НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ННК “Інститут прикладного системного аналізу”
(повна назва інституту/факультету)

Кафедра Системного проектування
(повна назва кафедри)

«До захисту допущено»

Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)

“___”_____2016 р.

Дипломна робота

першого (бакалаврського) **рівня вищої освіти**
(першого (бакалаврського), другого (магістерського))

зі спеціальності 7.05010102, 8.05010102 Інформаційні технології проектування
7.05010103, 8.05010103 Системне проектування
(код та назва спеціальності)

на тему: Застосування клітинних автоматів для обробки даних

Виконала: студентка _4_ курсу, групи __ДА-21_____
(шифр групи)

Свєтлова Олена Костянтинівна _____
(прізвище, ім’я, по батькові) (підпис)

Керівник доцент, к.т.н. Романов В.В. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант економічний проф. док. ек. н., Семенченко Н.В. _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Нормоконтроль ст. викладач Британов О.А. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2016 року

Національний технічний університет України
«Київський політехнічний інститут»

Факультет (інститут) ННК “Інститут прикладного системного аналізу”
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти Перший(Бакалаврський)
(перший (бакалаврський), другий (магістерський) або спеціаліста)

Спеціальність 7.05010102, 8.05010102 Інформаційні технології проектування
7.05010103, 8.05010103 Системне проектування
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)
«___»_____2016 р.

ЗАВДАННЯ
на дипломний проект (роботу) студенту

Свєтлової Олени Костянтинівни
(прізвище, ім’я, по батькові)

1. Тема проекту (роботи)) Застосування клітинних автоматів для обробки даних керівник проекту (роботи))_____,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від 12 травня 2016 р. № 50-ст

2. Строк подання студентом проекту (роботи) _____

3. Вихідні дані до проекту (роботи)

Зображення формату Bitmap (*.bmp)
Розмір зображення: 640×480 пікселів
Форма реалізації: у вигляді програми на Delphi.

4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити)

- Розглянути клітинні автомати та сфери їх застосування.
- Розглянути методи попередньої обробки зображень.
- Розробити алгоритм обробки зображень, використовуючи клітинні автомати.

- Виконати програмну реалізацію розробленого алгоритму.
- Проаналізувати роботу програмного продукту на основі тестових зображень.
- Виконати економічний аналіз програмного продукту

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників, плакатів тощо)

1. Блок-схема алгоритму роботи клітинного автомату – плакат.
2. Алгоритм обробки зображень, використовуючи клітинний автомат – плакат.
3. Результати роботи програмного продукту – плакат.

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний розділ	проф. док. ек. н. Семенченко Н. В.		

7. Дата видачі завдання 01.02.2016

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Отримання завдання	01.02.2016	
2	Збір інформації	15.02.2016	
3	Вивчення видів клітинних автоматів та їх застосування	28.02.2016	
4	Вивчення методів попередньої обробки зображень	10.03.2016	
5	Розробка алгоритму обробки зображень	15.03.2016	
6	Виконання програмної реалізації	01.04.2016	
7	Аналіз роботи програмного продукту	25.03.2016	
8	Оформлення дипломної роботи	31.05.2016	
9	Отримання допуску до захисту та подача роботи в ДЕК		

Студент _____ О.К. Свєтлова
(підпис) (ініціали, прізвище)

Керівник проекту (роботи) _____ В.В. Романов
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

бакалаврської дипломної роботи Свєтлової Олени Костянтинівни
на тему «Застосування клітинних автоматів для обробки даних»

Дана дипломна робота присвячена розробці програмного продукту для обробки даних. **Метою** даного дослідження є аналіз можливості застосування клітинних автоматів для обробки зображень.

В роботі розглянуті загальні відомості про клітинні автомати та методи попередньої обробки даних, розроблено алгоритм з використанням клітинного автомату для підрахунку об’єктів на зображенні та створено програмний продукт, що реалізує цей алгоритм. Також, було проведено тестування створеного програмного продукту на основі різних зображень аналізів крові.

Загальний обсяг роботи: 79 сторінок, 22 рисунків, 9 таблиць, 24 бібліографічних найменувань.

Ключові слова: клітинні автомати, обробка зображень, сегментація зображень, медіанна фільтрація, метод Оцу, підрахунок клітин.

АННОТАЦИЯ

бакалаврской дипломной работы Светловой Елены Константиновны
на тему «Применение клеточных автоматов для обработки данных»

Данная дипломная работа посвящена разработке программного продукта для обработки данных. Целью данной работы является анализ возможности применения клеточных автоматов для обработки изображений.

В работе рассмотрены общие сведения про клеточные автоматы и методы предварительной обработки данных, разработан алгоритм с использованием клеточного автомата для подсчета объектов на изображении и создан программный продукт, реализующий этот алгоритм. Также, было проведено тестирование созданого программного продукта на основе различных изображений анализов крови.

Общий объем работы: 79 страниц, 22 рисунок, 9 таблиц, 24 библиографических наименований.

Ключевые слова: клеточные автоматы, обработка изображений, сегментация изображений, медианная фильтрация, метод Оцу, подсчет клеток.

ANNOTATION

a bachelor’s degree work of Olena Svietlova

entitled "Application of cellular automata for data analysis"

This project is devoted to the research of software tools for data processing. The aim of this research is to analyze the the possibility of applying cellular automata for image processing.

The project covers the following: general information about cellular automata and data pre-processing methods; developed algorithm and software for counting objects in an image based on using cellular automata. Also, the testing of the program was carried out based on various images of blood tests.

Total volume of work: 79 pages, 22 figures, 9 tables, 24 bibliographic titles.

Keywords: cellular automata, image processing, image segmentation, median filtering, Otsu method, counting cells.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

ПП – програмний продукт.

КА – клітинний автомат

МФ – медіанний фільтр

ВСТУП

Сучасний період розвитку засобів обробки інформації характеризується масштабним впровадженням різних алгоритмів і технологій інтелектуалізації процесів обробки даних. Одним з важливих компонентів процесу інтелектуалізації інформаційних технологій є все більша необхідність використання інформації, що має форму фото- і відео- даних, зокрема цифрових зображень, оскільки такі технології найбільш прийнятних і зручні для використання в системах відеоспостереження, автофокусування в фото- і відео-камерах, медичних приладах і т.д.

Обсяг програмно-апаратних засобів, пов'язаних із захопленням, обробкою і зберіганням фото- і відео- зображень збільшується щорічно на 6-10%, що призводить до пропорційного щорічного приросту фото- і відео- зображень [1]. Тому проблема підвищення ефективності і якості обробки фото і відео зображень є актуальною і представляє безперечний інтерес.

Процес обробки зображень складається з ряду етапів, серед яких одним з найбільш важливих є попередня обробка зображень, яка представляє самостійний інтерес. Попередня обробка і виділення контурів на цифрових зображеннях мають широкий спектр застосування в різних областях, починаючи від астрономічних фотографій, електронної мікроскопії та робототехніки, і закінчуючи застосуванням алгоритмів попередньої обробки і виділення контурів в естетичних цілях, тому **тема цієї роботи є актуальною.**

Однією з найважливіших сфер застосування комп'ютерної обробки зображень є медицина. Сьогодні в медичній техніці широко застосовуються системи формування зображення, його перетворення в цифрову форму, візуалізація та документування шляхом введення в комп'ютер зображень за допомогою спеціалізованих пристроїв захоплення відео**.**

З розвитком інформаційних і комп'ютерних технологій в медицині з'явилися нові можливості для підвищення ефективності виявлення об'єктів

інтересу на зображеннях. Пошук об’єктів на зображенні є одним з ключових завдань розпізнання зорових образів. Для його вирішення розроблено велику кількість методів, один з них полягає у застосуванні клітинних автоматів. Потрібно зазначити, що область застосування клітинних автоматів майже безмежна: від найпростіших ігор до штучного інтелекту. Клітинні автомати наразі поширенні у багатьох сферах: математиці, фізиці, біології, економіці, соціології і т.д.

Метою даного дослідження є аналіз можливості застосування клітинних автоматів для обробки даних.

Для досягнення мети необхідно розв’язати задачу проектування та створення програмного комплексу, що дозволяє реалізувати виділення об’єктів на цифровому зображенні.

Об’єктом дослідження є методи цифрової обробки даних.

Предметом дослідження є алгоритми клітинних автоматів та їх придатність для сегментації зображень.

Розв’язання задачі передбачає виконання таких завдань:

1. Загальний аналіз алгоритмів клітинних автоматів та методів попередньої обробки зображень.
2. Вибір методів попередньої обробки зображень, які найбільш підходять для вирішення задачі підрахунку кількості об’єктів на цифровому зображенні.
3. Розробка алгоритму обробки зображень на основі КА з урахуванням результатів попередніх кроків.
4. Апробація розробленого алгоритму для вирішення задачі автоматизації процедури кількісного аналізу крові.

1 ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО КЛІТИННІ АВТОМАТИ

1.1. Визначення клітинних автоматів

Перед тим, як наводити математичне визначення клітинних автоматів, сформулюємо його словесно. Один із авторів цього поняття, Джон фон Нейман, дає таке:

«Клітинні автомати є дискретними динамічними системами, поведінка яких повністю визначається в термінах локальних залежностей. В значній мірі так само відбувається й тоді, коли є великий клас неперервних динамічних систем, визначених рівняннями в частинних похідних. У цьому сенсі клітинні автомати в інформатиці є аналогом фізичного поняття «поля»… клітинний автомат може представлятися як стилізований світ.

Простір представлений рівномірною сіткою, кожна клітина якої містить декілька бітів даних; час йде вперед дискретними кроками, а закони світу виражаються єдиним набором правил, скажімо, невеликою довідковою таблицею, за якою будь-яка клітина на кожному кроці обчислює свій новий стан за станами її близьких сусідів.

Таким чином, закони системи є локальними і всюди однаковими. «Локальність» означає, що для того, щоб узнати, що станеться тут через деяку мить, досить поглянути на стан найближчого оточення. «Подібність» означає, що закони скрізь одні й ті ж: я можу відрізнити одне місце від іншого лише за формою ландшафту, а не за якоюсь різницею в законах» [2].

Формально, клітинний автомат можна визначити як множину кінцевих автоматів, кожен із яких може знаходитися в одному із можливих станів $\sigma \in Z$, де Z – множина станів кожної клітини[2, 3].

Зміна стану автоматів відбувається згідно правилу переходу f : $Z \times Z^{|N|} \to Z$

наступним чином:

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

де N – множина автоматів, які складають окіл

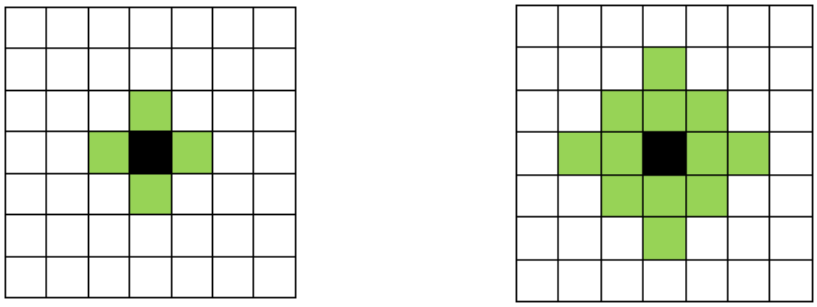
Окіл може обиратися будь-яким способом. На приклад, розглянемо найбільш відомі випадки: окіл фон Неймана (рис. 1.1) та окіл Мура (рис. 1.2).

Окіл фон Неймана радіуса 1 визначається наступним чином:

$$N^1_{фон\ Неймана}(i,j)=\{\sigma_{k,l},|i-k|+|j-l|\le 1\}=\{\sigma_{i,j},\sigma_{i-1,j},\sigma_{i,j-1},\sigma_{i+1,j},\sigma_{i,j+1}\} \tag{1.2}$$

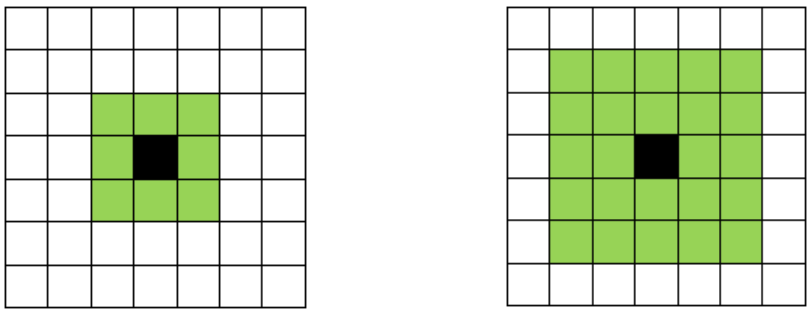
Окіл Мура радіуса 1 визначається наступним чином:

$$N^1_{Мура}(i,j)=\{\sigma_{k,l},|i-k|\le 1,|j-l|\le 1\}=\{\sigma_{i,j},\sigma_{i-1,j},\sigma_{i,j-1},\sigma_{i-1,j-1},\sigma_{i+1,j},\sigma_{i,j+1},\sigma_{i+1,j+1}\} \tag{1.3}$$



а) б)

Рисунок 1.1 – Приклади двовимірного окілу фон Неймана: а) – першого порядку б) – другого порядку [3]



а) б)

Рисунок 1.2 – Приклади двовимірного окілу Мура : а) – першого порядку б) – другого порядку [3]

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

$$n=\sigma^{\sigma^m} \quad (1.4)$$

Як бачимо з таблиці 1.1, загальна кількість правил переходу різко зростає при досить малих значеннях σ та m.

Таблиця 1.1 – Залежність кількості правил переходу від числа станів σ та кількістю сусідів m [3]

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}

Клітинні автомати в загальному випадку характеризуються наступними фундаментальними властивостями:

1. Локальність правил. На новий стан клітини можуть впливати лише елементи з її околу та іноді вона сама.
2. Однорідність системи. Жодна область решітки не може бути відрізнена від іншої за якимось відмінностями у правилах. Однак, на практиці можуть мати місце краєві ефекти, через те, що решітка є кінцевою.
3. Множина можливих станів клітини – кінцева.
4. Значення в усіх клітинах змінюються одночасно, у кінці ітерації, а не

помірі виконання. В іншому випадку, порядок перебору клітин істотно впливав би на результат.

1.2 Види клітинних автоматів

1.2.1 Класифікація С. Вольфрама

Завдання класифікації клітинних автоматів до сьогоднішнього дня залишається відкритим, про що свідчать дослідження динамічних систем, які проводяться. Найбільш поширена класифікація за типом їх еволюції, запропонована Стівеном Вольфрамом. У його книзі [4] наведено докладний опис клітинних автоматів, проведений аналіз їх поведінки на великій кількості емпіричних прикладів та запропоновані наступні чотири класи:

I клас. Еволюція системи закінчується переходом всіх клітин поля в однаковий стан. Результатом еволюції майже всіх початкових умов є швидка стабілізація стану та його гомогенність. Будь-які випадкові конструкції в таких правилах швидко зникають.

II клас. Існує багато кінцевих станів, але всі вони складаються з набору простих структур, які залишаються незмінними або повторюються через деякий невеликий число кроків. Більшість випадкових структур в початкових умовах швидко зникає, але деякі залишаються. Локальні зміни в початкових умовах надають локальний характер на подальший хід еволюції системи.

III клас: Поведінка складне, у багатьох відношеннях виглядає хаотично. Структури змінюються з певним періодом та залежать від початкової конфігурації. Будь-які стабільні структури, які виникають майже відразу ж знищуються оточуючим їх шумом. Локальні зміни в початкових умовах надають широкий, невизначений вплив на хід усієї еволюції системи.

IV клас: Суміш хаосу і порядку: породжуються локальні структури, які переміщаються і взаємодіють один з одним дуже складним

чином. Результатом еволюції майже всіх правил є структури, які взаємодіють складним і цікавим чином з формуванням локальних, стійких структур, які здатні виживати тривалий час. В результаті еволюції правил цього класу можуть виходити деякі послідовності II класу, описаного вище. Локальні зміни в початкових умовах надають широке, невизначені вплив на хід усієї еволюції системи.

Віднесення конкретного клітинного автомата до якого-небудь з класів є досить складним, так як не вказано, за яких початкових умовах очікується вказана вище поведінка. Передбачається, що клас слід вибирати по найбільш складній поведінці, яку вдається отримати.

У книзі наведені приклади типових представників кожного класу, але класифікація не проведена повністю ні для якої з областей клітинних автоматів. Виконати це за розумний час не представляється можливим навіть для області одновимірних двійкових клітинних автоматів через необхідність великого обсягу обчислень і складного аналізу.

1.2.2 Ймовірнісний метод класифікації клітинних автоматів

Лебедєв А. в роботі [5] запропонував імовірнісний метод класифікації двовимірних двійкових клітинних автоматів. Класи виділяються на основі ймовірності переходу випадкової клітини в стан 1 на першому кроці еволюції при випадкових початкових умовах.

Для характеристики ймовірності використовується термін “сприятливість”, клітинні автомати називаються іграми.

Виділяється п'ять класів ігор:

1. Строго сприятливі.
2. Сприятливі в середньому.
3. Нейтральні в середньому.
4. Несприятливі в середньому.
5. Строго несприятливі.

Причому всі системи, що відносяться до класу 1, також відносяться і до

класу 2, а ті, що належать до класу 5 – також входять і в клас 4.

Як єдиний приклад в роботі наводиться гра "Життя", що відноситься за цією класифікацією тільки до класу 4.

В роботі [6] розробляється аналогічний підхід до класифікації, але з використанням більш складного математичного апарату.

До недоліків цього підходу до класифікації клітинних автоматів можна віднести те, що класи характеризують лише перший крок еволюції і погано відображають складність і особливості поведінки клітинних автоматів.

1.2.3. Класифікація А. Вюнше

У роботі [7] А. Вюнше пропонує для класифікації клітинних автоматів використовувати автоматизоване знаходження локальних структур (використовується термін глайдер аналогічно структурам, характерним для гри "Життя").

Для такого автоматизованого знаходження використовуються оцінки ентропії на кожному кроці системи при випадковому початковому стані і фільтрація локальних структур.

В результаті такої класифікації виходять не чіткі класи, а деякі величини, що характеризують процеси, які відбуваються.

Пропонується виділити три області:

1. Впорядковані системи.
2. Хаотичні системи.
3. Системи зі складною поведінкою.

До переваг цього методу відноситься можливість повністю автоматизувати процес визначення класу і простоту реалізації.

Недолікам є відсутність можливості відділення складності початкового стану від складності поведінки системи.

1.2.4 Класифікація на основі породжуваних структур

В роботі [8] проведено класифікацію структур, породжуваних одновимірними двійковими клітинними автоматами з точкового зародка. Клітинні автомати можна класифікувати на основі структур, які вони породжують з деякого заданого початкового стану.

Перевагою такої класифікації є надзвичайна простота і висока швидкість визначення класу клітинного автомата.

Головний недолік – отримана таким чином класифікація буде мати дуже низьку практичну корисність, так як такий підхід лише поверхово зачіпає властивості самого клітинного автомата.

Наприклад, в цьому випадку гра "Життя" (широко відомий двомірний двійковий клітинний автомат, універсальність якого доведена) виявиться еквівалентним клітинному автомату, що переводить клітини з будь-якого стану в нульовий (самому найпростішого з можливих клітинних автоматів).

1.3 Сучасне застосування клітинних автоматів

1.3.1 Загальний огляд областей застосування

Область застосування клітинних автоматів майже безмежна: від найпростіших ігор до штучного інтелекту. Автор гри “Життя” Конуей вважав, що наш всесвіт можна уявити клітинним автоматом, який керує рухом елементарних часток відповідно деяких правил.

Клітинні автомати наразі поширенні у багатьох сферах: математиці, фізиці програмуванні, біології, економіці, соціології і т.д. При їх використанні з успіхом були вирішені наступні задачі моделювання:

- утворення спор
- розвиток дислокацій
- поширення теплових потоків
- зростання дендритів

- опис руху натовпу

У наступних пунктах буде розглянуто декілька найбільш цікавих сфер застосування клітинних автоматів.

1.3.2 Обробка зображень замінованих полей

У роботі [9] розглянута життєво важлива задача ідентифікації мінних полів за результатом отриманих знімків поверхні (рис. 1.3).

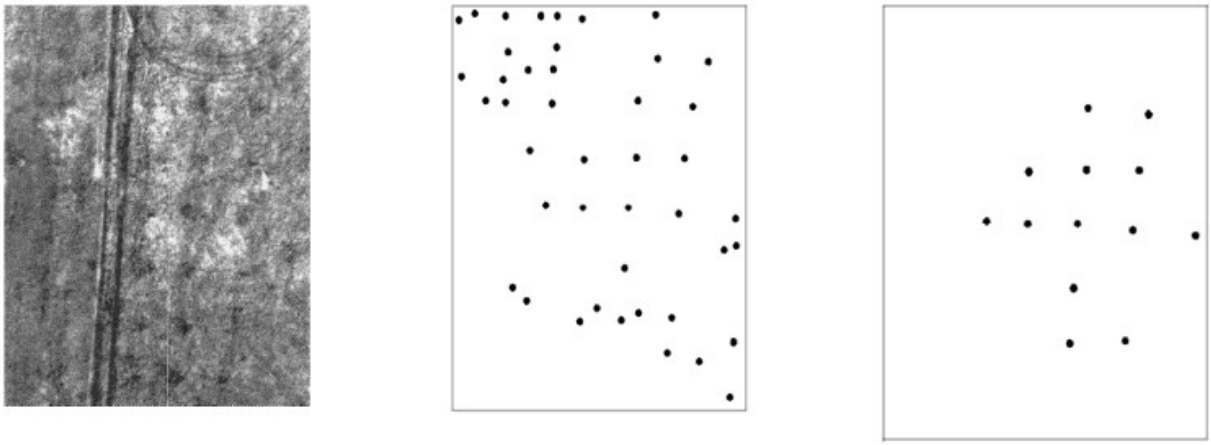
Особливістю цієї задачі є необхідність виділення малих за розміром об’єктів, зображення яких залежить від типу мін, способів їх встановлення, умовах спостереження та типу реєструвальних датчиків. При цьому фотозйомка може відбуватися з різних носіїв: космічний апарат, літак, дистанційно пілотований літальний апарат.

На основі аналізу аерокосмічного знімку (рис. 1.3 – а), автори розробили досить складний алгоритм, головні пункти якого є наступними:

1. “Розмиття яскравості” (низькочастотна фільтрація зображення). Використовувалися такі фільтри, як низькочастотний фільтр Гауса, рівномірно згладжуючий фільтр та інші. На цьому етапі видаляються дрібні перешкоди, але й контури потрібних об’єктів розмиваються.
2. Контрастування зображення з допомогою диференціальних фільтрів, робота яких заснована на локальній різниці яскравості.
3. Бінаризація зображення за допомогою спеціальних алгоритмів, детально описаних у роботі [9].
4. Виділення замкнених областей (кластерів). Саме на цьому етапі використовується клітинний автомат. Було реалізовано метод “зараження клітини”, коли точка (клітина) заражається (привласнює номер свого кластеру) усім сусідам, з якими вона контактує. Якщо точка відноситься до декількох кластерів, ці кластери об’єднуються. Після того, як будуть виявлені усі кластери, знаходяться координати, які дозволяють судити о розмірі та формі кластерів. Кластери, що мають форму та розмір, які

суттєво відрізняються від аналогічних характеристик потрібних об’єктів, видаляються. У результаті ми отримуємо об’єкти, що потенційно можуть бути мінами (рис. 1.3 – б).

5. Розпізнавання положення мін. В залежності від способу встановлення мін, існують різні конфігурації. У цьому дослідженні розглядаються клітинне поле.



а) б) в)

Рисунок 1.3 – Приклад виявлення мін : а) – знімок з космосу замінованого поля б) – нормалізовані за розміром елементи зображення в) – ідентифіковане заміноване поле [9]

1.3.3 Моделювання поведінки натовпу

У сучасних містах досить часто відбуваються різні масові заходи, під час яких одним з головних питань є безпека руху людей при наявності перешкод.

Автор роботи [10] порівнює цю проблему з певним класом задач газодинаміки (рух людей містить як хаотичну, так й направлену складові, аналогічно руху молекул у зовнішньому полі). Беручи до уваги методи вирішення схожих задач, у яких застосовувались клітинні автомати, можна використати їх й для моделювання руху людей.

Таким чином, була створена наступна модель:

- ортогональна сітка, що використовується в якості поля даного клітинного автомата, задає чотири можливі напрямки руху (уздовж ліній сітки)
- кожна людина в групі намагається рухатися у певному напрямі (одному для усіх). При неможливості руху у цьому напрямі, великої кількості людей або наявності перешкод, що не подолати, людина обирає такий напрямок, на якому присутність перешкод мінімальна.
- кожна людина може оглядати ситуацію на певній відстані (був обраний окіл фон Неймана).

Схожий принцип використав й Плотніков під час створення програми [11], що дозволяє промоделювати усі найбільш ймовірні сценарії розвитку подій при натовпі. Створена програми дозволяє задати перешкоди, загрози та місцевість, на якій буде проведено моделювання.

На рис. 1.4 зображений приклад роботи програми. Люди знаходяться у певній частині міста і в певний момент часу відбувається подія: поява терориста. Усі починають розбігатися, але прохід досить вузький, через що може відбутися давка.

Рисунок 1.4 – Приклад поведінки людей при виникненні небезпеки (позначені чотирма світлими клітинами, що обведені) [11]

1.3.4 Застосування клітинних автоматів в криптографії

Повсякденне життя людини XXI століття неможливо уявити без смартфонів, ноутбуків, плеєрів та інших важливих пристроїв, на яких може знаходитися досить конфіденційна інформація, що потребує шифрування.

Сучасні криптосистеми для досягнення високої надійності, як правило, застосовують прості криптографічні перетворення, запропоновані ще Клодом Шеноном. Однак, окрім широко поширених алгоритмів, існують й інші підходи, у тому числі з використанням клітинних автоматів.

Є два основних метода їх використання в симетричних криптосистемах [12]:

1. Алгоритм шифрування базується виключно на перетвореннях по правилам клітинних автоматів.
2. Клітинний автомат використовується в якості одного з елементів криптосистеми на окремому етапі шифрування.

1.4 Висновки

У першому розділі були отримані наступні результати:

1. Наведено математичне визначення клітинних автоматів.
2. Розглянуто різновиди клітинних автоматів та способи їх класифікації.
3. Розглянуті сучасні сфери застосування клітинних автоматів та наведені приклади реальних робіт на їх основі.

2 ОГЛЯД МЕТОДІВ ПОПЕРЕДНЬОЇ ОБРОБКИ ЗОБРАЖЕНЬ

2.1 Огляд найбільш поширених методів попередньої обробки зображень

Операції розпізнавання на зображеннях певних об'єктів, як правило, передує обробка зображень для створення умов, що підвищують ефективність і якість виділення і розпізнавання шуканих або досліджуваних об'єктів. Методи попередньої обробки залежать від завдань досліджень та є досить різноманітними [13].

Задача сегментації і розпізнавання зображень технічно комплексна. Вона ділиться на кілька великих сегментів (підзадач):

Рисунок 2.1 – Етапи роботи алгоритмів розпізнавання зображень [13]

Основне завдання розпізнавання: зрозуміти, чи стосується дані на зображенні до класу шуканих об'єктів. Основні технічні складнощі, що виникають в даному випадку:

1. Зображення пред'являються на складному тлі;
2. Шукані області мають складну геометрію;
3. Вхідні дані мають шум або дезорієнтуючу інформацію;

4. Різні частини зображення мають різні характеристики (підсвічування, освітленість, перешкоди).

Для вирішення завдання на різних етапах застосовують різні підходи і методи сегментації, нормалізації і розпізнавання. На на рис. 2.2. представлені основні методи, що існують і застосовуються на сьогоднішній день (методи, які мають підметоди, позначені жирною рамкою):

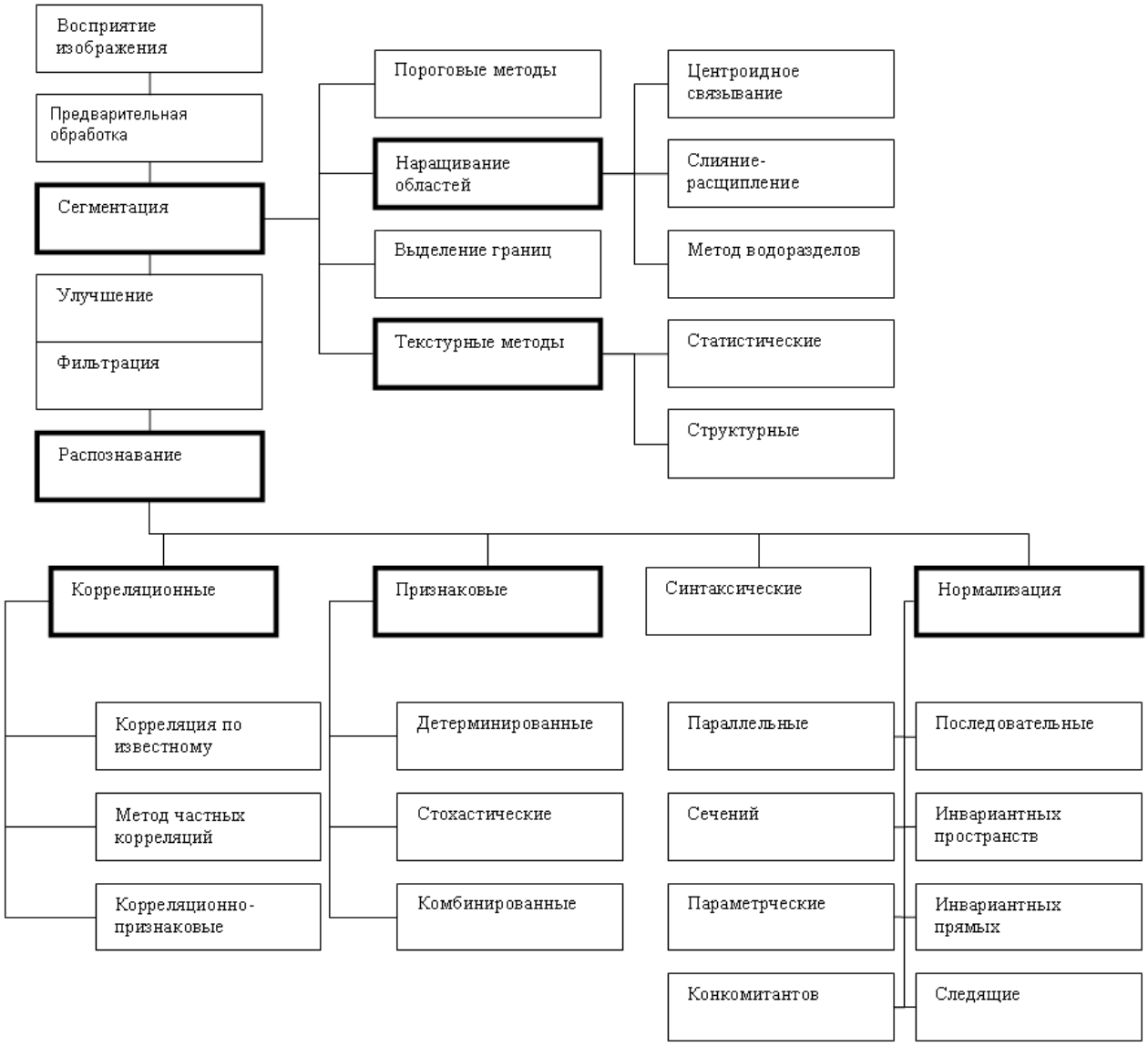


Рисунок 2.2 – Найбільш поширені методи, що застосовуються при розпізнаванні зображень [13]

Розглянемо основні методи більш детально:

1. **Попередня обробка** – застосовується практично завжди після зняття інформації з відеодатчика і має на меті зниження перешкод на зображенні, що виникли в результаті дискретизації і квантування, а також придушення зовнішніх шумів. Як правило, це операції усереднення і вирівнювання гістограм.
2. **Сегментація** – процес пошуку однорідних областей на зображенні. Цей етап дуже важкий і в загальному вигляді не алгоритмізований для довільних зображень. Найбільш поширені методи сегментації, засновані на визначенні однорідних кольорів або текстур.
3. **Поліпшення / фільтрація** – можуть використовуватися після проведення сегментації, і переслідує ту ж мету що і попередня обробка: зниження перешкод на зображенні, що виникли в результаті дискретизації і квантування, а також придушення зовнішніх шумів.
4. **Розпізнавання** – найчастіше кінцевий етап обробки, що лежить в основі процесів інтерпретації і розуміння. Вхідними для розпізнавання є зображення, виділені в результаті сегментації і, частково, відреставровані. Вони відрізняються від еталонних зображень геометричними і яскравості спотвореннями, а також збереглися шумами.

Основні види методів сегментації зображень

Кінцевий результат аналізу зображень багато в чому визначається якістю сегментації, а ступінь деталізації характеристик, що виділяються залежить від конкретного завдання. Тому не існує окремого методу або алгоритму відповідного для вирішення всіх типів завдань сегментації, кожен з методів має свої переваги і недоліки. У більшості випадків вибирається один або кілька алгоритмів, і модифікуються під специфічні умови задачі [14].

Сегментація вирішує в загальному сенсі дві основні задачі [14]:

- поділ зображення на частини, для здійснення подальшого аналізу;
- зміна форми опису елементів зображення, що дозволяє уявити точки як

високорівневі структури, що забезпечують ефективність подальшого аналізу зображення.

Існують різні класифікації методів, але більшість з них ґрунтуються на двох наступних властивостях сигналу яскравості – це розривність і однорідність.

До методів на основі розривності яскравості відноситься виявлення точок ліній і перепадів. При виявленні точок і ліній за допомогою спеціальних масок організовується відповідний пошук. В якості методів виявлення перепадів використовуються похідні і градієнти від функцій яскравості, такі методи засновані на більш загальних ідеях [14].

У теперішній час до основних методів сегментації зображень відносяться такі класи (методи згруповані від найпростіших до найбільш трудомістким)[14]:

1. Морфологічні методи – застосовуються в основному для роботи з двійковими (чорно-білими) зображеннями. Ці методи дозволяють отримувати компоненти зображення, які в наслідку можуть використовуватися для подання форми об'єкту.
2. Граничні методи – мають інтуїтивно зрозумілі властивості і прості в реалізації. Існують кілька основних видів порогової сегментації, але базовими є тільки два: метод з оптимальним порогом і метод з адаптивним порогом. Всі інші методи цього класу є похідними від двох згаданих алгоритмів. Більш детально вони розглянуті у розділі 2.3.
3. Методи нарощування областей – представляють собою алгоритми, які рекурсивно виконують процедуру угруповання пікселів в області за заздалегідь заданими критеріями. Одним з основних методів тут є метод вододілів.
4. Текстурні методи – спираються при аналізі на дифузні (колір, відбивна здатність) властивості поверхні аналізованого об'єкта. Представлені в цій категорії методи є наборами складних операторів, які здатні звести процес розпізнавання поверхонь до простого завдання розрізнення рівнів яскравості.

Фільтрація зашумлених зображень

Цифрові зображення можуть бути чутливими до різних типів шумів, які можуть виникати від способу отримання зображень, технологій передачі інформації, методів оцифровування даних. Процес усунення різних видів шумів на зображеннях, називається фільтрацією.

При здійсненні фільтрації характеристики яскравості кожної точки зображення у цифровому форматі, замінюються іншим значенням яскравості, яке визнається в найменшій мірі спотвореним перешкодою. Виділяють частотну і просторову фільтрацію.

Частотні методи перетворень зображень ґрунтуються на ідеї Фур'є перетворення, сенс якого полягає в поданні вихідної функції у вигляді суми тригонометричних функцій різних частот, помножених на задані коефіцієнти. У разі, якщо функція є періодичною таке уявлення називається рядом Фур'є. Інакше, неперіодичних функція, що має кінцеву площу під графіком, може бути виражена у вигляді інтеграла від тригонометричних функцій, помножених на деяку вагову функцію. Такий варіант називається перетворенням Фур'є і в більшості практичних задач виявляється більш корисним, ніж ряд Фур'є.

Важливою властивістю є те, що функцію, представлену Фур'є-перетворенням, після здійснення над нею перетворень можна назад повернути до початкового стану. Таким чином, даний підхід дозволяє обробляти функцію в частотній області, після чого без втрати інформації повернутися до початкового стану. Для вирішення завдань фільтрації зображень перетворення Фур'є також можуть застосовуватися.

У практичному застосуванні реалізація частотних підходів може бути аналогічна просторовим методам фільтрації [15].

Цифрові зображення можуть бути чутливими до різних типів шумів, які можуть виникати від способу отримання зображень, технологій передачі інформації, методів оцифровування даних.

Просторові методи поліпшення зображень застосуються до растрових зображень, представлених у вигляді двовимірних матриць. принцип

просторових алгоритмів полягає в застосуванні спеціальних операторів до кожної точки вихідного зображення. В якості операторів виступають прямокутні або квадратні матриці звані масками, ядрами або вікнами. Найчастіше маска являє собою невеликий двовимірний масив, а методи поліпшення, що базуються на такому підході, часто називають обробкою по масці або фільтрацією по масці. Просторова фільтрація є кращим рішенням в тих випадках, коли є присутнім тільки адитивна складова шуму.

Застосовуються різні фільтри:

1. Фільтр, заснований на обчисленні середнього арифметичного
2. Фільтр, заснований на обчисленні середнього геометричного
3. Фільтри, засновані на обчисленні середнього гармонійного
4. Фільтри, засновані на порядкових статистиках
5. Медіанний фільтр.
6. Фільтри, засновані на обчисленні максимуму і мінімуму
7. Фільтри, засновані на виборі середньої точки

Потрібно зазначити, що при вирішенні задач усунення шуму медіанний фільтр є часто більш ефективним, ніж звичайне усереднення, так як призводить до менших спотворень кордонів виділених об'єктів, саме тому цей фільтр було використано у цій роботі.

Основні типи методів розпізнавання зображень

Для реальних завдань розпізнавання застосовуються, в основному, чотири підходи, що використовують методи:

1. Кореляційні, засновані на прийнятті рішень за критерієм близькості з еталонами. Тобто, вони засновані на пошуку максимальної кореляції інтенсивностей точок шаблону і області зображення.
2. Ознакові, що засновані на виділенні і зіставленні характерних ознак шуканого об'єкта з областю зображення, яка має ці ознаки.
3. Синтаксичні – найменш трудомісткі ;
4. Нормалізації, що займають проміжне положення за обсягом обчислень.

Кожен з підходів в розпізнаванні має право на існування. Більш того, в

рамках кожного підходу є свої конкретні алгоритми, що мають певну область застосування, яка залежить від характеру відмінностей вхідних і еталонних зображень, від завадової обстановки в поле зору, вимог до обсягів обчислень і швидкості прийняття рішень [15].

Кореляційні методи знайшли широке застосування при виявленні і розпізнаванні зображень в системах навігації, спостереження, промислових роботів. При повністю заданому стандарті багатокрокова кореляція шляхом сканування вхідного поля зору є по суті повним перебором в просторі сигналів. Тому цю процедуру можна вважати базовою, потенційно найбільш завадостійкою, хоча і найбільш трудомісткою. Всі інші методи спрямовані на скорочення обчислювальних витрат при спробі забезпечення наперед заданої надійності розпізнавання.

Ознакові і синтаксичні методи – найбільш розроблені в теорії розпізнавання образів. Вони засновані як на статистичних, так і детермінованих підходах. Головні труднощі в ознакових методах становить вибір ознак. При цьому виходять з природних правил:

a) ознаки зображень одного класу можуть відрізнятися лише незначно (за рахунок впливу перешкод);
b) ознаки зображень різних класів повинні мати відчутні відмінності;
c) набір ознак повинен бути мінімально можливим, тому що від їх кількості залежить і надійність, і складність обробки.

У першому наближенні синтаксичні методи можна віднести до простору ознак, тому що вони засновані на отриманні структурно-лінгвістичних ознак, коли зображення дробиться на частини – непохідні елементи (ознаки). Вводяться правила з'єднання цих елементів, однакові для еталона і вхідного зображення. Аналіз отриманої таким чином граматики забезпечує прийняття рішень.

Алгоритми кореляційно-ознакових і ознакових методів досить близькі і перші можна розглядати як окремий випадок друге. Хоча ознакові методи і не володіють таким завадозахистом як чисто кореляційні або методи приватних

кореляцій (в сигнальних просторах), проте, завдяки меншим трудовитрат, їх застосування корисно на першому етапі. При цьому частина вхідних зображень, ознаки яких не відповідають жодному з еталонів, відкидаються відразу.

Методи нормалізації при розпізнаванні займають проміжне місце між сигнальними кореляційними і ознаковими алгоритмами. На відміну від ознакових методів, при нормалізації зображення не "втрачається", а тільки заміщається зображенням того ж класу еквівалентності. У той же час, на відміну від кореляційних методів, безліч вхідних зображень замінюється безліччю нормалізованих зображень.

Суть нормалізації полягає в автоматичному обчисленні невідомих параметрів перетворень, яким піддані вхідні зображення, і подальшому приведенні їх до еталонного вигляду. Процедура перетворень проводиться за допомогою операторів нормалізації, а обчислення параметрів виконується функціоналами [17].

2.2 Медіанний фільтр

Визначення медіани

Нехай $a_1, a_2, ..., a_{2k+1}$ – набір з непарної кількості чисел. Переставимо ці числа в порядку незростання (неспадання). Медіаною набору називають число, яке виявиться на $(k+1)$-му місці. Надалі будемо позначати це число через $m(a_1, a_2 ... a_{2k+1})$. У випадку, якщо набір містить парну кількість елементів, медіаною будемо називати середнє арифметичне чисел a_k та a_{k+1}.

Медіанний фільтр – це віконний фільтр, що послідовно проходить масив та повертає на кожному кроці один з елементів, що потрапив до вікна фільтрації. Розглянемо нескінченну в обидві сторони послідовність чисел $...x_{-2}, x_{-1}, x_0, x_1, x_2, ...$ Зафіксуємо натуральне число k. Нова послідовність $...y_{-2}, y_{-1}, y_0, y_1, y_2 ...$ **називається** отриманою зі старої (вхідної) шляхом медіанної фільтрації, якщо $y_n = m(x_{n-k}, x_{n-k+1}, ..., x_n, ..., x_{n+k})$. Число $2k+1$ має назву

ширина вікна фільтрації [18, 19].

Властивості медіанного фільтру

Позитивні властивості медіанного фільтру:

- простота структури, що дозволяє легко реалізувати фільтр як апаратними, так і програмними засобами;
- медіанний фільтр не впливає на ступінчасті і пилкоподібні функції;
- цей фільтр добре пригнічує поодинокі імпульсні перешкоди (випадкові шумові викиди звітів і промахи);
- концепцію медіанного фільтра легко узагальнити на два виміри, застосовуючи двомірне вікно бажаної форми;
- фільтр зберігає різкі контури об’єктів;
- фільтр дозволяє провести розділення об’єктів, які суттєво різні за розміром.

Крім позитивних рис, ***у медіанного фільтра є й свої недоліки:***

- складність математичного аналізу характеристик за рахунок нелінійності (медіана суми двох довільних послідовностей не дорівнює сумі їх медіан);
- фільтр викликає сплощення вершин трикутних функцій;
- пригнічення білого і гаусового шуму менш ефективно, ніж у лінійних фільтрів;
- при збільшенні розмірів вікна фільтра відбувається розмиття крутих змін сигналу і стрибків;

Недоліки методу можна зменшити, якщо застосовувати медіанну фільтрацію з адаптивною зміною розміру вікна фільтра в залежності від динаміки сигналу і характеру шумів (адаптивна медіанна фільтрація) [17, 18].

Обробка зображень за допомогою медіанного фільтру

Медіанний фільтр досить часто застосовується на практиці для попередньої обробки цифрових зображень. Фундаментальною проблемою в області обробки зображень є ефективне видалення шуму при збереженні важливих для подальшого розпізнавання частин зображення. Складність

вирішення даного завдання істотно залежить від характеру шумів.

Медіанна фільтрація зображень найбільш ефективна, якщо шум на зображенні має імпульсний характер і являє собою обмежений набір пікових значень на фоні нулів. В результаті застосування медіанного фільтру похилі ділянки і різкі перепади значень яскравості на зображеннях не змінюються. Це дуже корисна властивість саме для зображень, на яких контури несуть основну інформацію.

При медіанної фільтрації зашумлених зображень ступінь згладжування контурів об'єктів безпосередньо залежить від розмірів вікна фільтра та його форми. Приклади форм вікна наведені на рис. 2.3. При малих розмірах вікна краще зберігаються контрастні деталі зображення, але в меншій мірі пригнічується імпульсні шуми. При великих розмірах вікна спостерігається зворотна картина. Оптимальний вибір форми вікна залежить від специфіки розв'язуваної задачі і форми об'єктів. Особливе значення це має для завдання збереження перепадів (різких кордонів яскравості) в зображеннях [19].

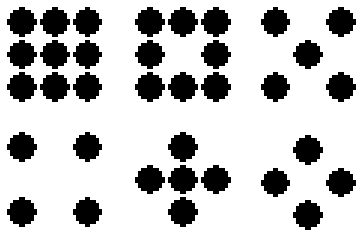


Рисунок 2.3 – Приклад форми вікна [19]

Щоб узагальнити концепцію медіанного фільтру на два виміри поступимо наступним чином:

- розіб'ємо чорно-білу фотографію на маленькі квадратики;
- зіставимо фотографії прямокутну таблицю чисел $x_{i,j}$ рівнів яскравості у кожному квадратику.

Тепер замість зображення ми маємо справу з таблицею чисел. А до них

можна застосувати процедуру медіанної фільтрації. Робиться це так. Виберемо певну форму вікна (коло, квадрат, хрест, кільце і т. д., приклади наведено на рис. 2.1) і накладемо її на зображення. Значення яскравості в центрі вікна замінюється медіаною чисел $x_{i,j}$, для усіх точок, що потрапили у це вікно [17].

Розглянемо приклад. Нехай $x_{i,j}$ дорівнює 100, якщо *i* та *j* обидва діляться на 10, і дорівнює 0 у противному випадку. Після медіанної фільтрації таке зображення буде складатися з одних нулів.

З цього прикладу випливає, що при обробці реальних зображень дуже темні і дуже світлі цятки повинні зникнути (рис. 2.4):

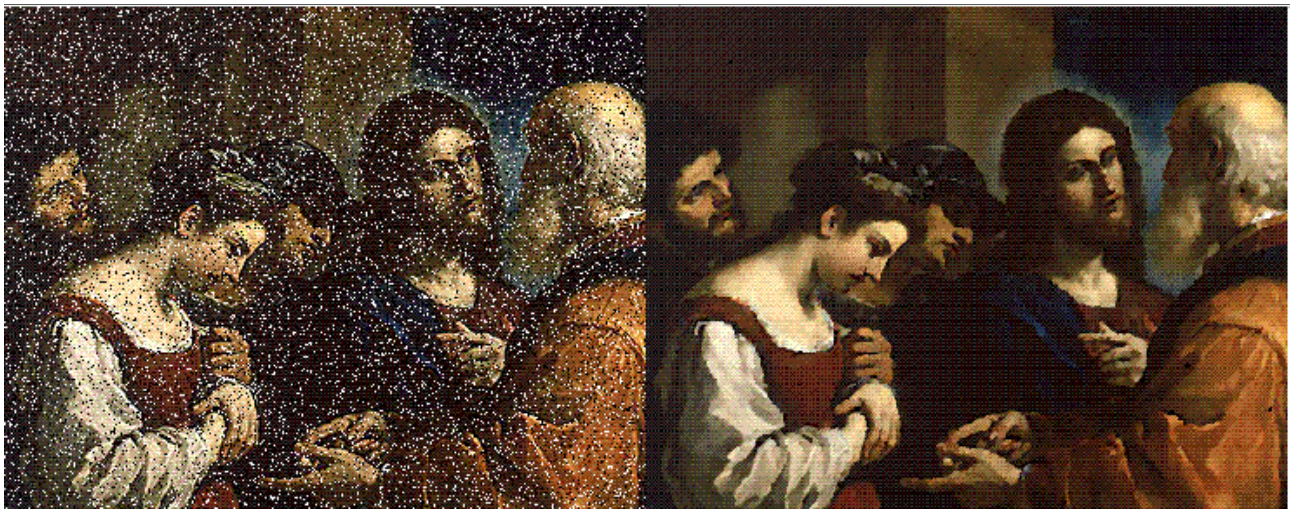


Рисунок 2.4 – Приклад застосування медіанного фільтру [20]

2.3 Огляд методів бінаризації зображення

2.3.1 Загальні відомості про бінаризацію зображень

Бінаризація зображень є однією з найбільш поширених задач під час їх обробки. З нею стикаються у багатьох сферах, наприклад:

- при обробки результатів, отриманих завдяки різного роду медичних приладів;
- при розробці алгоритмів комп’ютерного зору у робототехніці;

- у системах спостереження за рухом об’єктів у просторі.

При бінаризації зображення яскравість кожного пікселя $B(x, y)$ порівнюється з граничним значенням яскравості $B_T(x, y)$: якщо значення яскравості пікселя вище значення яскравості порогу, то на бінарному зображенні відповідний піксель буде «білим», або «чорним» в іншому випадку. Необхідність усунення великого числа помилок процесу бінаризації спричинила за собою появу великої кількості методів бінаризації, які діляться на дві групи за принципом побудови порогової поверхні: методи глобальної та локальної бінаризації. Пороговою поверхнею є матриця розмірністю $M \times N$, що відповідає розмірності вихідного зображення, кожна клітинка матриці задає поріг яскравості бінаризації для відповідного пікселя на вихідному зображенні. У методах глобальної бінаризації порогова поверхня є площиною з постійним значенням порогової яскравості, а в методах локальної бінаризації значення порогової яскравості змінюється від точки до точки зображення, і розраховується на основі деяких локальних ознак в оточенні пікселя [21].

2.3.2 Метод Оцу

Найбільш ефективним з методів глобальної бінаризації, як за якістю (помилок до 30% і менше), так і за швидкістю обробки є метод Оцу [21]. До його недоліків відноситься розмиття ліній, «злипання» об'єктів, особливо в місцях перетинань, втрата тонких ліній.

Метод використовує гістограму розподілу значень яскравості пікселів растрового зображення. Будується гістограма за значеннями $p_i = n_i / N$, де N – це загальна кількість пікселів на зображенні, n_i – це кількість пікселів з рівнем яскравості i. Діапазон яскравостей ділиться на два класи за допомогою порогового значення рівня яскравості t, де t приймає цілі значення від 0 до L.

Кожному класу відповідають ймовірності (ω_1 та ω_2) і середні арифметичні класів (μ_1 та μ_2) [21]:

$$\omega_1(t) = \sum_{i=1}^{k} p_i \tag{2.1}$$

$$\omega_2(t) = \sum_{i=k+1}^{L} p_i = 1 - \omega_1(t) \tag{2.2}$$

$$\mu_1(t) = \sum_{i=1}^{k} \frac{ip_i}{\omega_1} \tag{2.3}$$

$$\mu_2(t) = \sum_{i=k+1}^{L} \frac{ip_i}{\omega_2} \tag{2.4}$$

Метод Оцу шукає поріг, що зменшує дисперсію всередині класу, яка визначається як зважена сума дисперсій двох класів:

$$\sigma_\omega^2(t) = \omega_1(t)\sigma_1^2(t) + \omega_2(t)\sigma_2^2(t) \tag{2.5}$$

Оцу показав, що мінімізація дисперсії всередині класу – це те ж саме, що і максимізація дисперсії між класами:

$$\sigma_b^2(t) = \sigma^2 - \sigma_\omega^2(t) = \omega_1(t)\omega_2(t)\left[\mu_1(t) - \mu_2(t)\right]^2 \tag{2.6}$$

2.3.3 Метод Бернсена

Цей метод часто використовується для схематичних і картографічних зображень. Для кожного пікселя (x, y) вибирається поріг яскравості:

$$B_T(x, y) = (B_{\min} - B_{\max})/2, \tag{2.7}$$

де $B_{\min}$ та $B_{\max}$ – відповідно, найнижчий і найвищий рівень яскравості пікселів з квадратного оточення пікселя (x, y).

Якщо рівень контрасту (різниця найвищого і найнижчого рівнів) перевищує певний поріг, то піксель вважається або білим, або чорним. Для всього зображення цей поріг контрасту є константою і повинен підбиратися інтерактивно.

Метод має ряд недоліків: після обробки монотонних областей яскравості формуються сильні паразитні перешкоди, що в деяких випадках призводить до появи помилкових чорних плям. Недоліки можуть бути компенсовані за допомогою додаткової обробки – постпроцесингу. Метод є найбільш швидким серед інших, навіть в сукупності з етапом постпроцесингу [21].

2.3.4 Метод Ейквеля

Це один з найбільш продуктивних методів. Його часто застосовують для обробки чітких і контрастних зображень [21].

Відповідно до даного методу, зображення обробляється за допомогою двох концентричних вікон: маленького *S*, і великого *L*. Зазвичай форма вікон приймається квадратною. Обидва вікна послідовно зліва направо зверху вниз накладаються на зображення з кроком, що дорівнює стороні маленького вікна *S*. Для вікна *L* розраховується поріг *B* так, щоб поділити пікселі на два кластери. Якщо математичні сподівання рівня яскравості в двох кластерах мають різницю, що перевищує певний заданий користувачем рівень $|\mu 1-\mu 2| \geq l$, то всі пікселі всередині вікна *S* бінаризуються відповідно до порогу *T*. В іншому випадку яскравість пікселів з вікна *S* замінюється деяким близьким значенням.

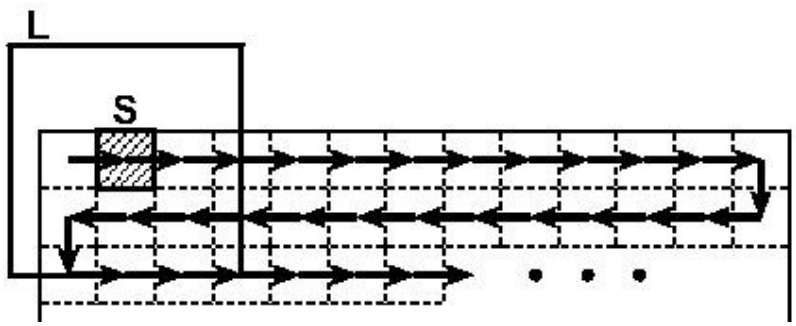


Рисунок 2.5 – Переміщення вікон в методі Ейквеля [21]

На отриманому в результаті зображенні в місцях низької контрастності можуть з'явитися розриви і помилкові чорні області, які можна видалити тільки за допомогою постпроцесингу. При обробці тонких ліній, що перетинаються, можуть виникати розриви, тому метод підходить для товстих ліній і великих об'єктів [21].

2.3.5 Метод Ніблека

Цей метод дозволяє досягти високої швидкості обробки зображень. Він використовується на практиці для швидкої фільтрації контрастних зображень, на яких відсутні сильно зашумлені області з плавними переходами яскравості.

Ідея даного методу полягає в варіюванні порога яскравості B від точки до точки на підставі локального значення стандартного відхилення. Поріг яскравості в точці (x, y) розраховується так:

$$B(x, y) = m(x, y) + k \cdot s(x, y), \tag{2.8}$$

де $m(x, y)$ та $s(x, y)$ – середнє і стандартне відхилення вибірки для деякого оточення точки.

Розмір оточення повинен бути мінімальним, але таким, щоб зберегти локальні деталі зображення. У той же час він повинен бути достатньо великим, щоб знизити вплив шуму на результат. Значення параметру k визначає, яку частину кордону об'єкта взяти у якості самого об'єкта. Значення $k = -0.2$ дає досить чітке розділення об'єктів, якщо вони представлені чорним кольором, а значення $k = +0.2$, – якщо об'єкти представлені білим кольором.

У місцях плавного переходу яскравості метод дає неправдиві об'єкти з невеликим шумом. Метод набув поширення на практиці завдяки його інтеграції з етапом постпроцесингу. При цьому швидкість обробки падає в 3 рази, і кількість помилок скорочується на 20% [21].

2.3.6 Метод Яновіц і Брукштейна

Найбільш ефективним для обробки сканованих паперових картографічних зображень є метод Яновіц і Брукштейна, Таким зображенням властива зональна нерівномірність яскравості, при якій одні і ті ж самі об'єкти зображення в різних частинах мають значні відмінності яскравості (до 20-50%).

В якості порогової поверхні бінаризації використовується поверхня потенціалів, що будується на основі локальної максимізації градієнта яскравості. Значення градієнта яскравості часто розраховується за допомогою контурного оператора Собеля або Кенні. Зображення фільтрується з метою отримання контурних ліній товщиною в 1 піксель, а потім усереднюються фільтром 3 × 3. Потенціальна поверхня будується за допомогою ітераційної інтерполяційної схеми. Розрахунок поверхні йде по порядку, починаючи від контурних пікселів. Для кожного не контурного пікселя розраховується інтерполяційний залишок R (x, y) і нове значення пікселя P (x, y) на n+1-му кроці має розраховуватися відповідно за формулами:

$$P_{n+1}(x, y) = P_n(x, y) + \beta \cdot R_n(x, y) / 4 \qquad (2.9)$$

$$R_n(x, y) = P_n(x - 1, y) + P_n(x + 1, y) + P_n(x, y + 1) - 4P_n(x, y) \qquad (2.10)$$

Автори методу пропонують використовувати значення β в межах $1 \le \beta \le 2$. Метод не дає паразитного шуму на бінарному растрі. Помилка розриву лінійних об'єктів низька, але серед методів локальної бінаризації він найменш продуктивний [21].

2.4 Висновки

У цьому розділі були розглянуті найбільш поширені методи попередньої обробки зображень. Проаналізувавши розглянуті методи, було обрано деякі з них для подальшої реалізації.

3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ ДЛЯ АНАЛІЗУ ДАНИХ З ВИКОРИСТАННЯМ КЛІТИННИХ АВТОМАТІВ

3.1 Вибір зображень для аналізу

Незважаючи на те, що ми живемо у 21 столітті, аналіз зображень цитологічних препаратів здійснюється переважне вручну, що сильно уповільнює процес та призводить до помилок, а погана якість зображення або наявність на зображенні великого числа клітин збільшує ймовірність помилки при з'ясуванні тих чи інших характеристик препарату.

Більшість медичних методик ручного мікроскопічного аналізу експлуатує властивості очей людини розпізнавати співвідношення форми, відтінків, текстури складних об'єктів без їх вимірювання, на основі якісних характеристик. Якщо досліджуваних клітин багато, а необхідний обсяг вибірки невеликий (100-200), то візуальний аналіз (наприклад, підрахунок лейкоформули) при наявності достатнього числа досвідчених лікарів-лаборантів може виконуватися в масовому порядку [22].

Значно гірше справляються очі людини з диференціальним підрахунком більш серйозних вибірок клітин (близько 500), що супроводжується великою кількістю вимірювань (наприклад, визначення індексу овалоцитоз еритроцитів).

Вельми важкі психологічно і фізично для людини завдання пошуку рідкісних клітин, підрахунку складних диференціальних формул, акуратного перегляду великих просторів препарату (наприклад, підрахунок формул при цитопенії, пошук юних клітин в мазку крові, підрахунок мієлограми в мазку кісткового мозку, підрахунок патологічних типів еритроцитів) [22].

У зв'язку з цим домінувала до останнього часу ручна мікроскопія навіть при високій кваліфікації лікаря-лаборанта важка, суб'єктивна і обмежена за своїми можливостями.

Розвиток алгоритмів машинного зору дозволяє поступово перейти від ручного способу аналізу клітинних структур.

Використання автоматичних методів аналізу зображень цитологічних препаратів може значно поліпшити якість аналізу і прискорити його, проте універсальний алгоритм розробити важко в силу різнорідності зображень. Клітинні структури можуть виглядати зовсім по-різному, мати прозору, напівпрозору або непрозору структуру, володіти яскраво вираженими межами.

Тому доцільно розробляти алгоритми під конкретні класи зображень, що підвищить їх ефективність, але, очевидно, застосування їх буде можливо тільки в межах відповідних класів [23].

При розробці необхідно враховувати специфіку зображень даного класу, тому необхідно вирішити ряд проблем, а саме:

1. Клітини можуть злипатися або перекриватися, внаслідок чого кілька клітин можуть сприйматися як одна, тому необхідно вирішити проблему поділу злиплих клітин.
2. Необхідно вирішити проблему відділення клітин від фону, так як в деяких випадках клітини мають нечітку кордон.

3.2 Огляд існуючих рішень

Розробка алгоритмів для підрахунку числа клітин ведеться, і для деяких класів зображень вони вже є, але такі алгоритми зазвичай засновані на сегментації, тобто метою є зіставлення клітини з конкретним сегментом зображення. Такий підхід ефективний в тому випадку, коли клітини непрозорі, тобто на зображенні колір такої клітини значно відрізняється від кольору фону, що дозволяє з'ясувати, чи є сегмент кліткою.

До того ж при наявності шуму на зображенні, а при зйомці цитологічних препаратів це не рідкість, можлива поява безлічі дрібних сегментів, подібних за кольоровими характеристиками з клітинами, внаслідок чого вийде

неякісний підрахунок. Існують класи зображень, для яких використання алгоритмів, заснованих на сегментації [24].

Було розглянуто декілька робіт з обробки зображень клітин і знайдені наступні рішення:

Алгоритм з використанням принципів машинного зору [22]

Ідея запропонованого підходу ґрунтується на побудові системи класифікації зображення на принципі побудови системи машинного зору. В алгоритмі можна виділити наступні етапи:

1. Виконується попередня обробка вихідної вибірки, з метою підвищення якості розпізнавання.
2. Використовується принцип алгоритму автоматичної класифікації кольорових зображень, заснований на побудові R-околиці.

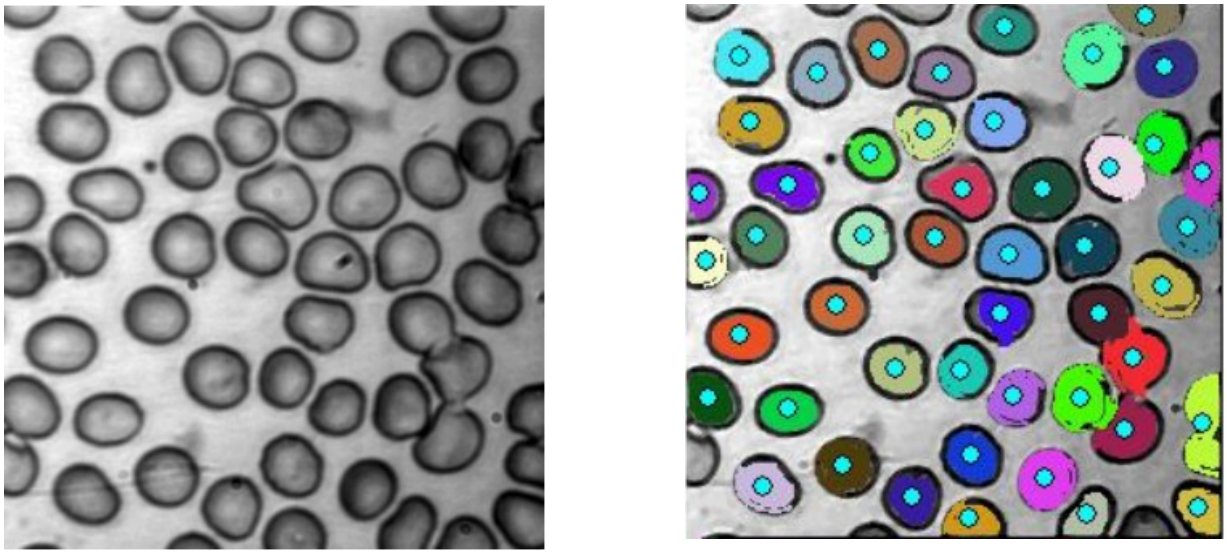


Рисунок 3.1 – Приклад результату роботи алгоритму виділення клітин [22]

Цей метод є досить складним з математичної точки зору, але дозволяє значно знизити залежність точності підрахунку клітин від кольору, прозорості та розташування об’єктів в просторі.

Алгоритм з використанням апроксимації контурів клітин еліпсами [23]

В алгоритмі можна виділити два етапи:

1. Виділення та попередня обробка контурів клітин. Виділення контурів

здійснюється за допомогою комбінованого використання морфологічних перетворень ерозії і ділатиції, а також порогового перетворення карти евклідових відстаней. Контури апроксимуються багатокутниками: з кожного контуру виділяються точки, які є вершинами апроксимуючого багатокутника. На контурах відшукуються точки угнутості. Цими точками контур сегментується на частини.

2. Виділення клітин за допомогою еліпсів. Будуються еліпси, що апроксимують контури клітин. Виявляються еліпси, які не можуть апроксимувати клітку (занадто вузькі або занадто довгі). Об'єднуються сегменти. Обробляються еліпси, що не відібрані на етапі селекції. Вони об'єднуються з другими найбільш підходящими еліпсами. Відбувається корекція виділення і підрахунок еліпсів.

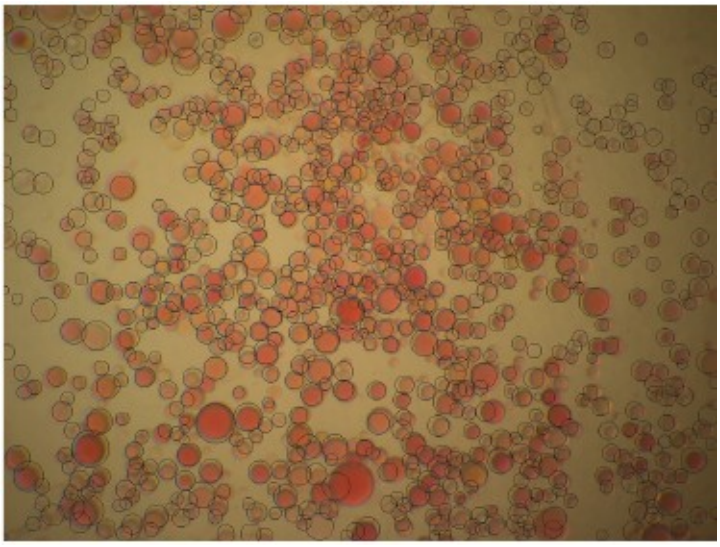


Рисунок 3.2 – Приклад результату роботи алгоритму виділення клітин [23]

Цей метод є досить складним математично та потребує багатої кількості підрахунківі.

Алгоритм з використанням каскадних класифікаторів Хаара [24]

Ознака Хаара складається з суміжних прямокутних областей. Вони позиціонуються на зображенні, далі сумуються інтенсивності пікселів в областях, після чого обчислюється різниця між сумами. Ця різниця і буде

значенням певної ознаки, визначеного розміру, певним чином позиційований на зображенні.

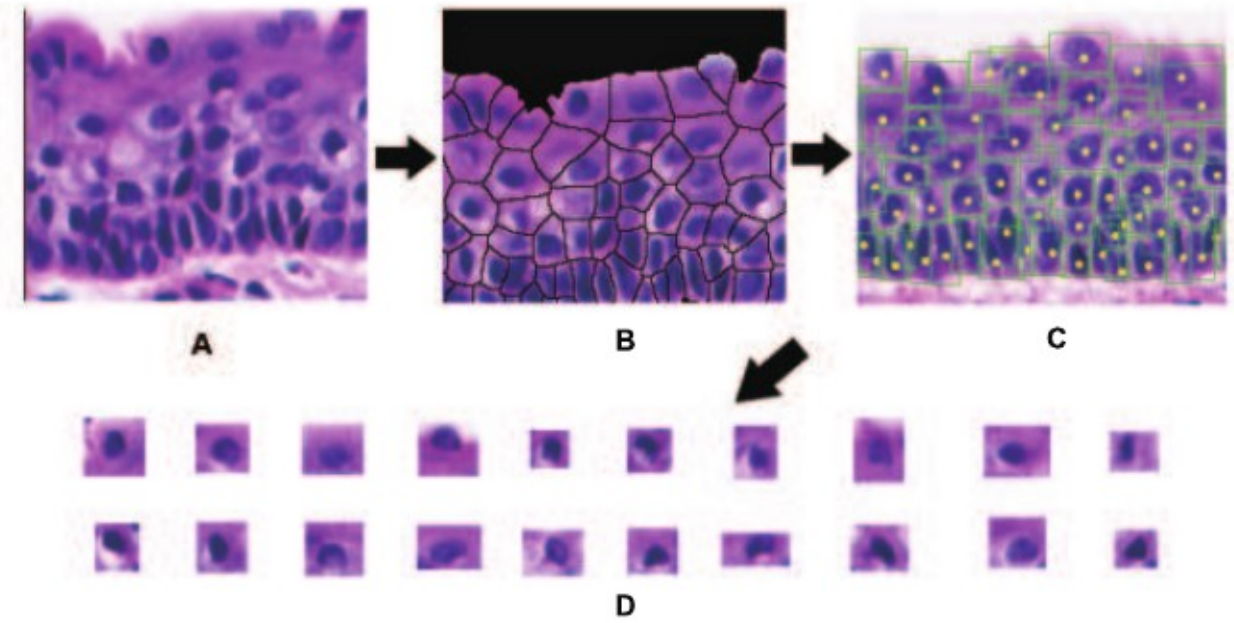


Рисунок 3.3 – Генерація суб-зображень для навчання. A. Оригінальне зображення. B. Сегментація з використанням алгоритму вододілу. C. Площі клітин і їх центр мас (жовті точки). D. Витягнуті суб-зображеня [24]

Цей алгоритм дозволяє не лише підраховувати, а й знаходити клітини певного типу з ймовірністю у 86%, але потребує досить значного за розміром тренувального набору зображень.

3.3 Реалізація алгоритму обробки зображень

3.3.1 Реалізація медіанного фільтра

Алгоритм медіанного фільтра виглядає наступним чином:

1. Завантаження зображення.
2. Формування на основі зображення трьох двомірних масивів. Кожен з трьох масивів відповідає одному з компонентів кольорової моделі *RGB*

та має розмір відповідно до розміру зображення.

3. Значення кожного елементу масиву замінюється на медіану, що обирається з навколишнього оточення (білою крапкою позначено елемент, для якого знаходиться медіана, чорними – ті елементи, яки ми будемо вважати його оточенням). Розмір вікна фільтрації (тобто кількість елементів, що входять в оточення) можна змінювати (3x3, 5x5,...).

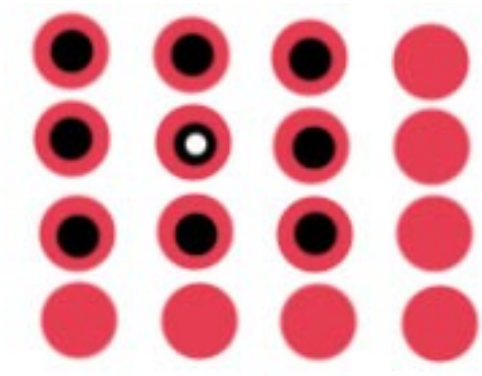


Рисунок 3.4 — Розмір вікна медіанного фільтра

4. З трьох масивів, що відповідають трьом компонентам кольору, формуємо вихідне зображення. Воно і буде результатом фільтрації за допомогою медіанного фільтру.

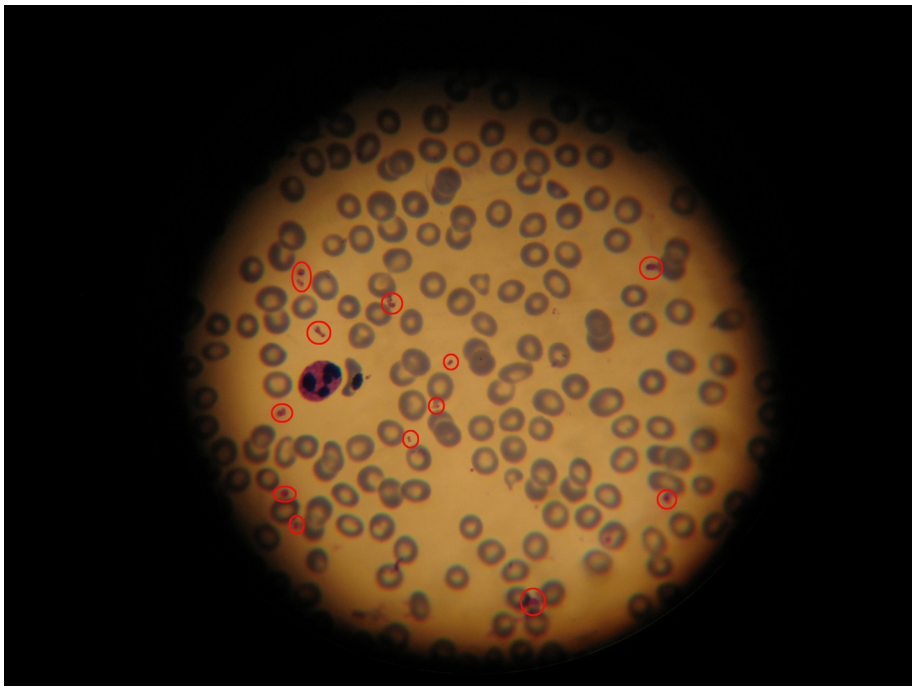


Рисунок 3.5 — Зображення до обробки МФ

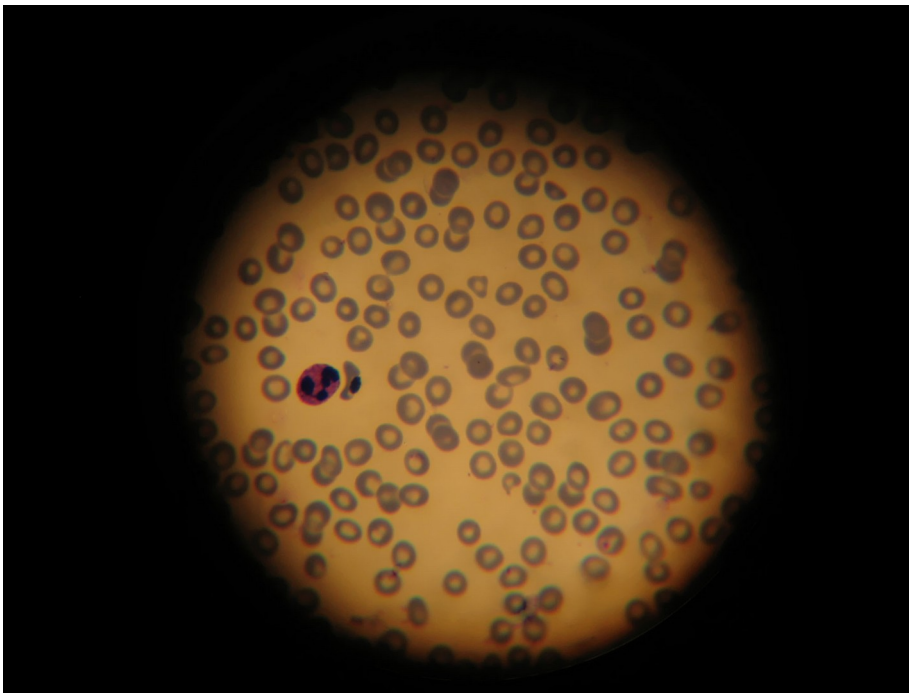


Рисунок 3.6 — Зображення після обробки МФ

3.3.2 Реалізація обраного алгоритму бінаризації

Спочатку була створена програма, у якій поріг порогової фільтрації доводилося обирати вручну, але це не задовольняло початковим умовам, адже було вирішено створити програму, що може працювати повністю автоматично.

Розглянувши відомі методи бінаризації зображень, було обрано метод Оцу для автоматизації обрання порога через те, що він, по-перше, дає мінімальну кількість помилок та працює досить швидко.

Алгоритм бінаризацію за методом Оцу:

1. Завантаження зображення.
2. Створення гістограми зображення.
3. Змінюючи значення t від 1 до 254 для кожного t…
   - …підраховуємо ω_1, ω_2, μ_1, μ_2;
   - …підраховуємо $\sigma_b^2(t)$.
   - Якщо $\sigma_b^2(t) > T$ (спочатку T=0) – запам’ятовуємо цей поріг t та змінюємо значення T (T=σ_b^2).
4. Порогова фільтрація за знайденим порогом t.

3.3.3 Реалізація алгоритму підрахунку клітин з використанням КА

1. Завантаження зображення аналізу крові після порогової фільтрації (докладніше про процес порогової фільтрації було написано у пункті 3.2).

2. Клітинний автомат.

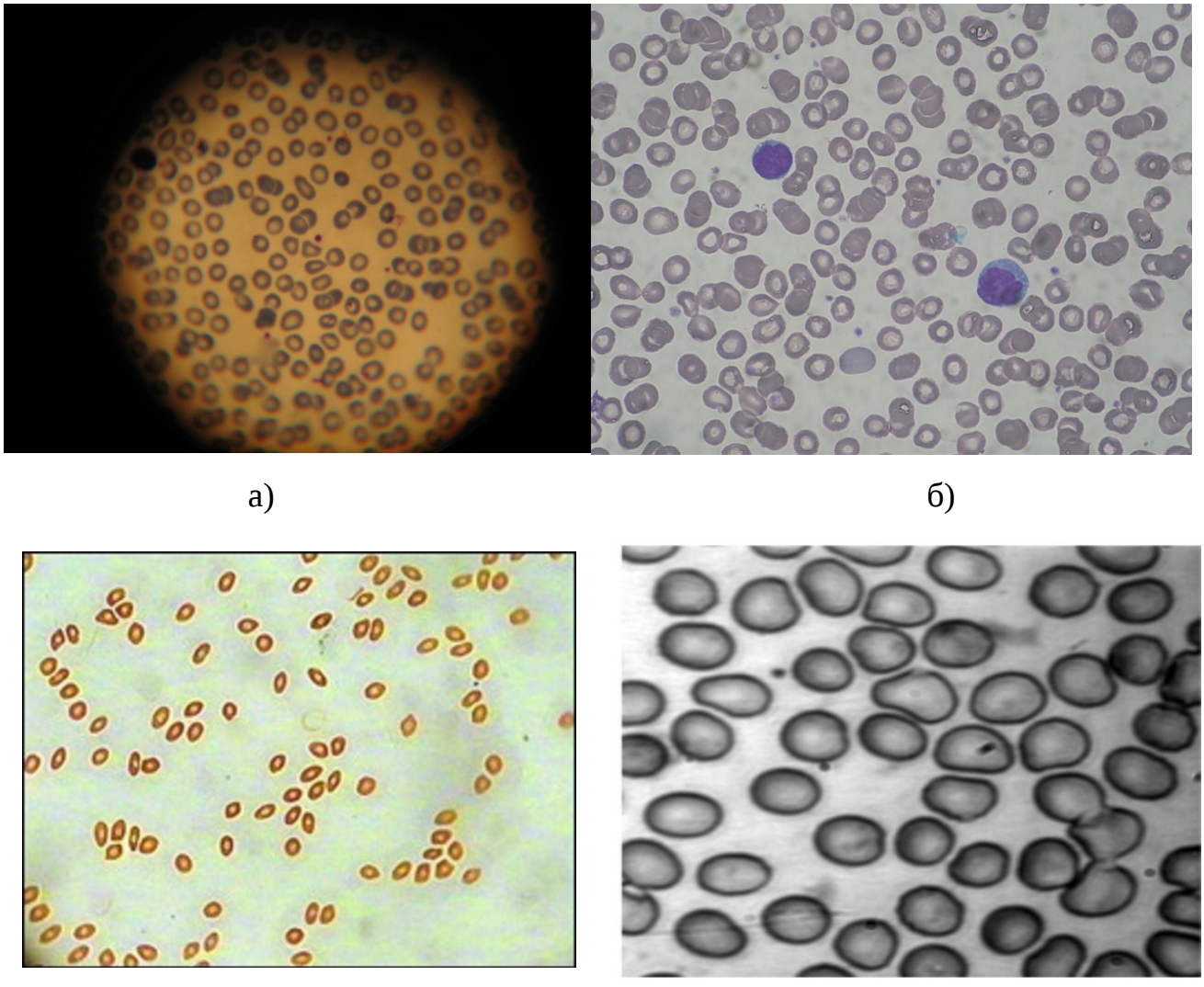
2.1. Нумерація усіх чорних пікселів послідовними натуральними числами, починаючи з одиниці. Білі пікселі позначаємо нулем.

2.2. Заміна у циклі значення номеру кожного пікселя на найбільше з сусідніх. Після того, як зображення стало інваріантним відносно цього перетворення, отримаємо замкнуті області, усередині кожної з яких усі пікселі мають однаковий номер.

3. Виділяємо області, знаходимо їх границі та площу. Рахуємо кількість.

3.4 Порівняння результатів роботи створеної програми

У якості вхідних даних виступають зображення чотирьох видів, приведені на рисунку 3.8:



а) б)

в) г)

Рисунок 3.7 – Приклад вхідних зображень: а) фотографія кількісного аналізу крові, зроблена власноруч; б) зображення мазка периферичної крові за методом Май-Грюнвальда) [23]; в) зображення аналізу крові лягушки [22] та г) зображення мазка переферичної крові за методом Романовського-Гімза [24]

Для кожного із зображень були розглянуті різні значення порогу (які автоматично підібрані, так і підібрані вручну) та підраховано кількість клітин вручну (таблиця 3.1).

Таблиця 3.1 Результати роботи програми для різного типу зображень

Тип зображення	**Значення порогу**	**Кількість клітин, порахована автоматично, Ка**	**Кількість клітин, порахована вручну, Кв**	**Відхилення від результату, отриманого вручну**
	Автоматично знайдене / підібране вручну			
а)	111 / 100	112 / 134	146	0.08 / 0.023
б)	58 / 45	208 /198	202	0.02 / 0.02
в)	31 / 27	94 / 94	98	0.04 / 0.04
г)	23 / 53	35 / 50	48	0.27 / 0.04

Як можна побачити з таблиці 3.1, результат роботи створеного ПП напряму залежить від зображення, що аналізується. Також, потрібно зазначити, що для кожного різновиду зображень необхідно емпірично підбирати значення площі клітини, щоб поділяти клітини, що склеїлися. На рис. 3.7 б) присутні великі включення, що порахуються як декілька склеєних клітин.

Створений ПП можна застосувати не тільки для обробки кількісних аналізів крові. Наприклад, при дослідженнях проб води та землі дуже часто необхідно порахувати кількість бактерій. На рис. 3.8. наведено приклад підрахунку кількості бактерій *Saccharomyces cerevisiae*.

Рисунок 3.8 – Приклад роботи програми з зображенням бактерії *Saccharomyces cerevisiae*

У таблиці 3.2 наведено результати порівняння алгоритмів обробки зображень клітин. Потрібно зазначити головний недолік, що належить усім розглянутим алгоритмам – вони розробляються для певного виду зображень клітин та можуть погано працювати на інших. Лише один з розглянутих методів дозволяє класифікувати клітини за їх видом, але й він є

К перевагам реалізованого алгоритму можна віднести його відносну простоту. Також, він не потребує великої кількості зображень для навчання та складних математичних обчислень, на відміну від розглянутих алгоритмів.

На жаль, є й певні недоліки: автоматично підібраний поріг за методом Оцу не завжди є оптимальним, що призводить до помилкового підрахунку кількості клітин.

Таблиця 3.2 Порівняння алгоритмів обробки зображень крові

	Алгоритм з використанням апроксимації контурів клітин еліпсами	Алгоритм з використанням каскадних класифікаторів Хаара	Алгоритм з використанням принципів машинного зору	Алгоритм з використанням клітинних автоматів
Відхилення від результату, порахованого вручну, %	12	15	7	10
Необхідна мінімальна кількість зображень	1	30	1	1
Переваги	Після виділення контурів, ісходне зображення не використовується	Дозволяє класифікувати клітини за типом	Дозволяє зменшити вплив на результат значення кольору та прозорості об’єктів	Простий алгоритм
Недоліки	Багато обчислень, складний алгоритм	Потребує наявність тренувального набору зображень	Багато обчислень, складний алгоритм	Метод Оцу

3.5 Висновки

В цьому розділі було наведено алгоритм роботи створеного ПП, де були програмно реалізовані обрані методи попередньої обробки зображень та алгоритм запропонованого клітинного автомату. Створений програмний продукт було апробовано на чотирьох різновидах зображень аналізів крові та порівняні отримані результати.

4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У даному розділі проводиться оцінка основних характеристик програмного продукту, призначеного для оцінки та прогнозування багатофакторних ризиків. Інтерфейс користувача був розроблений за допомогою мови програмування delphi у середовищі розробки Borland Delphi 7. Програмний продукт призначено для використання на персональних комп’ютерах під управлінням операційної системи Windows.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом:

- визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам: ті, що впливають на вартість продукту і ті, що не впливають. На цьому ж етапі оптимізується сама послідовність скороченням кроків, що

не впливають на цінність і відповідно витрат.

- для кожної функції визначаються повні річні витрати й кількість робочих часів.
- для кожної функції на основі оцінок попереднього пункту визначається кількісна характеристика джерел витрат.
- після того, як для кожної функції будуть визначені їх джерела витрат, проводиться кінцевий розрахунок витрат на виробництво продукту.

4.1 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод ФВА для проведення техніко-економічний аналізу розробки.

Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

- програмний продукт повинен функціонувати на персональних комп’ютерах із стандартним набором компонент;
- забезпечувати високу швидкість обробки великих об’ємів даних у реальному часі;
- забезпечувати зручність і простоту взаємодії з користувачем або з розробником програмного забезпечення у випадку використовування його як модуля;
- передбачати мінімальні витрати на впровадження програмного продукту.

4.1.1 Обґрунтування функцій програмного продукту

Головна функція F_0 – розробка програмного продукту, який, вкиористовуючи клітинні автомати, аналізує та оброблює вхідні дані. Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F_1 – вибір мови програмування;

F_2 – інтерфейс користувача.

F_3 – розпізнавання вхідних даних;

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) мова програмування delphi;

б) мова програмування C++;

Функція F_2:

а) інтерфейс користувача, створений за допомогою Delphi Forms;

б) інтерфейс користувача, створений за допомогою ПО Qt.

Функція F_3:

а) введення даних вручну;

б) написання окремого модулю для введення даних.

4.1.2 Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 4.1).

Рисунок 4.1 – Морфологічна карта

Морфологічна карта відображає всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

Таблиця 4.1 – Позитивно-негативна матриця

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	А	Кращі стандартні бібліотеки, більш «читабельний» код	Не кросплатформений
	Б	Можливість кросплатформеності	Складніша структура
F2	А	Легкий у створенні, швидший відгук на дії користувача	Відсутність кросплатформеності
	Б	Широкий набір інструментів для створення інтерфейсу користувача	Складніша реалізація
F3	А	Простий у використанні та реалізації	Потребує оператора
	Б	Повна автоматизованість створенного ПО	Час створювання більший, складніша реалізація

На основі аналізу позитивно-негативної матриці робимо висновок, що при обрані програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція *F1:*

Оскільки може потребуватися обробка різних зображень, то структура ПО повинна бути як умога простішою, тому варіант б) має бути відкинутий.

Функція *F2:*

Інтерфейс користувача має бути інтуітивно зрозумілим та швидкостворюванним, тож відкинемо варіант б).

Функція *F3:*

Введення даних користувачем не відіграє великої ролі в даному програмному продукті, тож вважаємо варіанти а) та б) гідними розгляду.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

1. F1а – F2а – F3а
2. F1а – F2а – F3б

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.2 Обґрунтування системи параметрів ПП

4.2.1 Опис параметрів

На підставі даних про основні функції, що повинен мати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

- *X1* – швидкодія мови програмування;
- X2 – об’єм пам’яті для збереження даних;
- X3 – швидкість взаємодії з користувачем;
- X4 – потенційний об’єм програмного коду.

4.2.2 Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.2.

Таблиця 4.2 – Основні параметри ПП

Назва Параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	Оп/мс	19000	11000	2000
Об’єм пам’яті для збереження даних	X2	Мб	32	16	8
Час обробки запитів користувача	X3	мс	1000	420	60
Потенційний об’єм програмного коду	X4	кількість рядків коду	2000	1500	1000

За даними таблиці 4.2 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

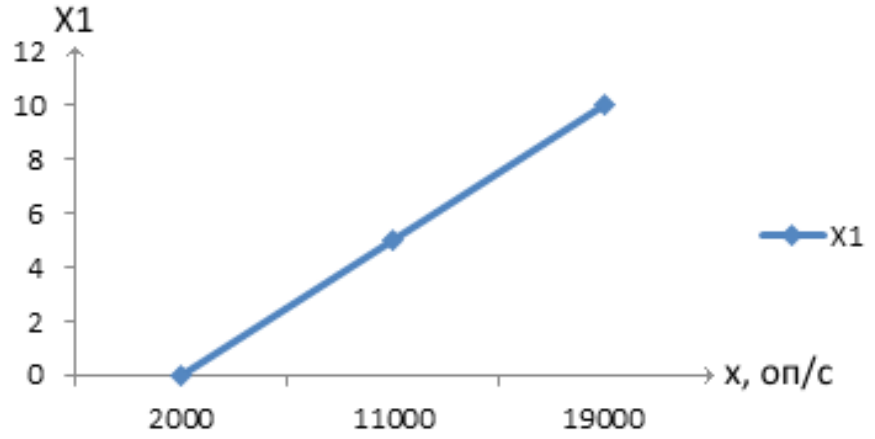


Рисунок 4.2 – X1, швидкодія мови програмування

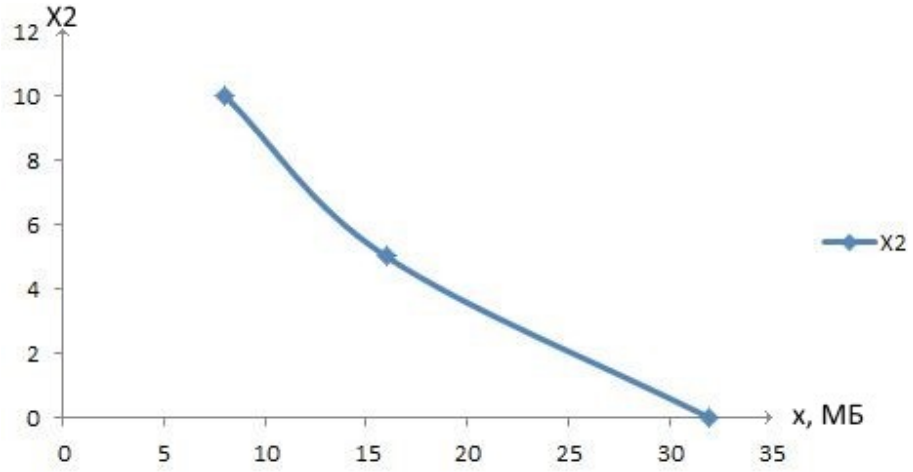


Рисунок 4.3 – X2, об’єм пам’яті для збереження даних

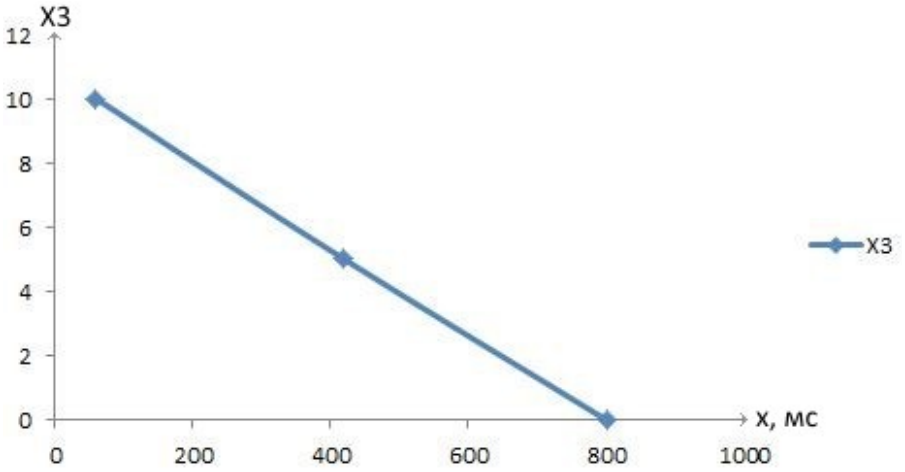


Рисунок 4.4 – X3, час виконання запитів користувача

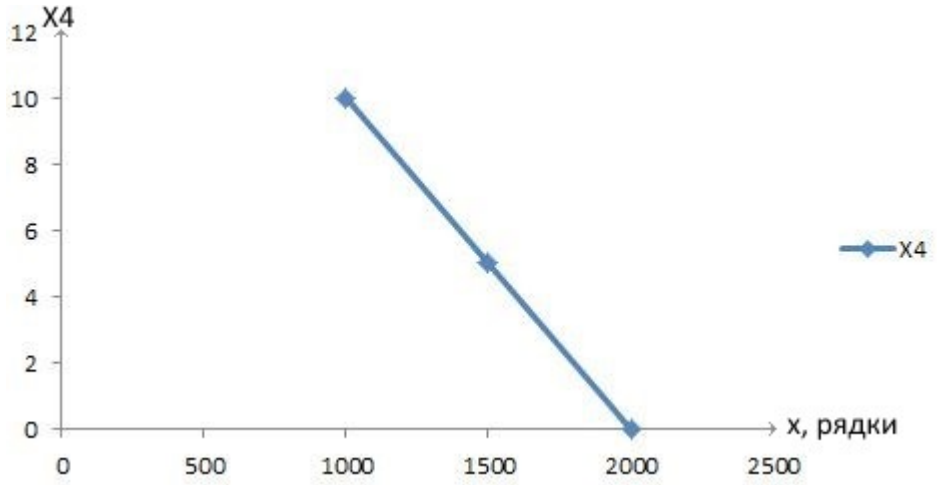


Рисунок 4.5 – X4, потенційний об’єм програмного коду

4.2.3 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – вибір програмного продукту, який має найбільш зручний та функціональний набір можливостей при низький необхідності у ресурсах.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 4.3.

Таблиця 4.3 – Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	4	3	3	3	2	3	4	22	4,5	20,25
X2	Об’єм пам’яті для збереження даних	Мб	1	2	1	1	3	1	1	10	-7,5	56,25
X3	Час обробки запитів користувача	Мс	2	1	2	2	1	2	2	12	-5,5	30,25
X4	Потенційний об’єм програмного коду	кількість рядків коду	3	4	4	4	4	4	3	26	8,5	72,25
	Разом		10	10	10	10	10	10	10	70	0	179

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70,$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17,5$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S=\sum_{i=1}^{N}\Delta_i^2=179.$$

Порахуємо коефіцієнт узгодженості:

$$W=\frac{12\,S}{N^2\left(n^3-n\right)}=\frac{12\cdot 179}{7^2\left(4^3-4\right)}=0,73>W_k=0,67$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	>	>	>	>	<	>	>	>	1,5
X1 і X3	>	>	>	>	>	>	>	>	1,5
X1 і X4	>	<	<	<	<	<	>	<	0,5
X2 і X3	<	>	<	<	>	<	<	<	0,5
X2 і X4	<	<	<	<	<	<	<	<	0,5
X3 і X4	<	<	<	<	<	<	<	<	0,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij}=\begin{cases}1,5\,при\,X\,i>X\,j\\1.0\,при\,X\,i=X\,j\\0.5\,при\,X\,i<X\,j\end{cases}$$

З отриманих числових оцінок переваги складемо матрицю $A=\left\| a_{ij} \right\|=$

$$\begin{pmatrix}1 & 1.5 & 1.5 & 0.5\\ 1.5 & 1 & 0.5 & 0.5\\ 1.5 & 0.5 & 1 & 0.5\\ 0.5 & 0.5 & 0.5 & 1\end{pmatrix}$$

Для кожного параметра зробимо розрахунок вагомості $K_{ві}$ за наступними формулами:

$$K_{в\,i}=\frac{b_i}{\sum_{i=1}^{n} b_i} \quad \text{,де} \quad b_i=\sum_{i=1}^{N} a_{ij} \quad .$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{в\,i}=\frac{b_i^{'}}{\sum_{i=1}^{n} b_i^{'}}, \quad \text{де} \quad b_i^{'}=\sum_{i=1}^{N} a_{ij} b_j \quad .$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{в\,i}$	b_i^1	$K_{в\,i}^1$	b_i^2	$K_{в\,i}^2$
X1	1,0	1,5	1,5	0,5	4,5	0,219	22,25	0,216	100	0,215
X2	1,5	1,0	0,5	0,5	3,5	0,281	27,25	0,282	124,25	0,283
X3	1,5	0,5	1,0	0,5	3,5	0,344	34,25	0,347	156	0,348
X4	0,5	0,5	0,5	1,0	2,5	0,156	14,25	0,155	64,75	0,154
Усього:					16	1	98	1	445	1

4.3 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо. Абсолютні значення параметрів X2 (об’єм пам’яті для збереження даних) та X1 (швидкодія мови програмування) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра X3 (час обробки даних) обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 800мс або варіанту б) 80мс.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j)=\sum_{i=1}^{n} K_{вi,j} B_{i,j},$$

де n – кількість параметрів; $K_{вi}$ – коефіцієнт вагомості i–го параметра;B_i – оцінка i–го параметра в балах.

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1(X1)	А	11000	3,6	0,215	0,774
F2(X2)	А	1200	8	0,154	1,232
F3(X3,X4)	А	800	2,4	0,348	0,835
	Б	80	1	0,154	0,154

За даними з таблиці 4.6 за формулою

$$K_K=K_{TУ}\left[F_{1k}\right]+K_{TУ}\left[F_{2k}\right]+...+K_{TУ}\left[F_{zk}\right],$$

визначаємо рівень якості кожного з варіантів:

K_{K1} = 0,774 + 1,232 + 0,835 = 2,841

K_{K2} = 0,774 + 1,232+ 0,154 = 2,16

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.4 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М}, \quad (5.1)$$

де T_P – трудомісткість розробки ПП;$K_П$ – поправочний коефіцієнт;$K_{СК}$ – коефіцієнт на складність вхідної інформації; K_M – коефіцієнт рівня мови програмування;$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм; $K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: T_P =90 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.7. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{СТ}$ = 0.8. Тоді, за формулою 5.1, загальна трудомісткість програмування першого завдання дорівнює:

$T_1 = 90 \cdot 1.7 \cdot 0.8 = 122.4$ людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто T_P =27 людино-днів, $K_П$ =0.9,$K_{СК}$ = 1,$K_{СТ}$ =0.8:

$T_2 = 27 \cdot 0.9 \cdot 0.8 = 19.44$ людино-днів.

Для третього завдання (використовується алгоритм другої групи складності, ступінь новизни Г з використанням перемінної інформації):

Тр = 12 людино-днів;

Кп = 0.72; Кст = 0.8;

То = 12 • 0.72 • 0,8 = 6.91.

Для четвертого завдання (використовується алгоритм третьої групи складності, ступінь новизни Г):

Тр = 8 людино-днів;

Кп = 0.6; Кст = 1;

То = 8 • 0.6 • 1 = 4.8.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$T_I = (122.4 + 19.44 + 4.8) \cdot 8 = 1173.12$ людино-годин;

$T_{II} = (122.4 + 19.44 + 6.91) \cdot 8 = 1190$ людино-годин;

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 6000 грн., один фінансовий аналітик з окладом 9000грн. Визначимо зарплату за годину за формулою:

$$С Ч = \frac{М}{T_m \cdot t} грн.,$$

де М – місячний оклад працівників; T_m – кількість робочих днів тиждень; t – кількість робочих годин в день.

$$С Ч = \frac{6000+6000+9000}{3 \cdot 21 \cdot 8} = 41,67 грн.$$

Тоді, розрахуємо заробітну плату за формулою

$$СЗП = C_ч \cdot T_i \cdot К Д ,$$

де $С_ч$– величина погодинної оплати праці програміста; T_i – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП} = 41,67 \cdot 1173.12 \cdot 1.2 = 58660.69$ грн.

II. $С_{ЗП} = 41,67 \cdot 1190 \cdot 1.2 = 59504.76$ грн.

Відрахування на єдиний соціальний внесок становить 22%:

I. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 58660.69*0.22 = 13091.05$ грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 59504.76*0.22 = 12905.35$ грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 6000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 6000 \cdot 0,2 = 14400$ грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_Г \cdot (1 + К_З) = 14400 \cdot (1 + 0.2) = 17280$ грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД} = С_{ЗП} \cdot 0.3677 = 17280 * 0,22 = 3801.6$ грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 8000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 8000 = 2300$ грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 8000 \cdot 0.05 = 460$ грн.,

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4$ годин,

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t – кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1706,4 \cdot 0,156 \cdot 0.2 \cdot 2.0218 = 107,64$ грн.,

де $N_С$ – середньо-споживча потужність приладу; $К_З$– коефіцієнтом зайнятості

приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$С_Н = Ц_{ПР} \cdot 0.67 = 8000 \cdot 0,67 = 5360$ грн.

Тоді, річні експлуатаційні витрати будуть:

$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$

$С_{ЕКС} = 17280 + 3801.6 + 2300 + 460 + 107.64 + 5360 = 29309,24$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М-Г} = С_{ЕКС} / Т_{ЕФ} = 29309,24 / 1706,4 = 17,18$ грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$С_М = С_{М-Г} \cdot Т$

I. $С_М = 17.18 \cdot 1173.12 = 20154.2$ грн.;

II. $С_М = 17.18 \cdot 1190 = 20444.2$ грн.;

Накладні витрати складають 67% від заробітної плати:

$С_Н = С_{ЗП} \cdot 0,67$

I. $С_Н = 58660.69 \cdot 0.67 = 39302.66$ грн.;

II. $С_Н = 59504.76 \cdot 0.67 = 39868.19$ грн.;

Отже, вартість розробки ПП за варіантами становить:

$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$

I. $С_{ПП} = 58660.69 + 12905.35 + 20154.2 + 39302.66 = 131022.9$ грн.;

II. $С_{ПП} = 59504.76 + 13091.05 + 20444.2 + 39868.19 = 132908.2$ грн.;

4.5 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$K_{ТЕРj} = K_{Kj} / C_{Фj}$,

$K_{ТЕР1} = 2,841 / 132908.2 = 0,21 \cdot 10^{-4}$;

$K_{ТЕР2} = 2,16 / 132908.2 = 0,16 \cdot 10^{-4}$;

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $K_{ТЕР1} = 0,21 \cdot 10^{-4}$.

4.6 Висновки

В даному розділі проведено повний функціонально-вартісний аналіз ПП, який було розроблено в рамках дипломного проекту. Процес аналізу можна умовно розділити на дві частини. В першій з них проведено дослідження ПП з технічної точки зору: було визначено основні функції ПП та сформовано множину варіантів їх реалізації; на основі обчислених значень параметрів, а також експертних оцінок їх важливості було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій ПП.

Другу частину ФВА присвячено вибору із альтернативних варіантів реалізації найбільш економічно обґрунтованого. Порівняння запропонованих варіантів реалізації в рамках даної частини виконувалось за коефіцієнтом ефективності, для обчислення якого були обчислені такі допоміжні параметри, як трудомісткість, витрати на заробітну плату, накладні витрати.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту.

У нього виявився найкращий показник техніко-економічного рівня якості $K_{ТЕР} = 0,21 \cdot 10^{-4}$.

Цей варіант реалізації програмного продукту має такі параметри:

- мова програмування – Delphi;
- інтерфейс користувача, створений за допомогою Delphi Forms;
- введення даних вручну.

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, непоганий функціонал і швидкодію.

ВИСНОВКИ

Дана дипломна робота присвячена дослідженню клітинних автоматів при їх застосування для обробки даних та розробці програмного засобу для автоматичного підрахунку об’єктів на зображені.

У першому розділі було надане математичне визначення клітинних автоматів та розглянуті їх способи класифікації. Також, були приведені приклади застосування клітинних автоматів на практиці.

У другому розділі були розглянуті найбільш популярні методи попередньої обробки зображень. Було досліджено один з найбільш поширених методів обробки сигналів, що застосовуються в інформаційно-телекомунікаційних системах з метою усунення небажаних шумів – медіанну фільтрацію. Також, були розглянуті різні методи бінарізації зображень, один з яких (метод Оцу) було обрано для реалізації.

У третьому розділі було пояснено причини обрання у якості тестового зображення фотографії кількісного аналізу крові. Були проаналізовані існуючи реалізації ПП для обробки зображень клітин.

Також, було сформульовано алгоритм обробки зображень, який включає наступні етапи:

- медіанну фільтрацію,
- бінарізацію за порогом, обраним за методом Оцу або власноруч
- підрахунок об’єктів за допомогою клітинного автомату.

Створений програмний продукт було апробовано на чотирьох різних варіантах зображень клітин крові. Отримані результати були порівнянні з підрахованими вручну. Похибка підрахунку клітин на зображенні складає у середньому 10%, а в випадку обробки певного типу зображень — 4.7%.

Також, була розглянута можливість застосування створеного ПП не лише для обробки кількісних аналізів крові, а й для підрахунку бактерій.

Запропонований алгоритм обробки зображень було порівняно з існуючими алгоритмами. До переваг алгоритму з використанням клітинних автоматів можна віднести його відносно просту структуру та принцип роботи.

Отже, в роботі показано, що цифрова обробка сигналів, яка зараз здійснюється за допомогою комп’ютера – це одна з перспективних галузей високих наукомістких технологій, приваблива для докладання зусиль. Вже зараз прогрес суспільства, особливо в сфері охорони здоров'я, багато в чому залежить від досягнень комп'ютерної обробки зображень.

Для подальшого розвитку необхідно удосконалити алгоритм роботи програми, реалізувавши наступні ідеї:

- збільшити кількість можливих варіантів цитологічних зображень;
- реалізація алгоритму, що буде не лише підраховувати кількість клітин, а й класифікувати та визначати відсоток клітин, що захворіли.

Тоді запропонований алгоритм можливо буде використовувати в якості елементу додатку, що дозволить лаборанту отримати результати аналізу одразу ж після фотографування цитологічного препарату та зменшити вірогідність медичних помилок.

ПЕРЕЛІК ПОСИЛАНЬ

1. Хрящев Д. О. Попередня обробка та аналіз зображень, отриманих в умовах недостатнього освітлення. автореф. дис. на здобуття наук. ступеня канд. техн. наук: спец. 05.13.01 “Системний аналіз і управління” / Д .О. Хрящев. – Астрахань, 2013. – 145 с.
2. Фон Нейман Дж. Теория *самовоспроизводящихся автоматов* / Дж. Фон Нейман, А. Бёркс – М.: Мир, 1971. – 382 с.
3. A.G.Hoekstra. Simulating complex systems by cellular automata. / Hoekstra A.G., Kroc J., Sloot P. – London: Springer, 2010. – 235 p.
4. S. Wolfram. A New Kind of Science. / Wolfram S. – Wolfram Media. Inc. – 2002. – 1125 p.
5. Лебедев А. В. Вероятностные методы классификации клеточных автоматов / А. В. Лебедев // Фундаментальная и прикладная математика. Открытые системы. – 2002. – № 2 – С. 621–626.
6. Gutowitz H.A. A hierarchical classification of cellular automata / H.A. Gutowitz // Physica D. – №4 – 1990. – pp. 136–156.
7. Wuensche A. Classifying Cellular Automata Automatically / Wuensche A. // COMPLEXITY. – 1999. – № 4 – pp. 47-66.
8. Наумов Л.А. Классификация структур, порождаемых одномерными двоичными клеточными автоматами из точечного зародыша. / Л.А. Наумов, А.А. Шалыто // Известия РАН. Теория и системы управления. – 2005. – №5. – С.137–145. – Режим доступу: http://is.ifmo.ru/works/classif/. – Дата доступу: 02.05.2016
9. 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-dlya-

obrabotki-izobrazheniy-minnyh-poley. – Дата доступу: 05.05.2016

10. Малинецкий Г. Г. Применение клеточных автоматов для моделирования движения группы людей / Г. Г. Малинецкий, М. Е. Степанцов // Журнал вычислительной математики и математической физики. – 2004. – № 11. – С. 2108 – 2112.
11. Применение клеточных автоматов для моделирования динамических процессов. – Режим доступу: http://uran.donetsk.ua/~masters/2012/fknt/plotnikov/library/article2.htm. – Дата доступу: 02.05.2016
12. Зотов Я. А. Использование клеточных автоматов в симметрической криптосистеме / Я. А. Зотов // Вопросы кибербезопасности. – 2015. – № 3. – С. 43 – 45.
13. Чичварин Н. В. Предварительная обработка изображений. / Н. В. Чичварин – Режим доступу: http://ru.bmstu.wiki/Предварительная_обработка_изображений. – Дата доступу: 07.05.2016
14. Стругайло В. В. Обзор методов фильтрации и сегментации цифровых изображений / В. В. Стругайло // Наука и Образование. – 2012 . – №77 – С. 270 – 281.
15. Путятин Е.П. Нормализация и распознование изображений. / Е.П. Путятин – Режим доступу: http://sumschool.sumdu.edu.ua/is-02/rus/lectures/pytyatin/pytyatin.htm. – Дата доступу: 17.05.2016
16. Путятин Е.П. Обработка изображений в робототехнике / Е.П. Путятин, С.И. Аверин – М.: Машиностроение, – 1990. – 320 с.
17. Кельберт. М. Медианная фильтрацияс / М. Кельберт., Л. Питербарг // Квант. – 1990 р. – №10. – С. 8–13, 47.
18. Питербарг Л.И. Медианная фильтрация случайных процессов / Л.И. Питербарг // Проблемы передачи информации. – 1984 р. – №1. – С. 55-73

19. Сойфер В.А. Компьютерная обработка изображений. Часть 2. Методы и алгоритмы / В.А. Сойфер // Соросовский образовательный журнал. – 1996. – №3 – С. 110-121.
20. Черненко С.А. Медианный фильтр. / С.А. Черненко. – Режим доступу: http://www.logis-pro.kiev.ua/math_power_medianfilter_ru.html. – Дата доступу: 7.05.2016
21. Федоров А. Бинаризация черно-белых изображений: состояние и перспективы развития. / А. Федоров. – Режим доступу: http://iu5.bmstu.ru/~philippovicha/ITS/IST4b/ITS4/Fyodorov.htm. – Дата доступу: 17.05.2016
22. Ковригин А.В. Приминение принципов построения систем машинного зрения в задаче анализа изображений клеточных структур / А.В. Ковригин // Научный журнал КубГАУ. – 2007. – №29.
23. М.С. Тарков Оценивание числа клеток на изображениях цитологочиских растительнвх препаратах. / Тарков М.С., Осипов М.И. // Известия Томского политехнического университета. – 2007. – №5.
24. Radicular cysts and odontogenic keratocysts epithelia classification using cascaded – Режим доступу: http://breckon.eu/toby/publications/papers/han08cell. – Дата доступу: 17.05.2016

ДОДАТОК А

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, Grids, ValEdit, ComCtrls;

type
  TForm1 = class(TForm)
    countButton: TButton;
    Image1: TImage;
    Image2: TImage;
    TrackBar: TTrackBar;
    OpenDialog: TOpenDialog;
    OpenFileButton: TButton;
    ThreshholdLevel: TLabel;
    doThreshholdButton: TButton;
    countLabel: TLabel;
    ProgressBar: TProgressBar;
    procedure countButtonClick(Sender: TObject);
    procedure OpenFileButtonClick(Sender: TObject);
    procedure TrackBarChange(Sender: TObject);
    procedure doThreshholdButtonClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
type cell=record
  x0,y0,x1,y1: Integer;
  black,white:Integer;
end;

var
  Form1: TForm1;
  plot: array [0..639, 0..479] of Integer;
  temp: array [0..639, 0..479] of Integer;
  cells: array [0..500] of cell;
  cellscount: Integer;
  cellscount_corrected: Integer;
  manual_threshhold: Boolean;
  const picWidth=640;
  const picHeight=480;

implementation
```

```
function count(): Integer;
var n:Integer;
    k,l: Integer;
    flag: Boolean;
    itemcount: Integer;
    found:array [0..640*480] of Boolean;
    i,j:Integer;
    a: array [0..639*479] of Word;
begin
  itemcount:=0;
  for i:=0 to 640*480 do
    found[i]:=False;
  n:=0;
  flag:=True;
  for i:=0 to picWidth-1 do
    for j:=0 to picHeight-1 do
    begin
      if (plot[i][j]=1) then
      begin
        n:=n+1;
        plot[i][j]:=n;
      end;
    end;
  while (flag) do
  begin
    flag:=False;
    for i:=1 to picWidth-2 do
      for j:=1 to picHeight-2 do
        if (plot[i][j]<>0) then
        begin
          n:=plot[i][j];
          for k:=i-1 to i+1 do
            for l:=j-1 to j+1 do
              if (plot[k][l]>n) then
              begin
                n:=plot[k][l] ;
                flag:=True;
              end;
          temp[i][j]:=n;
        end
        else
          temp[i][j]:=0;

    for i:=1 to picWidth-2 do
      for j:=1 to picHeight-2 do
        plot[i][j]:=temp[i][j];
  end;
  for i:=1 to 638 do
    for j:=1 to 478 do
    begin
```

```
      if (plot[i][j]<>0) then
        if not found[plot[i][j]] then
        begin
          found[plot[i][j]]:=True;
          itemcount:=itemcount+1;
          a[plot[i][j]]:=itemcount;
          cells[a[plot[i][j]]].x0:=i;
          cells[a[plot[i][j]]].y0:=j;
          cells[a[plot[i][j]]].x1:=i;
          cells[a[plot[i][j]]].y1:=j;
        end
        else
        begin
          if (i<cells[a[plot[i][j]]].x0) then cells[a[plot[i][j]]].x0:=i;
          if (i>cells[a[plot[i][j]]].x1) then cells[a[plot[i][j]]].x1:=i;
          if (j<cells[a[plot[i][j]]].y0) then cells[a[plot[i][j]]].y0:=j;
          if (j>cells[a[plot[i][j]]].y1) then cells[a[plot[i][j]]].y1:=j;
        end;
    end;
  count:=itemcount;
end;

procedure
doThreshhold(input:TImage;output:TImage;trackbar:TTrackBar;progressbar:TProgressBar);
var i,j:Integer;
    histogram: array [0..255] of Integer;
    w1,w2,a1,a2,T,threshhold,threshhold_temp: integer;
    label no_autothreshhold;
begin
  threshhold:=trackbar.Position;
  for i:=0 to 255 do
    histogram[i]:=0;
  output.Picture:=nil;
  T:=0;

  if (manual_threshhold) then goto no_autothreshhold;
  for i:=0 to picWidth-1 do
    for j:=0 to picHeight-1 do
      Inc(histogram[GetRValue(input.Canvas.Pixels[i,j])]);
  //for i:=0 to 255 do
  //Begin
  //  output.Canvas.MoveTo(i,output.Height);
  //  output.Canvas.LineTo(i,output.Height-Trunc(histogram[i]/40));
  //End;
  //Application.ProcessMessages;
  //Sleep(2000);


  for threshhold_temp:=2 to 254 do
  Begin
    w1:=0; w2:=0; a1:=0; a2:=0;
```

```
    for i:=0 to threshhold_temp do
    Begin
      Inc(w1,histogram[i]);
      Inc(a1,histogram[i]*i);
    End;
    if (w1<>0) then
      a1:=round(a1/w1);

    for i:=threshhold_temp to 255 do
    Begin
      Inc(w2,histogram[i]);
      Inc(a2,histogram[i]*i);
    End;
    if (w2<>0) then
      a2:=round(a2/w2);

    if (w1*w2*sqr(a1-a2) > T) then
    Begin
      threshhold:=threshhold_temp;
      T:=w1*w2*sqr(a1-a2);
    end;
  end;
  trackbar.Position:=threshhold;
  no_autothreshhold:

  ProgressBar.Visible:=True;
  for i:=0 to picWidth-1 do
  begin
    ProgressBar.Position:=trunc(1000*i/638);
    Application.ProcessMessages;
    for j:=0 to picHeight-1 do
      if ((GetGValue(input.Canvas.Pixels[i,j])+GetRValue(input.Canvas.Pixels[i,j])/2)>255-
threshhold) then
      begin
        output.Canvas.Pixels[i,j]:=clWhite;
      end
      else
        output.Canvas.Pixels[i,j]:=clBlack;
  end;
  ProgressBar.Visible:=false;
end;

{$R *.dfm}

procedure TForm1.countButtonClick(Sender: TObject);
var i,j: integer;
begin
  for i:=1 to picWidth-2 do
  begin
    for j:=1 to picHeight-2 do
      if (Image2.Canvas.Pixels[i,j]=clblack) then
```

```
      plot[i][j]:=1
    else
      plot[i][j]:=0;
  end;

  cellscount:=count();
  cellscount_corrected:=0;
  Image2.Canvas.Pen.Color:=clred;
  for i:=1 to cellscount do
  begin
   if  ((cells[i].x1-cells[i].x0)*(cells[i].y1-cells[i].y0)>200) then
   begin
     Inc(cellscount_corrected);
     with Image2.Canvas do
     begin
       MoveTo(cells[i].x0,cells[i].y0); LineTo(cells[i].x0+4,cells[i].y0);
       MoveTo(cells[i].x0,cells[i].y0); LineTo(cells[i].x0,cells[i].y0+4);
       MoveTo(cells[i].x1,cells[i].y1); LineTo(cells[i].x1-4,cells[i].y1);
       MoveTo(cells[i].x1,cells[i].y1); LineTo(cells[i].x1,cells[i].y1-4);
     end;
     if ((cells[i].x1-cells[i].x0)*(cells[i].y1-cells[i].y0)>600) then
       Inc(cellscount_corrected);
   end;
  end;
  countLabel.Caption:=IntToStr(cellscount_corrected);
end;



procedure TForm1.OpenFileButtonClick(Sender: TObject);
begin
if OpenDialog.Execute then
  Image1.Picture.LoadFromFile(OpenDialog.FileName);
manual_threshhold:=false;
end;

procedure TForm1.TrackBarChange(Sender: TObject);
begin
ThreshholdLevel.Caption:=IntToStr(TrackBar.Position)+'/255';
end;

procedure TForm1.doThreshholdButtonClick(Sender: TObject);
begin
//  ShowMessage(IntToStr(GetRValue(Image1.Canvas.Pixels[0,0])));
doThreshhold(Image1,Image2,TrackBar,ProgressBar);
manual_threshhold:=True;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
Image1.Canvas.Pixels[0,0]:=clWhite;
```

```
Image2.Canvas.Pixels[0,0]:=clWhite;
end;

end.
```