**МIНIСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Західноукраїнський національний університет**

**Факультет комп’ютерних інформаційних технологій**

Кафедра комп’ютерної інженерії

**МАЧУЛЯК Михайло Володимирович**

**«Напівавтоматична сегментація біомедичних зображень / Semi-automatic segmentation of biomedical images»**

спеціальність: 123 - Комп’ютерна інженерія

освітньо-професійна програма - Комп’ютерна інженерія

Кваліфікаційна робота

Виконав студент групи КІм-22

МАЧУЛЯК Михайло Володимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Науковий керівник:

д.т.н., Березький О.М.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кваліфікаційну роботу допущено

до захисту:

"\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ р.

Завідувач кафедри

\_\_\_\_\_\_\_\_\_\_\_\_\_\_О. М. Березький

**Тернопіль – 2021**

ЗМІСТ

[Вступ 7](#_Toc89968674)

[1 Аналіз алгоритмів сегментації зображень 11](#_Toc89968675)

[1.1 Сегментація зображень: основні поняття та принципи 11](#_Toc89968676)

[1.2 Алгоритми сегментації зображень 18](#_Toc89968677)

[1.2.1 Алгоритми порога яскравості 19](#_Toc89968678)

[1.2.2 Алгоритми центроїдного зв’язування (підключення до центру ваги) 22](#_Toc89968679)

[1.2.3 Алгоритми склеювання-розщеплювання 24](#_Toc89968680)

[1.2.4 Алгоритми розфарбовування зображень 27](#_Toc89968681)

[1.2.5 Алгоритми граничної сегментації 29](#_Toc89968682)

[1.3 Аналітична презентація програмного забезпечення для аналізу зображень 34](#_Toc89968683)

[1.4 Висновки до розділу 1 43](#_Toc89968684)

[2 Алгоритми сегментації зображень біомедичної природи 44](#_Toc89968685)

[2.1 Сегментація зображень шляхом кластеризації колірного простору 44](#_Toc89968686)

[2.2 Алгоритм сегментації за водорозподілом 49](#_Toc89968687)

[2.3 Алгоритм сегментації шляхом виділення контурів зображень 56](#_Toc89968688)

[2.4 Висновки до розділу 2 60](#_Toc89968689)

[3 Програмна реалізація алгоритмів сегментації зображень 62](#_Toc89968690)

[3.1 Узагальнена структура програмної системи 62](#_Toc89968691)

[3.2 Алгоритми роботи програмної системи 73](#_Toc89968692)

[3.3 Тестування розробленої програмної системи 87](#_Toc89968693)

[3.4 Висновки до розділу 3 92](#_Toc89968694)

[Висновки 94](#_Toc89968695)

[Список використаних джерел 95](#_Toc89968696)

[Додаток А 100](#_Toc89968697)

[Лістинг основних модулів програми 100](#_Toc89968698)

Додаток Б Довідка про використання ………………………………………………131

Додаток В Світлокопії публікацій ……………………………………………. …...132

ВСТУП

Актуальність теми. Одна з актуальних тенденцій розвитку сучасної прикладної науки – повсюдне впровадження комп'ютерних засобів у медицину. Зокрема, комп'ютерні технології з набором спеціалізованого програмного забезпечення широко використовуються для діагностики пацієнтів, швидкої обробки інформації та обміну даними, а також підготовки майбутніх фахівців [1]. На сучасному етапі поєднання прогресивних інформаційних технологій, нових методів та алгоритмів обробки, аналізу та синтезу зображень та медицини призвело до народження нової галузі – телемедицини, яка передбачає дистанційну діагностику на основі аналізу та обробки зображень клітин органів людини. Як відомо [2-4], кожен тип клітин має свої особливості: відповідну геометричну форму та характерний комбінований колір. Вибір цих функцій в автоматичному режимі – одне з основних завдань, що стоять перед розробниками алгоритмів та програмного забезпечення у цій галузі.

Одним із важливих кроків в аналізі зображення є правильний та швидкий вибір та проходження контуру зображення. Розберемо відомі алгоритми визначення зовнішнього краю зображення. Переваги алгоритму «Square Tracing» [5] – простота реалізації та швидкість, недоліки – неефективна обробка зображень з діагональними сторонами та зображень, що мають «гілки» товщиною в 1 піксель. Цей алгоритм, незважаючи на його простоту та швидкість, не може ефективно використовуватися для визначення контуру складної фігури.

Покращанням алгоритму Square Tracing є алгоритм Moore-Neighbor Tracing. Він, на відміну від попереднього, більш ефективний при ітерації зображень із довільними сторонами. Велика ефективність досягається за рахунок збільшення числа суміжних точок, що проходять процес перевірки. Переваги цього алгоритму – підвищена точність обробки та можливість обробляти складніші зображення, недоліками – підвищена обчислювальна складність та зниження продуктивності, а також можливість некоректної роботи.

Алгоритм зміщення контуру "Radial Sweep" заснований на алгоритмі "Moore-Neighbor Tracing", але має ряд поліпшень, наприклад, коли виявляється точка контуру, вона прив'язується до попередньої. Крім того, додано додатковий критерій зупинення алгоритму – при досягненні точки, яка вже належить контуру. Переваги алгоритму - додатковий критерій зупинки, що дозволяє більш ефективно уникнути зациклювання алгоритму і правильно завершити його роботу, а також можливість роботи з фігурами зі складною контурною лінією, до недоліків - збільшення обчислювальної складності, а також не зовсім коректному завершенні алгоритму.

Інший алгоритм обходу контуру зображення – алгоритм „Theo Pavlidi’s Algorithm” [6]. Переваги цього алгоритму – висока якість роботи із зображеннями, що не мають складної контурної лінії, можливість використання будь-якої точки контуру як відправна точка та висока продуктивність, до недоліків – складність реалізації, проблеми обробки зображень з виступами на краях (товщиною в один піксель) та недосконалими критеріями завершення алгоритму.

Важливу роль ефективності та працездатності алгоритмів обходу контуру зображення грає критерій зупинення роботи алгоритму. Серед відомих критеріїв зупинки можна назвати такі [6]:

* зупинитись, коли алгоритм відвідає початкову точку *n* разів. Цей критерій ефективний, коли кількість повернень від алгоритму до початкової точки відомо заздалегідь.
* критерій зупинки по тому, як алгоритм двічі відвідав початкову точку (критерій зупинки Якоба). Цей критерій є окремим випадком попереднього. До переваг цього алгоритму можна віднести високу ефективність при роботі з простими контурами фігур та простоту реалізації, до недоліків можна віднести вибір початкової точки.
* критерій зупинки при наближенні до точки, вже визнаної точкою контуру. Перевага цього критерію – довільний вибір початкової точки, недоліки – необхідність відзначати пройдені точки контуру.

Мета і завдання дослідження. Метою роботи є розроблення алгоритмів сегментування зображень на площині і проектування програмної системи аналізу біомедичних зображень.

Об’єкт дослідження – процес сегментації зображень.

Предмет дослідження – алгоритми сегментації зображень.

Для досягнення поставленої мети необхідно розв’язати такі задачі:

* проаналізувати алгоритми сегментації зображень, показати їхні переваги та недоліки;
* проаналізувати відомі універсальні та спеціалізовані програмні засоби аналізу зображень;
* розробити алгоритми сегментації зображень на основі виділення контурів зображень;
* програмно реалізувати запропоновані алгоритми аналізу біомедичних зображень.

Методи дослідження базуються на теорії алгоритмів, аналітичній геометрії, теорії множин, теорії матриць.

Наукова новизна одержаних результатів. Розроблено і досліджено алгоритми сегментації зображень на прикладі біомедичних зображень, а також проведено їх моделювання.

Практичне значення одержаних результатів. На основі запропонованих алгоритмів сегментації зображень спроектовано комп’ютерну систему аналізу зображень.

Публікації результатів досліджень. За результатами кваліфікаційної роботи опубліковані у співавторстві двоє тез доповідей на V-ій науково-практичній конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі». Конференція відбулася на кафедрі комп’ютерної інженерії Західноукраїнського національного університету, 2 грудня 2021 р., м. Тернопіль [7, 8]:

Кваліфікаційна робота складається із трьох розділів, висновків, списку використаної літератури та додатків [9].

У першому розділі проаналізовано алгоритми центроїдного зв’язування, порогового обмеження за яскравістю, склеювання-розщеплювання, алгоритми зафарбовування зображень і алгоритми сегментації шляхом виділення границь і здійснено аналітичний огляд програмного забезпечення аналізу зображень.

У другому розділі розроблено алгоритми сегментації зображень на основі виділення границь зображень.

Третій розділ присвячений програмній реалізації алгоритмів аналізу зображень: структуру програмної системи, опис розроблених модулів і результати роботи і тестування програмної системи.

У додатках приведені лістинги розроблених програм, світлокопії публікацій та довідка про використання результатів кваліфікаційної роботи.

1 АНАЛІЗ АЛГОРИТМІВ СЕГМЕНТАЦІЇ ЗОБРАЖЕНЬ

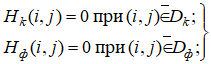
## 1.1 Сегментація зображень: основні поняття та принципи

Нехай задано растр *D*, який маєрозмір *п * *т,* де *п* і *т* — число рядків і стовпців – дискретне поле зору. Область *D* є растром, матриця *B* є функцією на цьому растрі: значення  дорівнює яскравості в точці , *i=* 1, …, *n*, *j* = 1, ..., *т* . Тоді функція  називається цифровим зображенням (або просто зображенням) на *D*-растрі*.*

Зображення  дійсного фрагменту є набір зображень окремих об'єктів і фону. Представляємо його у вигляді

** (1.1)

де *s* – кількість об'єктів у фрагменті; *Hk* (*i, j*) – зображення *k*-го об'єкта або його видимої частини (*k* = 1, ..., *s*); *H*ф (*i*, *j*) – фонове зображення. Або

 (1.2)

де *Dk  D* – область *k*–го об'єкту; *D*Ф *D –* область фону;

** (1.3)

Формування образів *H*1(*i*, *j*), ..., *H*s (*i*, *j*) і *Н*Ф (*i*, *j*) з образу , для яких виконується умова (1.2), є завданням пошуку областей об'єктів *D*1,…, *D*s та фонових областей *D*Ф. Перехід від зображення набору об'єктів і фону до зображень окремих об'єктів включає визначення кількості об'єктів *s*. Реалізація цих завдань є завершальною метою сегментації зображень [10-13]. Таким чином, сегментація поміщає два завдання: задачу пошуку фону та об'єктів та задачу ідентифікації об'єктів.

Розглянемо задачу виділення об'єктів з фону, яка є окремим випадком загальної задачі сегментації. Для представлення вхідного зображення у вигляді (1.1) побудуємо відображення

 (1.4)

що володіє наступними властивостями:

|  |  |  |
| --- | --- | --- |
|  |  | (1.5) |

Відображення (1.4) є правилом маркування точок растру *D*:кожна точка  отримує змістову мітку з номером . При цьому точки з однією міткою є областю окремого об'єкту або областю фону.

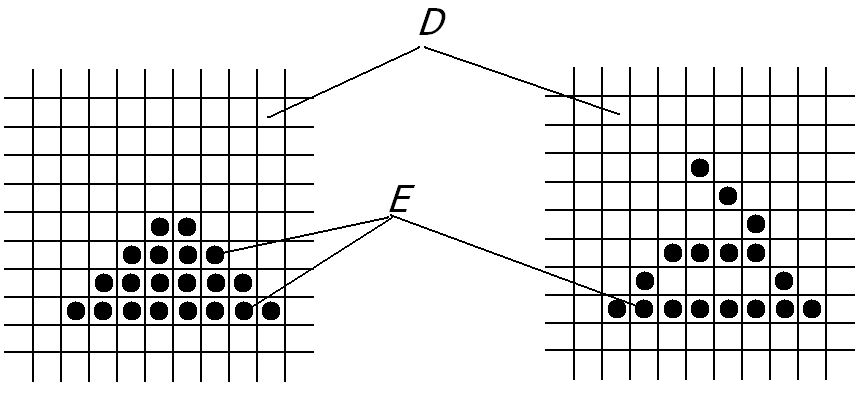
Побудову відображення можна провести тільки на основі евристичних міркувань. Поняття "об'єкт" і "фон" в термінах властивостей матриці *В* розміром *nт* формалізувати неможливо. Евристика зменшується в тому випадку, коли є апріорна інформація про розподіл яскравостей в області об'єкту і в області фону – в цьому випадку поняття "об'єкт" і "фон" піддаються частковій формалізації.

З апріорної інформації про об'єкти в переважній більшості випадків є інформація про їх зв'язність: об'єкт , як правило, не може складатися з декількох розрізнених частин. Поняття зв'язність піддається майже повній формалізації.

*Визначення*. Дві точки растру *D* з координатами (*i*, *j*) і (*p, s*)назвемо сусідніми в сенсі 4-ох зв’язності, якщо | *i* – *p* | + | *j* – *s*| = 1, і сусідніми в сенсі 8-ми зв’язності, якщо max  *=* 1.

*Визначення*. Множина точок *E * *D* називається 4-ох зв’язною (відповідно 8-ми зв'язною), якщо для будь–яких двох точок *А, В * *E* знайдеться множина точок ,  що володіє наступними властивостями: ; точки  і *Ai+*1 - сусідні в сенсі 4-ох зв’язності (відповідно 8-ми зв’язності).

Будь-яка 4-ох зв'язна множина є в той же час 8-ми зв'язною (рисунок 1.1). Зворотне твердження невірне.



а) б)

Рисунок 1.1 – Зв'язні множини точок растру:

*а* – 4-ох зв’язна; *б* – 8-ми зв'язна, що не є 4-ох зв’язною, (елементам растру *D* відповідають клітинки, множина *E * *D* виділено точками)

Часто можна заздалегідь припустити, що області об'єктів – 4-ох зв'язні множини. Іноді слід користуватися іншою інформацією – про 8-ми зв’язність.

Інформація про зв'язність об'єктів дозволяє у лінійці випадків розділти задачу сегментації на дві частини – груба сегментація і розфарбовування бінарного зображення.

Під грубою сегментацією розуміють побудову характеристичної функції об'єднання областей тільки об'єктів, тобто функції

.

Звідси можна зробити висновок, що побудова характеристичної є виділення фону.

Після грубої сегментації досить здійснити маркування для бінарного зображення , що складається з сукупності зображень декількох об'єктів.

Розбиття бінарного зображення на сукупність зображень-об’єктів називається розфарбовуванням*.* На відмінність від грубої сегментації розфарбовування здійснюється без частки евристики – це пов'язано з точною формалізацією поняття "об'єкт" для бінарного зображення: областю об'єкту є множина точок растру, що складається з точок (*i, j*)*,* для яких  = 1, воно зв'язне і не міститься усередині ніякої більшої зв'язної множини, що складається з точок (*i, j*), для яких *=* 1. Така формалізація має місце тільки за умови не перетину областей об'єктів.

Зупинимося ще на одному підході, що дозволяє отримати вираз (1.1). Він пов'язаний з іншою інтерпретацією об'єкту – за допомогою його границі. Нехай  - деяка область. Точку  назвемо граничною, якщо не всі її сусідні точки відносно 4-ох зв’язності лежать в області *E.* Множина граничних точок позначимо *дЕ* і назвемо границеюобласті *E.* Якщо область *E* є 4-ох зв'язною і відома її границя *дЕ,* то можна вважати визначеною і саму область, правда, тільки після застосування одного з алгоритмів відновлення множини по його границі. Тому для виділення і поділу об'єктів необхідно побудувати відображення 

 (1.6)

Задача побудови відображення  також може бути вирішена в два етапи. На першому етапі виділимо множину всіх границь об'єктів, тобто побудуємо характеристичну функцію типу  відносно *.*

Після цього необхідно ідентифікувати межу кожного об'єкту окремо.

Кінцевою метою сегментації зображень є поділ поля зору *D* на область фону *D*ф і області об'єктів *D*1, *..., Ds.* Сегментація шляхом побудови відображення з властивістю (1.5) називається методом маркування точок. Якщо ж метою є побудова відображення , тоговорять про сегментацію шляхом виділення границь.

Складність побудування відображень , обумовлена тим, що визначення "об'єкт", "фон", "межа об'єкту" умовні. Вони не точно формалізуються в термінах цифрового зображення *В*(*i, j*)*.* Інформуванням для такої побудови є наступні міркування, самі по собі що потребують формалізації:

* об'єкти "виділяються" з території фону;
* області об'єктів являються зв'язними областями;
* модуль градієнта яскравості в точках границі об'єктів істотно переважає його значення в інших точках зображення;
* за умови що області об'єктів не перетинаються, то область кожного об'єкту не міститься ні в жодній більшій зв'язній області, що складається з точок об'єктів.

Методи і алгоритми сегментації можна розглядати як формалізацію поняття можливості виділення об'єкту з фону або понять, пов'язаних з градієнтом яскравості. Надійність алгоритмів сегментації залежить від того, наскільки точно і повно при цьому врахована додаткова інформація, яка в основному складається з наступних відомостей:

* число об'єктів *s*;
* деякі характеристики розподілу яскравості в областях об'єктів або фону, наприклад екстремальні значення яскравості, кількість перепадів яскравості;
* оцінки перепаду яскравості при переході з області фону в область об'єктів;
* конфігурація об'єктів;
* інформація про те, яку частину поля зору займає об’єднання областей об'єктів.

Як і для будь-якої задачі оброблення зображень, для задачі сегментації слід шукати підхід за алгоритмами, який буде максимально наближатися до візуального способу рішення.

Введемо поняття часткової, грубої і багатозначної сегментації.

Під частковою сегментацією розуміється така, яка полягає в будові відображення



де  - деяка множина точок. При цьому побудова проводиться так, щоб були виконані наступні умови:

* якщо  є точками. області *Dk* одного об'єкту (*k*=1, ..., s), то ;
* якщо точки *,* але належать областям *Dp* і *Dk* різних об'єктів *,* то ;
* якщо   *-* точка з області фону *D*ф*,* то .

Очевидно, при  часткова сегментація є кінцевою. Перехід від часткової сегментації до остаточної тим простіший, чим менша кількість точок доповнення *.*

Сенс часткової сегментації полягає в тому, що багатьом (але не всім) точкам поля зору легко зіставити змістову мітку. Маркування точок, що залишилися, відбувається з урахуванням інформації про вже марковані точки, що істотно спрощує задачу.

Груба сегментація. При грубій сегментації здійснюється побудова відображення



що володіє двома властивостями:

* якщо ,то точки (*i*, *j*), (*т*, *п) D* не можуть міститися в області одного об'єкту або в області фону;
* якщо , то *.*

При грубій сегментації однаково марковані точки можуть відповідати областям різних об'єктів. Тому побудова відображення  значно простіша за правило маркування з властивостями (1.5). Зручність використання грубої сегментації як попередній етап повної (остаточною) сегментації пояснимо для випадку *t* = 1, коли груба сегментація рівносильна переходу до бінарного зображення: точкам фону відповідає 0, точкам об'єктів – 1. Тоді для остаточної сегментації досить провести розфарбовування цього бінарного зображення. Після розфарбовування, тобто побудови відображення  ( –об'єднання областей об'єктів), перехід до остаточного маркування з властивостями (1.5) у разі неперетину об'єктів здійснюється за правилом



Грубу сегментацію у вказаному випадку (*t* =1) можна назвати також відділенням фону або виділенням області об'єктів. Випадок *t =* 1 при грубій сегментації є найбільш поширеним.

Багатозначна сегментація. У багатьох інцидентах на попередніх стадіях сегментації частковим точкам поля зору неможливо зіставити однозначну змістову мітку. Проте апріорна інформація або інформація про вже марковані точки дозволяє заключити, що мітка даної точки растру може бути однією з деякого набору міток. Приклад багатозначного маркування представлено на рисунку 1.2. Нехай всі точки області *А* маютьмітку 0, а всі точки області *В* –мітку 1. Тоді область *А* належить фону, область *В* є частиною деякого об'єкту. Якщо відомо, що області об'єктів не перетинаються, то точка *С* може бути або точкою фону, або точкою об'єкту з міткою 1*.* Тому мітка точки *С* може приймати два значення: 0 або 1.

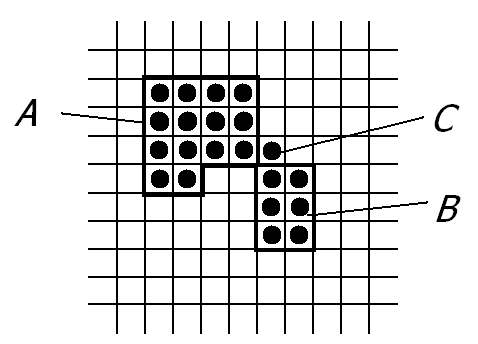


Рисунок 1.2 – Приклад багатозначного маркування: *А*, *В* – області; *C* точка

Багатозначним маркуванням називається багатозначне відображення , де *Ts* — множина підмножин множини . Тим самим, кожній точці  співставляється деяка множина міток , де , *i*=1, ..., *v*,а *v* залежить від точки (*i, j*).Число (*v*-1) називається показником неоднозначності маркування в точці (*i*, *j*) і позначатимемо його . Якщо =0, то мітка в точці (*i*, *j*) визначена однозначно. Багатозначне маркування співпадає з остаточною, якщо =0 для всіх *.*

Перехід від багатозначного маркування до остаточного (однозначного) здійснюється на основі аналізу множини  для тих точок , де 7 .

Проаналізуємо методи сегментації. Вони діляться на методи маркування точок і виділення границь. Метод виділення границь (контурів) має самостійне значення. Метод маркування точок ділиться, у свою чергу, на метод порогового обмеження по яскравості і метод нарощування областей.

## 1.2 Алгоритми сегментації зображень

### 1.2.1 Алгоритми порога яскравості

Цей вид алгоритмів є розповсюдженими методами сегментації, оскільки зображення об'єктів нерідко мають рівномірну яскравість і чітко виділяються на тлі [13-15]. Найпростіша гранична обробка виконується в тому випадку, коли заздалегідь відомо, що зображення складається з об'єкта (*s* = 1) і фону, і що яскравість точок об'єкта включена між [*Т*1, *Т*2], і яскравість фонових точок або менша, ніж *Т*1 або більша , ніж *Т*2. У цьому випадку кожна точка  відображається на мітку 1, якщо , і мітку 0 в іншому випадку. Виконана таким чином груба сегментація є вирішальною через умову *s* = 1. У випадку, коли *s*2 ця інформація також допускає грубу сегментацію, але перехід до остаточної сегментації також вимагає реалізації алгоритму розмальовки двійкових зображень. Таким чином, у цьому випадку для остаточної сегментації потрібна комбінація глобального методу (порогового відсікання) та локального методу (розмальовка за критерієм зв'язності). Іноді ми знаємо, що яскравість об'єктів різна і до того ж відомі пороги: яскравість *j*-гo об'єкту знаходиться всередині . І тут можна сформувати таке правило маркування точок:



Однак таке маркування буде точним тільки за виконанняумов:



у точці  відбувається умова 

По-іншому, інтервали яскравості об'єктів не зобов'язані перекриватись, яскравість фону повинна змінюватися за межами інтервалів яскравості об'єктів.

Однак у більшості випадків ці умови не виконуються і неможливо отримати остаточну сегментацію з використанням порогової інформації . Все-таки, можна добитися часткової або багатозначної сегментації. Припустимо, що поза інтервалами  відомий набір яскравості  у фонових точках. Якщо про фон апріорі нічого не відомо, то можна припустити, що він  складається з рівнів квантування світності. Так само можна вказати неприсутність інформації про яскравість *j*-го об'єкта , , де і  - нижній і верхній рівень квантування відповідно.

Набори  і  можуть бути такими, що на основі певних значень яскравості дозволено однозначно встановити приналежність точки, яка відповідає певній області. Число , яке розраховується як кількість множин з наборі , якому припадає значення яскравості *В*(*т, п)*, зв'яжемо з кожною точкою *.*  Число . є індикатором неоднорідності у точці (*т, п)*. Точки з показником ступеня, що дорівнює 0, позначаються однозначно. Справді, якщо , то ми повинні покласти , де *k* визначається тільки з умови .

У разі коли пороги яскравості об'єктів і фону є невідомі, алгоритм обмеження порогових значень повинен бути доповнений методом визначення порогових значень. Визначення порогових значень пов'язані з аналізом гістограм. Гістограмою є відображення набору  значень яскравості на набір натуральних чисел, кожне з яких  пов'язане з кількістю точок , для яких  (рисунок 1.3).

0 5 10 15 20 25 30 35 40 *b*

*N*

Рисунок 1.3 – Гістограма розподілу яскравості

Глобальний максимум гістограми відповідає  - найпоширенішому значенню яскравості. В більшості завдань фон є домінуючим, тому значення  відповідає фону . Треба знати, що ті, що близькі до  яскравості, також відповідатимуть фону. Для визначення порога, що відокремлює яскравість об'єктів від яскравості фону, достатньо мати додаткову інформацію. Нижче наведено найпоширеніші приклади такої інформації.

Припустимо, ми знаємо якесь співвідношення (наприклад, нерівність), яке пов'язує яскравість будь-якої точки на тлі та об'єктів, наприклад:

Припустимо, ми знаємо якесь співвідношення (наприклад, нерівність), яке пов'язує яскравість будь-якої точки на фону та об'єктів, наприклад:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | (1.7) | |
|  |  | |  | |
| для будь-яких точок | |  | |

Для цього випадку робимо висновок, що для будь-якої точки (*u, v*)будь-якого об'єкту виконується умова:

** (1.8)

Тепер знайдемо глобальний максимум частини гістограми – в області [, ]. Нехай це буде досягнуто у певний момент у точці . Слід очікувати, що   – це найбільш поширена яскравість у точках об'єктів. Зі співвідношення (1.7) можна зробити висновок, що для будь-якої точки (*т* , *п)* фонової області виконується умова

 (1.9)

Співвідношення (1.8) і (1.9) представляють пороги яскравості точок фону і об'єктів, тобто дозволяють побудувати часткове або багатозначне маркування.

### 1.2.2 Алгоритми центроїдного зв’язування (підключення до центру ваги)

Перераховані вище методи були глобальними. Вони використовували властивості гістограми. Розглянемо локальні методи сегментації, які називаються нарощуванням площі областей. Алгоритми даного типу використовують інформацію про зв'язність об'єктів і засновані на рекурсивному методі маркування точок. На кроці *k* позначаються лише точки, які мають сусіди серед тих, які позначені на попередньому кроці (*k* -1). Точки маркуються за певним критерієм однорідності. Конкретні алгоритми відрізняються вибором критерію однорідності,. способом візуалізації точок і вибором вихідних «початкових» точок, які позначаються з нульовим кроком.

Існує два основних підходи до стратегії вибору вихідних точок і порядку їх візуалізації: центроїдний зв’язок і склеювання поділу областей [15]. Вибір початкових точок та їх позначок в алгоритмах зв’язування центроїдів слід здійснювати так, щоб не було двох точок з різними суміжними позначками.

Крім того, якщо деяка інформація. про розміщення об’єктів в полі зору відома апріорі, слід докласти зусиль для виконання таких вимог:

* точки з різними мітками мають. відповідати площам різних об’єктів;
* точки з однаковою міткою мають. відповідати одному об’єкту.

Дві кінцеві вимоги будуть дотримані, якщо кожна вихідна точка має свою мітку, а самі точки вибрано на досить великій відстані одна від одної, що перевищує максимальний розмір об’єкта. Коли виділено кілька точок з однією позначкою, вони мають утворювати досить рівномірний набір за яскравістю; зазвичай вони зв'язані.

Отже, в алгоритмах центроїдного зв'язування апріорна інформація про об'єкти враховується переважно на етапі вибору початкових точок.

Вибір початкових точок – це нульовий крок для алгоритмів центроїдного зв’язування. Нехай  - множина точок, які отримали мітки після виконання кроку за номером *k.* Перехід від кроку з номером *k.* до кроку з номером (*k*+1) здійснюється наступним чином. На наступному (*k* + 1) – му кроці маркування мають бути суміжні точки до , тобто такі, що не належать , але мають хоча б одного сусіда з *Sk*. Припустимо  є однією з цих точок. Мітками  позначимо теги, які використовуються для позначення точок *Sk*. Розглянемо два можливі випадки.

Усі сусіди, позначені літерою *A*, мають однакову мітку . В цьому випадку точка *А* відзначена знаком  або знаком , тобто ще не використаним. Для першого знаку виконується за дотримання певної умови однорідності. Зазвичай вона має вигляд , де *В*(*A*) – яскравість у точці *A*,.  – середня яскравість точок з позначкою , *T* – обраний поріг.

Точки з *Sk* , що примикають до точки *A*, мають. різні мітки  (з 4-ох зв'язністю ). У цьому випадку умова перевіряється

 (1.10)

для кожного *i* = 1, ..., *l*. У випадку якщо (1.10) виконується для лише одного , то точка *А* отримує мітку .

Якщо умова (1.10) не виконується для всіх *і* = 1, ..., *l*, то точці *A* приписується нова мітка . У найскладнішому разі умови (1.10) виконуються для кількох значень, наприклад для *i* = 1 і *i* = 2. У цьому разі зливаються дві області: області точок з мітками  і , відповідно. Склеювання відноситься до переходу від міток  і  до одної мітки min( і ).

Коли області поєднуються, часткове маркування, отримане на етапі k, змінюється. Тому аналіз для більшої кількості точок зображення необхідно повторити. Звісно, ​​необхідність багаторазового проходження полем зображення значно збільшує час реалізації алгоритму.

Одночасне здійснення умов (1.10) за i = 1,2 є дуже грубим критерієм склеювання відповідних областей. Більш тонкий критерій склеювання регіонів заснований на усуненні про «слабких кордонів» [13].

### 1.2.3 Алгоритми склеювання-розщеплювання

Алгоритми склеювання-розщеплювання (Fusion-split) відрізняються від алгоритмів зв'язування центроїдів насамперед порядком, за яким переглядаються точки. Вони прагнуть виділити максимально можливі однорідні зони у зору *D*. Однорідні задані області інтерпретуються як елемент зображення [15].

Ми припускаємо, що поле зору *D* є растр розміром *т т ,* причому *т* =  (рисунок 1.4, *k* = 4). Окремі точки растру (пікселі) називатимуться областями нульового рівня,. позначимо їх . Означимо зараз області рівня першого об'єднанням чотирьох суміжних областей нульового рівня, а саме:

,

де .

За довільного означимо області *р*–го рівня як об'єднання областей (*p –* l) –го рівня:

,

.Останній рівень має *k*-ий номер*.* Для рівня *k* є лише одна область, яка збігається з растром *D*. Звернемо увагу на основні властивості зазначеного розбиття:

* кількість зон рівня *p*, де , що дорівнює ;
* об'єднання областей усякого рівня – растр *D*;
* усяка область рівня *p* містить 22*р* точок растру;
* усяка область рівня (p + 1) являє собою об'єднання чотирьох сусідніх областей рівня *p*.

Перший етап алгоритму полягає в аналізі всіх областей 1-го рівня виділення у тому числі однорідних зон, розкид світності точок яких перевищує фіксований поріг. Однорідні області при наступному аналізі наступних рівнів розуміються як елемент зображення зі яскравістю, що дорівнює середній яскравості точок відповідної області. Отже, коли виконується критерій однорідності для області , області  не розглядаються окремо – вони зливаються.

|  |  |
| --- | --- |
| 10-fig3 | |
| а) | б) |
| 10-fig3-2 | |
| в) | г) |

Рисунок 1.4 – Розбиття растру розміром 1616 на області різних рівнів:

*а* – 0–го, *б* – 1–го, *в* – 2–го; *г* – 3–го

На наступному етапі розглядаються зони другого рівня, але не всі, а ті зони , для яких виконується умова: для усякої із зон  виконується критерій однорідності. (перевірка пройшла на попередньому етапі), тобто вони вже є елементом зображення. Тепер на критерій однорідності перевіряється набір , тобто обчислюється. розкид чотирьох значень яскравості, що відповідають цим областям (тепер елементи зображення). За виконання умови однорідності зазначені області поєднуються в один елемент зображення середньої яскравості. Ця процедура виконується послідовно на 1-му, 2-му, ..., *l*-му рівнях. Вона закінчується на певному рівні *l*, коли виконується така умова: для будь-якої зони , або жодна із зон  не є єдиним елементом зображення (інформація про це після аналізу (*l* - l) -го рівня), або ті з цих зон, які є єдиними елементами зображення, для яких виконуються умови однорідності, тобто де вони можуть зливатися.

Результатом є поділ растру *D* на ряд однорідних областей, які припадають різним рівням, і точок, що складаються з різної кількості. Так, ділянки поля зору, у яких яскравість коливається у незначних кордонах, відповідають великим ділянкам поділу растру, неоднорідні за яскравістю ділянки розбиваються на дрібні частини.

### 1.2.4 Алгоритми розфарбовування зображень

Методи відсікання порога та нарощування областей в деяких випадках дозволяють лише виділити область об'єктів *D*об та фону. Тобто дозволяють отримати двійкове зображення (яскравість 1 –точки області об'єктів, яскравість 0 – у точках фону). Щоб отримати остаточну сегментацію, залишається розфарбувати двійкове зображення, що вийшло, або, що те ж саме, розфарбувати множину *D*об. І тут поняття «об'єкт» формалізується у термінах двійкового зображення, отже, для розмальовки існують алгоритми, які не містять евристичних елементів.

Оскільки зв'язність областей окремого об'єкта грає вирішальну роль у розфарбовуванні бінарного зображення, всі алгоритми розфарбовування приписуються, критично говорячи, до методу нарощування областей. Точного фарбування можна досягти як методом склеювання-розщеплення, і методом центроїдного зв’язування за центром тяжіння.

Алгоритми центроїдного. зв’язування вирізняються насамперед критерієм однорідності зони, а також правилом відображення точок. Для двійкових зображень критерій однорідності є очевидним: однорідна область складається лише з точок яскравості 1 або точок яскравості 0. Отже, порівняно з двійковими зображеннями, порядок відображення точок буде вирішальним для алгоритмів центроїдного зв’язування. У зв'язку з цим широкого поширення набули алгоритми, засновані на природному порядку відображення точок растру – рядково. Найпростіший спосіб полягає є наступний: точка (*т, п)* пов'язана з міткою (кольором) з номером, який є мінімумом із числа зазначених міток сусідніх точок. Якщо таких сусідів не існує, то точці (*т , п)* надається нова, ще не використана позначка (тег).

Такий алгоритм є простий. Він має обмежену сферу застосування – тільки для простих зображень, наприклад, краї яких паралельні лініям растру або зображень трохи складніших (рисунок 1.5 *а*)*.* Для ряду простих зображень (рисунок 1.5*б*) цей спосіб не підходить: на місце одного кольору ми отримуємо 5 кольорів (колір з цифрою 1 з'являється при розфарбовуванні верхнього рядка, колір з цифрою 2 – при розфарбовуванні наступного після верхнього рядка і т. д. При розфарбовуванні наступного рядка з'являється інший колір). Розподіл кольорів, що відповідає двійковому зображенню рисунку 1.5 *б* із зазначеним «прямим» розфарбуванням, показано на рисунку 1.5 *в*.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| а) | б) | в) | г) | д) | е) |

Рисунок 1.5 – Приклад прямого розфарбовування:

*а*, *б* - вхідні зображення; *в* -результат обробки зображення *б*; *г, д, e* - результат повторних проходів (відповідно 2, 3, 4)

Результати застосування процедур послідовного розфарбовування показано на рисунку 1.5 *г, д, e.*

Введемо кілька додаткових позначень та визначень. Нехай обраний порядок точок растру буде наступним: .  позначає підмножину множини , що складається з точок, суміжних з *Аk*  (не більше чотирьох елементів, а для порядкового сканування – не більше двох).

Алгоритм розфарбовки наступний:

1. Якщо *В*(*A*1)= 0, то , то інакше . Ми визначаємо відображення  як те саме.

2. Нехай за значенням  точок  знайдено та визначено відображення . Якщо, , то .

Якщо , але  (або ), то



Якщо , то

.

Якщо при цьому серед точок *U* (*Ak*)знайдуться дві такі , , що , , то відображення (тобто ) зміниться, а саме:

* при , то ;
* при , то



3. Після побудови відображення  всіх точок  будується відображення  відповідно до правила.



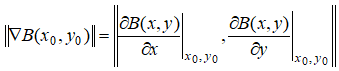
Свідчення того, що відображення  дає точне забарвлення будь-якого двійкового зображення для будь-якого порядку візуалізації точок (це явище не є очевидне), міститься в [14].

### 1.2.5 Алгоритми граничної сегментації

Розглянемо безкомпромісно інший метод сегментації (проти методу точкового маркування) – сегментація з виділення границь об'єктів. За допомогою цього методу сегментації об'єкти представлені своїми границями. Вибір меж об'єкта можна подати як самостійне практичне завдання, безпосередньо не пов'язане з сегментацією. Границі є основою для формування різних функцій та граматик у розпізнаванні зображень. Обмежує критерій – кількість обчислень.

Поняття границі об'єкта, як і поняття самого об'єкта, не може бути формалізовано в термінах цифрового зображення . З евристичних причин граничні шукаються як точки сильного падіння функції яскравості.

Опрацюємо безперервну модель зображення. В цьому випадку замінимо растр *D* прямокутною областю у площині (*x, у*)та визначимо зображення відповідно до функції *В*(*x, у*). Для цього (загального) зображення ми припускатимемо, що *В*(*x, у*) є диференціальною функцією своїх аргументів. Різниця яскравості у точці (*х*0, *у*0)визначається у разі як норма градієнта функції *В*(*x, у*) у точці (*х*0, *у*0), за формулою:



Норма вектора означується різними способами [16]:

 (1.11)

 (1.12)

 (1.13)

Однак через обмеження поля зору *D* прямий розрахунок неможливий. Щоб використовувати диференціацію для визначення різниці в яскравості в точці, необхідно вибрати один із наступних двох способів:

* шляхом інтерполяції перейти від функції  до функції *В*(*x, у*), заданої у прямокутній зоні площини (*x*, *у*), що містить усі точки растру *D*;
* використовувати методи чисельного диференціювання, тобто диференціювати функцію набору таблиць.

Перший метод називається методом функціональної апроксимації. Він з високими обчислювальними витратами і широко використовується у завданнях робототехніки. Найбільш поширений другий метод, градієнтний метод – метод контрасту або метод просторового диференціювання. Наявний ще інший метод виділення меж, не пов'язаний з диференціюванням – метод високочастотної фільтрації [15].

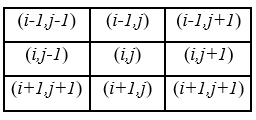
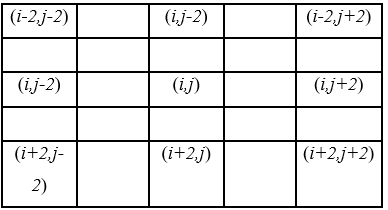
Алгоритми просторового диференціювання засновані на обчисленні градієнта функції яскравості  з використанням однієї з багатьох формул. Головна відмінність одного алгоритму від іншого – вибір формули дискретного диференціювання.

Після обчислення норми градієнта у кожній точці , ми переходимо від вихідного зображення  до зображення градієнта. Зображення  відрізняється від  підкресленими змінами яскравості. Перехід  називається контрастуванням зображення.

Контрастність – це лише частина проблеми вибору границь. Для вибору границь потрібно сформувати критерій належності точки  границі зони об'єкта. Для формування такого критерію вибирається певний поріг *T* і точка  розглядається як гранична, якщо . За відсутності апріорної інформації для вибору порога *T* ми знову знаходимо його, аналізуючи гістограму зображення  більш-менш так само, як поріг, що позділяє яскравість об'єктів та фону. Формули чисельного диференціювання. переходу до зображення  можуть бути різними. Ось їх основні характеристики:

* + кількість точок в околі точки , значення яскравості, на які впливає значення ;
  + той чи інший тип норми;
  + тип диференціювання (одномірний або двовимірний).

Звернемо увагу на ці властивості. Означимо «вікно» якого центр у точці як набір точок , де , *q* – параметр вікна, який впливаї на його розмір (вікно матиме (2*q* + 1)2 точок). На рисунку 1.6 *а* показано вікно з параметром *q* = 1, з параметром *q* = 2 – на рисунку 1.6 *б.* Вікно з центром у точці  і параметр *q* буде позначено через .

а) б)

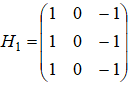
Рисунок 1.6 – Вікна розмірами 33 (*а*) і 55 (*б*)

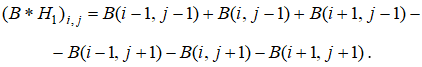
Припустимо, що інформація про яскравість у певних точках вікна  використовується для обчислення . Нехай  - набір цих точок. Якщо всі точки *R* належать одній і тій же горизонталі, вертикалі або діагоналі растру, то контраст називається одномірним, інакше – двомірним. Наприклад, одновимірне диференціювання вздовж лінії з орієнтацією 45° виходить вздовж . Диференціація по лініях з орієнтацією, що відрізняються від 0°, 45°, 90°, 135°, утруднена, оскільки ці лінії не є природними для растру [15].

При виборі норми (1.13) відбувається лінійний контраст, за норми (1.11) або (1.12) – нелінійний.

Значення  може відрізнятись від приблизного значення  при постійному множнику (незалежному від точки ). Для обчислення  рекомендується ввести маски, усяка з яких пов'язана з вікном певного розміру. Маска *Hq*, що відповідає вікну параметру *q*, має вигляд матриці розміру (*2q +* 1)  (*2q +* 1). Для усякого зображення  та маски *Hq* означимо згортку



Згортка при вікні 33 і масці 

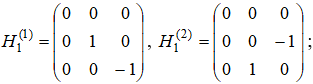


Основна кількість використовуваних обчислювальних формул грунтуються на застосуванні масок розміром 33. Головними є оператори Робертса, Превітта, Собела, Кірша, різницевий оператор, які мають загальний вид для операторів:

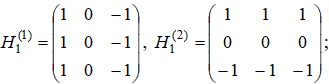




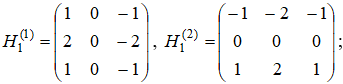
Робертса:



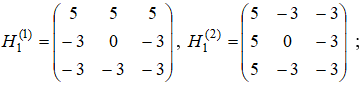
Превітта:



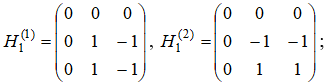
Собела:



Кірша:



різницевого:



Всі ці маски відповідають двовимірній диференціації. Для обчислення норми  можна застосувати одну із формул (1.11) - (1.13). Звернемо увагу, що маска, що використовується для диференціювання, повинна мати ту властивість, що сума всіх елементів відповідної матриці дорівнює 0.

Аналізовані алгоритми сегментації зображень показали неможливість точного та однозначного визначення «фону» та «об'єкта», а також відсутність універсального алгоритму сегментації, що веде до пошуку алгоритмів за класом зображень.

## 1.3 Аналітична презентація програмного забезпечення для аналізу зображень

Сегментація зображень – одна з найважливіших функцій будь-якого програмного забезпечення обробки зображень. Щоб розглядати сучасне програмне забезпечення, ми розділимо програмні пакети на два типи: універсальні програми та спеціальні програми. Пакети інженерних комп'ютерних програм Matlab, Scilab та Matchcad можна назвати універсальними системами обробки зображень. Ці системи містять інструменти поліпшення зображень, розпізнавання сегментації тощо. Складні аналізатори зображень відносяться до спеціальних систем. Аналізатор зображень – це автоматизоване комп'ютерне робоче місце, яке дозволяє спеціалісту у певній галузі виконувати поставлені завдання за допомогою спеціального обладнання та програмного забезпечення (електронні мікроскопи, телесистеми).

Розглянемо докладніше аналізатори зображень. Більшість девелоперських компаній, що існують сьогодні на українському, російському та зарубіжних ринках виробників систем аналізу зображень, були сформовані у 80-90-ті роки 20 століття [17]. Компанії-виробники аналізаторів умовно можна поділити на дві категорії:

1. Компанії, що пропонують укомплектовані робочі місця, включаючи мікроскоп (для роботи з мікрооб'єктами), систему введення даних, комп'ютерне обладнання та програмне забезпечення. Це наступні компанії: Leica Microsystems (універсальні та спеціалізовані системи для різних програм), Leco (спеціалізується на створенні робочих станцій для матеріалознавства та металографії), Applied Imaging (системи для цитогенетики: автоматичне каріотипування, методи люмінесцентного аналізу).
2. Компанії, що продають «легкі» комплекти (системи введення та програмне забезпечення) або варіант коробкового програмного забезпечення. Цих компаній безліч: Clemex (Канада), Soft Imaging System (Німеччина), Noesis (Франція), Universal Imaging Corporation (США), SIAMS (Росія).

Система аналізу зображень (аналізатор зображень) складається із системи введення зображення з об'єктивом для макрозйомки або встановленою на мікроскопі для фотографування мікроскопічних об'єктів, комп'ютера та програмного забезпечення. Незалежно від можливостей програмного забезпечення для аналізу зображень, наступні етапи обробки завжди будуть однаковими:

* отримання (ввід) зображення;
* його перетворення і редагування;
* виділення об'єктів чи фази на зображенні;
* проведення вимірів;
* збереження та друк зображень і результатів аналізу.

Об’єктами дослідження у таких системах є статичні зображення мікроскопічних об'єктів: мінеральних зерен, пор, мікродефектів, включень, клітинних структур, мікроорганізмів.

Як приклад сучасної промислової системи аналізу зображень можна привести систему SIAMS 700™ [18] (компанія SIAMS, Росія). Вона призначена для аналізу макро- і мікроструктури матеріалів: металів і сплавів, скла, кераміки і ін. Таблична технологія SIAMS Photolab забезпечує визначення кількісних характеристик макро- і мікроструктур матеріалів; статистичний аналіз і автоматичне створення звітів за наслідками дослідження; складання атласів цифрових еталонних зображень. Система містить готові модулі для автоматизованого вирішення типових завдань матеріалознавства. Вона дозволяє проводити наступні види аналізу: аналіз зерна в сталях і сплавах, виявлення структурних і фазових складових в сталі, дослідження фазового складу глинозему.

Аналізатор зображень NEXSYS ImageExpert™ Pro 3 – універсальний інструмент для чисельного аналізу зображень в науці і на виробництві [19-20]. Ряд завдань в області медицини і біології, що вирішуються сьогодні користувачами: підрахунок колоній бактерій, оцінка ядерно-клітинного (цитоплазматичного) відношення при аналізі патології кліток крові, аналіз морфології гістологічних зрізів та інші (рисунок 1.7). Серед інших у пакеті доступні наступні операції: морфологічна ерозія, дилатація; розкриття, замикання; розділення, з’єднання; скелетизація, відсікання; видалення граничних об'єктів, видалення пор, сегментація за кольором. За допомогою сегментації зображення бінаризується, тобто розділяється на декілька нових. Таким чином, наприклад, виділяються ядра клітин у зображенні тканини.

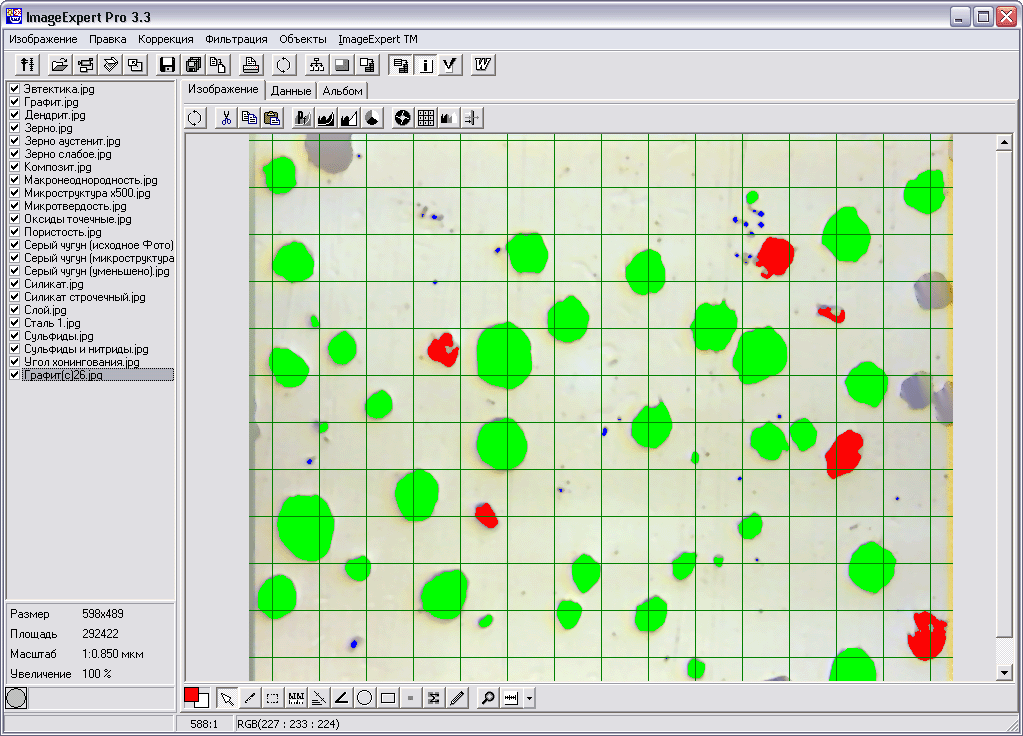


Рисунок 1.7 – Парадне вікно програми ImageExpert™ Pro 3

Окрім бінаризації введена сегментація палітри за кольором. Це дієвіший і наочніший інструмент для виділення структурних складових, оскільки замість одного порогу в бінаризації захоплюється діапазон кольорів по кожній колірній компоненті. Таких діапазонів сегментації може бути до 16 за один прохід, що задовольнить потреби будь-яких користувачів і стандартів. Наочність досягається за рахунок того, що захоплення і зафарбовування потрібних об'єктів відбувається на тлі оригінального зображення, дозволяючи значно точніше передати контури об'єктів.

Програмний пакет imageWarp містить в собі вбудовану автоматичну мову сценаріїв [21]. За допомогою написаних користувачем сценаріїв вона дозволяє записувати і автоматизувати обробку зображень, дії редагування і аналізу, створення інтерфейсу користувача. Багатопотоковий механізм виконання забезпечує паралельне виконання множини функцій в той же час і, також, автоматичне розпаралелювання для систем з багатьма центральними процесорами.

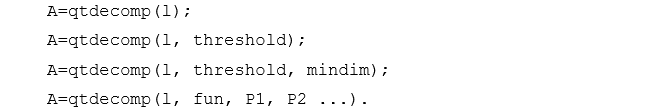
Розглянемо тепер універсальну систему інженерних обчислень Matlab [22]. Для сегментації зображень у бібліотеці Image Processing Toolbox містяться такі функції:

* poly2mask – трансформація декотрої області в маску;
* qtdecomp – сегментація методом розділу;
* edge – виділення границь;
* roipoly – задання області інтересу з допомогою полігону;
* roicolor – бінаризація за заданими кольорами;
* watershed – алгоритм маркерного водо розподілу.

1) Функція POLY2MASK служить для перетворення котроїсь області в маску. Синтаксис наступний: BW=poly2mask(x, у, m, n).

Функція BW=poly2mask(x, у, m, n) проводить обчислення бінарної маски області інтересу BW на основі аналізу цієї ж локальної області, представляючи її у вигляді векторів *x* і *у*. Розмірність результату обробки BW рівна *m**n*. Піксели в BW, які знаходяться усередині багатокутника (*x*, *у*), рівні 1; піксели поза цією областю рівні 0. Дані BW представляються в логічному форматі. Функція poly2mask проводить автоматичне закриття області багатокутника.

2) Функція QTDECOMP застосовується для сегментації методом розділення. Синтаксис наступний:



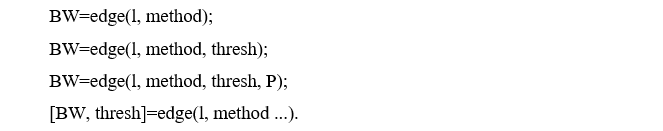
Функція qtdecomp здійснює сегментацію півтонових зображень методом розділення. Суть методу полягає в наступному [23]. Зображення розділюється на області, що не перекриваються. Кожна область за допомогою деякого критерію перевіряється на однорідність. Якщо область неоднорідна, то вона розбивається на області меншого розміру, кожна з яких, у свою чергу, перевіряється на однорідність. Процес завершується тоді, коли жодна з областей не може бути розділена, тобто або області однорідні, або їх розміри досягли гранично малих величин. В результаті роботи алгоритму виходить набір однорідних областей різного розміру.

У функції qtdecomp кожен блок розбивається на 4 не перекритих блоки однакового розміру. На першому кроці алгоритму блоком вважається все зображення. Найдрібнішим по розмірах є блок, який не можна розділити на 4 блоки однакового розміру, тобто такий блок, у якого число рядків або число стовпців непарне. Таким чином, у функції qtdecomp рекомендується використовувати зображення з розмірами, рівними степеню двійки. В цьому випадку найдрібніший блок складатиметься з одного піксела. При реалізації алгоритму розділення використовуються структури даних, базовані на квадро-деревах. З цієї причини даний алгоритм часто називають декомпозицією або розділенням за допомогою квадродерев.

Функція А=qtdecomp(I) здійснює сегментацію півтонового зображення методом розділення і поміщає результат в розріджений масив А (тип даних sparse MATLAB). Розріджений масив А конструюється таким чином. Елементам матриці А, відповідним координатам лівих верхніх кутів блоків на початковому зображенні I, присвоюються значення, що визначають розміри кожного блоку. Таким чином, більшість елементів матриці рівна нулю і тому для зберігання квадро-дерева застосовується розріджений масив, який ефективно використовує пам'ять, коли більшість елементів масиву рівна нулю. Для даної функції критерієм однорідності блоку є рівність всіх пікселів блоку один одному.

Функція А=qtdecomp(I, threshold) працює аналогічно описаній вище, але в ній блок вважається однорідним, якщо різниця між максимальним і мінімальним значенням пікселів блоку менше параметра threshold.

3) Функція EDGE служить для виділення границь. Її Синтаксис наступний:



Для сегментації на зображеннях часто необхідно виявити границі об'єктів - ділянки зображення, в яких є перепад яскравості. Функція BW=edge(I, method) призначена для виділення границь на початковому півтоновому зображенні I. Дана функція повертає двійкове зображення BW такого ж розміру, як початкове I. Піксел BW(r, з) рівний 1, якщо піксел I(r, з) належить границі. Для виявлення границь може використовуватися декілька методів. Потрібний метод задається в параметрі method у вигляді одного з наступних рядків: 'sobel', 'prewitt', 'roberts', 'log', 'zerocross', 'canny'. Якщо параметр method при виклику функції опущений, то за замовчанням він вважається рівним 'sobel'. Для кожного з методів визначення меж можна задати додаткові параметри. Для цього використовується одна з функцій BW=edge(I, method, thresh), BW=edge(I, method, thresh, P), де параметр thresh задає поріг для визначення того, чи належить піксел до межі, а в параметрі Р передаються налаштування, специфічні для кожного з методів. Якщо при виклику функції параметр thresh опущений, то значення порогу вибирається автоматично. Набути значення порогу можна, додатково визначивши вихідний параметр thresh: [BW, thresh]=edge(I, method...).

4) Функція ROIPOLY служить для визначення області інтересу за допомогою полігону. Синтаксис наступний: BW=roipoly(S).

Функція BW=roipoly(S) дозволяє інтерактивно задати область інтересу на початковому зображенні S будь-якого типу і помістити результат в двійкове зображення BW. Отримане двійкове зображення BW можна використовувати як область інтересу для функції roifilt2 або для завдання положення об'єктів і їх подальшого пошуку за допомогою функції bwlabel.

Дана функція окреслює зображення S у вікно і очікує від користувача, щоб він визначив область інтересу. Якщо за умови виклику функції параметр S пропускається, то зображення вибирається з поточного вікна. Область зображення, що цікавить, має бути обведена полігоном, вершини. якого задаються одноразовим натисненням лівої клавіші миші. Задану вершину попередньо можна видалити, при натисненні клавіш Backspace або Delete. Натиснення правої клавіші миші чи подвійне натиснення лівої клавіші вказує на кінцеву вершину полігону. Також завершити процес завдання вершин без вказівки останньою можна натисненням на клавішу Enter. Зображення BW і S мають однаковий розмір. Пікселу бінарного зображення BW(r, з) привласнюється значення 1, якщо піксел S(r, з) знаходиться усередині полігону, інакше BW(r, з) рівне 0. Для знаходження точок, що лежать всередині полігону, треба використати правило Non-Zero Winding.

5) Функція WATERSHED призначена для реалізації алгоритму маркерного водорозподілу. Синтаксис наступний:



Функція L= - Отримати (захоплення) зображення;

- його перетворення та редакція;

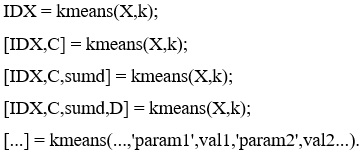
- виділяти об'єкти чи фази на зображенні;

- проводити виміри;

- збереження та друк зображень та результатів аналізу.watershed(A) обчислює матрицю маркерів, яка ідентифікує області водорозподілу початкової матриці A. Матриця А може бути будь-якій розмірності. Елементи матриці L – це цілі числа, які більше або рівні 0. Елементи, помічені нулями не належать ніякій області водорозподілу. Вони називаються пікселами водорозподілу. Елементи помічені одиницями належать області водорозподілу, елементи помічені двійкою належать іншій області водорозподілу і так далі. За замовчуванням, реалізований алгоритм маркерного водорозподілу використовує 8-зв'язні околиці для двовимірних і 26-зв'язні околиці для тривимірних зображень. Для –мірних зображень, алгоритм маркерного водорозподілу використовує зв'язність, яка визначається як conndef(ndims'(A),'maximal).

Функція L=watershed(A,CONN) визначає зв'язність для використання в алгоритмі маркерного водорозподілу. Зв'язність може бути визначена, в більшості випадків, для будь-якої розмірності при використанні для параметра CONN матриці нулів і одиниць розмірності 33...3. Значення 1 визначає місце розташування околиці щодо центрального елементу CONN. Запис значень параметра CONN повинен бути симетрична щодо центрального елементу. Масив A може бути числовим або логічним масивом будь-якої розмірності. Також він повинен бути не розрідженим. Вихідний масив L належить класу double.

6) Для кластеризації в колірному просторі використовується функція з Statistics Toolbox kmeans. Вона виконує кластеризацію на основі внутрішньогрупових середніх. Синтаксис наступний:



IDX = kmeans(X, до) функція дозволяє розділити множину початкових об'єктів, заданих матрицею Х, на *k* кластерів. Розмірність Х рівна , де *п* - кількість спостережень *p*-мірної випадкової величини. Рядки Х відповідають спостереженням, стовпці Х - ознакам багатовимірної випадкової величини. Процедура кластеризації є ітераційною. Критерієм кластеризації є мінімум внутрішньокластерної суми відстаней точок кластера до його центроїда. Сума відстаней знаходиться по всіх кластерах. В якості відстані точок кластера до його центроїда використовується квадрат евклідової відстані. Функція повертає вектор IDX індексів кластерів для кожного спостереження багатовимірної випадкової величини. Число елементів IDX рівне *n*.

[IDX,C] = kmeans(X,k) функція повертає матрицю координат центроїдів кластерів C. Розмірність матриці C рівна .

[IDX,C,sumd] = kmeans(X,k) функція повертає вектор сум відстаней об'єктів кластера до його центроїда sumd. Число елементів sumd рівне *k*.

[...] = kmeans(...,'param1',val1,'param2',val2...) додаткові вхідні параметри 'param1', 'param2' :, призначені для управління роботою алгоритму кластеризації і задаються у вигляді пари <назва параметра, значення> детально описані в в роботі [24].

Функція kmeans використовує ітераційний алгоритм мінімізації внутрішньокластерної суми відстаней об'єктів кластера до його центроїда по всіх до кластерів, що складається з двох етапів:

* Перша фаза призначена для пошуку наближеного значення центроїдів кластерів і попереднього угрупування об'єктів в кластери. Всі об'єкти групуються в найближчі до них кластери з подальшим розрахунком відстаней між об'єктами і центроїдами. Одноразове наближення при групуванні всіх об'єктів виконується за одну ітерацію.
* Друга фаза призначена для пошуку точного і остаточного рішення. Групування кожного з об'єктів по кластерах, з подальшим розрахунком його відстані до центроїда, на цій фазі виконується по критерію мінімізації внутрішньокластерної суми відстаней об'єктів до центроїдів кластерів. На кожній ітерації розрахунок виконується по всіх об'єктах початкової множини.

Недоліком ітераційного алгоритму функції kmeans є можливість попадання вирішення в локальний мінімум внутрішньокластерної суми відстаней точок кластера до його центроїда по *k* кластерів. Ця проблема може бути розв’язана відповідним вибором початкової точки на початку процедури кластеризації об'єктів.

## 1.4 Висновки до розділу 1

Отже, в даному розділі зроблено аналіз алгоритмів сегментації зображень, а також проведено аналіз спеціалізованих та універсальних програмних засобів аналізу зображень, що дало можливість розробити та застосувати алгоритми сегментування зображень біомедичної природи.

2 АЛГОРИТМИ СЕГМЕНТАЦІЇ ЗОБРАЖЕНЬ БІОМЕДИЧНОЇ ПРИРОДИ

## 2.1 Сегментація зображень шляхом кластеризації колірного простору

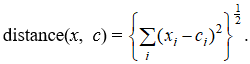
Сегментація і кластеризація співвідносні поняття в залежності від контексту їх використання для конкретних задач аналізу зображень. У загальному випадку кластеризація – це процес поділу об'єктів на групи (кластери) на основі властивостей, що описують суть об'єктів [25]. Стосовно квантування зображення це процес поділу значень атрибутів на групи (кластери), щоб у кожній групі знаходилися тільки тісно пов'язані значення. Якщо ми візьмемо два кластери (фон і об'єкти), то за допомогою кластеризації можна зробити деяку грубу сегментацію. При вказанні кількості кластерів можна отримати розфарбовування зображень.

Дослідимо алгоритм *k*-середніх для сегментації зображень. Алгоритм *k*-середніх – це метод кластеризації, що використовується для поділу набору об'єктів на *k* груп відповідно для заданої міри близькості. Узагальнений алгоритм складається із двох кроків.

1. Розрахунок центроїдів кластерів. Початкові центроїди визначаються відповідно до методу, вказаного користувачем. Існує три методи вибору початкових центрів: вибір *N* спостережень для максимізації початкової відстані, випадковий вибір *N* спостережень та вибір перших *N* спостережень. І тут *N* означає *k*. Після присвоєння всіх об'єктів найближчому центроїду нові центроїди обчислюються з використанням усіх призначених членів. Для неперервних змінних значення центроїду – це середнє всіх членів, привласнених цьому кластеру. Для категоріальних змінних значення центроїду є першим шаблоном із усіх присвоєних йому термінів.
2. Віднесення кожного об'єкта до найближчого центроїду. Найближчий центроїд використовує метод точно заданого ступеня близькості. Перед розрахунком значення всіх неперервних змінних нормалізуються. Звернемо увагу, що *c* є центром тяжкості (центроїд), а *x* являє собою спостереження. Якщо *i*-а змінна категоріальна, (*xi* - *ci*) дорівнює 0, якщо їх значення збігаються, інакше – 1. Якщо *i*-а змінна є неперервною, *xi* і *ci* спочатку нормалізуються з використанням мінімальних та максимальних значень цих змінних. Для визначення міри близькості використовуються евклідової відстані, квадрат евклідова відстані, манхеттенська відстань та відстань Чебишева.
3. Якщо всі спостереження належать кластеру, який належав до поточної ітерації, тоді перервіть ітерацію. Також, якщо кількість ітерацій дорівнює максимальній кількості ітерацій, відмовтеся від ітерації. Оновіть центроїди та отримайте остаточне угруповання.

Вимірювання відстані можуть бути визначені в одновимірному або багатовимірному просторі. Найпростіший спосіб розрахувати відстані між об'єктами у багатовимірному просторі – це обчислити евклідові відстані. Якщо у нас є двомірний або тривимірний простір, то цей вимір є фактичною геометричною відстанню між об'єктами в просторі.

Евклідова відстань. Це найпоширеніший тип дистанції. Це геометрична відстань у багатовимірному просторі, яка розраховується так:



Евклідова відстань (і її квадрат) розраховується з урахуванням необроблених даних, а не стандартизованих даних. Це звичайний спосіб розрахунку з деякими перевагами (наприклад, відстань між двома об'єктами не змінюється, коли аналізується новий об'єкт, який може виявитися викидом). Однак на відстань може сильно впливати різниця між осями, якими ці відстані розраховуються. Наприклад, якщо одна з осей вимірюється в сантиметрах, а потім перетворюється на міліметри, остаточна евклідова відстань (або квадрат евклідової відстані), обчислений за координатами, значно зміниться, і, отже, результати кластера аналізу можуть бути дуже різними від попередніх.

Евклідова відстань у квадраті. Іноді стандартну евклідову відстань збільшують у квадрат, щоб додати ваги об'єктам, що знаходяться далі один від одного. Ця відстань розраховується так:



Відстань міських кварталів (манхетенська відстань). Ця відстань є просто середнім різниць за координатами. В основному, ця міра відстані приводить до таких же результатів, як і для звичайної відстані Евкліда. Однак відзначимо, що для. цієї міри вплив окремих великих різниць (викидів). зменшується (тому що вони не підносяться до квадрату). Манхетенська відстань обчислюється за формулою:



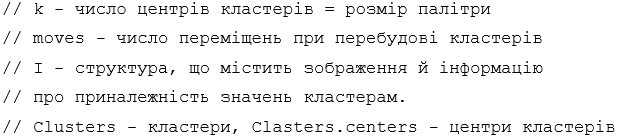
Відстань Чебишева. Ця відстань може виявитися корисною, у випадку, коли треба визначити два об'єкти як "різні", якщо вони відрізняються за якою-небудь однією координатою (одним виміром). Відстань Чебишева обчислюється за формулою:

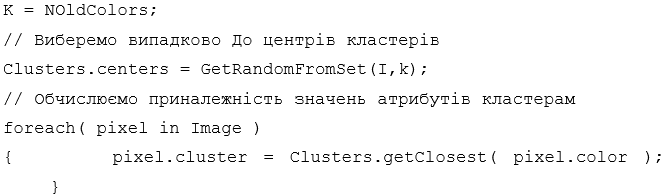


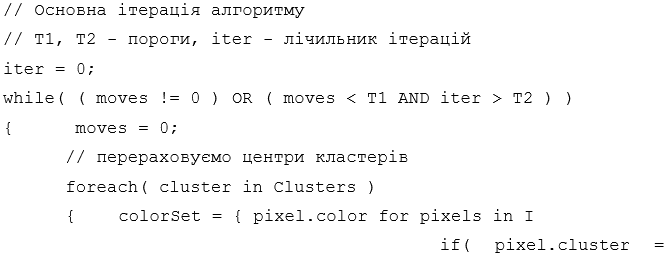
При використанні даного методу для сегментації зображення, при одноканальному зображенні ми працюємо в одномірному просторі *x*i = I(x, y). Тоді це є ітеративний алгоритм перерахування порогу. Якщо зображення трьохканальне (RGB) тоді ми працюємо в тривимірному просторі *х*i = (R(x, y), G(x,y), B(x, y)).

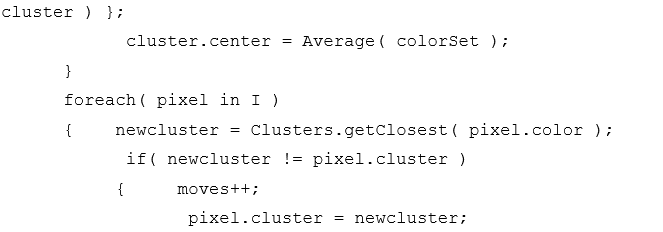
Наведемо алгоритм кластеризації колірного простору зображення. Фактично кількість кластерів - це кількість кольору в підсумковій палітрі зображення. Зафіксуємо число *k*, розмір палітри - і розділимо всі значення атрибутів зображення на k кластерів. Нижче наведено сам метод.

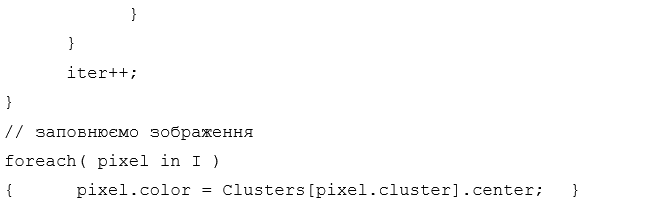
Виберемо випадково *k* значень атрибутів вихідного зображення і зробимо їх центрами кластерів. Згрупуємо точки кластерів, тобто зв'яжемо значення з кластером, центр якого найближчий до значення.







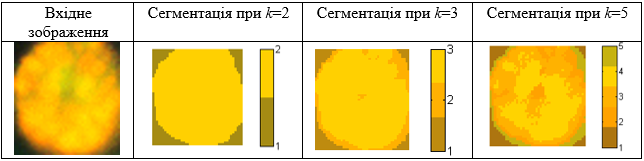




Крім того, для кожного кластера ми вказуємо його центр (тобто середнє арифметичне всіх значень, включених до кластера). Останню операцію слід повторювати до тих пір, поки не припиниться переміщення значень з одного кластера в інший або поки що після деякої (заздалегідь визначеної) ітерації відношення переміщених значень до набору не стане менше одиниці. задане значення. Таким чином, буде сформовано ряд кластерів, що відповідають палітрі. Палітра має бути заповнена центрами кластерів. Зверніть увагу, що водночас нам відома вся інформація про квантування, тобто не тільки палітра, а й індивідуальна належність значень атрибутів вихідного зображення до певного кластера.

Недоліком цього є те, що він дозволяє ефективно виділяти тільки кластери, форма яких близька до сферичної. Перевага цього методу – висока швидкість роботи. Щодо кількісної оцінки кольору зображень, цей метод показує задовільні результати (таблиця 2.1).

Таблиця 2.1 – Сегментація зображень, які тестуються



|  |  |  |  |
| --- | --- | --- | --- |
| 01-2 | 01-2-k2-manual | 01-2-k3-manual | 01-2-k5-manual |
| 06-1 | 06-1-k2-manual | 06-1-k3-manual | 06-1-k5-manual |

## 2.2 Алгоритм сегментації за водорозподілом

Такі підходи як виявлення розривів, порогове оброблення і оброблення областей мають свої переваги (швидкість) і недоліки (необхідність післяобробки). Розглянемо підхід, що базується на ідеї так званих морфологічних водорозподілів.

Поняття водорозподілу базується на представленні зображення як тривимірної поверхні, що задана двома просторовими координатами і рівнем яскравості як висота поверхні (рельєфу). У такій "топографічній" інтерпретації розглядаються точки трьох видів: (а) точки локального мінімуму; (б) точки, що знаходяться на схилі, тобто з яких вода стікає в один і той же локальний мінімум; і (в) точки, що знаходяться на гребені або піку, тобто з яких вода з рівною імовірністю стікає більш ніж в один такий мінімум. Стосовно конкретного локальному мінімуму, набір точок, що задовольняє умову (б), називається басейном (або водозбором) цього мінімуму. Множина точок, що задовольняють умову (в), утворюють лінії гребенів на поверхні рельєфу і називаються лініями водорозподілу.

Основна мета алгоритмів сегментації, базованих на наведених поняттях, полягає в знаходженні ліній водорозподілу. Припустимо, що кожен локальний мінімум заповнюється водою, що рівномірно заповнює рельєф. Коли вода, що піднімається, в двох сусідніх басейнах близька до того, щоб злитися разом, в цьому місці ставиться перегородка, що перешкоджає злиттю. Врешті-решт заповнення досягає фази, коли над водою залишаються видні тільки верхівки перегородок. Ці перегородки, відповідають лініям водорозподілів, і утворюють неперервні границі, виділені за допомогою алгоритму сегментації по водорозподілах. Щоб уникнути виливання води за межі країв всієї конструкції, уявимо, що все зображення по периметру обнесено перегородкою, що перевищує найвищу гору тобто максимально можливий рівень яскравості зображення.

Одним з найважливіших застосувань сегментації по водорозподілах є виділення на фоні зображення однорідних за яскравістю об'єктів (у вигляді плям). Області, що характеризуються малими варіаціями яскравості, мають малі значення градієнта. Тому на практиці часто зустрічається ситуація, коли метод сегментації за водорозподілами застосовується не до самого зображення, а до градієнта цього зображення. У такій постановці локальні мінімуми басейнів добре узгоджуються з малими значеннями градієнта, що зазвичай відповідає об'єктам, що цікавлять.

Розглянемо спосіб побудови перегородок вдовж ліній водорозподілів.

Побудова перегородок базується на двійкових зображеннях, які є підмножинами двовимірного цілочисельного простору *Z*2 [26, 27]. Простий спосіб побудови ліній розділення для множин, що утворені двійковими точками, полягає в використанні морфологічної дилатації.

Початкові відомості про побудову перегородок за допомогою дилатації ілюструє рисунок 2.1.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| а) | б) |  |
|  |  |
|  | в) | г) |
| Рисунок 2.1 – Два частково заповнених басейни на (*n*-1)-му кроці заповнення (а), *n*-ий крок заповнення, при якому два басейни зливаються разом (для наочності залита водою область показана білим кольором, а не чорним) (б), примітив, використаний в операції дилатації (в), результати дилатації і побудови перегородки (г) | | |

На рисунку 2.1(а) показані ділянки двох басейнів на (*n*-1)-му кроці заповнення, а на рисунку 2.1(б) ті ж ділянки на наступному, *n*-му кроці. Відбувається злиття цих двох басейнів, і, отже, повинна бути побудована перегородка що перешкоджає даній події. Позначимо через *M*1 і *M*2 множини точок, що відповідають локальним мінімумам двох розглянутих басейнів. Через С*n*-1(M1) і *Сn*(*М*2) позначимо множини точок, покритих водою в цих басейнах на (*n*-1) -му кроці заповнення. Ці дві множини показано чорним кольором на рисунку 2.1(а).

Нехай *C*[*п*-1] - об'єднання двох останніх множин. На рисунку. 2.1(а) маємо дві компоненти зв'язності, а на рисунку 2.1(б) - тільки одну, що охоплює дві колишні компоненти зв'язності (позначені пунктирними лініями). Той факт, що дві компоненти зв'язності перетворилися на одну, вказує, що на *n*-ому кроці заповнення відбулося злиття двох басейнів в один. Позначимо через *q* єдину зв'язну компоненту, що утворилася. Відмітимо, що дві компоненти, що були на кроці *n*-1 можна виділити з множини *q* однією операцією І: . Відмітимо також, що всі точки, що належать окремому басейну, утворюють одну компоненту зв'язності.

Допустимо, що до кожної компоненті зв'язності на рисунку 2.1 (а) застосовується операція дилатації по примітиву, показаному на рисунку 2.1 (в), з дотриманням двох умов:

1) застосування дилатації повинно обмежуватися множиною *q* (це означає, що центр примітива може розташовуватися тільки в точках *q*);

2) дилатація не повинна виконуватися в тих точках, де це приведе до злиття множин, так що вони стануть єдиною зв'язною компонентою.

З рисунка 2.1 (г) видно, що при першому проході дилатації межі кожної з початкових компонент зв'язності розширюються (показано світло-сірим кольором). Відмітимо, що при дилатації умова 1) виконується для всіх точок, а умова 2) не застосовувалося жодного разу, так що границі обох областей розсувалися рівномірно.

При другому проході дилатації (показаному темно-сірим кольором) для деяких точок виконувалася умова 2), але порушувалася умова 1), що привело до розривності множини точок, що додаються по периметру, як це видно з рисунка 2.1. Очевидно також, що єдиними точками множини *q*, для яких виконана умова 1) і не виконана умова 2), є точки (перекреслені хрест-на-хрест на рисунку 2.1(г)), що утворюють зв'язну лінію завтовшки в один піксел. Ця лінія і складає шукану розділюючу перегородку на *n*-му кроці підйому рівня води. Побудова перегородки на цьому кроці завершується тим, що всім точкам знайденої лінії присвоюється значення яскравості, що перевищує максимальне в зображенні. Зазвичай висота всіх перегородок приймається рівною максимально можливому рівню яскравості, плюс одиниця. Це запобігає можливості злиття басейнів поверх побудованої перегородки в майбутньому, у міру подальшого підйому води. Важливо відзначити, що перегородки, побудовані із застосуванням даної процедури, що являються шуканими границями сегментації, представляють собою зв'язні компоненти. Інакше кажучи, даний метод виключає проблеми, пов'язані з появою розривів в лініях сегментації.

Розглянемо власне алгоритм сегментації за водорозподілами.

Нехай *M*1,*M*2,...,*MR* – множина точок координатної площини, що відповідають локальним мінімумам поверхні *g*(*x*, *у*); *g*(*x*, *y*) зазвичай є градієнтним зображенням. Позначимо через *C*(*Mi*) множину точок басейну, що відповідає локальному мінімуму *Mi*. Позначення min і max будемо використовувати для позначення найменшого і найбільшого значень зображення *g*(*x*, *у*). Нарешті, запис *T*[*п*] означає множину точок (*s*, *t*), для , тобто

 (2.1)

З геометричної точки зору, *T*[*п*] є множиною точок, в котрих поверхня *g*(*x*, *у*) лежить нижче за площину g(x, у)= *n*.

При заповненні рельєфу водою рівень піднімається у вигляді цілочисельних дискретних приростів від *п* = *miп* + 1 до *п* = *тах* + 1. У процесі підйому води на будь-якому кроці *п* алгоритму необхідно знати число точок, що лежать нижче за рівень води. Уявимо, що всі точки множини *T*[*п*] (тобто ті, які лежать нижче площини *g*(*x*,*у*)=*п)* позначені чорним кольором, а всі інші – білим. Тоді при довільному (*n*-му) кроці підйому рівня води розглянута тривимірна поверхня в проекції на площину *xy* може бути представлена двійковим зображенням, в якому чорні точки відповідають точкам початкової функції, що лежать нижче за площину *g*(*x*, *у*)=*n*.

Нехай *Cп*(*Mі*) позначає множину точок басейну з локальним мінімумом *Mі*, які виявилися залитими водою на кроці *n*. Із врахуванням вищесказаного *Cn*(*Mі*) можна розглядати як двійкове зображення, що задається відношенням

 (2.2)

Іншими словами, *Cп*(*Mі*) = 1 в тих точках (*x*, *у*), для яких одночасно виконується (*x*, *у*)  *C*(*Mi*) і ; у решті точок зображення *Cn*(*Mі*) = 0. Геометрична інтерпретація виразу в правій частині (2.2) зрозуміла: за допомогою операції перетину на *n*-ом кроці підйому рівня води ми виділяємо ту частину двійкового зображення *T*[*п*], яка відноситься до локального мінімуму *Mі*.

Нехай тепер *C*[*п*] - об'єднання залитих водою частин всіх басейнів на кроці *n*:

 (2.3)

Тоді *C*[*тах*+1] є об’єднанням всіх наявних басейнів:

 (2.4)

Можна показати, що при роботі алгоритму ніколи не відбувається видалення елементів із множин *Cn*(*Mі*) і *T*[*п*]; таким чином, при збільшенні *п* число елементів цих множин або зростає, або залишається незмінним. Тому, *C*[*п*-1] є підмножиною *C*[*п*]. За рівністю (2.2) *C*[*п*] також є підмножиною *T*[*п*], а значить, *C*[*п*-1] також є підмножиною *T*[*п*]. Звідси слідує важливий результат: кожна компонента зв'язності множини *C*[*п*-1] міститься рівно в одній зв'язній компоненті множини *T*[*п*].

Алгоритм знаходження ліній водорозподілу починається з визначення *C*[min+1]= *Т*[min+l]. Після цих дій алгоритм. виконується peкурентно, припускаючи на *п*-му кроці множину *С*[*п*-1] вже побудованою. Для отримання множини *C*[*п*] з множини *C*[*п*-l] застосовується наступна процедура. Нехай *Q*[*п*] - множина компонент зв'язності множини *T*[*п*]. Тоді для кожної зв'язної компоненти  є три можливості:

а)  є порожньою множиною;

б)  містить одну компоненту зв'язності множини ;

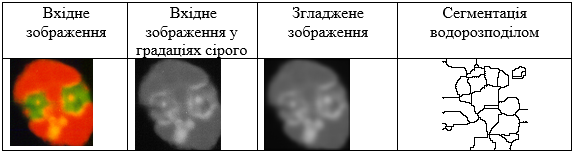
в)  містить більше ніж одну компоненту зв'язності множини .

Спосіб побудови *C*[*п*] по  залежить від того, яка з цих трьох умов має місце. Умова (а) означає, те що зустрівся новий локальний мінімум. (починається наповнення нового басейну); у цьому випадку компонента *q* додається до для побудови множини *C*[*п*]. Умова (б) має місце, коли *q* лежить усередині басейну деякого локального мінімуму; в цьому випадку компонента *q* також додається до  для побудови множини *C*[*п*]. Умова (в) виникає, коли зустрілися точки гребеня, що розділяє два або більше басейнів. В цьому випадку подальший підйом води привів би до злиття цих басейнів, тому всередині зв'язної компоненти *q* повинна бути побудована перегородка (або перегородки, якщо об'єднується більше двох басейнів), що не дозволяє басейнам злитися разом. Перегородку завтовшки в один піксел при необхідності можна побудувати, застосовуючи до множини  операцію дилатації по примітиву 33, заповненому одиницями, і потім обмежуючи результат дилатації точками множини *q*.

Ефективність описаного алгоритму можна підвищити, використовуючи тільки ті значення *п*, які відповідають рівням яскравості що зустрічається в зображенні *g*(*x*, *у*); ці значення, як і величини min і max, можна визначити за гістограмою зображення *g*(*x*, *у*).

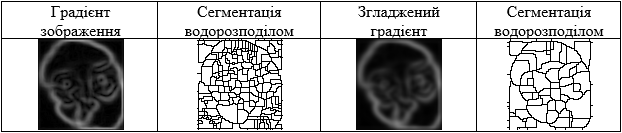
Проведемо сегментацію тестових зображень клітин (таблиці 2.2-2.4).

Таблиця 2.2 – Сегментація за полем яскравості



Для всіх наступних експериментів застосовано 8-ми зв’язність. Сегментацію кольорового зображення можна проводити за його яскравістю, за його градієнтом або за кольоровим відтінком в системі HSV [26]. На наведених ілюстраціях є багато ліній водорозподілів, які не оточують об'єкти, що нас цікавлять. Це інший випадок надмірної сегментації.

Таблиця 2.3 – Сегментація за полем градієнта яскравості



Таблиця 2.4 – Сегментація за полем кольорового тону

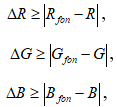


Надмірна сегментація може бути настільки значною, що зробить результат обробки практично даремним. У контексті нашого обговорення це означає величезне число областей, виділених при сегментації. Практичне розв’язання цієї проблеми полягає в тому, щоб обмежити допустиме число областей шляхом включення до складу процедури кроку попередньої обробки, що служить для привнесення додаткової інформації в процедуру сегментації.

Підхід, що вживається для управління надмірною сегментацією, базується на ідеї маркерів. Маркер є зв'язною компонентою, що належить зображенню. Розрізнятимемо внутрішні маркери, що відносяться до інтересуючих нас об'єктів, і зовнішні маркери, що відповідають фону зображення. Потім ці маркери використовуються для підправлення градієнтного зображення. Розроблені різноманітні методи для побудови внутрішніх і зовнішніх маркерів, багато з яких використовують лінійну або нелінійну фільтрацію, а також морфологічне оброблення. Вибір методу сильно залежить від природи конкретних зображень, які будуть оброблятися.

## 2.3 Алгоритм сегментації шляхом виділення контурів зображень

Для визначення зовнішньої межі зображення використовувався метод, при якому ділянки зображення не аналізуються, якщо їх колір збігається або близький до кольору зображення з декотрою помилкою Δ [28, 29]. Цей підхід визначає пікселі фону. Колір точки (пікселя) передається у просторі RGB. Щоб точку можна було розпізнати як частину фону, значення її компоненти RGB має відповідати таким умовам:

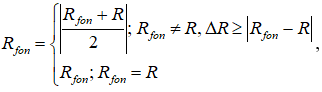
 (2.5)

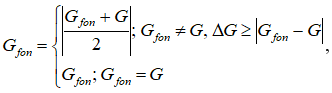
де ΔR, ΔG, ΔB – величина можливого відхилення компонентів RGB від заданих значень фону;

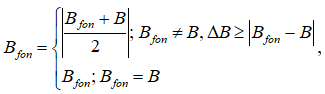
R, G, B – колірне значення фону в базі RGB;

R, G, B – колірне значення базового пікселя RGB;

Для покращення результатів алгоритму пропонується використовувати динамічне значення кольору тла. Значення динамічного фону визначається за такими правилами:

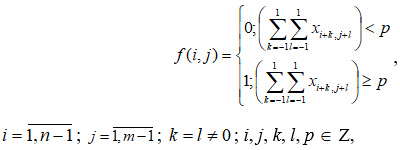


 (2.6)



Використання динамічного фону може зменшити ефекти шуму та спотворення, спричинені датчиками камери. Недоліком такого підходу є зниження ефективності роботи з недостатньою якістю зображення (шум та спотворення перекривають зображення осередків або мають відтінок, подібний до кольору осередку протягом значного періоду часу), тому перед запуском Програма, при необхідності зображення вимагає додаткової обробки.

Щоб відсікти неінформативні точки, а також подолати деякі недоліки, що виникають під час фотографування (окремі точки (пікселі), які відрізняються від фону, але не є частиною осередків), сусідні пікселі перевіряються на приналежність до фону. Для цієї перевірки вводиться відображення рисунка зі значеннями 0 та 1, де 0 – точка, яка належить, а 1 – не належить фоновому режимі. Алгоритм фільтрації зображень виконується за такими правилами:

 (2.7)

де *f*(*i*,*j*) – логічна функція, що визначає наявність зв'язків із сусідніми пікселями;

*xi,j* - значення маски для *i*-го, *j*-го пікселя;

*п* – в пікселях ширина зображення;

*т* – в пікселях висота зображення;

*р* – мінімальна кількість сусідніх точок, необхідних присвоєнню точці фонового значення приналежності.

Число *р*= 6 було визначено експериментально як найоптимальніший критерій якості/продуктивності. Крім того, зі збільшенням параметра *p* якість обробки зображення збільшується, але швидкість роботи програми знижується, але при високих значеннях параметра *p* відкидаються точки, які можуть мати інформативне значення. Приклад дефініції параметра *p* показаний рисунку 2.2. Для точки *M*(*x*, *y*) управління сусідніми точками почнеться з точки Р8 (оскільки попередня точка контуру була Р6), потім за годинниковою стрілкою P1, P2, P3, P4, P5, P6, потім управління відповідно до правила (3) можна припинити, тому що *p* = 6. Як показано на рисунку 2.2, значення *p* = 6 цілком достатньо, щоб розпізнати точку як інформативну.

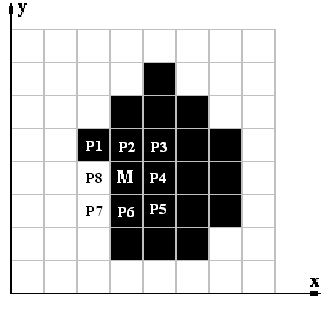


Рисунок 2.2 – Визначення параметра *р*

Для визначення точок зовнішнього кордону (контуру) зображення використовується такий алгоритм:

1. вибір початкової точки;
2. під час руху за годинниковою стрілкою вибирається наступний (сусідний) піксель, що межує з ним, але не належить фону;
3. перевіряється наявність наступної (сусідної) точки, яка відповідає умовам належності точки контуру. В противному переходимо до кроку 5);
4. при позитивному результаті кроку 3 (координати активної точки не змінилися) виконується відкат. У цьому випадку призначається точці фоновий стан, а попередня точка вибирається як активна і виконується перехід до кроку 2);
5. перевірка проводиться на закінчення визначення (замикання) контуру. Активна точка повертається у вихідне положення із використанням критерію Джейкобса. Якщо умова не виконується, перехимо на крок 2);
6. після отримання замкнутого контуру (програма успішно виділила цілу область зображення, відмінну від фону), кожній точці зображення з обмеженим контуром надається ідентифікатор осередку, щоб уникнути повторної обробки пікселя дані, а також для полегшення подальшої роботи над зображенням та виділеною областю. Після закінчення місії переходимо до пункту 1).

Для перевірки якості алгоритму тестового зображення використовувалася фотографія пухлинних клітин (рисунок 2.3). Результат обробки зображення показано на рисунку 2.4.

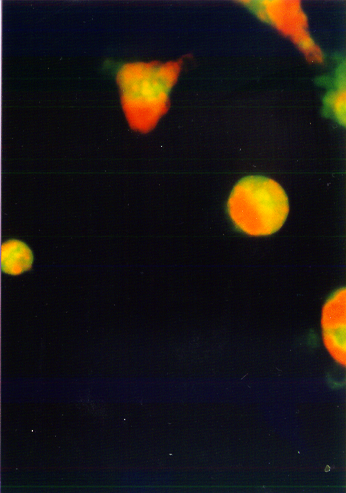
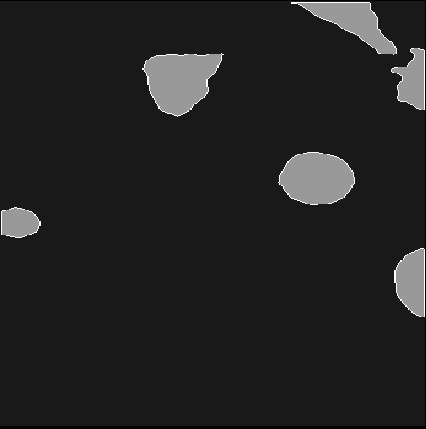
 

Рисунок 2.3 – Фотографія, яка Рисунок 2.4 – Оброблена фотографія

здана в обробку (з виділеними контурами)

На основі запропонованих алгоритмів розроблено програмний комплекс мовою програмування Delphi 6.0 [7, 8, 30-36]. Програма виділяє зовнішні контури клітин та проводить ряд досліджень, у тому числі форми клітини. Форма клітини визначається співвідношенням відрізка, що з'єднує дві найвіддаленіші точки контуру A і B, і перпендикуляра, що проходить через середину відрізка AB, точку M і точку контуру С. Програма в результаті роботи комплексу розраховує ряд геометричних параметрів клітини з можливістю подальшого аналізу.

## 2.4 Висновки до розділу 2

У розділі аналізуються алгоритми виділення контурів зображення та пропонується . алгоритм визначення зовнішнього контуру (краю) зображення. В розділі представлені результати роботи програмної системи. Цей алгоритм можна використовувати для розпізнавання зображень (зовнішнього контуру зображення) в системах розпізнавання зображень.

Для подальшого розвитку та підвищення ефективності процесу проходження . контуру необхідно розробити критерії правильних алгоритмів зупинки та методи більш точної обробки виступів зображення (товщиною одного пікселя).

3 ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМІВ СЕГМЕНТАЦІЇ ЗОБРАЖЕНЬ

## 3.1 Узагальнена структура програмної системи

Для написання програмного засобу для проведення тестування використовувалось програмне середовище Borland Delphi 7.0. Програмне середовище – це комплекс програмних засобів, що дозволяють реалізувати принципи швидкого програмування програм RAD (Rapid Application Development) [37, 38]. Borland Delphi 7.0 має ряд переваг, зокрема [39-43]:

* 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 необхідна робоча станція на базі операційної системи Windows [44-46]. Вибір даної операційної системи пов'язаний наступними перевагами:

* широка розповсюдженість операційних системи сімейства 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;

На схемі представлено структурну схему програми (рисунок 3.1):

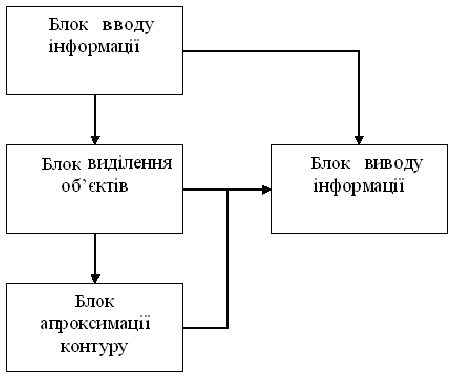


Рисунок 3.1 – Структурна схема програми

Розбиття програми на модулі відбувалось на основі подібності виконуваних дій. В програмі виділено наступні модулі:

1. Блок вводу інформації. який забезпечує інтерфейс вводу зображення в програму, перекодування у необхідний формат, додаткову обробку вхідного зображення. Даний модуль включає наступні процедури: LoadToMemory(), FontColorSell(integer; integer; integer) та функція PictureVisibl(). Функція PictureVisibl() здійснює перевірку на наявність зображення в робочій області програми. Якщо значення функції „false” то робота програми блокується. Оскільки програма працює з зображеннями в форматі „\*.bmp”, а на сьогоднішній час розроблено ряд інших форматів, то виникає необхідність процедури перекодування вхідного зображення у робочий формат. Оскільки формат „\*.jpg” є одним з найпоширенішим, то в програмі реалізовано перекодування з формату „\*.jpg” в формат „\*.bmp”. Для проведення порогової сегментації необхідно отримати матрицю яскравостей зображення, для цього зображення перекодовується з базису RGB в базис HLS (FontColorSell(integer; integer; integer)). На етапі додаткової обробки також будується гістограма яскравостей з метою визначення порогу сегментації. Результатом роботи даного модуля є визначене вхідне зображення, матриця яскравостей та поріг сегментації.

2. Блок виділення об’єктів, що здійснює набір процедур та функцій, який реалізує процес сегментації та визначення інформативних ознак об’єктів. Даний модуль включає наступні процедури: PixelFon(), DelKlitunu(integer, integer, integer, integer), CentralLine(integer). Процедура сегментації реалізується на базі алгоритму порогової сегментації. Після виділення зв’язних областей необхідно контурну функцію, тобто пройти контуром області. 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 виділення об’єкти та масив метричних ознак (проща, периметр, компактність тощо).

3. Блок апроксимації контуру реалізує процес кусково-поліноміальної апроксимації контуру об’єктів з метою отримання аналітичних ознак об’єктів. Даний модуль включає наступні процедури: TochkuAproxVerx(integer), TochkuAproxNuz(integer), Aprox(integer;integer;integer; integer; integer), AproxLine(integer;integer;integer; integer; integer). Для зменшення трудомісткості процесу апроксимації з вхідного масиву точок, що формують контур об’єкта, проводиться вибірка характерних точок. На основі аналізу вибірки визначальних точок проводиться вибір кривих для апроксимації контурної функції. Алгоритм апроксимації реалізований у процедурах Aprox(integer;integer;integer; integer; integer) для гіпербол та парабол та AproxLine(integer;integer;integer; integer; integer) для апроксимації відрізків. Результатом роботи модуля є масив аналітичних ознак об’єкта, тобто набір кривих n-го порядку за допомогою яких було апроксимовані контури об’єктів.

4. Блок виділення об’єктів реалізує процес сегментації та визначення інформативних ознак об’єктів. Даний модуль включає наступні процедури: PaintImage(), PaintKontyr(), showPoworotklitunu(nomer:integer), showRezyltat(integer). Процедури реалізовані в даному модулі забезпечують вивід результаті роботи кожного з модулів програми. Даний підхід дозволяє користувачу аналізувати не тільки кінцеві, але і проміжні результати роботи програми, що дозволяє отримати більшу кількість аналітичної інформації. Результатом роботи модуля є інтерактивних вивід поточних результатів роботи програми.

На рисунку 3.1 показані взаємозв’язки між різними модулями. Дана архітектура зв’язків забезпечує цілісність програми, дозволяє мінімізувати передачу даних між модулями. Оскільки модулі працюють автономно то на вихід кожного модуля подається лише результат роботи при цьому додаткові змінні залишаються всередині нього та вивід поточної інформації в автоматичному режимі, що дозволяє знизити вплив користувача на роботу програми вцілому.

Для зручності користування роботи з програмою було спроектовано інтуїтивно зрозумілий дизайн (рисунок 3.2), що окрім зручності забезпечує швидке опановування програмою користувачі без досвіду роботи не тільки з даною програмою, але і з подібними програмними засобами.

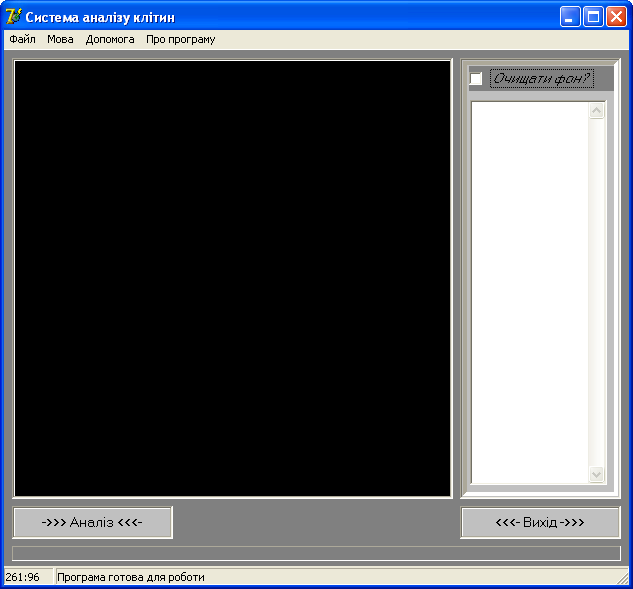


Рисунок 3.2 – Загальний вигляд основного вікна програми

В верхньому меню програми присутні наступні пункти:

1. „Файл” - забезпечує роботу з файлами.

2. „Завантажити” - дозволяє завантажувати в програму зображення для подальшого дослідження. Завантаження здійснюється в робочу область програми.

3. „Зберегти” - дозволяє зберігати результати обробки зображень для подальшого аналізу або продовження роботи. Робоча область зберігається у файл з розширенням „\*.bmp”.

4. „Вихід” - завершення роботи програми.

5. „Мова” - дозволяє вибрати робочу мову інтерфейсу програми. На сьогоднішній день реалізовано україномовний та англомовний інтерфейс користувача.

6. „Допомога” - виводиться додаткове вікно програми в якому перераховані питання які найчастіше виникають при роботі з програмою. Даний пункт меню реалізований для швидшого навчання роботи з програмою нових користувачів.

7. „Про програму” - виводиться інформація про програмний продукт: версія програми, дата виходу, інформація про розробників програмного засобу, контактна інформації.

Окрім меню на головному вікні програми присутні наступні об’єкти:

1. „Вікно робочої області” – в даному вікні відображаються зміни, що проводяться над вхідним зображенням: сегментації зображення, виділення контурів, присвоєння кожному об’єкту унікального ключа‑ідентифікатора, виділення великої осі, тощо.

2. „Вікно допоміжної робочої області” – дана область призначена для інтерактивного виводу аналітичної інформації, зокрема інформацію про координати активної точки (точка над якою проводяться певні дії) та активним об’єктом (об’єкт над яким проходить дослідження), координати точок які формують контури об’єктів, метричні ознаки (периметр, площа, координати точок які утворюють велику вісь, компактність, кут нахилу клітини до осі абсцис, тощо). Перевагою даного способу виводу інформації є її компактність та максимальна інформативність, що дозволяє досвідченому користувачу отримувати додаткову аналітичну інформацію опрацьовуючи цей масив даних. Недоліком можна вважати складність для недосвідченого користувача розібратись з структурою інформації.

3. „Очистка фону” – даний компонент забезпечує вивід об’єктів у робочу область, при цьому кольори всіх точок, що не належать об’єктам замінюються на один, наперед заданий колір. Якщо компонент є активним, у відповідному місті стоїть позначення, то колір фонових точок замінюється на один наперед заданий колір, при розробці та тестуванні програмного засобу використовувався чорний колір. Якщо компонент не є активним, відсутні позначення у відповідному місті, то заміна колір фонових точок не змінюється.

4. Кнопка „Аналіз” активує роботу набору функцій по обробці, опису та аналізу вхідного зображення. Функції обробки вхідного зображення запускаються послідовно в автоматичному режимі, що дозволяє знизити вимоги до навиків роботи з програмою користувачів.

5. Кнопка „Вихід” – активується функція завершення роботи програми. Вона дублює можливості пункту верхнього меню „Вихід”. і спроектована для зручності роботи користувачів з програмним засобом.

6. „Лінійка стану виконання процесу” – даний компонент в процентному відношенні показує стан виконання відповідного процесу чи функції і його було спроектовано для візуалізації процесу обробки вхідного зображення з метою надати користувачу інформації про темпи виконання завдання.

7. „Індикатор активного завдання” – даний елемент інтерфейсу програми показує користувачу що саме виконується програмою в даний час. Елемент складається з двох части. У лівій частині в інтерактивному режимі виводяться координати курсору в робочому вікні програми, у правій частині – інформація про дії, що виконує програма.

Окрім основного вікна програми для виводу результатів роботи програми було розроблено два додаткових вікна: вікно виводу результатів апроксимації та вікно виводу результатів повороту клітини (рисунок 3.3). В результаті обрання даного архітектурного вираження зменшується інформативне навантаження на головне робоче вікно програми та дозволяє виводити одночасно більшу кількість інформацій для аналізу. Тобто, одночасне використання трьох вікон дозволяє зробити роботу користувача більш ефективною.

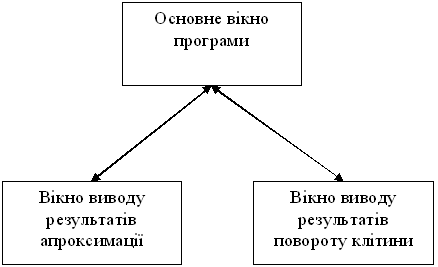


Рисунок 3.3 – Схема взаємодії вікон програми

Вікно виводу результатів повороту клітини візуально ілюструє процес повороту клітини на кут α та перехід від глобальних координат до локальних (рисунок 3.4).

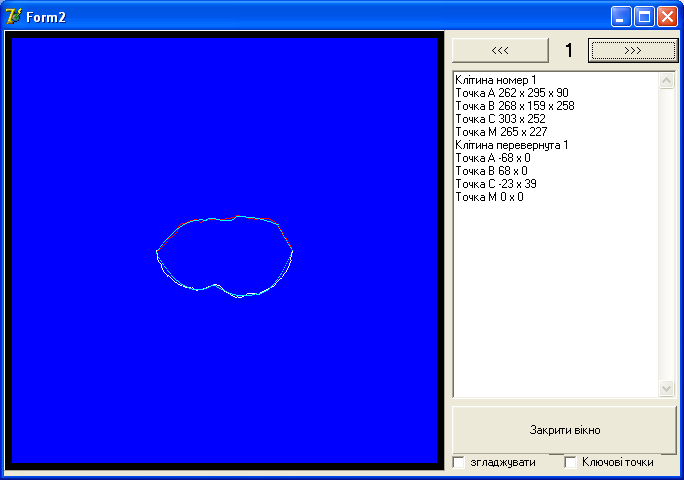


Рисунок 3.4 – Вікно виводу результатів повороту клітини

Кут α це кут між великою віссю та віссю абсцис. Перехід від глобальних координат до локальних здійснюється шляхом переносу точки початку відліку з верхнього лівого кута зображення в точку М(х,у) середину великої осі.

Окрім візуалізації повороту клітини для більшої інформативності додатково можна вивести визначальні точки контуру (точки на основі яких буде здійснюватися апроксимація) та текстури об’єкта. Для отримання необхідного представлення необхідно відмітити відповідні пункти меню „Визначальні точки” і/або “Згладжувати” в правому нижньому куті робочого вікна.

Якщо на зображенні виділено більше одного об’єкта, то перехід від одного об’єкта до іншого здійснюється шляхом вибору відповідних пунктів меню „>>” (вперед) та „<<” назад.

Для закриття вікна слід натиснути кнопку „Закрити вікно”.

Вікно виводу результатів апроксимації контуру об’єкта ілюструє результати апроксимації контурної функції (рисунок 3.5).

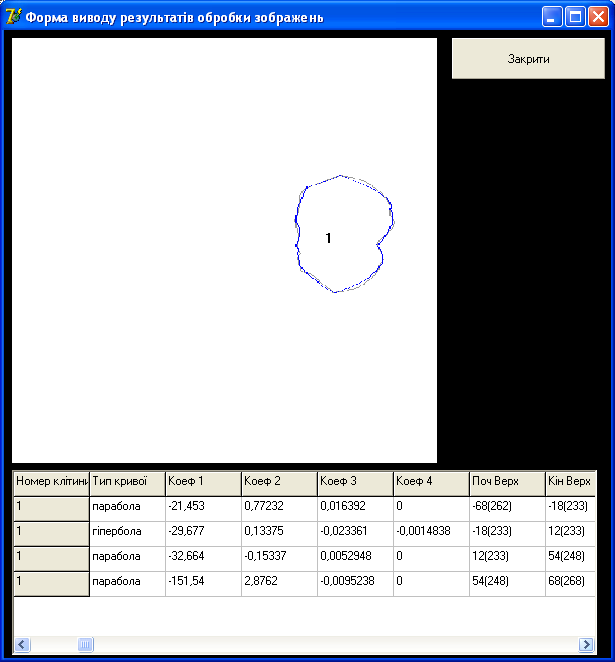


Рисунок 3.5 – Вікно виводу результатів апроксимації

За допомогою даного вікна можна візуально порівняти контури які отримуються в результаті кодування контуру за допомогою табличного представлення (контур промальовується сірим кольором) та отриманого за допомогою апроксимації (контур промальовується синім кольором), візуально оцінити якість апроксимації. Окрім візуальної інформації в робочому вікні виводяться аналітична інформація, що характеризує апроксимуючі криві. Дані представлені в табличному режимі: послідовність, тип кривих та їхні коефіцієнти, похибка. Тип кривої *n*-го порядку може приймати значення пряма, парабола, гіпербола. Як показали результати експериментів, найкращі результати отримуються при поєднанні різних типів кривих. Коефіцієнти виводяться в зворотному порядку, тобто: коефіцієнт 1 – це значення при х0, коефіцієнт 2 – це значення при х1, і т.д. Окрім цього візуалізується інформація про початок та кінець кривої, яка апроксимує частину контуру. Кінець та початок апроксимуючої кривої виводиться як в локальних (початок відліку знаходиться на середині відрізка, що з’єднує дві найбільш віддалені точки контуру) так і в глобальних координатах (початок відліку знаходиться в лівій верхній точці зображення). Додатково виводиться інформація про похибку кожної з апроксимуючих кривих n-го порядку як в кожній точці, так сумарно на всій довжині дуги.

Клас TSell був розроблений для збереження інформації про об'єкти. Серед параметрів цього класу - інформація про характеристики об'єкта, масив точок, що утворюють зображення, кількість і тип кривих, якими апроксимується контур об'єкта, додаткові характеристики тощо. Клас TSell має такі параметри:

* namber – унікальний ідентифікаційний ключ для кожного об'єкта. Цей ключ використається для роботи з об'єктом;
* рerumetr – цей параметр зберігає інформацію про довжину контурної лінії. Периметр об'єкта визначається традиційно, як сума відстаней між сусідніми точками контуру зображення;
* plowa – цей параметр зберігає інформацію про площу об'єкта. Оскільки на досліджуваних об'єктах немає «дірок», площу можна розрахувати як суму всіх точок, що належать об'єкту;
* fontcolor – цей параметр зберігає інформацію про колір фону, на якому знаходиться об'єкт. Цей принцип забезпечує велику гнучкість при обробці зображень, оскільки для кожного об'єкта можна визначити (автоматично або вручну) колір фону, що забезпечує більшу точність. Однак правильний колір тла знижує продуктивність програми;
* matrix – цей параметр являє собою масив для зберігання координат (*х*, *у*) контуру об'єкта. Збереження інформації про лінії контуру дозволяє виконувати додаткові операції без повторного визначення контуру. Наприклад, до додаткових операцій відносять визначення контурної функції з допомогою простих векторів, кодів Фрімена, наближення контурної функції, визначення параметрів об'єктів (компактність, периметр, визначення осей тощо і ін.);
* MaxX – критична точка об'єкта, найбільш віддалена від осі *x* на площині XoY з початком координат у лівому верхньому куті зображення, тобто .
* MinX – критична точка об'єкта, що знаходиться на мінімальній відстані від осі x на площині XoY з початком координат у лівій верхній точці зображення, тобто .
* MaxY – критична точка об'єкта, найвіддаленіша від осі Y на площині XoY з початком координат у лівій верхній точці зображення, тобто .
* MinY – критична точка об'єкта, що знаходиться на мінімальній відстані від осі Y на площині XoY з початком координат у верхній лівій точці зображення, тобто .
* M – інформація про координати точки, що містить середину головної осі. Ця точка є центром описаного кола для цього об'єкта;
* bl – параметр b рівняння , що описує головну вісь об'єкта.
* k – параметр k рівняння , що описує головну вісь об'єкта;
* dovzhunaSer – довжина великої осі, що дорівнює відстані між двома точками, найбільш віддаленими від контуру об'єкта. Визначається як:



де  і  координати найбільш віддалених один від одного точок на контурі об'єкта.

* kompactnist – коефіцієнт, що показує співвідношення периметра об'єкта до його площі. Визначається за такою формулою:



де *А* – площа об’єкта;

*Р* – периметр досліджуваного об’єкта.

Чим ближче Kol до 1, тим форма об'єкта більше схожа на коло. Інша назва цієї характеристики – округлість;

Для виконання кусково-поліноміальної апроксимації контур об'єкта розбивається на дві половини (дуги): верхню та нижню. Розподіл здійснюється на підставі даних головної осі. Для компактного зберігання інформації про криві, за допомогою яких апроксимується контур, у класі TSell введено такі параметри:

* verx – масив координат точок, що утворюють верхню дугу контуру об’єкта;
* nuz – масив координат точок, що утворюють нижню дугу контуру об’єкта;
* verxDovzhna – довжина верхньої дуги;
* nuzDovzhna – довжина нижньої дуги;
* KilverxAprox – число вузлових точок для наближення верхньої дуги;
* KilnuzAprox – число вузлових точок для наближення нижньої дуги;
* KilkistKyskivVerx – число кривих за допомогою яких апроксимується верхня дуга контуру об’єкта;
* KilkistKyskivNuz – число кривих за допомогою яких апроксимується нижня дуга контуру об’єкта;
* to4kuVerх – координати та тип визначення точок (зміна знака функції контуру, відношення приросту, рівень кривизни) для наближення верхньої дуги;
* to4kuNuz – координати та тип визначення точок (зміна знака функції контуру, відношення приросту, рівень кривизни) для наближення до нижньої дуги;
* SplaunuVerx – інформація про криві, якими апроксимується верхня дуга контуру об'єкта (початок і кінець кривих, коефіцієнти похибки тощо);
* SplaunuNuz – інформація про криві, якими апроксимується верхня дуга контуру об'єкта (початок і кінець кривих, коефіцієнти похибки тощо).

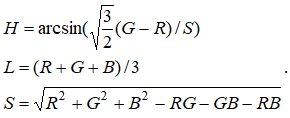
Така структура та модульна архітектура програмного засобу забезпечують зручність роботи з програмним кодом, можливість видалення, зміни, додавання нових модулів та функцій, що забезпечує більшу гнучкість програмного засобу. До недоліків можна віднести складність перенесення коду з програми на роботу станції, на якій немає операційної системи сімейства Windows.

## 3.2 Алгоритми роботи програмної системи

Обробка зображення полягає у виконанні над ним певних дій для отримання певного результату (поліпшення якості, модифікація зображення, виділення областей тощо). Вибір однотипних областей – одне з найважливіших завдань цифрового оброблення зображень. Більшість із них засноване на аналізі функції яскравості [26].

Оскільки вхідне зображення знаходиться в основі RGB, ви повинні отримати представлення, що базується на яскравості вхідного зображення, щоб виконати сегментацію. Ця операція виконується у процедурі FontColorSell (integer, integer, integer). Ця презентація не завжди візуально інформативна для людського ока, проте дозволяє швидко та ефективно працювати із зображеннями за допомогою комп'ютерних технологій.

Сьогодні існує множина підходів та баз (L\*a\*b\*, YIQ, HLS, HSB), які допускають розгляд зображення як набір яскравості в кожній точці. На базі аналізу наслідків проведених експериментів було використано базу HLS (відтінок, яскравість, насиченість) для отримання яскравого представлення зображення. Перетворення з бази RGB на базу HLS здійснюється за такими формулами [26]:



Оскільки для сегментації потрібні значення яскравості, розрахунок інших параметрів можна пропустити.

У процедурі PixelFon() відбувається процес сегментації. У вхідному зображенні вибір інформативних пікселів проводиться шляхом порівняння яскравості в точці, що досліджується, з пороговим значенням:



де *М(х,у)* – значення точки в масці зображення;

*f(А(х,у))* – значення функції яскравості в точці *А(х,у);*

*δ* – значення порогу.

Визначення порога – одне з найважливіших завдань теорії обробки цифрових зображень. Це пов'язано з тим, що надто високе значення порога яскравості може призвести до втрати інформаційних точок, що, у свою чергу, може призвести до послаблення якості обробки зображення. Однак недостатнє граничне значення може призвести до того, що фонові точки будуть називатися інформативними, що, в свою чергу, призведе до деякої кількості надмірної інформації (шум зображення), що призведе до зниження продуктивності пристрою. програма через збільшення кількості інформації, що надходить, і погіршення якості сегментації.

Програма використовує підхід гістограми до визначення порога. Цей підхід включає побудову гістограми яскравості та визначення порога у точці, де на гістограмі виявляється різке зростання рівня яскравості. Позаяк завчасно відомо, що фон зображень має вельми меншу яскравість (фон наближається до чорного), використання цього підходу дає хороші результати.

Для підвищення якості виділення нефонових точок у системі виконується додаткова перевірка на наявність необхідної кількості інформаційних пікселів. Збиток цінної інформації за такого контролю, як показують експериментальні дані, незначна.

В результаті обробки вхідного зображення формується двійкове зображення, в якому інформаційні пікселі кодуються як 1, а пікселі фону кодуються як 0.

Для виділення контурів певних ділянок системи [47-52] використовується удосконалений (адаптований) алгоритм радіального сканування („Radial Sweep”) з комбінованим критерієм зупинки Джакобса („Jacob's stopping criterion”) та зупинкою при наближенні до точки, що належить контуру. Головний результат покращення – можливість повороту на *п* кроків назад, що дозволяє уникнути прив'язки до контуру лініями 1 піксель. Експериментально доведено, що лінії завтовшки 1 піксель, як правило, є дефектом зображення (шум, низька якість фотоприймача тощо) і малоінформативні.

Критерій зупинки завершує роботу алгоритму, коли контур досягає точки, яка вже належить контуру, що дозволяє уникнути зациклювання системи.

Після вибору об'єктів необхідно їх описати. Описом зображення є процес виділення інформаційних характеристик об'єкта. Зокрема однією з важливих характеристик об'єкта є велика вісь. Щоб визначити велику вісь, необхідно визначити дві найвіддалені точки на контурі.

Для знаходження головної осі реалізовано процедуру CentralLine (integer). Початкова точка (піксель) з координатами  вибирається випадковим чином на контурі клітини, як показано рисунку 3.6.

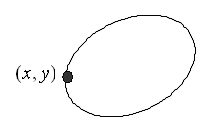


Рисунок 3.6 – Вибір початкової точки визначення головної осі об'єкта

Потім визначається наступна після початкової точки з координатами  точка контуру. Визначається вибором із масиву точок, що належать контуру об'єкта. Після визначення найближчої точки початкова точка  видаляється з подальшої обробки.

Рахується довжина відрізку  відповідно до формули:



Після виконання цих дій буде знайдено наступну точку контуру , що примикає до цієї точки. Точка  виключається з подальшої обробки, обчислюється довжина відрізка | |,

Описані кроки повторюються доти, доки не знайдено точку , на якій сегмент утворений точками  і  буде максимальної довжини , як показано рисунку 3.7. Критерієм максимальної довжини сегмента є зменшення або відсутність зростання довжин сегментів на наступних  етапах, де *P* – периметр об'єкта.

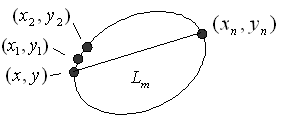


Рисунок 3.7 – Визначення першої визначальної точки головної осі контуру об’єкта

Точка (піксель) із координатами  буде першою визначальною точкою контуру об'єкта. Друга визначальна точка, яка за визначенням є найвіддаленішою від першої, шукається так само, як і перша визначальна точка. Як початкова точка вибирається задана точка з координатами . Точки обробляються у тому напрямі, у якому було визначено першу точку. Після кроків *т* буде отримана друга визначальна точка. Її координати: , вона зображена на рисунку 3.8. Точки  і  утворюють головну вісь, тобто вони утворюють сегмент, що з'єднує дві точки найбільш віддалені від контуру об'єкта. Головна вісь, без сумніву, є частиною прямої , що перетинає клітину і має найбільшу довжину .

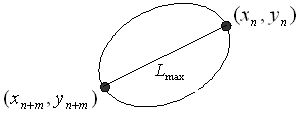


Рисунок 3.8 – Визначення головної осі об’єкта

Головна вісь – одна з основних метричних характеристик об'єкта. Цей метод визначення головної осі забезпечує високу точність результату за мінімальної трудомісткості процесу. Недоліком цієї техніки є необхідність пошуку кожної точки контуру. Шлях можна пройти з певним кроком *k,* проте цей варіант може подолати недолік, але, ймовірно, негативно позначиться на точності роботи. Також до мінусів можна віднести кількість кроків у критерії зупинки пошуку під час пошуку визначальних точок. Як показують результати тесту, зменшення кількості кроків у деяких випадках може мати негативні результати якості визначення головної осі.

Для спрощення апроксимації контурів об'єкта осередок повернено на кут α. Для повороту клітини виконується процедура Poworotklitunu(integer). Для повороту клітини будується координатна сітка з початком координат у точці M у середині відрізка AB. Відрізок AB з'єднує 2 найдальші точки контуру і найбільшу хорду клітини. Кут між віссю абсцис і найбільшою хордою обраний як кут повороту.

Кут визначається як arctan() нахилу прямої AB:



*k* – нахил рівняння прямої ;

α – кут повороту клітини.

Щоб повернути клітину як ціле, повертають кожну точку, що належить цій клітині. Розрахунок нових значень координат точок повороту здійснюється за такими формулами:



де *і* – загальна кількість точок, що ввійшли до контуру клітини, *і* = 1,…, *п*;

*xі* – абсциса *і*-ї точки контуру зображення до проведення повороту;

*yі* – ордината *і*-ї точки контуру зображення до повороту;

*Хі* – абсциса *і*-ї точки контуру зображення після повороту щодо точки M на кут α;

*Yі* – ордината *і*-ї i-ї точки контуру зображення після повороту щодо точки M на кут α;

α – кут повороту клітини.

Для зменшення обчислювальної складності при апроксимації від точок контурної лінії вибираються точки, в яких відбуваються найістотніші переміни (процедури TochkuAproxVerx. (integer) та TochkuAproxNuz. (integer) для дуг верхньої та нижньої відповідно). Для підвищення точності апроксимації вузли вибираються за трьома критеріями:

* + змінити знак приросту кривої. Цей критерій знаходить точки дуги AB, у яких змінюється знак приросту. Для визначення цього критерію необхідно: між двома сусідніми точками, які належать прямій лінії та вибраними з деяким кроком *п*, провести пряму лінію, яка описується рівнянням , та обчислити коефіцієнт *k*. Приклад визначення цього критерію показаний рисунку 3.9. На рисунку 3.9 між сусідніми точками, що належать дузі AB і взятими з відповідним кроком *п*, К(х1,у1) та М(х2,у2), коефіцієнт *k* має позитивне значення, а між точками М(х2,у2) та С(х3,у3) – негативне, тому точка М(х2,у2) буде вузловою.

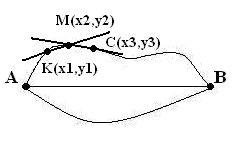
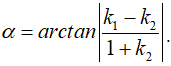


Рисунок 3.9 – Дефініція точки, де змінюється знак контурної функції

* визначення рівня кривизни функції. Цей критерій знаходить точки дуги AB, у яких кривизна дуги є більшою за граничний поріг d. На підставі експериментальних даних було визначено оптимальне значення точки відсічення d, що дорівнює 2,5. Приклад визначення цього критерію показано на рисунку 3.10. При визначенні кривизни дуги виконується низка додаткових обчислень. 1) Визначаються нахили відрізків, що поєднують дві . сусідні пари точок, що належать до прямої та обраних з певним кроком *п*. 2) Рисуться прямі, що описуються рівняннями  та  і обчислюються коефіцієнти k1 та k2. 3) Визначається довжина відрізків, що сполучають сусідні точки:



4)  коефіцієнт α визначається:



5) Кривизна дуги К визначається:



Для рисунка 3.10 між сусідніми точками, що належать дузі AB і взятими з відповідним кроком *п*: К(х1,у1), М(х2,у2) та С(х3,у3), кривизна дуги є більшою, аніж значення граничного порога d, тому, точка М(х2,у2) буде вузловою.

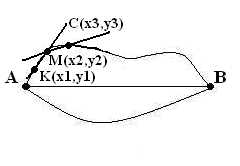


Рисунок 3.10 – Демонстрація пошуку точки, в якій кривизна контурної функції перевищує певний поріг

* відношення приростів функції до сусідніх точок. Цей критерій знаходить точки дуги AB, у яких співвідношення приростів . відрізняється більше ніж в *т* разів. Для визначення цього критерію необхідно між двома сусідніми парами точок, що належать прямій лінії та вибраних з певним кроком *п*, провести прямі, що описуються рівняннями  та , та обчислити коефіцієнти k1 та k2. На рисунку 3.11 показано визначення точки для цього критерію. Між сусідніми точками на рисунку К(х1,у1) й М(х2,у2), що належать дузі AB і взяті з відповідним кроком *п*, коефіцієнт k1 більший, ніж коефіцієнт k2, взятий між точками М(х2,у2) й С(х3,у3) в *т* разів, що дозволяє визначти точку М(х2,у2) як вузлову. Коефіцієнт *т* = 1,5 отриманий експериментально, і це є оптимально.

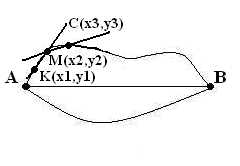
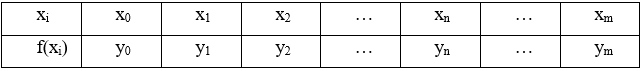


Рисунок 3.11 – Демонстрація пошуку точки, в якій приріст контурної функції перевищує певний поріг

Для перевірки добротності апроксимації, а також більш точної передачі конфігурації кривої використовується 4-ий тип визначальних точок – додаткові точки. Ці точки розраховуються, якщо довжина підібраної кривої більша, ніж коефіцієнт *γ*. Коефіцієнт *γ* розраховується у точках (пікселях) між двома сусідніми визначальними точками, що визначаються з використанням трьох критеріїв визначальної точки. Експериментально встановлено, що з якісної роботи алгоритму коефіцієнт *γ* ≥ 20. Приклад визначення цього критерію показаний рисунку 3.12. Точка L(х3,у3) – це середина наближеної кривої, що з'єднує М(х1,у1) та К(х2,у2). Слід зазначити, що цей тип визначальної точки не обов'язково зберігатиметься в пам'яті, точки використовуються тільки для додаткової перевірки якості апроксимації контуру.

Внаслідок об'єднання вузлів, обраних відповідно до критеріїв, виходить масив вузлів, збережений у вигляді масиву, як показано в таблиці 3.1.

Таблиця 3.1 – Запис критичних точок контуру



Завдяки критичним точкам кількість операцій скорочується та продуктивність програми збільшується в 4 рази. Це зв’язано з тим, що апроксимація складає основу вибірки, а не набору значень контурної функції.

Визначення типу кривої порядку *п* для апроксимації (лінія, парабола, гіпербола) здійснюється за алгоритмом, описаним у процедурах TochkuAproxVerx (integer) та TochkuAproxNuz (integer) для верхньої та нижньої дуг відповідно:

1. Визначальні моменти вибираються ліворуч (рисунок 3.12).

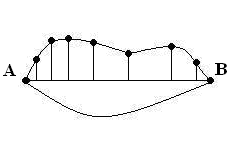


Рисунок 3.12 – Приклад дуги АВ з визначальними точками

2. Визначаються чотири сусідні визначальні точки, які визначаються за критерієм зміни знака прирісту функції. Якщо таких визначальних точок недостатньо, переходимо до кроку 4;

3. Пробуємо підійти до дуги по гіперболі. Якщо результат апроксимації від'ємний, то відхиляємо одну з визначальних точок (праворуч). Якщо ж результат апроксимації позитивний, переходимо до кроку 2, і відлік починається з кінця попередньої кривої;

4. Пробуємо підійти до дуги параболою. Якщо результат апроксимації є негативним, видаляємо одну з визначальних точок (праворуч);

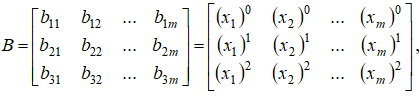
5. Перевіряється кількість визначальних точок, що визначаються критерієм зміни знака коефіцієнта приросту функції. Якщо таких визначальних точок більше двох, переходимо до кроку 4. Якщо результат апроксимації ствердний, переходимо до кроку 2 і відлік починається з кінця попередньої кривої;

6. Намагаємося підійти до дуги прямою. Якщо результат апроксимації від'ємний, то відкидаємо одну з визначальних точок (праворуч) та робимо іншу спробу апроксимувати прямою лінією. Якщо результат апроксимації ствердний, переходимо до кроку 2, і відлік починається з кінця попередньої кривої;

7. Здійснюється перевірка на наявність визначальних точок. Якщо всі визначальні точки не використовуються, переходимо до кроку 2. Якщо всі визначальні точки використовуються, завдання завершується.

Коефіцієнти а2, а1, а0 з процедур Aprox() та AproxLine() знаходяться за наступним алгоритмом (в даному випадку алгоритм ілюструє пошук коефіцієнтів параболи):

1. Матриця B формується:



де *т* – кількість визначальних точок при визначенні параболи;

*b*1*і* – дорівнює (*хі*)0=1, *і*=1,…*, т*;

*b*2*і* – дорівнює (*хі*)1, *і*=1,…, *т*;

*b*3*і* – дорівнює (*хі*)2, *і*=1,…, *т*.

2. Сформована матриця *С*:



де *т* – кількість точок при визначенні параболи;

*cі* – дорівнює *yі*, *і*=1..*m*;

3. Визначається добуток матриць B і C:

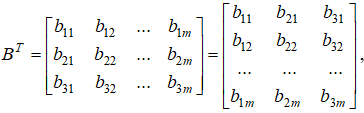


4. Щоб знайти коефіцієнти, помножимо обернену матрицю B-1 на матрицю D:



Щоб визначити зворотну матрицю B-1:

1) сформувати матрицю BT (транспонована для B):



де *т* – кількість визначальних точок при визначенні параболи;

*b*1*і* – дорівнює (*хі*)0=1, *і*=1,…*, т*;

*b*2*і* – дорівнює (*хі*)1, *і*=1,…, *т*;

*b*3*і* – дорівнює (*хі*)2, *і*=1,…, *т*.

2) обчислити матрицю H, яка дорівнює добутку матриць B та BТ:



3) обчислити мінори матриці H.

4) визначити алгебраїчні доповнення до матриці H:



де *Aij* – алгебраїчне доповнення елемента і-го рядочка, j-го стовпця;

i – кількість рядків в матриці Н, і=1…, *т*;

j – кількість стовпців в матриці Н, і=1…, 3;

Mij – мінор елемента і-го рядочка, j-го стовпця.

5) транспонувати матрицю алгебраїчних доповнень АТ;

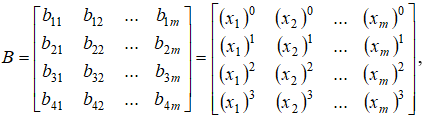
6) визначити визначники матриці АТ:



7) отримати обернену матрицю В-1, де

.

Знаходження коефіцієнтів гіперболи аналогічне, але матриця B матиме такий вигляд:



де *т* – кількість визначальних точок для визначення гіперболи;

*b*1*к* – дорівнює (*хі*)0=1, *к*=1…, *т*;

*b*2*к* – дорівнює (*хі*)1, *к*=1…, *т*;

*b*3*к* – дорівнює (*хі*)2, *к*=1…, *т*;

*b*4*к* – дорівнює (*хі*)3, *к*=1…, *т*.

Визначник обчислюється за правилом трикутника:



Лінії апроксимуються за такими формулами:



де М(х1,у1) – координати першої суміжної точки;

К(х2,у2) – координати другої суміжної точки;

k – коефіцієнт прямої, що з’єднує дві суміжні точки:

,

де М (х1, у1) – координати першої суміжної точки;

b - коефіцієнт прямої, що з'єднує дві суміжні точки;

k – коефіцієнт лінії, що з'єднує дві суміжні точки.

Для перевірки похибки наближення кривої використовуються такі критерії (процедури Aprox () та AproxLine ()):

а) перевіряється повна абсолютна похибка апроксимації:



де *Е* – сумарна похибка апроксимації;

*т* – кількість визначальних точок для визначення кривої апроксимації *n*-го порядку;

*f(xі)* – значення функції *f(х)* в *і*-ій точці;

*f(xі\*)* – значення функції *f(х)* в *і*-ій точці. розраховується з використанням кривих апроксимації *п*-го порядку.

Якщо E більша за вказану похибку, обчислена крива відповідності вважається неефективною.

б) перевірка правильності в *і*-ій точці:



де *еі* – похибка значення в *і*-ій точці;

*i* = 1…*,т*; *т* – кількість визначальних точок визначення кривої апроксимації *n*-го порядку;

*f(xі)* – значення функції *f(х)* в *і*-ій точці;

*f(xі\*)* – значення функції *f(х)* в *і*-ій точці, розраховане з використанням апроксимаційних кривих.

Якщо похибка апроксимації в деякій. точці *D*(*хп*,*уп*) більша за допустиму похибку, то спроба наближення визнається неефективною.

Точність апроксимації перевіряється для кожної апроксимаційної кривої порядку *п*. Якщо апроксимаційна крива виявилася неефективною, вводяться додаткові визначальні точки, і переобчислюється лише ця крива. Це дозволяє працювати з окремими елементами контуру. Такий підхід забезпечує мінімальну кількість обчислень, що дозволяє підвищити точність апроксимації з невеликою втратою продуктивності.

## 3.3 Тестування розробленої програмної системи

В якості досліджуваних зображень було обрано зображення. клітин людського організму (рисунок 3.13). Окремо, робилось виділення об’єктів на зображенні, проходження контуром, визначення метричних характеристик (площа, периметр, головна та мала осі, компактність, критичні точки тощо), кодування контуру об’єкта. за допомогою апроксимації. Отримані результати зберігаються у відповідному файлі.

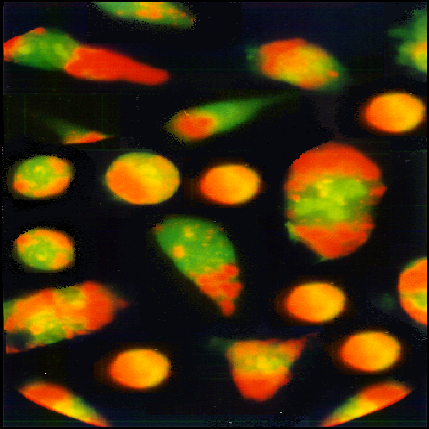


Рисунок 3.13 – Вхідне зображення

На першому етапі обробки зображення піддається пороговій сегментації з метою визначення фонових. та точок, які належать об’єктам. В результаті отримуються окремі області, які мають відмінні від фонових характеристики яскравості. Кожну область маркується унікальним ключем‑ідентифікатором. Після визначення інформативних областей проводиться проходження контурами даних областей, для визначення контурної функції. Отримавши контурну функцію програма знаходить дві найбільш. віддалені точки контуру для визначення найдовшої осі. Також проводиться перпендикуляр через центр відрізка, що сполучає дві найвіддаленіші точки контуру. Дана точка враховується. під час повороту об’єктів в процесі апроксимації. Результатом першого етапу роботи програми є виділені області з визначеними контурами, та визначеною великою віссю (рисунок 3.14).

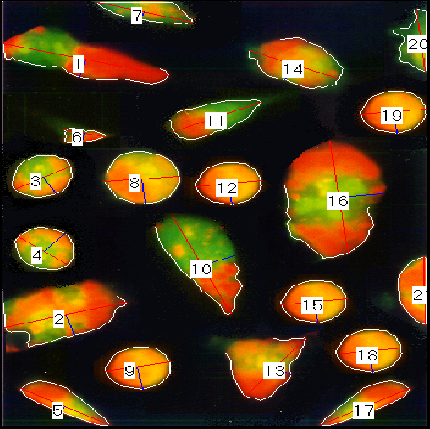


Рисунок 3.14 – Зображення з виділеними об’єктами

На другому етапі проводиться опис зображення. на основі метричних ознак, та проводиться апроксимація контурів об’єктів (рисунок 3.15).

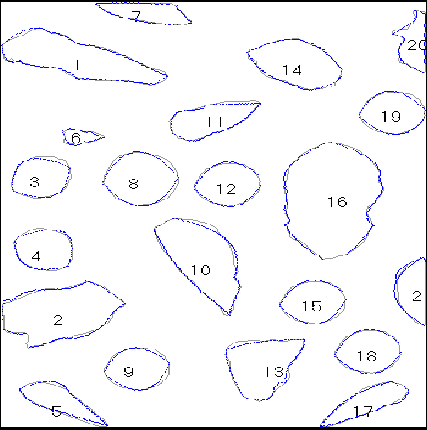


Рисунок 3.15 – Порівняння виводу контурів об’єктів заданих в табличній формі (сірий колір) та за допомогою апроксимуючих кривих (синій колір)

Апроксимація контурів проводиться з метою отримання аналітичних ознак об’єктів, зокрема кількості та типу кривих. за допомогою яких проводилась апроксимація контурної функції.

Результатом роботи є розрахований ряд геометричних параметрів. клітини для можливості подальшого аналізу. Приклад вихідної інформації наведено в таблиці 3.2.

Таблиця 3.2 – Результати роботи програми



Для визначення якості результатів роботи програми було проведено ряд експериментів. В них людина та машина. аналізували ті ж самі об’єкти. Результати порівняльного тесту наведені в таблиці 3.3.

Таблиця 3.3 – Результати наближення програмою та людиною

|  |  |  |  |
| --- | --- | --- | --- |
| Клітина | Контур, що не оброблений | Контур, що оброблений програмою | Контур, що оброблений людиною |
| 1 | 2 | 3 | 4 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Експериментально було показано ефективність розбиття програмою контуру на характерні ділянки з подальшою апроксимацією (таблиця 3.3). Як показано на прикладах кількість апроксимуючих кривих. та їхній тип ідентичні як при ручному так і при автоматичному визначенні визначальних точок та апроксимації.

Як видно з діаграми рисунку 3.16 при кодуванні контуру за допомогою кусково-поліноміальної апроксимації. отримано зменшення необхідної пам’яті для зберігання опису контурної функції.

Для проведення експериментів в програму був внесений модуль зупинки. Тривалість зупинки 0.01с. Зупинка здійснювалась після виводу кожного піксела на екран. Порівняно результати швидкодії роботи програми. Їх наведено в діаграмі рисунку 3.17.

Рисунок 3.16 – Порівняння кількості необхідної пам’яті, для збереження інформації. про контурну функцію

Рисунок 3.17 – Результати швидкості обробки контурної функції представленої у табличному вигляді та апроксимованої за допомогою кривих *п*-го порядку

Як видно з результатів проведених тестів при використанні апроксимованої контурної функції. час на її обробку зменшується на 10-15%. Це можна пояснити тим, що операція звертання до пам’яті займає відносно великий проміжок часу, що суттєво впливає на швидкодію програми в цілому. Отже використання опису контуру за допомогою масиву кривих *n*-го порядку дозволяє не тільки зменшити кількість використовуваної пам’яті, але і збільшити швидкодію.

Даний алгоритм може бути використаний для розпізнавання зображень (на основі аналізу зовнішнього контуру зображення) в системах розпізнавання зображень.

Для подальшого розвитку та збільшення ефективності процесу. проходження також способи. більш чіткого опрацювання виступів зображень (товщиною контурів необхідно розробити критерії коректної зупинки роботи алгоритмів, а декілька пікселів, що як показали результати експериментів, як правило є дефектами зображень).

## 3.4 Висновки до розділу 3

Для аналізу зображень було використано контурний аналіз. Перевагою даного типу обробки зображень є невеликі кількості необхідних операцій, що дозволяє суттєво знизити вимоги до апаратного забезпечення, при цьому якість отримуваної інформації залишається на високому рівні.

Основними вимогами до інформативних ознак є: висока ефективність, інваріантність в часі та просторових перетвореннях, простота визначення. Проте, не всі ознаки несуть однакове інформаційне навантаження, отже використання повного простору ознак не завжди є ефективним, розпізнавання об’єктів доцільніше робити досліджуючи множину максимально інформативних ознак;

Результати отримані за допомогою контурного аналізу можна в подальшому використовувати в поєднанні з іншими підходом до аналізу зображень, наприклад. використати семантичний аналіз. Поєднання декількох підходів аналізу зображень піднімає якість отримуваних результатів. на якісно новий рівень, що дозволяє врахувати не тільки геометричні чи текстурні особливості клітини, але і внутрішню структуру зображення.

ВИСНОВКИ

В результаті виконання кваліфікаційнної роботи одержані наступні результати:

1. Проведено аналіз відомих алгоритмів сегментації, а саме: порогового обмеження за яскравістю, центроїдного зв’язування, склеювання-розщеплювання, розфарбовування зображень,. проведено аналітичний огляд існуючих систем аналізу зображень. Це дало змогу показати актуальність задач сегментації і поставити завдання кваліфікаційного дослідження.

2. Досліджено алгоритми сегментації зображень – шляхом кластеризації колірного простору і за водорозподілом для цитологічних зображень пухлинних клітин.

3. Розроблено алгоритм сегментації зображень на основі виділення контурів, що дало змогу проаналізувати та провести моделювання біомедичних зображень.

4. Проведено моделювання розроблених алгоритмів. сегментації зображень і здійснено їх програмну реалізацію.

5. Використовуючи розроблений програмний засіб, проведено експериментальні дослідження біомедичних зображень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Дарченко А.О., Романов И.П., Креницкий А.П. и др. Автоматизация и компютеризация цитологических исследований в гематологии. Биомедицинские технологии и радиоэлектроника. 2003. № 2. С. 55–61.
2. Мельник А. Н. Цитоморфологическая диагностика опухолей : монографія. К. : Здоров’я, 1983. 240 с.
3. Фильченков А. А., Стойка Р. С. Апоптоз и рак. К. : Морион, 1999. 189 с.
4. Попова Г. М., Степанов В. Н. Анализ и обработка изображений медико-биологических объектов. Автоматика и телемеханика. 2004. № 1. С. 131–142.
5. Contour Tracing Algorithms. URL: <http://www.cs.mcgill.ca/~aghnei/alg.html>.
6. Theo Pavlidis, Algorithms for Graphics and Image Processing, 1982, chapter 7, section 5.
7. Мачуляк М. В., Галан В. Ю., Іпіроті В. О., Николин І. П. Системи підтримки прийняття рішень в медичній діагностиці. *Інтелектуальні комп’ютерні системи та мережі* : тези доп. V Наук.-практ. конф. молодих вчених і студентів (2 груд. 2021 р.). Тернопіль : ЗУНУ, 2021. С. 12.
8. Галан В. Ю., Мачуляк М. В., Іпіроті В. О., Николин І. П. Моделювання знань в системах медичної діагностики. *Інтелектуальні комп’ютерні системи та мережі* : тези доп. V Наук.-практ. конф. молодих вчених і студентів (2 груд. 2021 р.). Тернопіль : ЗУНУ, 2021. С. 13.
9. Березький О. М., Дубчак Л. О., Мельник Г. М. Методичні рекомендації до виконання кваліфікаційної роботи з освітнього ступеня “Магістр”. Спеціальність: 123 - Комп’ютерна інженерія. Магістерська програма – «Комп’ютерна інженерія». Тернопіль : ЗУНУ, 2021. 32 с.
10. Денисов Д. А., Низовкин В. А. Сегментация изображений на ЭВМ. Зарубежная радиоэлектроника. 1985. № 10. С. 5–30.
11. Дуда Р., Харт П. Распознавание образов и анализ сцен. М. : Мир, 1976. 512 с.
12. Зарубежная радиоэлектроника. 1987. №10. 95 с.
13. Путятин Е.П. Теоретические предпосылки нормализации изображений. Проблемы бионики. Харьков : Вища школа. 1973. Вып. 10. С.82-89.
14. Путятин Е.П., Житомирский М.Я., Лагиев А.У. Распознавание связных компонент несвязной области. АСУ и приборы автоматики. Харьков: Вища школа. 1987. Вып. 81. С. 3–10.
15. Путятин Е.П., Аверин С.И. Обработка изображений в робототехнике. М. : Машиностроение, 1990. 320 с.
16. Волков Е.А. Численные методы. М. : Наука, 1982. 254 с.
17. Akond. URL: http://www.akond.net/index.php?issue\_id=4.
18. SIAMS. URL: <https://siams.com/>.
19. NEXSYS ImageExpert™ Pro 3. URL: <http://www.s-i.ru/nexsys-imageexpert-sequencer/>.
20. Новые Экспертные Системы (НЭКСИС): программы для анализа изображений Image Experttm. URL: http://www.modificator.ru/ad/nexsys.html.
21. 17imageWarp. URL: <http://www.imagewarp.com/>.
22. Matlab. URL: https://www.mathworks.com/products/matlab.html.
23. Павлидис Т. Алгоритмы машинной графики и обработки изображений. М. Радио и связь, 1986.
24. Гонсалес Р., Вудс Р. Цифровая обработка изображений в среде MATLAB. М. : Техносфера, 2006. 618 с.
25. Jain A. K., Murty M. N., Flynn P. J. Data clustering: a review. *ACM Computing Surveys*. 1999. Vol. 31, № 3.С. 264–323.
26. Гонсалес Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс – М.: Техносфера. – 2005.–1072 с.
27. Форсайт, Дж. Понс Компьютерное зрение. Современный подход// Вильямс – 2004.
28. Березький О., Батько Ю. Аналіз алгоритмів виділення контурів зображень. Вісник Національного університету «Львівська політехніка». Комп’ютерні науки та інформаційні технології. - 2006. - № 565. - С. 212 - 216.
29. Перейма І. Алгоритми сегментації зображень. Матеріали Десятої наукової конференції Тернопільського державного технічного університету ім. Івана Пулюя. – Тернопіль: ТДТУ, 2007. – С.138.
30. Березький О. М. Методи сегментації біомедичних зображень / О. М. Березький, Ю.М. Батько, Г.М.Мельник // Вісник Хмельницького національного університету. Технічні науки. – 2010. – №1. – С.189- 197.
31. Березький О. М., Метод сегментації зображень на основі попередніх розміток зображення / О. М. Березький, Г.М.Мельник, Ю.М. Батько // Матеріали 4-ї Міжнародної науково-технічної конференції «Комп’ютерні науки та інформаційні технології 2009», 15-17 жовтня, 2009, м. Львів. – Львів: ПП «Вежа і Ко», 2009. – С.48-52.
32. Березький О. М. Метод вибору алгоритмів автоматичної сегментації біомедичних зображень / О. М. Березький, Ю. М. Батько // Системи обробки інформації. – 2013. – Випуск 2 (109). – С. 246-251.
33. Березький О. Методи кількісної оцінки якості сегментації зображень / Олег Березький // Матеріали дванадцятої всеукраїнської міжнародної конференції «Оброблення сигналів і зображень та розпізнавання образів» (УкрОБРАЗ’2014), Київ, 3-7 листопада 2014 р. − К., 2014. − С. 51−54.
34. Березский О. Н. Количественная оценка качества сегментации изображений на основе метрик / О. Н. Березский, Е. Н. Березская // Управляющие системы и машины. – 2015. – №6. – С.59-65.
35. Березький О.М. Аналіз алгоритмів співставлення областей зображень для кількісної оцінки результатів сегментації / О.М. Березький, Г.М. Мельник, Ю.М. Батько, О.Й. Піцун // Матеріали Міжнародної наукової конференції «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту» (ISDMCI’2016), м. Залізний Порт, 24–28 травня 2016 р. – Херсон: ПП Вишемирський В.С., 2016. – С. 252-253.
36. Березький О.М. Адаптивний метод сегментації зображень на основі метрик / О.М. Березький, О.Й. Піцун // Науковий вісник НЛТУ України: збірник науково-технічних праць. Львів: РВВ НЛТУ України. – 2018. – №. 28(3). – С.122-126.
37. Краснов М. В. OpenGL. Графика в проектах Delphi. — СПб.: БХВ-Петербург, 2002. - 352 с: ил.
38. Трухильо С. Графика для Windows: библиотека программиста-СПб.: Издательство 'Питер' 1998 – 498c.
39. Delphi: для пользователя: пер с нем. - Р. Баас., М. Фервай, Х. Гюнтер, - К.: Издательская группа BHV, 2000 – 496 с.
40. Бартлетт Н. Программирование на Delphi Путеводитель. The Coriolis Group,Inc., 1996, Издательство НИПФ "ДиаСофт Лтд.", 2016. 116 с.
41. Березин Б.И. Начальный курс Delphi. М.: ДИАЛОГ-МИФИ, 2016. 331 с.
42. Бондарев В.М. Основы программирования. Харьков: Фолио, Ростов н/Д: Феникс, 2017. 446 с.
43. Кен А. Язык программирования Delphi. Addison-Wesley Longman,U.S.A.,1996, Издательство "Питер-Пресс", 2017. 378 с.
44. Системное программное обеспечение / А.В. Гордеев, А.Ю. Молчанов - СПб.: “Питер” – 2003 – 736 с.
45. Леонтев В.П. Новейшая энциклопедия персонального компьютера 2003. – 5-е изд., перераб . и доп., - М.: ОЛМА-ПРЕСС, 2003 – 957с.
46. Жидков, О. М. Сетевые операционные системы / О. М. Жидков. – Москва : Лаборатория книги, 2011. – 114 с. – Режим доступа: URL: <https://biblioclub.ru/index.php?page=book&id=142238>.
47. Березький О. М. Аналіз і синтез зображень пухлинних клітин / О. М. Березький // Інформаційні технології і системи. – 2005. – Т. 8, № 1. – С. 76–85.
48. Березький О. М. Контурний аналіз зображень у локальних координатах / О. М. Березький // Моделювання та інформаційні технології: зб. наук. праць ІПМЕ НАН України. 2008. Вип. 46. С. 101–108.
49. Березький О. М. Методи та алгоритми перетворення контурів зображень в афінному просторі // Вісник Національного університету «Львівська політехніка». Комп’ютерні науки та інформаційні технології. − 2009. − № 638. − С. 185–189.
50. Березький О. М. Аналіз контурів зображень у глобальних координатах / О. М. Березький, К. М. Березька // Оптико–електронні інформаційно– енергетичні технології. – 2006. – № 2 (12). – С. 83–91.
51. Березький О. М. Алгоритм проходження контуром об’єкта з використанням зворотного ходу / О. М. Березький, Ю. М. Батько // Штучний інтелект. – 2009. – № 3. – С. 116–122.
52. Методи, алгоритми і програмні засоби опрацювання біомедичних зображень / Березький О. М., Батько Ю.М., Березька К.М. і ін. Тернопіль: Економічна думка, ТНЕУ, 2017. 330 с.