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

АНАЛІЗ ЗОБРАЖЕНЬ ЗА ДОПОМОГОЮ КЛІТИННИХ АВТОМАТІВ

Текстова частина до курсової роботи
за спеціальністю 122 «Комп'ютерні науки та інформаційні технології»

Керівник курсової роботи
канд.фіз.-мат. наук Жежерун О.П.
(прізвище та ініціали)
_____
(підпис)
"____" _____ 2020 р.
Виконала студентка Хміль Ю.В.
(прізвище та ініціали)
_____
(підпис)
"____" _____ 2020 р.

Київ-2020

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

ЗАТВЕРДЖУЮ
Зав. Кафедри мультимедійних систем,
_____ О.П.Жежерун
(підпис)
“____” _____ 2019 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

Студентці Хміль Юлії Василівні факультету інформатики 4 курсу
ТЕМА Аналіз зображень за допомогою клітинних автоматів

Вихідні дані:
Зміст ТЧ до курсової роботи:
Календарний план
Вступ
Частина 1: Теоретичні відомості про клітинні автомати
Частина 2: Обробка зображень
Частина 3: Обробка клітинним автоматом
Частина 4: Практична частина
Висновки
Список використаної літератури
Додатки

Дата видачі “____” _____ 2019 р. Керівник _____
(підпис)
Завдання отримала _____
(підпис)

Тема: Аналіз зображень за допомогою клітинних автоматів

Календарний план виконання роботи:

№	Назва етапу	Термін виконання	Примітка
1.	Отримання теми курсової роботи	08.10.2019	
2.	Пошук тематичної літератури	05.11.2019	
3.	Ознайомлення з літературою	05.12.2019	
4.	Визначення частин курсової роботи	20.12.2019	
5.	Написання текстової частини	01.03.2020	
6.	Виконання практичної частини	31.03.2020	
7.	Подання першої версії записки науковому керівнику	31.03.2020	
8.	Перегляд змісту роботи керівником	18.04.2020	
9.	Створення презентації	10. 04.2020	
10.	Захист роботи	24. 04.2020	

Студентка Хміль Ю.В.

Керівник Жежерун О.П.

“____” _____

АНОТАЦІЯ

бакалаврської дипломної роботи Хміль Юлії Василівни
на тему «Аналіз зображень за допомогою клітинних зображень»

Дана курсова робота присвячена розробці програмного продукту для обробки зображень. Метою курсової роботи є аналіз можливості застосування клітинних автоматів для обробки зображень.

В роботі розглянуті загальні відомості про клітинні автомати та методи обробки зображень, розроблено і реалізовано алгоритм з використанням клітинного автомату для пошуку країв на зображенні. Проведено порівняння з звичними методами пошуку країв.

Загальний обсяг роботи: 53 сторінок, 22 рисунків, 1 таблиць, 18 бібліографічних найменувань.

Ключові слова: клітинні автомати, застосування автоматів, обробка зображень, сегментація зображень, медіанна фільтрація, навчання клітинних автоматів, метод Оцу.

ANNOTATION

a bachelor’s degree work of Yuliia Khmil

entitled «Image processing using cellular automata»

This course work is devoted to the development of image processing software. The purpose of the course work is to analyze the possibility of using cellular automata for image processing.

The paper covers general information about cellular automata and image processing methods, developed and implemented an algorithm using a cellular automata to edge detection in an image. A comparison is made with conventional edge detection methods.

Total workload: 53 pages, 22 drawings, 1 tables, 18 bibliographic titles.

Keywords: cellular automata, application of automata, image processing, image segmentation, median filtering, cellular automata training, Otsu method.

Зміст

Перелік прийнятих скорочень

КА – клітинний автомат

SFFS - алгоритм послідовного плаваючого пошуку вперед

ВСТУП

Проблема обробки, розпізнавання, ідентифікації зображень та виявлення їх автентичності із постійним збільшенням обсягів інформаційних потоків та підвищенням вимог до динаміки їх оброблення в реальному часі постає особливо гостро. Із зростанням розмірності зображень, які необхідно ідентифікувати, обчислювальна складність зростає експоненційно.

Процедури обробки зображень та математичної морфології природно паралельні. Інтенсивність пікселя оновлюється залежно від інтенсивності найближчих сусідніх пікселів. Інтенсивність пікселів оновлюється одночасно, за дискретний час і за тим же правилом. Саме так працює клітинний автомат, якщо інтенсивність пікселя приймається за стан комірки. Клітинні автомати можуть бути дуже корисними при розробці обчислювальних парадигм, архітектур та реалізацій, спрямованих на вирішення проблем обробки зображень, розпізнавання образів і генерації.

Цифрова обробка зображень у багатьох програмах вважається процесом у режимі реального часу, в якому швидкість процесу дуже важлива. Тому паралельні алгоритми в обробці зображень набагато важливіші порівняно з послідовними алгоритмами. Клітинні автомати можуть використовуватися як паралельний метод для будь-якого завдання з обробки зображень.

Робота складається з чотирьох розділів. Перший розділ присвячено визначенню клітинних автоматів, їх класів та застосуванню. В другому розділі наведено методи обробки зображень. Третій розділ включає обробку зображень за допомогою клітинних автоматів. Останній четвертий розділ є практичною реалізацією алгоритму обробки зображень.

ЧАСТИНА 1 Теоретичні відомості про клітинні автомати

1.1 Клітинні автомати

Клітинні автомати (КА) складаються з регулярної решітки комірок, кожна з яких може знаходитися лише в одному з кінцевої кількості можливих станів. Стан комірки визначається попередніми станами оточуючого сусідства (околу) клітинок і оновлюється синхронно за дискретні часові етапи. Ідентичне правило, що міститься в кожній комірці, по суті є машиною з кінцевим станом, зазвичай задається у формі таблиці правил із записом для кожної можливої конфігурації сусідніх станів. [1]

Клітинні автомати є дискретними динамічними системами, вони є корисними для моделювання та вивчення таких явищ, як впорядкування, турбулентність, хаос, порушення симетрії тощо, і мали широке застосування у моделюванні:

а) В фізиці: гідродинаміка, термодинаміка, електромагнітні явища (наприклад, модель Ізінга), фундаментальна фізика;

б) В хімії: коливальні процеси в хімічних реакціях (реакція Бєлоусова-Жаботинського), процеси абсорбції, сплавоутворення;

в) В біології: процеси самовідтворення, моделі нервової та імунної систем, процеси морфогенезу;

г) В екології: популяційні моделі, епідіміологія, ерозія ґрунтів, розвиток лісових пожеж;

д) В соціології і економіці.

Клітинні автомати знаходять своє застосування і при вирішенні різних обчислювальних задач:

а) розпізнавання образів;

б) обробка зображень і аналіз відеопотоку;

в) завдання класифікації;

г) дискретна і безперервна оптимізація;

д) генерація псевдовипадкових чисел;

е) криптографія

Протягом останніх п'ятдесяти років різноманітні дослідники, включаючи добре відомі імена, такі як Станіслав Улам (1962) та Джон фон Нейман (1966), Джон Голланд (1970), Стівен Вольфрам (1994) та Джон Конвей (1970) досліджували властивості клітинних автоматів. Зокрема, у 1960-1970-х роках було докладено значних зусиль для розробки апаратного забезпечення, а також правил застосування КА для завдань аналізу зображень. З недавніх пір спостерігається пожвавлення інтересу до властивостей КА, без зосередження на масивно-паралельних апаратних реалізаціях, тобто вони моделюються на стандартних послідовних комп'ютерах.

Вже до 1990-х років КА можна було застосувати для виконання ряду завдань з комп’ютерного бачення, таких як:

а) обчислення відстані на цифрових фото (Розенфельд, Пфальц - 1968)

б) обчислення властивостей бінарних областей, таких як площа, периметр, опуклість (Дайер, Розенфельд – 1981)

в) виконання обробки середнього рівня, такої як заповнення проміжків та збірки шаблонів (Сант П’єр і Мілграм – 1992)

г) виконання операцій із покращення зображення, таких як фільтрація шуму та різкість (Ернандес та Герман – 1996)

д) виконання простого розпізнавання об'єктів (Карафіллідіс – 1997)

Клітинні автомати мають ряд переваг над традиційними методами обчислень:

а) Хоча кожна комірка, як правило, містить лише кілька простих правил, поєднання матриці комірок з їх локальною взаємодією призводить до більш досконалої глобальної поведінки. Тобто, хоча кожна комірка має надзвичайно обмежене уявлення про систему (лише її найближчих сусідів), локалізована інформація поширюється на кожному етапі часу, що дає змогу отримати більш глобальні характеристики загальної системи КА.

б) Простота реалізації і складність поведінки означає, що КА може краще підходити для моделювання складних систем, ніж традиційні підходи. Наприклад, при моделюванні візерунку панцира було виявлено, що КА уникають значних числових проблем, пов'язаних з моделями на основі диференціальних

рівнянь з частковими похідними, а також були значно швидшими для обчислення. [2]

в) КА є як паралельними, так і обчислювально простими. Це означає, що вони ідеально підходять для реалізації схем надвеликого рівня інтеґрації.

г) КА розширювані; правила можна легко додавати, видаляти або змінювати.

1.2 Визначення клітинних автоматів

Найстаріші версії клітинних автоматів мають просту і пряму реалізацію. Вони засновані на використанні звичайного масиву комірок (часто реалізуються у вигляді матриці в комп'ютерному моделюванні), їх локальних змінних та функції переходу, що працює над околом. Ці компоненти класичного клітинного автомата, що використовують окіл фон Неймана, зображені на рис. 1.1

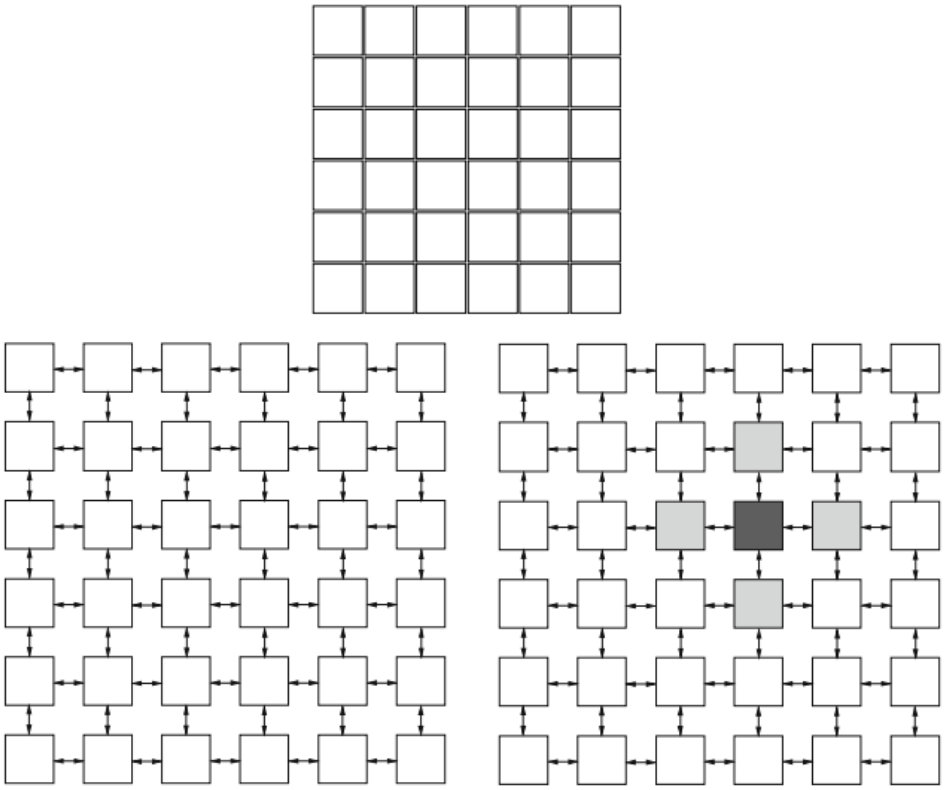


Рисунок 1.1 Показано двовимірну решітку, що визначає клітинний автомат, розмір якого становить 6 × 6 комірок (вгорі). Просторово розширена решітка з явно відображеними зв’язками (рядки зі стрілками) між сусідніми клітинками (знизу вліво). Така топологія, де пов'язані лише найближчі сусіди, визначає окіл фон Неймана з радіусом r = 1 (праворуч знизу), оновленою коміркою (темно-сіра) та сусідами (світло-сіра). Загалом, різні околи визначаються різними наборами зв’язків.

Визначимо клітинні автомати таким чином[3]:

а) Дискретний простір клітинного стану L - це набір локально пов'язаних між собою скінченних автоматів, який, як правило, створюється регулярною d-вимірною решіткою скінченних автоматів. Наприклад, у двовимірних розмірах клітинний автомат складається з решітки квадратів L = m × n, де кожен квадрат, названий клітиною, представлений одним КА.

б) Місцевий простір значень ∑ визначає всі можливі стани для кожного автомату. Стан σ кожного автомату може знаходитися в одному з скінченних чисел станів σ ∈ ∑ ≡ {0, 1, 2, 3, 4, ..., k - 1, k}. Склад усіх можливих станів усіх клітин створює простір станів усього клітинного автомата.

в) Граничні умови можуть бути періодичними, фіксованими або рефлексивними.

г) Окіл N створюється набором N топологічно сусідніх комірок, які впливають на зміну стану кожної оновленої комірки на наступному етапі моделювання. Зазвичай використовуються однорідні околи (що складаються з найближчих сусідів), див. Рис. 1.2 для прикладів околів фон Неймана та Мура.

д) Визначено правило переходу φ: $\overbrace{\sum \times \sum \times \cdots \times \sum}^{N} \to \sum$ опис зміни кожної оновленої комірки від її поточного стану до нового, працюючи над сусідством (розмір N).

е) Усі комірки змінюють свій стан синхронно. Зазвичай це називається одним кроком ітерації.

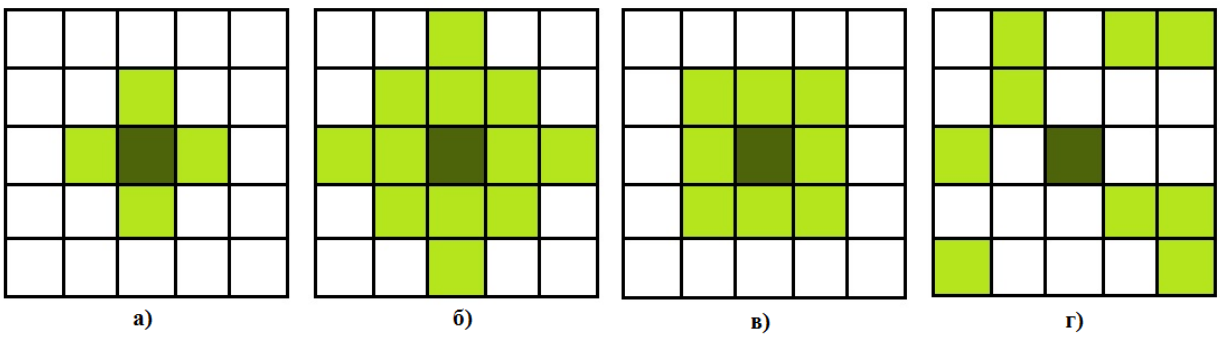


Рисунок 1.2 Різні типи сусідств: а) фон Неймана (радіус 1), б) фон Неймана (радіус 2), в) Мура і г) випадковий. Змінні комірок у межах даного

сусідства (світло-зелений плюс темно-зелений) використовуються для оцінки нових значень оновленої комірки (темно-зеленого) за допомогою функції переходу

Окіл фон Неймана радіуса 1 визначається наступним чином:

$$N^{1}_{\text{фон Неймана}}(i, j) = \{ \sigma_{k,l} \mid |i - k| + |j - l| \leq 1 \} = \{ \sigma_{i,j}, \sigma_{i-1,j}, \sigma_{i,j-1}, \sigma_{i+1,j}, \sigma_{i,j+1} \}$$

Окіл Мура радіуса 1 визначається наступним чином:

$$N^{1}_{\text{Мура}}(i, j) = \{ \sigma_{k,l} \mid |i - k| \leq 1, |j - l| \leq 1 \} = \{\sigma_{i,j}, \sigma_{i-1,j}, \sigma_{i,j-1}, \sigma_{i-1,j-1}, \sigma_{i+1,j}, \sigma_{i,j+1}, \sigma_{i+1,j+1} \}$$

Будь-яке правило переходу φ (t) може бути записане у вигляді функції φ (t), використовуючи стани σ (t) всіх комірок в околицях:

$$\sigma_{i,j}(t + 1) = \varphi(\sigma_{k,l}(t) \mid \sigma_{k,l}(t) \in N),$$

де N визначає околиці, включаючи саму клітинку. Правилом переходу φ може бути будь-яке поєднання арифметичних і логічних операцій, включаючи функції. У реальних додатках типове правило переходу має або форму таблиці переходу (визначає, для яких вхідних значень приймається певне вихідне значення), або форму комп'ютерної програми.

Щоб навести наочний приклад, загальну форму правила переходу для околиці фон Неймана $N^{1}_{\text{фон Неймана}}$ задано:

$$\sigma_{i,j}(t + 1) = \varphi(\sigma_{i,j}(t), \sigma_{i-1,j}(t), \sigma_{i,j-1}(t), \sigma_{i+1,j}(t), \sigma_{i,j+1}(t)),$$

містить усі п’ять сусідів, включаючи саму оновлену клітинку.

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

$$N_r = \sigma^{\sigma^n}$$

Як показано в таблиці 1.1, кількість правил різко зростає із кількістю сусідів n та станів σ.

Таблиця 1.1 Загальна кількість правил для клітинного автомата, що має кількість станів σ та кількість сусідів n. Очевидно, що навіть для автоматів із відносно невеликою кількістю станів та сусідів кількість усіх можливих правил різко зростає із кількістю станів та сусідів

Кількість станів σ	Кількість сусідів n	σ^{σ^n}	Кількість правил n
2	2	2^{2^2}	16
2	3	2^{2^3}	256
2	5	2^{2^5}	4 294 967 296
2	10	$2^{2^{10}}$	$1.797 \cdot 10^{308}$
5	2	5^{5^2}	$2.98 \cdot 10^{17}$
5	3	5^{5^3}	$2.35 \cdot 10^{87}$
5	5	5^{5^5}	$1.91 \cdot 10^{2184}$
10	2	10^{10^2}	10^{10}
10	3	10^{10^3}	10^{1000}
10	5	10^{10^5}	10^{100000}

1.3 Види клітинних автоматів. Класифікація С. Вольфрама

Двовимірні клітинні автомати включають цілу решітку комірок, при цьому колір кожної комірки оновлюється за правилом, яке залежить від сусідів у всіх чотирьох напрямках решітки. На малюнках нижче показано, що відбувається з особливо простим правилом, коли певна клітина приймається чорною, якщо будь-який з чотирьох її сусідів був чорним на попередньому кроці [4].

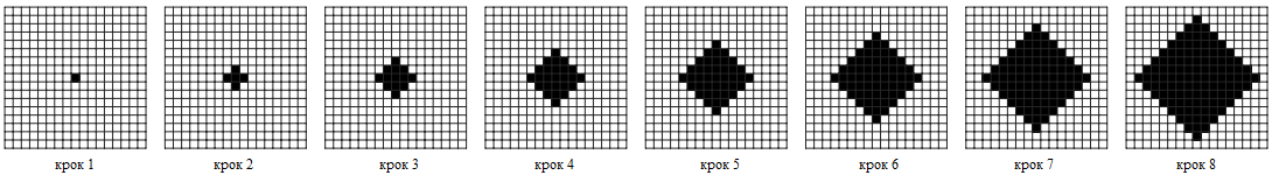


Рисунок 1.3 Послідовні кроки еволюції двовимірного клітинного автомата, правило якого вказує, що певна комірка повинна стати чорною, якщо будь-який з її сусідів був чорним на попередньому кроці

Але трохи змінивши правило, можна отримати більш складні структури зростання. На малюнках нижче показано, що відбувається, наприклад, з правилом, коли кожна клітина стає чорною, якщо лише один або всі чотири її сусіди були чорними на попередньому кроці, але в іншому випадку залишаються того ж кольору, що і раніше.

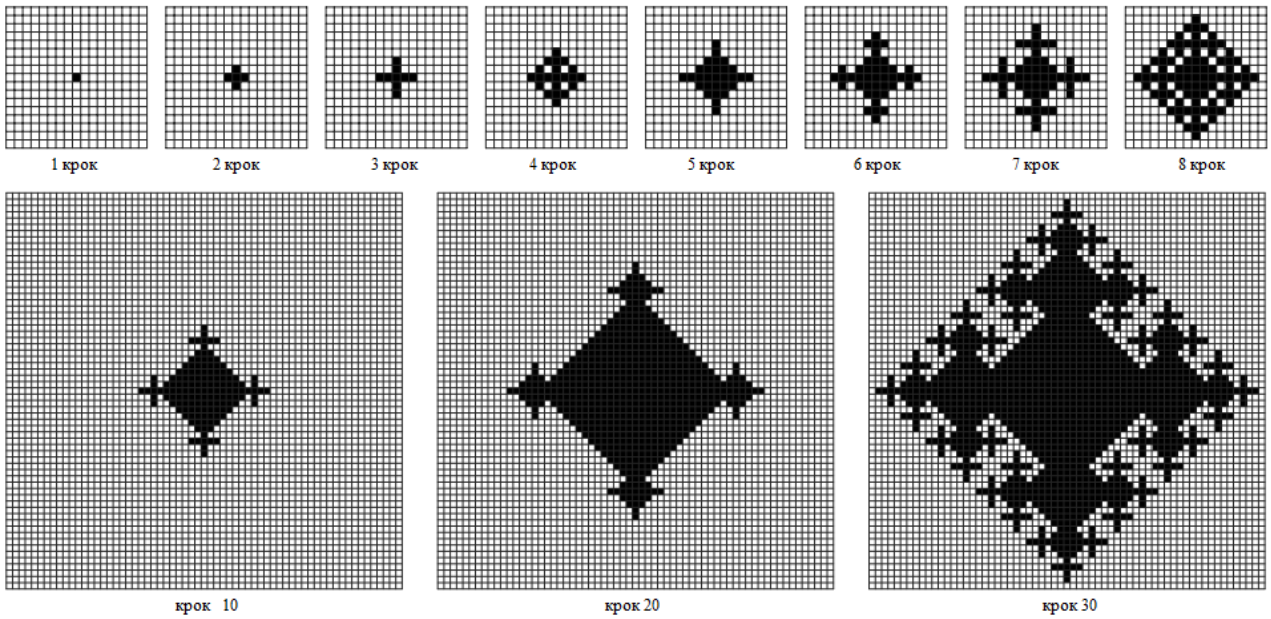


Рисунок 1.4 Етапи еволюції двовимірного клітинного автомата, правило якого вказує, що певна клітина повинна стати чорною, якщо саме на попередньому кроці рівно один або всі чотири її сусіди були чорними, але в іншому випадку повинні залишатися однаковими. Починаючи з однієї чорної клітини, це правило дає складну схему зростання.

Вироблені в цьому випадку візерунки більше не мають простої геометричної форми, а натомість часто мають складну структуру, яка трохи нагадує сніжинку. Але, незважаючи на цю складність, візерунки все ще демонструють велику закономірність.

За словами Вольфрама закономірності, що виникають, майже завжди можна легко віднести до одного із чотирьох основних класів, проілюстрованих нижче.

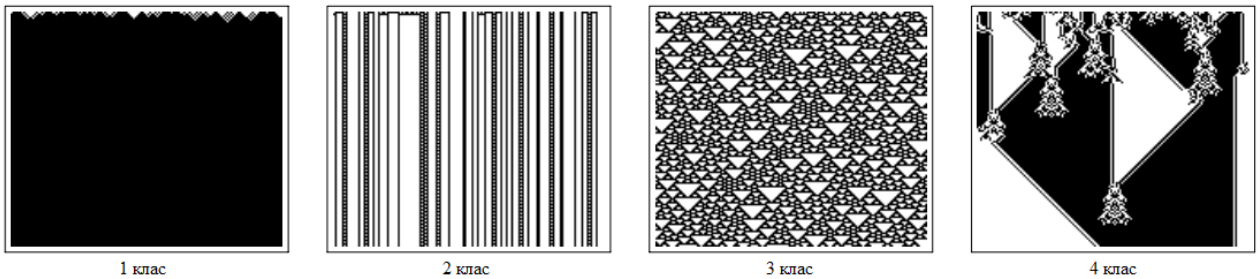


Рисунок 1.5 Приклади чотирьох основних класів поведінки, що спостерігаються в еволюції клітинних автоматів від випадкових початкових умов. Класифікація вперше розроблена в 1983 році.

Ці класи зручно нумерувати в порядку збільшення складності, і кожен має певні безпосередні відмінні риси.

У 1 класі поведінка дуже проста, і майже всі початкові умови призводять до абсолютно однакового остаточного стану.

У 2 класі існує безліч різних можливих кінцевих станів, але всі вони складаються лише з певного набору простих структур, які або залишаються незмінними назавжди, або повторюються кожні кілька кроків.

У класі 3 поведінка є більш складною і багато в чому здається випадковою, хоча трикутники та інші невеликі структури по суті завжди є на певному рівні.

Клас 4 включає суміш порядку і випадковості: створюються локалізовані структури, які самі по собі є досить простими, але ці структури рухаються навколо і взаємодіють одна з одною дуже складними способами.

Але ці чотири класи мають і інші важливі відмінні риси - і одним важливим прикладом цього є їх чутливість до невеликих змін у початкових умовах.

У 1 класі зміни завжди вимирають, і насправді досягається абсолютно такий самий кінцевий стан незалежно від того, які початкові умови були використані. У 2 класі зміни можуть зберігатися, але вони завжди залишаються локалізованими у невеликій області системи. Однак у класі 3 поведінка зовсім інша. Будь-яка зміна, яка робиться, зазвичай поширюється з рівномірною швидкістю, врешті-решт впливаючи на кожну частину системи. У 4 класі зміни також можуть поширюватися, але лише спорадично.

1.4 Варіації клітинних автоматів

Більш ніж за шістдесят років історії клітинних автоматів було придумано і досліджено дуже велике число різних варіацій клітинних автоматів. Найбільш загальними з них (і найбільш часто використовуваними) - є асинхронні, недетерміновані і блокові клітинні автомати.

1.4.1 Асинхронні автомати

Клітинні автомати є розподіленою системою паралельно функціонуючих об'єктів. Наявність загального таймера, за сигналами якого відбувається

оновлення станів всіх клітин виглядає очевидним з точки зору реалізації клітинного автомата на комп'ютері. Однак, якщо розглядати клітинні автомати в контексті моделювання, наприклад, біологічних систем, то таке синхронне оновлення станів всіх клітин вже є не зовсім природним.

При комп'ютерній реалізації класичних синхронних клітинних автоматів на кожному часовому кроці зберігається дві матриці для двовимірних автоматів, в першій зберігаються значення станів клітин на попередньої ітерації, які використовуються для обчислення нових станів, що записуються в другу матрицю (поточні стани). В асинхронних автоматах цей процес спрощується: їхні стани зберігаються в одній матриці, клітини автомата перебираються в випадковому порядку, для кожної з них обчислюється новий стан, який тут же записується в матрицю станів і далі використовується іншими клітинами для поновлення своїх станів. Так як клітини асинхронного автомата перебираються в випадковому порядку, то еволюція асинхронних автоматів є недетермінованою (зокрема, еволюція асинхронних автоматів слабо залежить від початкових умов).

Для оновлення клітин в асинхронних клітинних автоматах використовуються кілька різних схем:

а) для подальшого оновлення вибирається довільна випадкова клітина автомата;

б) на кожному кроці всі клітини перебираються в випадковому порядку

в) порядок перебору складається один раз і використовуються далі на всіх тимчасових кроках;

г) кожна клітина має свій власний таймер (детермінований або недетермінований) за сигналами якого і відбувається її оновлення.

1.4.2 Недетерміновані автомати

У недетермінірованних клітинних автоматах для деяких станів локальної околиці може бути визначено більше одного правила. В цьому випадку, кожному правилу приписується ймовірність його застосування. При чому, сумарна ймовірність всіх правил для кожного стану локальної околиці не повинна

перевищувати одиниці. Оновлення станів в таких автоматах проводиться за наступною схемою. З усіх можливих правил з заданим станом локальної околиці вибирається одне з ймовірністю рівної ймовірності, приписаної цьому правилу. При цьому може використовуватися як синхронна схема оновлення, так і будь-яка асинхронна.

1.4.3 Блокові автомати

У блокових клітинних автоматах стани клітин оновлюються не окремо в кожній клітині, а відразу в деякій групі клітин, званій блоком. Для цього увесь наявний простір клітин розбивається на набір однакових блоків. Для кожного можливого стану блоку визначається його новий стан (правило блокового клітинного автомата), причому новий стан залежить тільки від старого, і не залежить від станів інших блоків. Щоб забезпечити взаємодію всіх клітин, розбиття їх на блоки робиться динамічним - на першому кроці використовується одне розбиття, на другому – інше. Тому клітини, з якими взаємодіє дана клітина весь час змінюються. Найбільш простим видом такого клітинного автомата є автомат з околицею Марголуса. В такому автоматі клітини розбиваються на блоки розміру 2 × 2, які зсуваються на кожному часовому кроці на одну клітку по діагоналі.

1.5 Клітинні автомати в житті

1.5.1 Зростання кристалів

На мікроскопічному рівні кристали складаються з регулярних масивів атомів, розкладених так, як клітини в клітинному автоматі. Кристал утворюється, коли рідина або газ охолоджуються нижче точки замерзання. Кристали завжди починаються з насіння - часто стороннього предмета, такого як зерно пилу - і потім ростуть, поступово додаючи більше атомів до їх поверхні.

В якості ідеалізації цього процесу можна розглядати клітинний автомат, в якому чорні клітини являють собою області твердих тіл, а білі клітини являють собою області рідини або газу. Якщо припускати, що будь-яка клітина, яка

прилягає до чорної клітини, сама стане чорною на наступному кроці, то можна отримати схеми зростання, показані нижче.

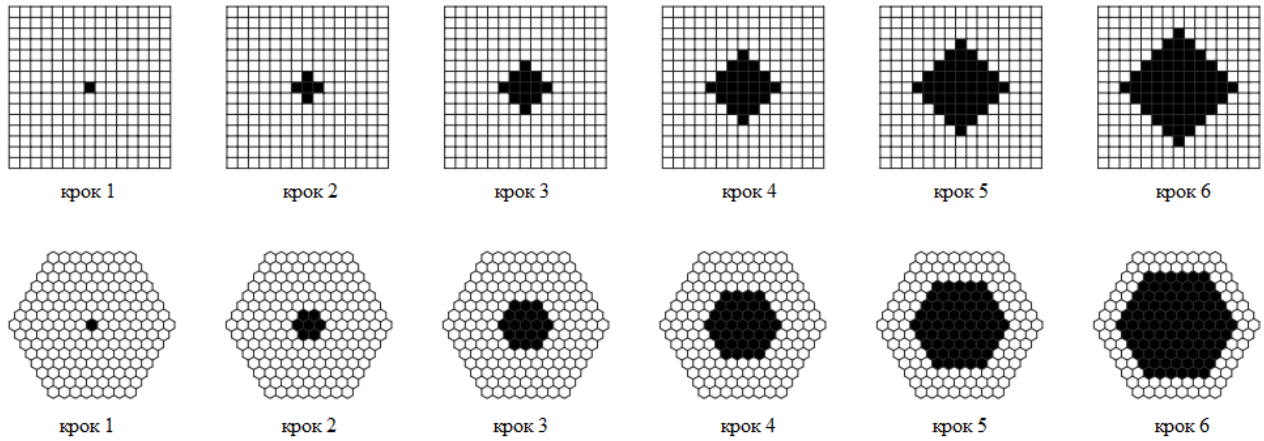


Рисунок 1.6 Клітинні автомати з правилами, які вказують, що комірка повинна стати чорною, якщо хтось із її сусідів уже чорний. Вироблені візерунки мають просту гранену форму, яка безпосередньо відображає структуру основної решітки клітин.

Форми, що утворюються в кожному конкретному випадку, дуже прості, і в кінцевому підсумку складаються просто з плоских граней, розташованих таким чином, що безпосередньо відображає структуру основної решітки комірок. І багато кристалів у природі, включаючи, наприклад, більшість дорогоцінних каменів, мають подібні прості гранітні форми. Але деякі ні. І як один відомий приклад, сніжинки можуть мати дуже хитромудрі форми, як показано нижче.

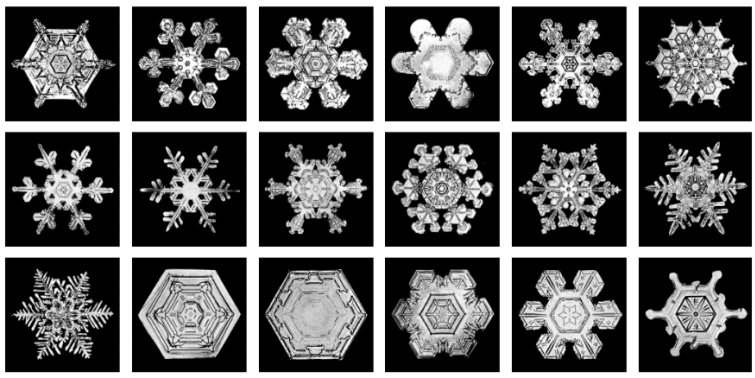


Рисунок 1.7 Приклади типових форм сніжинок

Для гарного наближення всі молекули сніжинки в кінцевому підсумку лежать на простій шестикутній сітці. Але в реальному процесі росту сніжинки не кожна можлива частина цієї сітки закінчується наповненням льодом. Головний ефект, що відповідає за це, полягає в тому, що щоразу, коли до сніжинки

додається шматочок льоду, виділяється деяка кількість тепла, яка потім, як правило, гальмує додавання подальших шматочків льоду поблизу.

Цей основний ефект можна зафіксувати, маючи клітинний автомат з правилами, в яких клітини стають чорними, якщо у них рівно один чорний сусід, але залишаються білими, коли у них більше одного чорного сусіда. На малюнках далі зображено послідовність кроків еволюції такого клітинного автомата. І незважаючи на простоту основних правил, те, що ми бачимо, полягає в тому, що візерунки, які вона виробляє, надзвичайно схожі на ті, що спостерігаються у справжніх сніжинок.

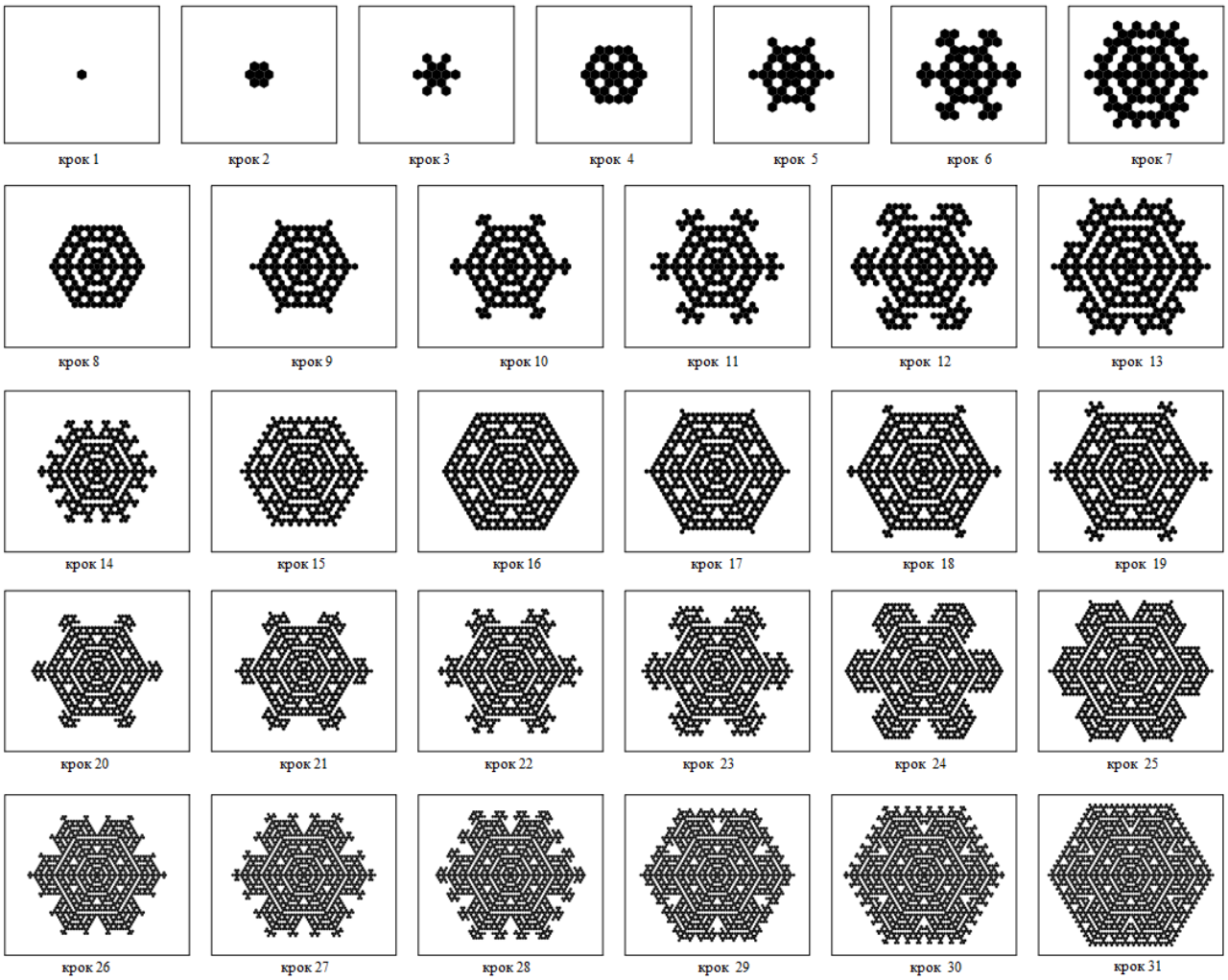


Рисунок 1.8 Еволюція клітинного автомата, в якій кожна комірка на шестикутній сітці стає чорною, коли саме один із її сусідів був чорним на крок раніше. Це правило фіксує основний ефект гальмування росту, який виникає у сніжинок. Отримані візерунки, отримані на різних етапах, надзвичайно схожі на багато справжніх сніжинок.

Поглянувши на поведінку клітинного автомата, можна відразу зробити різні прогнози щодо сніжинок. Наприклад, можна очікувати, що під час росту певної сніжинки має бути чергування між деревоподібними та гранеными формами, оскільки нові гілки ростуть, але потім стикаються між собою.

І якщо дивитися на справжні сніжинки, то є всі ознаки того, що саме так і відбувається. А насправді, показаний вище простий клітинний автомат здається надзвичайно успішним у відтворенні всіляких очевидних особливостей росту сніжинки. Але неминуче є багато деталей, які вона не захоплює.

1.5.2 Клітинний автомат в біології

Біологічні системи часто називають вищими прикладами складності в природі, і не рідко можна припустити, що їх складність повинна бути якось принципово вищого порядку, ніж в інших системах. Вважається, що це повинно бути наслідком досить унікальних процесів адаптації та природного відбору, що діють у біологічних системах.

Іноді елементарні клітинні автомати виявляються в зовсім несподіваних місцях. Текстильний конус, найнебезпечніший для людини молюск із сімейства Конуси. Протиотрути від його отрути поки немає. Малюнок на його раковині - не що інше як візерунок, породжений «Правилом 30».

Рисунок 1.9 Раковина молюска Conus textile, плями забарвлення якої дуже схожі на візерунок, який породжується за правилом 30

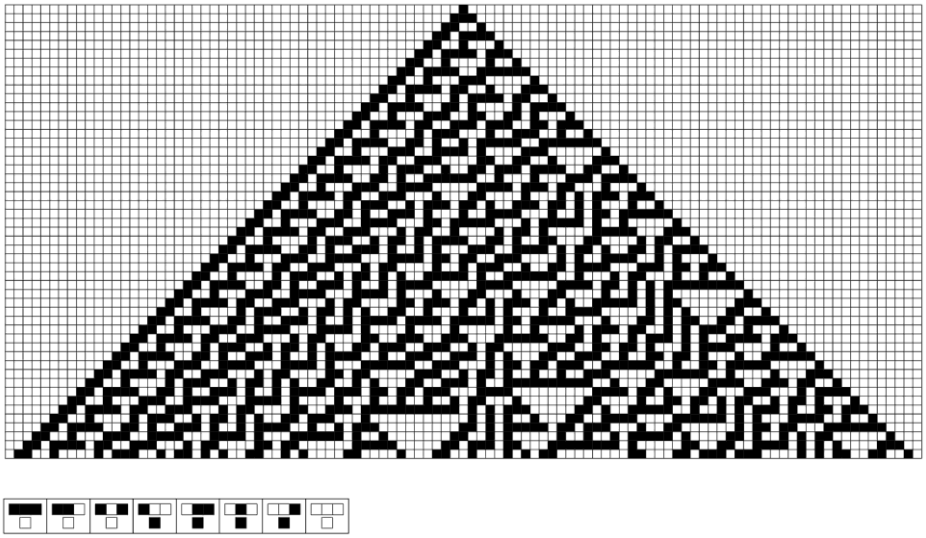


Рисунок 1.10 Розвиток Правила 30 з однієї точки

1.6 Застосування клітинних автоматів

1.6.1 Гра Життя

Класичним прикладом використання клітинних автоматів є гра «Життя». Вона була створена Джоном Хортоном Конвеєм (John Horton Conway) і стала відома світу у 1970 році завдяки праці Мартіна Гарднера (Martin Gardner) [11] .

Площина розбивається на клітки, які можуть знаходитися у двох станах: «живому» (значення у клітинці дорівнює 1) і «мертвому» (значення у клітинці дорівнює 0).

Моделюються такі біологічні процеси: народження, виживання, загибель. Нове покоління одержується з попереднього за такими правилами:

а) клітинка «оживає», якщо «мертва» клітинка межує з трьома «живими»;

б) клітинка «виживає», якщо вона межує з двома або трьома «живими»;

в) клітинка «гине», якщо вона межує з більш ніж трьома «живими» (від перенаселення) або з менше ніж дві (від самотності). Математично ці правила можна записати так:

$$x_{ij}^{t+1} = \begin{cases} 1, \text{якщо } x_{ij}^{t} = 0 \text{ і } k = 3, \\ 1, \text{якщо } x_{ij}^{t} = 1 \text{ і } k = 2 \text{ або } k = 3 \\ 0, \text{якщо } k < 2 \text{ або } k > 2 \end{cases}$$

Тут $x_{ij}{}^{t}$ – стан клітинки у момент часу t, k – кількість «живих» кліток, що межують з кліткою x_{ij}, у момент часу t. Гравець фактично не бере участі у власне

грі. Він тільки може задавати початкову конфігурацію «живих» («мертвих») кліток, які взаємодіють відповідно до визначених правил. Початкові конфігурації змінюються доти, поки їхній стан не набуває одного із можливих варіантів:

а) повне зникнення у зв’язку з перенаселенням або від самотності;

б) формування стабільної конфігурації, що залишається незмінною;

в) утворення конфігурацій, що повторюють свою форму через дві або більше ітерації (періоди).

Було показано, що гра «Життя» еквівалентна універсальній машині Тьюринга, що пояснюється наявністю у ній процесів, еквівалентних універсальним обчисленням.

1.6.2 Застосування клітинних автоматів в криптографії

Клітинні автомати відомі як універсальне обчислювальне середовище, що володіє властивостями самовідтворення, паралельності, однорідності та оновлення за визначеним набором правил. Завдяки здатності генерувати псевдовипадкові послідовності, КА широко застосовуються у криптографії для створення функцій автентифікації повідомлень та хешування. Програмна реалізація таких функцій автентифікації повідомлень дозволяла вдвічі зменшити необхідний час обробки, в порівнянні з традиційними методами, тоді як оптимальне апаратне рішення запропонованого механізму буде в 10 разів швидшим за будь-яку можливу паралельну реалізацію MD-5.[12]

Йдеться про модель ітеративної хеш-функції з використанням структури Меркле-Дамгарда, побудованої на правилах КА, в поєднанні з операціями XOR та побітового зсуву. Для забезпечення криптостійкості розробленої конструкції запропоновано використовувати комбінацію лінійних та нелінійних правил КА. Лінійні правила (наприклад, «60», «90», «150»), в яких використовується лише операція XOR, забезпечують стійкість до колізій. Тоді як нелінійна група правил з AND-OR логікою (такі як, «22», «30», «86», «135»), використовуються для підтримки однонаправленості та нелінійності хеш функцій.

ЧАСТИНА 2 Обробка зображень

2.1 Методи попередньої обробки зображень

Однією з областей техніки, що найбільш швидко розвиваються, є напрямок, пов'язаний з обробкою візуальних даних. Наразі існують десятки комерційних пакетів обробки статичних і динамічних зображень (фотографій, відеофільмів, текстів та інших). Існуючі системи контролю доступу використовують програми розпізнавання осіб, відбитків пальців і райдужної оболонки ока. Також відомі системи розпізнавання номерів транспортних засобів, штрих-кодів та ін. Багато з цих програм функціонують в реальному масштабі часу, виконуючи всі необхідні процедури в темпі надходження даних. Часто це вимагає реалізації ряду алгоритмічних функцій апаратними засобами.

Алгоритмічне забезпечення систем технічного зору можна умовно представити у вигляді двох груп алгоритмів, що виконують функції:

- обробки зображень;
- аналізу візуальних образів.

Сутність обробки зображень полягає у приведенні вихідного зображення до виду, достатнього для його розпізнавання. Методи попередньої обробки залежать від завдань досліджень, досить різноманітні. Сюди відносяться численні процедури формування і поліпшення зображення (які включають компенсацію оптичних перешкод і згладжування), бінаризацію, отримання контурного подання зображення, виділення елементів сцени і визначення їх ознак. Кінцевою метою обробки зображень є підготовка об'єктів сцени до розпізнавання, тобто віднесенню їх до деяких заздалегідь заданим класам. Незважаючи на різноманіття представлених процедур, обробка зображень розбивається на три основні етапи[13]:

а) введення і попередня обробка зображення;

б) сегментація;

в) опис.

Попередня обробка - застосовується з метою зниження перешкод на зображенні, що виникли в результаті дискретизації і квантування, а також

придушення зовнішніх шумів. Як правило, це операції усереднення і вирівнювання гістограм.

Сегментація - процес пошуку однорідних областей на зображенні. Найбільш поширені методи сегментації, засновані на визначенні однорідних яркостей (кольорів) або однородностей типу текстур.

Поліпшення / фільтрація - можуть використовуватися після проведення сегментації, і переслідують ту ж мету що і попередня обробка: зниження перешкод на зображенні, що виникли в результаті дискретизації і квантування, а також придушення зовнішніх шумів.

2.1.1 Сегментація зображень

Для якісної обробки зображення необхідно розбити його на елементи, тобто провести сегментування зображення. Задача сегментування у переважній більшості випадків визначається потребою виокремити об’єкт чи об’єкти на зображенні, причому в більшості випадків сегментування зображення проводиться в автоматичному режимі.[14]

Виділяють такі ознаки якісного сегментування :

а) однорідність області по характеристикам (в першу чергу, по кольору та текстурі);

б) відмінність значень обраних характеристик для суміжних областей зображення;

б) гладкість границь кожного сегменту зображення;

в)незначна кількість «дірок» у сегменті

Загальна класифікація класичних методів сегментування наведена на рисунку 2.1.

Найбільш відомі методи сегментації – порогова сегментація, центроїдне зв’язування, метод водорозділу. Всі ці методи використовують один принцип: групування в області пікселів, що розташовані поруч та мають рівень яскравості, що відрізняється не більш ніж на визначене число. Це число є порогом сегментування. В залежності від порога сегментування можна отримати різні

результати сегментування зображення: різну кількість сегментів, різні параметри сегментів.

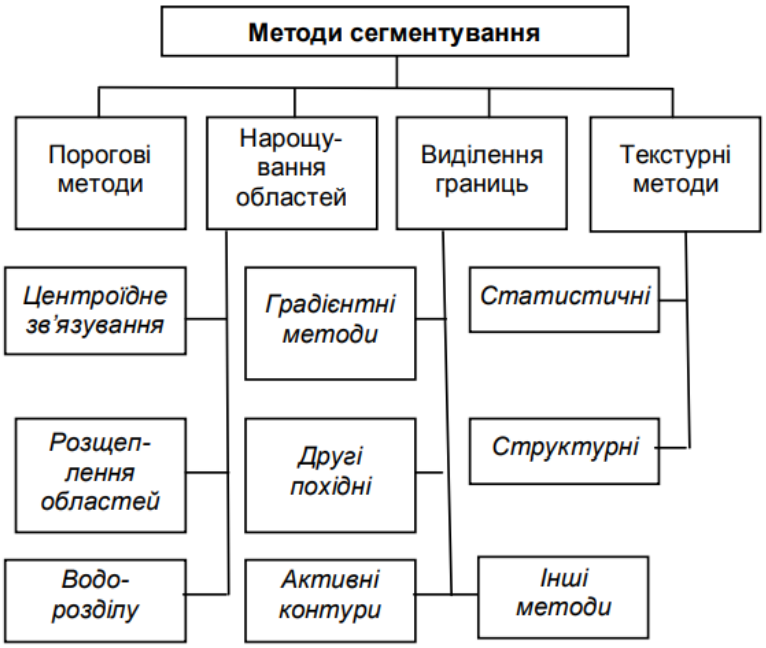


Рисунок 2.1 Загальна класифікація класичних методів сегментування зображення

Порогові методи сегментування зображення. При сегментуванні пікселя (i, j) зображення з яскравістю f_{ij} порогові методи визначають його значення у загальному випадку таким чином:

$$p_{ij} = \begin{cases} 1, \text{у випадку } f_{ij} \leq T, \\ 0, \text{в іншому випадку} \end{cases}$$

де T - величина порогу сегментування.

Порогові методи дозволяють проводити сегментування на простих зображеннях, але, як правило, не дають необхідного результату на зображеннях з наявністю нерівного освітлення, тіней та різного роду завад.

Методи нарощування областей. Якщо на зображенні є стійка зв'язність окремих сегментів, то використовують методи нарощування областей – проводиться групування сусідніх елементів з однаковими або близькими рівнями яскравості, які потім об'єднуються в однорідні області. Найбільш відомими методами нарощування областей є центроїдне зв'язування, розщеплення областей та водорозділу.

При центроїдному зв’язуванні з використанням інформації щодо об’єкту обираються стартові точки, яким присвоюється різні мітки. Точки з однаковими мітками утворюють окремі множини. Такий метод може використовуватися лише для сегментування простих зображень. Для більш складних зображень вибір точок проводиться по ітераціям, на кожній з яких розглядається набір точок на предмет належності їх сусідів даній множині. Точки, що включені у множину на попередніх ітераціях, не розглядаються. Так проводиться аналіз всіх множин по черзі. Точки, що додані до множині на даній ітерації, називаються фронтом, а об’єднання фронтів – хвилею, тому такий метод отримав назву – хвильовий.

Метод розщеплення областей розділяє точки зображення шляхом розбивання деяким чином зображення на квадрати, які потім аналізуються для їх перевірки на однорідність (частіше за все це однорідність за яскравістю). Якщо квадрат не задовольняє умовам однорідності, він замінюється чотирма «підквадратами», а чотири квадрати, що підходять за умови однорідності, можуть бути об’єднані в одну область.

Суть методу водорозділу полягає в тому, що після побудови поля контрасту зображення необхідно побудувати водорозділи (області високої контрастності).

До переваги методів нарощування областей відноситься їх ефективність при роботі з зображеннями, на яких присутні шуми. До недоліків методів нарощування областей відносять в першу чергу те, що вони виділяють загальні фрагменти, в багатьох випадках не показуючи інформації щодо змін яскравості всередині області та можливих внутрішніх границях.

Основою методів сегментування залишаються контурні методи (методи сегментування границь зображення) в силу їх стійкості до незначних варіацій рівня яскравості та контрастності зображення. При цьому важливою є та особливість контурних методів, що по результатах їх застосування при необхідності може бути отримано не тільки границя, але і область зображення.

Методи виділення границь складаються з фільтрації (покращення виділення границь при наявності шумів), підсилення (акцент на точках, де існує перепад яскравості), виділення (з використанням порогового значення приймається рішення щодо включення точок в границю) і локалізації (визначення місця розташування та напрямку). На даний час існує багата кількість методів виділення границь, наприклад, з використанням операторів Робертса, Собела, Превітта, Канні та інші. Існуючі методи поділяються на методи порівняння з еталоном та диференціально-градієнтні методи. Обидва методи визначають, коли коливання градієнта яскравості становиться достатньо великим, щоб стверджувати, що на цьому місці знаходиться границя об’єкта. Принципова різниця методів полягає в способі локальної оцінки градієнтного значення та визначенні локальної направленості границь.

Окремою групою виділяються методи сегментування, що засновані на кластеризації. Їх переваги – автоматичні та можуть бути використані для любої кількості ознак та класів. Існуючі методи кластеризації, такі як К-середніх, медоїдний, CURE, ROCK, DBSCAN, створені для знаходження кластерів, які відповідають будь якій статичній моделі. Такі методи можуть дати збій, якщо параметри моделі обрані некоректно, по відношенню до класифікованим даним, або якщо модель не враховує у повній мірі характеристики кластерів. Також деякі методи допускають помилки, якщо дані складаються з кластерів різної форми, щільності та розмірів.

2.1.2 Фільтрація шумів

Будь-які спотворення, перешкоди, шуми погіршують візуальне сприйняття та ускладнюють автоматичну обробку. Наявність шумів негативно впливає на якість зображення. Ослаблення дій перешкод досягається фільтрацією. Під фільтрацією розуміють операцію, у результаті якої є зображення, отримане того ж розміру, як і вхідне, з певними перетвореннями.

Найпростіший варіант фільтрації полягає у присвоєнні центральному пікселю нового значення, яке є середньоарифметичним усіх його сусідніх

пікселів, значення яких відрізняється від значення центрального не більше, ніж на деякий поріг. До таких фільтрів належать круговий усереднюючий фільтр, гаусівський фільтр, гаусівська фільтрація, медіанна фільтрація, вінерівська фільтрація.

Один з видів нелінійного усереднення - це медіанний фільтр. Значення пікселів записуються в ряд, ряд сортується і повертається центральний елемент, тобто медіана ряду. Застосування цього фільтра дає хороші результати для збереження перепадів відтінків, різних кордонів і усунення локальних піків яскравості. У порівнянні з лінійними фільтрами медіанний краще зберігає контури зображення.

Принцип медіанного фільтра полягає в заміні рівня сірого кожного пікселя медіаною рівнів сірого в околиці пікселів, замість використання операції усереднення. Для медіанної фільтрації ми вказуємо розмір ядра, перераховуємо значення пікселів, охоплені ядром, та визначаємо медіанний рівень. Якщо ядро охоплює парну кількість пікселів, використовується середнє значення двох медіан. Перш ніж розпочати медіанне фільтрування, нулі необхідно визначити навколо краю рядка та краю стовпця. Отже, на межах зображення вводиться викривлення краю.

2.2 Пошук країв

2.2.1 Оператор Собеля

Оператор Собеля використовується в обробці зображень для виділення границь. Це дискретний диференціальний оператор, що обчислює наближене значення градієнта чи норми градієнта для яскравості зображення. Оператор Собеля базується на згортці зображення невеликими сепарабельними цілочисельними фільтрами в вертикальному та горизонтальному напрямках. Хоча, апроксимація градієнта досить груба, особливо на високочастотних ділянках зображення.[17]

Оператор використовує ядра 3×3, з якими згортає зображення для обчислення наближених значень часткових похідних по горизонталі та по

вертикалі. Якщо A вихідне зображення, а Gx та Gy — два зображення, де кожна точка містить часткові похідні по x та по y відповідно. Вони обчислюються наступним чином:

$$G_y = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix} * A \quad and\ G_x = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix} * A$$

де $*$ позначає двовимірну операцію згортки.

Координата x зростає «направо», а y — «вниз». Для кожної точки зображення наближене значення градієнта обчислюється через наближенні значення часткових похідних:

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

а також напрямок градієнта:

$$\theta = arctg\frac{G_y}{G_x}$$

де, наприклад, кут Θ рівний нулю для вертикальної границі, в якої темна сторона зліва.

Значення функції існує тільки на регулярній сітці, тому, строго кажучи, не можна знаходити похідні, але припустивши неперервність функції можна застосувати скінченні різниці, а саме оператор Собеля для апроксимації часткових похідних.

2.2.2 Алгоритм Кенні

Алгоритм Кенні (детектор границь Кенні, оператор Кенні) в дисципліні комп'ютерного зору — оператор виділення границь зображення. Був розроблений Джоном Кенні. Метою Кенні було розробити оптимальний алгоритм виявлення границь, що задовільняють трьом критеріям:[18]

а) гарне виявлення (Кенні трактував цю властивість, як підвищення відношення сигнал/шум);

б) гарна локалізація (вірне виявлення положення границі);

в) єдиний відгук на одну границю.

Основні етапи алгоритму. Розмиття зображення для видалення шуму. Оператор Кенні використовує розмивання Гауса з σ = 1,4

$$\mathbf{B} = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * \mathbf{A}.$$

Пошук градієнтів. Границі відмічають там, де градієнт набуває найбільшого значення. Вони можуть мати різні напрямки, тому алгоритм Кенні використовує чотири фільтри для визначення горизонтальних, вертикальних і діагональних ребер в розмитому зображенні.

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

$$\theta = arctg\frac{G_y}{G_x}$$

Кут нахилу градієнту округлюється і може набувати значень 0, 45, 90, 135.

Пошук локальних максимумів. Тільки локальні максимуми відзначаються як границі.

Подвійна порогова фільтрація. Потенційні межі визначаються порогами.

Трасування області неоднозначності. Підсумкові межі визначаються шляхом придушення всіх країв, не пов'язаних з певними (сильними) межами.

Перед застосуванням детектора зазвичай перетворюють зображення в відтінки сірого, щоб зменшити обчислювальні витрати. Цей етап характерний для багатьох методів обробки зображень.

2.2.3 Алгоритм Марра – Хілдрета (фільтр Лапласа-Гауса)

У комп'ютерному зорі алгоритм Марра – Хілдрета - це метод виявлення ребер у цифрових зображеннях, тобто суцільних кривих, де є сильні та швидкі зміни яскравості зображення.

Оператор Марра-Хілдрета має дві особливості. По-перше, він реалізовує диференціальний оператор, який враховує першу або другу просторову похідну зображення. По-друге, він адаптований на роботу на необхідному масштабному

рівні для того, щоб оператори, призначені для обробки великих фрагментів зображення, можна було використовувати для виявлення границь на розмитих, затемнених ділянках зображення, і, навпаки, призначені для обробки малих фрагментів зображення – для виявлення малих елементів зображення на його добре сфокусованих ділянках.

Марр і Хілдрет показали, що в найбільш загальному випадку цим вимогам відповідає поєднання Лапласа, який загострює границі, із згладжуючим фільтром, що задається гаусівською функцією. Регулюючий оператор Марра-Хілдрета має вигляд:

$$\nabla^2 G(r) = \frac{-1}{\pi\sigma^2}\left(\frac{1-r^2}{2\sigma^2}\right)\exp\left(\frac{-r^2}{2\sigma^2}\right)$$

де σ^2 – дисперсія нормального розподілу, а r – відстань до даної точки від центра фільтра.

ЧАСТИНА 3 Обробка клітинним автоматом

3.1 Використання клітинних автоматів в обробці зображень

В обробці зображень зазвичай беруть участь двовимірні КА. Пікселі зображення являють собою комірки КА і вони оновлюють свій стан на основі станів сусідніх комірок (пікселів). Кілька станів комірок КА дозволяють обробляти зображення сірого масштабу або кольорові зображення. Визначення правил, що застосовуються до комірок, щоб відповісти на певний запит при обробці зображень, все ж є нетривіальним завданням.

Клітинні автомати використовуються для різних завдань з обробки зображень, серед яких: геометричні перетворення, фільтрація шуму, виявлення особливостей, виявлення краю. Також були кілька спроб застосовувати КА до сегментації зображень. Цей спосіб має кілька переваг: простоту реалізації, її паралельність, розширюваність, кількість класів не потрібно вказувати до початку сегментації, можливість роботи із зображеннями будь-якого виміру[6].

Зв'язок двовимірного КА з зображенням. Зображення може бути описане як двовимірна функція I.

$$I = f\ (x, y)\ ,$$

де x і y - просторові координати. Амплітуда f в будь-якій парі координат (x, y) називається інтенсивністю I або сірим значенням зображення. Коли просторові координати та значення амплітуди всі кінцеві, дискретні величини, зображення називається цифровим зображенням. Цифрове зображення I представлене єдиним двовимірним цілим масивом для зображення сірого масштабу та серії з трьох двовимірних масивів для кожної кольорової смуги. Оскільки цифрове зображення є двовимірним масивом m × n пікселів, то нас цікавить двовимірна модель КА. Зображення розглядається як двовимірний КА, де кожна комірка представляє піксель у зображенні, а інтенсивність пікселя представлена станом цієї комірки. Значення кольорів пікселів оновлюються синхронно на дискретному етапі часу. Так що для вирішення будь-якої задачі з обробки зображення потрібно дуже мало часу.[7]

Навчання клітинних автоматів для завдання обробки зображень означає вибір оптимального набору правил серед усіх можливих правил, які найкраще відповідають саме конкретному завданню. Поки більша частина роботи над КА вивчає вплив застосування вручну розроблених функцій переходу (правил). Іншими словами, отримати відповідні правила для досягнення бажаного ефекту дуже важко. Навіть якщо брати до уваги лише бінарні зображення, комбінації всіх можливих правил дуже великі. В цілому без вичерпного перерахування всіх комбінацій оптимальний вибір правил не може бути гарантований. Одна проста альтернатива - вимірювати вплив кожного правила окремо, а потім використовувати результати для прямої побудови комбінацій правил. Однак для виявлення хоча б деяких поєднань між правилами потрібні деякі методи.

Після того, як КА будуть навчені початковим рішенням простої проблеми, вони будуть вдосконалені та розширені, розвиваючи більш складні дані. Традиційні методи визначення правил вручну - це повільний і виснажливий процес, через який він непривабливий. Сахота та інші [8] використовували генетичний алгоритм для таких завдань, як виявлення ребер у бінарних зображеннях. У цій роботі створена узагальнена система, яка намагається розкрити точні правила КА, необхідні для вирішення проблеми виявлення краю. Генетичний алгоритм був використаний для пошуку правил, і коли система навчається, вона може обробляти інші зображення. В основному ця модель працює в два чіткі етапи - етап навчання та етап виконання. На малюнку показаний цей навчальний процес.

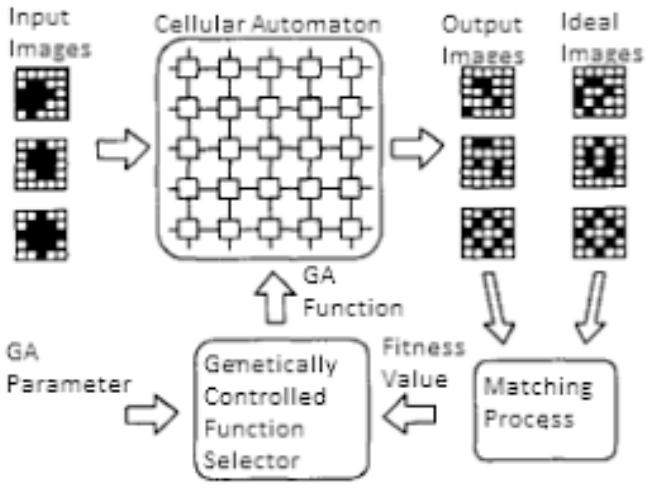


Рисунок 3.1 Модель навчання клітинних автоматів за генетичним алгоритмом

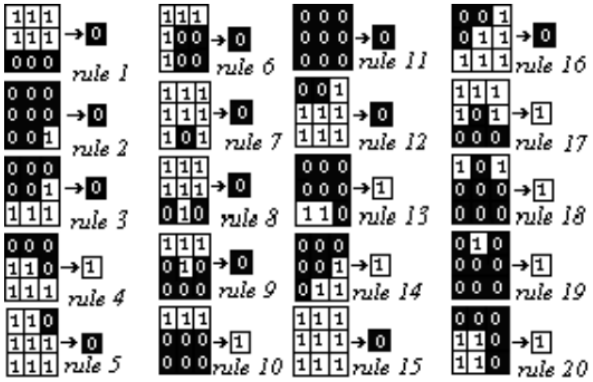


Рисунок 3.2 Найкращий пакет правил клітинних автоматів, знайдений за генетичним алгоритмом

На етапі навчання користувач повинен надати пари вхідних і вихідних зображень. Модель намагається знайти правила для клітинних автоматів, які забезпечать бажаний тип обробки. Шаблони введення спочатку відображаються на масив правил, після чого автомату дозволено оновлюватись на фіксовану кількість циклів. Селектор правил керування КА намагається швидко виявити правила, які дадуть бажані результати за допомогою генетичного алгоритму. Середньоквадратична похибка, що створюється кожною парою зображень, вибирається як об'єктивна функція для керування генетичним алгоритмом. На етапі виконання, коли буде знайдено оптимальний набір правил для бажаної проблеми, система готова. Нові зображення подаються як вхід до тренованих КА, і він виконує завдання на основі оптимальних правил, знайдених на етапі тренувань. Результати цієї роботи показують, що генетично розвинені клітинні автомати можуть успішно вивчити процес виявлення краю для бінарних зображень без шуму.

Слатнія та Казар [9] використовували еволюційний процес для пошуку набору правил клітинних автоматів серед набору оптимальних правил для добування країв у бінарному зображенні, застосовуючи генетичний алгоритм. Генетичний алгоритм був ініціалізований на основі випадкової побудови пакетів правил, добутих із моделі сусідства, як показано на рисунку.

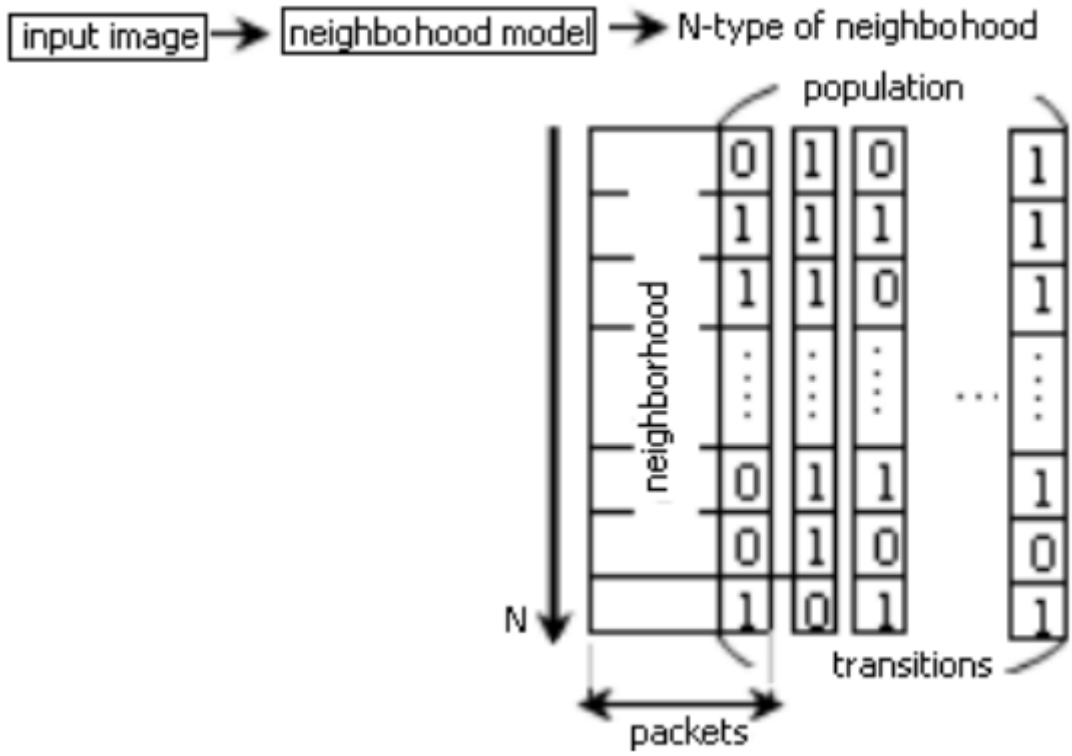


Рисунок 3.3 Побудова моделі сусідства

Для кожного пакета правил КА (аналогічне правило відповідно до його сусідства) шукають зображення, а потім знайдений пакет модифікує центральний піксель відповідно до визначеного переходу. Після цього відстань між результатом країв, виявлених цим методом, та ідеальними краями обчислюється разом з похибкою пропущених пікселів. Знову ж таки, цей метод генерує нову сукупність, застосовуючи селекцію, кросовер та мутацію, використовуючи знаходження краю, описане вище. Цей процес триває до тих пір, поки не відбудеться подальше поліпшення цільової функції або до встановленої максимальної кількості ітерацій.

Зусилля Пола Л. Росіна в галузі навчання клітинних автоматів помітні. У його роботі [1] ефективно проводилось навчання КА для виконання різних завдань обробки зображень. По-перше, можливий набір правил зменшується з 256 до 51 на основі симетрії та рефлексії для центральних чорно-білих пікселів. Після цього проводиться навчання на основі цього скороченого набору правил. Експерименти проводяться за наступних умов:

а) Набори правил можна вивчити та застосувати окремо як для центральних чорних, так і для білих пікселів.

б) Два набори правил (з центральним чорно-білим пікселем) вважаються рівнозначними

в) Лише один із наборів правил чорно-білих вважається відповідним, а інший ігнорується.

У цій роботі кожен зразок правила узгоджується із схемою сусідства пікселів зображення. Якщо він збігається, то колір центрального пікселя інвертується. Перед обробкою кожне правило перетворюється на 8-бітний рядковий шаблон і потім узгоджується з перетвореною схемою околиці пікселя зображення. Для тренувальної мети використаний метод послідовного плаваючого пошуку вперед(SFFS) для вибору функції, для спрямування методу існує вимога цільової функції. Середньоквадратична похибка між вхідним та цільовим зображенням розглядається як цільова функція.

Для того, щоб навчити клітинні автомати для фільтрації шуму солі і перцю, (чергування чорних і білих частинок), застосовується вищезазначений алгоритм, і клітинні автомати навчаються методом SFFS, використовуючи середньоквадратичну похибку як цільову функцію. Встановлено, що при низькому рівні шуму лише одне правило здатне ефективно виконувати таку фільтрацію. З підвищенням рівня шуму до отриманого набору додається більше правил. Отриманий результат порівнюють із медіанним фільтром та математичним морфологічним фільтром. Експериментально встановлено, що навчений набір правил працює краще, ніж інші фільтри.

У разі фільтрації зображень у відтінках сірого, є складність, що кількість можливих правил занадто велика. Цього можна уникнути за допомогою методу порогового розкладання. У цьому методі зображення рівня сірого розкладається шляхом встановлення порогів на всіх можливих рівнях сірого на набір відповідних бінарних зображень. Потім результати тренованого правила, встановленого для бінарних зображень, застосовуються до кожного бінарного зображення, отриманого методом порогової декомпозиції. Потім результати об'єднують. У роботі Пола Росіна [1] результати фільтрації бінарних зображень просто додаються. Перевага цього методу полягає в тому, що результат

тренування бінарного зображення використовується безпосередньо. Недоліком є те, що процес розкладання - це процес, що займає час.

3.2 Навчальна стратегія

Зображення можна розглядати як решітку комірок двовимірних клітинних автоматів, де кожен піксель відповідає конкретній комірці КА. У випадку бінарних зображень піксель може отримати одне з двох значень, тобто 0 і 1. Якщо розглядати сусідство Мура, то для центрального білого пікселя можливі 2^8, тобто 256 правил, так само для центрального чорного пікселя. Використовуючи лемму перерахування Пойя [10], це число можна зменшити, визначивши кількість різних шаблонів після видалення еквівалентних симетричних версій. У випадку сусідства Мура кількість класів еквівалентності у перерахунку на кількість можливих значень пікселів x визначається як:

$Class = \frac{x^8+2x^2+x^4+4x^5}{8}$, де терміни в чисельнику - це 0^o обертання (тотожність) ± 90^o (два обертання), 180^o (одинарне) та чотири обертання, що відповідають дзеркальній симетрії через вертикальні, горизонтальні та діагональні лінії відображення. Для двійкового зображення, у якому дозволені значення пікселів дорівнюють лише 0 і 1. Після вилучення симетрії зменшений набір правил містить 51 правило для центрального пікселя 1 і аналогічно 51 правило для центрального пікселя 0. Отримане правило, встановлене для центрального пікселя 1, показано на малюнку. Той самий набір можна отримати для центрального пікселя 0, замінивши центр 1 на 0. Тепер мета - знайти найкращий набір правил, який найбільше підходить для даної проблеми (виявлення краю).

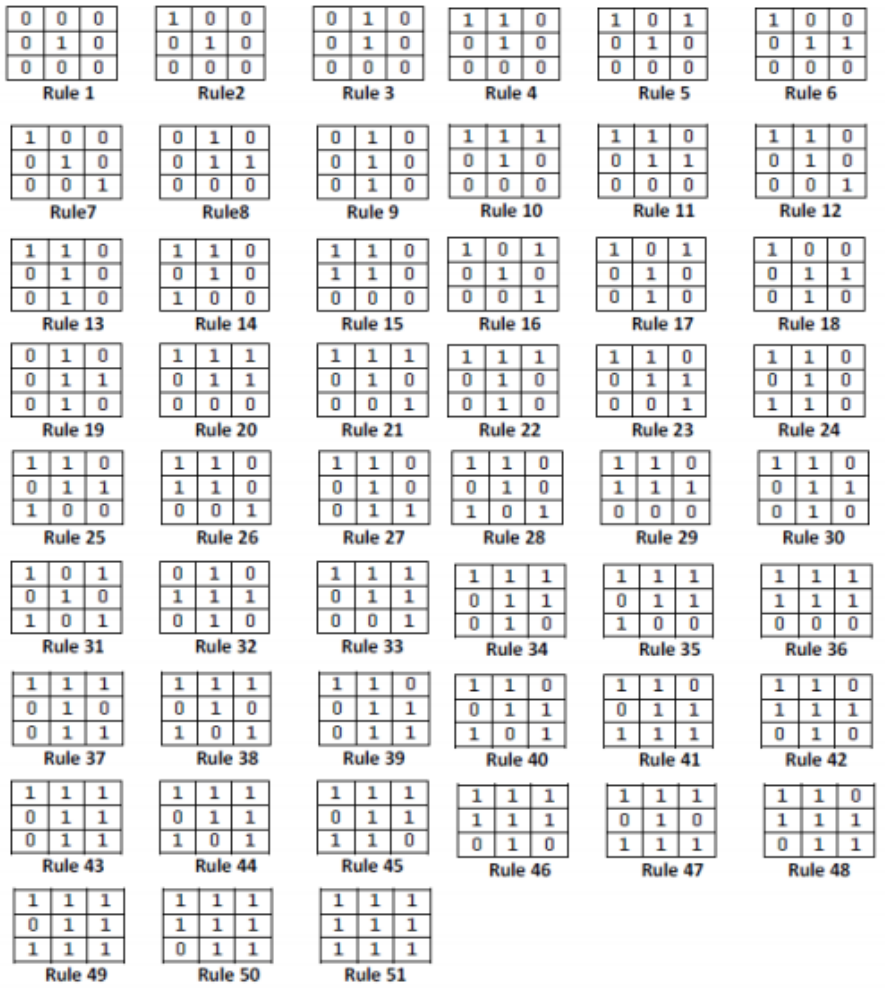


Рисунок 3.4 Набір правил з центральним пікселем 1

3.3 Навчальна стратегія виявлення країв

Краї є значущими локальними змінами інтенсивності зображення або іншими словами, це точки в цифровому зображенні, в яких яскравість зображення різко змінюється або має розриви, і це зазвичай відбувається на межі між двома різними областями зображення. Виявлення країв - один із важливих кроків в обробці зображень, техніці комп'ютерного зору, розпізнаванні образів зображення та аналізі зображень, оскільки це може значно зменшити кількість оброблюваних даних і, отже, може фільтрувати менш релевантну інформацію з цифрових зображень, але зберігає важливу структурні властивості зображення, що спрощує подальше завдання інтерпретації інформаційного змісту на зображенні.

Зосередимось на методиці навчання правилам КА для виконання виявлення краю для бінарних та сірих зображень до значного рівня

продуктивності. Навчальний процес КА займає значно більший час, але це не проблема, оскільки його можна проводити в автономному режимі. Після того, як КА буде навчений для виявлення країв і встановленого набору правил, тоді він може бути безпосередньо застосований до зображень, в яких потрібно виявити край.

3.4 Навчальна стратегія для бінарних зображень

Розглядається сусідство Мура. Граничними пікселями нехтують. Це називається граничною умовою фіксованого значення. Стартовий стан кожного пікселя в клітинних автоматах вважається значенням пікселя вхідного зображення.

Для здійснення виявлення країв в бінарному зображенні кожен набір правил розглядається серед 51 правила, показаного на рисунку 3.4 . Відповідність шаблону виконується лише у білій частині зображення. Кожен піксель зображення береться до розгляду, і якщо шаблон сусідства (сусідство Мура) співпадає з шаблоном правила або набором розглянутих правил, тоді центральний піксель перевернуто на чорний колір, інакше він не зміниться. Чорна частина зображення залишається незмінною. Одне з завдань полягає в моделюванні паралельної поведінки клітинних автоматів у послідовній машині за допомогою одного процесора та послідовних операцій. Для вирішення цього питання алгоритм розроблений таким чином, що при кожній ітерації всі пікселі зображення умовно обробляються паралельно. Для цього оброблені пікселі зберігаються на другому зображенні, а потім копіюються назад у вихідне зображення в кінці кожної ітерації. Псевдокод цього алгоритму наведено далі[15]:

Require: input image of size M ×N, Ruleset

1: [row, column] ← size(A)

2: for i = 2 to M − 1 and j = 2 to N − 1 do

3: for k = 1 to 4 do

4: if A[i,j] = 1 and 3x3pattern(A[i,j]) = Rotate90^{o} (Rule matrix) then

5: C[i, j] ← 0

6: else

7: C[i, j] ← 1

8: end if

9: end for

10: end for

11: B ← C

По-перше, в алгоритмі розглядане правило враховується в наборі RuleMatrix, який є набором 3x3 матриць. Потім для кожного пікселя у зображенні зі значенням 1 околиця пікселя 3x3 перетворюється у формат матриці. Якщо ця модель узгоджується з розглянутими правилами, то значення центрального пікселя інвертується до нуля. Інакше він залишається незмінним. Такі ж дії виконуються для обертання схеми сусідства 3x3 чотири рази на 90^o, щоб також можна було врахувати симетричні візерунки.

3.5 Навчальна стратегія для зображення у відтінках сірого

У випадку із зображеннями у відтінках сірого тренувальний процес дуже ускладнений. У випадку сусідства Мура, де комірки можуть мати 256 можливих інтенсивностей, для центрального пікселя зі значенням 0 можливі 256^8 моделей сусідства. Така ж кількість шаблонів для центрального пікселя з іншими значеннями (від 1 до 255). Навіть якщо скоротити кількість правил, кількість класів перевищуватиме $2x10^{18}$, що очевидно надто багато. Якщо задане зображення сірого масштабу перетворюється на бінарне шляхом порогової декомпозиції, тоді результат попереднього розділу можна застосувати до отриманого зображення. Розін в своїй роботі [1] використав порогову декомпозицію для всіх можливих значень рівня сірого, а потім об'єднав результати кожного розкладу, щоб знайти остаточне зображення. Цей метод працює для усунення шумів, але його ефективність у разі виявлення країв є підозрілим. Крім того, це дуже тривалий процес.

Щоб вирішити вищезгадану проблему, пропонується розкласти зображення сірого рівня методом порогової декомпозиції Оцу. Завдяки властивості мінімізувати дисперсію в межах класу та максимізувати між дисперсією класу, метод Оцу здатний відображати максимальну кількість об'єктів, зображених на зображенні, на порогове зображення.

3.6 Метод Оцу

У комп'ютерному баченні та обробці зображень метод Оцу, названий на честь Нобуюкі Оцу є методом, заснованим на кластеризації, для автоматичного обчислення порогового зображення, або зведення сірого зображення до бінарного зображення[16]. Алгоритм передбачає, що зображення містить два класи пікселів, наступної бі-модальної гістограми: пікселі переднього плану і пікселі тла, потім обчислюється оптимальний поріг, що розділяє два класи, так, що їх комбінований діапазон (дисперсія кластера) є мінімальною або рівноцінною (тому що сума попарних квадратичних відстаней постійна), так, що їх міжкластерна дисперсія є максимальною.

У методі Оцу ми вичерпно шукаємо поріг, що мінімізує дисперсію всередині класу, яка визначається як зважена сума дисперсій двох класів:

$$\sigma_\omega^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$$

Вага w_0 та ω_1 ймовірності двох класів, розділені порогом t і σ_0^2 та σ_1^2 — відхилення цих двох класів.

Клас ймовірностей $\omega_{0,1}(t)$ обчислюється з L гістограм:

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i)$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p(i)$$

Оцу показує, що мінімізація дисперсії всередині класу збігається з максимізацією міжкласової дисперсії:

$$\sigma_b^2(t) = \sigma^2 - \sigma_\omega^2(t)$$
$$= \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 = \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2$$

що виражається в термінах ймовірностей класу ω та класу значень μ.

Коли клас $\mu_{0,1,T}(t)$ означає:

$$\mu_0(t) = \sum_{i=0}^{t-1} i\frac{p(i)}{\omega_0}$$

$$\mu_1(t) = \sum_{i=t}^{L-1} i\frac{p(i)}{\omega_1}$$

$$\mu_T = \sum_{i=0}^{L-1} ip(i)$$

Ймовірності класу і засоби класу можуть бути обчислені ітеративно. Ця ідея дає ефективний алгоритм.

Алгоритм:

1. Обчисліть гістограму та ймовірність кожного рівня інтенсивності
2. Встановіть початкові $\omega_i(0)$ та $\mu_i(0)$
3. Пройдіть усі можливі порогові значення $t = 1, \ldots$ максимальна інтенсивність

   Оновіть ω_i та μ_i

   Обчисліть $\sigma_b^2(t)$
4. Бажаний поріг відповідає максимуму $\sigma_b^2(t)$

ЧАСТИНА 4 Практична частина

Для виконання експериментальної роботи весь процес розділений на дві основні частини. Спочатку було проведено навчання клітинних автоматів для бінарних зображень і було вивчено найкраще правило для виявлення країв. В другій частині реалізовано алгоритм обробки.

4.1 Пошук найкращих правил для виявлення країв

Край будь-якого зображення розпізнається за певними наборами правил. Ці набори правил в основному є частиною клітинних автоматів, що складається з пікселів як комірок, а сусідніх клітин - як сусідніх пікселів. КА складаються з регулярної сітки комірок, кожна з яких являє собою одне з кінцевих чисел можливих станів одночасно. Стан клітини визначається попередніми станами навколишніх клітин на дискретних часових етапах. Ці набори правил застосовуються до бінарного зображення, щоб знайти точку різких змін інтенсивності значення пікселя. Набір точок зміни інтенсивності вважається краєм зображення. У цій техніці існує 2^8 (= 256) можливих станів пікселя, що відповідають їх сусіднім пікселям. Застосовуючи двократну симетрію, ці можливі стани зводяться лише до 51 набору правил. Серед цих 51 правил є одне правило, яке може виконувати еквівалентно інших 50 операцій. Це правило вважається найкращим.

Спочатку виконується операція виявлення країв усіма можливими правилами, а потім отриманий результат порівнюється з іншим методом виявлення країв. Застосовується алгоритм послідовного плаваючого пошуку вперед (SFFS) для пошуку групи найкращих наборів правил серед використаних наборів правил на зображенні для пошуку краю. Після застосування SFFS визначається єдине правило, еквівалентне іншим 51. Псевдокод цього алгоритму наведено далі:

Sequential Floating Forward Search

1: Step 1:

2: Y ← {φ}

3: Step 2:

4: Select the best feature

5: $x^{+} \leftarrow \text{argmax}J(Y_k + x)|x /\in Y_k$

6: $Y_k \leftarrow Y_k + x^{+}$

7: k=k+1

8: Step 3:

9: Select the worst feature

10: $x^{-} \leftarrow \text{argmax}J(Y_k - x)|x \in Y_k$

11: Step 4:

12: if $J(Y_k - x^{-}) > J(Y_k)$ then

13: $Y_k \leftarrow Y_k - x^{-}$

14: k = k + 1

15: Go to Step 3

16: else

17: Go to Step 2

18: end if

Алгоритм SFFS можна описати наступним чином. Нехай Y_k позначає правило, встановлене при ітерації k, і його оцінка буде J (Y_k). Тут J (Y_k) визначається результатом алгоритму пошуку країв шляхом застосування правила КА, встановленого до вхідного зображення, та обчислення цільової функції. Крок 1 показує, що початковий набір правил порожній. На кроці 2 кожної ітерації всі правила, розглядаються як доповнення до набору правил. До отриманого набору правил додається лише правило, що дає максимальний бал. Цей процес повторюється до тих пір, поки не буде досягнуто покращення балів шляхом додавання правил. На кроці 3 кожне правило з набору правил, знайденого на кроці 2, видаляється, щоб знайти правило, видалення якого забезпечує отриманий набір правил із покращеним значенням цільової функції. Як показано на кроці 4, якщо видалення правила спричиняє кращу оцінку цільової функції, вона відкидається від набору правил і знову намагається вилучити наступне правило для видалення, і процес переходить до кроку 3. В

іншому випадку процес іде крок 2 для додавання нового правила до набору правил.

Цільова функція має наступний вигляд:

$$error = 1 - \frac{|NE_0 \cap NE_T| + |E_0 \cap E_T|}{NE_0 + E_0}$$

де E_0 та NE_0 показують крайові та некрайові пікселі зображення, отримані за правилом КА, де E_T і NE_T показує цільове зображення, яке розглядається як опорне зображення з метою обчислення.

Однією з головних проблем для перевірки продуктивності є вибір цільового зображення, з яким слід проводити порівняння результатів. Оскільки ефективність алгоритму залежить від цільового зображення, тому його роль є дуже важливою. У випадку бінарних зображень використовуються зображення, генеровані детектором країв Кенні, оскільки це добре відомий оптимальний алгоритм пошуку країв, що має властивість низького коефіцієнту помилок, хорошої локалізації (Відстань між виявленими крайовими пікселями та реальними крайовими пікселями мінімальна) та мінімальна реакція (лише одна відповідь детектора на край).

Отже, для пошуку спочатку цільової функції краї зображення шукають детектором країв Кенні, і отримане зображення розглядається як опорне зображення. Потім набір правил застосовується до одного і того ж зображення для отримання результуючого зображення краю. Потім обчислюється похибка помилкової класифікації між опорним зображенням і зображенням країв, отриманим набором правил. Ця помилка використовується для керування алгоритмом SFFS для пошуку найкращого набору правил для пошуку країв.

Як результат навчання, найкращим визначено правило 51 (Рисунок 3.4).

4.2 Алгоритм обробки зображення для пошуку країв

Визначено наступний алгоритм:

1. Завантаження зображення
2. Фільтрація зображення медіанним фільтром, визначеним у розділі 2.1.2
3. Бінаризація зображення за методом Оцу, розділ 3.6

4. Пошук країв на зображенні за допомогою правила 51 клітинного автомата, розділ 3.2, рисунок 3.4.
5. Порівняння з стандартними методами пошуку країв на зображенні

Приклад результатів роботи програми:

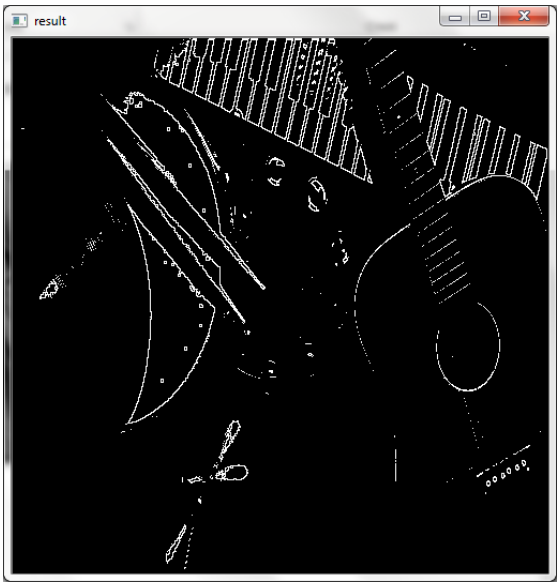


Рисунок 4.1 Пошук країв КА

Рисунок 4.2 Пошук країв методом Кенні

Порівняння з іншими методами:

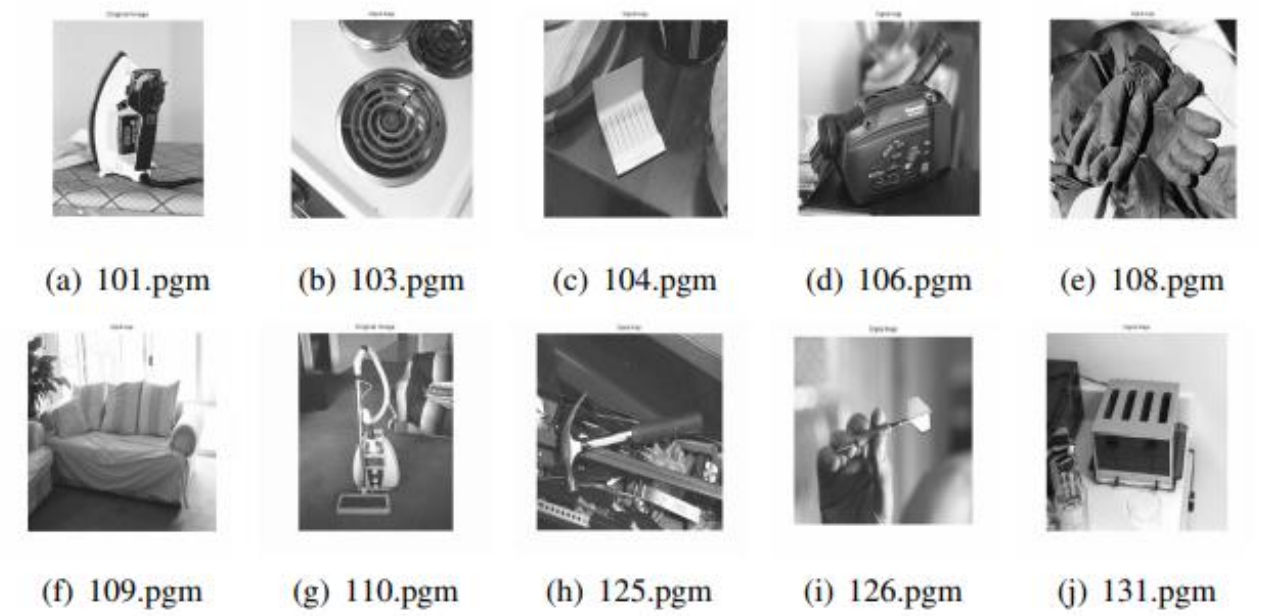


Рисунок 4.3 Вхідні зображення

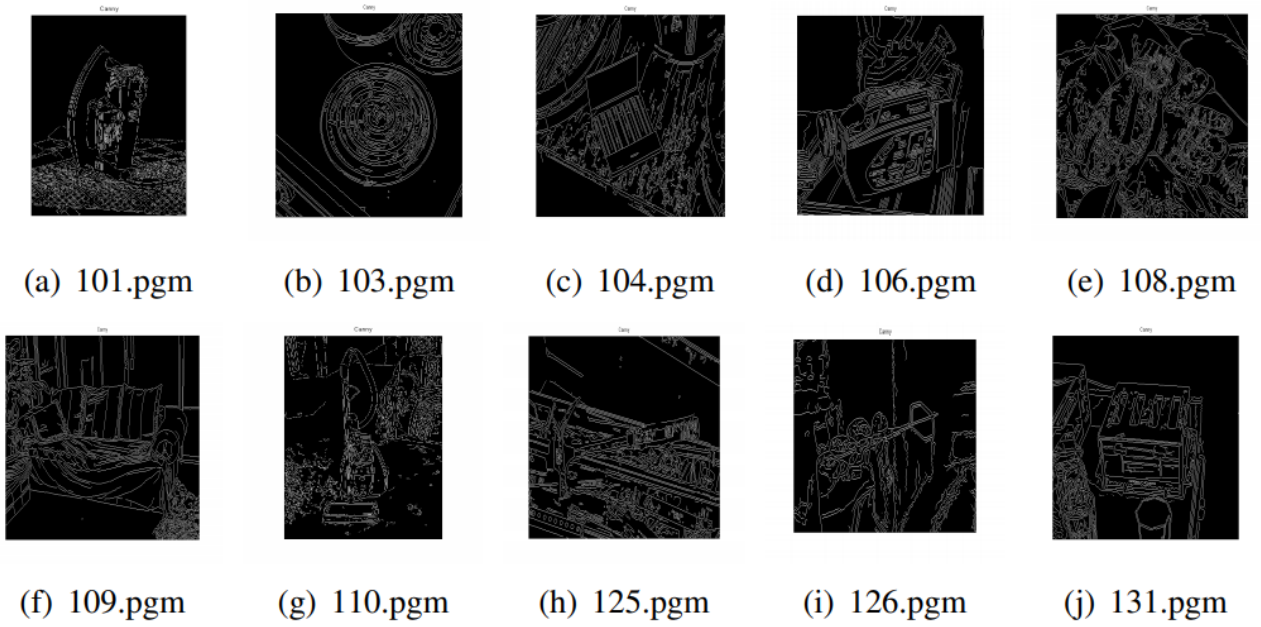
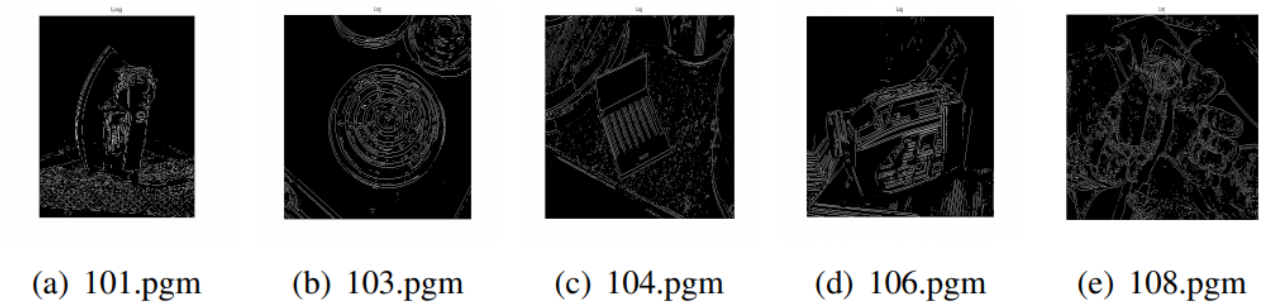
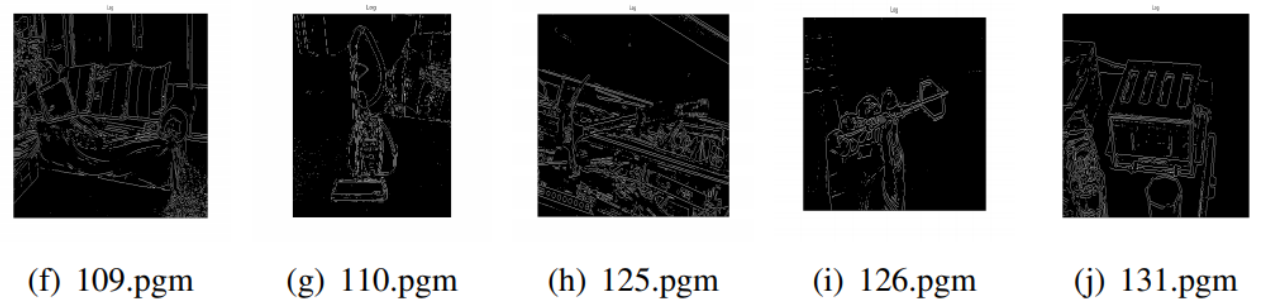


Рисунок 4.4 Детектор Кенні

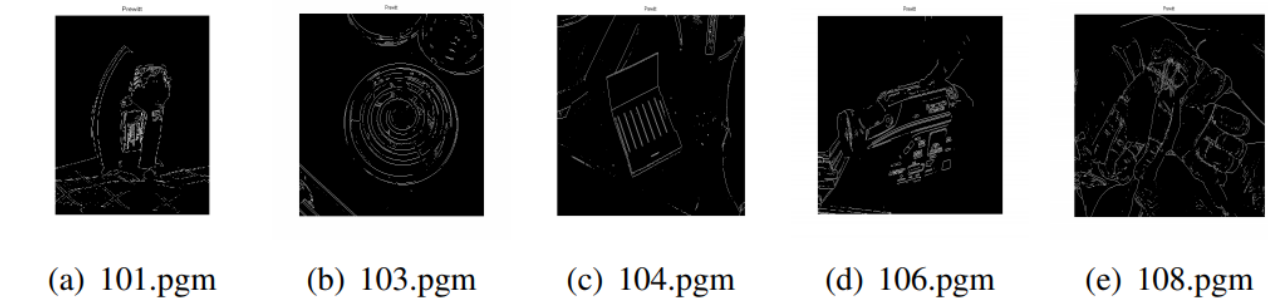


(a) 101.pgm (b) 103.pgm (c) 104.pgm (d) 106.pgm (e) 108.pgm

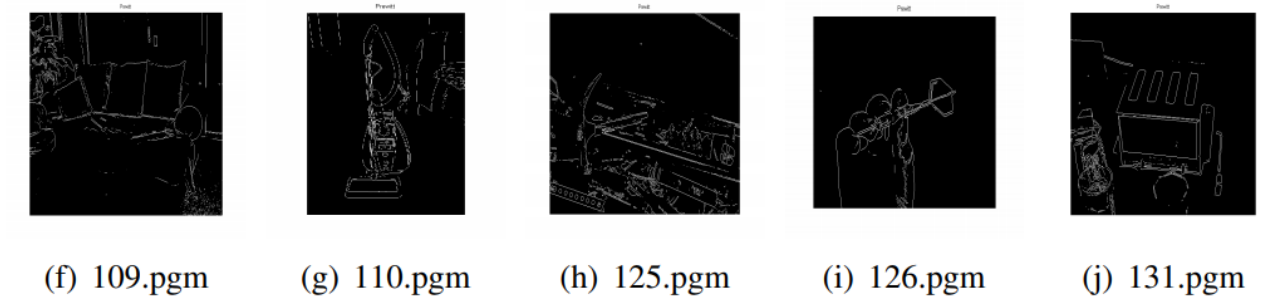


(f) 109.pgm (g) 110.pgm (h) 125.pgm (i) 126.pgm (j) 131.pgm

Рисунок 4.5 Лапласа-Гауса

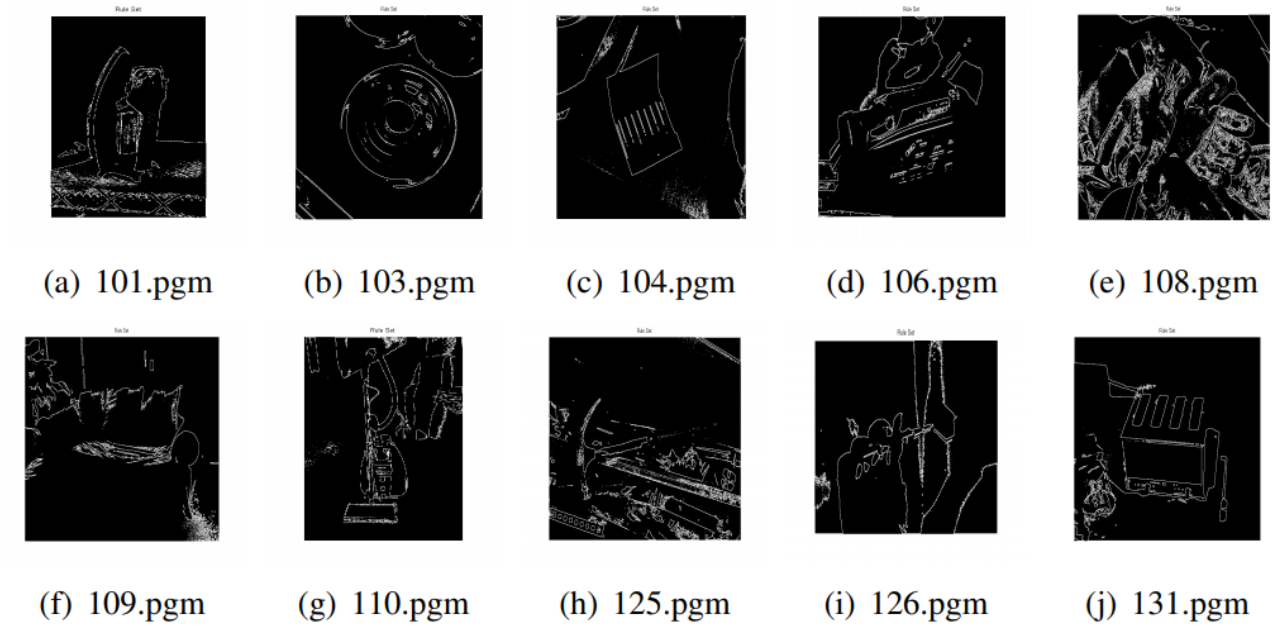


(a) 101.pgm (b) 103.pgm (c) 104.pgm (d) 106.pgm (e) 108.pgm



(f) 109.pgm (g) 110.pgm (h) 125.pgm (i) 126.pgm (j) 131.pgm

Рисунок 4.6 Собеля



(a) 101.pgm (b) 103.pgm (c) 104.pgm (d) 106.pgm (e) 108.pgm

(f) 109.pgm (g) 110.pgm (h) 125.pgm (i) 126.pgm (j) 131.pgm

Рисунок 4.7 Правила КА

Висновки

У цій роботі було досліджено клітинні автомати, їх застосування, стандартні методи обробки зображень. Були розглянуті методи навчання клітинних автоматів, здійснено пошук найкращих правил для пошуку країв.

Виявлений набір правил дуже простий, оскільки він містить лише єдине правило. Результат правил також порівнюється з стандартним детектором країв Кенні. Важливо зазначити, що мета клітинних автоматів забезпечити простоту виконання складних завдань.

Виявлення країв залежить від чутливості алгоритму Оцу. В майбутньому пропонується застосовувати інші алгоритми, в яких передбачено контроль чутливості.

Додаток А

Список використаної літератури

1. Rosin, P. L., & Xianfang, Sun. Cellular Automata as a Tool for Image Processing

2. Kusch, I., Markus M. Mollusc Shell Pigmentation. Cellular Automaton Simulations and Evidence for Undecidability

3. A.G.Hoekstra. Simulating complex systems by cellular automata.

4. Wolfram S. A. New Kind of Science

5. Базовые алгоритмы обработки изображений. Московский государственный технический университет им. Н.Э.Баумана https://studfile.net/preview/785302/page:16/

6. Laura Diosan, Anca Andreica, Alina Enescu. The use of simple cellular automata in image processing. Studia Universitatis Babes-Bolyai, Informatica . Jun2017, Vol. 62 Issue 1, p5-14. 10p.

7. Deepak, R., Nayak, Sumit K. Sahu, & Jahangir, Mohammed. A cellular automata based Optimal Edge Detection Technique using twenty-five neighborhood model. International Journal of Computer Applications. International Journal of Computer Applications (0975 – 8887) Volume 84 – No 10, December 2013

8. Sahota P., Daemi M., Elliman, D. Training genetically evolving cellular automata for image processing. In Speech, Image Processing and Neural Networks, 1994. Proceedings, ISSIPNN’94., 1994 International Symposium on (1994), IEEE, pp. 753–756.

9. Slatnia, S., Kazar, O. Images segmentation based contour using evca approach, evolutionary cellular automata.

10. https://uk.wikipedia.org/wiki/Теорема_перерахування_Пойя

11. Кліткові автомати як засіб моделювання складних систем / Т. П. Кобильник // Фізико-математична освіта. - 2018. - Вип. 4. - С. 71-75. - Режим доступу: http://nbuv.gov.ua/UJRN/fmo_2018_4_13

12. Розробка і дослідження криптографічних хеш-функцій на основі клітинних автоматів, Ю.В. Танасюк, Х.В. Мельничук, С.Е. Остапов

13. Московский государственный технический университет им. Н.Э.Баумана https://ru.bmstu.wiki/Предварительная_обработка_изображений

14. Аналіз відомих методів сегментування зображень, що отримані з бортових систем оптико-електронного спорядження, В.Г. Худов , Г.А. Кучук , О.М. Маковейчук , А.В. Крижний. Обробка інформації в складних технічних системах

15. Training Cellular Automata for Image Edge Detection. Anand Prakash Shukla. ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY Volume 19, Number 4, 2016, 338–359

16. https://uk.wikipedia.org/wiki/Метод_Оцу

17. https://uk.wikipedia.org/wiki/Оператор_Собеля
18.https://uk.wikipedia.org/wiki/Алгоритм_Кенні