різних аспектах науки від моделювання біологічної системи до створення віртуальної соціальної мережі [4].

Обробка зображень за допомогою КА – актуальний та перспективний напрямок, його особливість полягає в розгляді зображень, як системи, що складається з простих компонентів (пікселів), а поведінка кожного компонента виходить і реформується згідно поведінки сусідів і їх попередньої поведінки [4].

КА досягли значних результатів у багатьох областях аналізу та обробки зображень, зокрема: фільтрація шумів [12,13] , відновлення та розпізнавання образів та текстів, згладжування, контурування [4]. Серед актуальних та цікавих конкретизованих задач для вирішення яких

застосовують КА, є наступні: аналіз рентгенівських знімків, аналіз супутникових зображень, ідентифікації мінних полів [14,12].

Аналіз та обробка зображень за допомогою КА є перспективним напрямком. Його особливість полягає в розгляді зображень, як системи, що складається з простих компонентів, а поведінка кожного компонента змінюється згідно поведінки сусідів і їх попередньої поведінки. Завдяки заданню певних правил поведінки компонентів КА досягли значних результатів у різних областях аналізу й обробки зображень, проте застосування цих досліджень й досі залишається лише для конкретизованих задач [4].

1.5. Висновки до розділу 1

В даному розділі здійснено дефінітивну характеристику основних положень дослідження. Розглянуто області застосування КА. Спираючись на отримані дані, зроблено постановку задачі для дослідження.

РОЗДІЛ 2. ОГЛЯД МЕТОДІВ АНАЛІЗУ ЗОБРАЖЕНЬ

2.1 Методи аналізу зображень

2.1.1 Виділення та аналіз зв’язних областей

При аналізі дискретного цифрового зображення розглядають два сусідства (див. рисунок 1.1) й два відповідні їм види зв’язності:

- сусідство по «хресту» і 4-зв’язність;
- сусідство по «квадрату» і 8-зв’язність.

Частіше використовується друге з них.

Зв’язною областю зображення вважається така, для якої виконуються наступні дві умови [15]:

- всі точки області мають однакові значення (ознаки, що розглядається);
- між будь-якими двома точками, що належать даній області, існує неперервний шлях, який складається з точок, що також належать даній області й при цьому є сусідніми.

На першому етапі аналізу зображень виконується виділення зв’язних областей або зв’язних контурів. Метод випалювання області застосовують для виділення зв’язних областей. Ідея така: область ,,підпалюється” в одній точці. Після цього кожна така точка в свою чергу ,,підпалює” всіх своїх сусідів, які мають ту ж саму яскравість. ,,Спалені ” точки повторно не підпалюються. Згідно з означенням зв’язної області, всі точки зв’язної області будуть залучені до цього процесу [15].

2.1.2 Методи виділення геометричних примітивів на основі класичного перетворення Хафа

Для пошуку прямих ліній на бінарному зображенні застосовують класичне перетворення Хафа .

Розглянемо рівняння прямих:

$$Y = kX + b \qquad (1)$$

$$X\cos(\theta) + Y\sin(\theta) = p \quad (2)$$

Оскільки для характеристики прямих на площині необхідно і достатньо два параметри, то простір параметрів буде двовимірний. Саме параметри рівняння (2) використовують у перетворенні Хафа. Суть цього методу базується на тому, що необхідною та достатньою умовою перетину двох синусоїд у просторі параметрів $E = (X, Y)$ є належність одній прямій у початковому зображенні точок, що їх породжують [15] .

Аналогічно метод Хафа працює й для будь-яких інших кривих площини, які задано параметричним рівнянням. Має місце узагальнення методу голосування контрольних точок (на випадок кривих, які не можуть бути описані в аналітичній формі. [15]. Така найбільш загальна його форма відома як узагальнене перетворення Хафа.

2.1.3 Методи виявлення об’єктів на зображеннях, заданих еталонами

Нехай для кожного класу об’єктів відомі одне або декілька еталонних зображень.

Кореляційне виявлення порівнює зображення з еталоном. Саме зображення розглядають як двовимірну функцію яскравості, при цьому вимірюється відстань між зображеннями або визначається міра їх близькості. За умови двовимірного випадку, об’єкти розуміються як вектори, а відстань між ними – це Евклідова відстань [15].

Розглянемо випадок, коли дано n еталонних зображень $\{g_i\}, i = 1, \dots, n$ Встановимо відповідність: кожному $i-$му зображенню співставимо g_i клас. При появі нового фрагмента зображення f , йому ставиться у відповідність деякий клас j. Як варіант, це може відбуватися за методом мінімальної відстані до відповідного еталону.

Лінійні розміри еталону та зображення впливають на критерій виявлення – це є недоліком методу. Навіть у випадку найпростіших перетворень яскравості, критерій виявлення не є інваріантним стосовно них. Важливі недоліки кореляційних схем виявлення мають місце і при наявності

геометричних спотворень поточного зображення, яке порівнюється з еталоном [15].

Ще один приклад цього методу - узгоджена фільтрація. Застосовується для аналізу бінарних зображень. Це один з видів віконної фільтрації. Для виявлення об’єкта створюється вікно. Його форма визначається формою шуканого об’єкта (форми мають бути однакові). У створеному вікні призначається центральний піксель і здійснюється прохід вікном по вхідному зображенню. Число ненульових елементів вікна підраховується в кожному з можливих положень вікна. За умови, що це число більше від деякого порогу (рангу), приймається рішення про виявлення об’єкта в цій точці. В центральний піксель поточного вікна на вихідному зображенні виставляється значення 1 [15].

2.2 Метод КА аналізу та обробки зображень

Незважаючи на високий рівень розвитку сучасної комп’ютерної техніки, на теперішній час залишається багато практичних задач, вирішення яких виявляється дуже проблемним. До них належать задачі автоматизованого аналізу зображень. Це зумовлено перш за все складністю формалізації процесу сприйняття образів і зображень. Тому все ще немає «універсального» математичного чи технологічного підходу, який дозволяв би конструктивно розробляти методи та алгоритми, які б ефективно здійснювали процес. Проте, для деяких частинних випадків, коли математичні моделі виявляються відповідними для тієї чи іншої практичної задачі, вдається отримати гідні результати [16].

Застосування КА ще й досі залишається лише для вирішення конкретизованих задач [4]. Розглянемо деякі приклади таких задач.

2.2.1 Виявлення контурів на основі КА

Контуром називатимемо межу між двома різними областями зображення, які відрізняються інтенсивністю, кольором або текстурою. Для їх виявлення ефективним є метод на основі КА. Виявлення контурів – це основний засіб при вирішенні задачі виявлення ознак [керівник]. Ідея виявлення контуру бінарного зображення полягає у застосуванні оптимальних лінійних правил [7].

2.2.2 Задача фільтрації шумів на зображеннях. Приклади методів фільтрації

Задача пошуку та усунення шумів на зображенні є комплексною. Її вирішення передбачає аналіз зображення на предмет виявлення шумів та водночас обробку цього зображення з метою шумозаглушення. При цьому потрібно забезпечити збереження та мінімальне спотворення корисної інформації зображення. .

Цифрові зображення можуть бути чутливими до різних типів шумів. Ці шуми виникають у залежності від способу отримання зображень, технологій передачі інформації, методів оцифрування даних [4].

Процес усунення різних видів шумів на зображеннях називається фільтрацією.

Медіанний фільтр – це метод нелінійної обробки сигналів, розроблений Тьюкі [17]. Характеризується найбільшою ефективністю при фільтрації шумів.

Нехай $a_1, a_2, \dots, a_{2k+1}$ - набір , що містить непарну кількість чисел. Переставимо ці числа в порядку незростання (неспадання). Медіаною набору є число, що знаходиться на $(k+1)$ – му місці. Позначають його так: $m(a_1, a_2, \dots, a_{2k+1})$. У випадку, якщо набір містить парну кількість елементів, то медіаною будемо називати середнє арифметичне чисел a_k та a_{k+1}.

Розглянемо нескінчену в обидва боки послідовність чисел

$\dots, x_{-2}, x_{-1}, x_0, x_1, x_2, \dots$.

Зафіксуємо натуральне число k. Нова послідовність $\dots, y_{-2}, y_{-1}, y_0, y_1, y_2, \dots$ називається отриманою з вхідної шляхом медіанної фільтрації, якщо $y_n = m(x_{n-k}, x_{n-k+1}, \dots, x_n, \dots, x_{n+k})$. Число $2k+1$ має назву ширина вікна фільтрації [18,19].

Одновимірний медіанний фільтр – це ковзаюче вікно, яке охоплює непарну кількість елементів зображення. Центральний елемент вікна замінюється медіаною всіх елементів у вікні. Вікно переміщується вздовж сигналу, що фільтрується, і обчислення повторюються [20].

Позитивною характеристикою медіанного фільтру є простота його структури. Цей фільтр не впливає на ступінчасті та пилкоподібні функції, добре пригнічує випадкові шумові викиди звітів і промахи. Застосовуючи двомірне вікно бажаної форми, фільтр можна узагальнити на два виміри. Медіанний фільтр дає можливість розділити об'єкти, які суттєво відрізняються за формою. Він зберігає різкі контури об'єктів [12].

Проте, оскільки медіана суми двох довільних послідовностей не дорівнює сумі їх медіан (нелінійність), то для медіанного фільтру характерна порівняна складність математичного аналізу характеристик. Цей фільтр викликає сплощення вершин трикутних функцій і є менш ефективним у порівнянні з лінійними фільтрами щодо пригнічення білого та Гаусового шумів. За умови збільшення розмірів вікна фільтра відбувається розмиття крутих змін сигналу та стрибків. Недоліки медіанного фільтра можна зменшити, використовуючи медіанну фільтрацію з адаптивною зміною розміру вікна фільтра в залежності від динаміки сигналу та характеру шумів (адаптивна медіанна фільтрація) [21,18].

Видалити шум за умови збереження важливих для подальшого розпізнавання частин зображення - головна проблема аналізу зображень.

Медіанна фільтрація часто застосовується для попередньої обробки цифрових зображень і є найбільш ефективною за умови, що шум на зображенні імпульсного характеру та є обмеженим набором пікових значень на фоні нулів. У результаті застосування медіанного фільтру похилі ділянки

та різкі перепади значень яскравості на зображеннях не змінюються. Для зображень, на яких контури є носіями основної інформації, ця властивість надзвичайно корисна. При медіанній фільтрації зашумлених зображень ступінь згладжування контурів об’єктів безпосередньо залежить від розмірів та форми вікна фільтра. Приклади форм вікна фільтра наведені на рис. 2.1. За умови, що розміри вікна малі, контрастні деталі зображення зберігаються краще, проте в меншій мірі пригнічуються імпульсні шуми і навпаки при великих розмірах вікна. Від специфіки розв’язуваної задачі та форми об’єктів залежить оптимальний вибір форми вікна. Це дуже важливо для завдання збереження перепадів (різких контурів яскравості) у зображеннях [19].

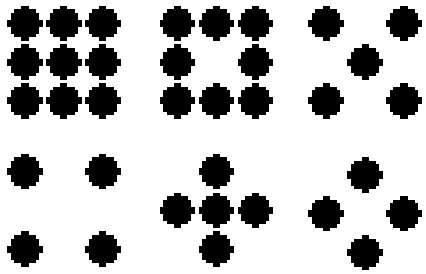


Рисунок 2.1 – Приклад форми вікна [19]

При використанні медіанного фільтру (на два виміри) задаємо розбиття чорно – білої фотографії на маленькі квадрати та зіставляємо до фотографії прямокутну таблицю чисел $x_{i,j}$ рівнів яскравості в кожному з маленьких квадратів. До одержаної таблиці чисел застосовуємо процедуру медіанної фільтрації. Обираємо певну форму вікна (див. рисунок 2.1) і накладаємо її на зображення. Значення яскравості в центрі вікна замінюємо медіаною чисел $x_{i,j}$ для усіх точок, які потрапили до цього вікна [21].

Розглянемо приклад. Нехай $x_{i,j}$ дорівнює 100, за умови , що i та j обидва кратні до 10 та дорівнює 0 - в противному випадку. Після медіанної фільтрації таке зображення складатиметься лише з нулів.

При обробці реальних зображень дуже темні і дуже світлі точки (пікселі) повинні зникнути (рис. 2.2).

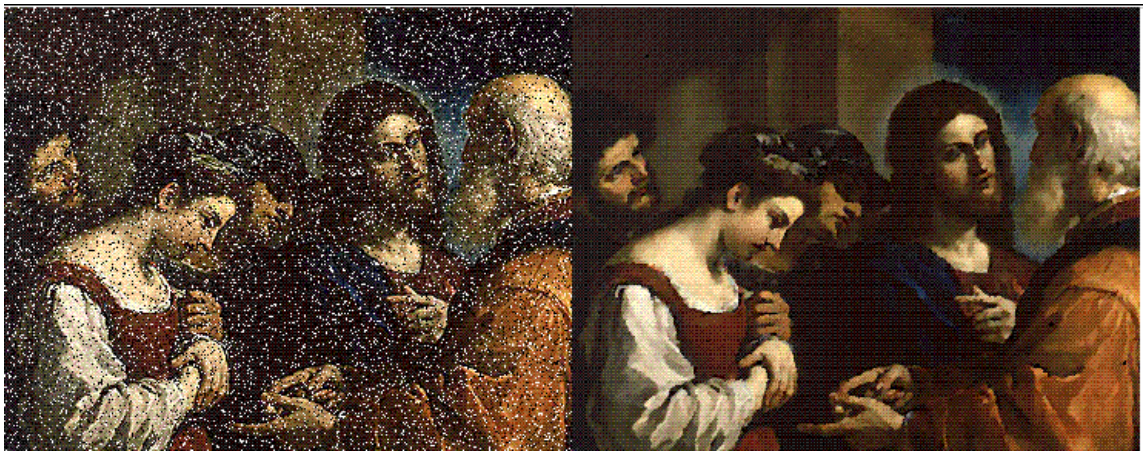


Рисунок 2.2 – Приклад застосування медіанного фільтру [22].

Зауважимо, що при розв’язанні задач шумозаглушення ефективним є медіанний фільтр, оскільки призводить до найменших спотворень контурів виділених об’єктів. Проте, коли рівень шуму перевищує 30%, цей фільтр «замилює» деякі контури та деталі [4].

Для вирішення задачі фільтрації шумів, особливо імпульсних шумів, використовують правило „більшості” [13] і сусідство Мура. Ідея застосування цього правила викладена в [4], а саме: якщо 0 - значення центрального пікселя (чорний , „перець”) або 255 (білий, „сіль ”), то йому присвоюється значення більшості сусідніх пікселів. За умови неможливості визначити більшість, це правило має стан „рівності” і значення центрального пікселя визначається детерміновано або у випадковий спосіб, що означає наступне: або наперед визначено значення якого сусіда буде присвоєно центральній клітинці, або цей сусід буде визначатися у випадковий спосіб кожного разу, як тільки правило набуває стану „рівності” [4]. Кожне з вказаних правил має свій недолік. За умови визначення у детермінований спосіб, можна отримати спотворення границь об’єктів, на яких був знайдений шум; якщо ж у випадковий спосіб визначати, то можна натрапити на інший піксель шуму. Це ще додатково потребуватиме ітерації шумозаглушення (однієї або кількох). Тому в роботі [самого автора]

застосовано додатковий аналіз значень сусідів. У процесі цього додаткового аналізу відбираються ті значення, які зустрічаються частіше, у порівнянні з іншими, і значення для центральної клітини обирається у випадковий спосіб вже з отриманих значень.

2.3. Висновки до розділу 2

У другому розділі було зроблено огляд методів аналізу зображень; наведено приклад використання КА для аналізу зображень; розглянуто задачу фільтрації шумів і наведено приклади методів її вирішення, вказано на їх недоліки у застосуванні. На основі цього для розробки алгоритму для фільтрації шумів на зображеннях було обрано КА.

РОЗДІЛ 3. ПЕРЕВАГИ І НЕДОЛІКИ NVIDIA CUDA та OPENCL ПРИ ЗАСТОСУВАННІ

3.1 Огляд переваг і недоліків NVIDIA CUDA та OPENCL при застосуванні

Розробка CUDA була анонсована разом з чіпом G80 у 2006 році.

CUDA – це програмно-апаратна архітектура, що надає можливість виконувати обчислення з використанням графічних процесорів NVIDIA, які підтримують технологію GPGPU (довільних обчислень на відеокартах) [23].

Вона базується на мові C++, що дає можливість організації доступу до набору інструкцій графічного прискорювача та керування його пам’яттю, за умови організації паралельних обчислень. За допомогою CUDA можна реалізувати алгоритми, що є виконуваними на графічних процесорах відеоприскорювачів Geforce восьмого покоління та старше, а також Quadro та Tesia [24].

До архітектури CUDA належить уніфікований шейдерний конвеєр, який дозволяє програмі, що виконує обчислення загального призначення, задіяти будь-який арифметично-логічний пристрій, який у свою чергу входить до мікросхеми.

Переваги CUDA у використанні наступні [25, 26]:

- інтеграція в єдиний простір із іншими технологіями від NVIDIA;
- підтримка Multi - GPU Programming;
- високий рівень інноватівності, у порівнянні з аналогами;
- порівняно висока ефективність транзакції між пам’яттю CPU та GPU.

Для розробників однією з найбільш вагомих особливостей CUDA є вільний доступ до пам’яті з можливістю побайтової адресації. Графічний процесор, що має колосальну обчислювальну здатність, суттєво розвантажує

CPU, беручи на себе навантаження, які пов’язані з найбільш трудомісткими та складними задачами.

NVIDIA CUDA – найбільш ефективний обчислювальний засіб, який дає ряд переваг перед іншими засобами у практичному застосуванні завдяки найвищій продуктивності, доступності, простоті у вивченні, наявності підтримки практично будь-якою сучасною відеокартою NVIDIA [23].

CUDA має свої обмеження, зокрема: відсутність підтримки рекурсії для функцій, що виконуються, мінімальність ширина блоку в 32 потоки, закритість архітектури CUDA, яка належить NVIDIA [24].

OPENCL - це є відкритий стандарт для паралельного програмування, який пропонує ефективний і переносний спосіб використання можливостей різнорідних обчислювальних багатоядерних платформ (CPU, GPU та інші).

Він містить у собі програмний інтерфейс (API) (для координування паралельних обчислень у середовищі різнорідних процесорів) і кросплатформенну мову, яка використовується в певному обчислювальному середовищі [27].

Головною проблемою OPENCL є низька продуктивність у порівнянні з CUDA, проте з кожним новим релізом драйверів його продуктивність зростає.

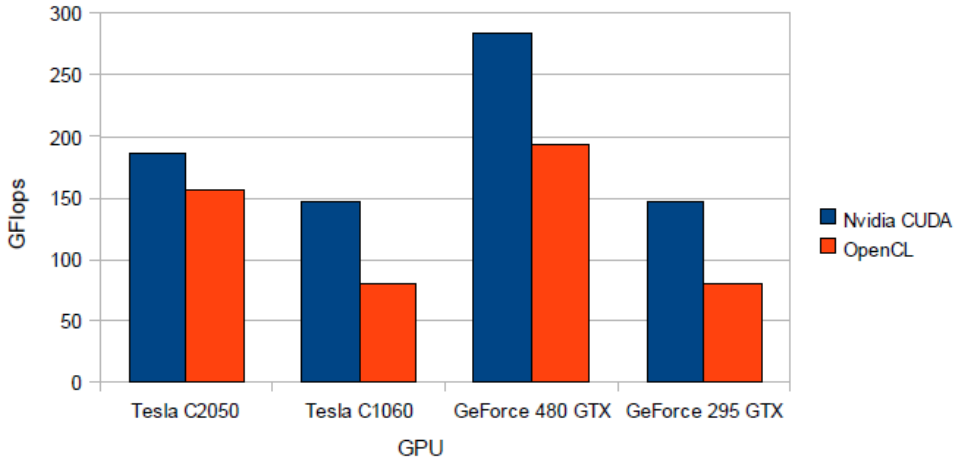
Недоліками OPENCL є наступні [28]:

- заборона посилань на функції;
- посилання на посилання можливе лише всередині ядра, проте не як аргумент;
- не підтримуються бітові поля;
- немає підтримки структур та масивів змінної довжини;
- немає підтримки рекурсії;
- запис за посиланням типу з розміром меншим, ніж 32 біта, не підтримується;
- тип даних Double не підтримується, хоча зарезервований;

- запис у 3D картинці не підтримується.

Зауважимо, що деяких із цих обмежень можна уникнути завдяки розширенню.

Для порівняльної характеристики NVIDIA CUDA та OPENCL першочергово варто співставити ефективність конкуруючих API для програмування під графічні прискорювачі.



Графік 3.1 – Порівняння ефективності технологій NVIDIA CUDA та OPENCL [29]

Отже, CUDA стабільно випереджає стандарт OPEN, причому ця перевага у відсотковому співвідношенні мало залежить від відеокарти, яка використовується.

3.2. Висновки до розділу 3

У третьому розділі дано порівняльну характеристику переваг та недоліків NVIDIA CUDA та OPENCL, на основі якої було обрано програмне забезпечення для паралельних обчислень на графічних процесорах з метою його використання при розробці ПП для фільтрації шумів на зображеннях.

РОЗДІЛ 4. РОЗРОБКА ПП НА ОСНОВІ КА З ВИКОРИСТАННЯ NVIDIA CUDA ДЛЯ ФІЛЬТРАЦІЇ ШУМІВ З ВИКОРИСТАННЯМ КА ТА NVIDIA CUD

4.1 Теоретичні основи написання алгоритму на основі КА

У результаті роботи, проведеної мною в першому та другому розділах даної курсової роботи, я обрав фільтрацію шумів з допомогою КА, зокрема для фільтрації мною обрано імпульсний шум, також відомий як шум «Солі та Перцю» [4]. Задачу розглядатиму для зображень «у відтінках сірого».

Як наслідок роботи, проведеної мною в розділі 3, обрано програмне забезпечення для паралельних обчислень на графічних процесорах INVIDIA CUDA.

За основу алгоритму (як зразок) взято алгоритм медіанного фільтру та реалізовано його за допомогою КА. (Зауважимо, що медіанний фільтр було розглянуто в пункті 2.2.2 курсової роботи).

У пункті (1.1) курсової роботи було дано визначення граничних правил, до якого було дано відповідний рисунок (див. рисунок 1.4). Граничними правилами називають значення, якими будуть заповнені клітини, котрі мають не повний набір сусідів.

Всього є чотири наступних типи граничних умов:

а) фіксована (крайні клітини з’єднанні з логічним станом нуль/один);

б) періодична (крайні клітини прилягають одна до одної);

в) адіабатична (крайні клітини реплікують цей стан);

г) рефлексивна (дзеркальні стани замінюють крайніми клітинами).

4.2 Алгоритм фільтрації шумів на основі КА

Будемо окремо розглядати ті клітинки, у яких є всі сусіди та окремо розглядатимемо ті, у яких сусідів не вистачає.

Спочатку розглянемо випадок, коли клітинка має всіх сусідів.

У цьому випадку будемо обраховувати кольори пікселів, починаючи з пікселя, координати якого (1;1), і до пікселя з кординатами (height-1;width-1). Інші клітинки будемо обраховувати в другому випадку.

Розглянемо алгоритм для першого випадку.

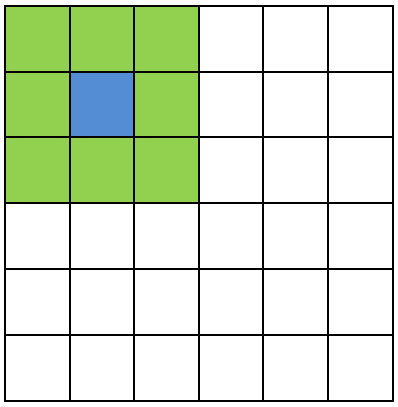


Рис. 4.1 Випадок, коли клітинка має всіх сусідів.

Позначимо синім кольором піксель, значення якого потрібно знайти. Зеленим кольором позначимо пікселі, які є сусідніми до обраного. Для знаходження значення пікселя після фільтрації розглянемо сусідство Мура розглядуваного пікселя. Воно складається з дев’яти пікселів.

Обчислимо для них медіану.

За допомогою КА це можна реалізувати у такий спосіб:

кожен піксель матиме 4 параметри, а саме:

a – значення пікселя;

i, j – координати пікселя

res - додаткова змінна.

Для знаходження нового значення пікселя створюємо масив з 9 комірками (сусідство Мура), в які запишемо відповідно значення шуканого пікселя та його сусідів. Назвемо matrixValues.

Створюємо ще один масив з 9 комірками та заповнимо його нулями. Назвемо matrixRes.

Створюємо цикл, у якому порівнюємо значення кожного пікселя відповідно з усіма значеннями пікселів цього сусідства (у тому числі самого з собою).

На кожному кроці циклу перевіряємо чи виконується якась з умов КА, якщо так, тоді виконує її.

Умови КА:

1) Якщо matrixValues[i] > matrixValues[i+1], тоді matrixRes[i] збільшуємо на 1.

2) Якщо matrixValues[i] <matrixValues[i+1], то matrixRes[i] збільшуємо на 1.

3) Якщо matrixValues[i] = matrixValues[i+1], то matrixRes[i] залишаємо без змін.

Після завершення циклу, значенню кожної комірки matrixRes присвоюємо модуль значення цієї комірки.

Знаходимо комірку масива matrixRes з мінімальним значенням. У масиві matrixValues обираємо комірку, номером якої є номер тієї комірки, яка мала мінімальне значення в масиві matrixRes. Одержане значення є новим значенням клітинки.

Розглянемо другий випадок (клітинки, у яких не вистачає сусідів).

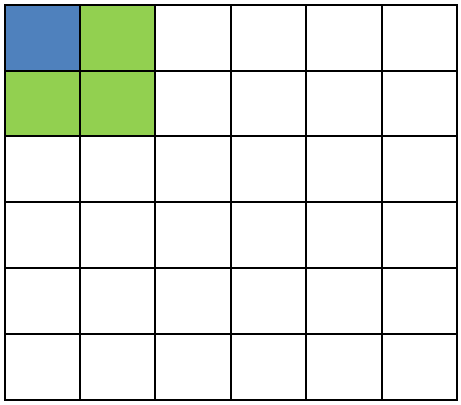


Рис. 4.2 Випадок, коли у клітинки не вистачає сусідів

Розглянемо граничне періодичне правило (б)

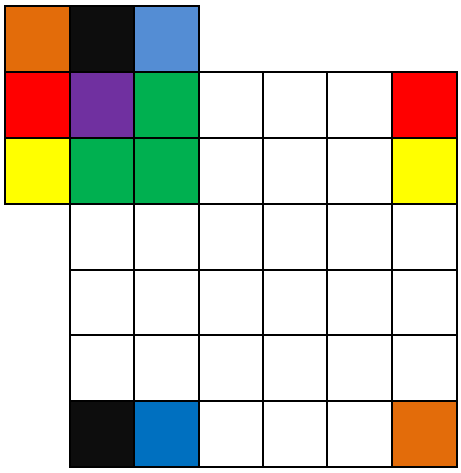


Рис. 4.3 - Випадок для граничного періодичного правила

На малюнку фіолетовим кольором позначено клітинку, у якої не вистачає сусідів. Доповнимо відсутні клітинки у такий спосіб: на місці відсутньої клітинки створюємо клітинку, яка є аналогом до найбільш віддаленої або по стовпчику, або по рядку, або по діагоналі (за зразком, даним на рисунку 4.3). На рисунку відповідні клітинки позначено однаковим кольором.

Зауважимо, що у клітинки може бути відсутніх точно п’ять сусідніх, якщо вона кутова, або точно три в іншому випадку. Процес доповнення відсутніх клітинок для них аналогічний.

Розглянемо граничне адіабатичне правило (в).

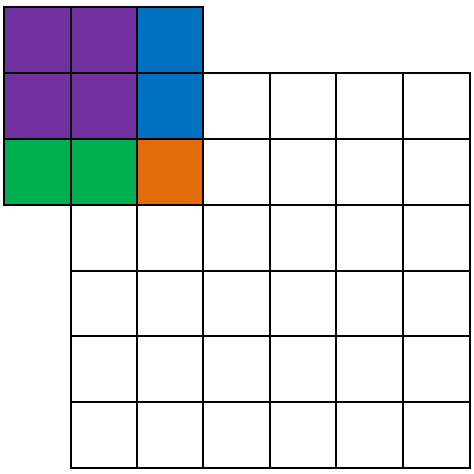


Рис. 4.4 - Випадок для застосування граничного адіабатичного правила

Знову на малюнку фіолетовим кольором позначено клітинку, у якої не вистачає сусідів. Доповнимо відсутні клітинки в наступний спосіб: на місці відсутньої клітинки створюємо таку клітинку, яка є аналогом до найближчої або по рядку, або по стовпчику, або по діагоналі (за зразком даним на рисунку 4.4). На рисунку відповідні клітинки позначено однаковим кольором.

Розглянемо граничне рефлексивне правило (г)

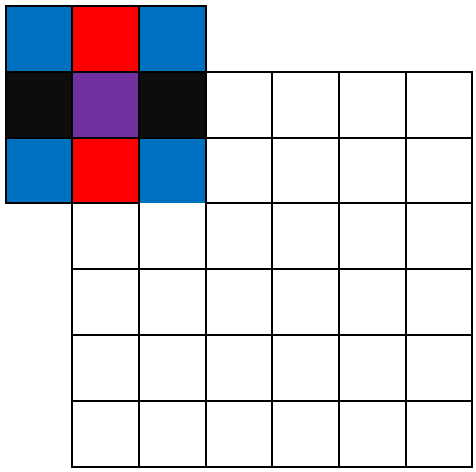


Рис. 4.5 - Випадок для застосування граничного рефлексивного правила

І в цьому випадку знову на малюнку фіолетовим кольором позначено клітинку, у якої не вистачає сусідів. Доповнимо відсутні клітинки в наступний спосіб: на місці відсутньої клітинки створюємо клітинку, яка є повним аналогом до центрально симетричної до неї (див. рисунок 4.5, центр симетрії в даному випадку – синя клітинка). На рисунку відповідні центрально симетричні клітинки позначено однаковим кольором.

Розглянемо граничне фіксоване правило (а)

В даному випадку значення створених клітин не залежать від значень інших клітин. Створені клітини можуть набувати значень 0 або1 довільним чином Оскільки ми працюємо із зображеннями «у відтінках сірого», то значення, яких можуть набути створені клітинки 0 або 255.

Після цього, для обраної точки будуть відомі всі її сусіди.

Оскільки всі її сусіди вже відомі, то можемо знайти її нове значення так, як зазначено у першому випадку.

Для написання програми мною було обрано граничне адіабатичне правило (в).

4.3. Реалізація алгоритму фільтрації шумів на основі КА

Для написання програми мною було обрано граничне адіабатичне правило (в).

Оскільки використовується INVIDIA CUDA, то для реалізації алгоритму обрано мову програмування С ++. Наводжу фрагмент коду:

```
for (int i = 0; i < sizeMatrix; i++) {
    matrixResults[i] = 0;
    for (int j=0; j< sizeMatrix; j++)
    {
       if (matrixValues[i] > matrixValues[j])
       {
          matrixResults[i]++;
       }
       if (matrixValues[i] < matrixValues[j])
       {
          matrixResults[i]--;
       }
       if (matrixValues[i] == matrixValues[j])
       {

       }
    }
    matrixResults[i] = abs(matrixResults[i]);
  }
```

```
newValue = matrixResults[0];
for (int i = 1; i < sizeMatrix; i++)
{
   if (matrixResults[1] < newValue)
   {
      newValue = matrixResults[i];
   }
}
```

4.4 Порівняння результатів фільтрації шумів

Оскільки медіанний фільтр є найбільш ефективним для фільтрації шумів на зображеннях, то для порівняння одержаних результатів на основі фільтрації шуму на зображенні обрано саме його.

Рис. 4.6 ілюструє реконструйоване стандартне зображення Lena 228x256 пікселів: а) зашумлене; б) медіанний фільтр; в) реконструйоване за допомогою фільтра на базі КА.

а) зашумлене [30]; б) медіанного фільтр [30] в) фільтр на основі КА

Рис. 4.6 Фільтрація шумів зображення Lena 10%

4.5 Висновки до розділу 4

У даному розділі на основі КА з використанням NVIDIA CUDA створено ПП для фільтрації шумів на зображеннях та здійснено порівняння з існуючими рішеннями.

ВИСНОВКИ ПО РОБОТІ

В роботі дано дефінітивну характеристику поняття «клітинний автомат», зроблено огляд методів аналізу зображень, розглянуто приклади застосування КА для аналізу й обробки зображень, дано критичну характеристику методів вирішення задачі фільтрації шумів на зображеннях.

На основі КА з використанням NVIDIA CUDA розроблено алгоритм для фільтрації шумів.

Виконано програмну реалізацію розробленого алгоритму.

СПИСОК ЛІТЕРАТУРИ

1. Wolfram S. A New Kind of Science. / S. Wolfram. – Wolfram Media. Inc. – 2002. – 1125 p.

2. Rosin, P. L., Adamatzky, A., & Xianfang, Sun. (2014). *Cellular Automata in Image Processing and Geometry*. Springer.

3. Чичварин Н. В. Предварительная обработка изображений. – Режим доступу: http://ru.bmstu.wiki/ Предварительная_обработка_изображений. – Дата доступу: 26.04.2020.

4. Жежерун О. П., Калітовський Б. В. Використання клітинних автоматів для вирішення задач фільтрації шумів та виявлення контурів зображень / О.П.Жежерун, Б.П.Капітовський // Наукові записки НаУКМА. Комп'ютерні науки. – 2019. – Том 2. – С. 66 – 72.

5. Тоффоли Т., Марголус Н. Машины клеточных автоматов / Т. Тоффоли, Н. Марголус / Перевод с англ. – М. : Мир, 1991. – 280 с.

6. Hoekstra A.G., Kroc J., Sloot P. (2010). Simulating complex systems by cellular automata. London: Springer, 235 p.

7. Mohammed, J., & Deepak, R. N. (2010). *An Effcient Edge Detection Technique by Two Dimensional Rectangular Cellular Automata.*

8. Choudhury, P. P., Nayak, B. K., Sahoo, S., & Rath S. P. (2008). *Theory and Applications of Two-dimensional, Null boundary, Nine neighborhood, Cellular Automata Linear rules.* arXiv:0804.2346, cs.DM;cs.CC; cs.CV.

9. Wuensche A. Classifying Cellular Automata Automatically / Wuensche A. COMPLEXITY. – 1999. – №4. – pp. 47–66.

10. Лебедев А. В. Вероятностные методы классификации клеточных автоматов / А. В. Лебедев // Фундаментальная и прикладная математика. Открытые системы. – 2002. – № 2. – С. 621–626.

11. Shukla, A. P. (2016). Training Cellular Automata for Image Edge Detection. *Romanian Journal of Information Science and Technology, 19,* 4, 338–359.

12. Peer, M. A., Fasel, Qadir, & Khan, K. A. (2012). Investigations of Cellular Automata Game of Life Rules for Noise Filtering and Edge Detection. *I. J. Information Engineering and Electronic Business, 2,* 22–28.

13. Selvapeter, P. J., & Wim, Hordijk (2009). Cellular Automata for Image Noise Filtering. In *World Congress on Nature Biologically Inspired Computing,* 193– 197.

14. Большаков А. А., Булдаков Н. С. Использование клеточных автоматов для обработки изображений минных полей / А. А. Большаков, Н. С. Булдаков // Вестник Саратовского государственного технического университета. – 2010. – №2. – С. 120–124. – Режим доступу: http://cyberleninka.ru/article/n/ispolzovanie-kletochnyh-avtomatov-dlyaobrabotki-izobrazheniy-minnyh-poley. – Дата доступу: 26.04.2020.

15. Обработка и анализ цифровых изображений с примерами на LabVIEW IMAQ Vision / [Визильтер Ю. В., Желтов С. Ю., Князь В. А., Ходарев А. .Н., Моржин А. В.]. – М. : ДМК Пресс, 2007. – 464 с.

16. Методы распознавания образов и анализа изображений: учеб. пособие / [Глумов Н. И., Коломиец Э. И., Мясников В. В., Сергеев В. В.]. – Самара: Изд-во Самар. гос. аэрокосм. ун-та, 2013. – 142 с.

17. Tukey J. W. (1971). Exploratory Data Analysis. Addison-Wesley, Reading, Mass.

18. Кельберт М. Медианная фильтрация / М. Кельберт, Л. Питербарг // Квант. – 1990. – №10. – С. 8–13, 47.

19. Питербарг Л. И. Медианная фильтрация случайных процессов / Л. И. Питербарг // Проблемы передачи информации. – 1984. – №1. – С. 55–73.

20. Прэтт У. Цифрова обробка изображений. Книга 2: [пер. з англ. В. П. Андреевым, А. Л. Зайцевым, Д. С. Лебедевым, В. Г. Поляковым, Н. Н. Тетекиным, В. А. Хлебэродовым]. – М. : Мир, 1982. – Кн.2. – 480 с.

21. Сойфер В. А. Компьютерная обработка изображений. Часть 2. Методы и алгоритмы / В. А. Сойфер // Соросовский образовательный журнал. – 1996. – №3. – С. 110–121.

22. Черненко С. А. Медианный фильтр / С. А. Черненко. – Режим доступу: http://www.logis-pro.kiev.ua/math_power_medianfilter_ru.html. – Дата доступу: 27.04.2020.

23. Дубровская В. М. Использование технологии CUDA для вычислительных процессов [Електронний ресурс]. – Режим доступу: https://www.nvidia.ru/content/EMEAI/PDF/VKR_Ispolzovanie_tekhnologii_CUDA_Dubrovsk.pdf.

24. Берилло А. NVIDIA CUDA – неграфические вычисления на графических процесорах. – Режим доступу: https://www.ixbt.com/video3/cuda-1.shtml. – Дата доступу: 27.04.2020.

25. NVIDIA CUDA — Неграфические вычисления на графических процессорах [Електронний ресурс]. – Режим доступу: http://www.ixbt.com/video3/cuda-1.shtml.

26. NVIDIA – World Leaderin Visual Computing Technologies [Електронний ресурс] – Режим доступу: http://www.nvidia.ru/page/home.html.

27. Антонюк В. В. OPENCL Открытый язык для параллельных программ / В. В. Антонюк. – Москва: Физический факультет МГУ им. М.В.Ломоносова, 2017. – 88 с.

28. Романенко А. А. Введение в OpenCL [Електронний ресурс]. – Режим доступу: http://ccfit.nsu.ru/arom/data/CUDA_2012/10%20OpenCL.pdf

29. Кривов М. А., Казеннов А. М. Сравнение вычислительных возможностей графических ускорителей NVidia при решении различных классов задач (Издательство SCR-Media, журнал «Суперкомпьютеры», Московский Физико-Технический Университет) [Електронний ресурс]. – Режим доступу: http://agora.guru.ru/hpc-h/files/017_Krivov_NvidiaGpuComparision.pdf

30. Медианный фильтрация. Теоретическая часть ЦОС. Медианный фильтр. – Режим доступу: https://studbooks.net/2334624/tehnika/teoreticheskaya_chast