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

ГОРДІЙ Ігор Володимирович

Алгоритми розпізнавання об'єктів на цифрових супутникових зображеннях / Algorithms for object recognition of digital satellite images

спеціальність: 123 - Комп’ютерна інженерія
магістерська програма - Комп’ютерна інженерія

Магістерська робота

Виконав студент групи КІзм-21
І. В. Гордій

Науковий керівник:
к.т.н., Г. Мельник

Магістерську роботу допущено до захисту:

ТЕРНОПІЛЬ -2018

РЕЗЮМЕ

Магістерська робота на тему «Алгоритми розпізнавання об'єктів на цифрових супутникових зображеннях» зі спеціальності 123 «Комп'ютерна інженерія» написана обсягом 88 сторінки і містить 18 ілюстрацій, 6 таблиць, 3 додатки та 50 джерел за переліком посилань.

Метою роботи є розроблення алгоритму аналізу супутникових зображень, який сегментує зображення на текстурні області і визначає тип кожної області.

Методи дослідження включають методи: теорію алгоритмів і систем, теорію обробки цифрових сигналів, методи аналізу зображень, методи об'єктно-орієнтованого програмування.

Двокроковий підхід до класифікації текстури дозволив збільшити ефективність роботи зі змішаною базою текстур, що складається з кольорових зразків і зразків в градаціях сірого та уніфікувати роботу алгоритму для користувача бібліотек текстур.

Розроблено алгоритм, на основі якого написана програма, що дозволяє аналізувати зображення на предмет визначення представлених на них типів місцевості. Було проведено тестування її роботи на кількох знімках. В результаті проведених експериментів було встановлено, що точність класифікації зросла на 3-9%.

Перевагою розробленого алгоритму є інформаційна ємність отриманих за допомогою нього результатів обробки. Інформація про кожну окрему область дозволяє виділити її характерні особливості, на основі чого в подальшому можлива побудова 3D-моделі місцевості.

КЛЮЧОВІ СЛОВА: СУПУТНИКОВЕ ЗОБРАЖЕННЯ, ТЕКСТУРА, НАРОЩУВАННЯ ОБЛАСТЕЙ.

RESUME

Master's thesis on the topic «Algorithms for object recognition of digital satellite images» from the specialty 123 «Computer engineering». Thesis contains 88 pages, 18 figures, 6 tables, 3 appenixes and 50 references sources.

The purpose of the work is to develop a satellite image analysis algorithm that segments the image into a texture area and determines the type of each region.

Methods of research include methods: the theory of algorithms and systems, the theory of digital signal processing, image analysis methods, methods of object-oriented programming.

The divorced approach to texture classification has allowed to increase the efficiency of working with a mixed texture base, consisting of color samples and samples in grayscale, and unifying the algorithm work for user libraries of textures.

An algorithm is developed, on the basis of which a program is written that allows analyzing images for the purpose of determining the types of terrain represented on them. It tested her work on several photos. As a result of the experiments, it was found that the accuracy of the classification increased by 3-9%.

The advantage of the developed algorithm is the information capacity of the processing results obtained by it. Information about each separate area allows to distinguish its characteristic features, on the basis of which in the future it is possible to construct 3D-model of terrain.

KEYWORDS: SATELLITE IMAGES, TEXTURE, REGION GROWING.

ЗМІСТ

ВСТУП

Актуальність теми. В останнє десятиліття для України важливу роль набули супутникові дистанційні методи дослідження її території. Це пов’язано як з подальшим удосконалюванням космічної техніки, так і зі згортанням авіаційних і наземних методів моніторингу. Особливо велике значення супутникові методи мають для півдня України.

Основні області застосування супутникового дистанційного зондування - одержання інформації про стан навколишнього середовища й землекористування, вивчення рослинних співтовариств, оцінка врожаю сільськогосподарських культур, оцінка наслідків стихійних лих: повеней, землетрусів, виверженьвулканів, лісових пожеж. Засоби дистанційного зондування ефективні при вивченні забруднення ґрунту й водойм, льодів на суші й на воді, в океанології. Ці засоби дозволяють одержувати відомості про стан атмосфери, у тому числі в глобальному масштабі.

Дані зондування надходять у вигляді зображень, як правило, у цифровій формі, обробка ведеться на комп’ютері, тому проблематика дистанційного зондування тісно пов'язана із цифровою обробкою зображень.

Існуючі підходи для виявлення об'єктів із зображень дистанційного зондування, як правило, припускають, що розташування об’єктів вже відоме або визначається вручну. Тому актуальною задача розроблення автоматичного і швидкого методу для виявлення об'єктів на супутниковому знімку великого розміру, який є передумовою для детального розпізнавання об'єктів.

Мета і завдання дослідження. Метою роботи є розроблення алгоритму аналізу супутникових зображень, який сегментує зображення на текстурні області і визначає тип кожної області.

Об’єкт дослідження – процес аналізу цифрових супутникових зображень.

Предмет дослідження – методи і алгоритми сегментації та класифікації супутникових зображень.

Для досягнення поставленої мети необхідно розв’язати такі задачі:

- проаналізувати системи аерокосмічного моніторингу;
- проаналізувати методи та алгоритми структурного аналізу зображень;
- розробити алгоритм нарощування областей;
- розробити алгоритм класифікації текстури;
- розробити структуру та компоненти програмного забезпечення;
- провести експериментальне дослідження розроблених алгоритмів.

Методи досліджень базуються на використанні методів гістограмного аналізу зображень, методів комп'ютерного зору, положень теорії алгоритмів і аналітичної геометрії.

Наукова новизна одержаних результатів**.** Розроблено алгоритми класифікації текстури супутникових зображень на основі локальних бінарних шаблонів, що дозволило підвищити точність класифікації.

Практичне значення отриманих результатів. Розроблено компоненти програмних систем аналізу супутникових зображень.

Публікації та апробація дипломної роботи. Отримані результати апробовані в межах VII Міжнародної науково-технічної конференції молодих учених та студентів Тернопільського національного технічного університету ім. І. Пулюя та опубліковано одні тези доповіді по темі роботи [1].

Впровадження результатів ДР. Результати роботи планується використати в роботі науково-дослідному інституті інтелектуальних комп'ютерних систем (додаток Б).

Дипломна робота складається із трьох розділів, висновків, списку використаної літератури та додатків. У першому розділі проаналізовано системи аерокосмічного моніторингу та методи структурного аналізу зображень.

Другий розділ присвячений розробці алгоритму нарощування областей та дворового алгоритму класифікації текстури.

У третьому розділі здійснено програмну реалізацію розроблених алгоритмів та компонентів системи аналізу супутникових зображень. Проведено експериментальне дослідження алгоритму класифікації текстури.

1 АНАЛІЗ СУПУТНИКОВИХ ЗОБРАЖЕНЬ

1.1 Системи аерокосмічного моніторингу

Система аерокосмічного моніторингу передбачає включення основних технологічних етапів: одержання знімка об'єкта дослідження; фотограмметричну обробку, дешифрування; створення цифрових карт за даними аерокосмічного моніторингу; інтерпретація, оцінка й прогноз розвитку ситуації, в окремих випадках - підтримка прийняття рішень [3].

Необхідна для досліджень інформація (предметно-змістовна й геометрична) отримана зі знімків двома основними методами, це дешифрування й фотограмметричні вимірювання. Дешифрування дозволяє одержувати предметну, тематичну (в основному якісну) інформацію про досліджуваний об'єкт або процес, його зв'язки з навколишніми об'єктами. Фотограмметрична обробка (вимірювання) показує місцезнаходження об'єкта і його геометричні характеристики*:* розмір, форму. Для цього виконується трансформування знімків, їх зображення приводиться в певну картографічну проекцію. Це дозволяє визначати по знімках положення об'єктів і їх зміну в часі [4, 5].

У існуючих системах обробки й розпізнавання аерокосмічних даних передбачається реалізація наступних етапів: прив'язка зображення, його геометрична корекція, дешифрування, створення векторних шарів за результатами дешифрування, при необхідності створення цифрових карт по аерокосмічних знімках, аналіз зміни об'єктів території в часі по повторних знімках.

Основу інформаційного забезпечення аерокосмічного моніторингу територій повинні становити аерокосмічні знімки різного масштабу й роздільної здатності, що містять нові відомості про зміну території і які забезпечують можливість просторово-часової екстраполяції даних локальних спостережень [4].

Процес дешифрування й фотограмметричної обробки необхідно максимально автоматизувати. Якщо трансформування й прив'язку знімків можна

повністю проводити з використанням автоматизованих систем обробки зображень, процес дешифрування досить складний як сукупність ступенів пізнання від виявлення до визначення характеристик об'єкта, і може бути автоматизований частково.

Застосування фотографічної, оптичної, оптико-електронної фільтрації дозволить виділити досліджуваних об'єктів на фоні зашумленого інформаційного зображення й виготовлення синтезованих, кольороподілених, кольорокодованих матеріалів дистанційного зондування. Ці перетворення первинної інформації забезпечують виявлення принципово нових відомостей, що містяться в первинних матеріалах, але, що не виявляються при їхньому візуальному дешифруванні.

Методи фотографічної й оптичної обробки дозволяють змінити масштаб, тональність і розподіл кольорів у зображенні, а оптико-електронної і цифрової обробки - покращити контрастність і фактичну роздільну здатність фотознімків. Можна також використовувати наступні групи методів попередньої обробки знімків: перетворення яскравості, моделювання властивостей ока, підкреслення країв зображення й псевдоколірні методи (дискретне колірне кодування), усунення загального фону й підвищення контрасту [4].

Аерокосмічні зображення, отримані при екологічному моніторингу, у переважній більшості є текстурними. Текстуру, якщо вона розкладена, можна описувати у відповідності з двома головними вимірами. Перший вимір відноситься до тонових складових непохідних елементів текстури і локальним ознакам. Другий вимір пов'язаний із просторовим розташуванням тонових непохідних елементів, тобто із просторовою взаємодією й взаємозалежністю непохідних елементів. Тонові непохідні елементи - це області зображення з певними тоновими ознаками. Оскільки текстура - просторова властивість, вимірювання її ознак повинно бути обмежене областями, що володіють відносною однорідністю. Таку область можна охарактеризувати її площею й формою.

Відомі різні підходи до виміру й опису текстури зображення - статистичні, геометричні, структурні. Важлива властивість багатьох текстур - повторюваний

характер розташування текстурних елементів у зображенні. На якісному рівні текстуру можна описати в такий спосіб: грубозерниста, дрібнозерниста, гранульована, гладка, безладна, лінійчата, строката, нерегулярна, горбкувата. Кожна з них виражається ознаками тонових непохідних елементів і/або просторової взаємодії між ними [2].

Сегментація є найбільш критичною процедурою процесу автоматизації аналізу зображень, оскільки її результати впливають надалі на всі наступні дії, пов'язані з аналізом зображення: представлення виділених об'єктів і їх текстовий опис, вимір ознак, а також інші задачі більш високого рівня. Сегментовані по кольорах і текстурі шари зберігаються у векторному форматі. Векторизація цифрових аерокосмічних зображень раніше виконувалася операторами-дешифровщиками вручну, при поелементному «окресленні» об'єктів і являла собою тривалий і трудомісткий процес.

Для автоматизації процесу розпізнавання об'єктів можна використовувати в сукупності з ознаками кольору й текстури об'єкта також ознаку форми. Форма зображення - це основний пряма дешифровочна ознака, по якій установлюються наявність об'єкта і його властивості. Геометрично певна форма служить надійним дешифровочною ознакою і відноситься переважно до штучних споруджень. Невизначена форма характерна для багатьох природних об'єктів площинного типу (лугу, лісу й ін.) і часто не може служити певною дешифровочною ознакою. Для визначення форми об'єктів зручно використовувати метрики форми, а також характерні точки контурів об'єктів зображення.

Для процесу розпізнавання велику роль відіграють еталонні знімки, еталони об'єктів. Це знімки, на яких показаний приклад дешифрування заданого (або схожого) району, тобто визначені колірні характеристики для відомих об'єктів. Для дешифрування по еталонах передбачається використовувати опорні фрагменти еталонних знімків з апріорно відомими колірними характеристиками. Опорні фрагменти являють собою фрагменти кожного основного кольору (або текстури при використанні процедури еквалізації) вихідного зображення, максимальної величини, необхідної для прямого вимірювання їх характеристик.

Геодинамічні процеси розпізнаються по характеру загального розміщення площ і ступені ураженості території тем або іншим процесом, розташуванні вогнищ процесів, стадії процесів і їх зміна в просторі, а при використанні матеріалів повторних зйомок - і в часі. Тому, щоб на основі виявлених відомостей про об'єкти здійснювати прогноз динаміки розвитку явища або зміни об'єкта, необхідна розробка операції просторового аналізу:

– вимірювальні операції, включаючи обчислення довжин відрізків прямих і кривих ліній, обчислення площ, периметрів;

– визначення топологічних характеристик геопростростору;

– побудова карти безперервного розподілу параметрів у вигляді поля із застосуванням інтерполяції значень;

– побудова буферних зон об'єктів карти й оверлея; збереження результатів в окремий шар;

– відображення динаміки зміни значень полів у часі й у просторі, як у вигляді карти, також у вигляді тренда [1].

1.2 Огляд програмного забезпечення

Користувачі мережі Інтернет, усвідомивши переваги, що надаються можливістю аналізу просторових даних, потребують інструменту, що дозволяє здійснювати швидкий і зручний пошук і доступ до цифрових знімків місцевості і іншої географічної інформації, зосередженої в багатьох урядових, комерційних і академічних організаціях [3].

Мережеві пошукові системи грають дуже важливу роль. У Мережі зосереджена така кількість інформації, що її пошук вже перетворюється на окреме завдання і віднімає дуже багато часу. Пошукові сервери видають на запит, окрім необхідної інформації тисячі непотрібних посилань [4].

Багато організацій збирають географічні дані в різній формі, або для свого власного використання, або для продажу в інші організації. Ці організації

створюють центри обміну інформацією, дозволяючи користувачам дістати доступ до цих даних, проте для цього потрібний спеціалізований пошуковий інструментарій. Розглянемо наявні на ринку модулі пошуку метаданих.

MapInfo Metadata Browser (MMDB) - це інтелектуальний пошуковий клієнт, створений для користувачів просторових даних. MMDB дозволяє користувачам збирати інформацію про наявність просторових даних, що надається різними центрами обміну просторовою інформацією, а також порівнювати і аналізувати отримані метадані. За допомогою MMDB можна формулювати запити, що стосуються просторових даних, таких як: Чи існують такі дані? Де є необхідні дані і як їх можна придбати? Чи відповідають ці дані моїм вимогам? [5-16]. На рисунку 1.1 наведено головне вікно програмного засобу. MMDB виконує наступні функції:

– побудова запиту, заснованого на списку атрибутів метаданих, відповідних стандартам FGDC;

– «сканування» Інтернету для знаходження метаданих, що відповідають запиту користувача;

– підтримку картографічного інтерфейсу для вибору території, на яку потрібні просторові дані;

– багаточисельні функції аналізу і уточнення результатів пошуку метаданих;

– роботу на різних обчислювальних платформах завдяки реалізації на Java;

– можливість налаштування під користувача;

– дружній інтерфейс і контекстно-чутлива Довідкова система;

– дозволяє обійти обмеження поточного стандарту FGDC по пошуку метаданих;

– дозволяє підтримувати одночасний доступ до необмеженої кількості центрів обміну інформацією. MMDB дозволяє бачити і аналізувати перебування запиту на будь-якій стадії пошуку просторових метаданих;

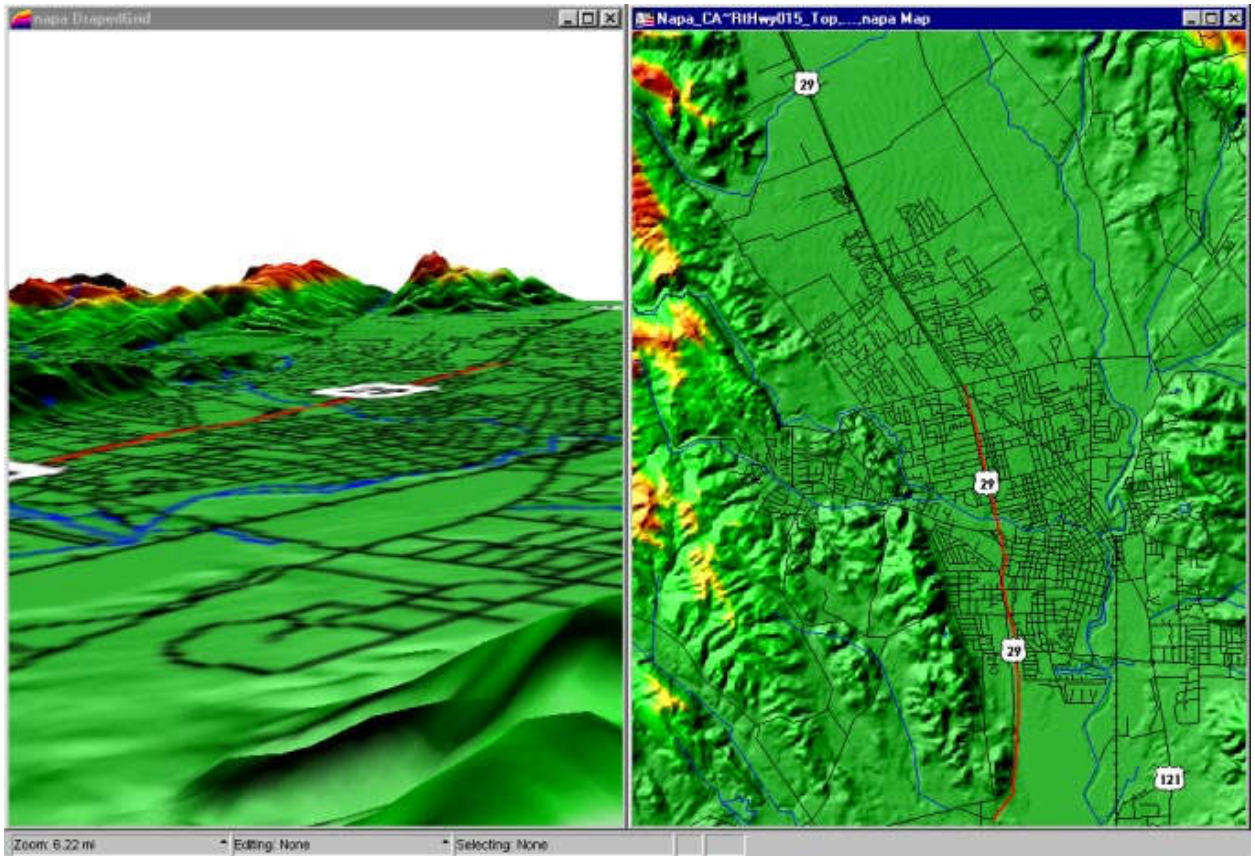


Рисунок 1.1 – Головне вікно програми MapInfo Metadata Browser

– дозволяє паралельно обробляти до десяти запитів.

До недоліків системи можна віднести наступні:

– працює лише як клієнтське застосування, тобто якщо групі користувачів потрібно отримати якісь дані, то кожен з них запускає на своєму комп'ютері Metadata Browser, і отримує дані, але при цьому відбувається дублювання роботи, і завантажується мережевий трафік;

– здійснює лише пошук метаданих, функції редагування метаданих відсутні;

– список серверів даних не поповнюємо;

– може працювати лише з тими серверами даних, на яких встановлено спеціальне програмне забезпечення;

Система ESRI ARCCatalog надає інструменти для пошуку і перегляду географічних даних, створення і перегляду метаданих, швидкого перегляду будь-якого набору даних, а також інструменти для структуризації географічних даних.

Набір метаданих створюється в програмі ARCGIS ArcCatalog і публікується на сервері. Опубліковані метадані можуть бути легко надані іншим

користувачами системи [16]. ArcCatalog в системі ARCGIS Desktop виконує, по суті, ті ж функції, що Провідник (Windows Explorer) в середовищі Windows (рисунок 1.2). Основна відмінність полягає в тому, що Провідник Windows працює зі всіма даними, присутніми в системі (на комп'ютері, в локальній мережі), а ArcCatalog спеціалізується лише на географічних даних. Причому, що розуміти під географічними даними, можна визначити використовуючи спеціальний діалог налаштування ArcCatalog.

Можна самостійно формувати дерево Каталогу. Такі функції, як Підключитися до каталогу і Відключитися від каталогу, дозволяють додати в Каталог лише ті каталоги, які буде потрібно користувачеві для роботи. Окрім підключення до тек, в ArcCatalog можливо підключатися до баз даних і Інтернет-серверів. Каталог - це своєрідне представлення карт і географічних даних, що зберігаються на локальних і мережевих дисках.

Підключення до каталогу вказує на кореневу директорію вашого локального диску або на вибрану директорію диску, доступного в мережі. У Каталозі відображуються всі теки, що знаходяться усередині підключеної теки. Якщо директорія не містить географічних даних, вона відображається звичайним значком, як в провіднику Windows. Каталоги, що містять географічні дані, такі як бази геоданих, покриття, шейп-файли, файли САПР і пов'язані з ними файли, відображаються спеціальним значком. Причому, географічні дані певного формату відображаються власними спеціальними значками.

За замовчуванням в ArcCatalog до географічних відносяться дані наступних форматів: база геоданих і весь її вміст (класи просторових об'єктів, растрові набори даних, класи відношень, геометричні мережі і так далі), шейп-файли, покриття, файли САПР, документи і шаблони карт, шари, растрові зображення декількох форматів.

Всі функції модуля ArcCatalog можна розділити на декілька груп: перегляд і пошук географічної інформації, функції роботи з метаданими, побудова структури даних.

ArcCatalog дозволяє швидко проглядати доступні дані. Є декілька режимів відображення даних: крупні значки, список, деталі (список з вказівкою типу

даних) і зразки. Окрім цього, за допомогою закладки Перегляд можна проглядати вміст структурних одиниць просторових даних (шейп-файлів, покриттів, наборів класів і так далі), не відкриваючи їх у переглядачі (наприклад, в ArcMap). На закладці Перегляд є два режими перегляду: Географія і Таблиця. Перемикаючись між режимами, можна працювати і з просторовими, і з атрибутивними даними.

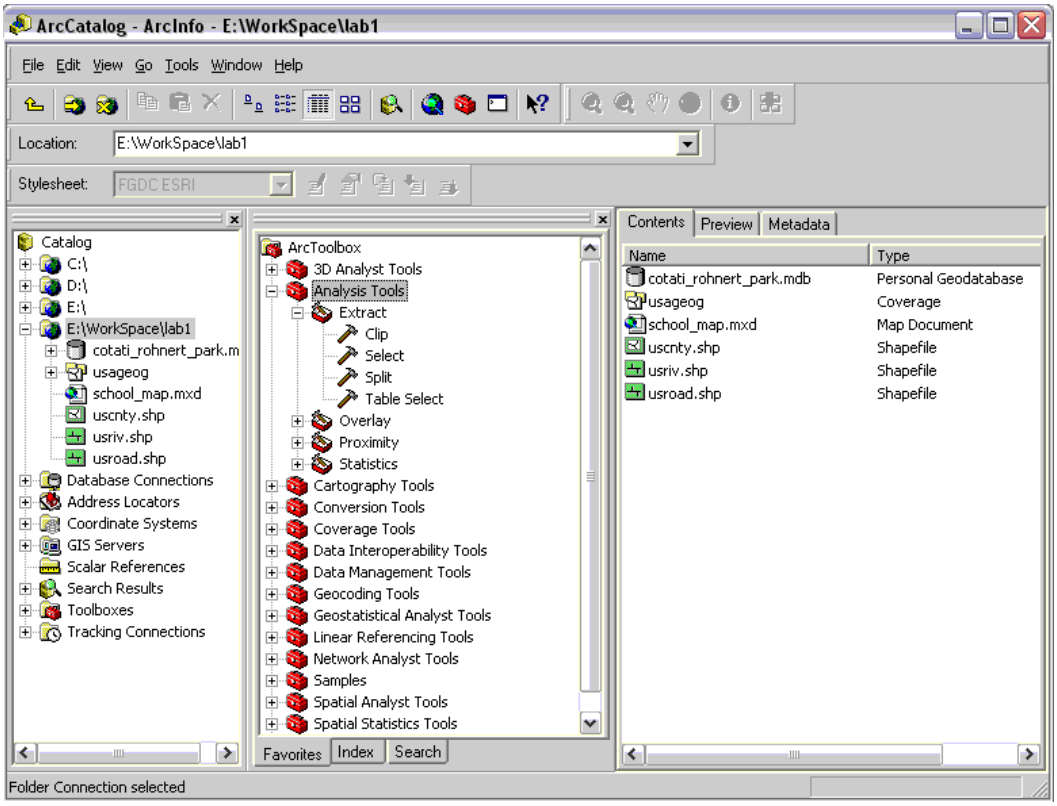


Рисунок 1.2 – Вікно програми ArcCatalog

Функція Пошук дозволяє швидко знаходити потрібні дані, як по їх географічних ознаках, так і по системних параметрах (наприклад: місце розташування, дата, тип даних).

У ArcCatalog є набір важливих інструментів, що дозволяють підвищити ефективність роботи і віддачу від наявних даних, - інструменти для роботи з метаданими. Метадані - це «дані про дані». Вони створюються відповідно до існуючих стандартів і містять інформацію, що дозволяє чітко ідентифікувати дані, починаючи від назви організації, що володіє даними, і закінчуючи географічними координатами ділянки покритої даними території. Метадані

полегшують управління даними, оскільки вони допомагають швидко орієнтуватися у великих об'ємах інформації. Зокрема, пошук, здійснюваний в ArcCatalog, працює і на основі метаданих: можна вказати значення деяких параметрів метаданих, які враховуватимуться при пошуку. ПЗ ArcCatalog може створювати метадані автоматично, вносячи до них інформацію, що отримується з системи (назва організації, дата створення, екстент даних і так далі). У спеціальному редакторі метаданих можна змінити автоматично створені метадані, внісши додаткову інформацію, таку як призначення даних, коментарі і так далі.

Основна функція ArcCatalog - це створення структури даних. Під структурою даних тут розуміється як структура файлів і каталогів на диску комп'ютера або мережевих дисках, що містять дані ГІС, так і структура бази геоданих. Як відомо, деякі формати даних, такі як шейп-файли або покриття, складаються з декількох окремих файлів. При переміщенні, копіюванні або видаленні цих об'єктів в Провіднику Windows необхідно ретельно відстежувати, щоб переміщалися всі компоненти шейп-файлу або покриття. Одиниці даних, що є фізично декількома файлами на диску, відображуються у вікні ArcCatalog як єдиний об'єкт, і дії над ними проводяться відповідно як над єдиним цілим - всі компоненти створюються, переміщаються або видаляються одночасно. До недоліків даного застосування можна віднести: працює лише як клієнтське застосування, тобто якщо групі користувачів потрібно отримати якісь дані, то кожен з них запускає на своєму комп'ютері Metadata Browser, і отримує дані, але при цьому відбувається дублювання роботи, і завантажується мережевий трафік; може працювати лише з тими серверами даних, на яких встановлено спеціальне програмне забезпечення; працює лише з визначеним форматами даних.

Гібридний редактор Consistance Software Spotlight. Основні можливості:

– зображення складається з растра, на який накладені структурні елементи, це дозволяє редагувати растрові зображення так само легко, як і векторні;

– векторизація сканованих зображень у напівавтоматичному й автоматичному режимах;

– можливість роботи з кольоровими й монохромними вхідними растрами, наявність інструментів і фільтрів, що поліпшують зображення;

– інструменти автокорекції векторних структур.

Серед недоліків програмного продукту можна згадати:

– недостатня ефективність фільтрів, що роблять редукцію кольору (зменшення кількості кольорів);

– недостатня ефективність інструментів по поділу кольорів (зменшення колірних компонентів до однієї тоно- колірної шкали).

ПЗ Spotlight є унікальним продуктом, який відмінно справляється з обробкою штучно створених зображень. Однак він представляє обмежені можливості при роботі з кольоровими зображеннями.

Список програмних продуктів, безумовно, може бути розширений, але все-таки самі характерні й популярні розробки в нього включені.

Серед програмних продуктів, присвячених обробці штучних зображень, можна простежити кілька тенденцій:

–растрові зображення легко одержувати сканометрично, однак складно редагувати;

–векторні зображення набагато простіше редагувати, при цьому складний процес їх одержання з реального світу;

–програмні продукти, що поєднують два підходи, недостатньо добре й ефективно справляються зі структурним аналізом зображень. Допускають багато помилок, на виправлення яких іде майже стільки ж часу, скільки буде потрібно для аналізу зображення вручну.

1.3 Огляд методів структурного аналізу зображень

У багатьох алгоритмах машинного зору і оброблення зображень, для спрощення припускають, що в локальних областях зображення інтенсивність пікселів незмінна. Проте, зображення реальних об'єктів часто не містять області

рівномірного розподілу інтенсивносей. Наприклад, зображення дерев'яної поверхні не однорідне а містить варіації інтенсивності, які формують певний повторюваний зразок (узор), який називається текстурою. Текстура може бути результатом фізичних властивостей поверхні, її опуклість і нерівність, або вона може бути результатом у різниці відбиття світла як наприклад колір.

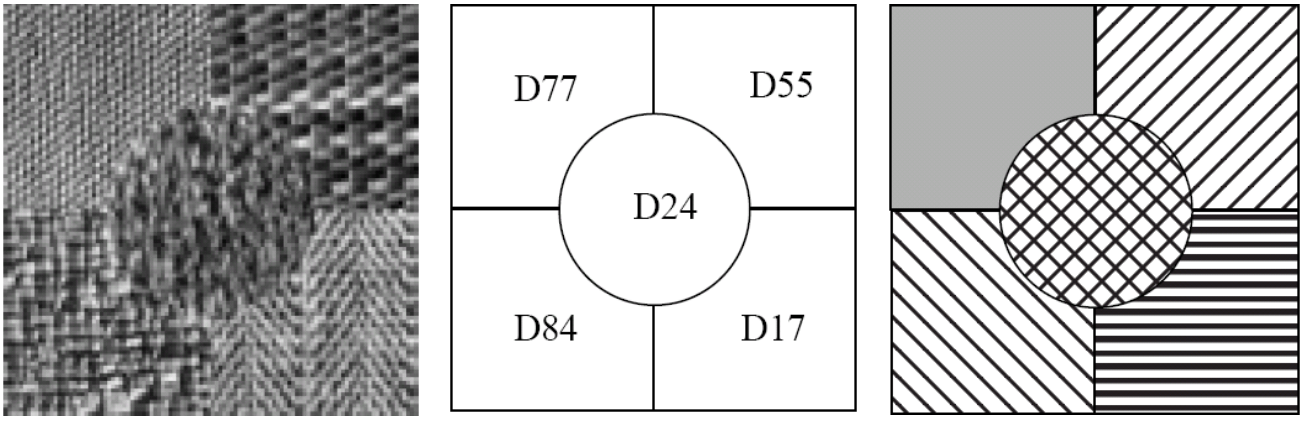
Людина розпізнаємо текстуру, коли бачить, але дати їй визначення важко. Ці труднощі демонструються числом різних визначень текстури, наведених різними дослідниками систем машинного зору. У [1] наведено декілька визначень текстури взятих з літератури про машинний зір:

– "ми можемо розцінювати текстуру як, те що утворює макроскопічну область, а структуру можна описати як повторюваний образ, у якому елементи (примітиви) впорядковані відповідно до правила розміщення";

– "область зображення має постійну текстуру якщо статистичні характеристики області постійні, повільно змінюються, або є приблизно періодичними";

– "текстура визначена як ознака області, що не має ніяких злічуваних елементів".

Текстура зображення [1] – це функція просторової зміни розподілу інтенсивностей пікселів, що використовується для різноманітних задач і є темою інтенсивного вивчення багатьма дослідниками. Одним з безпосередніх застосувань текстури зображення є розпізнавання областей зображення, із використанням їх текстурних властивостей. Наприклад, на рисунку 1.3,а), ми можемо розрізнити п'ять різних текстур, це бавовняна тканина, солом'яне покриття, волокно, ялинковий ткацький узор, і тиснена теляча шкіра. Текстура - найголовніший візуальна ознака для ідентифікації однорідних (гомогенних) областей. Така ідентифікація називається текстурною класифікацією. Мета класифікації текстури – створити карту класифікації вхідного зображення, де кожен однорідна текстурна область ідентифікується з класом текстури, до якого це належить, як показано в рисунку 1.3, б).



а) б) в)

Рисунок 1.3 – Аналіз текстурного зображення

Ми можемо також знайти границі текстурних областей, навіть якщо ми не можемо класифікувати їх. Це друга задача текстурного аналізу – сегментація текстури. Ціль сегментації текстури полягає в тому, щоб одержати карту границь, який показують на риснку 1.3,в).

Аналіз зображення – це виділення із зображення необхідної інформації за допомогою автоматичних систем. Він включає такі етапи: визначення характеристик зображення (колір, текстура, форма/контур), сегментація зображень і групування, представлення і організація ознак. З текстурою зазвичай пов'язують такі основні задачі як сегментація, класифікація, визначення форми по текстурі та її синтез. Хоча загальноприйнятого визначення поняття текстура немає, більшість авторів погоджуються, що текстурне зображення повинно бути просторово однорідне, містити повторювані структури, часто з деякими випадковими змінами. Текстура є відображенням фізичних властивостей поверхні об’єкту і відповідно до психології сприйняття володіє таким властивостями як зернистість, контрастність, напрямленість, лінійна подібність і шершавість [1].

Текстури класифікують за різними критеріями: за закономірностями розміщення елементів на регулярні, близько регулярні, нерегулярні, близько стохастичні, стохастичні, за джерелом походження на природні (фотографічні), штучні.

Можна виділити такі основні напрямки текстурного аналізу зображень: текстурна сегментація і текстурна класифікація. Для текстурної сегментації

існує два основних підходи, що аналогічні методам сегментації зображення: область-орієнтований і контурно-орієнтований. У область-орієнтованому підході, намагаються ідентифікувати області зображення, які мають однорідну текстуру. Пікселі або невеликі локальні області об'єднуються відповідно до подібності текстурних ознак. Контурно-орієнтований підхід базується на виявленні відмінностей в текстурі. Тому границі виявляються там, де є відмінності в текстурі (не потрібно заздалегідь знати число текстурних областей в зображенні).

Текстурна класифікація полягає у віднесенні зображення до певної категорії текстури. Для цього потрібні апріорні знання про відповідні класи. Текстурна класифікація використовується в задачах текстурного оброблення аерокосмічних фотографій, контролі якості та обробленні біомедичних зображень.

Для опису зображення проводиться оцінка контуру та форми сегменту, його кольору, текстури і просторового розташування. Оцінка текстури зображення використовується для сегментації зображення та класифікації сегментів. При описі текстури можна виділити такі її властивості [2]: однорідність, щільність, грубість або зернистість, шершавість, регулярність, лінійність, напрямленість. Така різноманітність ознак зумовлює появу великої кількості моделей текстури. Першим кроком при створенні математичних моделей текстури є опис її ознак. Методи опису можна розділити на статистичні, геометричні, випадкових процесів, фрактальні та методи оброблення сигналів.

Статистичні методи описують просторовий розподіл рівнів сірого зображення. Матриці імовірності розподілу рівнів інтенсивності пов'язані з статистикою другого порядку. У матриці суміжності P_d для вектора зміщення $\boldsymbol{d} = (dx, dy)$ значення елементу $p_{i,j}$ є кількістю входжень пари значень рівнів сірого i та j, які розміщені на відстані d [3]. Кожній точці зображення (x, y) відповідає матриця суміжності P_d, яка характеризує розподіл яскравості в квадраті розміром $W \times W$ з центром в точці (x, y) . Елементи матриці P_d визначаються таким чином:

$$P_d(i,j) = \sum_{(m,n)\in D} f_{i,j}(x_{m,n}; x_{m+d,n+d}),$$

Функція $f_{i,j}(x_{m,n}; x_{m+d,n+d})$ визначає рівні яскравості між точками.

$$f_{i,j}(x_{m,n}; x_{m+d,n+d}) = \begin{cases} 1, (x_{m,n} = i \text{ та } x_{m+d,n+d} = j) \\ \quad \text{або } (x_{m,n} = j \text{ та } x_{m+d,n+d} = i), \\ 0, \text{інакше} \end{cases}$$

де D – квадрат розміром $W \times W$ (W - непарне число),

$i, j = 0..255$ – значення яскравості точок,

$x_{m,n}$ – яскравість точки з координатами (m,n) .

Функція $f_{i,j}(x_{m,n}; x_{m+d,n+d})$ є індикатором того, що точки, які знаходяться на заданій відстані, мають певні рівні яскравості. Параметр d визначає відстань, на якій проводиться аналіз сусідніх точок. На основі матриць суміжності визначаються такі текстурні ознаки як енергія, ентропія, контраст, однорідність, кореляція, затінення. Матриці суміжності використовуються перш за все в задачах класифікації текстури.

Важлива якість багатьох текстур – це повторювана природа розміщення елементів в зображенні. Функція автокореляції зображення може використовуватися, щоб оцінити значення регулярності і зернистості текстури. Передбачається, що зернистість пропорційна розкиду функції автокореляції. Функція автокореляції $A_F(m,n)$ визначається так:

$$A_F(m,n) = \sum_i \sum_j F(i,j)F(i-m, j-n),$$

де (i, j) - координати точки,

m, n - зміщення.

Геометричні методи. Геометрична модель визначає текстуру як таку, що складається з „текстурних елементів”. Якщо текстурні елементи визначені, то

існує два основних підходи до їх аналізу. Перший розраховує статистичні властивості текстурних елементів, які стають ознаками. При другому підході визначають закономірності їх розміщення.

Для визначення текстурного елементу можна використовувати різні способи. Автори статті [3] застосовують для цієї цілі розбивання поверхні на багатокутники Вороного. Геометричні властивості багатокутників Вороного служать текстурними ознаками.

Геометричний метод застосовано у статті [4] для аналізу близько-регулярних текстур. В цій роботі близько-регулярна текстура розглядається як статистично спотворена регулярна текстура з розміщенням елементів у вигляді сітки і можливими варіаціями форми елементів або їх кольору. Запропоновано три групи близько-регулярних текстур в залежності від регулярності кольору і розміщення. Потрібно відмітити, що ступінь спотворення координатної сітки розміщення текстурних елементів (кут між осями) визначається людиною візуально. Текстурні ознаки обчислюються на основі векторів зміщення елементу та зміни їх кольору.

Структурні методи побудови текстурних моделей припускають, що текстури складені з примітивів. Текстура утворюється розміщенням цих примітивів згідно певного правила. Цей клас алгоритмів обмежується регулярними текстурами. Структурний аналіз текстури складається з двох основних етапів: визначення елементів текстури і виведення правила розміщення.

Методи базовані на моделях. Моделі зображення можуть бути використані не тільки для опису текстури, але і для її синтезу. Параметри моделі можуть відтворювати видимі якості текстури.

Випадкові Марківські процеси (ВМП) визначають, що інтенсивність кожного пікселя залежить тільки від його сусідів. ВМП використовуються для текстурної класифікації, сегментації, компресії, відновлення зображення та текстурного синтезу [5-10].

Фрактали. Нерівності багатьох природних поверхонь володіють статистичними властивостями і самоподібністю при різних масштабах. Для

опису самоподібності використовується фрактальна геометрія запропонована Б. Мандельбротом [6]. Шершавість поверхні оцінюється за допомогою обчислення фрактальної розмірності (чим більша фрактальна розмірність, тим шершавіша текстура).

Перші роботи по застосуванню методів оброблення сигналів, а саме, просторових фільтрів були сконцентровані на оцінці кількості контурів (границь на одиницю площі). Фільтри є природнім механізмом пошуку простих шаблонів, оскільки найбільше вони реагують на елементи які подібні на сам фільтр. Таким чином, можна представити текстуру зображення у вигляді реакції набору фільтрів. Набір різних фільтрів повинен складатися із серії узорів – як правило плям і смуг – при певному наборі масштабів. При цьому фільтри плям можуть визначати неорієнтовану структуру, а фільтри смуг краще реагують на орієнтовану. Один із способів отримання потрібного фільтру – це знаходження зваженої різниці гаусових фільтрів при різних масштабах. Кількість масштабів та напрямків підбирають експериментально (кількість масштабів лежить у межах від чотирьох до одинадцяти, число напрямів – від двох до вісімнадцяти). Набір відфільтрованих зображень ще не є описом текстури, оскільки потрібно додатково описати загальний розподіл різних елементів текстури. Для цього збирають статистичні дані виходів певного набору фільтрів, які подають на класифікатор, що описує текстуру. Текстуру можна представити середнім значенням і середньоквадратичним відхиленням виходу фільтрів взятих по вікну і використати результати для побудови характеристичного вектора. Для аналізу і опису текстури за допомогою статистичних даних блоку фільтрів необхідна згортка зображення з декількома фільтрами при декількох масштабах. Для систематичного виконання цієї операції використовуються Гаусова та Лапласова піраміди зображень [7]. В них застосовуються низькочастотні смугові фільтри. Для аналізу просторових частот використовуються орієнтовані фільтри Габора, а для явного опису орієнтації смуг - орієнтовані піраміди. Кожен шар піраміди зображення розкладають на компоненти, які задають енергію при визначеній орієнтації, а кожен компонент в свою чергу є реакцією на орієнтований фільтр [8, 9].

Спектр Фур'є. Аналіз текстур за допомогою спектру Фур’є представляє залежність просторових частот від зернистості текстури. Як показують психофізичні дослідження, людський зір окремо аналізує просторову частоту текстури та її орієнтацію. Для аналізу різних частот використовується багатомасштабний аналіз. Визначення текстурних характеристик відбувається по Фур’є образу зображення. При цьому застосовуються орієнтовані фільтри.

Вейвлет модель базується на дискретному вейвлетному перетворенні (ДВП). Якщо при перетворенні Фур’є функціями розкладу є синусоїди змінної частоти і нескінченної тривалості, то функції розкладу ДВП є "хвилі" змінної частоти і скінченої тривалості. Вейвлет метод використовується для текстурної класифікації [10, 11]. Перевагою даної моделі є інваріантність текстурних ознак до повороту та масштабування.

1.4 Постановка задач дослідження

Завдання виділення об'єктів – одне з перших завдань аналізу цифрових знімків. В роботі більшості алгоритмів виділення об'єктів використовується для виявлення їх контурів. Легко помітити, що контур є границею областей, для внутрішніх елементів яких використовують умову однорідності по будь якій ознаці. У випадку растрових зображень можна сказати, що кожна область - це набір точок (пікселів), що мають однакові властивості, за якими вони і об'єднані. Таким чином, з'явилася можливість виділяти окремий об'єкт по його текстурі.

У текстурах відображені індивідуальні характеристики поверхні, такі як здатність відбивати або розсіювати світло, однорідність. Цю особливість текстур активно використовують в 3D-моделюванні, де спочатку створюють «пластиліновий» макет, всі частини якого однакові, а потім накладають на окремі частини відповідні зображення текстури. Однак це тільки приклад використання текстур для задання властивостей об'єкта. Більш цікава задача текстурного аналізу зображення.

Дві найбільші сфери застосування текстурного аналізу - це медицина і картографія. Медики за допомогою нього здатні виявити пухлину або патології на медичних знімках, а також визначити об’єми тканин і органів. Найчастіше він застосовується для виявлення злоякісних новоутворень на рентгенівських знімках. Однак на сьогоднішній день такий підхід не досить точний для того, щоб повністю на нього покластися і прибрати людський фактор при аналізі рентгенівських знімків. Але він набув широкого поширення в якості допоміжного механізму діагностики.

Стартовим майданчиком для розвитку перших методів аналізу аерофотознімків стала військова сфера. Саме в ній вперше почали масово використовувати алгоритми розпізнавання зображень. Прикладом цього може служити виявлення ворожих позицій. Пізніше завдання виявлення окремих об'єктів переросла в задачу комплексного аналізу фотознімку, після чого, як це часто буває, вона поширилася і на громадянську сферу. На сьогоднішній день аналіз супутникових знімків і аерофотознімків широко використовується в картографії. На жаль, при всьому рівні розвитку технологій, повністю автономного методу, що гарантує точність перетворення фото в карту розроблено не було, але існуючі алгоритми є невід'ємними інструментами, істотно заощаджують час будь-якого сучасного картографа.

У картографії текстурний аналіз є одним із засобів автоматичного створення карти з аерофотознімків. Текстурний аналіз знімка далеко не тривіальна задача. Першим кроком обробки зазвичай являється сегментація фото на області різного типу. Однак потрібно не просто розбити зображення на області, а й виявити аномалії - місця, які явно виділяються з однорідного фону. Крім того, в результаті виявлення області зі схожими властивостями повинні бути і позначені схожим чином. Для цього за допомогою аналізу відбувається виділення характерних властивостей кожної області. Цей аспект задачі і розглядається текстурною класифікацією.

Основною складністю завдання текстурного аналізу аерофотознімків являється підбір методів сегментації і текстурної ідентифікації, враховують

особливості даної предметної області і підбір дескрипторів, які максимально точно описують саме текстури природного походження.

Визначимо основні терміни, необхідні для постановки задачі.

Визначення 1. Будемо називати текстурою зображення, що складається з довільних елементів, що повторюються і відображають будь які візуальні властивості фізичного об'єкта. За типом і розміром примітивів текстуру можна розділити на регулярні, коли структурні елементи текстури розподілені рівномірно, нерегулярні, коли елементи розташовані нерівномірно і фрактальні, коли структурні елементи утворюють фрактал. На рисунку 1.4 показані приклади різних типів текстур.

а) б) в) г)

Рисунок 1.4 – Приклади різних текстур:
а) регулярна, б) нерегулярна фрактальна, в) нерегулярна, г) має ознаки як регулярної, так і нерегулярної текстури.

Визначення 2. Дескриптором текстури називається функція, яка описує текстуру за основними характеристиками. Можна вважати дескриптор текстури альтернативним представленням текстур для якого, можна ввести міру схожості.

Приклади дескриптора:

- словесний опис (назва);
- гістограма;
- багатомірний вектор.

Визначення 3. Сегментацією зображення називається процес розбиття на кілька зв'язаних замкнутих непересічних областей, при цьому зв'язки між елементами всередині однієї області сильніші зв'язків між елементами різних областей.

Визначення 4. Класифікацією текстур називається визначення належності текстури до одного із заданих наперед типів (класів) текстур.

Метою даної роботи є розробка алгоритму аналізу супутникових зображень, який сегментує фото на текстурні області і визначає тип кожної області. Перелік можливих областей задається заздалегідь у вигляді бази текстур. Для позначення кожного класу на зображенні використовується індивідуальний колір або умовні позначення, тому необхідно отримати план місцевості з позначеннями типів місцевості, який може служити одним з інструментів автоматичного створення карти з фотознімку.

На сьогоднішній день існує безліч алгоритмів і навіть програмних пакетів для автоматичної генерації карти або плану місцевості по фото, результатом роботи яких є карта або план місцевості у вигляді растрового або векторного зображення з метаданими і умовними символами. У контексті даної роботи відмінність плану місцевості від карти полягає в масштабі (у карти він значно менше, ніж у плану місцевості) і в точності (лінії доріг, межі будь-яких областей наносяться приблизно, схематично). Однак більша частина метаданих задається користувачем вручну. Розроблюваний в даній роботі алгоритм частково автоматизує процес додавання метаданих. У даній роботі метаданими є тип місцевості кожної області на знімку.

Супутниковим зображенням в контексті даної роботи будемо вважати будь який фотознімок, зроблений з повітря. Таким чином, між аерофотознімком і знімком з супутника відмінностей не має. Можливість використання аерофотознімків нарівні з супутниковими обумовлюється тим, що часто для вирішення поставлених завдань не потрібно настільки великий обсяг даних. Для створення карт міста, району чи окремої місцевості не має сенсу використання величезних знімків супутника. Більш ефективним буде використання локальних аерофотознімків. Очевидним плюсом таких вихідних даних є той факт, що такі знімки можна самому отримати за допомогою будь якого літаючого об'єкта, оснащеного камерою. Однак, для коректного аналізу зображень до них пред’являються вимоги, щоб на них не було хмар і літаючих об'єктів, так як вони є чужорідними тілами, що не належать ландшафту. Варто зауважити, що до

якості знімків жорстких вимог не висувають, так як використовувані алгоритми стійкі до шуму. На рисунку .2 Приведенні приклади аерофотознімків

а) б)

Рисунок 1.5 – Приклади аерофотознімків: а) оброблений супутниковий знімок, б) знімок з літального апарату, зроблений під кутом

Узагальнений алгоритм обробки супутникових зображень.

1.Обробка. Даний етап включає в себе корекцію освітлення для занадто темних або занадто освітлених фотознімків, підвищення контрастності. Цей етап не є обов'язковим, але в деяких випадках значно підвищує точність подальшої обробки.

2.Сегментація на текстурні блоки. Все зображення розбивається на окремі області.

3. Після проведення сегментації можуть виникнути області, площа яких недостатня для проведення класифікації. Тому для обробки необхідно об'єднати кожну область з пов'язаною з нею найбільш схожою областю. У підсумку отримуємо набір областей, розмір кожної з яких, достатній для текстурної класифікації. На цьому етапі належить визначити критерій «прилягання» областей і підібрати мінімальну класифіковану площу.

4.Класифікація текстур кожної області. Для всіх отриманих на третьому кроці регіонів обчислюється дескриптор, за яким за допомогою міри схожості обчислюється ступінь приналежності текстури розглянутій області кожного класу текстур. База текстур з уже обчисленими для кожного класу дескрипторами підбирається заздалегідь і не залежить від використаного алгоритму класифікації та сегментації, але залежить від предметної області. На

цьому етапі необхідно вибрати тип дескриптора (безпосередньо пов'язаний з методом класифікації), міру схожості (ввести метрику на безлічі дескрипторів) і підібрати базу текстур найбільш точно відображає особливості предметної області.

5.Обєднання областей одного класу. Це дозволить скоротити об'єм інформації і прискорити будь-яку подальшу роботу з результатами обробки фотознімку за рахунок скорочення числа областей.

Проблема розпізнавання текстур на супутникових знімках ставилася для вирішення використовувався аналіз енергетичного спектра зображення. Будувалася гістограми відносних частот спектра зображення і типів текстур, після чого проводилася їх бінаризація. На основі бінарної моделі проводиться визначення меж, після чого проводиться порівняння спектрів кожної області зі спектрами текстур кожного класу і визначається найбільш ймовірний клас кожної області методом найближчого сусіда.

Перевагами такого підходу є особливість щодо повороту і масштабованість, що важливо для ідентифікації текстур на знімках, і стійкість до шуму, що важливо при аналізі фотознімків поганої якості. Крім того, метод не вимагає складних обчислень і не є ресурсомістким, якщо використовувати стандартні алгоритми бінаризаціїі відстеження кордонів. При більш складних і точних методах бінаризації відстеження кордонів складність всього методу залежить від них.

Головним недоліком є робота з зображеннями та текстурами в градаціях сірого через що втрачається інформація, що міститься в кольорі, що при аналізі текстур природного походження іноді грає важливу роль. Так само точність методу безпосередньо залежить від точності , бінаризації зображення.

Для усунення недоліків, наведених вище, для класифікації текстур слід вибрати алгоритм, який працює з кольоровим представленням зразків текстур. Таким чином, весь процес текстурного аналізу зображення можна розділити на два великі етапи: текстурну сегментацію і текстурну класифікацію. Так як дії на одному з цих етапів не зачіпають другий, то і огляд існуючих до них підходів має сенс виробляти для кожного з них окремо.

Так як часто текстура ділянки зображення може заважати точному виявленню його кордонів, слід вибрати ті методи, які враховують цю особливість і використовують фільтри габора. Вище наведених методах для опису зображення використовується набір фільтрів габора. Після чого проводиться кластеризація за виявленими ознаками текстур.

Перевагами використання фільтрів габора є можливість роботи з кольоровими зображеннями в просторі RGB і HSL, невелика кількість обчислень, стійкість до градієнтів і висока точність.

Мінусом же є необхідність мати банк фільтрів габора. На даний момент не існує методу, який дозволяє заздалегідь визначити мінімальне число фільтрів, необхідне для точного опису зображення. Отже, для забезпечення найбільшої точності банк текстур повинен підбирати самостійно експериментальним або експериментально аналітичним чином. Існує методика адаптивного вибору філь трів , однак її застосування можливе не скрізь. Крім того, зворотною стороною стійкості до градієнта є неможливість розпізнати межу між схожими, але не однаковими текстурами, з чим не виникає проблем у інших методів. Так само метод вимагає досить тонкої ручної настройки окремо для кожної предметної області.

Проблема підбору банку фільтрів використанні при сегментації методом нарощування областей. Класичний алгоритм сегментації це ітераційний процес, при якому до однієї початкової області, яка представлена єдиним пікселем, на кожному кроці приєднується тільки один сусідній піксель, для якого міра відмінності від пікселя, що належить до сфери і безпосередньо прилягає до розглянутого пікселя, мінімальна. Зупинка відбувається коли мінімальна міра відмінності більше деякого значення. Відмінність даного підходу від текстурної сегментації полягає в виявленні однорідної області, а не меж областей.

У чистому вигляді метод нарощування областей мало використовується, тому розглядається його модифікація. Нарощування відбувається з декілька областей, які вибираються автоматично.

Недоліками методів, описаних вище, є приєднання тільки одного пікселя, що виключає можливість застосування такої сегментації для текстурованих

областей. Так само методи працюють з зображеннями в градаціях сірого та не враховують інформацію, що міститься в кольорі. Однак ця проблема вирішена в інших роботах. Наприклад методи роботи алгоритму з кольоровими зображеннями, в них розглядається сегментація в колірному просторі RGB

Для забезпечення можливості застосування сегментації нарощуванням областей для текстурованих ділянок необхідно приєднувати на кожної ітерації не один піксель, а блок сусідніх пікселів. Для опису текстур кожного блоку можливе використання різноманітних дескрипторів.

Вибір дескриптора є основоположним кроком текстурної класифікації. Всі методи можна розділити на ті що використовують банки фільтрів для обчислення дескрипторів. Перевагами методів, які використовують банки фільтрів, є їх універсальність. Для кожної предметної області завжди можна підібрати набір фільтрів, що дозволяють точно описати текстури. Крім того з усіх фільтрів можна виділити які відповідають за інваріантність щодо повороту, масштабу, перспективного спотворення, стійкість до шуму. Таким чином для забезпечення інваріантності досить додати певні фільтри і буде необхідності зміни самого методу.

Основним недоліком усіх методів, названих вище, є необхідність підбору банків фільтрів. Для ефективної роботи потрібно підібрати набір фільтрів, що забезпечує визначення основних текстурних ознак і при цьому має мінімальний можливий обсяг. Для багатьох банків фільтрів, наприклад, фільтрів габора, автоматичний вибір такого набору на даний час не розроблено.

Використання локальних бінарних шаблонів (ЛБШ) для обчислення дескрипторів текстур вирішує проблему попереднього вибору фільтрів. В наведеному визначенні локального бінарного шаблону, котрий описує кожну точку за допомогою сусідніх з нею точок з допомогою даного методу описуються зображення в градаціях сірого. Отриманий опис точок неінваріантний щодо масштабу, повороту спотворення і нестійкий до шуму.

2 РОЗРОБЛЕННЯ АЛГОРИТМІВ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

2.1 Алгоритм нарощування областей блоками

Розглянемо форму блоків на зображенні і зв'язність.

Замощенням області S називається таке розбиття на множину непересічних багатокутників, об'єднання яких дає саму область S. Альтернативне назва замощення - мозаїка. Багатокутники, на які розбивається область, називаються полігонами.

За типом полігонів, замощення можна розділити на такі:

– регулярні, це мозаїки, складені з полігонів одного типу і розміру. Узагальненням можна вважати полігони однакової площі без уточнення щодо типу.

– Нерегулярні, це мозаїки, складені з декількох різних типів полігонів або полігонів одного типу, але різних розмірів.

На рисунку 2.1 представлені приклади різних видів замощення.

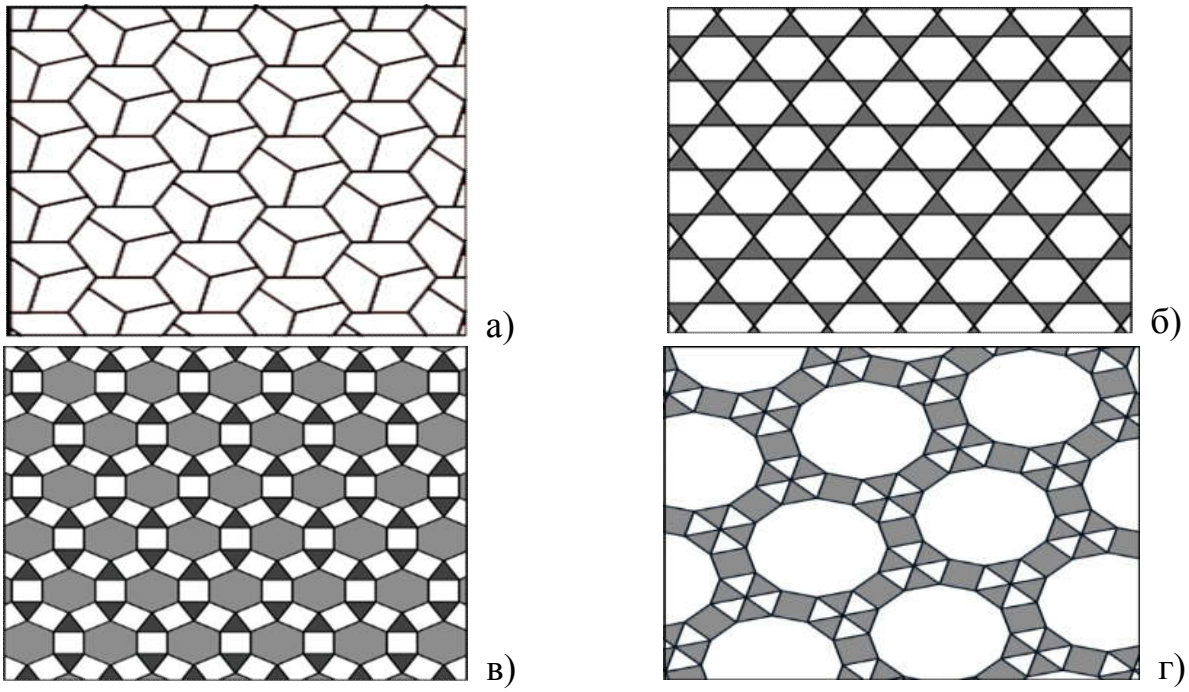


Рисунок 2.1 – Приклади замощення: а) регулярне, б) нерегулярне квазіпрівіальне, в) нерегулярне напів правильне г) нерегулярне

Регулярні замощення по типу полігонів, на які розбивається площина, можна розділити на правильні і неправильні. При неправильному регулярному замощенню всі полігони мають однакову форму, але не являються правильними.

Наприклад, на рисунку 2.2 наведено всі відкриті на сьогоднішній день варіанти регулярних неправильних замощень, полігони яких є опуклими п'ятикутниками.

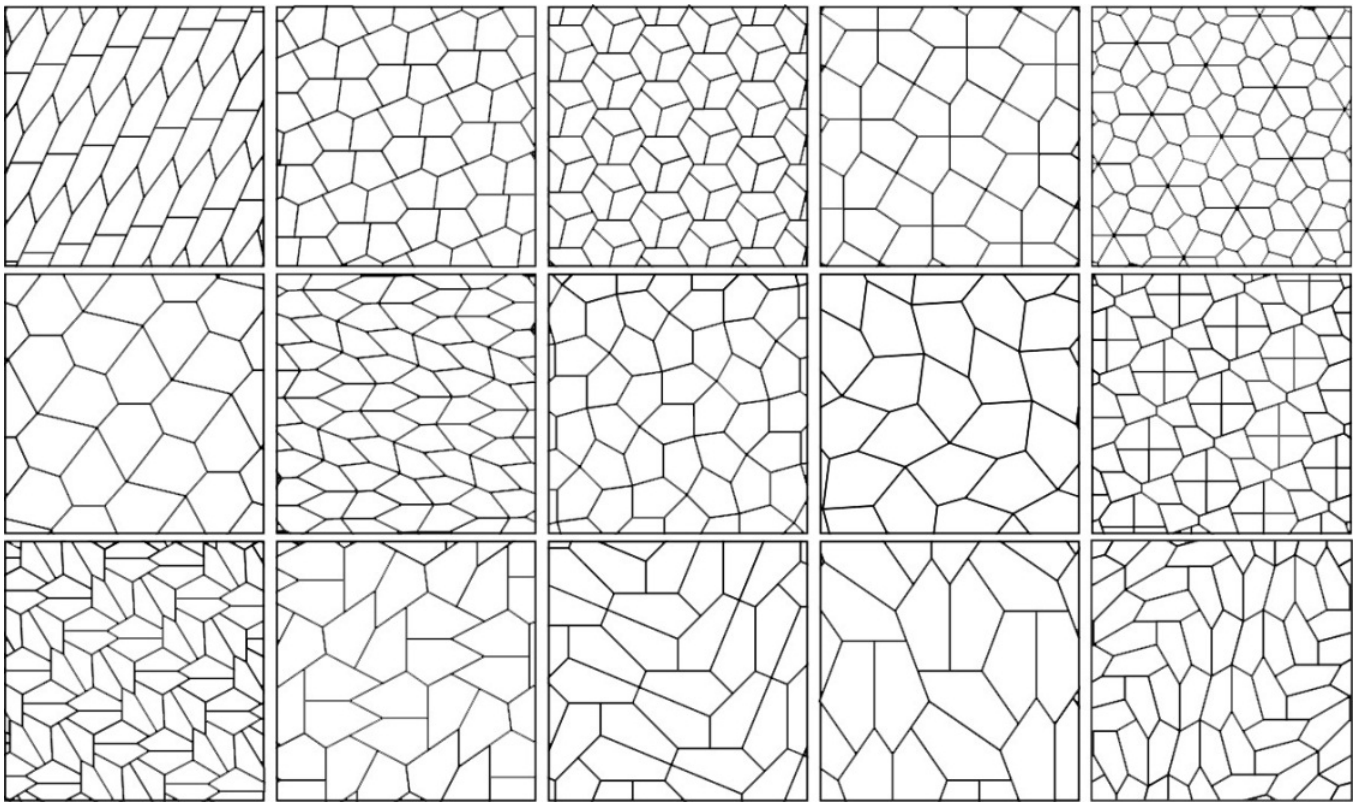
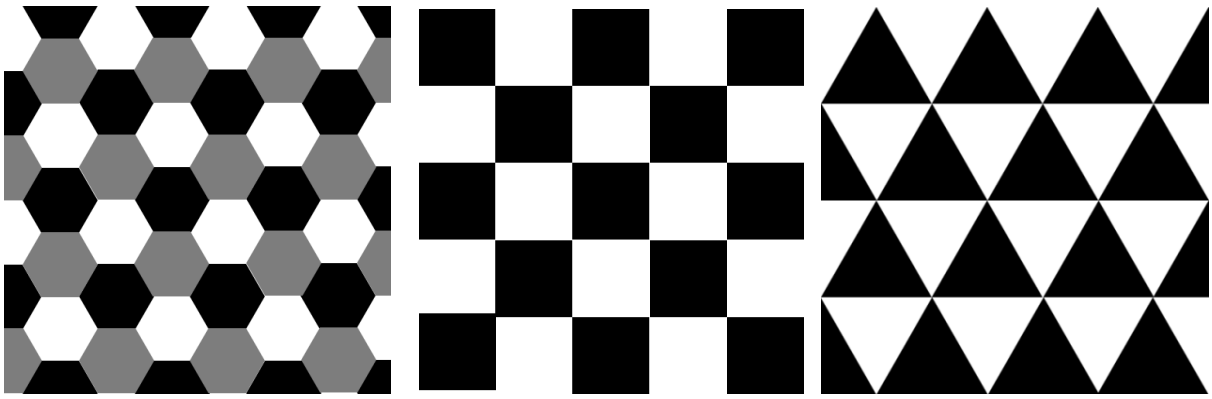


Рисунок 2.2 – Варіанти регулярних неправильних замощень

Правильні замощення складають правильні багатокутники, доведено, що існує тільки три правильних замощення: трикутне, квадратне і гексагональних. Дані мозаїки представлені на рисунку 2.3.

Для нарощування областей блоками можна використовувати будь-який з представлених вище варіантів замощення, як регулярних, так і нерегулярних. Прикладом нерегулярної сітки може служити будь яка сітка, отримана в результаті тріангуляції. Однак на практиці використання складних блоків недоцільно, так як для визначення приналежності пікселів конкретного блоку

необхідно проводити додаткові розрахунки, що враховують форму блоку. З цієї причини в комп'ютерній графіці використовують правильні замощення.



а) б) в)

Рисунок 2.3 – Види правильних замощення: а) трикутне, б) квадратне, в) гексагональних

Визначення 6. Існує два підходи до визначення суміжності:

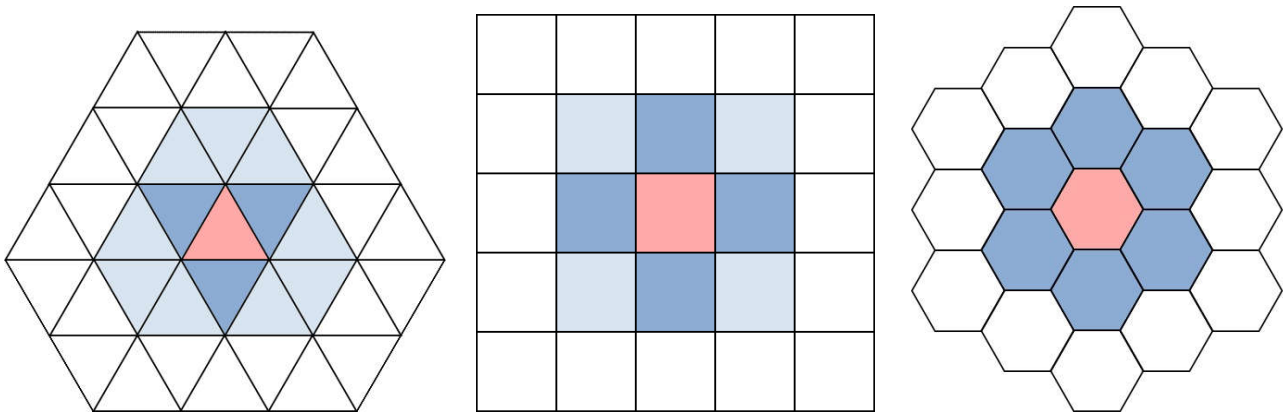
– суміжні називаються полігони мають спільну сторону.

– суміжні називаються полігони мають спільну сторону або загальну вершину.

Визначення 7. Полігон суміжний з n полігонами називається n-зв'язковим.

На рисунку 2.4 для полігонів правильних заміщень продемонстровані типи зв'язності відповідно до двох вищенавединих визначень. Області, суміжні з виділеною, по загальній стороні виділені синім, по загальній вершині блакитним.

У даній роботі в якості блоків при нарощуванні областей будуть використовуватися квадратні 4-зв'язні полігони. Вибір форми блоків обумовлений тим, що вона найбільш природно відображає положення пікселів растрового зображення. Обчислення приналежності пікселів кожному полігону в даному випадку не вимагає обчислювальних витрат. 4-зв'язність забезпечує відсутність «одиночних» блоків, прилягающись нарощуванні області тільки кутами. Таким чином виключаються ділянки областей, ідентифікація яких неможлива через їх недостатню площу.



а) б) в)

Рисунок 2.4 – Суміжність при правильних замощення

а) 3-зв'язкова (сині суміжні блоки) і 12-зв'язкова (сині і блакитні блоки) трикутна область, б) 4-зв'язкова (сині суміжні блоки) і 8-зв'язкова (сині і блакитні блоки) квадратна область, в) 6-зв'язкова гексагональна область.

Алгоритм нарощування блоками єдиної області.

Ідея нарощування області блоками полягає в розширенні алгоритму, що працює з пікселями, зображення представлені блоками. При такому поданні блок є найменшою структурною одиницею. В цьому випадку ключовим моментом алгоритму є вибір функції переходу від групи пікселів до блоку, тобто задання дескриптора.

Алгоритм можна розбити на наступні етапи:

1.Переведення растрового представлення зображення в блочне, шляхом обчислення дескриптора кожної області:

$$I = \{x_1,..,x_N\} \to Y = \{y_1,..,y_M\},$$

де N - кількість точок зображення (пікселів),

M - кількість блоків в блоковому поданні, $M < N$.

Після чого з новим поданням можна буде працювати як з растровим зображенням. Для початкової точки визначається відповідний блок, який є стартовим для алгоритму, який працює з блоками, тобто $x^0 \to y^0$. Задається порогове значення функції схожості μ.

2. Стартовий блок приймається в якості початкової області $S^{(0)} = y^0$. Лічильнику приєднаних блоків присвоюється значення 1. $m = 1$.

3. Для початкової області складається список суміжних з нею ще не з'єднаних блоків кандидатів на приєднання до неї. $S_i = \{yk,.., yl\}$ - список кандидатів на приєднання на i -му кроці.

4. Для кожного блоку кандидата обчислюються значення функції схожості з блоками, які належать нарощуванні області, і безпосередньо з'єднанні з нею $D_{ij} = \max\{d(S_{ij,}S_k^{(i)})S_k^{(i)} \in S^{(i)}\}$ - значення функції схожості блоку кандидата $S_{ij} \in S_i$ з наростаючою областю на i-му кроці.

5. Визначити блоки кандидати, значення функції схожості яких більше порогового значення μ, яке задається заздалегідь. $S = \{S_{ij} \mid S_j \in S_i, D_{ij} > \mu\}$

Знайдені блоки приєднуються до наростаючої області $S^{(i+1)} = S^{(i)} \cup S$.

Лічильник приєднаних областей збільшується на потужність множини приєднуваних блоків $m = m + \mid S \mid$.

6. Якщо $m = M$ або $\mid S \mid = 0$, то подальше приєднання блоків неможливе. Перехід до кроку 7. Інакше - перехід до кроку 3.

7.Створення маски виділеної області.

$$I(Yj) = \begin{cases} 1, & Yj \in S \\ 0, & Yj \notin S \end{cases},$$

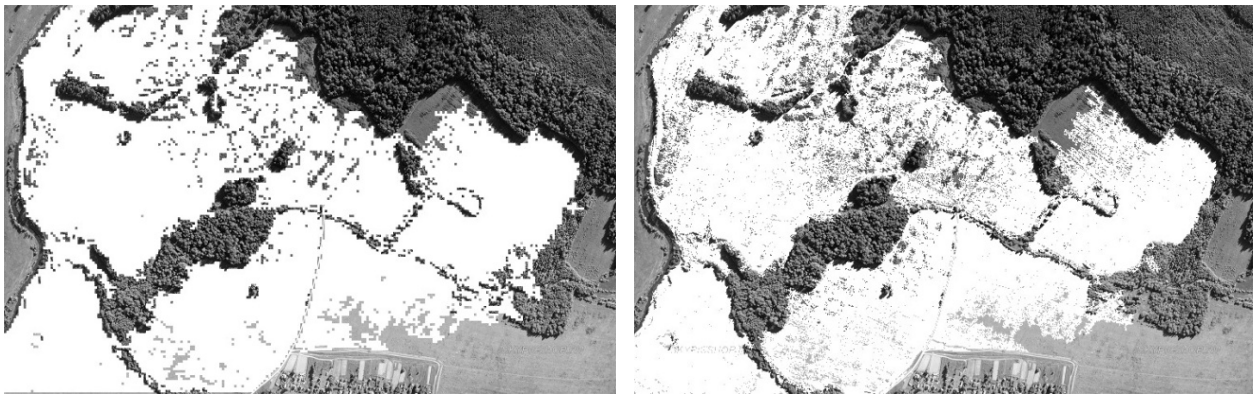
$$I(Yj) \to I(X_i).$$

Перехід від блоків до пікселів.

На рисунку 2.5-2.6 показані приклади застосування немодифікованого методу для кольорових зображень (рисунок 2.5,б), методу для зображень в градаціях сірого (рисунок 2.6) і методу нарощування блоками (рисунок 2.6,б) до вихідного фото (рисунок 2.5,а).

а) б)

Рисунок 2.5 – Початкове фото (а) і сегментованим немодифікованим методом (б)



а) б)

Рисунок 2.6 – Сегментація немодифікованим методом нарощування областей (а) і методом нарощування областей блоками (б)

Автоматичний вибір початкових областей.

Підходи до вибору початкових областей. Особливістю всіх методів сегментації на базі нарощування областей є необхідність завдання початкових точок. Кількість областей в результаті безпосередньо залежить від кількості початкових точок. Однак при автоматичні сегментації зображень не можна заздалегідь сказати на скільки областей вони будуть розділені. З цього випливає питання визначення кількості і координат початкових точок.

Залежно від моменту визначення початкових точок всі методи автоматичного визначення можна розділити на статичні і динамічні.

Статичний підхід характеризується визначенням кількості і положення початкових точок до запуску алгоритму сегментації. У цьому випадку на вхід

функції, що виконує сегментацію методом нарощування областей, подається вже заздалегідь певний набір координат стартових точок.

Статичні підходи можна розділити на два напрямки: завдання точок в діалоговому режимі і обчислення точок програмними методами.

1. В діалоговому режимі користувач для кожного зображення сам задає кількість початкових точок і їх координати. Для досвідченого фахівця не складе труднощів визначити кількість областей на фото, а отже і стартових точок. Але сама наявність спеціального листа зводить нанівець спроби автоматизувати процес.

2. Програмний підхід являє собою проведення попередньої обробки зображення для визначення кількості початкових точок. Для обробки використовується алгоритм виявлення границь. Так як обчислювальні витрати на обробку не повинні наближатися до витрат основної частини алгоритму сегментації, то метод виявлення кордонів повинен бути максимально простим і швидким, що може негативно позначитися на точності.

Так як точність на етапі обробки не важлива, то можуть бути виявлені в повному обсязі області або, навпаки, структурні одиниці текстури можуть бути прийняті за окремі області. Тому даний підхід часто застосовується для попередньої оцінки кількості і розташування початкових областей з метою подальшого їх динамічного уточнення.

При динамічному підході початкові точки визначаються безпосередньо в процесі виконання алгоритму сегментації. Кількість частин, на які буде розділено зображення, заздалегідь не відомо. Алгоритм отримує на вхід тільки аналізоване зображення. На кожній ітерації генерується одна початкова точка і для неї проводиться нарощування відповідної області. Так як розташування кожної наступної початкової точки залежить від уже виявлених структурних областей, результат сегментації може залежати від послідовності вибору цих точок.

Даний підхід найбільш зручних, тому що не вимагає попередньої обробки зображення, дозволяє повністю розбити фото на області і забезпечує максимальну ступінь автоматизації сегментування.

При динамічному визначенні початкових точок існує можливість реалізації інтерактивного режиму. У цьому випадку на кожній ітерації користувач вибирає з доступною області одну початкову точку, для якої проводиться нарощування, після чого оброблені точки або блоки з доступною областю прибираються, а користувач знову вибирає наступну точку з уже скороченої доступної області. Однак такий підхід не є автоматичним і відноситься до динамічних тільки формально. На практиці його використання при великому числі можливих областей незручне.

Алгоритм автоматичної сегментації.

1. Перетворення растрового представлення зображення в блочне, шляхом обчислення дескриптора кожної області:

$$I = \{x_1, ..., x_N\} \to Y = \{y_1, ..., y_M\},$$

де N - кількість точок зображення (пікселів),

M - кількість блоків в блочному представленні, $M < N$.

Після чого з новим поданням можна буде працювати, як з растровим зображенням. Задається початкове значення функції схожості μ. Лічильнику виявлених областей, який буде використовуватися для присвоєнняння міток обробленим блокам, присвоюється стартове значення $k = 0$. $Q = \emptyset$ - множина приєднаних блоків (блоків з мітками).

2. Вибирається стартовий блок $y^0 \notin Q$ який приймається в якості початкової області. Лічильник областей збільшується на одиницю $K = K + 1$. Початкова блоку присвоюється поточна мітка K. Початкова область $Q = Q \cup y^0 S^{(K)} y_0$ з міткою K.

3. Для початкової області складається список суміжних з нею блоків, що не мають позначок, кандидатів на приєднання. Список кандидатів $S_i = \{y_1 \mid y_1 \notin Q, y_1 S^{(K)}\}$ на присвоєння на i-му кроці.

4. Для кожного блоку кандидата обчислюються значення функції схожості з блоками, які належать нарощуванні області і безпосередньо суміжні з нею.

$D_{ij} = \max\{d(S_{ij}, S_k^{(i)}) \mid S_k^{(i)} \in S_i^{(K)}\}$ – значення функції схожості блоку кандидата $S_{ij} \in S$ з нарощуваною областю на і -му кроці.

5. Визначаються блоки кандидати, значення функції схожості яких більше порогового значення μ, яке задається заздалегідь

$$S = \{S_{ij} \mid S_j \in S_i, D_{ij} > \mu\}.$$

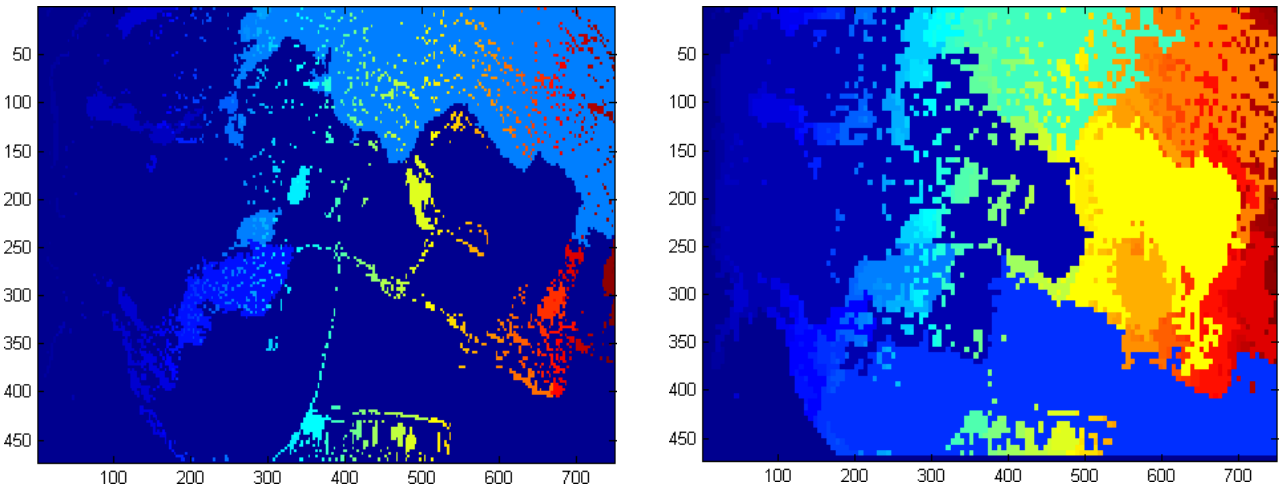
Знайдені блоки приєднуються до наростаючої області. $S = \{S_{ij} \mid S_j \in S_i, D_{ij} > \mu\}$

Приєднаним блокам присвоюються мітки поточної області $Q = Q \cup S$.

6. Якщо $|Q| = M$, то не залишилося необроблених блоків, все зображення сегментоване. Перехід до кроку 8. Інакше перехід до кроку 7.

7. Якщо $|S| = 0$, то подальше приєднання блоків до даної області неможливо. Перехід до кроку 2. Інакше - перехід до кроку 3.

8.Перехід назад від блоків до пікселів $y_j \to x_i . x_i = x_i^{(K)}$ для $y_j \in S^{(K)}$



а) блок 3x3
1492 сегмента

б) блок 6x6
6998 сегментів

Рисунок 2.7 – Сегментація з автоматичним вибором початкових областей

На рисунку 2.7 представлені результати автоматичної сегментації для блоків різних розмірів. Кожному сегменту відповідає свій відтінок кольору. На

другому кроці алгоритму початковий блок кожної області вибирався довільним чином з безлічі непомічених блоків.

Об'єднання областей малої площі.

Головним недоліком методу сегментації нарощуванням областей з автоматичним вибором початкових точок є поява великої кількості областей малої площі, тобто одиночних областей, що складаються з одного або декількох блоків. Причинами їх появи можуть бути погана якість вихідного зображення, шум, особливості текстури або інші особливості зображення.

Так як при сегментації площа приєднуваних блоків вибиралася таким чином, щоб одночасно забезпечити достовірне представлення текстурних ознак кожного блоку і при цьому гарантувати в достатні степені високу точність, проте розмір таких блоків недостатньо точний для максимального виділення текстурних ознак. Таким чином області, представлені одиночними блоками або малою групою блоків, загальна площа яких є меншою за мінімальну класифікуючу область не можуть бути оброблені. Для усунення цього проводиться об'єднання областей.

Визначення 8. Зв'язною областю називається область, де будь які дві точки можуть бути з'єднані кривою, яка повністю належить даній області.

Залежно від вимог зв'язності існує дві точки зору на мету і процес об'єднання областей.

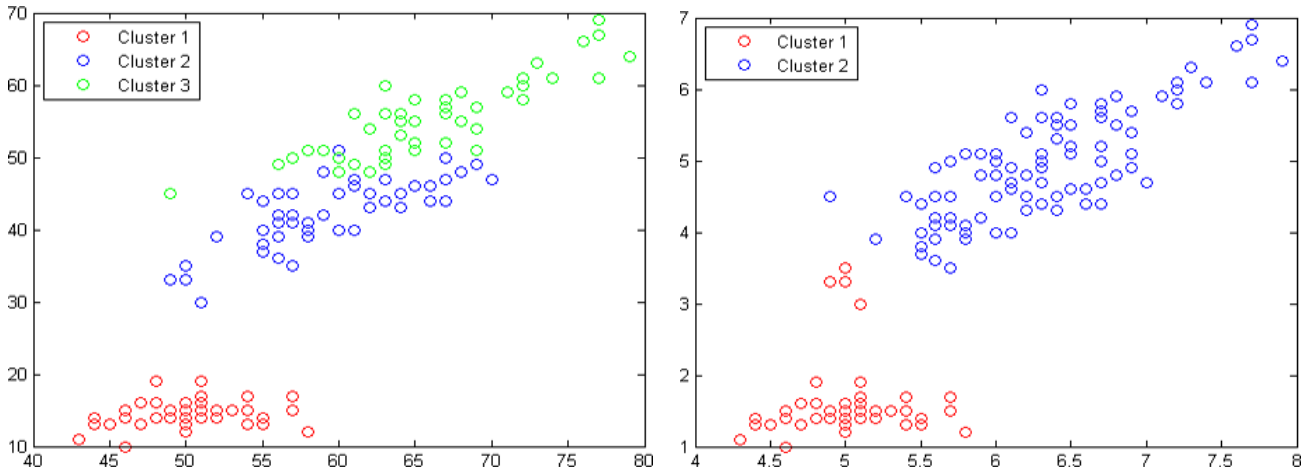
Якщо ставиться вимога обов'язкової зв'язності кожної області, при об'єднанні розглядаються тільки суміжні сегменти. У цьому випадку метою являється виключення суміжних сегментів одного типу шляхом їх об'єднання. В даному контексті тип сегмента не визначається. Для кожної конкретної задачі поняття типу сегмента визначається індивідуально шляхом вибору відповідних дескрипторів і визначення функції схожості типів.

Якщо допускаються несуміжні області, то для об'єднання часто приміняється кластеризація. Як і в попередньому випадку, при кластерному аналізі необхідне введення функції схожості, яка виконує роль функції відстані в просторі текстурних ознак. Так як аналізується довільне зображення і неможливо заздалегідь сказати, скільки на ньому представлено типів текстур,

слід зазначити, що для кластеризації не підходять методи, що вимагають попередньої вказівки числа кластерів, такі як *K*-means.

Перевагою об'єднання областей за допомогою кластерного аналізу є істотне спрощення подальшої класифікації, так як в ідеальному випадку зображення буде розділено на текстурні області, тип текстури кожної з яких зустрічається рівно один раз. Це дозволить після класифікації текстур пропустити крок об'єднання. На практиці ж через схожість текстур деяких типів можлива їх втрата через об'єднання кількох кластерів в один.

На рисунку 2.8 показана можлива ситуація з об'єднаннями кластерів . Альтернативою є використання алгоритмів кластеризації з малим радіусом впливу для об'єднання тільки тих областей, заходи схожості яких приймають значення близькі до 1 (в тому випадку, якщо міра схожості змінюється на відрізку [0; 1]). Це допоможе скоротити число сегментів, але не призведе до підвищення відсотка можливих неправильно класифікованих через невірне сегментування ділянок.



а) б)

Рисунок 2.8 – Неточності кластеризації

Алгоритм об'єднання областей.

Для вирішення поставленого завдання на етапі сегментації досить забезпечити відсутність некласифікованих ділянок. Для цього проводиться проста перевірка:

1.Нехай зображення розділене на N сегментів $P = \{P_1,...,P_N\}$. Для кожної області обчислюється її площа $S = \{S_1,...,S_N\}$. Задається мінімальна класифікуюча площа μ.

2.Створується список областей з недостатньою площею

Для прикладу були взяті часткові дані з набору даних «іриси Фішера», а саме довжина чаші листка, довжина пелюстка і вид ірису (для порівняння з отриманим результатом). Графіки згенеровані пакетом MATLAB. На прикладі видно, що при радіусі впливу 0,7 субтрактивний алгоритм нечіткої кластеризації виявляє тільки два кластери (рисунок 2.8, б), тоді як насправді їх три (рисунок 2.8)

$$P^*\{P_i \mid S_i < \mu, i = 1N\}.$$

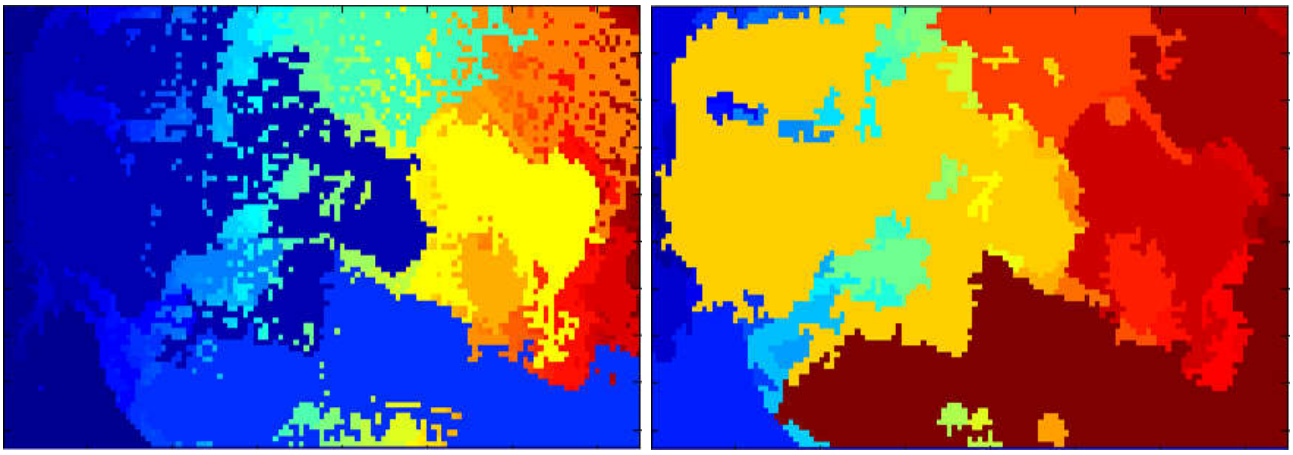
3.Якщо $P^* \neq \emptyset$ перехід до кроку 4, інакше - кінець.

4. Для $P_j \in P^*$ серед суміжних з ним сегментів вибирається найбільш похожій $P = \max\{P_j, P_k\}$ де $P_k \in P(P_k, P_K)$ - функція схожості текстур областей $P_j i P_K$.

5. Якщо $P_j \in P^* i P_k \in P^* i S_j + S_k < \mu,$ то $P_i = P_j \cup P_k$. Змінюючий список некласифікуючих участків: $P^* = P^* / (P_j \cup P_k) \cup P_i$. Змінючий список відповідних площ сегментів $S = S / (P_j \bigcup P_k) \bigcup S_i$, де $S_i = S_j + S_k$. Перехід до кроку 3.

6. Якщо $P_j \in P^* i P_k \in P$ то змінючий список некваліфікованих участків: $P^* = P^* / P_j$. Змінюючий список відповідних площ сегментів $S = S / P_j, S_i = S_j + S_k$.

На рисунку 2.9 продемонстровано сегментоване зображення до (рисунка 2.9, а) і після (рисунка. 2.9, б) об'єднання блоків. Об'єднання блоків ппроводиться за алгоритмом, що об'єднує тільки некласифікуючі блоки, описані вище.



а) б)

Рисунок 2.9 – Сегментовані зображення до (а) і після (б) об'єднання областей

2.2 Текстурна класифікація

Бібліотека текстур необхідна для навчання нейронної мережі, яка широко застосовується при вирішенні задач класифікації образів. Від вибору бази текстур залежить точність роботи алгоритмів класифікації. При недостатньому обсязі бібліотеки висока ймовірність виявлення області, для яких не виконується необхідна умова класифікації. Під недостатнім обсягом бібліотеки в цьому випадку розуміють як обсяг всіх зразків, так і число класів.

Твердження 1. Необхідною умовою приналежності текстури даному класу є вимога, щоб значення функції схожості текстури з цим класом було вище деякого порога впевненості.

$$f(S,T) > \eta,$$

де $f(S,T)$ - міра приналежності текстури S класу T,

η - поріг упевненості.

Твердження 2. Необхідною умовою класифікації текстури являється виконання необхідної умови приналежності текстури хоча б для одного класу.

$$\max \{f(S,T_i) > \eta\} \text{ для } i = 1,N,$$

де $f(S,T_i)$ - міра приналежності текстури S класу T_i,

N - число можливих класів,

η - поріг впевненості.

Поріг впевненості η задається користувачем і дозволяє регулювати точність класифікації. За замовчуванням $\eta = 0$, тобто необхідні класифікації виконується завжди. У цьому випадку кожна текстура належить класу, міра приналежності до якого більша за міру приналежності до інших класів.

Прагнення до відсутності некласифікованих текстур може привести до зайвого збільшення числа класів бібліотеки. Неможливо створити універсальну базу текстур без шкоди точності і трудомісткості класифікації. В першу чергу шкідливі текстури, ймовірність появи яких на уже згадуваному зображенні вкрай мала або взагалі дорівнює нулю. Наприклад, безглуздо додавати в бібліотеку клас «хутро», коли вона буде використовуватися для аналізу аерофтознімков. З цього випливає, що важливою умовою ефективної класифікації є правильний вибір бібліотеки.

Перед безпосереднім збором зразків текстур для вирішення конкретної задачі слід визначити наступні моменти.

1.Визначити природу предметної області. Це можуть бути рентгенівські знімки, мікроскопічні, фото різних матеріалів, фото місцевості.

2.На основі предметної області можна визначити передбачувані класи текстур. По можливості виділення класів проводити від спільних, дроблячи їх на більш чатскові в процесі аналізу предметної області. Це позволить уникнути перетинання класів і при необхідності дасть можливість забезпечити рівень деталізації без зайвих зусиль.

3. Визначити приблизний діапазон зміни масштабу текстур. Наприклад, на якому збільшенні робляться знімки електронним мікроскопом або з якої висоти робляться фотографії з повітря. На основі цього проводити відбір текстур для кожного класу. Нехай предметною областю є аерофотознімки, а серед класів

текстур обраний клас «трава», то фотографії газону, зроблені під кутом 90^0 відносно поверхні, формально підходять, але, так як аерофотознімки робляться з висоти не менше 500 м, на практиці не застосовуються, а з цього виходить послідовно не повинні бути додані в якості зразків в даний клас.

Для поставленої задачі класифікації текстур на аерофотознімках предметом дослідження є фото земної поверхні зроблені з повітря. Щоб по можливості скоротити кількість класів на етапі перевірки роботи алгоритмів було прийнято рішення обмежити аналізовані аерофотознімки знімками середньої смуги. За даних умов було виділено 8 основних класів текстур: «вода» (річки, озера, моря), «Грунт», «болото», «ліс», «степ», «гори», «асфальт» (великі заасфальтовані ділянки як, наприклад, аеродроми), «будови» (міста, населенні пункти). Так як при постановці завдання було визначено, що аерофтознімками вважаються будь-які знімки, зроблені з літального апарату, то текстури на зразках повинні бути сфотографовані не менше ніж з висоти 500 метрів над рівнем землі.

В якості образів текстур використовувалися елементи різних баз даних, а так само вручну виділені області на зображеннях, знайдених в мережі Інтернет. Приймалися кольорові зображення і зображення в градаціях сірого. Кольорові зображення переводилися в колірну модель RGB для однаковості уявлення і спрощення подальшої роботи бібліотеки. Низький рівень спеціалізації класів дозволив на даному етапі обійтися без звернення до спеціалізованих екологічних сайтів. В остаточному підсумку кожен клас складеної бібліотеки текстур налічував по 100 зразків розміру 200 × 200 з різною часткою кольорових і монохромних пікселів (таблиця 1).

Таблиця 2.1 – Число кольорових і монохромних зразків кожного класу бібліотеки текстур

	Вода	Грунт	Болото	Степ	Гори	Ліс	Асфальт	Будівлі
Кольорові	7	67	51	41	46	62	30	44
Монохромні	28	33	49	59	54	38	70	56
Всього	100	100	100	100	100	100	100	100

Локальний бінарний шаблон (Local Binary Pattern - ЛБШ) оператор, що описує точку зображення за допомогою сусідніх пікселів.

Нехай $T = t(p_c, p_0, ..., p_{n-1})$ - фрагмент текстури, p_i - точки, що його утворюють. Тоді локальний бінарний шаблон визначається наступним чином

$$LBP_{r,n} \sum_{n-1}^{i=0} f(p_i - p_c)2^i$$

де r - радіус кола, на якій беруться точки для опису центральної,

n - кількість точок для опису центральної.

$$f(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}.$$

Інваріантність щодо повороту.

Класичний ЛБШ-оператор не має інваріантності відносно повороту, що призводить до необхідності додавання в банк текстур однакових зразків, повернених на різні кути. З метою скорочення розміру текстур і підвищення точності опису вводиться модифікований, інваріантний щодо повороту, оператор LBP^{ri}.

$$LBP_{r,n}^{ri} \min\{ROR(LBP_{r,n}, i)\},\ i = 0, n-1$$

де ROR(X,i) - функція повороту блоку X на i пікселів за годинниковою стрілкою.

Стійкість до шуму. ЛБШ-оператор в класичному уявленні вкрай чутливий до шуму, який неминучий на аерофотознімках. Отже невеликий, що не впливає на текстуру, шум при використанні ЛБШ, особливо на коротких відстанях, може привести до великої частки помилкових класифікацій.

Для забезпечення стійкості до шуму використовується усереднення. Якщо піксель описується за допомогою сусідніх точок, а усереднення проводиться по точках, то для опису кожного пікселя необхідно брати qn точок.

$$\overline{p}_{r,q} = \frac{1}{q}\sum_{i=0}^{q-1}\overline{p}_{r,kq+i}, \quad k = 0, n-1.$$

Забезпечення інваріантності щодо масштабу. Так як не завжди існує можливість підбору банку текстур таким чином, щоб масштаб текстур на зразках збігався з масштабом будь яких аналізованих текстур, постає необхідність забезпечити певний діапазон допустимої відмінності масштабів зображення і зразків з банку даних. Для ЛБШ-операторів це можливо здійснити шляхом зміни їх радіусу. При цьому обчислюються відразу кілька дескрипторів з різними радіусами. Число різних радіусів можна міняти, тим самим контролювати складність і ефективність класифікації.

Для опису текстурних ознак областей і блоків використовуються функції-дескриптори, які отримують на вхід набір пікселів кожного блоку, а на виході визначають вектор ознак кожного блоку.

Визначимо дескриптор $LBP^{ri}\_S_{r,n}$, описує центральний піксель через сусідні. Нехай r - радіус ЛБШ-оператора, q - число пікселів для опису однієї сусідньої точки, n - число сусідніх пікселів для опису центрального. Будемо вважати, що $n = 8_r$, так як такий вибір сусідніх пікселів найбільш природний для растрових зображень. p_c - центральний піксель ЛБШ-оператора, $\overline{p}_{r,n}$ визначений в (1).

$$\mathrm{LBP\_S}_{r,n} = \sum_{n}^{i=0} f(\overline{p}_{r,i} - p_c)2^i,$$

$$\mathrm{LBP\_S}^{ri}_{r,n} = \min\{\mathrm{ROR}(\mathrm{LBP\_S}_{r,n}, i)\}, \; i = 0, n-1$$

Визначимо дескриптор $LBP^{ri}\_D_{r,n}$ що описує ступінь відмінності між центральним пікселем і сусідніми.

$$z_{r,k} = \frac{1}{n}\sum_{i=0}^{n-1} | \overline{p}_{r,k+1} - p_c |, \qquad i = \overline{0, n-1}$$

$$\text{LBP\_D}_{r,n} = \sum_{n}^{i=0} f(z_{r,i} - \frac{1}{n}\sum_{j=0}^{n-1} z_{r,j}) 2^i$$

$$\text{LBP\_D}^{ri}_{r,n} = \min\{\text{ROR}(\text{LBP\_D}_{r,n}, i)\}, \quad i = 0, n-1$$

$$\text{LBP}_{r,n} = \text{LBP\_S}^{ri}_{r,n} \bigcup \text{LBP\_D}^{ri}_{r,n}$$

де під операцією об'єднання мається на увазі об'єднання гістограм $\text{LBP\_S}^{ri}_{r,n}$ та $\text{LBP\_D}^{ri}_{r,n}$

Підсумковим дескриптором, стійким до зміни масштабу, є дескриптор ЛБШ:

$$LBP = \bigcup_{k=1}^{m} LBP_{k,8k}$$

де m - максимальний розглянутий радіус ЛБШ-оператора.

Для кольорових зображень обчислюються ЛБШ_R, ЛБШ_G, і ЛБШ_B, за значеннями насиченості відповідних каналів. Після чого одержані гістограми частот об'єднуються в загальний дескриптор ЛБШ_RGB.

За замовчуванням вхідне зображення представлено в кольоровому просторі RGB. Перехід від інших колірних моделей, наприклад, YCbCr, HSV, CMYK, до RGB і назад здійснюється за виведеним для кожної моделі стандартними формулами.

2.3 Двокроковий алгоритм класифікації текстури

При перетвоенні кольорового зображення в градації сірого неминуча втрата інформації. Це призводить до того, що різним кольорам можуть відповідати однакові значення рівня сірого. Так як в природі колір відіграє основну роль, і забарвлення рослин, піску, ґрунту і т.д. ніколи не буває випадкове, а визначається групою екологічних і кліматичних чинників, то при класифікації в градаціях сірого результати можуть бути недостатньо точними. Однак часто дослідники стикаються з неможливістю збору бібліотеки текстур повністю з кольорових зразків. У цьому випадку доводиться працювати з усією бібліотекою в градаціях сірого, так як перехід до колірного простору для сірого зображення неоднозначний.

Двокроковий підхід до класифікації текстур полягає в роботі на різних етапах з різними представленнями зображень і банку текстур.

Етап 0. Для роботи з різними представленнями зображень необхідно провести перетворення бібліотеки текстур. Для цього вона ділиться на дві частини: кольорові зразки і зразки в градаціях сірого. Після чого формується дві нових бази текстур (рисунок 2.10) в градаціях сірого, яка формується шляхом об'єднання частини початкової бібліотеки в рівнях сірого і кольорових частин, перекладеної в рівні сірого, і повністю заповнені якимось кольором, що представляє собою кольорову частину початкової бібліотеки.

Етап 1. На даному етапі аналізуються структурні особливості текстури. Здійснюється перетворення копії аналізованої текстури представленої в градаціях сірого і наступна класифікація з використанням відповідної бібліотеки текстур. В результаті для кожного класу отримуємо ступінь приналежності текстури до нього. На наступний етап передаються тільки ті класи, ступінь належності до яких вище порогового значення. Якщо такий клас один, то другий етап ігнорується. Якщо таких класів немає зовсім, то вибирається найбільш ймовірний клас і другий етап так само ігнорується. Можливий альтернативний

підхід до відбору класів, згідно з яким вони сортуються за ступенем ймовірності іна другий етап передаються n найвірогідніших.

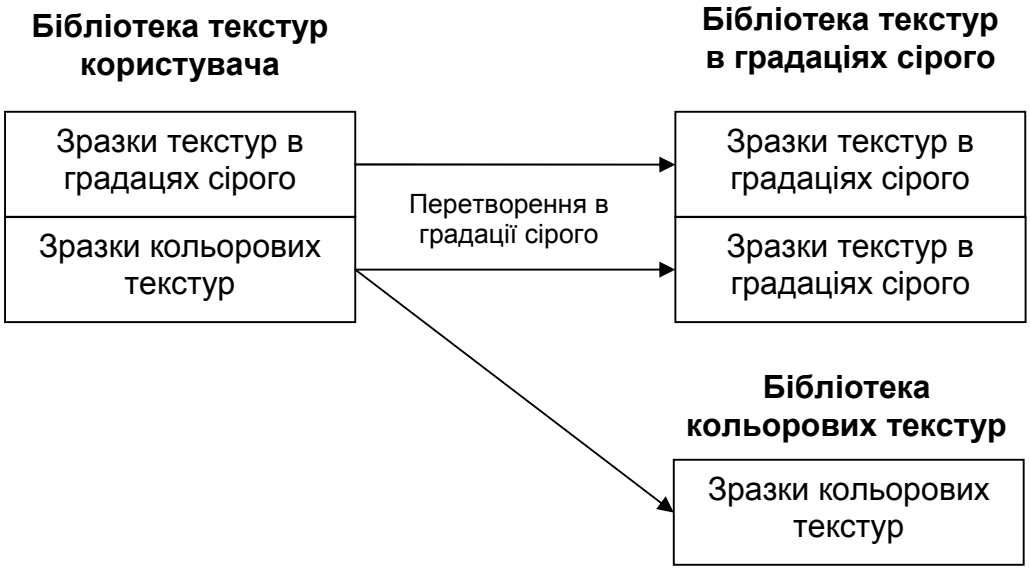


Рисунок 2.10 - Схема формування бібліотек текстур для кожного етапу класифікації

Етап 2. На даному етапі для найбільш схожих за фактурою з вихідним зображенням класів текстур проводиться колірний аналіз, що дозволяє уточнити результати. Обчислюються ймовірності приналежності текстур класам, представленим кольоровими зразками. Після чого обчислюється підсумкова ймовірність приналежності шляхом множення ймовірностей отриманих на першому і на другому етапах. У підсумку отримуємо клас текстури який являється клас з найбільшою ймовірністю приналежності.

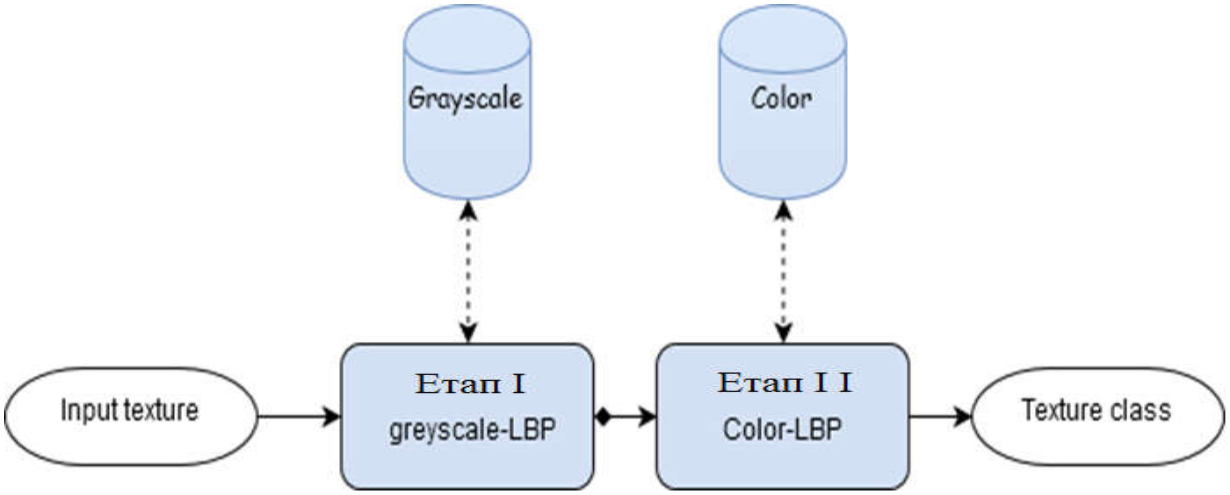


Рисунок 2.11 – Два етапи класифікації текстур

В другому розділі розроблено алгоритми нарощування областей блоками та текстурної класифікації.

3 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ РОЗРОБЛЕНИХ АЛГОРИТМІВ

3.1 Інструментальні засоби

Для написання програмного засобу для проведення тестування обрано мови програмування Python, Java, C++. Обрано середовище IntellijIDEA, PyCharm. Програмне середовище – це комплекс програмних засобів, що дозволяють реалізувати принципи швидкого програмування програм Rapid Application Development. Середовища мають ряд переваг, зокрема:

– широкі можливості по написанню, компіляції та відлагодженню програмних засобів;

– зручний та інтуїтивних інтерфейс (головне вікно);

– наявність додаткових вікон, для пришвидшення роботи з програмним кодом (інспектор об’єктів, дерево об’єктів, тощо);

– великий набір компонент та інструментів;

– наявність компонент для роботи з графікою;

– можливість підключення додаткових бібліотек та компонентів;

– можливість створювати консольних та діалогових програмних засобів;

– наявність бази підказок та файлів з додатковою інформацією про команди середовища та компоненти.

Для роботи програми використано робочу станцію на базі операційної системи Windows. Вибір даної операційної системи пов'язаний наступними перевагами, даної операційної системи над іншими:

– широка розповсюдженість ОС Windows;

– велика кількість користувачів, що мають навики роботи з даною системою;

– наявність програмних засобів для проведення додаткової обробки зображення перед завантаженням в програму аналізу;

– інтуїтивність та зручність роботи в операційній системі;

Для проведення аналізу вхідного зображення використано бібліотеку функції програмного засобу ImageJ. До базових можливостей програми можна віднести наступне.

Мультиплатформеність: Оскільки ImageJ написана на Java, те її можна запустити на Linux, Windows (32-bit або 64-bit), Mac OS X .

Відкрита програма: ImageJ і її вихідний Java код вільно доступні. Ні яких ліцензійних обмежень.

Співтовариство користувачів: ImageJ підтримується величезним співтовариством користувачів в усьому світі, які допоможуть розв'язати будь-яке питання. Більш 1700 користувачів і розроблювачів підписані в ImageJ mailing list. І я один з них.

Макроси: Для автоматизації часто повторюваних однакових операцій можна використовувати легко записувані макроси. Що б згенерувати код макросу необхідно тільки скористатися command recorder, при необхідності потім його можна підправити й відтестувати в macro debugger. Більше трьох сотень готових макросів доступно на офіційному сайті програми.

Плагіни: Плагіни дозволяють ще більше підвищити функціональність ImageJ, програма забезпечена власним текстовим редактором для редагування вихідного тексту макросу й Java компілятором. Більш п'ятисот уже готових плагінів доступно на офіційному сайті програми й на деякі інших спеціалізованих сайтах. Плагіни дозволяють одержувати доступ до графічних (і відео) файлів самих екзотичних форматів, виконувати морфологічні операції, згортку, обробку різними фільтрами, вейвлет оброблення, перетворювати в різні формати, працювати з 3-, 4- і т.д. вимірах.

Інструмент розроблювача: Бібліотека ImageJ дозволяє не тільки розширювати функціональність програми за рахунок написання плагінів, але на її основі можна розробляти аплети, сервлети або самостійні застосування.

Швидкість:ImageJ спростує представлення багатьох про неповороткість програм на Java. Деякі бібліотеки на C++ дозволяють обробляти зображення з такою ефективністю. За заявою на офіційному сайті вона здатна відфільтрувати

зображення розміром 2048x2048 за 0.1 seconds - 40 мільйонів пікселів за секунду!

Типи даних: 8-bit grayscale або індексований колір, 16-bit unsigned integer, 32-bit floating-point і RGB колір.

Формати файлів: Форматом за замовчуванням для даної програми є TIFF (без стиснення). У її базовий функціонал закладені можливості відкриття й збереження файлів у форматах GIF, JPEG, BMP, PNG, PGM, FITS, Open DICOM. Підтримка відкриття й збереження в множині інших форматів підтримується за допомогою спеціальних плагінів.

Вікно зображення: Кожне зображення відкривається у власному вікні з можливістю зміни масштабу від 1:32 до 32:1. Якщо зображення не міститься повністю у вікні доступна функція скролінга. Усі функції аналізу працюють із зображеннями представленими в будь-якому масштабі.

Виділення: Реалізовано різні способи виділення ділянок зображення: прямокутне, еліпс, полігон або "вільне" виділення, "чарівна паличка", виділення лінії й точки. Область виділення можна надалі редагувати, інвертувати, зберігати й застосовувати до будь-якого іншого зображення. В області виділення можна проробляти будь-які операції по аналізі зображення, заливанню й очищенню, фільтрації й практично будь-якої іншої функції, доступної для всього зображення.

Покращення зображення: У базовій версії підтримується розмиття, підвищення різкості, виявлення границь, медіанний і інші лінійні й нелінійні фільтри, граничний фільтр, як для 8-bit, так і для кольорових RGB зображень. Інтерактивне й автоматичне підстроювання яскравості й контрасту. Можливість поділу кольорового зображення на канали, обробка кожного каналу окремо й складання обробленого кольорового зображення.

Геометричні операції: Масштабування, кадрування, довільне обертання, дзеркальне відображення (вертикальне й горизонтальне).

Аналіз: Вимірювання площі, середньої яскравості ділянки або цілого зображення, стандартного відхилення, мінімуму й максимуму. Вимірювання довжин і кутів з автоматичним перетворенням з пікселів у реальні одиниці

виміру - міліметри або мікрони, наприклад. Калібрування з використанням стандартних зразків щільності. Гістограми й профілі зображень.

Редагування: Копіювання/вирізання/вставка виділеної частини або всього зображення. Вставка з використанням операторів AND, OR, XOR або режиму «Blend». Додавання тексту, стрілок, прямокутних, овальних і полігональних елементів, а також цілих зображень. У свій час я був уражений багатством математичних операцій, які можна без усяких зусиль зробити із зображеннями.

Робота з кольором: Поділ 32-бітного кольорового зображення на RGB або HSV канали. Об'єднання 8-бітних каналів у кольорове зображення. Конвертування між різними типами даних, переклад RGB зображення в 8-бітне індексоване. Застосування псевдо-кольорових палітр (LUT) до півтонових чорно-білим зображенням.

Стеки: Стек можна представити як стопку зображень, що мають однаковий розмір і формат, розташованих в одному вікні. У вигляді стека в ImageJ можна відкрити .avi відео, при цьому кожний кадр буде представлений у вигляді окремій картинці в стеці. Потім зі стеком можна здійснити самі різні операції, які тільки доступні для окремого зображення, у тому числі й з використанням виділення, при цьому виділити можна той самий ділянка на всіх зображеннях стека, а для виконання операції досить один раз викликати необхідну.

Розглянемо компоненти прототипу програми.

Блок «Отримання зображення» забезпечує інтерфейс вводу зображення в програму, перекодування у необхідний формат, додаткову обробку вхідного зображення. Даний модуль включає наступні процедури: LoadToMemory, LoadTWAIN та функцію PictureVisibl. Функція PictureVisibl здійснює перевірку на наявність зображення в робочій області програми. Якщо значення функції «false» то робота програми блокується. Основний формат графічних файлів, з якими працює програма, «*.bmp», проте для зручності користування програмою були розроблені функції перекодування зображень з інших форматів. Оскільки формат «*.jpg» на сьогоднішній день є одним з найпоширенішим, то в програмі реалізовано перекодування з формату «*.jpg» в формат «*.bmp». Для отримання

додаткової статистичної інформації необхідно отримати матрицю значень яскравостей зображення, для цього зображення перекодовується з базису RGB в базис HLS (FontColorSell(integer; integer; integer)). Після отримання зображення від давача, можна провести корекцію зображення (на лаштування контрасту, яскравості, кольорової гами, тощо), а також будується гістограма яскравостей. Результатом роботи даного модуля є відкоректоване вхідне зображення отримане з давача (відеореєструюча апаратура, жорсткі носії даних).

Блок «Сегментація та виділення об'єктів» – набір процедур та функцій, що реалізує процес ручної сегментації (ручного виділення об'єктів). Процес виділення об'єктів інтересу проходить в трьох режимах: попіксельне, проходження контуром області, виділення за ключовими точками. Підчас попіксельного виділення користувач виділяє кожний піксель, що належить об'єкту. Перевагою даного підходу є висока точність виділення об'єктів, недоліком тривалість. Підчас проходження контуром області користувач проводить криву границею об'єкт інтересу, після чого програма в автоматичному режимі замальовує всі внутрішні точки об'єкту. Перевагою даного підходу є швидкість та зручність, недоліком відносно великі похибки сегментації. Даний недолік можна усунути за допомогою додаткової попіксельної корекції. Для виділення об'єкту за ключовими точками користувач повинен в ручному режимі виділити точки які в найбільшій мірі характеризується границя об'єкта. На основі ключових точок програма в автоматичному режимі проводить апроксимацію контуру прямими, при цьому отримується крива, що співпадає з контуром об'єкту. Перевагою даного підходу є простота, до недоліків слід віднести похибки апроксимації та велику залежність від початкової розмітки контуру об'єктів. Результатом роботи модуля є масив виділених об'єктів.

Блок «Обчислення інформативних ознак» – обчислення метричних характеристик об'єктів. На основі даних про периметр (довжину контуру) та площі (сума точок, що належать області) проводиться аналіз на інформативність області. Якщо інформативність області низька, то дану область прирівнюють до фону, якщо велика, то область починає розглядатися як об'єкт. Для визначення

додаткових метричних ознак проводиться аналіз контуру, наприклад пошук двох найвіддаленіших точок. Результатом роботи модуля є масив метричних ознак (проща, периметр, компактність, тощо).

Блок «Формування звітів та виводу інформації» – проводиться групування, форматування та вивід (власними засобами системи або за допомогою зовнішніх програмних засобів) результати роботи програми. Форматування звітів налаштовується в залежності від потреб користувача або досліджуваної задачі (стиль шрифтів (діаграм, графіків), тип даних (поля даних), пристрій виводу, тощо).

Спроектовано інтуїтивно зрозумілий інтерфейс (рисунок 3.1), що окрім зручності забезпечує швидке опановування програмою користувачами без досвіду роботи не тільки з даною програмою, але і з схожими програмними засобами.

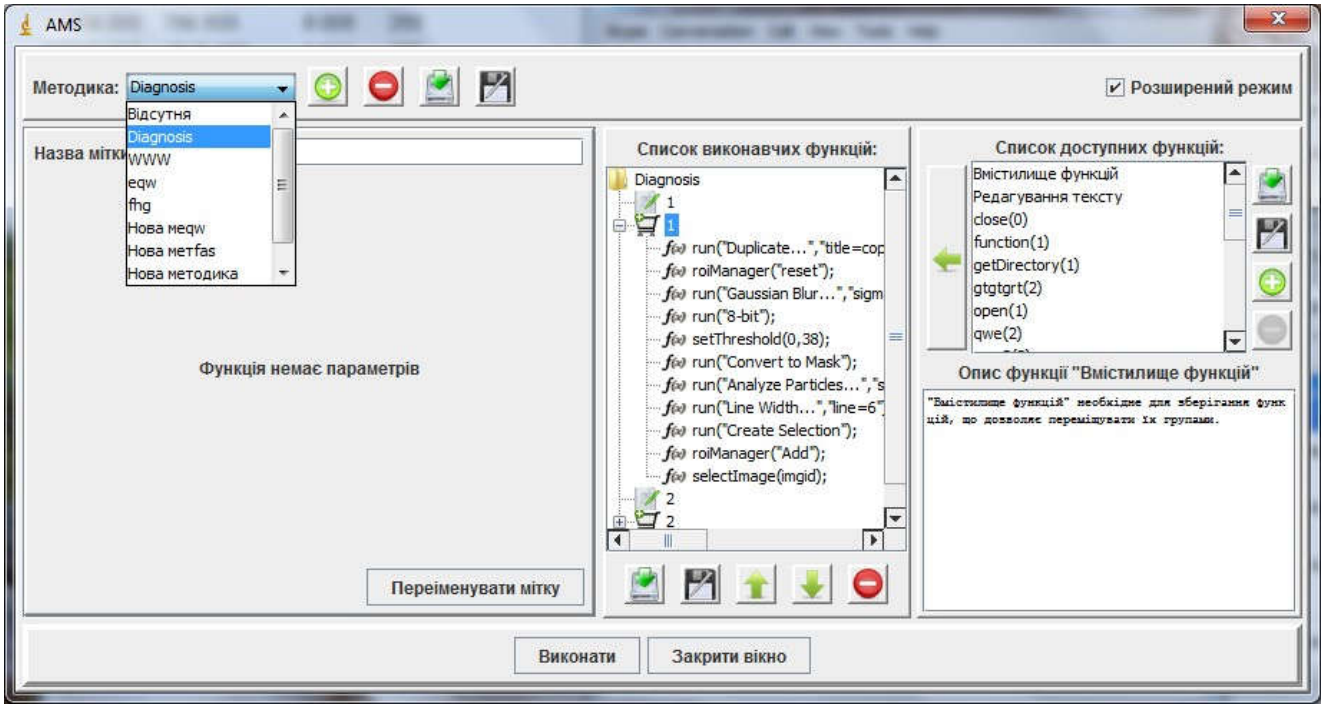


Рисунок 3.1 – Головне вікно програмної системи

Аналіз зображень та процедуру навчання класифікаторів автоматизовано за допомогою мови сценаріїв-макросів. Рекордер – спеціальний редактор, що дозволяє фіксувати всі макрокоманди, які виконуються в цей момент. Для написання сценарію в полі Record користувач змінює Macro на Plugin. Усі

команди, які будуть виконуватися в програмі ImageJ, будуть записуватися в рекордері. Наприкінці роботи з макросом користувач збереже його у файл, і при запуску цього файлу наступного разу всі команди будуть виконуватися саме в тій послідовності, яку задали.

Python – високорівнева мова програмування загального призначення, орієнтована на підвищення продуктивності програміста й читаності коду. Синтаксис ядра Python мінімалистичний. У той же час стандартна бібліотека включає великий об'єм корисних функцій.

Python підтримує кілька парадигм програмування, у тому числі структурне, об'єктно-орієнтоване, функціональне, імперативне й аспектно-орієнтоване. Основні архітектурні риси - динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки виключень, підтримка багатопоточних обчислень і зручні високорівневі структури даних. Код організовується у функції й класи, які можуть поєднуватися в модулі, котрі можуть бути об'єднані в пакети.

Перевагою є те, що підключивши спеціальні модулі, є можливість запуску програми на Web серверах, що дозволить збільшити область застосування в багато разів.

Графічний інтерфейс буде реалізований з використанням бібліотеки PyQT. PyQT – набір «прив'язок» графічного фреймворка Qt для мови програмування Python, виконаний у вигляді розширення Python. Pyqt працює на всіх платформах, підтримуваних Qt: Linux і інші Unix-Подібні ОС, Mac OS X і Windows. Pyqt також містить у собі Qt Designer (Qt Creator) – дизайнер графічного інтерфейсу користувача. Програма pyuic генерує Python код з файлів, створених в Qt Designer. Це робить PyQT дуже корисним інструментом для швидкого прототипування. Крім того, можна додавати нові графічні елементи керування, написані на Python, в Qt Designer.

PyQT має більше 600 класів, більш 6000 функцій і методів:

- існуючий набір віджетів графічного інтерфейсу;
- стилі віджетів;
- доступ до баз даних за допомогою SQL (ODBC, Mysql, Postgresql, Oracle);

- Qscintilla, заснований на Scintilla віджет текстового редактора;
- підтримку інтернаціоналізації;
- парсер XML;
- підтримку SVG;
- інтеграцію з Webkit, движком рендеринга HTML;
- підтримку відтворення відео й аудіо.

Для створення шаблонів форми, буде використовуватися Qt Designer.Qt Designer є крос-платформним компоновщиком макетів і форм графічного інтерфейсу користувача. Він дозволяє швидко спроектувати віджети й діалоги, використовуючи екранні форми з використанням тих же віджетів, які будуть використовуватися в застосуванні. Форми, створені з Qt Designer, є повністю функціональними, а також можуть бути переглянуті в режимі реального часу.

Для побудови графіків, буде використовуватися графічна бібліотека Matplotlib. Matplotlib – бібліотека мовою програмування Python для візуалізації даних двовимірної (2D) графікою. Одержувані зображення можуть бути використані в якості ілюстрацій у публікаціях. Matplotlib поширюється на умовах BSD- подібної ліцензії. Генеровані в різних форматах зображення можуть бути використані в інтерактивній графіці, у наукових публікаціях, графічному інтерфейсі користувача, веб-застосуваннях, де потрібне побудова діаграм. Matplotlib. Пакет підтримує багато видів графіків і діаграм:

- графіки;
- діаграми розкиду;
- стовпчасті діаграми й гістограми;
- кругові діаграми;
- стовбур-аркуш діаграми;
- контурні графіки;
- поля градієнтів;
- спектральні діаграми.

Користувач може вказати осі координат, решітки, додати написи й пояснення, використовувати логарифмічну шкалу або полярні координати.

Нескладні тривимірні графіки можна будувати за допомогою набору інструментів. Є й інші набори інструментів: для картографії, для роботи з Excel, утиліти для GTK і інші.

За допомогою Matplotlib можна робити й анімовані зображення. Типові підтримувані формати: Encapsulated Postscript (EPS); Enhanced Metafile (EMF); JPEG; PDF; PNG; Postscript; RGBA («сирий» формат); SVG; SVGZ; TIFF.

Для завантаження й перетворення зображень, відмінно підходить бібліотека Python Imaging Library (скорочено PIL). Можливості бібліотеки:

– підтримка бінарних, півтонових, індексованих, повнобарвних і CMYK зображень;

– підтримка форматів BMP, EPS, GIF, JPEG, PDF, PNG, PNM, TIFF і деяких інших на читання й запис;

– підтримка множини форматів (ICO, MPEG, PCX, PSD, WMF і ін.) тільки для читання;

– конвертування зображень із одного формату в інший;

– редагування зображень (використання різних фільтрів, масштабування, малювання, матричні операції і т.д.);

– використання бібліотеки з Tkinter і PyQT.

Використовувані засоби для виконання роботи: Мова програмування Python. Причини вибору цієї мови:

– мова не пов'язана з якою-небудь однієї операційною системою або машиною;

– мова підтримує основні парадигми програмування, які потрібні для виконання даної роботи (ООП, Функціональне програмування, Процедурне програмування);

– код легко читається;

– множина корисних бібліотек.

Бібліотека Numpy мови Python додає підтримку великих багатомірних масивів і матриць, разом з великою бібліотекою високорівневих і математичних функцій для операцій із цими масивами, які працюють досить швидко за рахунок використання вставок на мовах: C, C++ і Fortran. Бібліотека

алгоритмів комп'ютерного зору OpenCV, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом.

Узагальнена структура програмного засобу відображає інструментальні бібліотеки котрі використовуються при реалізації програмного забезпечення (рисунок 3.2).

Модуль SciPy містить модулі для оптимізації, інтегрування, спеціальних функцій, обробки сигналів, обробки зображень, генетичних алгоритмів, розв'язування звичайних диференціальних рівнянь та інших задач, які розв'язуються в науці і при інженерній розробці. Модуль NumPy надає підтримку великих багатовимірних масивів і матриць, бібліотеку високорівневих математичних функцій для операцій з цими масивами. Модуль CV дозволяє підключати в сценарій бібліотеку опрацювання зображень OpenCV.

Модуль scikit-image це бібліотека обробки зображень, яка реалізує алгоритми і утиліти для використання в науково-дослідному, освітньому і промисловому ПЗ. Модуль scikit-learn реалізує основні методи машинного навчання: алгоритми навчання з учителем (Supervised Learning), алгоритми навчання без вчителя (Unsupervised Learning). Модуль matplotlib використовується для візуалізації наукових даних у вигляді графіків.

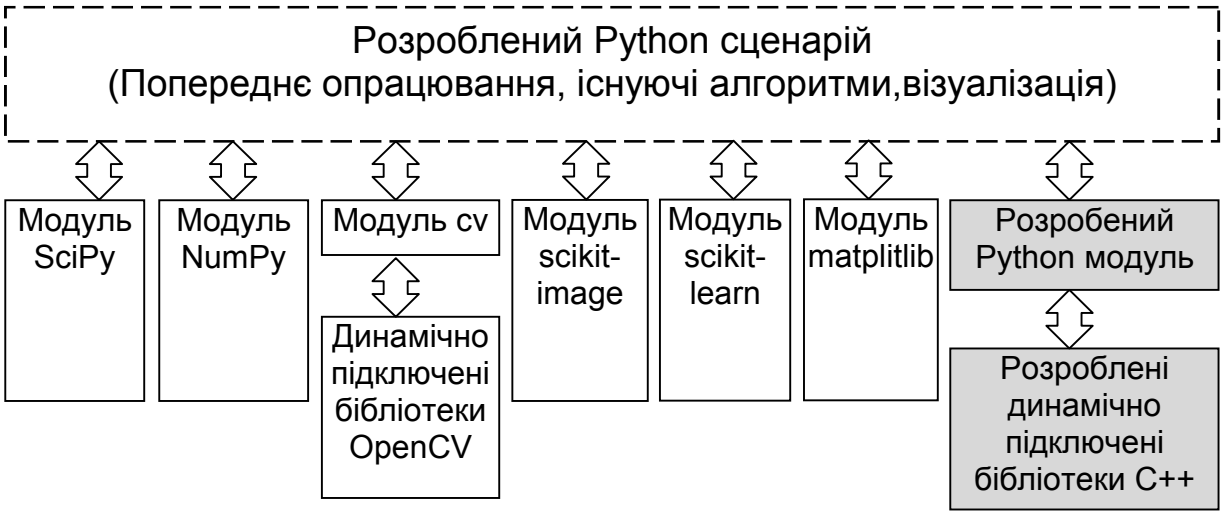


Рисунок 3.2 – Структура програмного засобу оброблення асиметричних зображень

Для програмної реалізації використано структурний та об’єктно-орієнтований підходи. Окремі сутності предметної області представляють у

вигляді класів, взаємодія між ними проходить чітко визначеним способом через інтерфейс класу і технологію наслідування.

3.2 Реалізація дескриптора текстурних блоків

Реалізуємо функцію порівняння текстур на основі локальних бінарних шаблонів (ЛБШ) мовою Python. Опишемо обчислення ЛБШ дескриптора. По-перше, ми перетворюємо вхідне кольорове зображення в шкалу градацій сірого, оскільки ЛБШ працює над зображенням рівнів сірого. Для кожного пікселя в зображенні сірого, навколо поточного пікселя вибирається сусідство а потім ми обчислюємо значення ЛБШ для пікселя, користуючись сусідством. Після обчислення ЛБШ значення поточного пікселя, ми модифікуємо відповідне розташування пікселя в зображенні-масці ЛБШ (Воно такої ж висоти і ширини як вхідне зображення.) зі значенням ЛБШ, обчисленим, як показано нижче. У зображенні (рисунок 3.3), ми розглядаємо 8 сусідніх пікселів.

Щоб обчислити значення ЛБШ для пікселя в зображенні рівнів сірого, порівнюємо центральне значення пікселя з значеннями сусідніх пікселів. Ми можемо почати з будь-якого сусіднього пікселя а потім ми можемо перейти або в напрямку за годинниковою стрілкою, або проти але ми повинні користуватися тим же порядком для усіх пікселів. Оскільки є 8 сусідніх пікселів ми виконаємо 8 порівнянь. Результати порівнянь зберігаються в масиві 8-розрядному бінарному масиві .

Якщо поточне значення пікселя більше або рівне значення сусіднього пікселя, відповідний біт в двійковому масиві встановлюється до 1 ще, якщо поточне значення пікселя є меншим ніж значення сусіднього пікселя, відповідний біт в двійковому масиві встановлюється до 0.

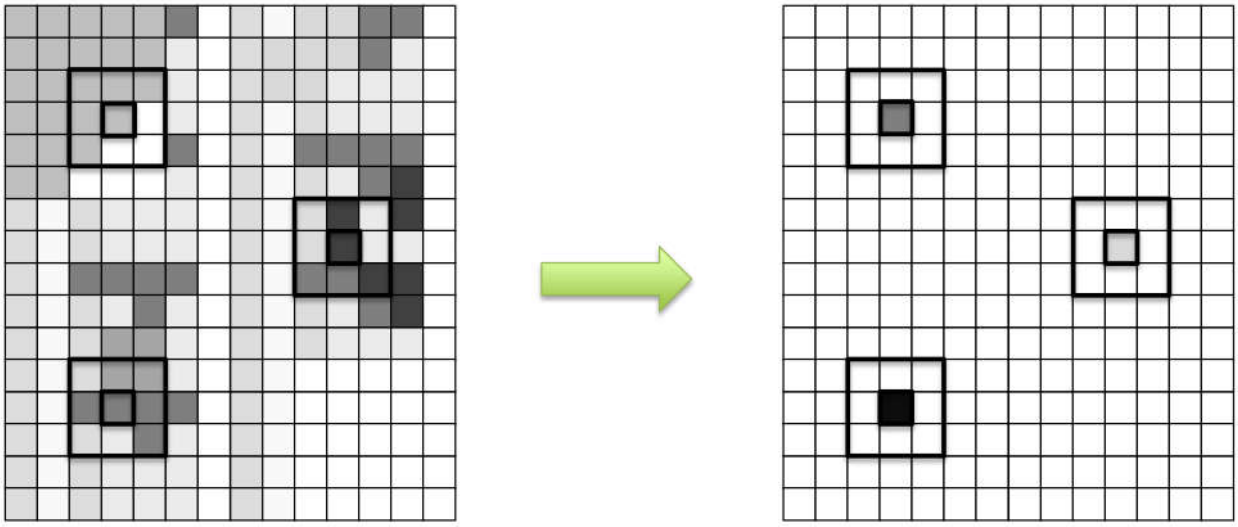


Рисунок 3.3 – Перетворення зображення в гарадціях сірого в ЛБШ Маску *

Процес показано на зображенні нижче (рисунок 3.4). Поточний (центральний) піксель має значення 7. Ми починаємо порівняння з сусіднього пікселя, де мітка 0. Значення сусіднього пікселя з міткою 0 складає 2. Оскільки це менше ніж поточне значення пікселя 7, ми скидаємо 0-й біт в 8 масиві до 0. Ми потім виконуємо ітерацію в напрямку проти годинникової стрілки. Наступний піксель із міткою 1 має значення 7, що дорівнює поточному значенню пікселя, тому ми встановлюємо перший розряд в 8 розрядному масиві в 1. Ми продовжуємо переміщатися на наступний сусідній піксель поки не досягаємо 8-го сусіднього пікселя. Потім 8-бітний шаблон перетворюється в десяткове число котре зберігається у відповідній комірці ЛБШ маски (рисунок 3.6).

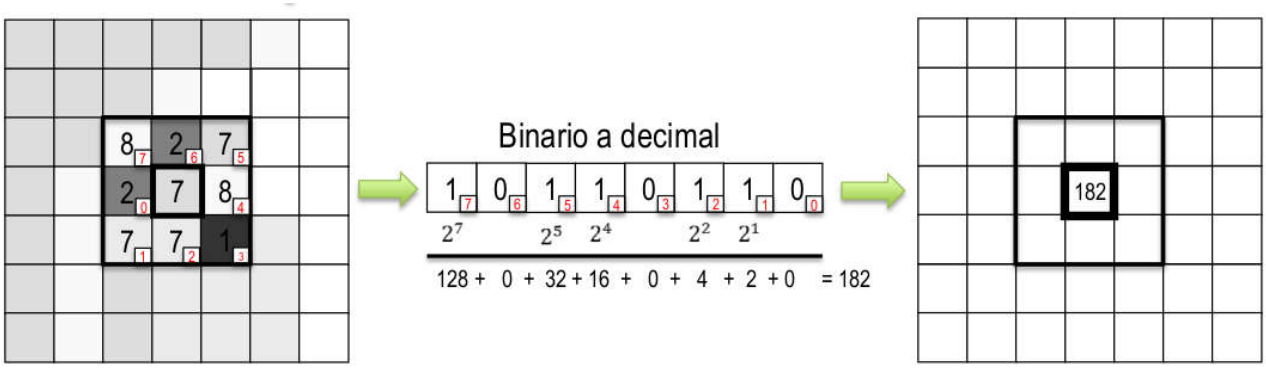


Рисунок 3.4 – Обчислення значень ЛБШ

Після обчислення маски, обчислюємо гістограму. Значення ЛБШ маски знаходяться в діапазоні від 0 до 255, тому ЛБШ Дескриптор буде розміром 1x256. Ми потім нормалізуємо ЛБШ гістограму. Нижче показано узагальнену схему алгоритму.

1. Завантажити кольорове зображення.

2. Перетворити в зображення рівнів сірого.

3. Обчислити ЛБШ маску.

4. Обчислити ЛБШ Гістограму і нормалізувати.

Одна з переваг ЛБШ - це інваріантність до освітлення і переносу. Ми вибрали 8-звязне сусідство, проте більшість реалізацій використовують кругове сусідство як показано нижче. Реалізуємо алгоритм із круговим сусідством (рисунок 3.5).

Використаємо модуль cv2, щоб відкривати зображення, виконувати перетворення колірного простору, і т.п. Використаємо модуль os щоб виконувати операції над шляхом у файловій системі (path). Розширимо функцію local_binary_pattern із модуль skimage.feature, щоб обчислити ЛБШ маску.

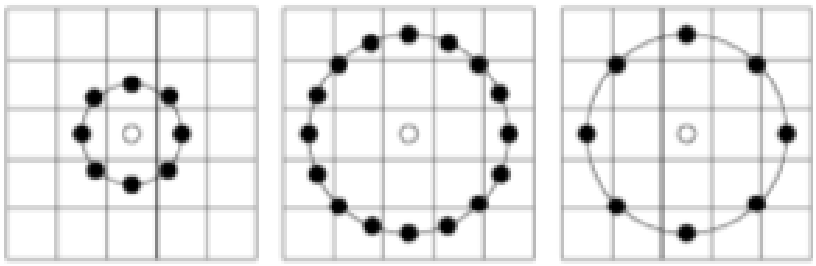


Рисунок 3.5 – Сусідство у вигляді кола

На основі ЛБШ маски ми обчислимо гістограму, користуючись функцією scipy.stats.itemfreq а потім ми скористаємося функцією sklearn.preprocessing.normalize, щоб нормалізувати гістограму. Пакет cvutils корисний для роботи з комп'ютерною системою технічного зору і пакетами обробки зображень. Нарешті, csv модуль забезпечує функціональність для розбору текстових файлів.

```
%pylab inline --no-import-all
# OpenCV
import cv2
# До виконання маніпуляцій шляху
import os
# функція Локальна бінарного шаблону
from skimage.feature import local_binary_pattern
# Щоб обчислити нормалізовану гістограму
from scipy.stats import itemfreq
from sklearn.preprocessing import normalize
# Utility пакет
import cvutils
# для читання класу з файлу
import csv
```

Підготовка 6 навчальних зображень, створення- файлу class_train.txt, який виглядає наступним чином.

```
rock-1.jpg 0
rock-2.jpg 0
grass-1.jpg 1
grass-2.jpg 1
checkered-1.jpg 2
checkered-2.jpg 2
```

Кожен рядок складається з імені зображення, завершеного міткою класу (таблиця 3.1).

Таблиця 3.1 – Мітки класів зображень

Клас	Мітка
Камінь	0
Трава	1
Текстура	2

Завжди краще підготувати структуру, яка може бути використана пізніше. Наприклад, ці мітки можуть бути корисні, для класифікації якщо використати SVM.

Запишемо код, щоб зберігати шлях усіх зображень в навчальному наборі.

```
1 # зберегти шлях навчальних зображень в train_images
2 train_images = cvutils.imlist("../data/lbp/train/")
3 # Словник, що містить шляхи зображення як ключі і мітку як значення
4 train_dic = {}
5 with open('../data/lbp/class_train.txt', 'rb') as csvfile:
6     reader = csv.reader(csvfile, delimiter=' ')
7     for row in reader:
8         train_dic[row[0]] = int(row[1])
```

У рядку 2, ми зберігаємо усі шляхи навчальних зображень в списку train_images. У рядку 4, ми створюємо словника train_dic, який міститиме ім'я зображення і відповідну мітку класу. З 5 по 8 рядок, ми читаємо рядки з документ class_train.txt описаного вище а потім ми додаємо пару ключ-значення (ім'я зображення, мітка класу) до train-dic.

Об’єкт train_images містить наступні шляхи зображення

['../data/lbp/train/rock-1.jpg', '../data/lbp/train/rock-2.jpg', '../data/lbp/train/grass-2.jpg', '../data/lbp/train/checkered-2.jpg', '../data/lbp/train/checkered-1.jpg', '../data/lbp/train/grass-1.jpg']

Об’єкт train-dic містить наступні пари ключ-значення

{'grass-1.jpg': 1, 'rock-1.jpg': 0, 'checkered-2.jpg': 2, 'rock-2.jpg': 0, 'checkered-1.jpg': 2, 'grass-2.jpg': 1}

Наступний крок - обчислити нормалізовані ЛБШ гістограми навчальних зображень.

```
1 # Скласти список для зберігання Гістограм ЛБШ, адреси зображень і відповідної мітки
2 X_test = []
3 X_name = []
4 y_test = []
5
6 # Для кожного зображення в навчальній множині обчислити гістограму ЛБШ
7 # і модифікують випробування X_, ім'я X_i y_випробування
8 for train_image in train_images:
10     im = cv2.imread(train_image)
12     im_gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
13     radius = 3
14     # Число точок, які розглядаються як сусіди
15     no_points = 8 * radius
16     # використано однорідний ЛБШ
17     lbp = local_binary_pattern(im_gray, no_points, radius, method='uniform')

        Обчислити і нормалізувати гістограму.

19     x = itemfreq(lbp.ravel())
21     hist = x[:, 1]/sum(x[:, 1])

        Зберегти шляхи зображень.

22     # Приєднати шлях зображення в X_name
23     X_name.append(train_image)
24     # Приєднайте гістограму до X_name
25     X_test.append(hist)
26     # Приєднайте класу мітку в y_випробуванні
27     y_test.append(train_dic[os.path.split(train_image)[1]])
```

Відобразити навчальні зображення.

```
30 nrows = 2
31 ncols = 3
32 fig, axes = plt.subplots(nrows,ncols)
33 for row in range(nrows):
34     for col in range(ncols):
35             axes[row][col].imshow(cv2.cvtColor(cv2.imread(X_name[row*ncols+col]),
cv2.COLOR_BGR2RGB))
36         axes[row][col].axis('off')
37
axes[row][col].set_title("{}".format(os.path.split(X_name[row*ncols+col])[1]))
```

Розглянемо код. У рядку 2-4 створюємо 3 списки:

1. x_test зберігає нормалізовані Гістограми ЛБШ;
2. x_name зберігає адресу зображень;
3. y_test зберігає відповідну мітки класу.

Кожен індекс в усіх 3 списках відповідає одному тому ж зображенню.

В рядках з 8 по 27 в циклі по усіх зображеннях в навчальній множині обчислюємо нормалізовані гістограми ЛБШ для навчальних зображень. Спочатку в рядку 10, ми читаємо поточне зображення, користуючись функцією cv2.imread. Потім перетворюємо зображення в рівні сірого оскільки ЛБШ працює над зображенням рівнів сірого. У рядку 17 обчислюємо маску ЛБШ. Встановлюємо радіус сусідства 3, а число точок (пікселів) дорівнює 24. Як тільки створимо маску, ми обчислюємо гістограму ЛБШ в рядку 19 і нормалізуємо в рядку 21. Потім, приєднуємо гістограму до списку X_testin в рядку 25. Також приєднуємо ім'я зображення до списку X_name і мітку класу зображення до y_test.

Отримати тестові зображення. Щоб перевірити виконання алгоритму ЛБШ, я знову взяв 3 зображення кожного класу не присутніх в навчальній множині. Читатимемо шляхи 3-х зображень і приєднуємо їх до списку

test_images. Ми створимо словник test_dic, подібний до словника train_dic, створеного вище.

```
1 # Зберегти шлях тестових зображень у випробувальних_зображеннях
2 test_images = cvutils.imlist("../data/lbp/test/")
3 # Словник із шляхами зображення у вигляді ключів, мітки як значення
4 test_dic = {}
5 with open('../data/lbp/class_test.txt', 'rb') as csvfile:
6     reader = csv.reader(csvfile, delimiter=' ')
7     for row in reader:
8         test_dic[row[0]] = int(row[1])
```

Вміст test_images і test_dic наступний

```
# test_images
['../data/lbp/test/rock-1.png', '../data/lbp/test/checkered-1.jpg', '../data/lbp/test/grass-1.jpg']
# test_dic
{'rock-1.png': 0, 'grass-1.jpg': 1, 'checkered-1.jpg': 2}
```

Обчислити відстань Chi-squared для кожного тестового зображення. Обчислюємо нормалізовані ЛБШ гістограми для кожного зображення в тестовій множині а потім порівнюємо нормалізовані ЛБШ гістограми навчальних зображень з ними, користуючись значенням відстані Хі-квадрат. Потім сортуємо результати, засновані на відстані і відображаємо результати в сортованому порядку. Чим нижча відстань Хі-квадрат, тим краща відповідність.

```
1 for test_image in test_images:
2     # Читайти зображення
3     im = cv2.imread(test_image)
4     # Convert , оскільки ЛБШ працює з зображенням рівнів сірого
5     im_gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
6     radius = 3
```

```
7   # Число точок, які розглядаються як neighbourers
8   no_points = 8 * radius
9   # використано Однорідний ЛБШ
10  lbp = local_binary_pattern(im_gray, no_points, radius, method='uniform')
```

Обчислити і нормалізувати гістограму

```
12  x = itemfreq(lbp.ravel())
14  hist = x[:, 1]/sum(x[:, 1])
```

Показати зображення-запит.

```
16  cvutils.imshow("** Query Image -> {}**".format(test_image), im)
17  results = []
18  # For each image in the training dataset
19  # Calculate the chi-squared distance and the sort the values
20  for index, x in enumerate(X_test):
21          score = cv2.compareHist(np.array(x, dtype=np.float32), np.array(hist, dtype=np.float32), cv2.cv.CV_COMP_CHISQR)
22      results.append((X_name[index], round(score, 3)))
23  results = sorted(results, key=lambda score: score[1])
```

Показати на екрані результат опрацювання.

```
25  nrows = 2
26  ncols = 3
27  fig, axes = plt.subplots(nrows,ncols)
28  fig.suptitle("** Scores for -> {}**".format(test_image))
29  for row in range(nrows):
30      for col in range(ncols):
```

```
31
axes[row][col].imshow(cv2.cvtColor(cv2.imread(results[row*ncols+col][0]),
cv2.COLOR_BGR2RGB))
32          axes[row][col].axis('off')
33          axes[row][col].set_title("Score {}".format(results[row*ncols+col][1]))
```

В рядках з 1 по 14, обчислюємо нормалізовані гістограми ЛБШ як у випадку з навчальними зображеннями. У рядку 21 обчислюємо Відстань Хі-квадрат між тестовим зображенням та з усіма навчальними зображеннями по черзі, користуючись функцією cv2.compareHist. Потім сортуємо отримані значення в рядку 22. В рядках з 25 по 33, ми відображаємо навчальні зображення з відповідними значеннями відстаней.

3.3 Класифікація супутникових зображень

Для експериментального дослідження було виділено 8 основних класів текстур: «вода» (річки, озера, моря), «Грунт», «болото», «ліс», «степ», «гори», «асфальт» (великі заасфальтовані ділянки), «будівлі» (міста, населенні пункти).

У таблиці 2.1 наведенні порівняльні результати роботи двокрокового алгоритму і однокрокових алгоритмів для бібліотек текстур в градаціях сірого і в колірному просторі RGB. Тестування було проведено на 50 довільно взятих зразках текстур.

Таблиця 2.1 - Точність результатів роботи алгоритмів класифікації

	Color Texture Base	Greyscale Texture Base	Two-step Texture Base
Точність	86%	92%	95%

Розроблений алгоритм володіє кращою точністю ніж одно крокові від 2 до 9 процентів.

Також на рисунку 3.6 показано результати класифікації текстури на зображенні розробленими алгоритмами.

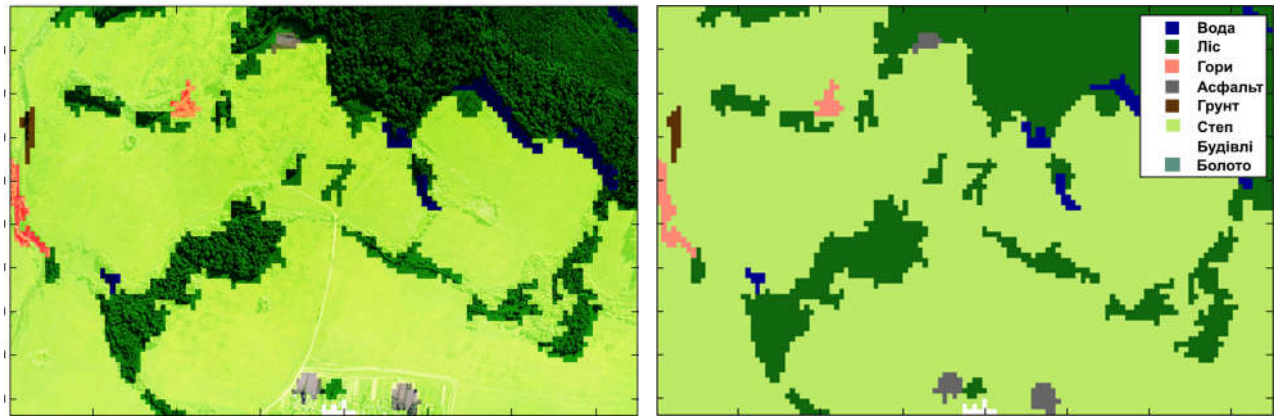


Рисунок 3.6 – Результати текстурного аналізу зображень за розробленим алгоритмом

Перевагою розробленого алгоритму є інформаційна ємність отриманих за допомогою нього результатів обробки. Інформація про кожну окрему область дозволяє виділити її характерні особливості, на основі чого в подальшому можлива побудова 3D-моделі місцевості.

ВИСНОВКИ

В результаті виконання дипломної роботи одержані наступні результати:

1. Проаналізовано структуру та функції систем аерокосмічного моніторингу. Встановлено актуальність задачі класифікації об’єктів на зображеннях.

2. В якості класів був обраний скорочений набір типів місцевості, результати не відображають всього різноманіття існуючих типів місцевості. Це може бути вирішено шляхом розширення числа класів. Але більш точне формування нових класів і збір зразків з відповідних текстур неможливі без допомоги фахівців-екологів і картографів.

3. В результаті аналізу текстурних ознак на основі локальних бінарних шаблонів модифіковано ЛБШ-оператор і забезпечено його інваріантність щодо повороту, що дозволило підвищити точність опису текстури.

4. Двокроковий підхід на етапі класифікації дозволив збільшити ефективність роботи зі змішаною базою текстур, що складається з кольорових зразків і зразків в градаціях сірого та уніфікувати роботу алгоритму для користувача бібліотек текстур.

5. Розроблено алгоритм, на основі якого написана програма, що дозволяє аналізувати зображення на предмет визначення представлених на них типів місцевості. Було проведено тестування її роботи на кількох знімках. В результаті проведених експериментів було встановлено, що точність класифікації зросла на 3-9%.

6. Перевагою розробленого алгоритму є інформаційна ємність отриманих за допомогою нього результатів обробки. Інформація про кожну окрему область дозволяє виділити її характерні особливості, на основі чого в подальшому можлива побудова 3D-моделі місцевості.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Галян А.А. Комбінування ознак при сегментації зображень / А.А. Галян, І.В. Гордій, В.О. Лазарчук // Актуальні задачі сучасних технологій: зб. тез доповідей міжнар. наук.-техн. конф. Молодих учених та студентів, (Тернопіль, 28–29 листоп. 2018) – Тернопіль: ТНТУ, 2018. – с. 32.

2. Блаттер К. Вейвлет-анализ. Основы теории: Учебное пособие для вузов: пер. с нем. / К. Блаттер; пер. Т. Э. Кренкель, ред. пер. А. Г. Кюркчан. – М.: Техносфера, 2006. – 2711 с.

3. Малла С. Вейвлети в обробці сигналів: Навчальний посібник для вузів / С. Малла; пер. Я.М. Жилейкин. – М.: Мир, 2005. – 671 с.

4. Ng R. Fourier slice photography / Ren Ng // ACM Transactions on Graphics - 2005 - № 24(3). - P. 735–744.

5. Singh M. SAR Image Classification Using PCA and Texture Analysis / Mandeep Singh, Gunjit Kaur // Proceedings of the International Conference “Information Technology and Mobile Communication” (AIM). – Nagpur-Maharashtra- India, 2011. – P. 435-439.

6. Мельник Г. М. Зменшення простору текстурних ознак гістологічних зображень за допомогою методу головних компонент // Моделювання та інформаційні технології. - 2016. - Вип. 77. - С. 176-180.

7. Schaffalitzky F. Geometric grouping of repeated elements within images / F. Schaffalitzky, A. Zisserman // In Proc. 9th British Machine Vision Conference (BMVC), Southampton, September 1998 – Southampton, UK: Springer-Verlag, 1998. – С. 13-22.

8. Liu Y. A Computational Model for Periodic Pattern Perception Based on Frieze and Wallpaper Groups / Yanxi Liu, Robert T. Collins, Yanghai Tsin // IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) - 2004. - Т. 1, № 26. - С. 354-371.

9. Tuceryan M. Texture Segmentation Using Voronoi Polygons / M. Tuceryan, A. K. Jain // IEEE Transactions on Pattern Analysis and Machine Intelligence - 1990. - Т. 2, № 12. - С. 211-216.

10. Liu Y. Near-regular Texture Analysis and Manipulation / Yanxi Liu, Wen-Chieh Lin, James H. Hays // ACM Transactions on Graphics (SIGGRAPH 2004) – 2004. – P. 368-376.

11. Wolter F. Spatio-temporal representation and reasoning based on RCC-8 / Frank Wolter, Michael Zakharyaschev // In Proceedings of the seventh Conference on Principles of Knowledge Representation and Reasoning (KR2000) – Breckenridge, Colorado, USA: Morgan Kaufmann, 2000. – P. 3-14.

12. Rajkovic N. Comparison of Monofractal, Multifractal and gray level Co-occurrence matrix algorithms in analysis of Breast tumor microscopic images for prognosis of distant metastasis risk / N. Rajković, D. Kolarević, K. Kanjer, N. T. Milosević, D. Nikolić-Vukosavljević, M. Radulovic // Biomedical Microdevices. - Springer, 2016. - Vol. 5, Num. 18. - P. 83.

13. Saito A. A novel method for morphological pleomorphism and heterogeneity quantitative measurement: Named cell feature level co-occurrence matrix / A. Saito et al. // Journal of Pathology Informatics. – 2016 - Vol. 7, Num. 36. - P.315

14. Mohanty A. K. Classifying Benign and Malignant Mass using GLCM and GLRLM based Texture Features from Mammogram / A. K. Mohanty, S. Beberta, S. K. Lenka // International Journal of Engineering Research and Applications. – 2011. - Vol. 1, Issue 3. – P. 687-693.

15. Belsare A. D. Classification of breast cancer histopathology images using texture feature analysis / A. D. Belsare, M. M. Mushrif, M. A. Pangarkar, N. Meshram // 2015 IEEE Region 10 Conference – 2015. – P. 1-5.

16. Herve N. Statistical color texture descriptors for histological images analysis / N. Herve, A. Servais, E. Thervet, J. Olivo-Marin, V. Meas-Yedid // 2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro – 2015. – P. 724-727.

17. Onder D. Automated classification of cancerous textures in histology images using quasi-supervised learning algorithm / D. Onder, S. Sarioglu, B. Karacali // 2010 15th National Biomedical Engineering Meeting – 2010. – P. 1-4.

18. Riaz F. Content-Adaptive Region-Based Color Texture Descriptors for Medical Images / F. Riaz, A. Hassan, R. Nisar, M. Dinis-Ribeiro, M. Tavares Coimbra

// IEEE Journal of Biomedical and Health Informatics. – 2017. – Vol. 21, No. 1. – P. 162-171.

19. Мельник Г.М. Аналітичний огляд методів аналізу та синтезу текстурних зображень / Г.М. Мельник // Вісник Хмельницького національного університету - 2007. - Т. 2, № 1. - С. 110-114

20. Burges C. J. A tutorial on support vector machines for pattern recognition // Data Mining and Knowledge Discovery. – 1998. – Vol. 2, no. 2,- P. 121-167.

21. Burt P. J. The laplasian pyramid as a compact image code // IEEE Transactions on Communications. – 1983.– Vol. COM- 31, no. 4. – P. 532-540.

22. Chang T. Texture analysis and classification with tree-structured wavelet transform / Chang T., Jay C.-C. K. // IEEE Transactions on Image Processing.- 1993. – Vol. 2, no. 4, – P. 429-441.

23. Chapelle 0., Vapnik P. II. V. Svms for histogram based image classification // IEEE Transactions on Neural Net-works.– 1999.– Vol. 10,- P. 1055-1064.

24. Chen Y. One-class svm for learning in image retrieval / Chen Y., Zhou X. S., Huang T. S.// IEEE International Conference on Image Processing (ICIP 2001).- 2001.- p. 34-37.

25. Chitkara V. Color-based image retrieval using compact binary signatures: Tech. Rep. TR 01-08: University of Alberta Edmonton, 2001.

26. Chuang G. Wavelet descriptor of planar curves: theory and applications / Chuang G. C.-H.. Kuo C.-C. J. // IEEE Transactions on Image Processing – 1996. - January. - Vol. 5, no. 1. - P. 56-70.

27. Coifman R. R. Entropy-based algorithms for best basis selection // IEEE Transactions on Information Theory. – 1992. – March. - Vol. 38, no. 2. - P. 713-718.

28. Cover T. M. Nearest neighbor pattern classification / Cover T. M., Hart P. E. // EEE Transactions on Information Theory. – 1967. - Vol. 13, no. 1. – Pp. 21-27.

29. Cross G. R., Markov random field texture models // IEEE Transactions on Pattern Analysis and Machine Intelligence:– 1983.– Vol. PAMI-5. - Pp. 25-39.

30. Deer P. On the fusion of image features. – [Електронний ресурс]. Режим доступу http://citeseer.ist.psu.edu/162546.html.

31. Dennis T. J. Fractal modelling in image texture analysis // IEEE Proc. of Radar and Signal Processing. – Vol. 136. – 1989. - p. 227-235.

32. Do M. N. Texture similarity measurement using kullback- leibler distance on wavelet subbands // IEEE International Conference on Image Processing (ICIP-2000). - Vol. 3. - 2000. - P. 730-733.

33. Efficient and effective querying by image content: Tech. rep. / C. Faloutsos, W. Equitz, M. Flickner et al.: IBM Research, 1993.

34. An efficiently computable metric for comparing polygonal shapes / E. M. Arkin, L. Chew, D. Huttenlocher et al. // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1991. – Vol. 13. – P. 209-216.

35. Fan S. Shape representation and retrieval using distance histograms: Tech. Rep. 01-14: Department of Computing Science, University of Alberta, 2001.

36. Fast algorithm for the computation of moment invariants / M. F. Zakaria, L. J. Vroomen, P. J. A. Zsombor-Murray, J. M. H. M. van Kessel // Pattern Recognition. – 1987. - Vol. 20, no. 6. - p. 639-C43.

37. Field D. J. Relations between the statistics of natural images and the response properties of cortical cells // Journal of the Optical Society of America. – 1987. - Vol. 4, no. 12. - P. 2370-2393.

38. Fox E. A. Combination of multiple searches // 2nd Text Retrieval Conference (TREC-2). National Institute of Standards and Technology Special Publication 500-215, 1994. - p. 243-252.

39. Freeman H. In computer processing of line-drawing images // ACM Computing Surveys (CSUR). - 1974. - March. – Vol. 6. - p. 57-97.

40. Ghafoor A., Iqbal R. NKhan S. A. Modified chamfer matching algorithm // Lecture Notes in Computer Science. – 2003. – Vol. 2690. – p. 1102-1106.

41. Gotlieb C. CKreyszig H. E. Texture descriptors based on co-occurrcnce matrices // Computer Vision, Graphics and Image Processing. - 1990.– July. - Vol. 51, no. 1- p. 70-86.

42. Grosky W. An image data model / Grosky W., Stanchev P. //In Proceedings of Advances in Visual Information Systems: 4th International Conference. - 2000. - Pp. 227-243.

43. Guerin-Dugue A. Image retrieval: a first step for a human centered approach / Guerin-Dugue A., Ayache S., Berrut C. // Joint Conference of ICI, CSP and PRCM. - 2003. - p. 21-25.

44. Haddadnia J. An efficient feature extraction method with pseudo-zernike moment in RBF neural network-based human face recognition system / Haddadnia J., Ahmadi M., Faez K. // EURASIP Journal on Applied Signal Processing. – 2003.- p. 890-901.

45. Hateren J. H. V. Independent component filters of natural images compared with simple cells in visual cortex / Hateren J. H. V., der Schaaf A. V. // Transactions of Royal Society of London. – 1998. – Vol. B265. – p. 359-366.

46. Heller K. A. simple bayesian framework for content- based image retrieval / Heller K. A., Ghahramani Z. // IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006). - 2006. - p. 2110-2117.

47. Hew P. Geometric and zernike moments. - Diary, Department of Mathematics, The University of Western Australia. - 1996. - Octobcr. [Електронний ресурс]. Режим доступу: http://citeseer.ist.psu.cdu/hew96geometric.html.

48. Hoqve S. K. A new chain-code quantization approach enabling high performance handwriting recognition based on multi- classifier schemes / Hoqve S., K. Sirlantzis M. C. F. // Seventh International Conference on Document Analysis and Recognition (ICDAR'03). -Vol. 2.- 2003.- 843 p.

49. Howarth P. Evaluation of texture features for content-based image retrieval / Howarth P., Riiger S. // Proceedings of CIVR'04. - 2004. - p. 326-334.

50. Методичні рекомендації до виконання дипломної роботи з освітньо-кваліфікаційного рівня “Магістр”. Спеціальність „Комп’ютерні системи та мережі” / О.М. Березький, Л.О. Дубчак /Під ред. О.М. Березького – Тернопіль: ТНЕУ, 2013.– 47 с.