Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(рівень вищої освіти)

Спеціалізований обчислювач для реалізації фрактально-векторного способу аналізу зображень.
(тема)

Виконав: студент 2 курсу, групи СКСм-18-2
Іорін І.Р.
(прізвище, ініціали)
Спеціальність 123 – Комп'ютерна інженерія
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма
Спеціалізовані комп'ютерні системи
(повна назва освітньої програми)
Керівник к.т.н. доц. Рахліс Д.Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ _____
(підпис) (прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Автоматизації проектування обчислювальної техніки
Рівень вищої освіти другий (магістерський)
Спеціальність 123 – Комп'ютерна інженерія
Тип програми Освітньо-професійна
Освітня програма Спеціалізовані комп'ютерні системи

(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Іоріну Іллі Романовичу
(прізвище, ім'я, по батькові)

1. Тема роботи _____
Спеціалізований обчислювач для реалізації фрактально-векторного
способу аналізу зображень
затверджена наказом по університету від 04.11.2019 р. № 1624Ст
2. Термін подання студентом роботи до екзаменаційної комісії ____ _____ 20__ р.
3. Вихідні дані до роботи _____
Зображення енергограм
4. Перелік питань, що потрібно опрацювати в роботі _____
Застосування Спеціалізованих обчислювачів аналізу зображень в медицині, Енергетична діагностика, фрактальний аналіз, методи класифікації зображень, методи та моделі аналізу зображень операційні методи прискорення, апаратні методи прискорення, програмування на графічних процессорах, робота штучних нейронних мереж, розробка інтерфейсу користувача

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)

_____
_____
_____
_____
_____
_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання	03.09.2019 - 04.09.2019	
2	Аналіз предметної області	05.09.2019 - 15.09.2019	
3	Аналіз джерел з проблемної галузі	16.09.2019 - 01.10.2019	
4	Здійснити посекторний фрактальний аналіз	02.10.2019 - 19.10.2019	
5	Результати аналізу подати у вигладі векторів	20.10.2019 - 28.10.2019	
6	Класифікувати в. для встановл. відхилень	29.10.2019 - 08.11.2019	
7	Здійснити апаратну реалізацію	09.11.2019 - 18.11.2019	
8	Оформлення пояснювальної записки	19.11.2019 - 29.11.2019	
9	Оформлення графічного матеріалу	30.11.2019 - 04.12.2019	
10	Перевірка виконаного проекту керівником	05.12.2019 - 10.12.2019	

Дата видачі завдання _____

Студент ___
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 86 сторінок, 43 рисунків, 3 таблиць, 14 джерел за переліком посилань..

ОБРОБКА ЗОБРАЖЕНЬ, АНАЛІЗ ЗОБРАЖЕНЬ, ЕНЕРГОГРАМА, ГРВ-ГРАМА, ФРАКТАЛ, ФРАКТАЛЬНИЙ АНАЛІЗ, НЕЙРОННІ МЕРЕЖІ.

Метою атестаційної роботи є дослідження спеціалізованого апаратно-програмного комплексу, який орієнтований на виконання задачі визначення поля фрактальної розмірності для зображень з елементами самоподібності з метою їх подальшої класифікації на основі одержуваних матриць фрактальних розмірностей фрагментів зображень.

У ході виконання атестаційної роботи було спроектовано та реалізовано алгоритм секторального аналізу зображень з елементами самоподібності методом розрахунку фрактальної розмірності для секторів оцифрованої енергограми. Обробка та класифікація результатів фрактального аналізу зображень виконується з використанням нейронних мереж. Для оптимізації та прискорення процесу обробки аналізу нами використані апаратні засоби відеокарт.

ABSTRACT

Research practice thesis 86 pages, 43 figures, 3 tables, 14 sources.

IMAGE ANALYSIS, IMAGE PROCESSING, ENERGYGRAPH, GRAVGRAPH, FRACTAL, FRACTAL ANALYSIS, COMPARATIVE ANALYSIS, NEURAL NETWORKS.

The purpose of the research practice is to study a specialized hardware and software complex, which is focused on the task of determining the field of fractal dimension for images with elements of self-similarity with the purpose of their further classification on the basis of obtained matrixes of fractal dimensions of image fragments.

During the performance of the appraisal work, the algorithm for sectoral analysis of images with elements of self-similarity was designed and implemented by the method of fractal dimension calculation for sectors of the digitized energyogram. The processing and classification of the results of fractal image analysis is performed using neural networks. We have used video card hardware to optimize and speed up the analysis process.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ГРВ – метод газоразрядної візуалізації

ГРВ-грамма – зображення отрімане методом газоразрядної візуалізації

ЕТД – енергетична діагностика по термінальним точкам

Енергограма – зображення отримане методом енергитичної діагностики

ПФР – поле фрактальної розмірності

ШНМ – штучні нейронні мережі

DoG (англ. Difference of Gaussian) – різниця гаусианів

CPU (англ. central processing unit) – центральний процесор

GPU (англ. graphics processing unit) – графічний процесор

ВСТУП

Одна з найпоширеніших форм представлення інформації – це графічна. Вона може виступати як об’єктом, так і результатом дослідження. Внаслідок ускладнення науково-технічних задач, а також збільшення кількості наукових досліджень, аналіз та обробка графічної інформації потребують винайдення нових методів, зокрема таких, що будуть спиратися на новітні математичні відкриття.

Обробка і розпізнавання цифрових зображень – це окрема дисципліна, що набуває стрімкого розвитку. Передбачається, що для проведення обробки зображень використовується комп’ютер або інший пристрій, побудований на основі цифрових сигнальних процесорів. Слід зауважити, що процес обробки включає не тільки покращення візуального сприйняття зображень, а також і класифікацію об’єктів, яка може виконуватись при аналізі зображень.

Іконіка – специфічна наука про зображення, що розвивалася у 60-ті роки минулого століття. Вона присвячена вивченню характеристик та властивостей зображень, а також цілям і задачам їх перетворення, обробці, відтворенню в розпізнанню графічних образів. Сам термін походіть від грецького слова «eikon» – образ, зображення. У сьогоденні його використовують для відзначення процесу створення і обробки зображень із застосуванням ЕОМ.

Комп’ютерна обробка зображень може слугувати для вирішення цілого різноманіття завдань, зокрема для підвищення якості, розпізнавання і зменшення розміру зображень, а також визначення їхніх параметрів і проведення спектрального аналізу багатомірних сигналів.

У сьогоденні попит на застосування цифрової обробки значно зростає, як и сфери в яких вона набуває свого призначення. Їх широко використовують в мистецтві, промисловості, медицині и навіть у космічних дослідженнях, витісняючи аналогову обробку зображень. Цифрові методи

дозволяють проводити аналіз та класифікацію зображень, виявляти складні структури та неочевидні особливості і закономірності, як правило такі, що можуть бути виявлені лише експертами в певній області.

Стосовно біології та медицини, існує можливість значного втручання у протікання дослідження для прискорення і полегшення процесу діагностики пацієнта. Досягти цього можна за допомогою автоматизованої обробки великих наборів зображень, що дозволить визначити тип аналізованої тканини, виявити пухлини або присутність певних сторонніх з'єднань.

У сьогоденні метод газорозрядної візуалізації (ГРВ) широко застосовують для вирішення проблем діагностики технічних, а також біологічних об'єктів.

Він спирається на реєстрацію та обробку світінь, які виникають при розміщенні об'єктів різноманітної природи в імпульсному електромагнітному полі високої напруженості. Світіння необхідно перетворити цифровий код. Для цього в комп’ютері, за допомогою матриць, формують ГРВ-граму – двовимірне півтонове зображення.

Одними з найбільш сучасних, є дослідження кільцеподібних ГРВ-грам кінчиків пальців людини, адже саме на них зосереджені початкові і кінцеві точки енергомеридіани, які відображають енергостан організму. Дану методику називають енергетичною діагностикою по термінальним точкам. Енергетична картина даних знімків дає можливість заглянути в сферу інформації, енергоінформаційного обміну в організмі, розширити розуміння здоров'я і хвороби [2].

Окрім відомого методу газорозрядної візуалізації, кафедра ЕОМ запатентувала новий метод, який спирається на ефект поляризації діелектрики, що допомагає отримати, подібні до ГРВ-грам, кільцеві енергограми.

Необхідно відмітити, що знаходження загальної фрактальної розмірності для подібних типових зображень буде неінформативним. Уточнення та деталізація дослідження потребує аналізу окремих фрагментів

зображення.

В останньому десятиріччі відмічається тенденція щодо зростання цікавості до застосування нейронних мереж при вирішенні задач аналізу і прогнозування даних. Нейронні мережі слугують доповненням до традиційних методів аналізу даних і можуть бути застосованими при побудові складних нелінійних залежностей.

Для вирішення задач класифікації скористаємося мережами з прямим зв'язком, оскільки вони можуть виступати зручним засобом апроксимації функцій. Найчастіше використання нейронних мереж – це найефективніший засіб класифікації, адже вони створюють багато регресійних моделей, що використовують при рішенні завдань класифікації статистичними методами. Ці методи у сьогоденні – це найкраще рішення задачі класифікації результатів фрактального аналізу, адже вони зорієнтовані на пошук прихованих залежностей і можуть самонавчатися.

Необхідно зазначити, що вищеперелiковані завдання обчислювально-трудомісткі. Одним із дієвих методів прискорення є паралелізація шляхом перенесення реалізації обчислювального алгоритму на апаратні засоби відеокарт, частково або повністю. Відеокарта може слугувати універсальним обчислювачем з паралельною архітектурою, тобто призначена виконувати операції паралельно для цілого масиву даних, а не тільки реалізовувати певну програму.

Актуальність атестаційної роботи полягає в розробці аппаратно-програмного комплексу, що здійснює параметризацію зображень методом фрактально-векторного аналізу для подальшої класифікації енергограм з пошуком прихованих закономірностей. Атестаційна робота є рішенням затребуваної задачі аналізу та класифікації енергограм, сучасними методами з метою встановлення міри відхилення від норми показників, що відображають стан здоров’я людини. Таким чином можна неінвазивно та нетравматично встановити захворювання на ранніх стадіях та запобігти розвитку патогенних процесів.

1 ЗАСТОСУВАННЯ СПЕЦІАЛІЗОВАНИХ ОБЧИСЛЮВАЧІВ АНАЛІЗУ ЗОБРАЖЕНЬ В МЕДИЦИНІ

1.1 Енергетична діагностика

У сьогоденні методи енергетичної діагностики набувають популярності у сфері завдання діагностики біологічних і технічних об'єктів. Наприклад, енергетична діагностика у медичній сфері дозволить детектувати мінімальні зміни стану енергетики людини в концепції узагальненого підходу до здоров'я людини та оцінки можливостей її адаптації. Серед переваг методу відносно загальноприйнятих методів можна відзначити:

– неінвазивність;

– нетравматичність;

– відсутність протипоказань за віком та станом здоров'я обстежуваного;

– можливість застосовувати декілька разів в процесі терапії, що необхідно для контролю її ефективності.

Сам по собі ефект світіння об'єктів в електромагнітних полях високої напруженості знайом людству вже понад два століття. Проте необхідність у складній, специфічній апаратурі, до певного часу, стояла на перешкоді дослідженням учених. Завдяки подружжю Кірліан, російським винахідникам, що виявили цей ефект ще в 1930-40 рр, метод «високочастотного фотографування» став широко розповсюдженим і, в подальшому, саме на ефект Кірліан базувався розвиток енергетичної діагностики.

Сьогодні під терміном ефект Кірліан розуміється візуальне спостереження або реєстрація на фотоматеріалі світіння газового розряду, що виникає поблизу поверхні об'єкта при розміщенні його в електричному полі високої напруженості (біоелектрографія або газорозрядна візуалізація) [1].

Доктором Петером Манделєм було розроблено систему діагностичних карти, що викривають відповідності ділянок пальців рук і ніг певним органам і

системам організму, що частково наслідує системі китайських енергомерідіанів.

Для формулювання положення про існування коливального ритму між клітинними і енергетичними функціями Петер Мандель обстежив сотні тисяч пацієнтів. Дане положення стверджує, що біоенергія людини – це носій інформації, і наслідком нормального перебігу енергії буде правильне функціонування клітин. При умові, що з деяких причин інформація змінюється, втрачаючи рівноважний ритм, має змінитися і функціонування клітин. Крім того, оскільки, полярність має зберігатися, можна стверджувати, що клітинні зміни, також будуть впливати на інформаційний зміст біоенергетичних процесів.

Доктор Мандель також вказує, що зображені на фотознімках променеві вінці ілюструють замкнені енергетичні циркуляції, що мають свою власну природу, відмінну від описаної в літературі симптоматики меридіанів.

Кірліан-фотографія кінчиків пальців людини, у яких знаходяться початкові і кінцеві точки енергомерідіанів, ілюструють інтегративний енергостан організму. Згідно з Манделєм, дана методика була названа ЕТД, тобто енергетична діагностика по термінальним точкам.

Енергетична картина ЕТД-знімків охоплює весь рельєф фізичного, психічного та духовного життя людини і, таким чином, дає можливість заглянути в сферу інформації, енергоінформаційного обміну в організмі, розширити розуміння здоров'я і хвороби [2]. Фотопсихічне випромінювання енергетичної кінетики може ілюструвати нормальний і патологічний стан протікання життєвих процесів, згідно їх симптоматиці, у внутрішньосистемних зв'язках сенсорної мережі.

1.1.1 Метод газорозрядної візуалізації

Метод газорозрядної візуалізації (ГРВ) – це метод дослідження біологічних об'єктів, і, зокрема, людини шляхом аналізу характеристик світіння, що виникає поблизу поверхні об'єкта при розміщенні його в електричному полі високої напруженості. Заснований на ефекті Кірліан [3].

Згідно з результатами досліджень, інтенсивність, характер і структура ГРВ-світіння значно залежить від вихідного стану досліджуваного об'єкта. Метод ГРВ дає можливість оцінити структурно-функціональний стан організму з отриманням стабільних відтворюваних результатів в реальному масштабі часу [2].

Принципова схема пристрою, зображена на рисунку 1.1 ілюструє закономірності роботи ГРВ-приладів.

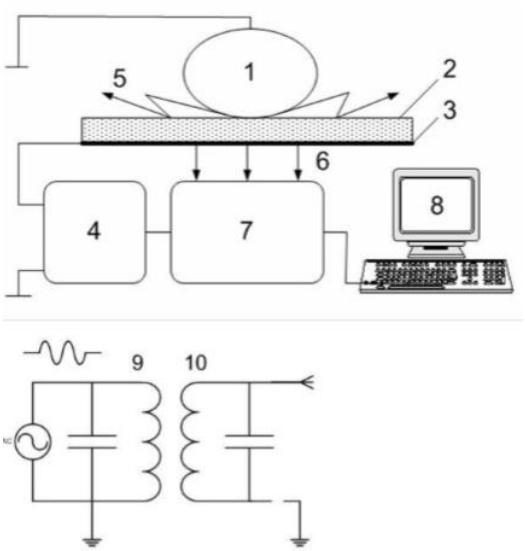

Рисунок 1.1 – Принципова схема методу ГРВ

Розшифрування позначень схеми:

– 1 – досліджуваний об'єкт;

– 2 – діелектрична пластина (кварц);

– 3 – прозоре струмопровідне покриття;

– 4 – генератор імпульсів;

– 5 – ковзний газовий розряд;

– 6 – світіння розряду;

– 7 – оптична система і ПЗС-камера;

– 8 – комп'ютер;

– 9 і 10 – система пов'язаних LC контурів, утворених елементами схеми приладу і еквівалентної ємністю досліджуваного об'єкта [2].

На рисунку 1.2 зображені ГРВ-грами, отримані в ході дослідження.

Вони представляють собою просторовий розподіл освітленості, в залежності від стану досліджуваного об'єкта.

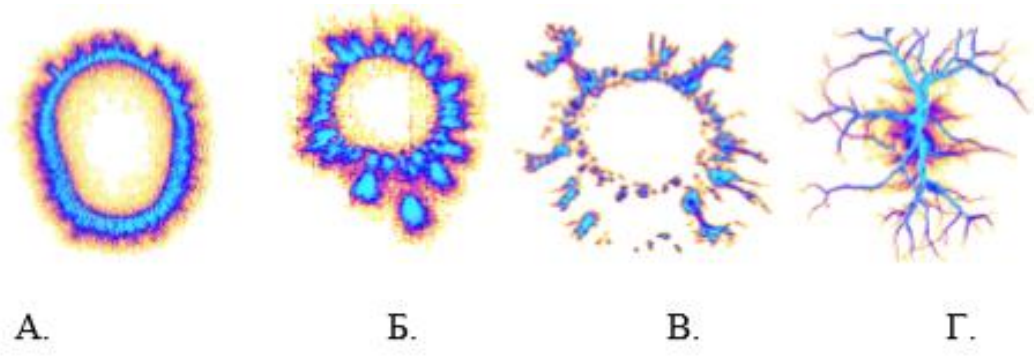

Рисунок 1.2 – Приклади ГРВ-грам

Розшифрування позначень:

– А – ГРВ-грама пальця руки здорової людини;

– Б – палець кардіологічного хворого;

– В – палець людини в стресі;

– Г – ГРВ-грами краплі рідини.

1.1.2 Метод отримання енергограм що базується на поляризації діелектриків

Метод аналізу стану біологічних і технічних об’єктів, що базується на явищі поляризації діелектриків може виступати як альтернатива газорозрядній візуалізації.

На відміну від Кірліан ефекту, який, за рахунок газорязрядной візуалізації здійснюється при високих рівнях напруги, створюється приховане зображення розподілу зон електропровідності на поверхні досліджуваного об'єкта за рахунок поляризації деяких видів діелектриків в імпульсному високочастотному електричному полі [2].

При натисканні на діелектричну пластину, на певний час вмикається високовольтна напруга, що дозволяє створити енергограму, тобто приховане зображення досліджуваного об’єкта, на поверхні діелектрика. Енергограма

виглядає як півтонове зображення високої чіткості та контрастності, що створюється через розпилення принтерного тонеру на поверхні діелектрика (рис. 1.3). Його структура подібна до структури зображення, отриманої під час газорозрядної візуалізації.

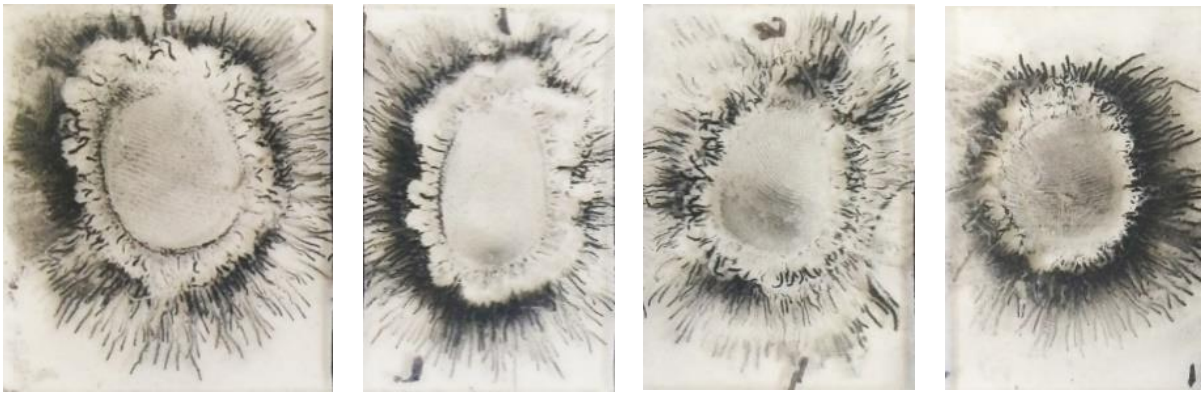

Рисунок 1.3 – Приклади зображень енергограм пальців

Рисунок 1.4 ілюструє принципову схему отримання енергограми на основі поляризації діелектриків.

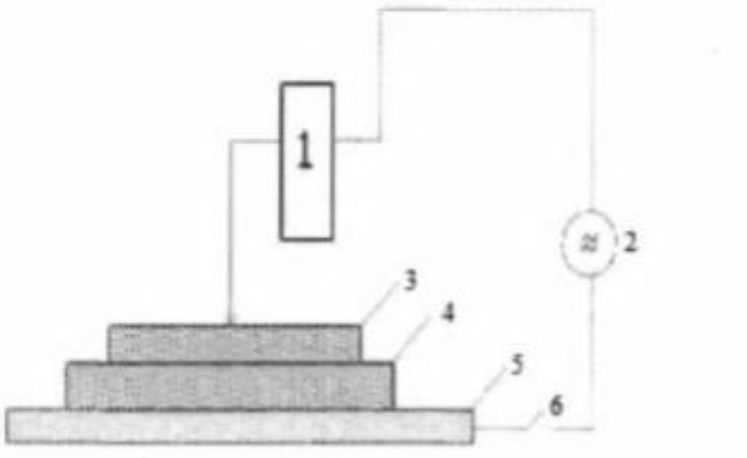

Рисунок 1.4 – Принципова схема методу отримання енергограм

Розшифрування умовних позначень:

– 1 – об'єкт дослідження;

– 2 – імпульсний генератор;

– 3 – ізоляційна пластина;

– 4 – предметна пластина;

– 5 – електрод;

– 6 – вмикач геренатора.

Переваги методу отримання енергограми на основі поляризації діелектриків в тому, що цей процес протікає при суттєво менших напругах, в порівнянні з методом ГРВ, а також пристрій для отримання енергограм, зображений на рисунку 1.5, дозволяє одночасно отримати вимірювання з усіх пальців пацієнта (рисунок 1.6) [3].

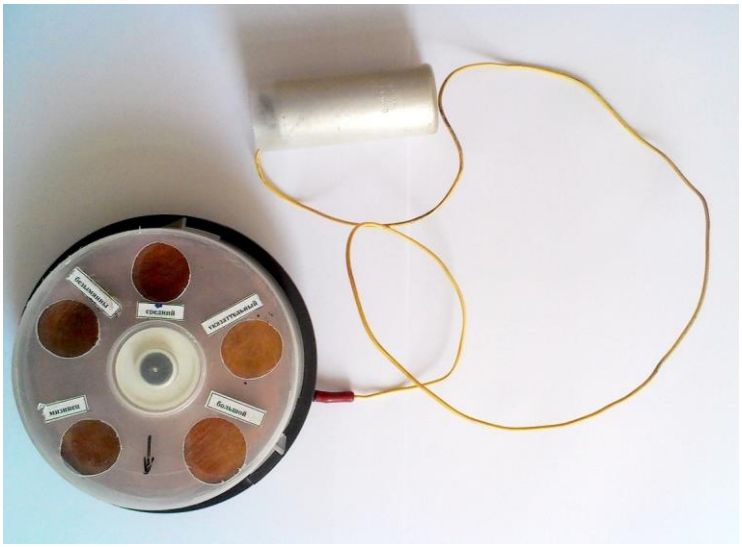

Рисунок 1.5 – Пристрій для зняття енергограм пальців пацієнтів

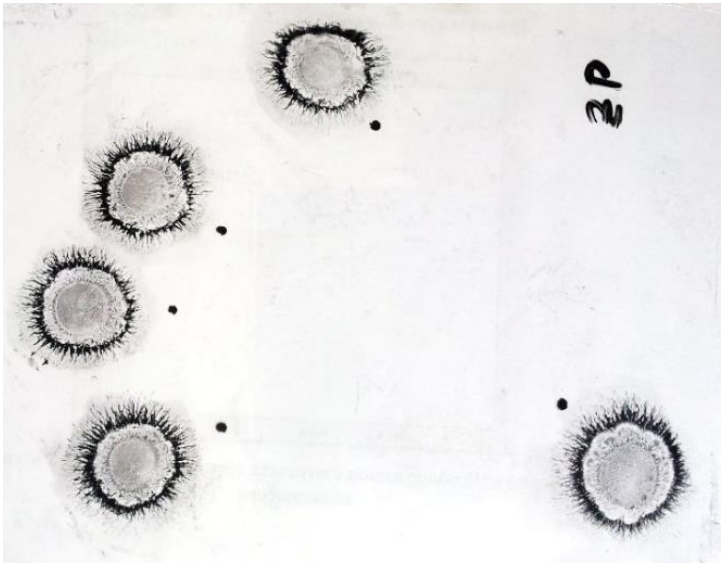

Рисунок 1.6 – Результат зняття енергограми пальців пацієнтів

1.1.3 Типи випромінювань

Якість типу випромінювання відображає стадії патологічного процесу задовго до появи наочної симптоматики захворювання. Це має важливе значення для профілактичної медицини [1].

Нормальний тип випромінювання, зображений на рисунку 1.7, представляє собою корону навколо пальців рук. Він має чіткий суцільний внутрішній контур, стримерний внутрішній шар і зовнішнє люмінесцентне випромінювання.

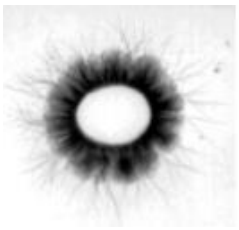

Рисунок 1.7 – Нормальний тип випромінювання

На рисунках 1.8 і 1.9. зображено ендокринний тип випромінювання, тобто стан корони з випаданнями, які з’являються під час емоційної лабільності, астенізації організму.

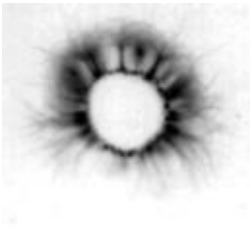

Рисунок 1.8 – Ендокринний тип випромінювання

Рисунок 1.9 – Психоемоційна лабільність

На рисунку 1.10 зображено токсичний тип випромінювання, тобто стан корони з точковими протуберанцями, які з'являються під час інтоксикації і запалення.

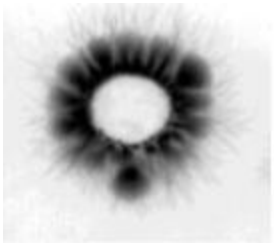

Рисунок 1.10 – Токсичний тип випромінювання

На рисунку 1.11 зображено дегенеративний тип випромінювання, який характерний при розвитку патологічного процесу з появою органічної патології і свідчить про глибокі і важкі порушення енергетичних циркуляцій. У випромінювання збільшується окресленість і товщина внутрішнього кільця, а також товщина стримерного шару.

Рисунок 1.11 – Дегенеративний тип випромінювання

1.1.4 Секторальна діагностика

Зображена на рисунку 1.12 діаграма побудована Петером Манделем. Вона відображає відповідність точок на кінчиках пальців, органам і системам організма, з якими вони пов'язані

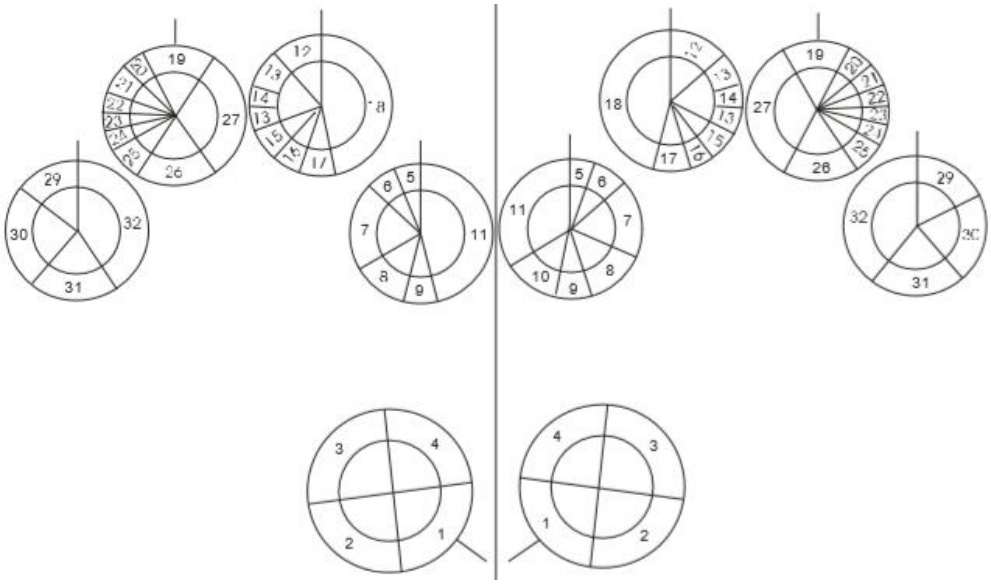

Рисунок 1.12 – Секторальная діаграма пальців рук

Розшифрування умовних позначень наведемо в таблиці 1.1. У праву колонку даної таблиці внесені значення для секторів пальців правої руки, а в ліву, відповідно, – лівої; підзаголовками позначені сектори, які відносяться до одного пальця [3].

Таблиця 1.1 – Органи і системи в секторальної діаграмі пальців рук по системі П. Манделя

Ліва рука	Права рука
1	2
Великий палець (1)	
1. Лобова пазуха, придаткових пазуха, верхня щелепа	1. Лобова пазуха, придаткових пазуха, верхня щелепа
2. Ніс	2. Ніс
3. Нижня щелепа, лімфоглоткового кільце, вухо	3. Нижня щелепа, лімфоглоткового кільце, вухо
4. Мигдалик, шия	4. Мигдалик, шия
Вказівний палець (2)	
Нервова дегенерація-товстий кишечник	Нервова дегенерація-товстий кишечник
5. Головний мозок	5. Головний мозок
6. Шийний відділ спинного мозку	6. Шийний відділ спинного мозку
7. Грудний відділ спинного мозку	7. Грудний відділ спинного мозку

Продовження таблиці 1.1

1	2
8. Поперечно-кресцовий відділ спинного мозку	8. Поперечно-кресцовий відділ спинного мозку
9. Пряма кишка	9. Копчик
	10. Апендикс
11. Товстий кишечник	11. Товстий кишечник
Середній палець (3)	
12. Зона голови і очей	12. Зона голови і очей
13. Грудна клітка	13. Грудна клітка
14. Лімфа	14. Лімфа
15. Зона живота	15. Зона живота
16. Печінка (ноги)	16. Печінка (ноги)
17. Нирки	17. Нирки
18. Циркуляція крові каналу голованогі	18. Циркуляція крові каналу голованогі
Безіменний палець (4)	
19. Голова	19. Голова
20. Прищитовидні залози	20. Прищитовидні залози
21. Щитовидна залоза	21. Щитовидна залоза
22. Тимус	22. Тимус
23. Підшлункова залоза	23. Підшлункова залоза
24. Наднирники	24. Наднирники
25. Яєчник (яєчко)	25. Яєчник (яєчко)
26. Матка (простата)	26. Матка (простата)
27. Нервова система (психіка)	27. Нервова система (психіка)
Мізинець (5)	
29. Сліпокишкова заслінка	29. Гастро-дуоденальна зона
30. Клубова кишка	30. Тонка кишка
31. Застійна зона лімфи, молочні заліза, легені, бронхи	31. Застійна зона лімфи, молочні заліза, легені, бронхи
32. Серце	32. Серце

1.2 Фрактальний аналіз

Як правило, зображення енергограм мають складну структуру, і при постановці завдання їхнього дослідження, доречно скористуватися методами фрактального аналізу.

Фракталами називають певні геометричні об'єкти – лінії, поверхні, просторові тіла, тощо, які мають сильно порізану форму і властивість самоподібності.

Бенуа Мандельброт ввів поняття «фрактал» і «фрактальна розмірність» [4]. Пізніше вони набули широкого застосування в фізиці, біології та інших природничих науках. Використання фракталів дозволило зробити значні зрушення в завданнях моделювання формоутворення природних об'єктів. Фрактали досить точно описують безліч фізичних явищ: гори, хмари, турбулентні течії, структури дерев так кровоносних судин, тощо. Необхідність правильної оцінки фраклальной розмірності в тому, що це число пов'язане з морфологічними характеристиками природніх явищ. Фрактальна розмірність або розмірність Хаусдорфа широко використовується для аналізу подібних систем [5].

1.2.1 Точковий метод. Метод 1

Точковий метод являє собою один з можливих підходів до обчислення фрактальної розмірності компакта. Компакт-метризуючий бікомпактний хаусдорфовий простір. Бікомпактний простір – топологічний простір, в кожному відкритому покритті якого міститься кінцеве підпокриття того ж простору. Підпростір n-мірного евклідового простору бікомпактний тоді і тільки тоді, коли він замкнутий і обмеженій [5]. Розглянемо сітку, яка покриває весь компакт. Її вузли будемо називати комірками. Кожну комірку, що має з компактом непорожні перетин, будемо вважати за одну точку. Ясно, що саме ця схема реалізується при графічному виведенні компакта на екран як масиву пікселів. «Підрахунок чиста точок в клітинці» означає підрахунок числа пікселів в клітинці.

Для спрощення обчислень будемо вважати клітки квадратними. Розмір L клітини означає число комірок з кожного боку. обмежимося непарними значеннями L; в цьому випадку центральна комірка буде рівновіддалена від усіх сторін. Спочатку підрахуємо ймовірності P(m,L) того, що клітина

розміру L містить m точок (комірок) компакта. Для цього навколо кожної точки фрактала, вважаючи її центральною, побудуємо клітку розміру L і підрахуємо число точок, що попали до неї. Припустмо, що компакт містить M точок. Тоді P(m, L) дорівнює числу клітинок, що містять m точок, m = 1,…, M, поділити на M. Зауважимо, що сума всіх ймовірностей дорівнює одиниці

$$\sum_{m=1}^{M} P(m,L) = 1 \qquad (1.1)$$

N(L) – число клітинок розміра L, необхідних для покриття компакту. Число клітинок розміру L, що мають m точок, дорівнює (M/m)P(,L). Цьому оцінка кількості клітинок, що покривають все зображення дорівнює

$$\langle N(L)\rangle = \sum_{m=1}^{K}\left(\frac{M}{m}\right)P(m,L) = M\sum_{m=1}^{K}\left(\frac{1}{m}\right)P(m,L), \qquad (1.2)$$

Де K – можлива кількість точок в клітині. Звідси слідує

$$\tilde{N}(L) = \sum_{m=1}^{K}\left(\frac{1}{m}\right)P(m,L) \qquad (1.3)$$

Цей вираз також пропорційний L^{-d} та може використовуватись для оцінки фрактальної розмірності $\dim_M(X)$

1.2.2 Оптимальний клітковий метод. Метод 2

Оптимальний клітковий метод представляє собою метод визначення фрактальної розмірності компакта, заснований на математичному визначенні розмірності Маньківського. Нехай F – компакт і $\rho(x,y)$ – метрика (зазвичай використовується евклідова метрика $\rho(x,y) = \sqrt{\sum_i (x_i - y_i)^2}$, визначена на компакті F. Визначмо $N_F(\varepsilon)$ – мінімальна кількість множин з діаметром, не

більшим за ε, необхідних для того, щоб вони покривали F. Оцінку фрактальної розмірності можливо отримати якщо побудувати лінійну регресію для $\log N(\log\varepsilon)$, використовуючи співвідношення $N(\varepsilon) \propto \varepsilon^{-\dim_M}$. Основну частину даного методу складає алгоритм визначення мінімального покриття компакта, який був розроблений авторами. Такий метод визначення взагалі залежить від метрики на F. Однак, згідно з теоремою про еквівалентні метрики (нехай $\rho_1(x,y)$ и $\rho_2(x,y)$ – еквівалентні метрики (тобто $\exists\lambda,\mu>0$ $\lambda\rho_2(x,y) \le \rho_2(x,y)$), F – компакт з розмірністю Мінковського $\dim_M(F) = d$ в ρ_2 – метрикі. Тоді $\dim_M(F) = d$ ρ_1 – метрики) ми можемо використовувати метрики, еквівалентні евклідовій.

1.2.3 Метод Гранссбергера-Прокаччіа. Метод 3

У 1983 р. Грассбергера і Прокаччі припустили алгоритм обчислення кореляційної розмірності. Кореляційна розмірність збігається з фрактальною розмірністю для однорідних фракталів і менше її для мультифракталів. Нехай ми маємо набір точок $\{x_i, i = 1,2,\dots,M\}$. Якщо задати деякі ε, можна використовувати наш вибір даних для оцінки суми $C(\varepsilon)$, що фігурує у визначенні кореляційної розмірності. Маємо: $C(\varepsilon) = \sum_{i=1}^{N(\varepsilon)} p_i^2$ = {вирогідність того, що дві точки розділені відстанню менше ε} = $\lim_{M\to\infty} \frac{1}{M(M-1)} \times$ { число пар $i \ne j$ таких, що $\|x_i - x_j\| < \varepsilon\} = \lim_{M\to\infty} \frac{1}{M(M-1)} \sum_{i,j=1}^{N(\varepsilon)} \theta(\varepsilon - \|x_i - x_j\|) = C_M(\varepsilon)$

Величину $C_M(\varepsilon)$ називають кореляційним інтегралом, який може сгугувати статистичною оцінкою для $C(\varepsilon)$, отже, може бути використаний для обчислення кореляційної розмірності. Оцінку кореляційної розмірності можна отримати, побудувавши лінійну регресію для $\log C_M(\log\varepsilon)$, використовуючи

співвідношення $C(\varepsilon) \propto \varepsilon^{-\dim c}$. Також як і у випадку методу оптимальних клітин, метод Грассбергера-Прокаччіа дозволяє використовувати метрики, еквівалентні евклідовій.

1.2.4 Дослідження алгоритмів обчислення фрактальної розмірності

За допомогою алгоритмів обчислення фрактальної розмірності отримують фрактальную розмірність компакта, як функцію від двох змінних мінімального розміру покриття (LMin) і максимального розміру покриття (LMax), тобто dim (F)≈MDim(LMax, LMin). Мінімальний розмір покриття (LMin) повинен дорівнювати 1 (одній точці), так як у визначенні фрактальної розмірності є відповідна межа $(\mathrm{LMin} \to 0)$. З максимальним розміром покриття (LMax) виникають певні складнощі, так як при підвищенні LMax збільшується похибка апроксимації межі регресійної кривої, але зменшується похибка апроксимації регресійної кривої [4].

Математично це можна записати так:

$$\Delta \dim(\mathrm{LMax}) = \big(\dim(\mathrm{F}) - \mathrm{MDim}(1, \mathrm{LMax})\big) = \\ = \mathrm{P}(\mathrm{LMax}) + \mathrm{U}\left(\frac{1}{\mathrm{LMax}}\right) + \mathrm{I}(\mathrm{F}) \qquad (1.4)$$

де P(LMax) – похибка апроксимації межі регресійної кривої $\mathrm{U}(\frac{1}{\mathrm{LMax}})$ – похибка апроксимації регресійної кривої, $\mathrm{I}(\mathrm{F})$ – непереборна похибка, обумовлена тим, що ми не маємо сам компакт, а тільки його зображення на деякій лінійній сітці. Запишемо властивості функцій

$$\mathrm{P}(\mathrm{LMax}) \text{и } \mathrm{U}\left(\frac{1}{\mathrm{LMax}}\right): \lim_{\mathrm{LMax}\to 0} \mathrm{P}(\mathrm{LMax}) = 0; \lim_{\mathrm{LMax}\to \infty} \mathrm{P}(\mathrm{LMax}) = \infty; \qquad (1.5)$$

$$\lim_{1/\mathrm{Lmax}\to 0} \mathrm{U}(\frac{1}{\mathrm{LMax}}) = 0; \lim_{1/\mathrm{Lmax}\to \infty} \mathrm{U}\left(\frac{1}{\mathrm{LMax}}\right) = \infty; \qquad (1.6)$$

Так як функції P(LMax) и U(1/LMax) зростаючі, то

$$\frac{dP}{d(LMax)} > 0; \frac{dU}{d(1/LMax)} > 0. \quad (1.7)$$

Знайдемо мінімум для $\Delta\dim(LMax)$, тобто

$$\frac{d(\Delta dim)}{d(LMax)} = 0 = \frac{dP}{d(LMax)} + \frac{dU}{d(\frac{1}{LMax})}\frac{-1}{(LMax)^2} \Rightarrow$$

$$\Rightarrow LMax = \sqrt{\frac{dU}{d(\frac{1}{LMax})}\frac{d(LMax)}{dP}} \Rightarrow LMax = \sqrt{\frac{U'}{P'}} = \lambda(\lambda > 0). \quad (1.8)$$

λ – характеристичний параметр компакта і методу обчислення фрактальної розмірності. Даний параметр визначає мінімум похибки для заданих параметрів. Цей параметр істотно залежить як від компакта, так і від методу визначення фрактальної розмірності.

Введемо також параметр щільності заповнення вихідного зображення компакта, $\rho = \frac{\ln(1/2)}{\ln(NCount/Total)}$, $0 < \rho \leq \infty$ – логарифмічна щільність зображення, де NCount – кількість пікселів компакта. Total – загальна кількість пікселів в зображенні [5].

Для класичних кривих дуже малої щільності (ρ <0,13) фрактальная розмірність добре визначається всіма методами з похибкою, що не перевищує 1% (рисунок 1.13) і немає залежності обчисленої розмірності від максимального розміру покриття. При підвищенні щільності (рисунок 1.13 с та d) є залежність обчисленої розмірності від максимального розміру покриття. Для рисунка 1.13 с найменшою похибкою володіє метод 1, більшою похибкою – метод 3. Однак за допомогою всіх методів можна отримати, правильне значення фрактальної розмірності, якщо вибрати максимальний розмір покриття рівний характеристичному параметру (λ = 10-15). На рисунку правильне значення фрактальної розмірності може бути

отримано тільки методами 1 і 2, в даному випадку найменшою похибкою володіє метод 2. При правильному виборі максимального розміру покриття для даних об'єктів (LMax < λ) можна оцінити фрактальну розмірність з точністю 5%.

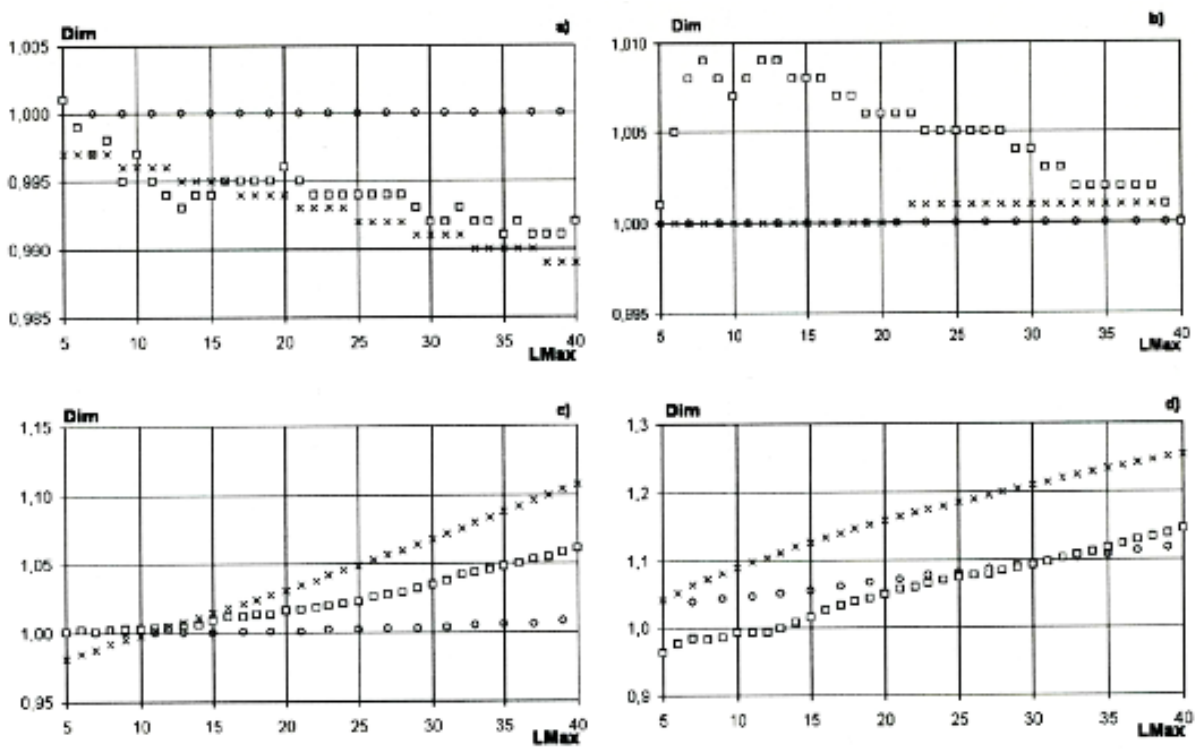

Рисунок 1.13 – Графіки залежності фрактальної розмірності від максимального розміру покриття для класичних кривих $(\rho < 0{,}2)$

Для сімейства класичних кривих високої щільності (рисунок 1.14) має місце яскраво виражена залежність фрактальної розмірності від максимального розміру покриття, і при розмірі покриття, що перевищує характеристичний параметр, правильне визначення фрактальної розмірності неможливо. Метод 1 (λ = 15-20) має найбільший характеристичний параметр, що перевищує характеристичні параметри методів 2 (λ = 10) і 3 (λ = 7-10) практично в 2 рази. Найменшу швидкість росту похибки має метод 2. Тому при аналізі зображень високої щільності ($\rho > 0{,}25$) треба ретельно дослідити залежність фрактальної розмірності від максимального розміру покриття і визначати характеристичний

параметр [5]. Для зображень більшої щільності ($\rho > 0,5$) практично неможливо відрізнити об'єкти, які мають різну фрактальную розмірність. На рисунку 1.14 с видно, що для хаотичних класичних кривих визначення фрактальної розмірності є складним (особливо методом 3). Методи 1 і 2 дозволяють визначити фрактальну розмірність квадрата з прийнятною точністю (рисунок 1.14 d), в той час як метод 3 в даному випадку дає лише нижню межу фрактальної розмірності.

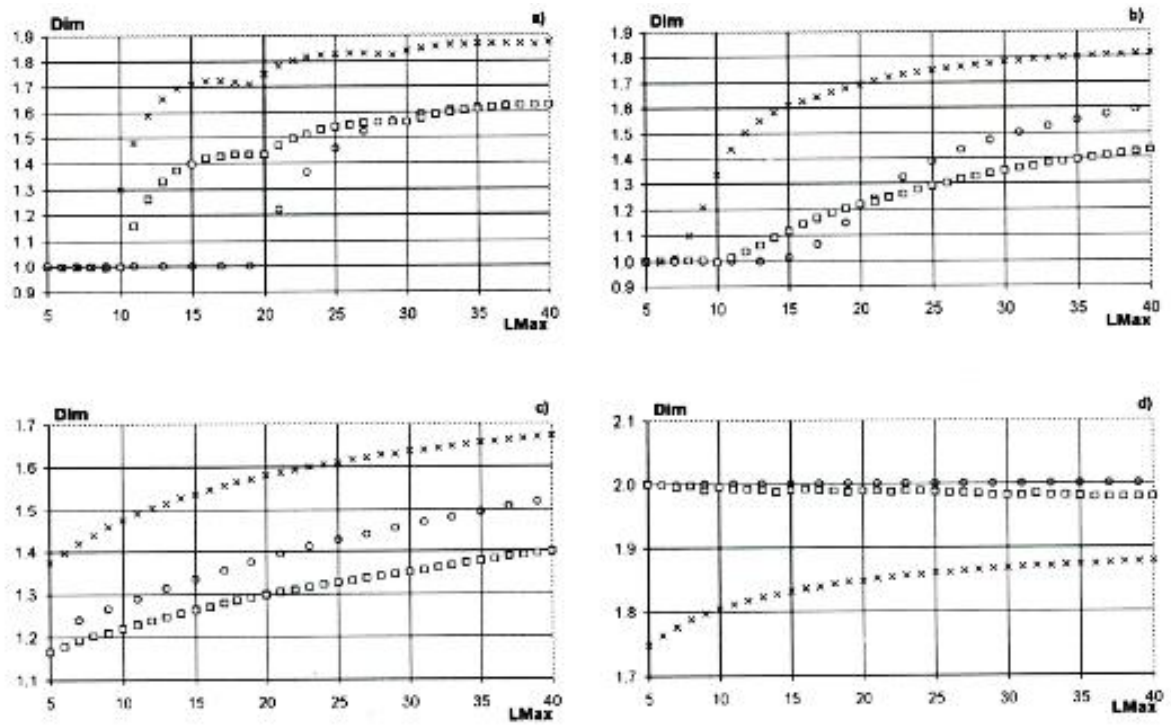

Рисунок 1.14 – Графіки залежності фрактальної розмірності від максимального розміру покриття для кривих великої щільності $(\rho > 0,2)$

Розшифрування умовних позначень:

а) сімейство горизонтальних ліній з відстанню між ними 10 пікселів ($\rho = 0,3010$);

b) сімейство концентричних кілець з різницею радіусів 10 пікселів (ρ= 0,2864);

c) сімейство 100 випадкових ліній ($\rho = 0{,}2772$);

d) повний квадрат розміром 640x480 пікселів ($\rho = \infty$). О – метод 1, □ – метол 2. х – метод 3.

Для математичних «монстрів» (рисунок 1.15) метод 3 дасть оцінку лише нижньої межі фрактальної розмірності, а методи 1 і 2 дають задовільні результати з похибкою, що не перевищує 5%.

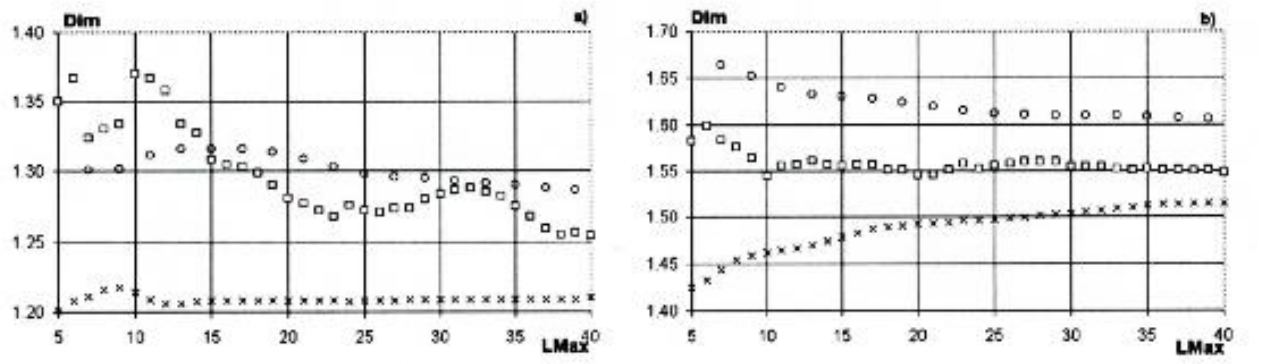

Рисунок 1.15 – Графіки залежності фрактальної розмірності від максимального розміру покриття для «математичних» монстрів

Розшифрування умовних позначень:

a) сніжинка Коха ($\rho = 0{,}1572, \dim_M \ln(4)/\ln(3) \approx 1{,}2618$);

b) Килим Серпиньского ($\rho = 0{,}3016, \dim_M \ln(3)/\ln(2) \approx 1{,}5850$). О– метод 1, □ –метол 2. х – метод 3.

1.2.5 Вибір методу фрактального аналізу

До переваг методу 1 слід віднести більший ніж у інших алгоритмів, характеристичний параметр і найбільшу швидкість обчислення фрактальної розмірності; даний метод можна рекомендувати для найбільш швидкого обчислення фрактальної розмірності.

До переваг методу 2 можна віднести кращі результати обчислення

фрактальної розмірності «математичних» монстрів і найменьший темп зростання похибки, проте даний метод має найменшу швидкість обчислення і менший, ніж у методу 1, характеристичний параметр. Його можна рекомендувати для найбільш точного обчислення фрактальної розмірності, бажано в сукупності з іншими методами.

Метод 3 не дозволяє визначати фрактальную розмірність, якщо вона близька до двох, однак він дасть оцінку нижньої межі фрактальної розмірності, що може бути корисно для додаткового аналізу. Цей метод можна використовувати для оцінки нижньої межі фрактальної розмірності і не слід використовувати для самостійних обчислень.

При обробці реальних зображень необхідно спочатку виділив, контури і перевести зображення в двокольорове. Далі побудувати графіки залежності обчисленої фрактальної розмірності від максимального розміру покриття для всіх методів. Метод 3 використовується для оцінки нижньої межі фрактальної розмірності; в разі, коли метод 3 дає результати більші, ніж методи 1 і 2, фрактальную розмірність неможливо визначити за наявними даними.

1.3 Методи класифікації результатів аналізу зображень

На сьогоднішній час для задач класифікації популярним рішенням є використання машинного навчання. Машинне навчання — це підгалузь штучного інтелекту в галузі інформатики, яка часто застосовує статистичні прийоми для надання комп'ютерам здатності «навчатися» (тобто, поступово покращувати продуктивність у певній задачі) з даних, без того, щоби бути програмованими явно. Зокрема це актуально для нерегулярних зображень та об’єктів з великої кількістю параметрів, що не мають чітких та явних закономірностей.

1.3.1 Метод логістичної регресії

Логістична регресія або логіт-регресія – статистичний регресійний метод, що застосовують у випадку, коли залежна змінна є категорійною, тобто може набувати тільки двох значень (чи, загальніше, скінченної множини значень). В нашому випадку йде річ про вірогідність того, що деякий орган чи система мають порушення, можна розбити на категорії «хворий» та «здоровий». Також, окрім визначення до якого значення належить змінна, цей метод дозволяє встановити рівень впевненості, орієнтуючись на рівень віддалення від логістичної прямої [6].

Нехай є деяка випадкова величина Y, що може набувати лише двох значень, які, як правило, позначаються цифрами 0 і 1. Нехай ця величина залежить від деякої множини пояснювальних змінних $x = (1, x_1, \dots, x_n)^T$. Залежність Y від $x_1, \dots, x_n$ можна визначити ввівши додаткову змінну y*, де $y^* = \theta^T x = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n + \varepsilon$. Тоді

$$Y = \begin{cases} 0, & y^* \le 0 \\ 1, & y^* > 0 \end{cases} \tag{1.9}$$

При визначенні логістичної моделі стохастичний доданок ε вважається випадковою величиною з логістичним розподілом ймовірностей. Відповідно для певних конкретних значень змінних $x^* = x^*, \dots, x^n$ одержується відповідне значення y* і ймовірність того, що Y=1, така

$$p(Y = 1) = p(y^* > 0) = p(\theta^T x^* + \varepsilon > 0) =$$
$$= p(\varepsilon > -\theta^T x^*) = p(\varepsilon \le \theta^T x^*) = \Lambda(\theta^T x^*). \tag{1.10}$$

Передостання рівність випливає з симетричності логістичного розподілу, Λ позначає логістичну функцію — функцію розподілу

логістичного розподілу:

$$\Lambda(x) = \frac{e^x}{1+e^x} = \frac{1}{1+e^{-x}} \qquad (1.11)$$

Таким чином, для конкретного значення x^i випадкова величина Y^i має розподіл Бернуллі: $Y^i \sim B(1, \Lambda(\theta^T x^i))$.

Логіт-модель задовольняє наступній умові

$$\ln \frac{p(1|X)}{1-p(1|X)} = \ln \frac{p(1|X)}{p(0|X)} = b_0 + b_1 x_1 + \cdots + b_j x_j \qquad (1.12)$$

Оцінка параметрів $\theta_0, \theta_1, \ldots, \theta_n$ на основі деякої вибірки $(x^{(1)}, Y^{(1)}), \ldots, (x^{(m)}, Y^{(m)})$, де $x^{(i)} \in R^n$ – вектор значень незалежних змінних, а $Y^{(i)} \in \{0,1\}$ – відповідне їм значення Y, як правило здійснюється за допомогою методу максимальної правдоподібності, згідно з яким вибираються параметри θ, що максимізують значення функції правдоподібності на вибірці

$$\hat{\theta} = \operatorname{argmax}_0 L(\theta) = \operatorname{argmax}_0 \prod_{i=1}^{m} \Pr\{Y = Y^{(i)} | x = x^{(i)}\}. \qquad (1.13)$$

Максимізація функції правдоподібності еквівалентна максимізації її логарифма

$$\log L(\theta) = \sum_{i=1}^{m} \log \Pr\{Y = Y^{(i)} | x = x^{(i)}\} =$$

$$= \sum_{i=1}^{m} Y^{(i)} \log \Lambda(\theta^T x^{(i)}) + (1 - Y^{(i)}) \log\left(1 - \Lambda(\theta^T x^{(i)})\right) \qquad (1.14)$$

Для максимізації цієї функції може бути застосований, наприклад, метод градієнтного спуску, метод Ньютона чи стохастичний градієнтний спуск [6].

Метод логістичної регресії являється поширеним та простим в застосуванні для задач класифікації. Розглянемо приклад на реальних даних Ірисів Фішера.

Завдання класифікації ставиться таким чином: потрібно визначити приналежність до класу Цетоза (Setosa) по довжині і ширині чашолистків.

На графіку (рисунок 1.16) показані результати класифікації. По осі абсцис відкладено значення однієї ознаки (довжина чашолистків в см.), а по осі ординат – значення другої ознаки (ширина чашолистків в см.). Різні класи показані хрестиками різних кольорів, а результат класифікації показаний кружечками відповідного кольору. Пурпурової лінією показана межа між класами, побудована алгоритмом.

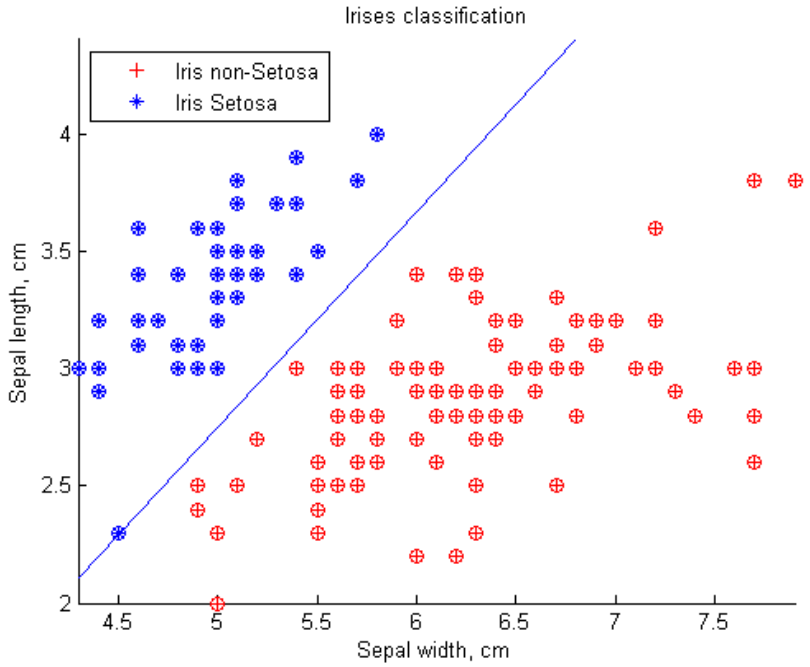

Рисунок 1.16 – Результат класифікації Ірісів Фішера логістичною регресією

Алгоритм не допустив жодної помилки класифікації, оскільки дані лінійно нероздільні. Це просте завдання для будь-якого алгоритму

класифікації. Розглянемо приклад модельованих даних з класами, що сильно перетинаються (рисунок 1.17).

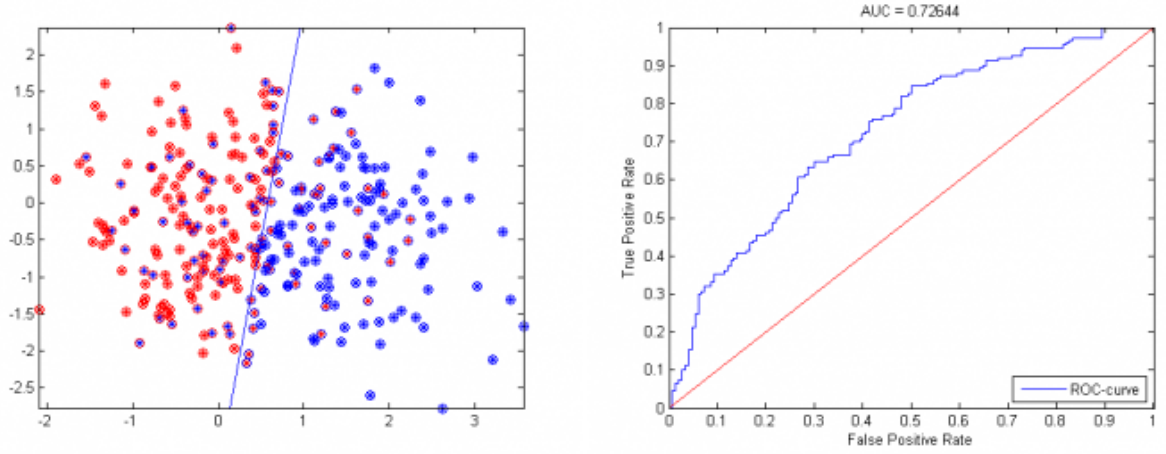

Рисунок 1.17 – Результат класифікації модельованих даних, з класами, що сильно перетинаються, логістичною регресією

Алгоритм допустив близько 30% помилок класифікація, але це і слід було очікувати, тому що вхідні дані були принципово лінійно нероздільні. Для цього алгоритму побудована ROC-крива на цих модельних даних.

1.3.2 Штучні нейронні мережі

Штучні нейронні мережі (ШНМ) – це програмна імплементація нейронних структур нашого мозку. Штучні нейронні мережі імітують поведінку мозку у простішому вигляді. Вони можуть бути навчені контрольованим та неконтрольованим шляхами [7].

У контрольованій ШНМ, мережа навчається шляхом передавання відповідної вхідної інформації та прикладів вихідної інформації. Наприклад, спам-фільтр у електронній поштовій скриньці: вхідною інформацією може бути список слів, які зазвичай містяться у спам-повідомленнях, а вихідною інформацією – класифікація для відповідного повідомлення (спам, чи не спам). Такий вид навчання додає ваги зв'язкам ШНМ. Неконтрольоване навчання у ШНМ намагається "змусити" ШНМ "зрозуміти" структуру

переданої вхідної інформації "самостійно".

ШНМ являє собою систему з'єднаних і взаємодіючих між собою простих процесорів. Такі процесори зазвичай досить прості. Кожен процесор подібної мережі має справу тільки з сигналами, які він періодично отримує, і сигналами, які він періодично посилає іншим процесорам. На рисунку 1.18 зображена схема простої ШНМ

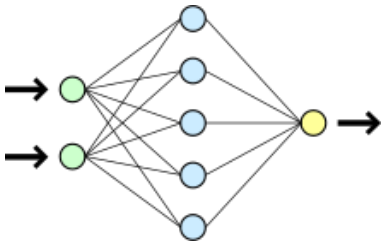

Рисунок 1.18 – Схема простої ШНМ

Нейронні мережі не програмуються в звичному сенсі цього слова, вони навчаються. Можливість навчання – одне з головних переваг нейронних мереж перед традиційними алгоритмами. Технічно навчання полягає в знаходженні коефіцієнтів зв'язків між нейронами [8]. В процесі навчання нейронна мережа здатна виявляти складні залежності між вхідними даними і вихідними, а також виконувати узагальнення. Це означає, що в разі успішного навчання мережа зможе повернути вірний результат на підставі даних, які були відсутні в навчальній вибірці, а також неповних і/або «зашумлених», частково спотворених даних.

1.4 Мета та задача дослідження

В межах даної дослідницької роботи, перед нами стоїть мета розробити апаратно програмний комплекс для розрахунку фрактальної розмірності деякого класу зображень. Ми зупиняємо наш вибір на дослідженні енергограм отриманих в ході енергетичної діагностики методом поляризації діелектриків, оскільки енергетична діагностика є популярним і затребуваним

напрямком в медичній діагностиці людського організму.

Існує ряд методів аналізу енергограм, однак нині в сфері обробки, аналізу та класифікації зображень з’явилося багато нових більш досконалих методів. Крім того об'єм даних, що аналізуються, дуже великий, а фрактальний аналіз зображень є обчислювально-трудомістким завданням. Тобто наш апаратно-програмний комплекс повинен виконувати функцію оптимізації та прискорення. Таким чином актуальність, та новизна нашої роботи полягає в отому щоб застосувати сучасні програмні та апаратні методи в популярній та затребуваній сфері – енергетичній діагностиці.

Об'єктом дослідження є спеціалізованний обчислювач, який має реалізувати фрактально векторній спосіб аналізу зображень нерегулярної структури з властивостями самоподібності.

Предмет дослідження – це спеціалізований апаратно-програмний комплекс, орієнтований на реалізацію завдання визначення поля фрактальної розмірності для зображень отриманих в результаті енергетичної діагностики, з метою їх подальшої класифікації на базі отриманих матриць фрактальних розмірностей часток зображень.

Для успішного виконання поставленої задачі нам необхідно:

1. Обрати методи попередньої обробки зображень, а саме обрати методи фільтрації, перетворення зображення, виділення контурів зображення;
2. Обрати методи розподілення зображення енергограми на сектори;
3. Здійснити посекторний фрактальний аналіз, методом розрахунку розмірності Мінковського;
4. Результати посектороного фрактального аналізу подати у вигладі набора векторів;
5. За допомогою методів штучних нейронних мереж класифікувати вибірки векторів, для встановлення міри відхилення показників від норми;
6. Обрати апаратні методи прискорення та оптимізації програмних методів аналізу та класифікації, що ми обрали.

2 МЕТОДИ ТА МОДЕЛІ АНАЛІЗУ ЗОБРАЖЕНЬ

2.1 Основні принципи комп'ютерної обробки зображень

Комп'ютерне зір – одна з наймолодших областей, яка використовує результати багатьох суміжних наукових напрямків таких, як машинне навчання, проектна геометрія, теорія ймовірності і т.п. Основне коло завдань комп'ютерного зору лежить в сфері якісного і кількісного аналізу зображень і потоків відеоданих (пошук і класифікація об'єктів, супровід об'єктів на відео та ін.). Більшість методів вирішення зазначених завдань тих чи чином використовує різні підходи для виконання попередньої обробки зображень або кадрів відеопотоку з метою подальшого підвищення якості роботи алгоритмів.

У даній роботі ми використовуємо ряд методів попередньої обробки, для покращення чіткості зображення, порогової обробки, виділення контурів та використовуємо алгоритми співвідношення зображень та повороту зображень. Реалізація методів попередньої обробки здійснюється з використанням відповідних операцій на базі засобів відкритої бібліотеки комп'ютерного зору OpenCV [9].

Передобробка має на увазі перетворення вихідного зображення в деяке нове зображення. Безумовно, найбільш популярним способом обробки є фільтрація, яка в більшості зостосунків використовується для видалення шумів. Результат фільтрації – зображення того ж розміру, що і вихідне, але містить значення інтенсивностей пікселів, оновлені відповідно до деякого правила.

Лінійні фільтри – найпростіші представники даного класу методів попередньої обробки. Лінійна фільтрація зводиться до перерахунку значень інтенсивності кожного пікселя зображення за допомогою обчислення зваженої згортки інтенсивностей пікселів, що належать деякій його околиці.

Розмиття або згладжування – ще один підхід в передобробці. Згладжування подібно лінійної фільтрації, в найпростішому випадку передбачає згортку з рівними ваговими коефіцієнтами, в більш складних додатках – згортку з дискретними значеннями функції розподілу Гаусса або вибір медіани серед набору інтенсивностей в околиці [9].

Морфологічні операції (ерозія, дилатація) замість обчислення згорток виконують пошук мінімального або максимального значення інтенсивності у фіксованій околиці кожного пікселя. Якщо ерозія і дилатація ефективно працюють на чорно-білих зображеннях, то для напівтонових і кольорових існують більш складні морфологічні перетворення такі, як замикання, розмикання і т.п.

Виділення країв (або ребер) об'єктів на зображеннях – принципово інше завдання, що виникає в процесі попередньої обробки тому на виході формується не просто перетворене зображення, а карта меж об'єктів. Типові методи виділення ребер базуються на застосуванні до вихідного зображення оператора Собеля (оператор перших похідних за напрямками) або дискретного оператора Лапласа (оператор других похідних). На даний момент найбільш відомим детектором ребер є детектор Канни [9].

Слабкий контраст – поширений дефект зображень і кадрів відео. Існує три основні методи підвищення контрасту зображення: лінійна розтяжка гістограми (лінійне контрастування), нормалізація гістограми, вирівнювання (лінеаризація або еквалізація, equalization) гістограми.

2.2 Порогова обробка напівтонових зображень

Напівтоновими перетвореннями називають такі перетвореннями при яких зображення, що має багато градацій яскравості трансформується в двохградаційне.

Таке перетворення здійснюється в першу чергу для того, щоб скоротити інформаційну надмірність, залишити тільки ту інформацію, яка

потрібна для вирішення конкретної задачі. У бінарному зображенні повинні бути збережені деталі, що нас цікавлять (наприклад, обриси зображених об'єктів) і виключені несуттєві особливості (наприклад, фон).

Порогова обробка півтонування полягає в поділі всіх елементів зображення на два класи за ознакою яскравості, тобто у виконанні поелементного перетворення за пороговим принципом.

В бібліотеці OpenCV порогове перетворення здійснюється функцією «cvThreshold», що виконує фіксоване граничне перетворення для елементів масиву.

Листинг 2.1 – Порогове перетворення в бібліотеці OpenCV

```
CVAPI(double)  cvThreshold( const CvArr*  src, CvArr*  dst,
                            double  threshold, double  max_value,
                            int threshold_type );
```

Розглянемо параметри даної функції:

– src – вихідний масив (зображення) (одноканальне, 8-бітове або 32-бітове)

– dst – цільової масив, повинен мати той же тип що і src або 8-бітний

– threshold – порогова величина

– max_value – максимальне значення

– threshold_type – тип порогового перетворення (листінг 2.2)

Листинг 2.2 – Тип порогового перетворення

```
      #define CV_THRESH_BINARY 0/* value = value > threshold ? max_value : 0
*/
      #define  CV_THRESH_BINARY_INV  1  /*  value  =  value>threshold?0  :
max_value*/
      #define CV_THRESH_TRUNC 2/*value =value > threshold ? threshold :value
*/
      #define CV_THRESH_TOZERO 3 /* value = value > threshold ? value : */
      #define CV_THRESH_TOZERO_INV  4  /* value = value > threshold ? 0 : */
      #define CV_THRESH_MASK 7
      #define CV_THRESH_OTSU 8 /* use Otsu algorithm to choose the optimal
threshold value;
      combine the flag with one of the above CV_THRESH_* values */
```

В результаті порогового перетворення (рисунок 2.1) вхідного зображення (зліва) ми отримали півтонове вихідне зображення (справа).

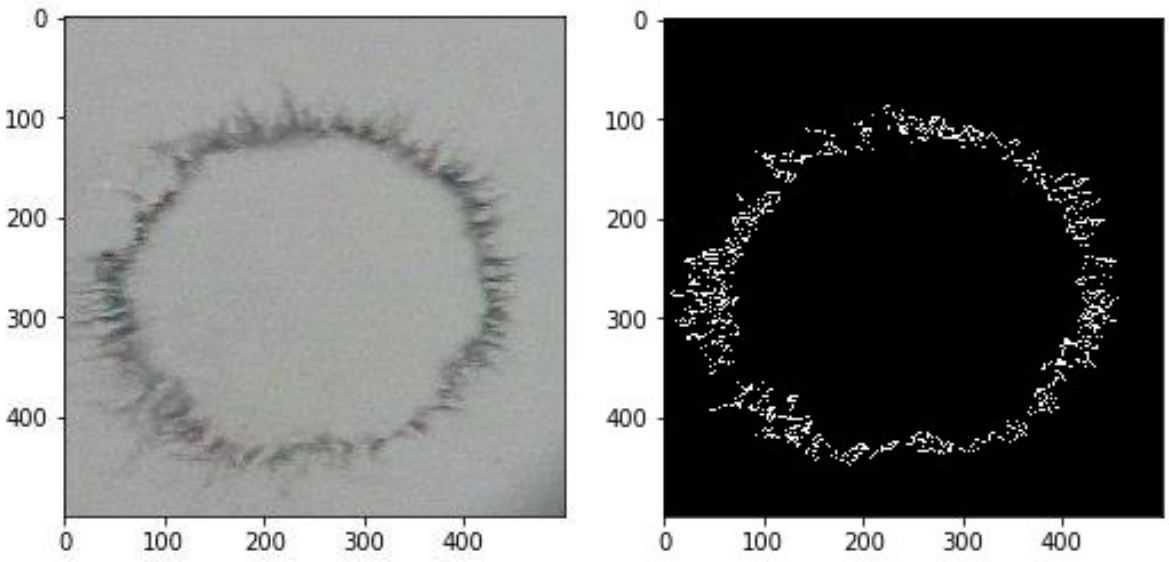

Рисунок 2.1 – Порогове перетворення

2.3 Морфологічні перетворення

2.3.1 Дилатація

Дилатація (морфологічний розширення) – згортка зображення або виділеної області зображення з деяким ядром. Ядро може мати довільну форму і розмір. При цьому в ядрі виділяється єдина провідна позиція (anchor), яка поєднується з поточним пикселем при обчисленні згортки. У багатьох випадках в якості ядра вибирається квадрат або коло з провідною позицією в центрі. Ядро можна розглядати як шаблон або маску. Застосування дилатації зводиться до проходу шаблоном по всьому зображенню і застосування оператора пошуку локального максимуму до интенсивностям пікселів зображення, які накриваються шаблоном. Така операція викликає зростання світлих областей на зображенні (рисунок 2.2, С). На рисунку сірим кольором відзначені пікселі, які в результаті застосування дилатації будуть білими [9].

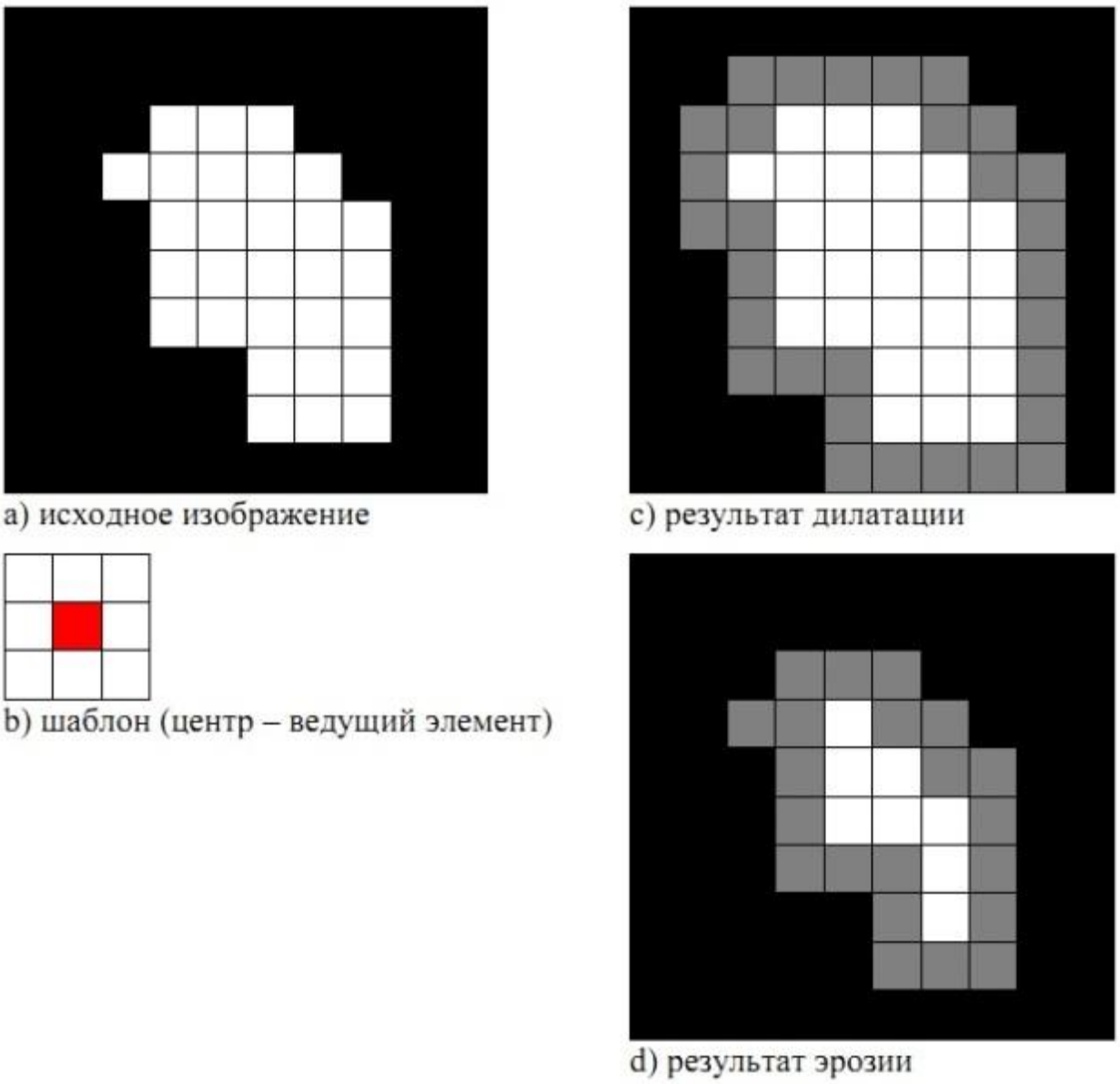

Рисунок 2.2 – Морфологічні перетворення – дилатація та ерозія

2.3.2 Ерозія

Ерозія (морфологічний звуження) – зворотна операція. Дія ерозії подібно дилатації, різниця лише в тому, що використовується оператор пошуку локального мінімуму (рисунок 2.2, d), сірим кольором залиті пікселі, які стануть чорними в результаті ерозії.

Прототипи відповідних функцій ерозії і дилатації, реалізовані в OpenCV, представлені лістингом 2.3 [7].

Лістинг 2.3 – Прототипи функцій ерозії і дилатації

```
void dilate (const Mat & src, Mat & dst, const Mat & element,
  Point anchor = Point (-1, -1), int iterations = 1,
  int borderType = BORDER_CONSTANT,
  const Scalar & borderValue =
    morphologyDefaultBorderValue ())

void erode (const Mat & src, Mat & dst, const Mat & element,
  Point anchor = Point (-1, -1), int iterations = 1,
  int borderType = BORDER_CONSTANT,
  const Scalar & borderValue =
  morphologyDefaultBorderValue ())
```

Розглянемо параметри даної функції:

– src – вихідне зображення.

– dst – результуюче зображення, має такий же розмір, що і вхідне зображення. Відзначимо, що результат операції може записуватися в вихідне зображення.

– element – шаблон, який використовується в процесі дилатації. Якщо element = Mat () , то застосовується квадратний шаблон розміром 3x3.

– anchor – позиція провідного пікселя в структурному елементі. Значення за замовчуванням (-1, -1) означає, що в якості ведучого елемента вибирається центр шаблону.

– interations – кількість разів, яке застосовується дилатація / ерозія.

– borderType – параметр, що визначає метод доповнення кордону, щоб можна було застосовувати дилатацію / ерозію до граничних пикселям вихідного зображення. Приймає будь-яке значення виду BORDER_ * за винятком BORDER_TRANSPARENT і BORDER_ISOLATED [9].

– borderValue – розмір кордону в разі, якщо вона має постійний розмір. Значення за замовчуванням дорівнює morphologyDefaultBorderValue , перетворюється в -inf для дилатації і + inf для ерозії. При використанні значення за замовчуванням операція застосовується тільки до внутрішніх пикселям зображень.

Для виконання нашої задачі ми використовуємо тільки функцію ерозії (рисунок 2.3), для морфологічного звуження, у нашому випадку дилатація може спотворити зображення, у той час як ерозія навпаки – прибирає шуми.

Фільтр Ерозія зменшує область зображення, приводячи до витончення пікселів, розширюючи і посилюючи світлі місця на зображенні. Суть даного перетворення полягає в тому, що небажані вкраплення і шуми розмиваються, а великі і, відповідно, значні ділянки зображення змінам не піддаються.

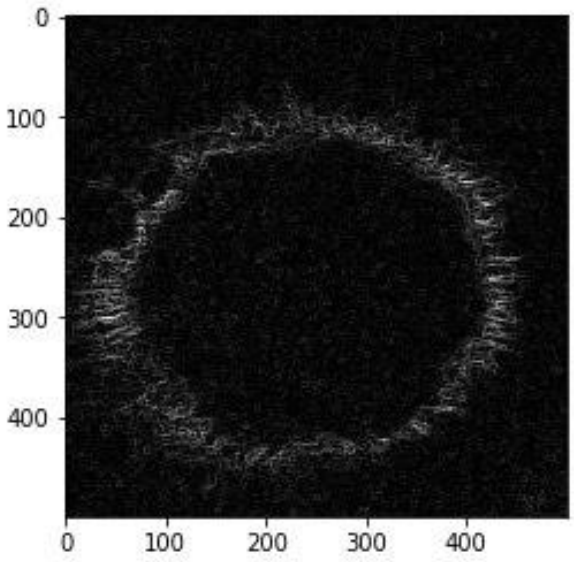

Рисунок 2.3 – Результат обробки зобряження морфологічною ерозією

2.3.3 Градієнт

Фільтр Градієнт виділяє різкі перепади яскравості на оригінальному документі, тобто підкреслює контури зображення, майже не посилюючи шуми. На чорно-білому зображенні даний оператор буде показувати як швидко змінюється яскравість, тобто він має найбільший результат там, де вихідне зображення змінюється найшвидше.

Перетворення Градієнт віднімає зображення після ерозії з зображення після дилатації. В отриманні морфологічного примітиву беруть участь симетричні примітиви, тому робота фільтра слабо залежить від спрямованості контурів [9].

Бібліотека OpenCV підтримує ряд додаткових морфологічних операцій, які реалізується в функції morphologyEx (лістинг 2.4) [7].

Лістинг 2.4 – Реалізація функції morphologyEx

```
void morphologyEx (const Mat & src, Mat & dst, int op,
    const Mat & element,
    Point anchor = Point (-1, -1),
    int iterations = 1,
    int borderType = BORDER_CONSTANT,
    const Scalar & borderValue =
      morphologyDefaultBorderValue ())
```

Розглянемо параметри функції morphologyEx:

– src, dst, element, anchor, borderType, borderValue мають такий же зміст, що і в функціях обчислення ерозії і дилатації.

– op – тип морфологічної операції.

Ми використовуємо морфологічну операцію MORPH_GRADIENT – морфологічний градієнт. Результатом застосування даної операції (рисунок 2.4) є різниця дилатації і ерозії вихідного зображення з однаковим ядром dst = morph_grad (src, element) = dilate (src, element) – erode (src, element) . Морфологічні градієнт забезпечує пошук контурів об'єктів, розмір яких перевищує розмір ядра.

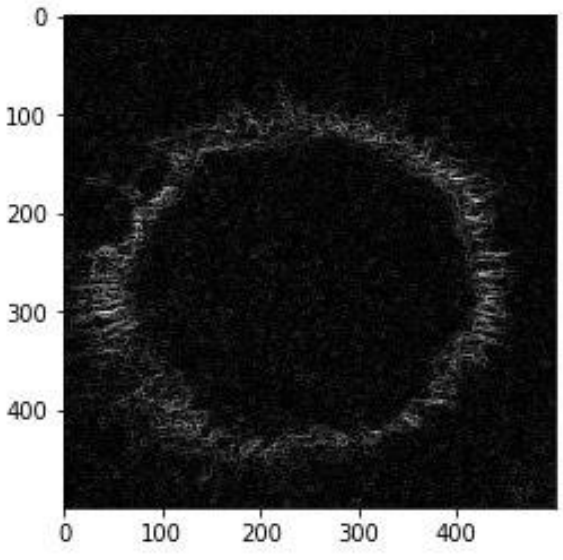

Рисунок 2.4 – Результат обробки зобряження
морфологічним градієнтом

2.4 Оператор Собеля

Оператор Собеля – дискретний диференціальний оператор, який обчислює наближені значення похідних різного порядку для функції яскравості пікселів. Найбільш поширеним прикладом практичного використання є визначення меж (ребер) об'єктів на зображенні, тобто точок різкої зміни яскравості.

Даний оператор заснований на згортку зображення з цілочисельними фільтрами. У найпростішому випадку оператор побудований на обчисленні згорток вихідного зображення з ядрами G_x і G_y, що забезпечують обчислення перших похідних за напрямками

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad (2.1)$$

Даний оператор використовується для наближеного обчислення градієнта функції інтенсивності пікселів. Застосування оператора G_x дозволяє визначити наближене значення першої похідної зміни інтенсивності в горизонтальному напрямку, G_y – в вертикальному. На підставі цієї інформації можна обчислити магнітуду градієнта для пікселя з координатами i, j згідно з формулою

$$|G^{ij}| = \sqrt{(G_x^{ij})^2 + (G_y^{ij})^2} \quad (2.2)$$

Також використовуючи отримані дані, можна визначити напрямок градієнта як $\theta^{ij} = \arctan\left(\frac{G_y}{G_x}\right)$. У бібліотеці OpenCV підтримується

обчислення перших, других, третіх і змішаних похідних функції інтенсивності пікселів з використанням розширеного оператора Собеля [7]. У лістингу 2.5 наведено прототип відповідної програмної функції.

Лістинг 2.5 – Оператор Собеля в бібліотеці OpenCV

```
void Sobel (const Mat & src, Mat & dst, int ddepth,
  int xorder, int yorder, int ksize = 3,
  double scale = 1, double delta = 0,
  int borderType = BORDER_DEFAULT)
```

Перерахуємо вхідні параметри функції Sobel:

– src – вихідне зображення,

– dst – результуюче зображення,

– ddepth – глибина результуючого зображення,

– xorder – порядок похідної по осі Ox,

– yorder – порядок похідної по осі Oy,

– ksize – розмір розширеного ядра оператора Собеля. Приймає одне зі значень 1, 3, 5 або 7.

У всіх випадках ядро має розмір kSize x kSize , крім ситуації, коли kSize = 1 . При kSize = 1 ядра мають розмір 3x1 або 1x3, по суті застосовується фільтр Гаусса. Вказане значення можна використовувати тільки при обчисленні перших і других приватних похідних по осях Ox і Oy. За замовчуванням ядро має розмір 3x3. Відзначимо, що додатково передбачено спеціальне значення параметра kSize = CV_SCHARR = -1 , який відповідає ядру розміру 3x3 фільтра Щарри (Scharr) і може давати більш точні оцінки похідних в порівнянні з оператором Собеля.

scale – опціональний параметр, який задає коефіцієнт масштабування для обчислюваних значень похідних. За замовчуванням масштабування не застосовується. delta – опціональний параметр зміщення інтенсивності, додається перед збереженням результату в матрицю dst. borderType – параметр, що визначає метод доповнення кордону.

В результаті обробки вхідного зображення (рисунок 2.1) ми отримуємо

вихідне зображення (рисунок 2.5) що показує, наскільки «різко» або «плавно» змінюється яскравість зображення в кожній точці.

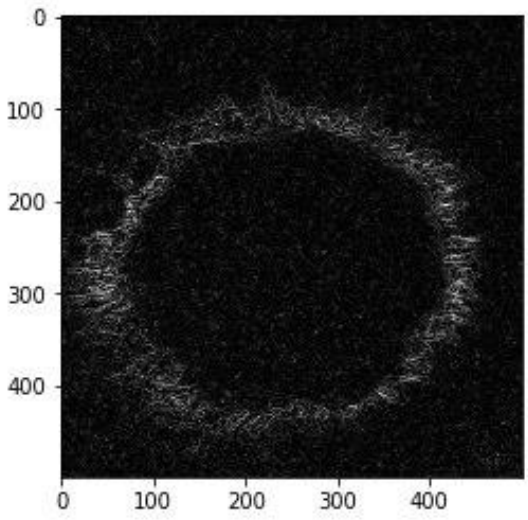

Рисунок 2.5 – Результат обробки зображення оператором Собеля

2.5 Детектор ребер Кенні

Оператор Кенні в дисципліні комп'ютерного зору – оператор виявлення кордонів зображення. Був розроблений в 1986 році Джоном Кенні і використовує багатоступінчастий алгоритм для виявлення широкого спектра кордонів в зображеннях.

Кенні дослідив математичну проблему отримання фільтра, оптимального за критеріями виділення, локалізації та мінімізації декількох відгуків одного краю. Він показав, що шуканий фільтр є сумою чотирьох експонент. Він також показав, що цей фільтр може бути добре наближений першої похідної Гауссіана. Кенні ввів поняття придушення немаксимумів, яке означає, що пікселями кордонів оголошуються пікселі, в яких досягається локальний максимум градієнта в напрямку вектора градієнта.

Детектор включає кілька етапів [9]:

1. Видалення шуму на зображенні за допомогою застосування фільтра Гаусса з ядром розміру 5.

2. Обчислення перших похідних (магнітуд і напрямків) функції

інтенсивності пікселів по горизонтальному і вертикальному напрямках за допомогою застосування оператора Собеля з ядрами G_x і G_y. Напрямки градієнтів округлюються до одного з можливих значень:0, 45, 90, 135.

3. Відбір пікселів, які потенційно належать ребру з використанням процедури non-maximum suppression. Пікселі, яким відповідають вектора похідних за напрямками, які є локальними максимумами, вважаються потенційними кандидатами на приналежність ребру.

4. Подвійне відсікання (гістерезис). Виділяються "сильні" і "слабкі" ребра. Пікселі, інтенсивність яких перевищує максимальний поріг, вважаються пікселями, які належать "сильним" ребрах. Приймається, що пікселі з інтенсивністю, що входить в інтервал від мінімального до максимального граничного значення, належать "слабким" ребрах. Пікселі, інтенсивність яких менше мінімального порогу, відкидаються з подальшого розгляду. Результуючі ребра містять пікселі всіх "сильних" ребер і ті пікселі "слабких" ребер, чия околиця містить хоча б один піксель "сильних" ребер.

Детектор Канни реалізований в бібліотеці OpenCV [7] у вигляді окремої функції, прототип якої наведено у лістингу 2.6.

Лістинг 2.6 – Детектор Канни в бібліотеці OpenCV

```
void Canny (const Mat & image, Mat & edges, double threshold1,
  double threshold2, int apertureSize = 3,
  bool L2gradient = false)
```

Функція приймає на вхід наступні параметри:

– image – одноканальне 8-бітове зображення;

– edges – результуюча карта ребер, являв собою таблицю, розмір якої збігається з розміром вихідного зображення;

– threshold1, threshold2 – параметри алгоритму, порогові значення для відсікання;

– apertureSize – розмір апертури для застосування оператора Собеля;

– L2gradient – прапор, який вказує, по якій нормі буде обчислюватися магнітуда градієнта. Приймає справжнє значення, якщо використовується

норма L_2 (корінь квадратний з суми квадратів приватних похідних), в іншому випадку L_1 (сума модулів приватних похідних) [9].

В результаті обробки отримуємо вихідне зображення (рисунок 2.6).

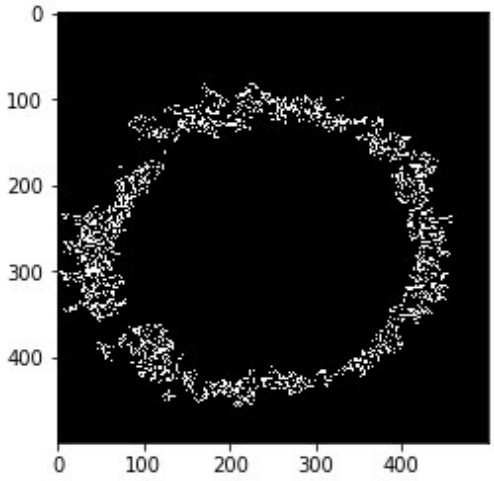

Рисунок 2.6 – Результат обробки вхідного зображення детектором Кенні

2.6 Масштабно-інваріантна трансформація ознак

Масштабно-інваріантна трансформація ознак (англ. Scale-invariant feature transform , SIFT) є алгоритмом виявлення ознак в комп'ютерному зорі для виявлення і опису локальних ознак в зображеннях.

Спочатку в SIFT витягуються ключові точки об'єктів з набору контрольних зображень і запам'ятовуються в базі даних. Об'єкт розпізнається в новому зображенні шляхом порівнювання кожної ознаки з нового зображення з ознаками з бази даних і знаходження ознак-кандидатів на основі евклидової відстані між векторами ознак.

З повного набору відповідностей в новому зображенні відбираються піднабори ключових точок, які найбільш добре узгоджуються з об'єктом по його місцю розташування, масштабу і орієнтації. Визначення відповідних блоків ознак здійснюється швидко за допомогою ефективної реалізації хеш-таблиці узагальненого перетворення Хафа.

Кожен блок з 3 або більше ознак, узгоджується з об'єктом і його

положенням, підлягає подальшій перевірки відповідності моделі, і блоки, що різко відхиляються, відкидаються. Нарешті, обчислюється ймовірність, що певний набір ознак говорить про присутність об'єкта, що дає інформацію про точність збігу і серед можливих промахів. Об'єкти, які проходять всі ці тести, можуть вважатися правильними з високим ступенем впевненості.

2.6.1 Пошук ключових точок

Основним моментом у пошуку особливих точок є побудова піраміди гауссіанов (Gaussian) і різниць гауссіанов (Difference of Gaussian, DoG). Гауссіаном (або зображенням, розмитим гаусовим фільтром) є зображення Тут L – значення гауссіана в точці з координатами (x, y), a sigma – радіус розмиття. G – гауссово ядро, I – значення вихідного зображення, * – операція згортки. Різницею гауссіанов називають зображення, отримане шляхом попиксельного віднімання одного гауссіна вихідного зображення з гауссіана з іншим радіусом розмиття.

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{2.3}$$

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned} \tag{2.4}$$

Простором зображення, що масштабується є набір всіляких, згладжених деяким фільтром, версій вихідного зображення. Доведено, що гауссовий простір, що масштабується є лінійним, інваріантним щодо зрушень, обертань, масштабу, що не зміщують локальні екстремуми, і має властивість напівгруп. Для нас важливо, що різна ступінь розмиття зображення гаусовим фільтром може бути прийнята за вихідне зображення, взяте в деякому масштабі.

Загалом, інваріантність щодо масштабу досягається за рахунок знаходження ключових точок для вихідного зображення, взятого в різних

масштабах. Для цього будується піраміда гауссіанов: все масштабується простір розбивається на деякі ділянки – октави, причому частина масштабується простору, займаного наступної октави, в два рази більше частини, займаної попередньої. До того ж, при переході від однієї октави до іншої робиться ресемплінга зображення, його розміри зменшуються вдвічі. Природно, що кожна октава охоплює безліч гауссіанов зображення, тому будується тільки деяке їх кількість N, з певним кроком по радіусу розмиття. З тим же кроком добудовуються два додаткових гауссіана (всього виходить N + 2), що виходять за межі октави. Далі буде видно, навіщо це потрібно

Паралельно з побудовою піраміди гауссианів, будується піраміда різниць гауссианів, що складається з різниць сусідніх зображень в піраміді гауссианів. Відповідно, кількість зображень в цій піраміді буде N + 1.

На малюнку зліва зображено піраміда гауссианів, а праворуч – їх різниць. Схематично показано, що кожна різниця виходить з двох сусідніх гауссианів, кількість різниць на одиницю менше кількості гауссианів, при переході до наступної о Девід Форсайт, Жан Понс

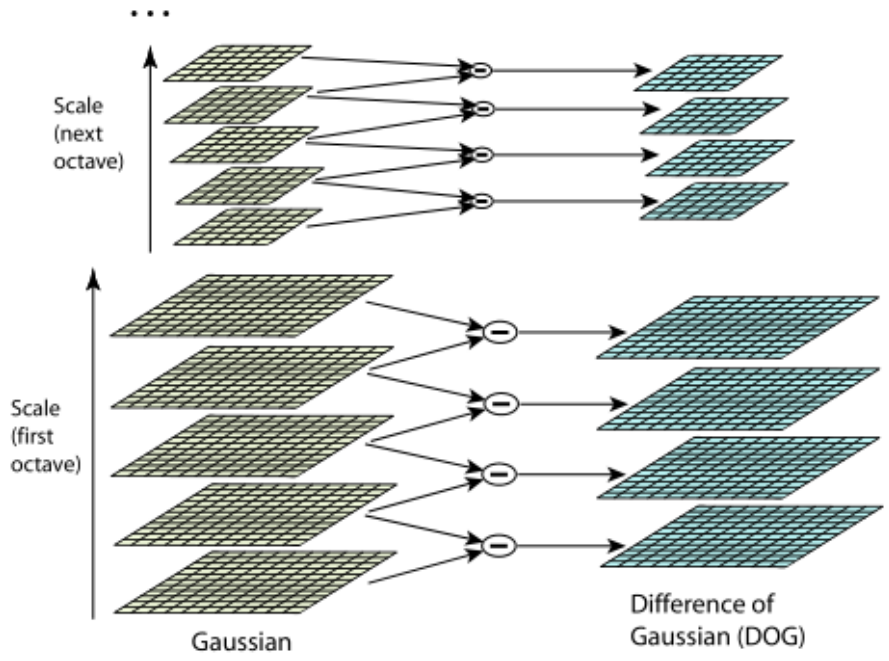

Рисунок 2.7 – Піраміда гауссианів та піраміда різниць гауссианів

Будемо вважати точку особливою, якщо вона є локальним екстремумів

різниці гауссіанов. Для пошуку екстремумів будемо використовувати метод, що схематично зображений на рисунку 2.8.

Якщо значення різниці гауссіанов в точці з позначкою хрестиком, більше (менше) всіх значень в точках, позначених кругляшками, то ця точка вважається точкою екстремуму.

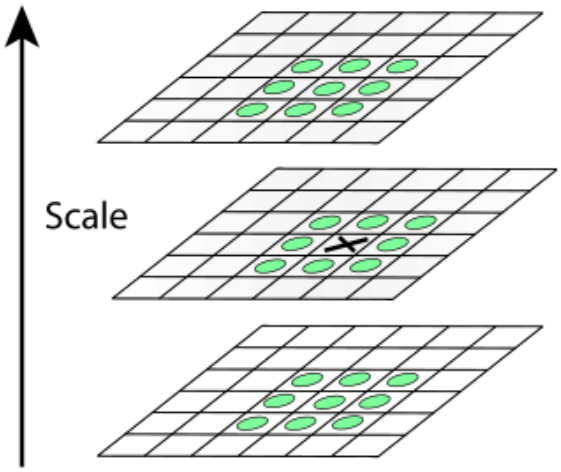

Рисунок 2.8 – Алгоритм пошуку екстремумів

В кожному зображенні з піраміди DoG шукаються точки локального екстремуму. Кожна точка поточного зображення DoG порівнюється з усіма її сусідами і з дев'ятьми сусідами в DoG, що знаходяться на рівень вище і нижче в піраміді. Якщо ця точка більше (менше) всіх сусідів, то вона приймається за точку локального екстремуму [9].

2.6.2 Уточнення ключових точок

Насамперед визначаються координати особливої точки з субпиксельной точністю. Це досягається за допомогою апроксимування функції DoG многочленом Тейлора другого порядку, взятого в точці обчисленого екстремуму. Тут D – функція DoG, X = (x, y, sigma) – вектор зміщення відносно точки розкладання, перша похідна DoG – градієнт, друга похідна DoG – матриця Гессе. Екстремум многочлена Тейлора знаходиться шляхом обчислення похідної та прирівнювання її до нуля. В результаті

отримаємо зсув точки обчисленого екстремуму, щодо точного.

$$D(\mathbf{x}) = D + \frac{\partial D}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^\mathbf{T}\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x} \tag{2.5}$$

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1}\frac{\partial D}{\partial \mathbf{x}} \tag{2.6}$$

Всі похідні обчислюються за формулами кінцевих різниць. У підсумку отримуємо СЛАР розмірності 3x3, щодо компонент вектора X ^. Якщо одна з компонент вектора X ^ більше 0.5 * шаг_сеткі_в_етом_направленіі, то це означає, що насправді крапки екстремуму була обчислена невірно і потрібно зрушити до сусідньої точки в напрямку зазначених компонент. Для сусідньої точки все повторюється заново. Якщо таким чином ми вийшли за межі октави, то слід виключити дану точку з розгляду.Коли становище точки екстремуму обчислено, перевіряється на малість саме значення DoG в цій точці за формулою Якщо ця перевірка не проходить, то точка виключається, як точка з малим контрастом.

2.6.3 Знаходження орієнтації ключової точки

Після того, як ми переконалися, що якась точка є ключовою, потрібно обчислити її орієнтацію. Як буде видно далі, точка може мати кілька напрямків.Напрямок ключовий точки обчислюється виходячи з напрямків градієнтів точок, сусідніх з особливою. Всі обчислення градієнтів виробляються на зображенні в піраміді гауссіанов, з масштабом найбільш близьким до масштабу ключовий точки. Для довідки: величина і напрямок градієнта в точці (x, y) обчислюються за формулами m – величина градієнта, theta – його напрямок

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{2.7}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y))) \tag{2.8}$$

Для початку визначимо вікно (околиця) ключовий точки, в якому будуть розглянуті градієнти. По суті, це буде вікно, необхідну для згортки з гаусовим ядром, причому воно буде круглим і радіус розмиття для цього ядра (sigma) дорівнює 1.5 * масштаб_ключевой_точкі. Для гауссова ядра діє так зване правило «трьох сигм». Воно полягає в тому, що значення гауссова ядра дуже близько до нуля на відстані, що перевищує 3 * sigma. Таким чином, радіус вікна визначається як [3 * sigma].

Напрямок ключовий точки знайдемо з гістограми напрямків O. Гістограма складається з 36 компонент, які рівномірно покривають проміжок в 360 градусів, і формується вона в такий спосіб: кожна точка вікна (x, y) вносить вклад, рівний m * G (x, y, sigma), в ту компоненту гістограми, яка покриває проміжок, що містить напрям градієнта theta (x, y).

Напрямок ключовий точки лежить в проміжку, що покривається максимальної компонентою гістограми. Значення максимальної компоненти (max) і двох сусідніх з нею інтерполюються параболою, і точка максимуму цієї параболи береться в якості напрямку ключовий точки. Якщо в гістограмі є ще компоненти з величинами не менш 0.8 * max, то вони аналогічно інтерполюються і додаткові напрямки приписуються ключовій точці [9].

2.6.4 Побудова дескрипторів

Тепер перейдемо безпосередньо до дескрипторів. Дане раніше визначення говорить про те, що повинен робити дескриптор, але не про те, що це таке. В принципі, дескриптором може виступати будь-який об'єкт (аби він справлявся зі своїми функціями), але зазвичай дескриптором є якась інформація про околиці ключовий точки. Такий вибір зроблено в силу декількох причин: на маленькі області менший вплив роблять ефекти спотворень, деякі зміни (зміна положення об'єкту на зображенні, зміна сцени, перекриття одного об'єкта іншим, поворот) можуть не вплинути на дескриптор зовсім.

У методі SIFT дескриптором є вектор. Як і напрямок ключовий точки,

дескриптор обчислюється на гауссіане, найближчому за масштабом до ключовій точці, і виходячи з градієнтів в деякому вікні ключовий точки. Перед обчисленням дескриптора це вікно повертають на кут напрямку ключовий точки, чим і досягається інваріантність щодо повороту. Для початку подивимося на малюноку (рисунок 2.9)

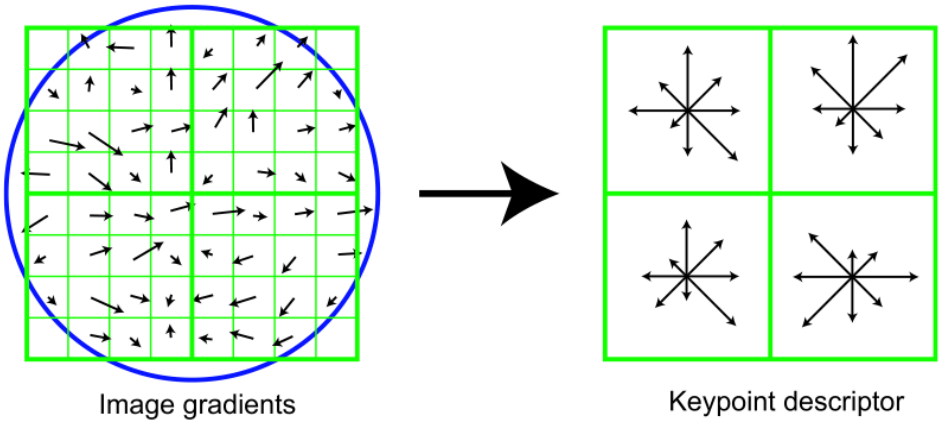

Рисунок 2.9 – Приклад побудови дескриптору

Тут схематично показана частина зображення (зліва) і (праворуч) отриманий на її основі дескриптор. Для початку подивимося наліво. Тут ви можете бачити пікселі, позначені маленькими квадратиками. Ці пікселі беруться з квадратного вікна дескриптора, яке в свою чергу поділено ще на чотири рівні частини (далі будемо називати їх регіонами). Маленька стрілочка, в центрі кожного пікселя позначає градієнт цього пікселя. Цікаво те, що центр цього вікна знаходиться між пікселями. Його треба вибирати якомога ближче до точних координат ключовий точки. Остання деталь, яку можна побачити – це коло, що позначає вікно згортки з гаусовим ядром (аналогічно вікну для обчислення напрямку ключовий точки). Для цього ядра визначається sigma, рівну половині ширини вікна дескриптора.

Тепер подивимося направо. Тут ми можемо бачити схематично зображений дескриптор особливої точки, розмірності 2x2x8. Перші дві цифри в значенні розмірності – це кількість регіонів по горизонталі і вертикалі. Ті квадрати, які охоплювали певний регіон пікселів на лівому зображень,

праворуч охоплюють гістограми, побудовані на пікселях цих регіонів. Відповідно, третя цифра в розмірності дескриптора означає кількість компонент гістограми цих регіонів. Гістограми в регіонах обчислюються так само, як і гістограма напрямків з трьома невеликими але:

1. Кожна гістограма так само покриває ділянку в 360 градусів, але ділить його на 8 частин

2. Як вагового коефіцієнта береться значення гауссова ядра, загального для всього дескриптора (про це вже говорилося)

3. В якості ще одних вагових коефіцієнтів беруться коефіцієнти трилинейной інтерполяції.

Кожному градієнту в вікні дескриптора можна приписати три речові координати (x, y, n), де x – відстань до градієнта по горизонталі, y – відстань по вертикалі, n – відстань до напрямку градієнта в гістограмі (мається на увазі відповідна гістограма дескриптора, в яку вносить вклад цей градієнт). За точку відліку береться лівий нижній кут вікна дескриптора і початкове значення гістограми.

За поодинокі відрізки беруться розміри регіонів по горизонталі і вертикалі для x і y відповідно, і кількість градусів в компоненті гістограми для n. Коефіцієнт трилинейной інтерполяції визначається для кожної координати (x, y, n) градієнта як 1-d, де d дорівнює відстані від координати градієнта до середини того одиничного проміжку в який ця координата потрапила.

Дескриптор ключовий точки складається з усіх отриманих гістограм. Як вже було сказано розмірність дескриптора на малюнку 32 компоненти (2x2x8), але на практиці використовуються дескриптори розмірності 128 компонент (4x4x8).

Отриманий дескриптор нормалізується, після чого всі його компоненти, значення яких більше 0.2, урізаються до значення 0.2 і потім дескриптор нормалізується ще раз. У такому вигляді дескриптори готові до використання.

2.7 Афінні перетворення

Афінні перетворення, іноді перетворення Афіни – перетворення площини або простору в саму себе, при якому паралельні прямі переходять в паралельні прямі, прямі що перетинаються – у пересічні, а перехресні – в перехресні.

Афінні перетворення є перетворення вигляду:

$$f(x) = M * x + v \tag{2.9}$$

де M – оборотня матриця (неспецифічний афінор)

Інакше кажучи, перетворення називається аффінним, якщо його можна отримати наступним чином:

1. Вибрати «новий» базис простору з «новим» початком координат {\ Displaystyle v}v;

2. кожній точці x простору поставити у відповідність точку f (x), Що має ті ж координати щодо «нової» системи координат, що і x в «старій».

Властивості афінних перетворень:

1. При афінному перетворенні пряма переходить в пряму;

2. Якщо розмірність простору n>2, то будь-яке перетворення простору (тобто біекція простору на себе), яке переводить прямі в прямі, є аффінним; це визначення використовується в аксіоматичному побудові афінної геометрії;

3. Афінний перетворення утворюють групу щодо композиції ;

4. Будь-які три точки, що не лежать на одній прямій і їх образи відповідно (що не лежать на одній прямій) однозначно задають Афінний перетворення площині.

Існують два типа афінних перетворень: еквіаффінное перетворення – афінні перетворення, що зберігають площину (також зберігається аффіння довжина) та центроаффінне перетворення – афінні перетворення, що зберігають початок координат.

В нашій роботі ми використовуємо центроафінні перетворення для оберту зображень, що аналізуються на необхудний кут.

3 АПАРАТНО-ПРОГРАМНІ МЕТОДИ ПРИСКОРЕННЯ

3.1 Прискорення обчислення фрактально-векторного аналізу

Алгоритмічна складність нашої реалізації обчислення фрактальної розмірності становить W*W*log(min(H,W)), де H- висота бінарної маски; W - висота бінарної маски; log (min (H, W)) – логарифм мінімальної розмірності.

Така алгоритмічна складність обумовлена необхідністю пошуку співвідношення самоподібності на різних масштабах, логаритмічна функція найефективніша для виконання цієї задачі. В найпростіших реалізаціях ця функція може бути лінійною, однак це уповільнює обчислення, та не дає значних переваг в точності.

3.1.1 Оцінка реалізації аналізу з допомогою .net application

Ми вирішили реалізовувати наш програмний комплекс фрактально-векторного аналізу з допомогою .net application, мовою C#. Однак в результаті дослідження ми виявили, що обчислення проводяться неефективно, оскільки при здійсненні ітерацій кожного разу здійснюється перевірка стосовно виходу за межі масиву даних. Таким чином, всі операції з зображеннями здійснюються повільно, через роботу віртуальної машини. В результаті пошуку способів оптимізації було вирішено здійснювати обчислювання в режимі unsafe, який передбачає можливість не здійснювати перевірку виходу за межі.

3.1.2 Використання Emgu CV

Безпосередня робота в режимі unsafe може призвести до великої кількості значних помилок, тож ми використовуємо Emgu CV, що являється обгорткою, крос-платформним .net доповненням для роботи з бібліотекою OpenCV, яку ми використовуємо як допоміжний інструмент для роботи в режимі unsafe.

Після реалізації аналізу з використанням Emgu CV ми отримали змогу

відокремлювати сектори на етапі векторного аналізу зі значно більншою скорістю. Однак Emgu CV не надає можливість прискорити обчислення фрактальної розмірності, оскільки не надає потрібного функціоналу для здійснення обчислення фрактальної розмірності в режимі unsafe .

3.1.3 Використання NumPy

В результаті пошуку способів оптимізації обчислення фрактальної розмірності було знайдено рішення щодо здійснення підрахунку фрактальної розмірності в режимі unsafe шляхом використання бібліотеки NumPy.

NumPy – це розширення мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами. Однак для цієї бібліотеки не існує обгортки для мови C#, тож було прийнято рішення здійснювати реалізацію фрактально-векторного аналізу мовою Python, а для процесу виділення секторів використовувати безпосередньо бібліотеку OpenCV. Даний підхід в значній мірі прискорив проведення фрактально-векторного аналізу (таблиця 3.1).

Таблиця 3.1 – Порівняльна таблиця швидкості обчислення наведених методів

Реалізація	C# (.net)	C# + Emgu CV	Python+ NumPy + OpenCV
Час (с)	19	12	0.3

Перевага реалізації з допомогою NumPy здійснюється за рахунок винесення всієї роботи з масивами в бібліотеку NumPy, що написана мовою C, де не відбувається перевірки виходу за межі масиву, до того ж, в цілому робота з пам’яттю на C швидше ніж на C #. Наведені розрахунки були здійснені для зображень 500x500 пікселів.

3.1.4 Оптимізація обчислень за допомогою графічних процесорів

Фрактальний аналіз зображень є обчислювально-трудомістким завданням. Для оптимізації та прискорення процесу обробки аналізу розглянемо апаратні засоби відеокарт. Відеокарти є не тільки пристроями для відтворення графічної інформації, а й універсальними обчислювачами з паралельною архітектурою: вони здатні не тільки виконувати задану програму, а й здійснювати операції одночасно над цілим масивом даних [10].

Найбільш поширеними та досконалими рішеннями для паралелізації обчислювальних процесів з передачею даних між CPU і GPU є платформа паралельних обчислень CUDA, що запропонована одним з виробників відеокарт – фірмою NVIDIA.

Для реалізації паралельних обчислень з допомогою CUDA мовою Python використовується бібліотека CuPy, що використовує операції схожі с бібліотекою NumPy.

Розглянемо порівнялюну діаграму CuPy та NumPy щодо швидкості обробки даних (рисунок 3.1).

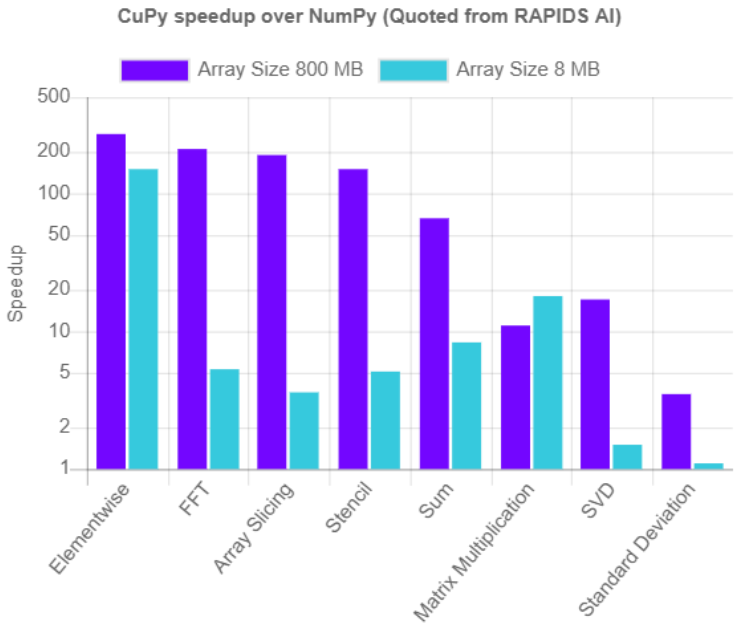

Рисунок 3.1 – Порівняльна діаграма CuPy та NumPy

На горизонтальній осі наведено математичні операції, а по вертикальній – оцінка швидкості винонання обчислень CuPy в порівнянні з NumPy. Темніші стовбці відповідають об’єму даних 800 мб, світліші – об’єму даних 8 мб [11].

З графіку ми бачимо, що чим більше даних передається тим швидше щодо NumPy буде працювати CuPy, це пов'язано в першу чергу з тим, що велика кількість ресурсів необхідна для передачі даних з центрального процесору на графічний і вони можуть перевищувати переваги продуктивності які нам надає CuPy. Проблема вимірів в тому, що вони проводилися на великій кількості даних за один замір і на поточний момент ми не можемо дати необхідне навантаження на відеокарту, щоб повторити приголомшливі показники в швидкості.За документації CuPy всі необхідні нам операції повинні працювати швидше ніж в NumPy, але ми бачимо, що прискорення значніше на великих обсягах даних. В нашому випадку бінарна маска в середньому займає 4 кілобайти і їх передається дуже багато по 9 на кожен сектор. І значно більше часу йде на транспортування кожної окремої маски ніж на обчислення самої фрактальної розмірності.

Цю проблему можливо вирішити передаючи багато масок одним пакетом. Так, за нашими оцінками при передачі від двох тисяч масок одним пакетом переваги в швидкості обчислювання будуть перевищувати витрати ресурсів на передачу даних з центрального процесору на графічний. Такі самі результати ми отримаємо, якщо передавати по 250 масок в одному пакеті, при умові підвищення розширення зображення з 500x500 пікселів до 1000x1000 пікселів. Нижче наведено порівняльну таблицю (таблиця 3.2) швидкість обчислення зображень різного розширення, в середньому маска підсектору еквівалентна зображенню розміром 50x50 пікселів.

Таблиця 3.2 – Швидкість обчислення зображень різного розширення

Розширення	50x50	100x100	300x300	500x500	1000x1000
NumPy	0.01c	0.02c	0.08c	0.20c	0.70c
CuPy	1.3c	1.3c	1.3c	1.4c	1.7c

Ми бачимо що витрати на одну передачу даних для CuPy постійні і рівні 1.3с так само ми бачимо, що кількість часу яке витрачає NumPy зростає швидше ніж CuPy. Цей підхід ми зможемо використовувати в майбутньому коли у нас будуть зображення більш високої роздільної здатності. Так що це рішення буде кращим при дуже великому потоці даних і якщо буде необхідність розвантажити центральний процесор.

3.2 Оптимізація процесу класифікації результатів аналізу

Для класифікації результатів фрактально-векторного аналізу ми вирішили використовувати метод штучних нейронних мереж. Нині існує ряд спеціалізованих апаратів, що прискорюють роботу нейронних мереж.

Нижче наведено порівняльну таблицю існуючого апаратного забезпечення для прискорення роботи нейронних мереж (рисунок 3.2).

Board	MobileNet v1 (ms)	MobileNet v2 (ms)
Coral Dev Board	15.7	20.9
Coral USB Accelerator	49.3	58.1
NVIDIA Jetson Nano (TF)	276.0	309.3
NVIDIA Jetson Nano (TF-TRT)	61.6	72.3
Movidius NCS	115.7	204.5
Intel NCS2	87.2	118.6
MacBook Pro	33.0	71.0
Raspberry Pi	480.3	654.0

Рисунок 3.2 – Порівняльна таблиця акселираторів для ШНМ

Для порівняння використовуються глибокі згорткові нейронні мережі – MobileNet v1 і MobileNet v2, що використовуються для задачі класифікації зображень. Вони досить складні, що б можна було порівняти продуктивність роботи акселираторів, а не похибку у вимірах часу. З цієї таблиці ми можемо зробити висновок, що як окремий акселератор, а не повноцінний одноплатний комп'ютер найшвидшим є Google coral USB accelerator, саме його ми і будемо використовувати в подальших експериментах [12].

Для того щоб здійснювати роботу ШНМ на Google coral USB accelerator нам необхідно виконати ряд операцій, що зображені на Рисунку 3.3.

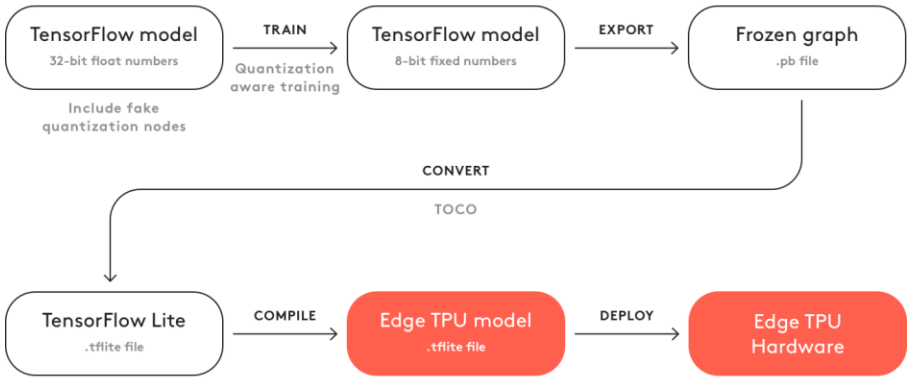

Рисунок 3.3 – Схема запуску роботи ШНМ на акселираторі

Пояснення до схеми:

1. Виконуємо тренування спеціально підготовленної для квантування моделі, тренування відбувається в числах з плающою точкою;

2. Здійснюємо кватнізацію моделі, тепер числа якими оперує модель це int8 а не float32 що істотно знижує місце яке займає модель і прискоює обчислення;

3. Експортуємо моделі в tensorflow .pb файл;

4. За допомогою Tensorflow convertor перетворюємо модель в tensorflow lite модель, яка працює вшивше завдяки внутрішній оптимізації;

5. Компілюємо tensorflow lite модель для конкретного пристрою в нашому випадку це Google coral usb accelerator;

6. Відправляємо модель на наш прискорювач і запускаємо режим прийому даних для їх обробки .

На поточному етапі потреби в апаратному прискоренні для класифікації нейронної мережі немає необхідності, але коли ми додамо ще й згорткову нейронну мережу в комплекс для класифікації, акселиратор прискорить класифікацію в 5 разів і майже повністю зніме навантаження на центральний процессор.

4 ПРОГРАМНА РЕАЛІЗАЦИЯ СПЕЦІАЛІЗОВАННОГО ОБЧИСЛЮВАЧА

Для реалізації спеціалізованого обчислювача було обрано, на наш погляд, оптимальні інструменти, що дозволяють використовувати застосунок декільком користувачам в режимі online, на будь-якій операційній системі.

4.1 Алгоритм роботи користувачів зі застосунком

Ми маємо користувачів двох типів – пацієнт та лікар.

В наведеній нижче схемі (рисунок 4.1) відображено поступові дії користувачів обох типів при роботі зі застосунком.

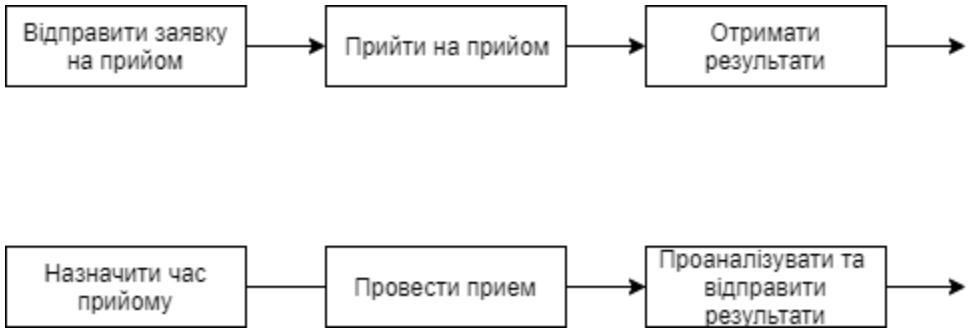

Рисунок 4.1 – Алгоритм роботи цористувачів зі застосунком

Верхня послідовність логічних блоків відображае послідовність дій користувача типу пацієнт, нижча - послідовність дій користувача типу лікар.

Кожен користувач повинен здійснити реєстрацію в системі для початку роботи.

Далі, пацієнт має відправити заявку на прийом до лікаря. Лікар отримує заявку від пацієнта та назначає час прийому.

Далі пацієнт має прийти на прийом, а лікар провести його. Здійснення прийому фіксується в системі.

Результатом прийому є зображення енергограм, що аналізются та

класифікуються апаратно-програмним комплексом, який використовує лікар.

Після того як були отримані результати аналізу лікарт відправляє їх до пацієнта, пацієт отримує результати.

4.2 Програмна реалізація фрактально-векторного аналізу

Нижче наведено код для реалізації обчислення фрактальної розмірності з використанням бібліотеки NumPy (лістинг 4.1) та векторний аналіз (лістинг 4.2) мовою Python.

Лістинг 4.1 – Обчислення фрактальної розмірності мовою Python

```
import scipy.misc
import numpy as np


def fractal_dimension(Z, threshold=0.9):

    # Only for 2d image
    assert(len(Z.shape) == 2)

    # From https://github.com/rougier/numpy-100 (#87)
    def boxcount(Z, k):
        S = np.add.reduceat(
            np.add.reduceat(Z, np.arange(0, Z.shape[0], k),
axis=0),
                                np.arange(0, Z.shape[1], k),
axis=1)

        # We count non-empty (0) and non-full boxes (k*k)
        return len(np.where((S > 0) & (S < k*k))[0])

    # Transform Z into a binary array
    Z = (Z < threshold)
    # Minimal dimension of image
    p = min(Z.shape)
    # Greatest power of 2 less than or equal to p
    n = 2**np.floor(np.log(p)/np.log(2))
    # Extract the exponent
    n = int(np.log(n)/np.log(2))
    # Build successive box sizes (from 2**n down to 2**1)
    sizes = 2**np.arange(n, 1, -1)
    # Actual box counting with decreasing size
    counts = []
    for size in sizes:
        counts.append(boxcount(Z, size))
```

```
        # Fit the successive log(sizes) with log (counts)
        coeffs = np.polyfit(np.log(sizes), np.log(counts), 1)
        return -coeffs[0]
    I = scipy.misc.imread("sierpinski.png")/256.0
    print("Minkowski-Bouligand    dimension    (computed):    ",
fractal_dimension(I))
    print("Haussdorf  dimension  (theoretical):                ",
(np.log(3)/np.log(2)))
```

Лістинг 4.2 – Векторний аналіз секторів мовою Python

```
    sub_sektors                                                =
tools.get_sub_sektors(_f['sektors'][_s]['start'],
_f['sektors'][_s]['end'])

                for _ss in sub_sektors:
                    for _cm in circle_masks:
                        result[_h][_s]['sub_sektors'][i] = {}
                        mask = _ss*_cm
                        img_masked = cv2.bitwise_and(edges,mask)

                        contours,hierarchy                     =
cv2.findContours(mask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
                        cnt = contours[0]
                        x,y,w,h = cv2.boundingRect(cnt)
```

4.3 Класифікація результатів фрактально-векторго аналізу за допомогою ШНМ

Розроблена штучна нейронна мережа (рисунок 4.2) має вхідний вектор з 20 елементів, що включають в себе:

- 9 показників фрактальної розмірності підсекторів;
- 9 показників щільності, що відображають відсоток білих пікселів після фільтрації в бінарній масці відносно площі підсектору;
- 1 показник фрактальної розмірності всього сектора;
- 1 показник щільності всього сектора;

Ми оперуємо трьома повнозв'язними шарами шириною в 20 нейронів.

```
model.summary()
```

```
Model: "L-D-BN-A"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
layer_0_dense (Dense)        multiple                  192
_________________________________________________________________
layer_0_dropout (Dropout)    multiple                  0
_________________________________________________________________
layer_0_BN (BatchNormalizati multiple                  256
_________________________________________________________________
layer_0_activation (Activati multiple                  0
_________________________________________________________________
layer_1_dense (Dense)        multiple                  2080
_________________________________________________________________
layer_1_dropout (Dropout)    multiple                  0
_________________________________________________________________
layer_1_BN (BatchNormalizati multiple                  128
_________________________________________________________________
layer_1_activation (Activati multiple                  0
_________________________________________________________________
layer_2_dense (Dense)        multiple                  264
_________________________________________________________________
layer_2_dropout (Dropout)    multiple                  0
_________________________________________________________________
layer_2_BN (BatchNormalizati multiple                  32
_________________________________________________________________
layer_2_activation (Activati multiple                  0
_________________________________________________________________
Output (Dense)               multiple                  9
=================================================================
Total params: 2,961
Trainable params: 2,753
Non-trainable params: 208
_________________________________________________________________
```

Рисунок 4.2 – Модель розробленої ШНМ

Нами було проведемо навчання розробленної штучної нейронної мережі для класифікації результатів фрактально-векторного аналізу, здіснено тестове випробування, результати наведені на рисунку 4.3, по вертикальній осі вказані оцінки точності, по горизонтальній – покоління випробувань.

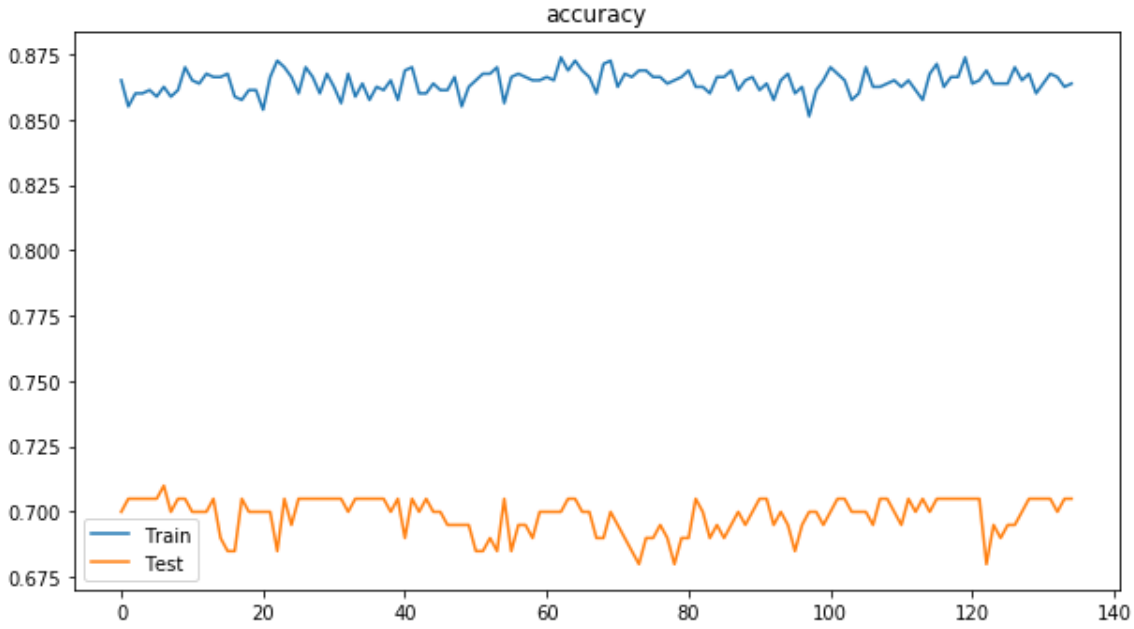

Рисунок 4.3 – Оцінка точності класифікації результатів аналізу методом ШНМ

Для порівняння було здійснено класифікацію отриманих результатів аналізу методом логістичної регрессії (рисунок 4.4).

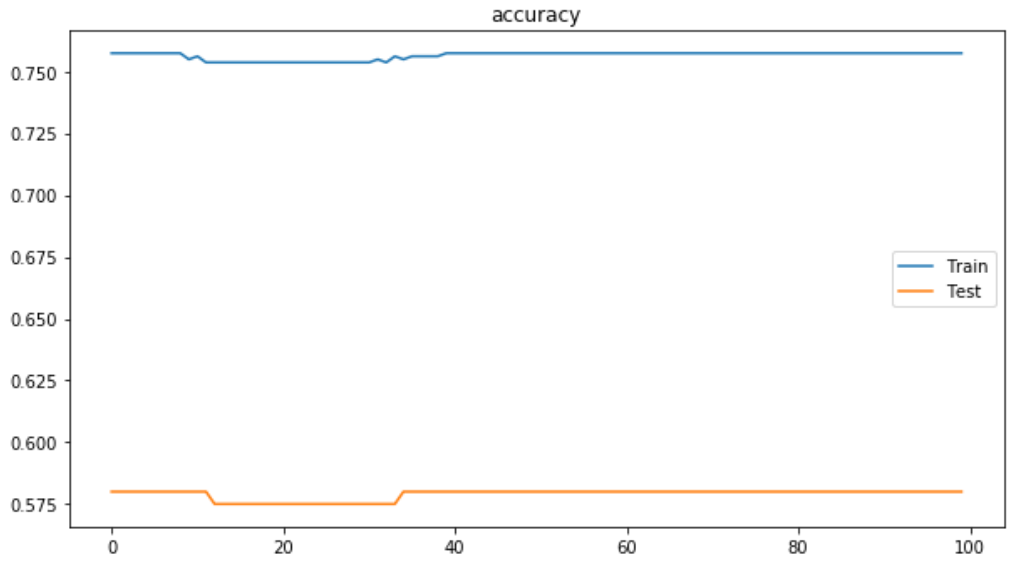

Рисунок 4.4 – Оцінка точності класифікації результатів аналізу методом логістичної регресії

Як ми бачимо при вирішенні поставленої задачі класифікації результатів аналізу, розроблена нами штучна неіронна мережа дає значну перевагу в

точності на 12% відносно логістичної регресії, отже для класифікації результатів аналізу зображень енергограм ми будемо використовувати ШНМ, яку ми пропонуємо прискорити за допомогою google coral usb accelerator. Основні проблеми в точності пов'язані з тим, що у людей які зараз не скаржаться на здоров'я серця можуть бути проблеми пов'язані з серцем, про які вони не знають, це може дуже спотворювати нашу вибірку і заважає коректному проведенню експерименту.

4.4 Архітектура апаратно-програмного комплексу

Клієнтські інтерфейс користувача і програмно-апаратна частина сервісу реалізовані через web-server з допомогою ASP.ner core та Razoz.

Обчислювання, аналіз та класифікація відділені в окремі аналітичні блоки, що можуть горизонтально масштабуватися за допомогою асинхронізації враємодії аналітичної та веб-серверної частин. Нижче наведена схема що відобрає архітектуру розробленного комплексу (рисунок 4.5).

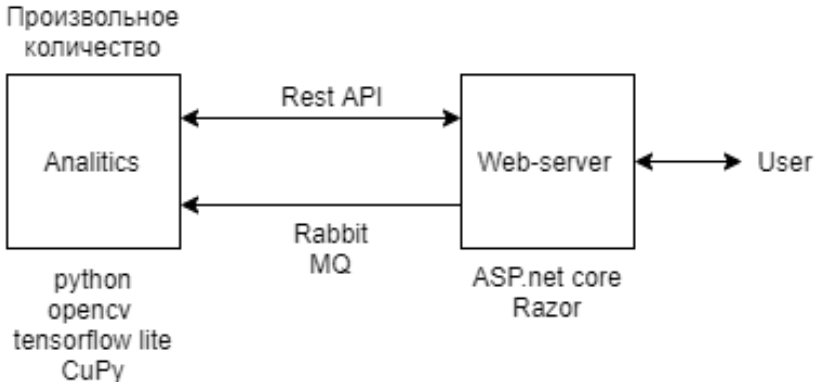

Рисунок 4.5 – Архітектура апаратно-програмного комплексу

Користувач типу лікар завантажує зображення на web-server. Після цього web-server відправив в чергу обробки, за допомогою Rabbit MQ, посилання в якому знаходиться ID зображення, що знаходиться в базі данних (рисунок 4.6). Коли один з блоків аналітики (Analitics) стане вільним, він отримує цей id зображення з черги. Після того як брок analitic звільнився та отримав id

зображення з черги, він через Rest API відправляє запит get на отримання зображення у web-server. Протокол передачі даних Rabbit MQ SMTP неефективний для передачі великих даних, цьому ми в Rabbit MQ посиланні відправляємо тільки id зображення, а зображення передаємо через https, оскілки цей протокол краще підходить для передачі великих данних.

Пілся того як блок analitic отримує зображення через get запит, який знаходиться на web-server, здійснюється аналіз та класифікація зображення. Результат аналітики відправляється через post запит на web-server. Результати аналізу та класифікації надходять до користувача типу лікар.

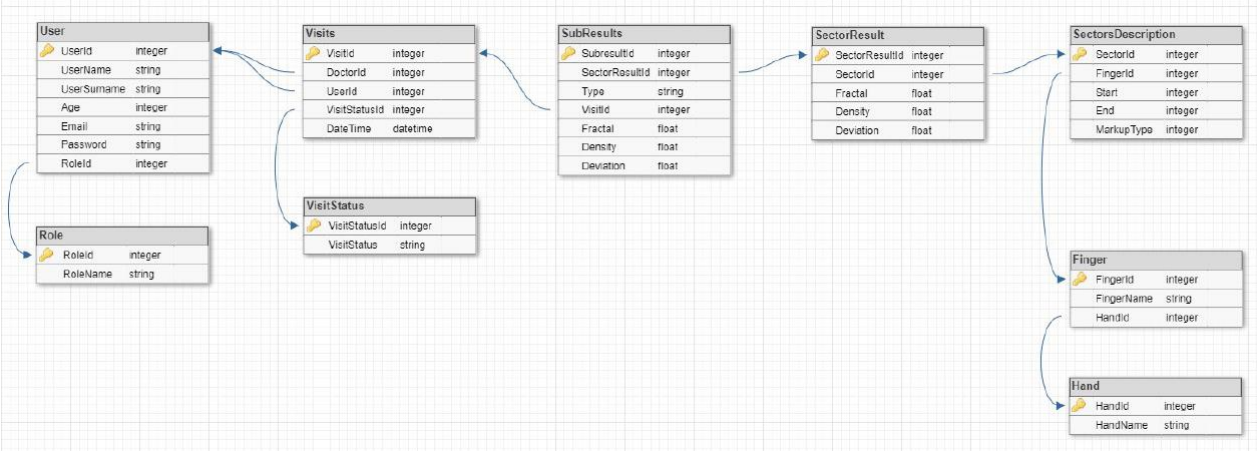

Рисунок 4.6 – Структура бази даних

4.5 Графічний інтерфейс

Клієнтські інтерфейс користувача реалізовано через web-server з допомогою Razoz, а програмна забезпечення - з допомогою ASP.ner core.

На першому етапі користовачі реєструються в системі, наведено приклади успішної (рисунок 4.7) та невдалої реєстрації (рисунок 4.8).

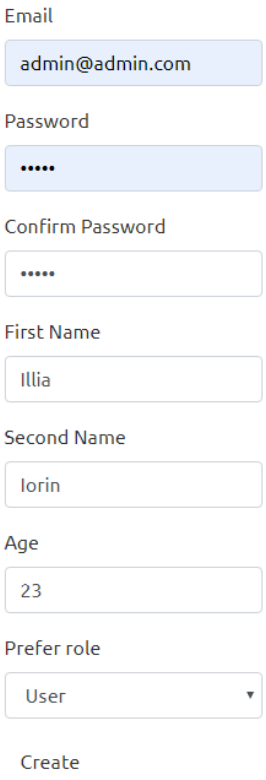

Рисунок 4.7 – Успішна реєстрація

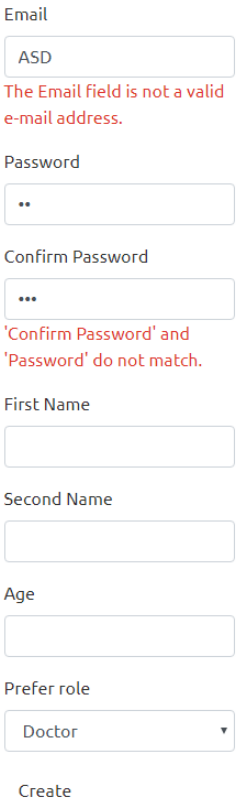

Рисунок 4.8 – Невдала реєстрація

Далі, як ми бачили на схемі алгоритму користування застосунком (рисунок 4.1) користувач типу пацієнт відправляє запит на прийом (рисунок 4.9).

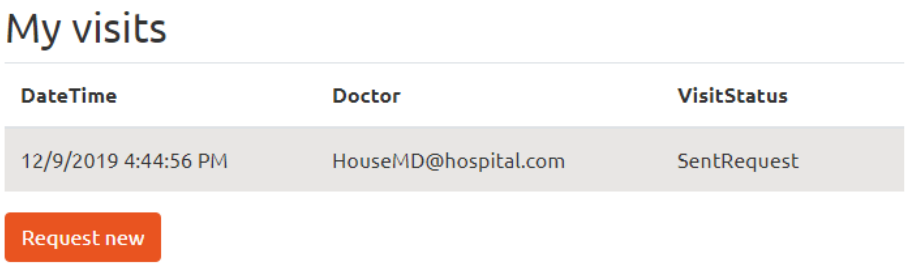

Рисунок 4.9 – Реалізація запиту на прийом від пацієнта

Після того, як лікар назначає час прийому, пацієнт отримує відповідне посилання (рисунок 4.10).

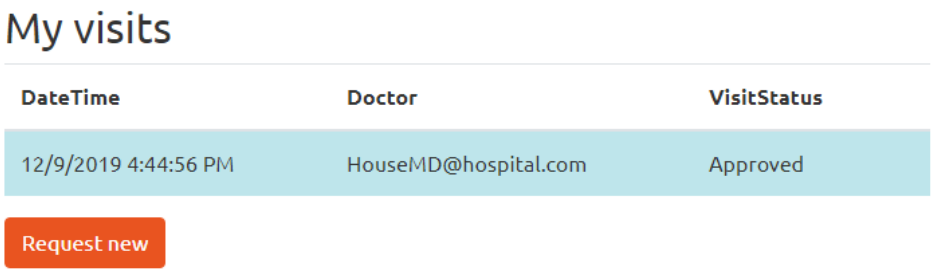

Рисунок 4.10 – Від лікаря назначено час прийому

Проведення прийому фіксується у системі (рисунок 4.11).

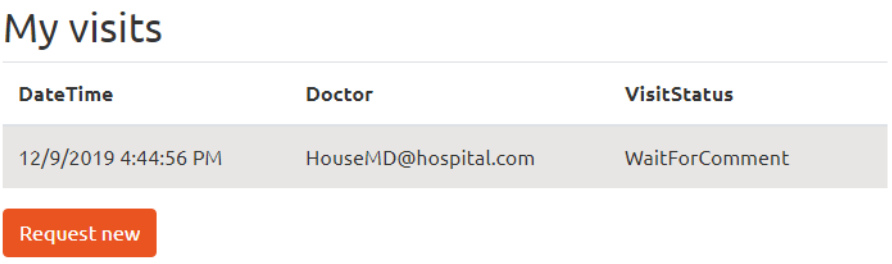

Рисунок 4.11 – Посилання, після здійснення прийому

Після проведення аналізу, пацієнт отримує повідомлення (рисунок 4.12), що результати готові і він може з ними ознайомитись, при натисканні «Result» (рисунок 4.13).

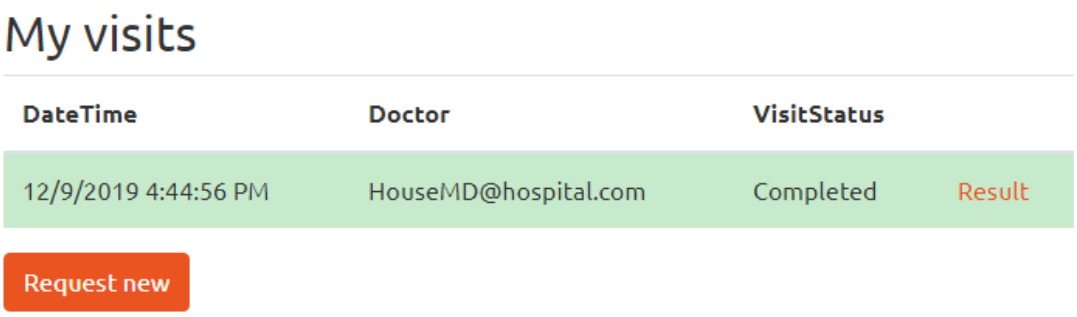

Рисунок 4.12 – Посилання з результатами аналізу

Рисунок 4.13 – Результат роботи програмно-апаратного комплексу

ВИСНОВКИ

В ході виконання атестаційної роботи було виконано дослідження та розробка спеціалізованого апаратно-програмного комплексу, який орієнтований на виконання задачі визначення поля фрактальної розмірності для зображень з елементами самоподібності з метою їх подальшої класифікації на основі одержуваних матриць фрактальних розмірностей фрагментів зображень.

Під час дослідження сучасних методів аналізу та класифікації ми орієнтувались на зображення нерегулярної структури в медицині, а саме на зображення енергограм, що отримуються в ході енергетичної діагностики.

Ми дослідили наявні методи обробки та фрактального аналізу для півтонових зображень з нерегулярною структурою. Також ми провели дослідження щодо розділення енергограм на сектори та підсектори, згідно з теорією П. Манделя щодо енергетичної діагностики, для подальшого векторного аналізу. Також було проведено аналіз сучасних методів програмного та апаратного прискорення обчислювальних процесів за допомогою паралелізації обчислень на графічних процесорах, а також шляхом використання спеціальних пристроїв для прискорення роботи штучних нейронних мереж.

В результаті дослідження та аналізу існуючих методів ми обрали оптимальні рішення для нашої задачі та реалізували їх в апаратно-програмному комплексі спеціалізованого обчислювача. Клієнтський інтерфейс було розроблено на веб сервері з відокремленням блоків обчислювачів з можливістю горизонтального маштабування обчислювачів, завдяки асинхронній взаємодії з клієнтською частиною на веб сервері.

Розроблений апаратно-програмний комплекс вирішує задачі фрактально-векторного аналізу, класифікації методом штучних нейроних мереж та взаємодії в користувачем за допомогою спеціального інтерфейсу.

СПИСОК ВИКОРИСТАННИХ ДЖЕРЕЛ

1. Коротков К. Г. Основы ГРВ биоэлектрографии [Текст] / К. Г. Коротков. – СПб : ИТМО (ТУ), 2001. – 356 с.

2. Коротков К. Г. Эффект Кирлиан / К. Г. Коротков. – СПб : Ольга, 1995. – 218 с.

3. Коротков К. Г. От эффекта Кирлиан к биоэлектрографии [Текст] / К. Г. Коротков. – СПб : Ольга, 1998. – 341 с.

4. Божокин С.В. Фракталы и мультифракталы / С.В. Божокин, Д.А. Паршин // СПб.: Изд-во СПбГУ, 2000. –134с.

5. Кроновер Р.М., Фракталы и хаос в динамических системах. Основы теории. / Р.М. Кроновер. – М.: Постмаркет, 2000. – 352с.

6. Николенко С.А. Глубокое обучение / С.А. Николенко, А.О. Кадурин, Е.Д. Архангельская. – СПб.: Питер, 2018. — 480 с.

9. Крылов Б.А. Методы регистрации, обработки и анализа изображений: Учебно-методическое пособие. / Б.А. Крылов, А.Ю. Гришенцев, Е.Н. Величко. – СПб: СПб ГУ ИТМО, 2010. – 60 с.

10. Сандерс Дж. Технологія CUDA в прикладах. Введення в програмування графічних процесорів. / Дж. Сандерс, Е. Кендрот. – Київ, 2011. – 232с.

11. A NumPy-compatible matrix library accelerated by CUDA [Електронний ресурс]. – Режим доступу до ресурсу: https://cupy.chainer.org/ – 2019.

12. Benchmarking Edge Computing [Електронний ресурс]. – Режим доступу до ресурсу: https://medium.com/@aallan/benchmarking-edge-computing-ce3f13942245 – 2019.

13. Форсайт Д. Комп'ютерний зір. Сучасний підхід. / Д. Форсайт, Ж. Понс. – Київ, 2004. – 943с.

14. Дауні А. Б. Цифрова обробка сигналів на мові Python. /А.Б. Дауні. – Київ, 2017. – 160 с.