МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
Тернопільський національний економічний університет
Навчально-науковий інститут інноваційних освітніх технологій
Кафедра комп'ютерної інженерії

Тишкун Назар Юрійович

Алгоритми скелетизації опуклих об'єктів на основі підходу вписаних квадратів / Algorithms of convex objects skeletalization based on the inscribed squares approach

спеціальність:123 – Комп'ютерна інженерія
освітньо-професійна програма – Комп'ютерна інженерія

Випускна кваліфікаційна робота

Виконав студент групи КІзм-21
Назар Юрійович Тишкун
_____

Науковий керівник:
к.т.н., Батько Ю. М.
_____

ТЕРНОПІЛЬ - 2019

РЕЗЮМЕ

Випускна кваліфікаційна робота на тему “Алгоритми скелетизації опуклих об’єктів на основі підходу вписаних квадратів” зі спеціальності 123 - Комп’ютерна інженерія написана обсягом 93 сторінки і містить 32 ілюстрації, 4 таблиці, 3 додатки та 50 джерел за переліком посилань.

Метою роботи є розробка алгоритму скелетизації об’єктів на цифрових зображннях на основі вписаний квадратів.

Методи досліджень базуються на використанні контурнго аналізу, методів цифрової обробки зображень, теорії алгоритмів та об’єктно-орієнтованого підходу при проектуванні програмної системи.

Розроблено алгоритм скелетизації опуклих об’єктів на цифрових зображення на основі вписаних квадратів для систем цифрової обробки зображень. Програмно реалізовано запропонований алгоритм та проведено на основі розробленої комп’ютерної системи експериментальні дослідження.

Результати роботи можуть бути використані при кодуванні та описі об’єктів на цифрових зображеннях в системах обробки інформації.

Можливими напрямками подальших досліджень є розробка підходів для підвищення точності алгоритмів сегментації областей зі складною контурною лінією, оскільки значні похибки скелетизації відбуваються через спотворення контурної лінії.

КЛЮЧОВІ СЛОВА: ОБРОБКА ЗОБРАЖЕНЬ, СИСТЕМИ ЦИФРОВОЇ ОБРОБКИ ЗОБРАЖЕНЬ, СЕГМЕНТАЦІЯ, КОНТУРНИЙ АНАЛІЗ, СКЕЛЕТИЗАЦІЯ, СКЕЛЕТ.

RESUME

Graduate qualification on the topic “Skeletonization algorithms for convex objects based on the inscribed squares approach” in specialty 123 - Computer Engineering, written in 93 pages, containing 32 illustrations, 4 tables, 3 appendices and 50 references in the list of links.

The aim of work is to develop an algorithm for skeletonization of objects on digital images based on inscribed squares.

Research methods are based on the use of contouring analysis, digital image processing methods, algorithm theory, and object-oriented approach in the design of a software system.

An algorithm for skeletonization of convex objects on digital images based on inscribed squares for digital image processing systems has been developed. The proposed algorithm was programmatically implemented and experimental studies were conducted on the basis of the computer system developed.

The results can be used to encode and describe objects on digital images in information systems.

Possible areas for further research are the development of approaches to improve the accuracy of segmentation algorithms for areas with complex contour lines, since significant skeletalization errors are due to contour line distortion.

KEYWORDS: IMAGE PROCESSING, DIGITAL IMAGE PROCESSING SYSTEMS, SEGMENTATION, BACKGROUND ANALYSIS, SKELETIZATION, SKELET.

ЗМІСТ

ВСТУП

Актуальність роботи. Вищого свого розвитку програмні системи й комп'ютери 5-го і 6-го поколінь досягли, вирішуючи проблеми комутаційної гнучкості й ущільнення нейроподібними методами й засобами обробки та передачі інформації, особливо вражаючі результати отримані при реалізації волоконно-оптичних око-процесорних комп'ютерних мереж, у тому числі з логіко-часовим кодуванням. Передача зображень по комп'ютерних мережах неможлива без попереднього ущільнення. Розроблено ряд стандартів і форматів файлів, які забезпечують передачу зображень у стислому вигляді по мережах. Однак, досягнуті коефіцієнти ущільнення зображень у порівнянні з необхідними обсягами передачі даних й пропускною здатністю каналів недостатні. Щоб максимально використати такі можливості необхідне збільшення коефіцієнта ущільнення зображень при збереженні високої якості, що є пріоритетним напрямком досліджень, які виконуються в області кодування зображень. Аналіз традиційних підходів до ущільнення зображень показав, що вони вичерпали свої можливості в цьому напрямку, тому потрібні нові підходи до вирішення завдань ущільнення при передачі й обробці зображень. Сьогодні синтез і аналіз зображень в основному реалізується за допомогою алгоритмічної обробки на ПК. У зв'язку з тим, що зображення стають масовою продукцією в промисловості і науці, а їх обробка, розпізнавання й аналіз – масовим потоковим виробництвом, – цифрова обробка зображень стає економічно вигідною і необхідною скрізь, де вона технічно можлива. Вирішення проблем обробки і перетворення великих масивів інформації шляхом застосування класичних прийомів програмного керування виявляється важким, особливо при введенні і виведені зображень у нестабільних ситуаціях.

Метою роботи є розробка алгоритму скелетизації об’єктів на цифрових зображннях на основі вписаний квадратів.

Для досягнення даної мети ставились наступні завдання:

– провести комплексний аналіз задач цифрової обробки зображень;

– провести дослідження існуючих методів та алгоритмів попередньої обробки зображень;

– проаналізувати існуючі програмні системи створення, редагування та аналізу цифрових зображень та виділити основні сфери їх застосування;

– провести аналіз методів та алгоримтів контурного аналізу об’єктів на цифрових зображеннях;

– розробити алгоритми скелетизації об’єктів на цифрових зображень на основі вписаних квадратів;

– реалізувати програмну систему аналізу та опису об’єктів на цифрових зображень, провести її тестування та порівняти з програмами аналогами.

Об'єкт дослідження – процес аналізу кольорових зображень в системах цифрової обробки зображень.

Предмет дослідження – методи та алгоритми скелетизації об’єктів на кольорових зображеннях.

Наукова новизна одержаних результатів визначається наступним чином:

– проведено комплексний аналіз та класифікацію алгоритмів скелетизації об’єктів на зображеннях, що дозлолило виділити їх переваги та недоліки та розробити власний алгоритм скелетизації об’єктів на основі писаних квадратів;

– розроблено алгоритм скелетизації об’єктів на основі писаних квадратів, що дозволило зменшити обчислювальну складність процесу аналізу зображень та підвищити відсоток виділення скелетів об’єктів.

Практична цінність одержаних результатів полягає в тому, що:

– розроблено та проведено моделювання програмної системи цифрової обробки зображень з елементами опису об’єктів на зображенні, що дозволило в подальшому програмно реалізувати та провести дослідження запропонованих алгоритмів;

– реалізовано програмне забезпечення для скелетизації об’єктів на основі писаних квадратів для систем цифрової обробки зображень на основі мови програмування Java та бібліотеки функцій обробки зображень ImageJ.

Публікації та апробація до випускної кваліфікаційної роботи. За результатами наукових досліджень, проведених у випускній кваліфікаційні роботі, підготовлено тези доповіді «Алгоритм скелетизації об’єктів на основі писаних квадратів» обсягом 1 сторінка на І Науково-практичній конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі», а також «Класифікація методів розпізнавання обєктів на цифрових кольрових зображеннях» обсягом 1 сторінка на ІІ Науково-практичній конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі».

1 ПРОГРАМНІ, АПАРАТНІ ТА ГІБРИДНІ СИСТЕМИ ОБРОБКИ ЦИФРОВИХ ЗОБРАЖЕНЬ

1.1 Цифрова обробка зображень, основні функції та сфери застосування

Зображення є миттєвими знімками мінливого світу. При використанні одного зображення інтерпретація міститься в ньому інформації може виявитися скрутною без знання того, як ця інформація змінюється. Наприклад, якщо дивитися на послідовні знімки рухається кулі, комп'ютер може пов'язувати зміну кольору певних пікселів з рухається, ідентифікуючи тим самим куля. Однак він не може використовувати одне нерухоме зображення для безпосередньої ідентифікації кулі. У таких випадках сегментація зображень часто є першим кроком. У набір методів обробки зображень входить метод виділення інформації про об'єкти зображення на основі відомих статичних властивостей об'єктів.

Порогова обробка зображень (Image thresholding) є одним з таких методів сегментації зображень, який зменшує формат зображення до двійкового, тобто вся інформація на зображенні зводиться до двох категорій: переднього плану та фону. Проаналізувавши наведений вище приклад – знаючи, що це є куля, комп'ютер може зіставити будь-кластер червоних частинок зображення з «кулею» (переднім планом), а вся інша інформація – "не куля» (фон). Поріг зображення використовується в якості першого етапу в множини інших алгоритмів. Порогова обробка застосовується в багатьох додатках, починаючи від поліпшення читаності архівних манускриптів та візуалізації в медицині, до доповненої реальності. Перш, ніж вивчати власне метод, необхідно зрозуміти, що таке зображення. Зверніть увагу, що зображення являють собою 2D проекції нашого 3D світу з різними властивостями, які можуть бути представлені у вигляді такої функції:

$$f(x,\ y) = W(\rho, \phi, \alpha, V,\ M, L, A, \varepsilon),$$

де *f(x, y)* – отримане 2D зображення, що складається з пікселів з координатами *x* в рядку і *y* в стовпці;

W – геометрична функція в сферичних координатах ρ, ϕ, α (як приклад) об'єктів, які подаються з точки спостереження *V*;

M – матеріал;

L – умови освітлення;

A – атмосфера;

ε – похибки.

Найпростішим поданням *f(x, y)* може служити 2D масив значень пікселів в форматі двійкових чисел. Подання 8-розрядними числами називається напівтоновим (Gray-scale) зображенням (значення лежать в діапазоні від 0 до 255). Використовуючи для пікселя 4 набори 8-розрядних чисел, відповідних червоному, зеленому, синьому і сірому кольорам можна отримати 32-бітове кольорове зображення тощо.

В області обробки візуальної інформації виділяють три основних наукових напрямки (рисунок 1.1):

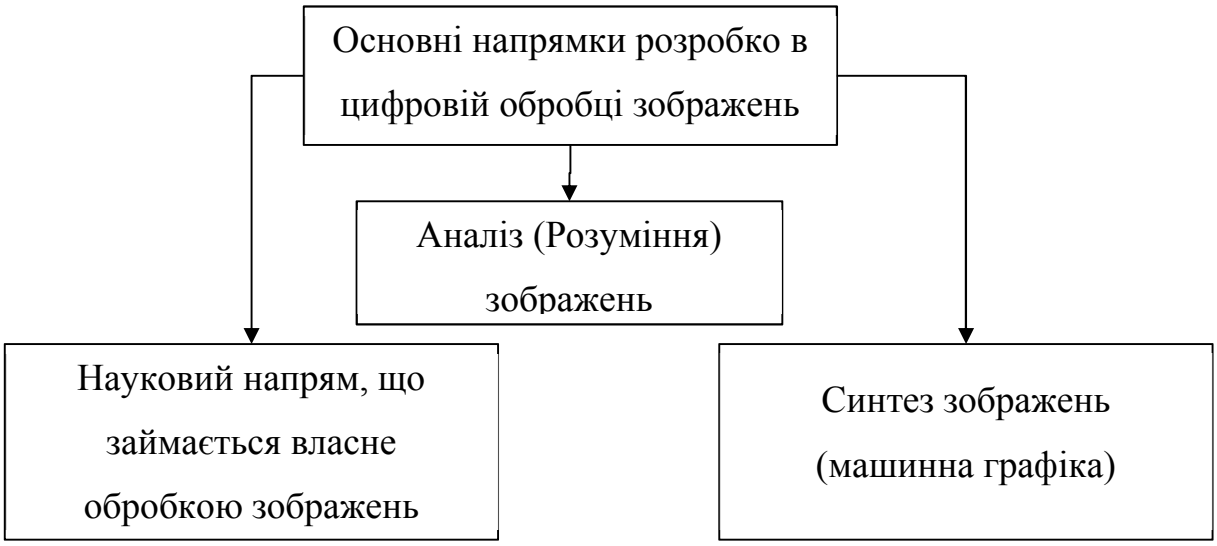


Рисунок 1.1 – Класифікація напрямків роботи методів в цифровій обробці зображень

Відповідно виділяють системи власне обробки зображень, системи аналізу (розуміння) зображень і системи синтезу зображень (машинної графіки). В першому випадку вирішуються завдання, в яких вхідні дані і результати обробки представляються в образотворчій формі. У другому випадку вхідні дані представляються в образотворчій формі, а результат обробки – в не картинній формі, наприклад у вигляді текстового опису сцени. В останньому випадку вирішуються завдання синтезу зображень в їх образотворчої формі за деяким їх опису або алгоритму побудови.

В якості основних операцій, що реалізуються в системах обробки візуальної інформації, можна виділити наступні (рисунок 1.2):

Рисунок 1.2 – Класифікація сучасних систем безпеки

В рамках вище наведених функцій системи обробки візуальної інформації на вхід пристрою введення зображень надходить безперервне зображення *f(x,y)*, що задає розподіл яскравості елементів деякої сцени по просторовим координатам *x, y*. У пристрої введення дане зображення потім піддається дискретизації, тобто формується деяка множина відліків функції *f(x,y)*. Далі отримані відліки піддаються квантованою. Таким чином, цифрове зображення являє двовимірним масивом квантованих відліків функції *f(x,y)*. Надалі елемент такого масиву називатимемо елементом цифрового зображення або пикселем, а саме цифрове зображення задавати функцією *f(i,j)* від дискретних аргументів *i* та *j* – індексів елементів цифрового зображення, що визначають їх положення в двовимірному масиві квантових відліків функції *f(x,y)*. Розглянуті вище функції зображень є вихідною формою представлення зображень. Результати обробки таких зображень можуть подаватися в різних формах:

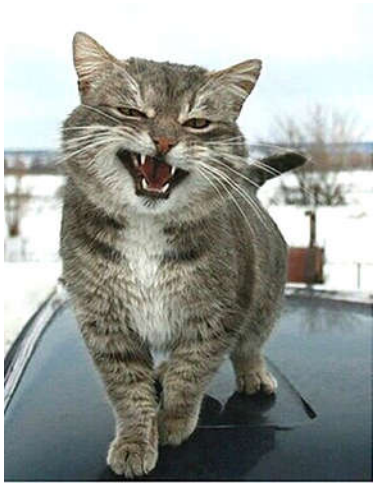
- списком сегментів зображення (для сегментованого зображення);
- геометричною моделлю у вигляді сукупності геометричних фігур;
- текстовим описом сцени тощо.

Найбільш складними за своєю реалізацією і виконуваних функцій є системи розуміння зображень. У найпростішому випадку такі системи являють собою системи розпізнавання образів. У такій системі в якості ознак, що використовуються для розпізнавання образів, можуть, наприклад, виступати параметри:

- текстури;
- контуру об'єктів;
- ознаки, сформовані з використанням перетворення Фур'є;
- різні моменти виділених сегментів зображення тощо.

Зауважимо, що в якості ознак можуть виступати також значення яркостей самих елементів зображення.

Традиційно цифрові зображення поділяють на растрові (рисунок 1.3,а) та векторні (рисунок 1.3,б).



а)

б)

Рисунок 1.3 – Приклади растрового (а) та векторного (б) зображень

Для того щоб краще розібратися в цих форматах, розглянемо два різні завдання. Перше завдання – отримати цифровий еквівалент звичайного фотографічного відбитка – це і буде растрове зображення. Друге завдання – намалювати креслення деякої деталі – векторне зображення. У першому випадку розумно розбити всю фотографію на дрібні впорядковані частинки (зазвичай зображення розбивають прямокутної сіткою), кожній такій клітинці поставити в відповідність деяке значення, що кодує колір. Масив таких значень називається растр, а зображення називається растровим. Кожен елемент зображення - осередок розбиває сітки, називається піксель (або піксель – від англ. pixel, скорочена словосполучення від англ. picture's element, елемент, що формує зображення зображення). Від кількості пікселів залежить детальність зображення. Максимальна деталізація растрового зображення задається при його створенні і не може бути збільшена. Якщо збільшується масштаб зображення, пікселі перетворюються в великі зерна. Для зображень в градаціях сірого яскравість пікселя частіше всього кодується значеннями від 0 (чорний) до 255 (білий). В разі кольорового зображення кількість колірних компонент може бути різному, однак для кожної колірної компоненти в межах пікселя визначається яскравість, так само як і для напівтонового зображення, тому колір кожного пікселя кодується вже набором значень. Наприклад, знімки, отримані з цифрової

камери, являють собою кольорові зображення, представлені трьома компонентами: червоного, зеленого і синього. Дана картинка має встановлений камерою розмір растра, вимірюваний в пікселях. Кожен елемент растра містить інформацію про яскравість кожної колірної компоненти.

До очевидних переваг растрової графіки можна віднести:

− простота і, а отже, як наслідок, це спрощена технічна реалізація автоматизації введення (оцифровки) образотворчої інформації. На сьогодні, існує розвинена система зовнішніх апаратних пристроїв для введення фотографій, слайдів, рисунків: сканери, відеокамери, цифрові фотоапарати;

− фотореалістичність. В результаті оцифрування отримуються реалістичні ефекти, які неможна було б створити штучно, наприклад туман або серпанок, домагатися найтонших нюансів кольору, створювати глибину і нерізкість, розмитість, акварельних і багато іншого.

Недоліки растрової графіки:

− великий розмір, займаний файлами;

− відсутність можливості для прямого визначення об'єкта, так як кожен об'єкт представляється множиною елементів (пікселів) зображення;

− спотворення, що виникають при виконанні будь-яких трансформацій (поворотів, масштабування, нахилів).

Векторна графіка значно відрізняється від растрової тим, що вона не є похідною від матеріального джерела, більш того, іноді її називають «геометричне моделювання» (рисунок 1.3,б). Суть векторної графіки зводиться до використання геометричних примітивів, таких як точки, лінії, сплайни і полігони, для представлення зображень. Інженер, який зображає деяку деталь, описує її в термінах побудови примітивів з різними параметрами і в результаті отримує не зображення само по собі, а деякий алгоритм побудови зображення з математично описаних складових. кожен примітив являє собою незалежний об'єкт, який можна переміщати, масштабувати і змінювати. Все це призводить до того, що векторне зображення можна редагувати, не побоюючись дій, що не входять в процес редагування.

У тих областях графіки, де принципове значення має збереження ясних і чітких контурів, наприклад в шрифтових композиціях, у створенні логотипів, векторна графіка незамінна. Вона використовує всі переваги роздільної здатності будь-якого вивідного пристрою (зображення завжди буде виглядати настільки якісно, наскільки здатне даний пристрій).

Переваги векторної графіки:

− векторні дані вимагають менше пам'яті;

− об'єкти векторної графіки легко трансформуються, і ними просто маніпулювати, що не має практично ніякого впливу на якість зображення;

− векторна графіка може включати в себе і елементи растрової графіки, фрагмент стає таким же об'єктом, як і всі інші (правда, зі значними обмеженнями в обробці);

− важливою перевагою програм векторної графіки є розвинені засоби інтеграції зображень і тексту, єдиний підхід до них і, як наслідок, можливість створення кінцевого продукту.

Програми векторної графіки незамінні в області дизайну, технічного малювання, креслярсько-графічних і оформлювальних робіт.

Недоліки векторної графіки:

− векторна графіка може здатися надмірно жорсткою, «Фанерною». Вона дійсно обмежена в мальовничих засобах. За допомогою векторної графіки практично неможливо створювати фотореалістичні зображення;

− векторний принцип опису зображення не дозволяє автоматизувати введення графічної інформації, як це робить сканер для растрової графіки.

Традиційно графічні програми чітко ділилися на що працюють з векторної і працюють з растровою графікою. Однак в даний час ця різниця стало менш помітно, оскільки практично всі дизайнерські програми використовують як векторну, так і растрову графіку. Крім того, на сьогоднішній день відомо кілька додатків, які дозволяють конвертувати растрові дані в векторні. На жаль, всі вони не досконалі і застосовуються у вузьких областях (розпізнавання тексту, розпізнавання схем, розпізнавання карт і деякі інші).

Для предствлення зображення в цифровій техніці використовуються різні типу зображень. Проаналізуємо їх основіні типи (рисунок 1.4):

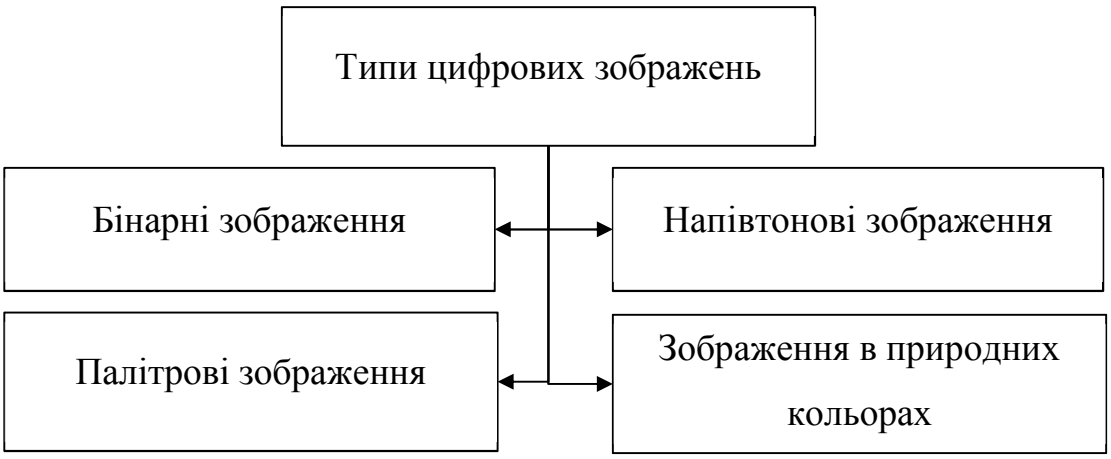


Рисунок 1.4 – Класифікація типів цифрових зображень

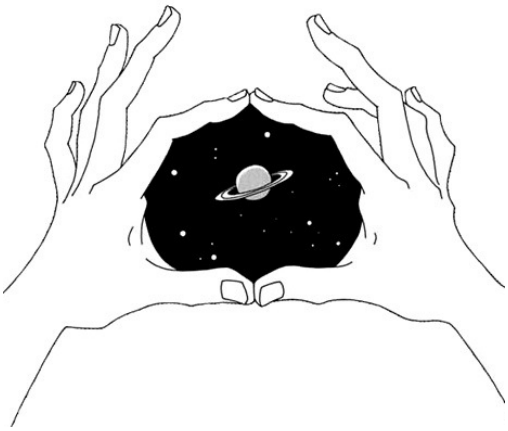
Бінарні зображення. Складаються тільки з чорних і білих пікселів. При їх запам'ятовуванні на кожен елемент досить одного біта. Такі зображення не дуже поширені, але використовуються на практиці для передачі простих зображень, наприклад при передачі факсів.

Напівтонові зображення. Характеризуються великою, на відміну від бінарних зображень, кількістю градацій яскравості. Один елемент зображення (піксель) займає в запам'ятовуючому пристрої 8 біт, тобто може приймати значення від 0 до 255. Це чисельне значення безпосередньо характеризує яскравість елемента зображення. Такі зображення поширені також не надто широко, проте основні дослідження в області обробки зображень проводяться саме на напівтонових зображеннях, оскільки вони представляють собою прості об'єкти з досить великим динамічним діапазоном. Існують спеціальні формати напівтонових зображень, які вимагають більшого динамічного діапазону. Наприклад, в медичній техніці зазвичай використовуються напівтонові зображення, в яких піксель кодується 12 або 14 бітами, в деяких випадках динамічний діапазон буває ще більше.

Палітрові зображення. Такий тип зображень нагадує напівтонові зображення, оскільки для зберігання одного пікселя також зазвичай використовується 8 біт. Істотна різниця полягає в тому, що на відміну від напівтонових зображень цими 8 бітами кодується НЕ яскравість пікселя, а його колір. Піксель може приймати значення від 0 до 255, і ця величина позначає індекс в масиві, що містить кольорову інформацію. Такий масив називається палітрою. Таким чином, кожному значенню пікселя ставиться у відповідність трійка чисел з палітри, що кодує колір. Зустрічаються також зображення, в яких кількість кольорів в палітрі може бути і менше (16, 32, 64, 128). При цьому для запам'ятовування одного елемента зображення потрібно 4, 5, 6 або 7 біт відповідно. Можливості обробки палітрових зображень сильно обмежені. Просту обробку, що включає зміну яскравості або контрасту, як правило, можна виробляти без труднощів, оскільки для цього потрібно змінювати лише колірну палітру. Для більш складних алгоритмів обробки потрібно попередньо перевести зображення в 24-бітовий формат природних кольорів. При багатьох операціях обробки зображень виникають кольору, яких не було в оригінальному документі. Їх необхідно погоджувати з наявної палітрою кольорів, що, з одного боку, вимагає великих витрат часу, а з іншого – виключає застосування багатьох операцій обробки зображень. Палітрові зображення застосовувалися в системах з обмеженою кількістю відеопам'яті, в даний час їх актуальність значно зменшилася.

Зображення в природних кольорах. Кожен елемент запам'ятовується у вигляді RGB-трійки. Оскільки червона, зелена і синя складові кольору елемента зображення задаються чисельним значенням від 0 до 255, для запам'ятовування кожного елемента зображення потрібно 24 біта. Таке зображення теоретично може містити до 16,8 мільйона різних кольорів. Цим пояснюється назва «зображення в природних кольорах». Зображення в природних кольорах забезпечують найширші можливості для подальшої обробки та художнього втілення.

Приклади різних типув зображень наведено на рисунку 1.5.



бінарне

напів тонове

Палітрове зображення

Зображення в природних кольорах

Рисунок 1.5 – Приклади цифрових зображень ріних типів

Цифрова обробка зображень в даний час широко використовується в системах телекомунікацій, радіо і гідролокації, сейсмології, робототехніці, радіоастрономії, медицині. Перехід на цифрове телерадіомовлення, зокрема, набуло особливого значення в сучасному світі, в якому ставляться підвищені вимоги до якості отриманої інформації. Технічною основою створюваної в світі інфокомунікаційного середовища є сучасні системи обробки і передачі мультимедійної інформації, їх впровадження вирішує проблеми:

− створення високоякісних систем інтерактивного цифрового телевізійного мовлення;

− забезпечення діяльності органів державної влади та створення інтерактивних систем опитування громадської думки;

− створення мобільного відеоконференцзв'язку між центральними установами та віддаленими центрами і районами і реалізації оперативного спостереження за об'єктами і дистанційного керування;

− оптимізації лікувальної роботи та створення систем мобільного телемедицини;

− створення систем дистанційного навчання на базі провідних вузів тощо.

1.2 Підвищення якості цифрових зображень на основі фільтрації

В останнє десятиліття розроблені ряд методів та технологій автоматичного розпізнавання графічних образів. Графічні образи цифрових документів зазвичай формуються за допомогою їх сканування. Однак 100% правильне розпізнавання відсканованих сцен не забезпечує жодна з відомих в даний час систем автоматичного розпізнавання. Це викликано, в першу чергу, такими причинами:

− написання однієї і тієї ж сцени, навіть однією і тією ж людиною може мати істотні відмінності (можливі розриви ліній, товщина ліній може суттєво відрізнятися від еталонної, можуть відрізнятися колір об’єктів, розміри та тип шрифту тощо);

− об’єкти, написані різними людьми, можуть відрізнятися ще більш істотно.

Це робить актуальним пошук методів поліпшення якості розпізнавання зображнь до того, як відбувається власне процес розпізнавання. У системах

автоматичного розпізнавання зображень перед власне розпізнаванням проводиться їх скелетизації (стоншення ліній до ширини в один піксель). Це накладає на зображення деякі вимоги, головне з яких – зображення повинно бути чорно-білим без градацій сірого. Тому в система розпізгавання зображень проводиться етап попередньої обробки зображення, а саме бінаризація. Результатом такого перетворення без попередньої фільтрації зображення, в загальному випадку, буде зашумлене зображення з деяким спотворенням об’єктів. З цієї причини спочатку буде доречно розглянути просторові методи поліпшення зображення (фільтрації зображення), спрямовані на те, щоб зробити відскановане зображення максимально «придатним» для скелетизації.

Особливості відсканованих цифрових фотокопів зображень. Необхідно підкреслити, що методи обробки різних відсканованих зображень можуть відрізнятися в силу різних налаштувань сканера. Зображення може бути темним або світлим, чітким або розмитим, високо чи низько контрастним, заповненим лише об’єктами чи містити області, що не мають інформаційної цінності, плямами або точками («шум»). Саме з цієї причини з одного боку, перед початком аналізу необхідно провести фільтрацію зображення (тобто видалити з зображення все те, що заважатиме розпізнаванню об'єктів). З іншого боку, досить складно однозначно назвати метод фільтрації, який був би ідеальним для застосування до будь-яких зображень. Можна лише говорити про те, наскільки використання того чи іншого методу не «зіпсує всю картину», якщо вжити його за базовий.

Згладжувальні просторові фільтри Згладжувальні фільтри [1,2] застосовуються для розфокусовки зображення та придушення шуму. Розфокусовка може застосовуватися як підготовчий етап обробки зображення, наприклад, для видалення дрібних деталей перед виявленням великих об'єктів, або ж для усунення розривів в лініях або деталях. Для придушення шумів може використовуватися розфокусовка із застосуванням як лінійної, так і нелінійної фільтрації. Вихід (відгук) найпростішого лінійного згладжувального просторового фільтра є середнє значення елементів по околиці, покритої маскою

фільтра. Такі фільтри іноді називають усередняючими фільтрами або згладжуючими фільтрами. Ідея застосування згладжуючих фільтрів досить очевидна. Зменшення «різких» переходів рівнів яскравості досягається заміною вихідних значень елементів зображення на середні значення по масці фільтра. Оскільки випадковий шум якраз характеризується різкими стрибками яскравості, найбільш очевидним застосуванням згладжування є придушення шуму. Для вирішення спільних завдань негативною стороною застосування згладжуючих фільтрів є розфокусовка контурів різних об'єктів (які, також, характеризуються різкими перепадами яскравості), однак для завдання скелетизації це тільки плюс, тому чим більше «гладким» буде контур об’єкту, тим більш рівним вийде його скелет.

Найпростішим згладжуючим фільтром є фільтр, який дає середнє значення яскравості пікселів в околиці 3x3, що досягається підсумовуванням значень всіх елементів в цій околиці, і діленням результату на 9. Такий просторовий фільтр іноді називають однорідним усереднюються фільтром обробки зображень (рисунок 1.6).

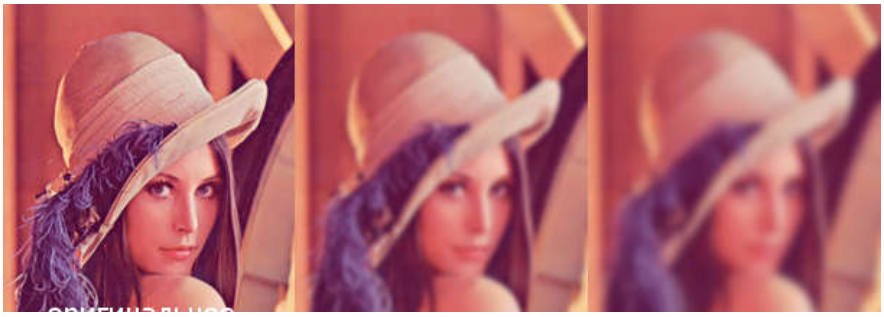


Рисунок 1.6 – Приклади роботи згладжуючого фільтра

Наступний фільтр є удосконаленням попереднього і результат його застосування – так зване зважене середнє. Цей термін застосовується, щоб показати, що значення елементів множаться на різні коефіцієнти, що дозволяє привласнити їм як би різні ваги в порівнянні з іншими. Основна стратегія присвоєння центральному пікселю найбільшої ваги, а іншим – обернено

пропорційно їх віддалі, має на меті зменшення расфокусировки при згладжуванні. Приклад маски такого фільтру наведено нище (рисунок 1.7):

1	2	1
2	4	2
1	2	1

Рисунок 1.7 – Приклади маски згладжуючого фільтра

У наведеній масці показані коефіцієнти, які використовуються для обчислення в такому фільтрі. Можна було б вибрати і інші значення коефіцієнтів, але сума показаних в цій масці дорівнює 16, що зручно при комп'ютерній реалізації, оскільки це ступінь двійки.

На практиці результати згладжування першим або другим фільтром виглядають досить схожими, однак застосування маски з коефіцієнтами дає кращі результати при подальшої скелетизації. Також слід зазначити, що маска фільтра може бути не тільки 3x3. Якщо застосувати фільтр по масці 5x5, то чіткість об’єктів падає настільки (зливаються контурні лінії), що скелетизації об’єктів стає проблематичною.

Фільтри, засновані на порядкових статистиках. Фільтри, засновані на порядкових статистиках, відносяться до класу нелінійних просторових фільтрів. Результат такого фільтру визначаться попередніми упорядкуванням (ранжуванням) значень пікселів, покриваються маскою фільтра, і подальшим вибором значення, що знаходиться на взначеній позиції впорядкованої послідовності (тобто має визначений ранг). Власне фільтрація зводиться до заміщення вихідного значення пікселя (в центрі маски) на отримане значення відгуку фільтра. Найбільш відомий медіанний фільтр, який, як випливає з назви, замінює значення пікселя на значення медіани розподілу яркостей всіх пікселів в околиці (включаючи і вихідний). Медіанні фільтри вельми популярні тому, що для визначення типів випадкових шумів вони демонструють відмінні можливості

придушення шуму при значно меншому ефекті розфокусовки, ніж у лінійних згладжувальних фільтрах з аналогічними розмірами.

Медіана набору чисел є таке число, що половина чисел з набору менше або дорівнюють йому, а інша половина – більша або рівна. Щоб виконати медіанну фільтрацію для елемента зображення, необхідно спочатку впорядкувати по зростанню значення пікселів всередині околу, потім знайти значення медіани, і, нарешті, привласнити отримане значення оброблюваному елементу (для околу 3x3 елементів значення медіани буде п'ятим значення за величиною, для околу 5x5 – тринадцятим, і так далі).

В контексті даної задачі поліпшення зображення перед скелетизації медіанний фільтр може бути застосований як до 256-колірного зображення, так і (після переведення) до чорно-білого. У другому випадку відбувається найчастіше корисне згладжування країв об’єктів (хоча на зображеннях з поганою якістю ця процедура призводить до злиття воєдино сусідніх об’єктів).

Також слід зазначити, що на практиці використовується не тільки медіанний фільтр, а також фільтр максимуму і фільтр мінімуму, відгуками яких є останній і перший елементи відсортованого масиву відповідно. Але такі фільтри можуть видаляти і дрібні деталі об’єктів, що згодом призведе до помилок аналізу.

Просторові фільтри підвищення різкості (рисунок 1.8).

Рисунок 1.8 – Приклади підвищення різкості рображення

Розглянуті вище фільтри орієнтовані на зниження різкості об’єктів. Незважаючи на той факт, що спочатку здається нерозумним використання

одночасно згладжуючих фільтрів і фільтрів підвищення різкості, практика показує, що така комбінація заслуговує на увагу. Головна мета використання фільтрів підвищення різкості полягає в тому, щоб підкреслити дрібні деталі зображення (кунтурні лінії об’єктів). Але використання цих фільтрів самостійно призводить до викривлення об’єктів і розривів в лініях. Розглянутий фільтр заснований на використанні других похідних – лапласіан. Це обумовлено тим, що підвищення різкості досягається просторовим диференціюванням. Величина відгуку оператора похідної в точці зображення пропорційна ступеню розривності зображення в даній точці. Таким чином, диференціювання зображення дозволяє посилити перепади і не підкреслювати області з повільними змінами рівнів яскравості. Даний підхід однозначно дозволяє покращити процес обробки зображень, оскільки, при використанні даного фільтру підвищення різкості зображення, на наступних етапах відбувається оббробка більш якіснішого вхідного зображення, що в свою чергу зменшує складність процесу аналізу. Практичною реалізацією Лапласіан (Оператора Лапласа) є віднімання з вихідного зображення, зображення відфільтрованого за такою маскою (рисунок 1.9):

0	1	0
1	-4	1
0	1	0

Рисунок 1.9 – Приклади маски фільтра підвищення різкості

Інший підхід наного фільтру базується на з'єднанні рузультатів операції віднімання з операцією фільтрації, за допомогою єдиної маски, що поєдную в собі ці два принципи (рисунок 1.10):

0	-1	0
-1	5	-1

0	-1	0

Рисунок 1.10 – Приклади маски фільтра підвищення різкості

Також, іноді використовується маска такого типу (рисунок 1.11):

-1	-1	-1
-1	9	-1
-1	-1	-1

Рисунок 1.11 – Приклади маски фільтра підвищення різкості

Застосування маски є більш потужним засобом підвищення різкості, але, саме з цього, ця маска є неприйнятною для збереження символів в початковому вигляді. Також маска сильно збільшує шум, який стає бачимо після перекладу зображення в два кольори.

Перекодування зображення в градаціях сірого в бінарне. Припустимо, що деяке зображення, заданий функцією *f(x, y)*, де *f(x,y)* – значення кольору в точці *(x,y)*, містить темні об'єкти на світлому фоні. Очевидний спосіб виділення об'єктів з навколишнього фону складається в обчисленні порогового значення *T*, такого що будь-яка точка *(x,y)* в якій $f(x,y)<T$ буде вважатися точкою об'єкта, а в іншому випадку – точкою фону. Таким чином, отримання бінарного зображення досягається шляхом поелементного сканування вихідного зображення і присвоювання кожному пікселю нового кольору (білого або чорного). Але найчастіше трапляється, що у зображення, отриманого в результаті сканування, одна частина затемнена, інша ж – світліша. Для вирішення цієї проблеми може бути запропонований наступний підхід:

- всі зображення розбивається на квадрати;
- для кожного квадрата окремо обчислюється порогове значення.

Практичні дослідження показують, що універсальним можна вважати розбиття зображення на 32 (2^5) квадрата по вертикалі та горизонталі. Хоча, природно, це число нічого не означає, і можна експериментувати і з іншими.

Граничне значення обчислюється підсумовуванням перемноження кожної градації сірого кольору (0..255) на кількість її «представників» (пікселів) в квадраті та діленням результату на загальну кількість пікселів (нагадаємо, що колір складається з трьох складових (RGB), але у зображень в градаціях сірого всі три складові рівні, тому, для визначення кольору пікселя можна брати будь-яку):

$$B = \frac{\sum_i c_i i}{\sum_i c_i},$$

де B – шукане порогове значення;

c_i – кількість пікселів i-того кольору.

У дослідженнях, метою яких ставилося зменшення шуму на зображенні, була знайдена така корисна операція: після обчислення граничного значення сильно його занизити (наприклад, на 50) і подивитися, чи залишиться після присвоєння кожного пікселя чорного або білого кольору хоч один чорний піксель в квадраті. Якщо залишиться, це буде означати, що в цьому квадраті з імовірністю 95% знаходиться об'єкт (тому що пікселі об'єкт – самі темні на зображенні), якщо не залишиться, то можна говорити про те, що в даному квадраті об'єкту немає та його необхідно виключити зподальшого аналізу. Метод перекладу зображення в градаціях сірого в бінарне зображення, а також методи фільтрації зображення були реалізовані для підготовки зображення до скелетизації, використовуваної в системі автоматичного розпізнавання текстів - Cunning Eye‖.

1.3 Програмні засоби обробки та опису цифрових зображень

Графічні редактори - це окрема група прикладних програмних систем, щовикористовується для створення та обробки графічних зображень. Графічні редактори можна розділяти за різними характеристиками, серед яких основним є тип графіки, з якою буде працювати програма. Програмні системи, що надають користувачеві в пдсумку створювати ті чи інші зображення. На сьогоднішньому ринку представлено безліч таких програмних систем. Розробника створюються як спеціалізовані системи, що призначені для обробки наперед визначеного типу графіки, так і багатофункціональні програмні системи, забезпечують користувача іункціоналом для роботи з різнии типами комп'ютерної графіки, або кі дозволяють поєднувати різноманітні графічні об'єкти в одне ціле.Окрім того, графічні програмні системи поділяються за системними платформами, відносно яких вони і були створені. Для прикладу, основна група поліграфічних систем проектувалась та була реалізована для платформи OS MAC, що є спеціалізованою цієї мети, в то й же час професійні 3D редактори або програми рідноманітного моделювання, як правило, проектуються для роботи під Windows; спеціалізовані програмні пакети для професійної обробки відео файлів розраховані для роботи на спеціалізованому обланенні для отримання максимальних результатів типу студій АВМ (аудіо-відео монтажу), даний перелік можна продовжувати для кожної із спеціалізованих груп програмних систем. Відзначемо, що незважаючи на здійснювані таку традиційну спеціалізацію, останнім часом отримали поштовх в розвитку, так звані «крос-платформні» системи. Суть їх полягає в тому, що оброблювальні об'єкти (в основному форматі файлів різних типів) можна імпортувати з однієї платформи на іншу без втрати якості обробки. Максимальний ефект від такого отримали користувачі, що працюють з поліграфією. На сучасному етапі різниця між комп'ютерами систем MAC та PC фактично зведена майже до нуля. Проаналізуємо класичні види графічних програмних систем і редакторів для платформи PC, оскільки дана архітектура є найбільш загальновідомою та поширеною серед користувачів:

- системи для обробки растрової графіки;

- векторні графічні програмні редактори;
- програмні пакети для верстки (настільні видавничі системи);
- пакети для роботи з 2D-анімацією;
- програми для створення та редагування Web-сторінок;
- 3D редактори;
- пакети для професійного інженерного моделювання або проектування;
- інші комп’ютерні програми для роботи з цифровою графікою (спеціалізовані 3D-додатки, програмні додатки створення об'ємних шрифтів, системи для монтажу та обробки відео, програми для візуалізації наукової досліджень тощо).

Графічні редактори поділяються на растрові, векторні і редактори тривимірної графіки.

Растрові програмні пакети для редагування зображень широко використовуються для обробки растрових цифрових зображень, наприклад для ретуші, створення художніх композицій або додавання фотоефектів. У даному великому класі програм для обробки растрової графіки почесне місце займає програмна система роботи з растровою графікою Photoshop компанії Adobe.

Група векторних редакторів використовується в основному для створення зображень, однак вони практично не застосовуються при обробці вже готових зображень. Проте, їх широко використовують в рекламному секторі, в поліграфії для оформлення обкладинок, а також всюди, де необхідний стиль художньої роботи повинен бути близький до креслярського. Серед програмних систем створення та обробки векторної графіки слід відмітити графічні редактори Adobe Illustrator, CorelDraw.

Редактори тривимірної графіки, це окрема група програмних пакетів, що мають дві характеристики, що виділяють їх на фоні інших:

- по-перше, вони надають можливість гнучко керувати взаємодією властивостей різних поверхонь об'єктів, що зображені на зображенні, з властивостями джерел освітлення;

− по-друге, забезпечують користувача молжливістю створювати тривимірну анімацію (тому їх нерідко називають 3D-аніматорами).

Серед основних програмних систем, що прадставлені на світовому ринку у секторі обробки тривимірної графіки є програмні продукти 3D Studio Max, Softimage-3D, програма Maya.

Сервіс Adobe Photoshop Express призначений для редагування зображень в режімі онлайн і створення власних альбомів. Для початку роботи з сервісом пройдіть реєстрацію на сайті. Після цього ви зможете створювати власні галереї та редагувати зображення.

У вікні редактора можна вибрати функції для роботи з завантаженим зображенням, використовуючи запропоновані інструменти. Використувуючи програму Decorate можете накладати до зображення написи, рамки і графічні об'єкти. Світлини, створені в сервісі Photoshop Express, можна зберігати або одразу роздруковувати або переслати іншим користувачам, використовуючи відповідні інструменти на нижній панелі вікна.

Онлайн редактор зображень Sumopaint. Сервіс www.sumopaint.com використовується для обробки зображень в режимі онлайн і створення коритсувацьких альбомів. У вікні редактора можна завантажити зображення зі робочого комп'ютера або альбому в сервісі Sumo Paint або знайти в глобальній мережі, обравши посилання на файл. Потім запускається процес редагування зображення, при цьому програма надає користувчеві різні інструменти, розташовані на лівій панелі редактора.

Графічний онлайн редактор фотографій Splashup. Графічний редактор www splashup com призначений для редагування фотографій в режимі онлайн і створення власних альбомів. У головному вікні редактора користувачу надається можливість завантажити зображення зі комп'ютера, з персонального альбому в одному з сервісів або завантажити з Інтернету, ввівши посилання. Потім користувач можете редагувати зображення, використовуючи запропоновані інструменти, розташовані на панелі редактора.

Online редактор растрових зображень Aviary. Сервіс aviary призначений для редагування растрових і векторних зображень, редагування звукових файлів у режимі онлайн і створення власних альбомів.

1.4 Постановка задачі

В даному розділі проаналізовано задачі, що ставляться перед розробниками систем обробки ображень, що використовують елементи аналізу та опису зображень. Проведено дослідження та класифікацію існуючих підходів до підвищення якості вхідного зображення на основі фільтрації, а також виділено їх основні переваги та недоліки. Проаналізовано існуючі програмні системи створення, редагування та аналізу цифрових зображень та визначено основні сфери їх застосування.

Для досягнення поставленої мети необхідно розв’язати наступні задачі.

- провести комплексний аналіз задач цифрової обробки зображень;
- провести дослідження існуючих методів та алгоритмів попередньої обробки зображень;
- проаналізувати існуючі програмні системи створення, редагування та аналізу цифрових зображень та виділити основні сфери їх застосування;
- провести аналіз методів та алгоритмів контурного аналізу об’єктів на цифрових зображеннях;
- розробити алгоритми скелетизації об’єктів на цифрових зображень на основі вписаних квадратів;
- реалізувати програмну систему аналізу та опису об’єктів на цифрових зображень, провести її тестування та порівняти з програмами аналогами.

2 МЕТОДИ ТА АЛГОРИТМИ ОБРОБКИ ТА ОПИСУ ОБ’ЄКТІВ НА ЦИФРОВИХ ЗОБРАЖЕННЯХ

2.1 Методи виділення та кодування контурів об’єктів

Аналіз зображення – це процес виділення потрібної інформації з зображення за допомогою автоматичних систем. Системи аналізу не обмежуються поділом областей сцени на фіксоване число класів. Вони призначені для опису складних сцен, різноманітність яких може бути настільки великим, що їх не можна описати за допомогою заздалегідь заданих термінів. В системі аналізу також можуть використовуватися методи штучного інтелекту для управління різними блоками системи і організації ефективного доступу до бази апріорних відомостей про об'єктах.

Ознака зображення – це його найпростіша характеристика або властивість. Деякі ознаки є природними в тому розумінні, що вони встановлюються візуальним аналізом зображення, тоді як інші, так звані штучні ознаки, виходять в результаті його спеціальної обробки і вимірювань. Природні ознаки:

- світлота (яскравість);
- текстура різних областей зображення;
- форма контурів об'єктів [1].

Зазвичай аналіз зображення включає в себе етапи отримання та опису зовнішнього контуру об'єктів, що зображені, а також запис в структуру координат точок, що формують даний контуру. На практиці, як правило, необхідно отримати зовнішній контур об’єктів у вигляді замкнутої кривої лінії або множини взаємопоєднаший відрізків дуг. Виділяють три основні групи підходів до поданням кордонів об'єкта:

- апроксимація за допомогою кривих;
- простежування точок, що формують контури;
- зв'язування множини точок перепадів.

Контури зображень є областями з високою концентрацією інформації, яка слабо залежить від кольору і яскравості. При розгляді будь-якого об'єкта в свідомості людини формується зоровий образ. При сприйнятті очима, свідомість відстежує лінію контуру, що призводить до створення в свідомості відповідного образу з характерними деталями.

Існує думка, що при сприйнятті в свідомості людини будуть сформовані два образи об’єкту: контур та внутрішньої частини об’єкту на зображені. Необхідним кроком сприйняття вважається сканування по лінії контуру. Наведені цифри показують, що роль контурів при розпізнаванні та обробці зображень є надзвичайно важливою [2].

Контурний аналіз є сукупністю методів виділення, опису та відповідного перетворення контурів об’єктів на зображенні та розпізнавання відповідних зорових образів. Контур в повній мірі описує форму об’єктів на зображенні, а також зберігає всю важливу інформацію для процесу розпізнавання об’єктів на зображені на основі їх форми. Описаний підхід, не потребуєпроводити аналіз та обробку внутрішніх точок зображення, що дозволяє, відповідно, значно скоротити обсяг інформації, що переробляється. Наслідком цього часто стає можливість забезпечення роботи системи в режимі реального часу. Під контуром розуміється просторово-протяжний розрив, перепад або стрибкоподібне зміна значень яскравості. Існує ряд проблем при виділенні контурів зображення:

- розриви контуру в місцях, де яскравість змінюється не дуже швидко;
- наявність помилкових контурів внаслідок шуму на зображенні;
- широкі контурні лінії через розмитість або шуму.

Контурний аналіз надає користувачеві можливість описувати, порівнювати, зберігати і проводити пошук об'єктів на зображеннях, що були представлені у форматі своїх зовнішніх обрисів - контурів, а також ефективно боротись з основними проблеми розпізнавання образів, серед яких:

- перенесення;
- поворот області;
- зміна масштабу зображення об'єкта [3].

У теорії контурного аналізу контур, як правило, кодується за допомогою послідовності, що складається з множини комплексних чисел. На іділеному контурі фіксується деяка точка, яка будемо умовно називати початковою. Після чого, проводиться процес проходження контуру (наприклад, у напрямку руху годинникової стрілки), і кожен з векторів зміщення буде записуватись за допомогою комплексного числа у форматі $a+ib$, де a – зміщення точки по осі x; b – зміщення по осі ординат. Зсув відраховується від значення попередньої точки контуру (рисунок 2.1).

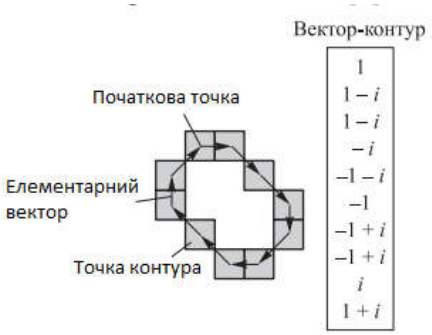


Рисунок 2.1 – Кодування контуру

Розглянемо різні методи контурного аналізу (рисунок 2.2).

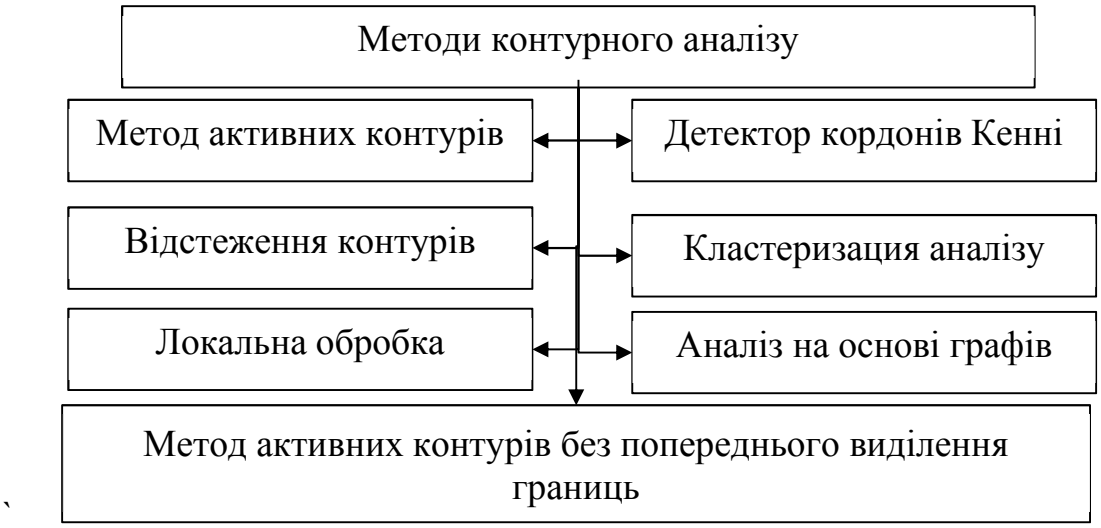


Рисунок 2.2 – Класифікація методів контурного аналізу

Метод активних контурів. Активні контури широко застосовуються в задачах виділення контурів, меж і сегментації зображень. Для виявлення контурів на зображенні тут використовують криві мінімальної енергії, або змійки. Алгоритм наступний: спочатку контур ініціалізується як проста лінія, а потім він деформується для створення області об'єкта. Точки в контурі прагнуть до кордону об'єкта при мінімізації енергії контуру.

Для кожної точки vi енергія обчислюється за формулою:

$$E_i = \alpha E_{\text{int}}(v_i) + \beta E_{ext}(v_i),$$

де α, β – константи, що забезпечують відносну корекцію енергії;

$E_{\text{int}}(v_i)$ – функція енергії, що залежить від форми контуру;

$E_{ext}(v_i)$ – функція енергії, що залежить від властивостей зображення і типу градієнта в околиці точки v_i.

Величини, $E_i, E_{\text{int}}(v_i), E_{ext}(v_i)$ є квадратними матрицями. Значення в центрі кожної матриці відповідає енергії контуру в позначці v_i.

Кожна вершина v_i потенційно може перейти в будь-яку точку v_i‘, що відповідетеме мінімальній енергії E_i.

Недоліки методу:

− якщо об'єкт не має чітких меж або площа неоднорідна і містить плавні градієнти, то алгоритм не вирішить завдання сегментації коректно, що призведе до неможливості подальшого автоматизованого аналізу;

− нормаль вектора дотичної у точки може сильно змінюватися в напрямку, що може спричинити злиття точок. Від цього контур може вийти грубим і сильно відрізнятися від кордонів виділяється об'єкта.

Метод активних контурів без попереднього виділення кордонів. На відміну від традиційного способу активних контурів цей метод не вимагає попереднього виділення кордонів об'єкта зображення, а вихідне зображення не обов'язково згладжувати. Крива, або змійка (замкнутої округлої форми), рухається з

довільної точки зображення. При перетині кордону вона починає деформуватися і приймати форму об'єкта на зображенні, як б заповнюючи внутрішню його частину.

Детектор кордонів Кенні. Дж. Кенні вивчив математичну задату отримання фільтра, що буде оптимальним за такими критеріями як:

- виділення,
- локалізації та мінімізації декількох відгуків одного краю.

Все це означає, що розроблений детектор повинен реагувати на межі, але при цьому всьому ігнорувати помилкові, точно визначати лінію кордону і реагувати на кожний кордон один раз, що дозволяє уникнути скупчення широких смуг зміни яскравості як набір кордонів.

Алгоритм включає в себе:

- згладжування – розмиття цифрового зображення для видалення шуму;
- пошук градієнтів - границі відзначаються там, де градієнт на цифровому зображенні приймає максимальне значення;
- придушення немаксимумів – до уваги беруться тільки локальні максимуми, які маркуються як границі;
- подвійну порогову фільтрацію – потенційні кордони ідентифікуються як пороги;
- трасування області неоднозначності – остаточні контури встановлюються шляхом придушення усіх границь, що є не пов'язані з відповідними (сильними) границі.

Для зменшення чутливості алгоритму до шуму застосовується перша похідна від Гауссіана. Після застосування фільтра, зображення ставати злегка розмитим.

Відстеження контурів. Метод полягає в послідовному кресленні кордону між об'єктом і фоном. Простежує точка у вигляді «жука» повзає по зображенню до тих пір, поки не доходить до темної області (об'єкт). Тоді «жук» повертається ліворуч і рухається по кривій, поки не досягне меж об'єкта, після цього

повертається направо і повторює процес, поки не досягне околиці початкової точки (рисунок 2.3).

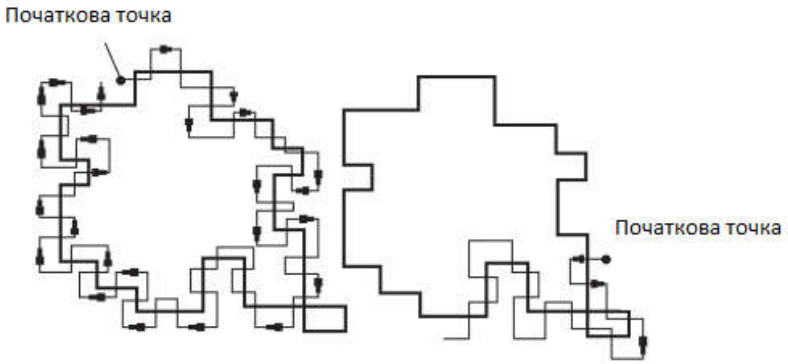


Рисунок 2.3 – Метод проходження контуром

Кластеризация. По відношенню до швидкості та відстані використовують кластеризацию найближчого сусіда. Позначимо дві лінії як $\{p_1,\ldots,p_m\} \in S_1$ та $\{q_1,\ldots,q_m\} \in S_2$ за умови, що вони задовольняють наступним нерівностям:

$$\{\left|x_{p_i} - x_{q_i}\right| + \left|y_{p_i} - y_{q_i}\right|\} \le \alpha_d;$$
$$\{\left|u_{p_i} - u_{q_i}\right|\} \le \alpha_u;$$
$$\{\left|v_{p_i} - v_{q_i}\right|\} \le \alpha_v,$$

де $\alpha_d, \alpha_u, \alpha_v$ – порогові константи;

u_p, v_p – складові швидкості точки p в координатах (x_p,y_p);

Кластеризація найближчого сусіда – найбільш ефективний метод для сцен з перешкодами. Перешкоди обробляються на етапі спостереження.

Локальна обробка. Методи виявлення кордонів виділяють в зображенні тільки пікселі, що лежать на контурі. На практиці це безліч пікселів рідко відображає контур досить точно через шумів, розривів контурів через неоднорідність освітлення тощо. Тому алгоритми виявлення контурів зазвичай

доповнюються процедурами зв'язування, щоб сформувати множини контурних точок.

Один із способів зв'язування точок контуру полягає в аналізі характеристик пікселів в невеликій околиці кожної точки зображення, яка була відзначена як контурна. Всі точки, які є подібними відповідно за деякими критеріями, повинні зв'язуються та утворюють однорідний контур, що складається з відповідають цим критеріям пікселів. При цьому використовуються два основних параметри для встановлення подібності пікселів контуру:

- відповідь оператора градієнта, що визначає значення пікселя контуру;
- напрямок вектора градієнта.

Піксель контуру *(x_0,y_0)*, розташований всередині заданої околиці точки *(x,y)*, вважається схожим з пикселом *(x,y)* по модулю градієнта, якщо

$$\nabla f(x,y) - \nabla f(x_0,y_0) \le E,$$

де E – заданий невід'ємних поріг.

По напрямку градієнта, якщо:

$$\alpha(x,y) - \alpha(x_0,y_0) \le A,$$

де $\alpha(x,y) = arctg\dfrac{\partial x}{\partial y}$;

A – заданий невід'ємних кутовий поріг.

Піксель в заданій околиці об'єднується з центральним пикселом *(x,y)*, якщо виконані критерії подібності і за значенням,і за напрямком. Цей процес повторюється в кожній точці зображення з одночасним запам'ятовуванням знайдених пов'язаних пікселів при русі центру області.

Простий спосіб обліку даних полягає в тому, що кожну множоину пов'язують пікселів контуру присвоюється своє значення яскравості.

Аналіз за допомогою графів. Підхід до виявлення і зв'язування контурів на основі подання у вигляді графа і пошуку на цьому графі шляхів з найменшою вартістю, які відповідають значущим контурам, дозволяє побудувати метод, добре працює в присутності шуму. Така процедура виявляється досить складною і вимагає великого часу обробки.

Елемент контуру – межа між двома пікселями p і q, які є сусідами.

Елементи контура ідентифікуються координатами точок p та q. Приклад отримання контуру на основі графів наведено на рисунку 2.4

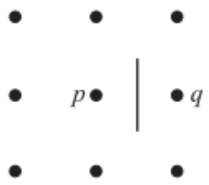


Рисунок 2.4 – Елемент контуру, що знаходиться між точками p та q

Контур - послідовність з'єднаних між собою елементів. Кожному елементу контуру, заданому пікселями p та q, відповідає якась вага:

$$c(p,q)=H-[f(p)-f(q)],$$

де H – максимальний рівень яскравості в зображенні;

$f(p), f(q)$ – яскравості пікселів p і q відповідно.

Завдання відшукання на графі шляху мінімальної вартості є нетривіальною по обчислювальної складності, тому доводиться жертвувати оптимальністю на користь швидкості обчислень.

Складність реалізації і велика ресурсомісткість – ось основні недоліки такого аналізу, гідністю якого є слабка чутливість до шумів.

Виділення контурів як підготовчий етап в процесі вилучення ознак зображення. Такі алгоритми залишають на зображенні контури. З цієї причини вихідний зображення може сильно відрізнятися від початкового. Хоча виділення контурів в основному використовується в машинному зорі, воно, звичайно, має і

інші застосування. наприклад, інформація про кордон об'єктів, отримана в процесі його посилення, може бути використана в оригінальному документі для збільшення його різкості.

Нижче будуть розглянуті трьома різними способами виділення контурів.

Посилення контурів по Лапласу. Метод посилення контурів по Лапласа відрізняється від подібних підходів тим, що даний метод не є залежним від напрямку, тобто границі підкреслюються незалежно від їх спрямування. Це перетворення виділяє кордону сильніше, ніж інші методи. Виділення контурів по Лапласу знаходить застосування в багатьох системах обробки зображень.

Виділення контурів методом зсуву та різниці. Даний алгоритм виділяє краю об'єктів, зрушуючи зображення на деяку величину, а на наступному кроці віднімаючи зміщене зображення від вихідного. В області постійної яскравості результатом вирахування буде нуль, що відповідає відсутності деталей. В області з різкими змінами яскравості, наприклад на кордоні, віднімання дасть велике значення, що призведе до появи світлого пікселя. При роботі з цим алгоритмом можуть з'являтися негативні значення, тому на практиці береться абсолютне значення різниці, щоб результат був невід'ємним. Коли необхідно посилення вертикальних країв, зображення зміщується вліво або вправо. Для посилення горизонтальних країв зображення зміщується вгору або вниз. Для посилення і горизонтальних, і вертикальних країв зображення зсувається в обох напрямках. Замість фактичного зсуву і віднімання використовується згортка.

Виділення контурів методом спрямованого градієнта. Метод зсуву і різниці, детально розглянутий вище, показує, як можна підсилити вертикальні і горизонтальні краю в зображенні. Іноді необхідно підкреслити і інші кордону зображення, не строго вертикальні або горизонтальні. Інтерес можуть викликати, скажімо, діагональні краю. Виборче виділення контурів в різних напрямках іноді продиктовано специфікою розв'язуваної задачі. Для підкреслення меж в 8 різних напрямках застосовуються 8 ядер згортки. Усі обрані напрямкиприйнято позначати як сторони світу: північ, північний схід, схід, південний схід, південь, південний захід, захід, північний захід. Наприклад, градієнт ядра «схід» буде

посилювати край, який містить перехід від чорного до білого зліва направо. Якщо сума коефіцієнтів ядра дорівнює нулю, то області постійної або повільно змінюється яскравості (низькочастотні компоненти) будуть ослаблені. Існує багато спеціальних ядер згортки для посилення і виділення меж об'єктів.

Графічні зображення (карти, схеми, плани, креслення тощо), для яких характерно апріорне їх структурування, утворюючи великий підклас всього різноманіття зображень, не надто ефективно обробляються існуючими методами кодування зображень. Причина цього в тому, що властиві їм внутрішні кореляції надзвичайно сильно і вибірково пов'язані з їх впорядкованої і організованною структурою. Тому відсутність обліку цієї структури призводить до відчутного зниження показників стиснення графічної інформації, що визначають ефективність та пропускну здатність каналів зв'язку телекомунікаційних систем, а також до зниження продуктивності функцій кодування та декодування, або до зростання їх ресурсовитратності.

Контурна інформація відіграє істотну роль при сприйнятті зображень людським зором. Отже, «неконтурная» інформація (області однорідного кольору або плавно мінливого кольору) є візуально надлишкової і може бути частково або повністю виключена з зображення.

На інтуїтивному рівні, перепад – це чіткий безліч пікселів, що лежать на кордоні між двома областями. Інтуїтивно ясно, що ідеальний перепад яскравості має властивості моделі, показаної рисунку 2.5,а.

Відповідно до виділеної моделі представлення перепаду, ідеальний контурний перепад – це набір взаємоз'єднаних пікселів (в нашому прикладі по вертикалі), кожен з яких розміщеється поруч з прямокутним стрибком яскравості, як показує горизонтальний профіль на розбраженні.

а) б)

Рисунок 2.5 – Приклади ідеального перепаду яскравості й чіткого контуру (а) та розмитого контуру (б)

В реальності оптичні обмеження, проведена дискретизація, а також присутні недосконалості та шуми інших елементів системи фото/відеореєстрації зображень призводять до того, що вхідні зображення містять розмиті перепади яскравості. Причому рівень расфокусовки визначається факторами серед яких, якість системи фото/відеореєстрації, крок алгоритму дискретизації, а також умови освітлення в яких відбувався процес отримання зображення. В результаті отримані перепади яскравості на зображенні в більшій мірі відповідають моделі представленій на рисунку рисунку 2.5,б. Крутизна похилої області є обернено пропорційна рівню расфокусовки виділеного перепаду. В моделі даного типу вже відсутня тонка траєкторія (шириною в 1 піксель). Замість цього, можна побачити, що точкою зміни яскравості тепер може бути довільна точка, що знаходиться на похилій ділянці профілю, а сам перепад є множина, утворена всіма такими точками. Така залежність є достовірною, оскільки розмиті перепади мають широкий проміжок зображення, а різкі перепади – тонкі.

З проведеного розгляду випливає, що значення першої похідної можна використовувати при виявленні наявності контуру в кожній точці зображення. Аналогічно, знак другої похідної дозволяє визначити, чи лежить піксель, що знаходиться на перепаді, на темній або світлій його частини.Властивість перетину нульового рівня другої похідної вельми корисно для локалізації середини широких перепадів.

Слід зазначити, що навіть невеликий шум може мати значний вплив на першу і другу похідні, що застосовуються для виявлення перепадів на зображеннях. Зокрема, в практичних завданнях, де можлива присутність помітних перешкод, доцільно розглянути питання про попереднє згладжування зображення.

2.2 Алгоритми скелетизації об’єктів на цифрових зображеннях

Одним із способів розпізнавання об'єктів на зображенні є структурний підхід, заснований на побудові скелета об'єктів. Під скелетом розуміється набір простих прімітвов (лінії, дуги, точки), які виходять в результаті утоншення зображення об'єкта від його зовнішнього кордону до центру тяжіння. Серед підходів скелетизації зображень є такі, що засновані на алгоритмі Зонга-Суня, і хвильовому алгоритмі виділення скелета зображення на основі сферичної хвилі.

Основна ідея алгоритму Зонга-Суня полягає в тому, що для виділення контуру, необхіднона кожному кроці, пробігаючи по зображення рамкою 3×3, перевіряється приналежність кожного пікселя до кордону заданої зв'язної області. Якщо умови перевірки виконуються, то піксель видаляється з області. І незалежно від кількості виконаних кроків область залишиться зв'язною, в граничному випадку вона виродиться в лінію товщиною в один піксель. Побудова скелета зводиться до процесу виділення окремих відрізків та знаходження місць їх з'єднання з послідуючим зберіганням виділених даних в результуючий граф зображення. Виділення відбувається на основі аналізу шляху руху хвилі, з маркуванням пройденого шляху (це проводиться длятого, щоб запобігти повторного проходження хвилі по вже опрацьованому зображенню). В результаті граф скелета об’єкта на зображенні заносяться до середніх точок для кожної з згенерованих хвиль. Зменшення кількості точок в процесі руху хвилі проводиться аналіз переміщення середньої точки останньої генерації хвилі, і в

граф збегігаються тільки точки, в яких проводилась зміна напрямку руху середньої точки. Для виділення ребер визначаються точки, де відбувається поділ хвилі на напівхвилі, тобто з'єднання або перетин відрізків, загасання хвилі тощо. Алгоритм пошуку скелета є рекурсивним. Для організації рекурсії використовується стек для зберігання генерацій хвиль. Пошук закінчується, коли весь об'єкт стає позначеним (тобто, немає можливості для подальшого поширення хвилі). Для оптимізації скелета необхідно виділити всі послідовності ребер такі, що відхилення від корелює прямий для них буде мінімальним і виправлення положення точки перетину відрізків. Знаходження послідовності ребер проводиться послідовним доповненням списку ребер ребрами інцидентними крайнім ребрах послідовності, а оптимізовані ребра позначаються.

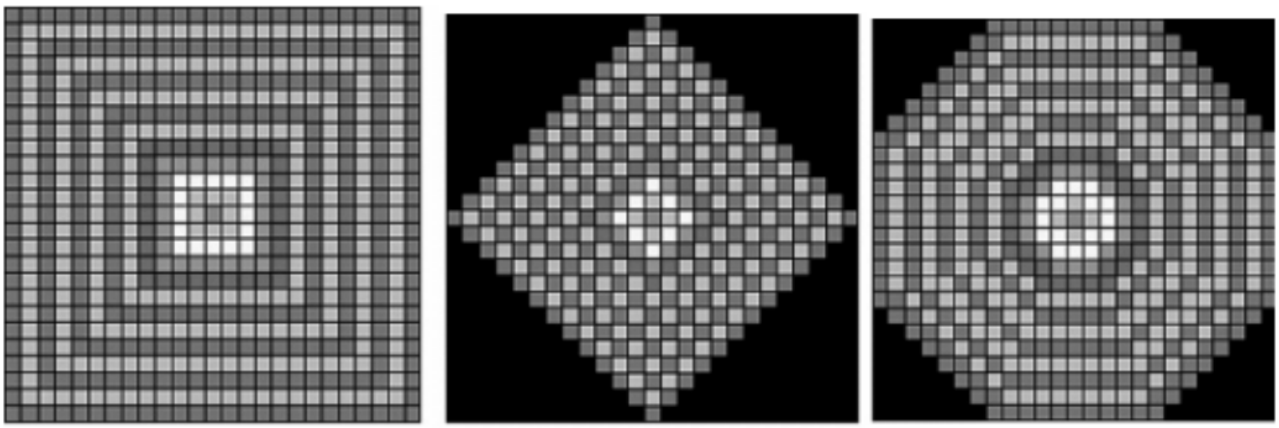
Умовно хвильовий алгоритм можна поділити на кілька етапів:

− побудова графа («запуск» хвилі, відстеження пройденого шляху, місць поділу, загасання);

− оптимізація отриманого графа (апроксимація прямими, колами, видалення зайвих вузлів графа).

Перший етап – отримання навантаженого графа, у якого навантаження вершин – пари координат x, y відповідних вузлових точок зображення [5]. За вузлові точки будемо приймати точки з'єднання ліній в растровому зображенні. Під зображенням лінії (відрізком) на растрі будемо розуміти таку множину чорних крапок растра, що можна провести відрізок прямої такий, що по обидва боки від цього відрізка буде лежати приблизно рівну кількість точок, і відстані від відрізка до найближчих крайніх точок зображення будуть відрізнятися не більше, ніж на наперед задану величину. Такий граф, по суті, буде скелетом зображення.

Побудова графа здійснюється шляхом відстеження шляху проходження сферичної хвилі по зображенню. Однак поширення такої хвилі має обмеження, пов'язане з дискретністю простору. Так для 8-ми зв'язкового растра поширення хвилі йде у вигляді квадрата, а для 4-х – у вигляді ромба [6]. Можливості підключення характеризує кількість сусідніх пікселів растра (рисунок 2.6).



а) б) с)

Рисунок 2.6 – Приклади поширення хвилі під час скелетизації: квадратний тип (а), ромбічний (б) та сферичний (с)

Генерація сферичної хвилі досягається поперемінним застосуванням 4-х і 8-й зв'язкового поширення. Спочатку застосовується 4-х напрямне поширення, потім 8-ми напрямне для кожного пікселя з попередньої 4-х зв'язної генерації подібно за принципом Гюйгенса-Френеля в хвильової теорії, згідно з яким кожен елемент хвильового фронту можна розглядати як центр вторинного обурення, що породжує вторинні сферичні хвилі. Тоді поширення хвилі йде у вигляді восьмикутника. Для такої хвилі характерні деякі особливості:

− не більше ніж через 2N кроків поширення хвилі набуває стійкого характеру незалежно від початкової точки поширення хвилі, де N – ширина лінії в пікселях;

− така хвиля «вміє» повертати, добре огинає різні перешкоди. Невеликі перешкоди в 1-2 пікселя мало впливають на поширення хвилі. Однак такі перешкоди краще видаляти на етапі отримання бінарного зображення для стійкості хвилі. Для побудови наступної генерації хвилі необхідно зберігати останню її генерацію.

Побудова скелета фігури проводиться шляхом відстеження центрів кожної генерації хвилі. Кожна генерація хвилі укладена між своїми крайніми пікселями з координатами (x_1,y_1) та (x_2,y_2). Під крайніми пікселями хвилі розуміються ті пікселі генерації, у яких не більше одного сусіднього пікселя з цієї генерації в 8-

ми звязному сусідстві. За точку скелета можна брати середину відрізка, що з'єднує крайні точки генерації хвилі, і точка буде визначатися як $((x_1,y_1)/2; (x_2,y_2)/2)$. Також за точку скелета можна приймати центр мас генерації хвилі (щодо якого кількість пікселів в генерації зліва і справа однаково), але точки скелета при повороті хвилі будуть лежати ближче до фронту хвилі, що проілюстровано на рисуноку 2.7.

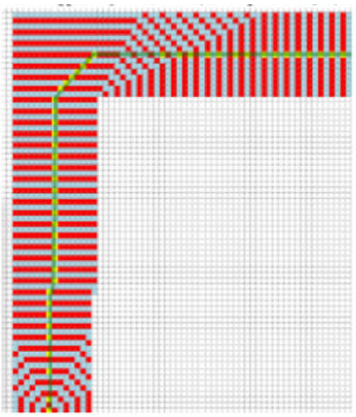


Рисунок 2.6 – Приклади виділення скелетної лінії в хвильовому алгоритмі

При хвильовому алгоритмівточками скелету можуть бути точки у таких варіантах:

− за точку скелета приймається середина відрізка, що з'єднує крайні точки генерації;

− за точку скелета приймається центральний піксель генерації хвилі, щодо якого число пікселів зліва і справа однакове.

При досягненні хвилею місця з'єднання двох або більше відрізків скелета спостерігається поділ хвилі на кілька дочірніх хвиль, що зберігають поведінку материнської хвилі. Перед поділом спостерігається безперервне збільшення ширини хвилі і відбувається збільшення кількості крайніх точок генерації хвилі. Відстежуючи це, можна приблизно визначити місце з'єднання відрізків скелета зображення. У разі повороту хвилі або потовщення елемента зображення кількість крайніх точок не змінюється, тому таке збільшення ширини хвилі можна не брати до уваги.

Сам момент збільшення ширини хвилі визначається шляхом порівняння "ширини" черговий генерації хвилі і її середнього значення за *N* попередніх генерацій (*N* задається заздалегідь) (рисунок 2.7).

а) б)

Рисунок 2.7 – Приклади поширення хвилі під час скелетизації: після 20 ітерацій (а), фінальне поширення (б)

Момент поділу хвилі відбувається, коли зростає кількість крайніх точок однієї генерації хвилі. Кількість дочірніх хвиль буде дорівнювати кількості крайніх точок в генерації, поділеній навпіл, за винятком дочірніх хвиль, що складаються з 1 пікселя. В окремому випадку, після поділу генерації хвилі на 2 напівхвилі, отримуємо ще 2 пари крайніх точок. Побудова скелета зводиться до виділення відрізків і місць їх з'єднання з занесенням знайдених даних в результуючий граф. Важливо позначати всі пікселі генерації хвилі, при цьому кольори для 4-х і 8-ми зв'язного поширення повинні бути різними, щоб не помічати ті пікселі, які вже позначені, і відслідковувати місце зустрічі хвиль. Місце зустрічі хвиль фіксується при мітці вже помічених пікселів. В цьому випадку граф скелета замикається.

Після поділу фронту хвилі можуть виникати поодинокі хвилі через дискретності зображення. Середину такої хвилі в граф заносити не потрібно.

Отже, на першому етапі в скелетний граф зображення заносяться середні точки для кожної генерації хвилі.

Для зберігання генерацій попередніх хвиль доцільніше використовувати чергу. Так як, наприклад, при використанні стека виникають «сплески» хвиль,

пов'язані з дискретністю зображення, що має деякі перешкоди. Використання черзі дає більш рівномірне поширення хвилі, яке імітує «природну» хвилю.

Алгоритм першого етапу має наступний вигляд:

- створюється порожня черга;
- в неї кладеться генерація хвилі, представлена 4-х зв'язковими сусідами початкової точки;
- поки черга не порожня, з неї дістається генерація хвилі;
- до неї застосовується спочатку 8-ми зв’язне поширення, потім 4-х зв’язне;
- виходить нова генерація, яка відправляється в чергу, а середина нової хвилі заноситься в кістковий граф;
- в разі поділу хвилі в чергу записуються відомості про всі дочірні хвилі, а також місце з'єднання відрізків зображення як центр багатокутника, утвореного крайніми точками материнської і дочірніх хвиль;
- при зустрічі з іншою хвилею відбувається загасання хвилі. В даному випадку граф замикається.

На першому етапі виходить скелет, що складається з середин всіх генерацій хвиль за винятком місць з'єднання. Наступний етап процесу скелетизації – оптимізація отриманого скелета.

На початку роботи для оптимізації графа (тобто видалення надлишкової інформації) використовується метод послідовних наближень, який полягає в наступному: через перші дві вершини графа проводиться пряма, і кожна наступна точка перевіряється на відхилення від цієї прямої. Якщо відхилення не більше e (задається порівнянним з товщиною ліній зображення), то ця точка належить поточної прямий, інакше утворює наступну пряму.

Отриманий набір точок, що належать одній прямій, апроксимується за допомогою ортогональної регресії. Після чого кінцеві точки коригуються, тобто проектуються на аппроксимирующую пряму, а решта проміжні точки видаляються з графа.

Всі пройдені вершини позначаються, щоб уникнути зациклення програми, так як граф в загальному вигляді може бути замкнутим.

Крім цього, може використовуватися оптимізація точок з'єднання відрізків зображення. Таке спотворення детектується наступним чином: через останні два вузла графа, що передують точці з'єднання, проводиться пряма. Інші останні дві точки, що передують цьому ж самому місцю поділу, але лежать на інший «гілці» графа, повинні відхиляються від неї з максимальним відхиленням E (вибирається сумарною з товщиною лінії), а точка з'єднання – перевищувати це відхилення.

Ще одним важливим етапом є етап отримання вузлових точок скелету. Алгоритм отримання таких точок грунтується на аналізу отриманої скелетної лінії та пошук місць, що можууть відповідати за вузловим точкам (рисунок 2.8):

- місця розгалуження;
- місця ріхкої зміни напрямку лінії;
- утворення спіралей.

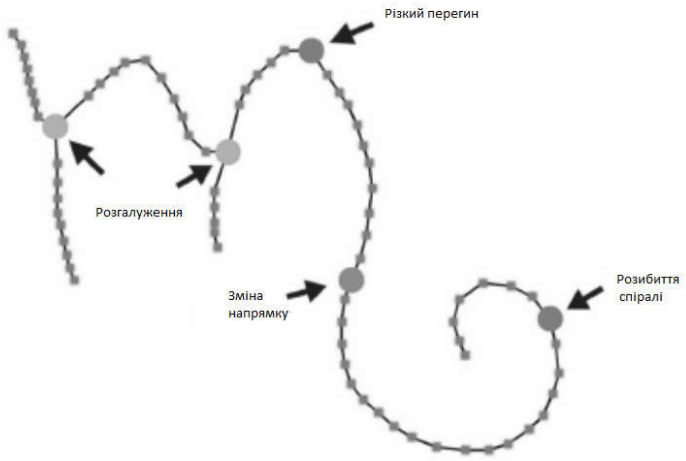


Рисунок 2.8 – Приклади виділення вузлових точок на скелеті

Переваги хвильового алгоритму для скелетизації зображень:

- скелет будується на оригінальному документі, при цьому вихідне зображення можна відновити;
- пам'ять потрібно тільки для зберігання графа скелета, черги генерації;

− використовуються прості машинно-арифметичні операції на множині дійсних чисел;

− легко розпаралелюється при використанні декількох початкових точок генерації хвилі;

− даний алгоритм добре підходить для розпізнавання букв, оскільки буква, як правило, є цілісною одиночній фігурою невеликого розміру з невеликою кількістю розгалужень і перетинів.

До недоліків даного алгориму можна віднести:

− залежність скелета від вибору початкової точки для деяких об'єктів, однак це усувається на наступному етапі алгоритму – оптимізації;

− в початковий момент хвиля набуває стійкого характеру не відразу, але і не більше, ніж через 2*N* кроків поширення;

− для створення ж скелета відбитків пальців даний алгоритм підходить менше, оскільки структура папілярного візерунка пальців складніше, поширення хвилі і її подальший аналіз проходить з великими похибками.

2.3 Алгоритми скелетизації об’єктів на основі вписаних квадратів

Провівши аналіз відомих алгоритмів та виділивши їх основні переваги та недоліки було спроектовано алгоритми скелетизації областей цифрового зображення. Запропонований алгоритм грунтується на виділенні квадратних областей, що є вписаними в область з подальшим записом координат точки перетину діагоналей даного квадрата. Розроблено два алгоритми виділення скелетів, відмінності полягають у способі вписання квадратів в виділенні області.

Перший алгоритм базується на послідовній спробі вписати квадрати шляхом промалювання максимальних відрізків у всі вертикальні та

горизонтальні напрями від кожної контурної точки області. Графічно даний алгоритм проілюстровано на рисунку 2.9.

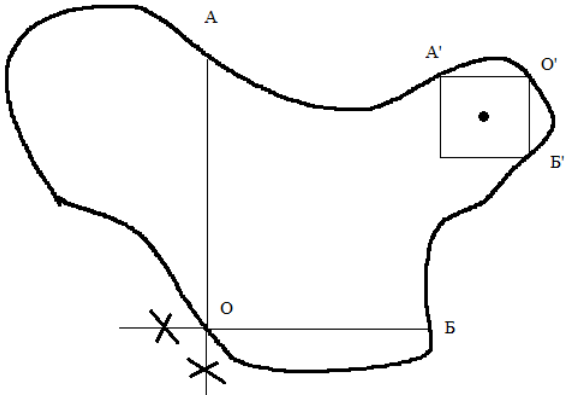


Рисунок 2.9 – Виділення скелетів на основі аналізу контурних точок

Як видно з вище наведеного рисунку, процес скелетизації проходить послідовно для кожної з контурних точок. При аналізу точки *О*, можна побачити. Що відбулась спроба провести максимальні відрізки, що з'єднують точку *О* з іншими точками контуру області. Причому дані відрізки повинні бути строго паралельні до осі ординат та абсцис та кожна з їх точок повинна належати області. В даному прикладі, видно, що з точки *О* було успішно проведено відрізки вгору *ОА* та вправо *ОБ*. Спроби провести відрізки вліво та вних не увінчались успіхом, оскільки в даних напрямках область не поширюється. На наступному кроці, проводиться перевірка отриманих відрізків *ОА* та *ОБ*, якщо їх довжини рівні, то проводиться спроба за дпомогою них утворити квадрат, при цьому повинна зберігатись умова, що всі точки квадрата повинні належати області. Якщо відрізки *ОА* та *ОБ* не рівні між собою, то така точка визнається малоінформативною та виключається з подальшої обробки. При аналізу іншої контурної точки *О'*, то як видно з ілюстрації, з нею було успішно проведено два відрізки *О'А'* та *О'Б'*, що є рівними між собою, та відбулося вписання квадрата. При цьому точка перетину діагоналей вписаного квадрата буде визнаною такою, що належить скелету області.

Блок схема алгоритму наведена на рисунку 2.10.

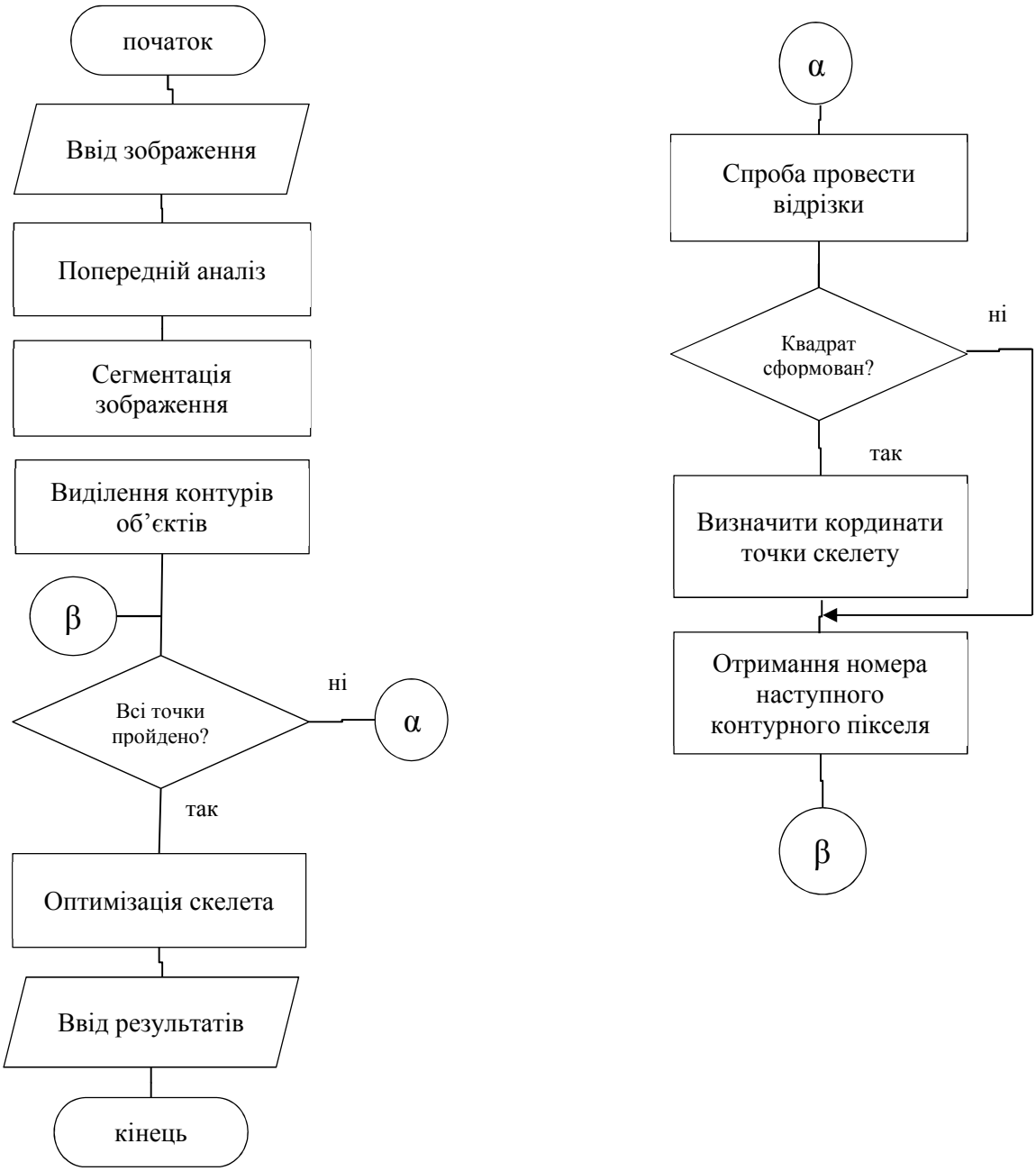


Рисунок 2.10 – Блок-схема алгоритму скелетизації зображення

Запропонований алгоритм у дозволяє проаналізуват всі точки контуру області, а отже з викокою точністю може промалювати скелет області. При цьому впроваджений етап оптимізації скелету, за разунок видалення малоінформацівних віток скелету, що в подальшому дозволило зменшити об'єми пам'яті для зберігання опису області.

Запропонований алгоритм складається з такої послідовності кроків:

Крок 1. Завантаження вхідного зображення.

Крок 2. Проведення попередньої обробки зображнння. На даному етапі проводиться перетворення вхідного зображення з кольорового у градації сірого. Проводиться видалення сторонніх шумів тощо.

Крок 3. Проведення процедури виділення областей, для подальшої скелетизації. Процедура виділенняобластей відбувається на основі сегментації вхідного зображення, при цьому вважаємо, що для подальшого аналізу необхідні однорідні області.

Крок 4. Проходження контуром виділеної області. На даному етапі формується масив контурних взаємозв'язних точок, що описують форму видраної області.

Крок 5. Послідовно проводиться спроба промалювати відрізки максимальної довжинив горизонтальних та вертикальних напрямках з кожної точки контуру. При цьому на відрізки накладається умова, що вони повинні повнюстю налажети області.

Крок 6. Якщо отримані відрізки мають рівну довжину, то проводиться спроба сформувати вписаний квадрат інакше прехід на крок 8.

Крок 7. Перевірка належності квадрата аналізовані області. Якщо вадрат належить області, то проводиться обчислення точки перетину діагоналей квадрата, інакше крок 8.

Крок 8. Обчислення індексу наступної контурної точки. Якщо всі точки пройдено то прехід на крок 9, інакше крок 5.

Крок 9. Обчислення ваг кожної точки скелету, вагу визначаємо як довжину сторони квадрата, який описав дану точку.

Крок 10. Пошук вузлових і кінцевих точок скелету та формування ребер скелету.

Крок 11. Визначення ваги кожного з ребер скелету, вага ребра скелету рівна сумі ваг всіх точок, що входять в дане ребро.

Крок 12. Відсікання малоінформативних ребер.

Крок 13. Перерахунок точності оптимізованого скелету. Скелет після відновлення повинен з заданою точністю відтворити область.

Крок 14. Підрахунок виділених областей та формування звіту.

Дана послідовність кроків забезпечує повноцінний аналіз вхідного зображення та дозволяє скелети областей, що в кінцевому варіанті дозволить отримати компактний опис області на основі її скелету.

До перваг даного алгоритму слід віднести його простоту та швидкість роботи. До недоліків: необхідність проведення процесу налаштування програми пред її роботою, оскільки для коректного та швидкого функціоування необхідно встановити початкові параметри роботи алгоритму (точність відтворення, параметри скелетизації тощо).

3 ПРОГРАМНА СИСТЕМА СКЕЛЕТИЗАЦІЇ ОБ’ЄКТІВ НА ЦИФРОВИХ ЗОБРАЖЕННЯХ

3.1 Структура програмної системи скелетизації об’єктів

Добре відомо, що більше 90% інформації людина отримує за допомогою зору. І тому не дивно, що технічний прогрес зачіпає в першу чергу процеси збору і обробки візуальної інформації.

Поява комп'ютерів неминуче викликало бажання підключити такий комп'ютер до мікроскопу і телескопу, звільнивши людину від нудного і що загрожує помилками процесу аналізу і підрахунку різних об'єктів, які потрапили в поле зору. Однак, виникла несподівана проблема: комп'ютер – не людина, він не бачить і не розуміє введене в нього зображення, для нього це просто набір даних, ніяк не пов'язаних між собою. У той же час людина, дивлячись на будь-яке зображення, практично завжди може виділити на ньому якісь характерні особливості, причому відбувається це майже миттєво.

Взаємодія людини і комп'ютера в зорової області стало дуже тісним, але розділилося на два напрямки: інтерактивний режим роботи і автоматичний.

В інтерактивному режимі комп'ютерні програми перетворюють зображення відповідно до бажання людини, що дозволяє йому краще побачити якісь частини або особливості зображення, виділити потрібні елементи і підрахувати їх кількість або геометричні параметри.

Автоматичний режим має на увазі автономну роботу комп'ютера, як по збору інформації (що не є проблемою вже давно), так і за її аналізу. І ось тут якраз проблема в тому, що на сьогоднішній день не існує способів повністю автоматизувати процес аналізу. При всьому швидкодії комп'ютерів і їх величезної пам'яті, ніяк не вдається змусити їх робити те, що людина (особливо фахівець в конкретній галузі) може зробити буквально одним поглядом.

Для виконання поставлених задач необхідно грамотно обрати середовище обробки даних, а також проаналізувати доступні бібліотек обробки зображень. Щодо бібліотек, в Java є декілька бібілотек для роботи з зображеннями.

AWT – це вбудована бібліотека Java, яка дозволяє користувачеві виконувати прості операції, пов'язані з відображенням, такі як створення вікна, визначення кнопок і слухачів тощо. Вона також включає методи, які дозволяють користувачеві редагувати зображення. Не вимагає установки, так як поставляється з Java. Щоб намалювати фігуру на зображенні, користувачу потрібно буде використовувати об'єкт Graphics, пов'язаний з завантаженим зображенням. Об'єкт Graphics інкапсулює властивості, необхідні для виконання основних операцій рендеринга. Graphics2D – це клас, який розширює Graphics. Це забезпечує більший контроль над двовимірними формами.Дл вирішення поставлених задач Graphic2D буде використаний, щоб збільшити ширину фігури, щоб вона була добре видна, це буде досягатись шляхом збільшення його властивості s _troke _.

Інша відома бібліотека функції дял роботи з зображеннями ImageJ – це програмне забезпечення на основі Java, створене для роботи з зображеннями. У ньому досить багато плагінів. Це досить потужна бібліотека, краще, ніж Swing і AWT, оскільки метою її створення була обробка зображень, а не операції з графічним інтерфейсом. Модулі містять багато безкоштовних алгоритмів, що дозволить прискорити обробку зображень і швидко побачити результати, а не вирішувати математичні та оптимізаційні задачі, що лежать в основі алгоритмів

OpenIMAJ - це набір бібліотек Java, орієнтованих не тільки на комп'ютерне зір і обробку відео, але також на машинне навчання, обробку звуку, роботу з Hadoop і багато іншого.

Бібліотека TwelveMonkeys ImageIO призначена для розширення API Java ImageIO з підтримкою більшої кількості форматів. У більшості випадків код буде виглядати так само, як вбудований код Java, але він буде працювати з додатковими форматами зображень після додавання необхідних залежностей.

За замовчуванням Java підтримує тільки наступні п'ять форматів зображень: JPEG, PNG, BMP, WEBMP, GIF.

Якщо працювати з файлом зображення в іншому форматі, додаток не зможе його прочитати і при доступі до змінної BufferedImage видасть NullPointerException. TwelveMonkeys додає підтримку наступних форматів: PNM, PSD, TIFF, HDR, IFF, PCX, PICT, SGI, TGA, ICNS, ICO, CUR, Thumbs.db, SMG.

Для роботи з зображеннями в Java є безліч бібліотек, проте навіть ті бібліотеками, які вільно поширюються надають великі функціональні можливості для обробки, аналізу та опису зображень. Для роботи з готовими алгоритми обробки зображень, таких як виділення країв, посилення контрасту, використання фільтрів або розпізнавання осіб. Для цих цілей оптимально підходять ImageJ або OpenIMAJ. І те, і інше легко включити в проект, і вони набагато функціональніші, ніж AWT, щодо обробки зображень.

Після проведеного аналізу поставлених задач, та розробки алгоритмів обробки даних або спроектовано структуру програмної системи аналізу та опису цифрових зображень (рисунок 3.1).

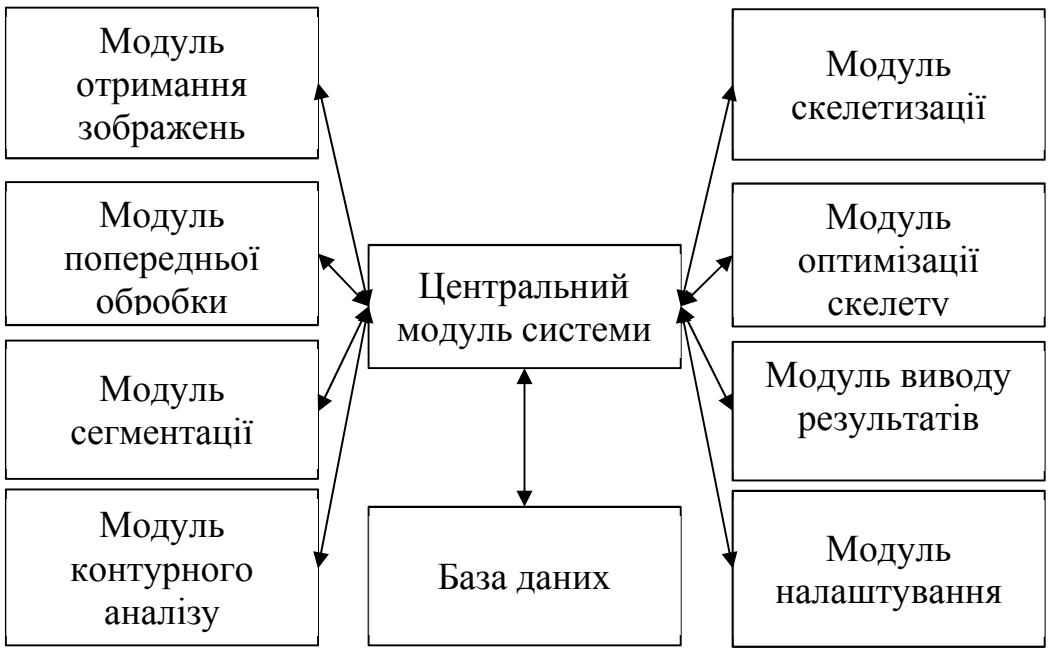


Рисунок 3.1 – Структура програмної системи скелетизації зображень

Розроблена структура програмної системи дозволяє в повній мірі забезпечити виконання поставдених завдань. Серед яких провести додаткову обробку вхідного зображення, процеси розбиття та виділяння областей, опис форм даних областей та їх опис за допомогою скелетів.

Розглянемо основні характеристики найбільш важливих функціональних модулів системи.

«Центральний модуль системи» – даний модуль призначений для коректної організації взаємодії між окремими модулями системи. В ньому містяться засоби перевірки цілісності програмної системи, доступність окремих моділів, актуальність версій та забезпечення передачі даних між окремими модулями під час роботи системи. Даний модуль є основним в системі.

«Модуль отримання зображень» – функції даного модуля забезпечують завантаження вхідних зображень та переконвертація їх у формати, що підтримуються програмною системою. Оскільки наперед не відомо який саме формат буде у вхідного зображення, то для запобігання критичних помилок в роботі системи необхідна система перевірок вхідного зображення. Якщо користувач має невеликий досвід роботи з програмами даного типу, то на вхід може бути подано зображення формат якого не підтримується, після цього програма спробує провести перекодування у зручний для неї формат, а при неможливості виконати дану дію, запропонує користувачеві змінити вхідне зображення.

«Модуль попередньої обробки» – для підвищення точності виконання процесу сегментації необхідно максимально зменшити вплив дефектів вхідного зображення. Саме для цієї мети в програмі передбачено модуль попередньої обробки, в який входять функції корекції яскравості та контрасту, функції виділення обремих областей на зображенні, для пришвидшення процесу аналізу, функції перекодування зображення з кольорових в бінарні або в градації сірого, для зменшення об’ємів пам’яті при зберіганні зображень.

«Модуль сегментації» – процес розбиття вхідного зображення на окремі одноріжні за деяким критерієм області наивається сегментація. Оскільки для

опису зображння необхідно виділити окремі об’єкти, то для цього використовуються функції саме цього модуля. Серед алгоритмів, що реалізовані в даному модулі є: алгоритм порогової сегментації, водоподілу та нарощування областей. Дані алгоритми в повній мірі забезпечують можливість обробки та виокрелення областей для подальшого опису.

«Модуль контурного аналізу» – для проведення процедури скелетизації необхідно описати та закодувати форму області інтересу. Для цього використовуються засоби контурного аналізу, серед яких окремою групою реалізовані алгоритми опису контуру шляхом його проходження. Вибір алгоритмів саме даної групи зводиться до того, що при своїй простоті та високій швидкодії мони можуть описувати форми овільної складності. Кодування контуру відбувається шляхом занесення виділених точок в структуру, що описує точку звязного контура як пару чисел, що відповідають координатам даної точки по осях абсцис та ординат на вхідному зображенні. Кодування в даному форматі оптимально підходить для подальшої обробки зображення запропонованим алгоритмом.

«Модуль скелетизації» – один з головних модулів системив якому реалізовані функції скелетизації. Для виділення скелету проводиться послідовний пошук точок, що є рівновіддалені від хоча б двох прямих за допомогою яких описано контур. Дана умова забезпечується якщо в вибрану область вписквати квадрати, оскільки в них сторони є однаковими, а отже якщо дві протилежні вершини квадрата лежить на контурі області, то точка претину діагоналей квадрата буде рівновіддаленою, а отже може бути визнана як точка скелету. Результатом роботи даного модуля є масив точок, що є рівновіддаленими від контуру області та формують скелет.

«Модуль оптимізації скелету» – Оскільки після завершення роботи алгоритмів скелетизації в програмі зберігаєть набір незв’язних точок, що відповідають скелету області,то необхідні додаткові операції для отримання моноскелета. Для цього проводиться пошук та з’єднання точок скелету, які розташовані поруч. Після чого проводиться спроба об’єднати окремі частини

скелету в однорідну структуру. Після отримання моно скелету на ньому може бути присутся деяка кількість малоінформативних ребер. Ребра даного типу утворюються в результати похибок дискретизації, зашумленості контуру тощо. Для їх відсікання та подальшої роботи з більш інформативним скелетом відбувається процедура відновлення області на основі скелету. Причому, якщо відновлена без деякого ребара область відповідає з деякою похибкою хідній області, то таке ребро визнається малоінформативним та видвляєтсья з структури скелету. Перевірку послідовно проводять для всіх кінцевих ребер скелету. В результаті рооти даного модуля отримується моноскелет, у якого відсічені малоінформативні ребра, а отже в подальшому це дозволить зменшити обєми памяті, що необхіні для зберігання опису скелету, а також підвищить час обробки даного скелету.

«Модуль виводу результатів» – для візуалізації отриманих результатів було спроектовано та реалізовано ряд функцій для формування звітів та виводу візуальних підтверджень роботоздатності програмної системи. Даний модуль єдопоміжним в структурі програмного додатку, та виконує завдання в залежності від команд користувача та параметрів налаштування роти системою.

«Модуль налаштування» – набір функції, що активують систему премикачів та діалогів для встановлення користувачем оптимальних для роботи параметрів. Серед можливих параметрів слід відмітити розміти та типи шрифтів, що використовуются для візуалізації роботи програмної системи, параметри завантаження та попередньої обробки зображень, параметри алгоритмів сегментації, порогові значення ля проведення контурного аналізу та процедури апроксимації зовнішнього контуру області, параметри скелетизації та порогові значення для процесу оптимізації скелету тощо.

«База даних» – допоміжний модуль для роботи з сервером бази даних, використовєтсья у випадку, якщо необхідно зберегти великі об’єми інформаці, або для передачі бази даних оброблених зображень зі скелетами між користувачами. Модуль носить допоміжний характер та може не

використовувати під час обробки простих завдань або невеликої кількості зображень.

Для отримання комплексної оцінки про розроблену структуру програмної системи було проведено моделювання даного аріхітектури за допомогою UML діаграм, а саме послідовності та прецедентів.

Для дослідженя системи взаємодії між окремими елементами системи та користувачем було проведено можелювання на основі діаграм прецедентів, яку наведено на рисунку 3.2.

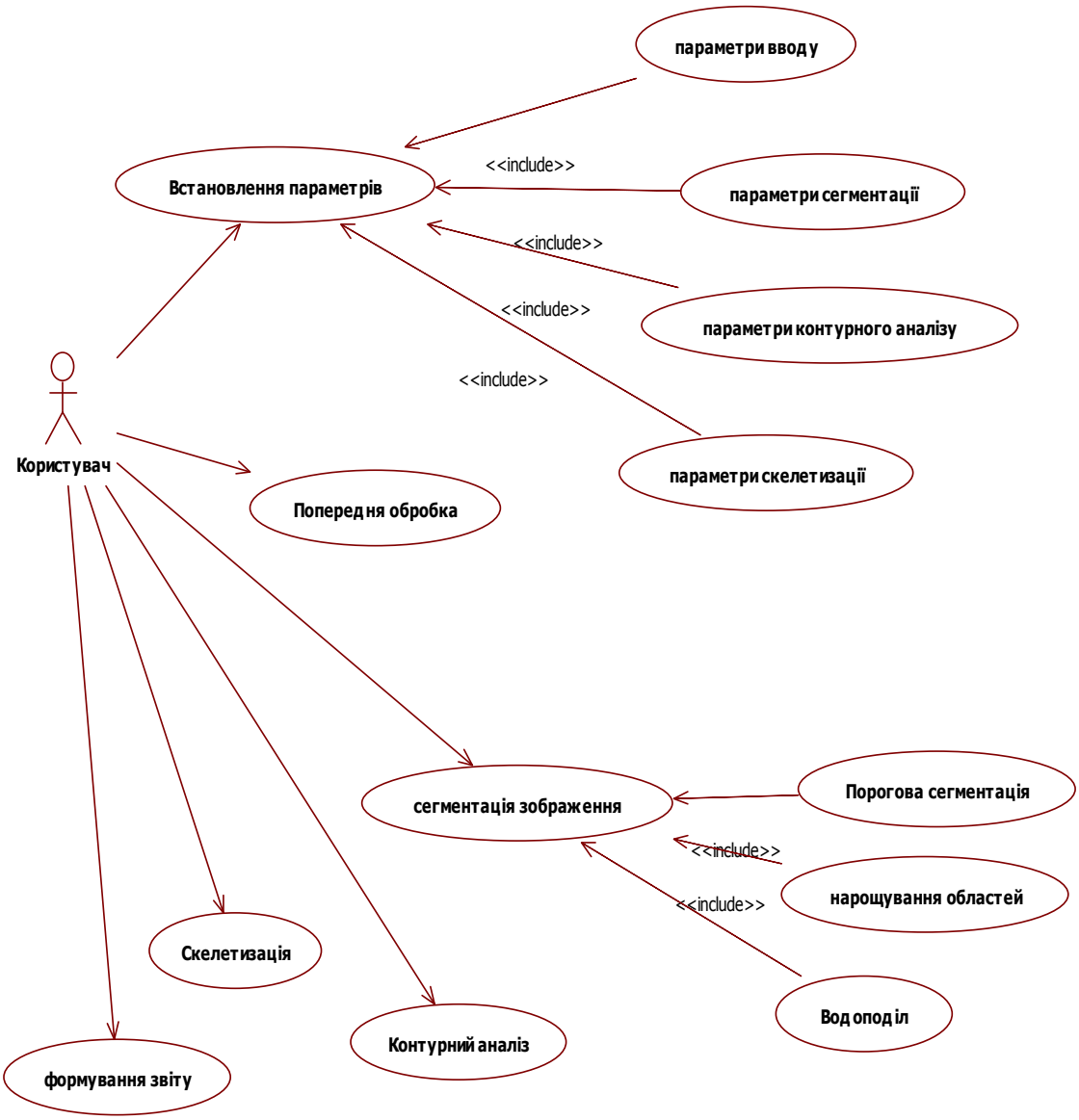


Рисунок 3.2 – Діаграма прецедентів системи скелетизації зображень

Як видно з наведеної діаграми, група «Користувач» отримає доступ до різних функціональних параметрів роботи програмної системи, що в достатній мірі забезпечить якісну роботу системи та дозволить користувачам отримати потрібні результати при вирішенні поставлених задач.

Іншим важливим кроком в перевірці якості спроектованої програмної системи слід проподелювати послідовності кроків, які здійснить система при роботі зкористувачем і чи там не відоубдеться зациклення роботи системи. Результат моделювання наведено на рисунку 3.3.

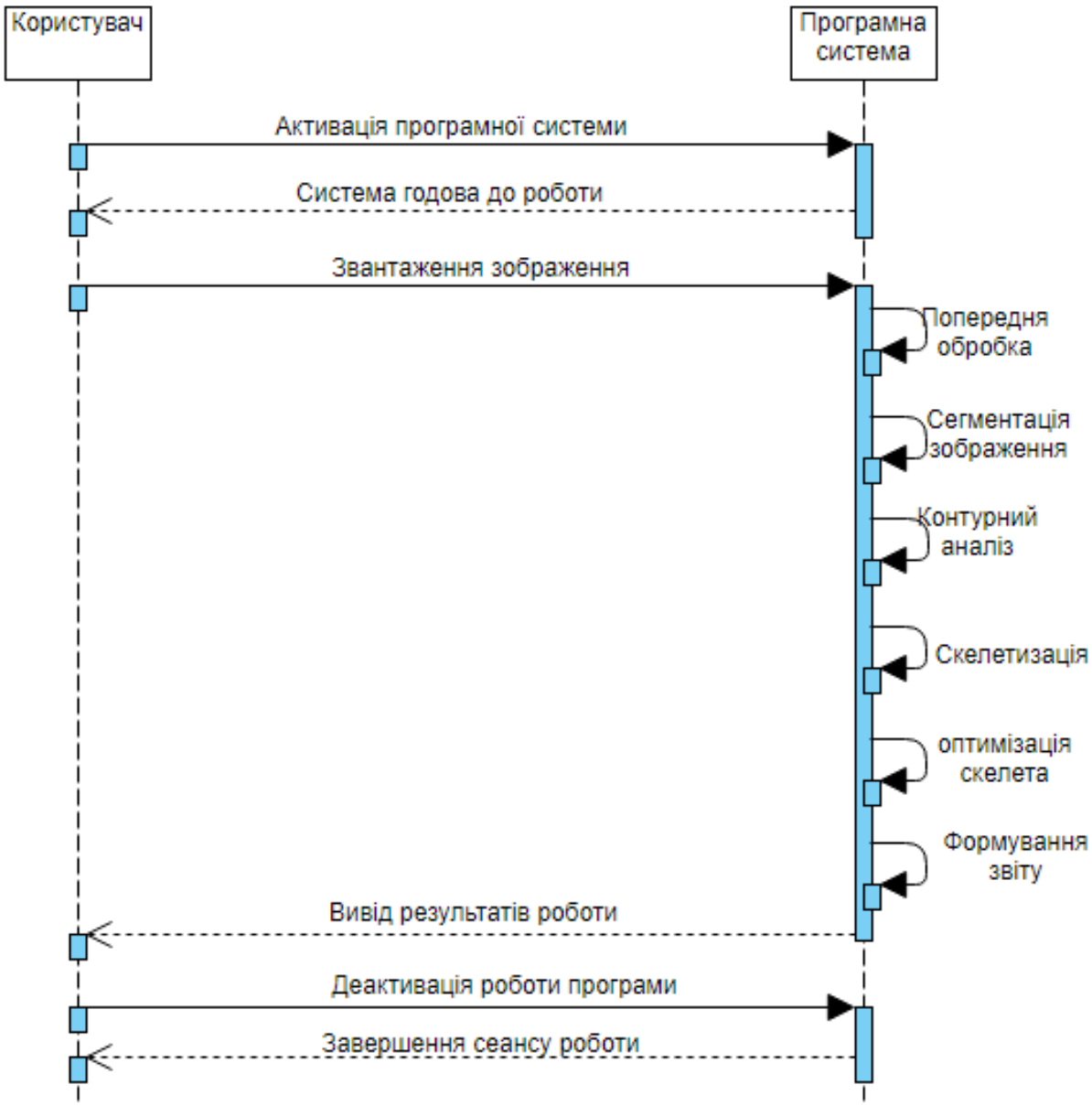


Рисунок 3.3 – Діаграма послідовності системи скелетизації зображень

Як показали результати моделювання програмної системи, всі процеси відувається послідовно, закінчення одного з процесів, автоматично активує початок іншого, а отже зависання роботи програмної системи неможливе.

Для роботи з системою було розроблено та реалізовано простий користуваціий інтерфейс, який представлено на рисунку 3.4

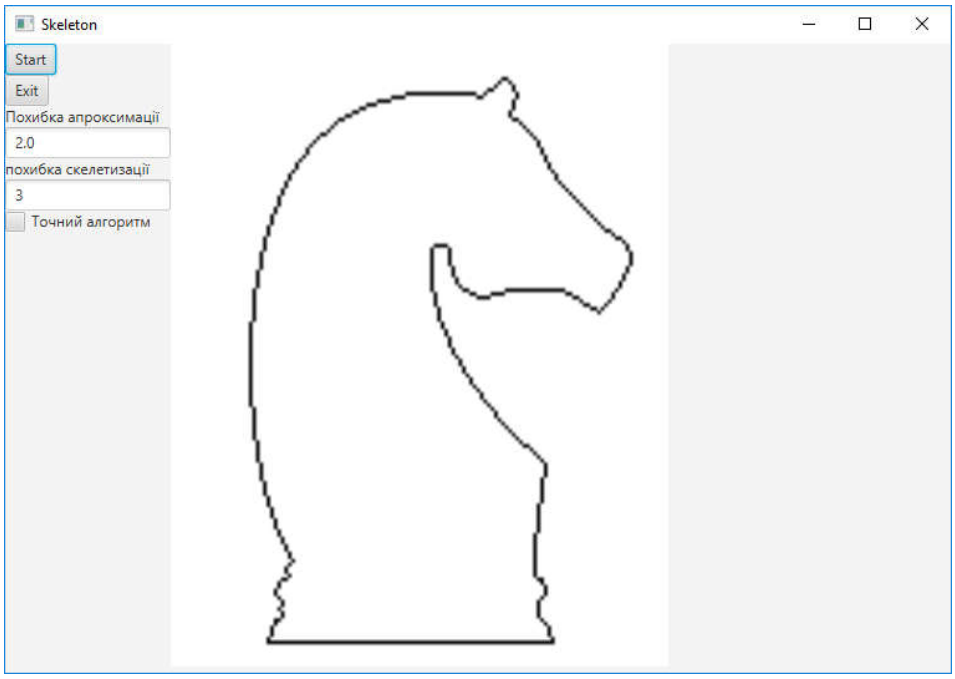


Рисунок 3.4 – Загальний вигляд головного вікна програмної системи

Головне вікно програмної системи розділено на два основіні функціональні елементи. По-перше, це область візуалізації результатів роботи програмної системи, в даній області головного вікна відбуваєть вивід проміжних та остаточни результатів опрацювання цифрових зображень. По-друге, область керування програмним додатком. В даній частині знаходяться ряд перемикачів для активації відповідних фукці програмної розробки. Окрім того користувач може самостійно впливати на рооту програми шляхом встановлення параметрів роботи програми.

Даний зовнішній вигляд програмної системи достатньо зручний та швидкий в опануванні, що є безперечною перевагою під час роботи з програмним продуктом.

3.2 Програмні модулі системи скелетизації об’єктів на цифрових зображеннях

Використовуючи класи з інтерфейсом BufferedImageOp і RasterOp, можна накласти фільтр на зображення або застосувати один з rgb ефектів, як збільшення яскравості. Успіх обробки також залежить від типу зображення. Наприклад, використовуючи RescaleOp на зображенні з типом BufferedImage.TYPE_3BYTE_BGR, отримано повідомлення про крах джава машини. А з типом BufferedImage.TYPE_INT_BGR все було відмінно.

При описі деяких класів використовується термін банк-масив значень одного компонента кольору зображення. Наприклад, червоний банк-масив червоної складової для всіх пікселів зображення. А взагалі в літературі по графіку можуть використовуватися синоніми: план або канал.

Для демонстрації можливостей роботи розроблених компонент, продемонструємо його роботу щоб він при створенні завантажував зображення, обробляв його і зберігав результат в іншому зображенні. А для відтворення вихідне зображення і результуюче використовуються методи роботи з зображенням винесені в клас ImageUtil.

```
public class MyComponent extends JComponent {
BufferedImage image_in;
BufferedImage tstimg;
public MyComponent() {
img = ImageUtil.loadImage("t.png");
// використовуємо ефект
tstimg = ImageUtil.gray(image_in);
}
@Override
protected void paintComponent(Graphics g) {
Graphics2D g2 = (Graphics2D) g;
g2.drawImage(image_in, 0, 0, null);
g2.drawImage(tstimg, image_in.getWidth()+15, 0, null);
}}
```

Використання фільтрації в програмному додатку. Для накладення фільтра використовується клас ConvolveOp. Матриця фільтра описується класом Kernel, якому передається одновимірний масив чисел типу float, і розміри, яким чином цей масив представляти у вигляді матриці. Використовуваний алгоритм не виробляє кінцевого розподілу на суму елементів матриць. Так матриця розмиття (пом'якшення) повинна задаватися:

```
float matr = 1.0f / 9.0f;
float[] filter = { matr, matr, matr, matr, matr, matr, matr, matr,
matr };
```

а не:

```
float matr = 1.0f;
float[] filter = { matr, matr, matr, matr, matr, matr, matr, matr,
matr };
```

Нижче наведено метод blur з класу ImageUtil, який накладає фільтр розмиття:

```
public static BufferedImage blur(BufferedImage image_in) {
float matr = 1.0f / 9.0f;
float[] filter = { matr, matr, matr, matr, matr, matr, matr, matr,
matr };
BufferedImageOp hhop = new ConvolveOp(new Kernel(3, 3, filter));
return op.filter(image_in, null);
}
```

Також дивіться метод sharpen, який накладає виділення країв. Для кращої якості, можна створити збільшене зображення на (ширина матриці-1)/2 по ширині і (висота матриці-1)/2 по висоті. Вивести вихідне зображення в центр нового, а потім застосувати фільтр з опцією не обробляти краю зображення.

```
BufferedImageOp hhop =
new ConvolveOp(new Kernel(3, 3, filter), ConvolveOp.EDGE_NO_OP,
null);
```

Для роботи з різними типами зображення можна використати клас ColorConvertOp дозволяє перетворити поточний простір кольорів зображення вказане. Це можна використовувати, наприклад, для створення чорно-білої копії зображення

```
public static BufferedImage gray(BufferedImage image_in) {
ColorSpace cs = ColorSpace.getInstance(
ColorSpace.CS_GRAY);
BufferedImageOp hhop = new ColorConvertOp(cs, null);
return op.filter(image_in, null);
}
```

Для виконання операцій з заміною кольорів, слід використовувати клас LookupOp дозволяє замінити значення в зазначеному банку або у всіх банках на відповідні значення з масиву. Наприклад, якщо в червоному банку зустрілося значення 64, то воно буде замінено на значення 64 елемента масиву. Це дозволяє досягти таких rgb ефектів як огрубіння і інверсія (кольоровий негатив).

```
public static short[] getPosterizePal() {
short ret[] = new short[256];
for (int i = 0; i < 256; i++)
ret[i] = (short) (i - (i % 32));
return ret;
}
public static BufferedImage posterize(BufferedImage image_in) {
short[] lookup = getPosterizePal();
BufferedImageOp hhop = new LookupOp(new ShortLookupTable(0, lookup),
null);
BufferedImage ret = copy(image_in, BufferedImage.TYPE_INT_RGB);
hhop.filter(ret, ret);
return ret;
}
```

Для проведення масштабування кольроів на зображнні використовується набур таких коофіціїнтів. Клас RescaleOp дозволяє помножити значення компонент кольору на коефіцієнт і після додати вказану зміщення.

```
newr = (oldr * fact) + offs
newg = (oldg * fact) + offs
```

```
newb = (oldb * fact) + offs
```

Множник вказується числом типу float, а зсув як ціле число від 0 до 255 Якщо зсув дорівнює 0, то результат буде аналогічний фільтрації зображення з матрицею ядра 1x1. Нижче наведено метод збільшення яскравості.

```
// factor = 1.45f підвищить яскравість на 45%
public static BufferedImage bright(BufferedImage image_in, float
fact) {
RescaleOp hhop = new RescaleOp(fact, 0, null);
BufferedImage ret = copy(image_in,BufferedImage.TYPE_INT_BGR);
hhop.filter(ret, ret);
return ret;
}
```

Також дивіться метод light для освітлення / затемнення зображення.

Ще однією важливою функцією роботи програмиє можливість працювати з зображеннями, що були оброблені за допомогою афінних перетворень. Клас AffineTransformOp> виконує аффіні перетворення над зображенням, використовуючи визначену наперед матрицю перетворення. Наприклад, нижче наведено програмний коддля проведення процедури масштабування зображення. Якщо аргументи більші за 1, то зображення збільшується, менші 1 – зменшують.

```
public static BufferedImage scale(BufferedImage image_in, float
fwtt, float fhtt){
AffineTransform mt =
AffineTransform.getScaleInstance(fwtt, fhtt);
AffineTransformOp hhop = new AffineTransformOp(mt,
AffineTransformOp.TYPE_BILINEAR);
return hhop.filter(img, null);
}
```

Деякі ефекти афінних перетворень можна отримати методом drawImage.

```
public static BufferedImage resize(BufferedImage image_in, int Nw,
int Nh) {
```

```
BufferedImage ret = new BufferedImage(Nw, Nh,
rightTypeImage(image_in));
Graphics2D g2 = ret.createGraphics();
g2.setRenderingHint(RenderingHints.KEY_INTERPOLATION,
RenderingHints.VALUE_INTERPOLATION_BILINEAR);
g2.drawImage(image_in, 0, 0, Nw, Nh, null);
g2.dispose();
return ret;
}
```

Деякі ефекти афінних перетворень можна отримати методом drawImage.

Трохи довше, і довелося коригувати тип зображення, так як для зображення метод getType класу BufferedImage повертав 0. А це не дозволяє виводити в зображення.

Метод rightTypeImage введений, щоб коригувати тип зображення, якщо getType вихідного зображення дорівнює 0. Був необхідний для ефектів реалізуються методом drawImage класу Graphics2D.

Не всі класи змогли створити автоматично коректне результуюче зображення методом filter інтерфейсу BufferedImageOp. Тому додатково було реалізовано метод copy, що робить копію вихідного зображення з корекцією типу зображення. Щоб результат тесту збігався з результатом інших графічних програм (XnView), було видалено альфа-канал з зображення другим методом copy, де можна вказати бажаний тип зображення.

Для комфортної роботи та для групування всіх іункції. Що задіяні в процесі скелетизації був розроблений програмний клас Skelet, основні його методи та атрибути наведено нище.

```
public class Skelet {
    public
    int max_kilk_elements = 5000; //максимальна кількість точок в
контурі
    int Height, Width; //висота та ширина вхідного зображення
    int perumetr=0; //периметр обєкта
    int contour[][] = new int [max_kilk_elements][2]; //контурні
точки
    int aprox_contour [][] =new int [max_kilk_elements/10][4];
//прямі апроксимуючі лінії,
другий параметр:
0 - X1,
```

```
1 - Y1,
2 - X2,
3 - Y2
    int skel[][] = new int [max_kilk_elements][3]; //точки що формують скелет, , другий параметр
0 - координата X,
   1- координата Y,
   2- 2 - вага точки
    int kilk_skelet_point;
    int main_line[][] = new int [2][3]; //координати максимальної осі обєкта, другий параметр 0 - координата X, 1- координата Y, 2 - номер точки в стартовому контурі
    byte matrix[][]; // бінарна матриця для зображення обєкта
   boolean start = false; //наявність-відсутність обєкту на зображенні
    int kilk_aprox_line;
```

В даному класі реалізовані функції скелетизації області на основі проходження контурною лінією, встановлення ваг скелетних точок, функції визначення вузлових та кінцевих точок, метод обєднання окремих частин скелета в моноскелет шляхом нарощування та стоншення елементів скелету, метод виділення ребер скелету на основі зв'язування сусідніх вузлових точок, метод обчислення ваг ребер на основі сумування ваг всіх точок, що належать даному ребру, мтод відновлення області на основі відомого скелету, метод відсікання малоінформативних ребер скелету. Окрім того середатрибутів даного класу слід відмітити: масив для зберігання скелету та самого вхідного зображення, масив для зберігання посдідовності апроксимуючих прямих, відомості про площу та периметр виділеної області тощо.

Окрім того в конструкторі даного класу одразу відуваєть попередня обробка зображення та його конертація в відтінки сірого.

```
    public Skelet (Image Ximage){
        Height = (int) Ximage.getHeight();
       Width  = (int) Ximage.getWidth();
        BufferedImage                temp_image                =
SwingFXUtils.fromFXImage(Ximage,null);
       matrix = new byte [Width][Height];
       for (int j = 0; j < Height; j++){
           for (int i = 0; i < Width; i++)  {
```

```
                int rgb = temp_image.getRGB(i, j) & 0x000000FF;
//перевіряємо тільки по одному каналу (зображення бінарне). Для
кольору, просто вставити можливий діапазон
                if (rgb > 10) { //ставимо 10, оскільки появляються
незначні похибки при конвертації JPG, можна і менше
                    matrix[i][j] = 0;
                } else {
                    matrix[i][j] = 1;
                }}}}
```

Дана реалізація окремих функції а також цілого класу в цілому дозволяли реалізувати на базі розробленого класу програмну систему обробки цифрових зображень та провести її тестування.

3.3 Тестування та аналіз реалізованої системи

Для тестування програмної системи обробки цифрових зображень було використано робочу станцію з такими технічними характеристиками як:

- корпус Zaerwerlman Z1 Black + блок живлення Chieftec APS-43SB;
- HDD WD 500GB;
- Asus PH-GTX1060-3G;
- ОЗУ 4Gb 2400GHz ;
- AMD Ryzen 3 2200G BOX 120;
- материнська плата Asus B350M-E 90.

Характеристики монітора:

- дисплея 19";
- роздільна здатність 1920 x 1080;
- матриця TN;
- частота оновлення 65 Гц;
- інтерфейси HDMI;
- відношення сторін 16: 9.

Технічні характеристики обраної робочої станції є цілком достатніми для проведення процесу тестування розробленої програмної системи обробки цифрових зображень та отримання аналітичних даних про коректність роботи, швидкість аналізу та опису вхідних зображень.

При проведнні тестування було проведено ряд тестів, на основі трьох різних груп областей:

− правильні області – дані області мають форму блиську до простих геометричних фігур, а отже їх контурні лінії вже наближені до апроксимоваих, що значно спрощеє процес отримання їх скелету;

− складні області – до областей даного типу було віднесено області, що мають в своїй контурній лінії елементи природніх об’єктів (деяку варіативність), що може повпливати на процес апроксимації та подальшої обробки зображення;

− області, що описують природні об’єкти – області, що мають складну контурну лінію з великою кількістю змін напрямку та знаку її приросту. При апроксимації важливу роль відіграє коофіцієнт точності, чим він блище до 0, тим точнішим буде опис контуру, проте зросте кількіст апроксимуючих прямих.

Для початку роботи програмної системи необхідно встановити параметри роботи та параметри обробки зображень (рисунок 3.5).

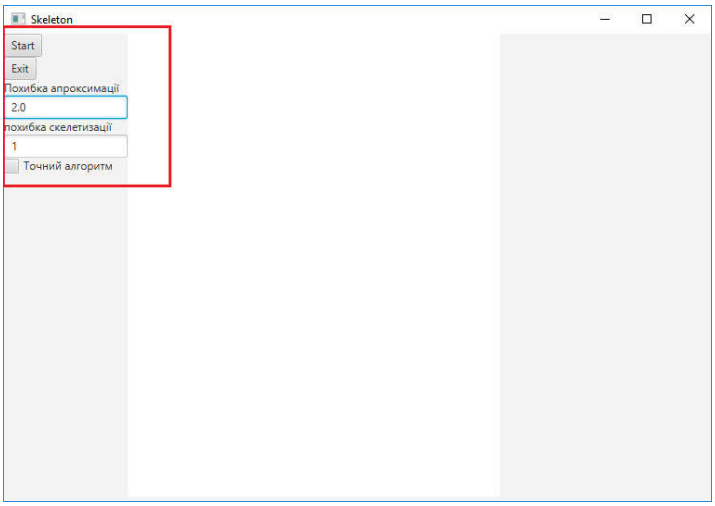


Рисунок 3.5 – Встановлення початкових параметрі роботи системи

Після встановленн параметрів роботи системи, необхідно завантажити вхідне зображення, для цього необхідно активувати відповідний пункт меню, та обрати зображення, що потребує обробки (рисунок 3.6).

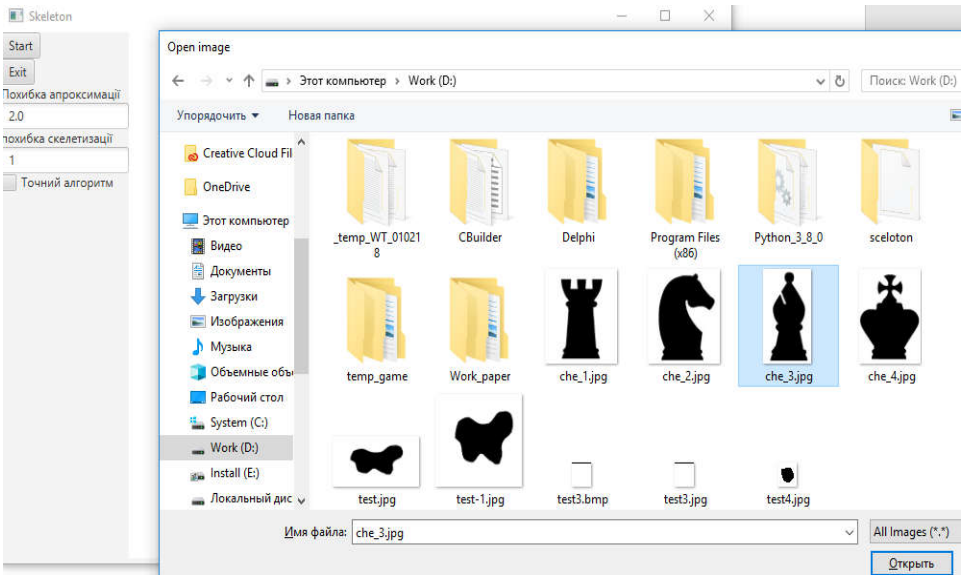


Рисунок 3.6 – Завантаження бінарного зображення в прогрмну систему

Після отримання зображнння програмна система виконае перший етап обробки зображення, а саме проведе процедуру сегментації, виділення контуру та апроксимації виділеної області (рисунок 3.7).

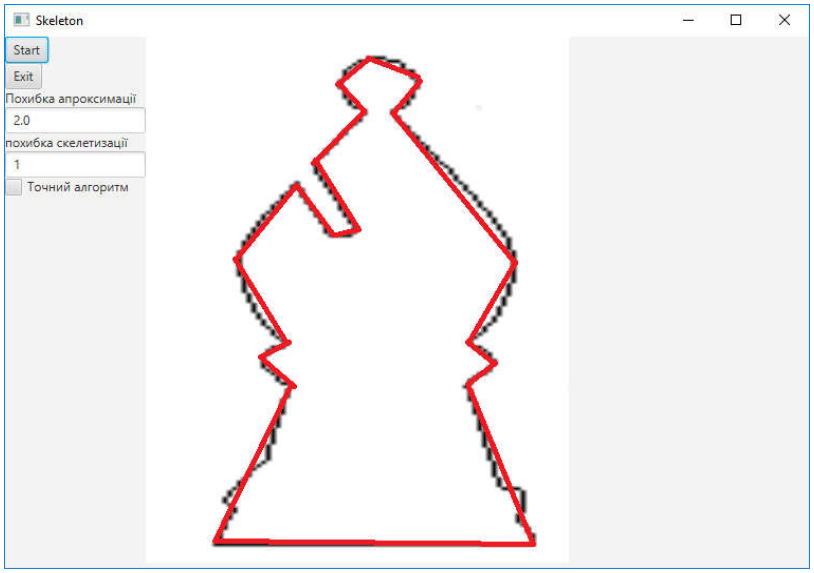


Рисунок 3.7 – Приклад апроксимації вхідної області

На наступному етапі відувається процес скелетизації виділеної області, проте на даному етапі скелет схожий на набір пікселів, тому одя отримання якісного скелета необхідна додаткова обробка зображення (рисунок 3.8).

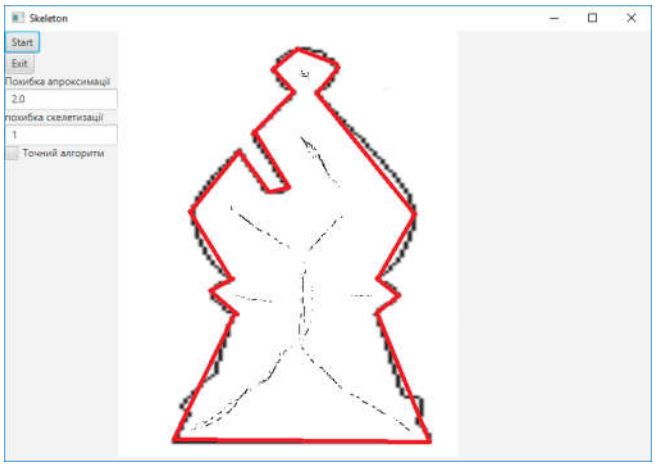


Рисунок 3.8 – Область з виділеними скелетними точками

На останньому етапі обробки вхідного зображення, програмна система проводить формування моно скелета та відсікання малоінформативних ребер скелету, в результаті роботи програмної ситеми було отримано осьтакий результат, що представлено на рисунку 3.9.

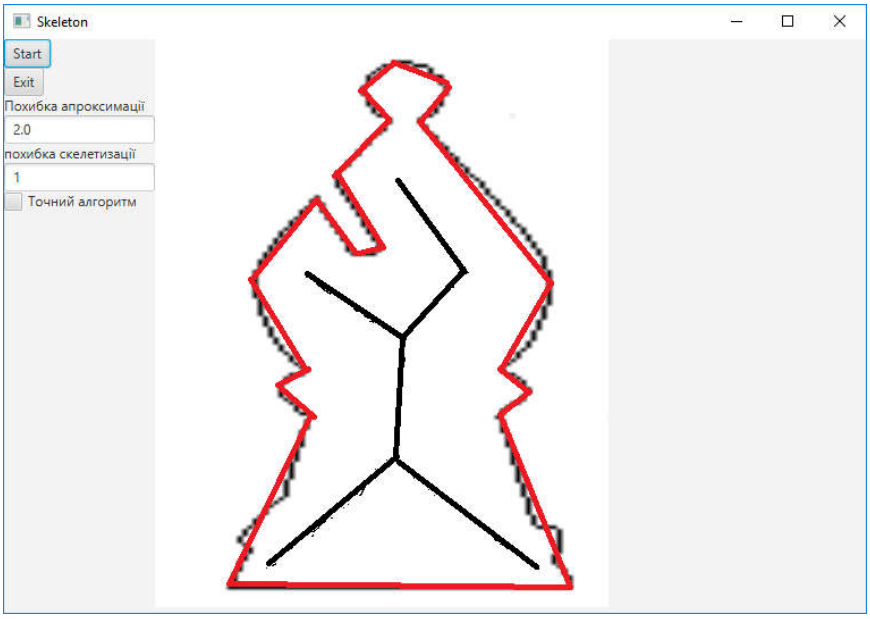


Рисунок 3.9 – Виділений скелет на вхідному зображенні

Після проведення тестування, результати роботи програмної системи були погруповані в групи в залежності від того, який тип зображення було передано на вхід. Під час роботи оцінювався відсоток формування моноскелетів таможливість відновлення області на основі отриманого скелета. Результати тестування наведено в таблиці 3.1.

Таблиця 3.1 – Узагальнені результати роботи програмної системи

	Правильні області	Складні області	Області, що описують природні об’єкти
Бінарні зображення	100%	>98%	>97%
Напівтонові зображення	100%	>95%	>91%
Палітрові зображення	>95%	>84%	>72%
Зображення з природнім кольром	>80%	>76%	>61%

Після отримання даних під час проведеного тестування можна зробити висновок, що на результат роботи значною мірою впливає тип вхідного зображення, оскільки від цього залежить результат процедури сегментації, що в свою чегру має вплив на подальші результати роботи програмної системи. При умові, якщо отримані якісно виділено область з чітким контуром, то результати скелетизації будуть на високому рівні.

ВИСНОВКИ

На основі аналізу сучасних програмних систем обробки цифрових зображень та аналізу алгоритмів скелетизації об’єктів на зображенні можна зробити наступні висновки:

1. Проведено комплексний аналіз задач цифрової обробки зображень, що дозволило виділити основні сфери їх застосування та основні архітектурні елементи;
2. Досліджено існуючі методи та алгоритми попередньої обробки зображень, на прикладі алгоритмів фільтрації, що дало можливість розробити алгоритми приведення зображень у необхідні формати приобробці в програмні системі;
3. Проаналізовано існуючі програмні системи створення, редагування та аналізу цифрових зображень, та досліджено їх структуру, що дозволило виділити основні сфери їх застосування та функціональні модулі;
4. Проведено аналіз методів та алгоримтів контурного аналізу об’єктів на цифрових зображеннях, що дозволило розробити алгоритми виідлення контурів та їх подальшого кодування за допомогою прямих;
5. Розроблено алгоритми скелетизації об’єктів на цифрових зображень на основі вписаних квадратів, що дало моливість реалізувати програмну систему аналізу та опису зображень;
6. Програмно реалізовано систему аналізу та опису об’єктів на цифрових зображень, та проведено її тестування й порівняння з програмами аналогами.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Баженова И.Ю. Язык программирования С++ АО "Диалог-МИФИ", 1997. 366с.

2. Бартлетт Н. Программирование на Delphi Путеводитель. The Coriolis Group,Inc.,1996, Издательство НИПФ "ДиаСофт Лтд.",1996. 116с.

3. Вебер Дж. Технология С++в подлиннике. QUE Corporation, 1996, "BHV-Санкт-Петербург",1997. 256с.

4. Волш А. И. Основы программирования на С++для World Wide Web. IDG Books Worldwide,Inc.,1996, Издательство "Диалектика",1996. 458с.

5. Марков А. С. «Базы данных. Введение в теорию и методологию. Финансы и статистика». 2006. С.24-35.

6. Абрамов С. А. Задачи по программированию. М.: Наука, 1988. 256с.

7. Березин Б.И., Начальный курс Delphi. М.: ДИАЛОГ-МИФИ, 1996. 331с.

8. Бондарев В.М. Основы программирования. Харьков: Фолио, Ростов н/Д: Феникс, 1997. 446с.

9. Вирт Н. Алгоритмы и структуры данных. М.: Мир, 1989. 345с.

10. Гладков В. П. Задачи по информатике на вступительном экзамене в вуз и их решения: Учебное пособие. Пермь: Перм. техн. ун-т, 1994. 516с.

11. Грогоно П. Программирование на языке Delphi. М.: Мир, 1982. 216с.

12. Дагене В.А. 100 задач по программированию. М.: Просвещение, 1993. 106с.

13. Джамса К. Библиотека программиста Java. Jamsa Press, 1996, ООО "Попурри", 1996. 656с.

14. Марков А. С. «Базы данных. Введение в теорию и методологію. Финансы и статистика». 2006. Р. 24-35.

15. Заварыкин В.М. Основы информатики и вычислительной техники. М.: Просвещение, 1989. 556с.

16. Касаткин В. Н. Информация. Алгоритмы. ЭВМ. М.: Просвещение, 1991. 219с.

17. Кен А. Язык программирования Delphi. Addison-Wesley Longman,U.S.A.,1996, Издательство "Питер-Пресс", 1997. 378с.

18. Керниган Б. Язык программирования Delphi. Пер. с англ. М.: Финансы и статистика, 1992. 391с.

19. Ляхович В.Ф. Руководство к решению задач по основам информатики и вычислительной техники. М.: Высшая школа, 1994. 127с.

20. Мейнджер Дж. Delphi Основы программирования. McGraw-Hill,Inc.,1996, Издательская группа BHV, Киев,1997. 346с.

21. Миков А. И. Информатика. Введение в компьютерные науки. Пермь: Изд-во ПГУ, 1998. 442с.

22. Могилев А. В. Информатика: Учеб. пособие для студ. пед. Вузов. М.: Изд. центр «Академия», 1999. 629с.

23. Нотон П. JAVA:Справ.руководство. М.:БИНОМ:Восточ.Кн.Компания, 1996: Восточ.Кн.Компания. 447с.

24. Нотон П. Полный справочник по Java. McGraw-Hill,1997, Издательство "Диалектика",1997. 556с.

25. Ренеган Э.Дж. 1001 адрес WEB для программистов :Новейший путеводитель программиста по ресурсам World Wide Web:Пер.. Минск:Попурри,1997. 512с.ил.

26. Родли Дж. Создание Java-апплетов. The Coriolis Group,Inc.,1996, Издательство НИПФ "ДиаСофт Лтд.",1996. 466с.

27. Секреты программирования для Internet на Java. Ventana Press, Ventana Communications Group, U.S.A.,1996, Издательство "Питер Пресс", 1997. 396с.

28. Семакина И. Г. Информатика. Задачник-практикум: В 2 т.. М.: Лаборатория Базовых Знаний, 1999. 476с.

29. Сокольский М.В. Все об Intranet и Internet. М.:Элиот,1998. 254с.

30. Тассел Д. Стиль, разработка, эффективность, отладка и испытание программ. М.: Мир, 1981. 56с.

31. Тюрин Ю.Н. Анализ данных на компьютере. М.: ИНФРА-М, Финансы и статистика, 1995. 384с.

32. Флэнэген Д. Java in a Nutshell. O'Reilly & Associates, Inc., 1997, Издательская группа BHV, Киев, 1998. 473с.

33. Чен М.С. Программирование на С++:1001 совет:Наиболее полное руководство по Java и Visual J++ :Пер.с англ. Минск:Попурри,1997. 640с.ил.

34. Эферган М. С++: справочник. QUE Corporation, 1997, Издательство "Питер Ком", 1998. 256с.

35. G. Yang. «Human face detection in a complex background. Pattern Recognition », 27 (1): 1994. P.53-63.

36. Kotropoulos C. «Acoustics, Speech, and Signal Processing», 1997. ICASSP-97, 1997. IEEE International Conference on pp.2537-2540 v. 4

37. Leung TK. «Finding Faces in Cluttered Scenes Using Random Labeled Graph Matching» 2005.p P.83-95.

38. Yow KC. Feature-based human face detection. Image and vision computing 15 (9), 1997. P.713-735.

39. Sinha, P. Perceiving and Recognizing threedimensional forms. PhD thesis, Massachusetts Institue of Technology, 1996. 278p.

40. Lanitis, A «Image Anal. Classifying variable objects using a flexible shape model »Image Processing and its Applications, 1995., P.70-74.

41. Viola P. «Rapid Object Detection using a Boosted Cascade of Simple Features», proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2001), 2001., vol. 1, 518p.

42. Jones MJ. «Robust real-time face detection», International Journal of Computer Vision, vol. 57, no. 2, 2004., P.137-154.

43. Buchatskiy AN. «Selection of the Optimal Color Space for Reducing False Positives Rate in the Viola-Jones Method», Актуальні проблеми інфотелекомунікацій в науці та освіті, II Міжнародна науково-технічна та науково-методична конференція. Санкт-Петербург, 2013.

44. Ethan R. ORB: an efficient alternative to SIFT or SURF. Computer Vision (ICCV), IEEE International Conference on. IEEE, 2011. P. 2564–2571.

45. Stefan L. BRISK: Binary Robust Invariant Scalable Keypoints. Computer Vision (ICCV), 2011. P. 2548–2555.

46. Pablo F. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In British Machine Vision Conference (BMVC), 2013.

47. Martin A. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. Of the ACM24: 2001. P.381–395,.

48. Патин М.В. Сравнительный анализ методов поиска особых точек и дескрипторов при группировке изображений по схожему содержанию. Молодой ученый. 2016. №11. С. 214-221.

49. Гонсалес Р. Цифровая обработка изображений в среде MATLAB //М.: Техносфера. 2006. 616с.

50. Методичні рекомендації до виконання дипломної роботи з освітньо-кваліфікаційного рівня “Магістр”. Спеціальність „Комп’ютерні системи та мережі”/О.М. Березький, Л.О. Дубчак, Г.М. Мельник /Під ред. О.М. Березького Тернопіль: ТНЕУ, 2018. 41с.