АНОТАЦІЯ

на наукову роботу під шифром _____РМ2019-1_____

Актуальність роботи полягає в побудові автоматизованої системи сегментації цифрових зображень. Створена система може бути використана для попередньої обробки даних, що потрапляють до модулів комп'ютерного бачення високого рівня, з метою зменшення їх об'єму.

Метою роботи є розробка автоматизованої системи для сегментації цифрових зображень.

Завдання:

1. вивчити метод ефективної сегментації на ґратковому графі;
2. розробити алгоритм сегментації цифрового зображення методом ефективної сегментації на ґратковому графі;
3. розробити обчислювальну схему для отримання сегментації цифрових зображень;
4. створити автоматизовану систему на основі розробленого алгоритму;
5. оцінити якість отриманої сегментації;
6. провести тестування автоматизованої системи на реальних даних.

Методи дослідження: метод аналізу та синтезу, методи порівняння та метод систематизації.

Структура та обсяг роботи: пояснювальна записка складається із вступу, трьох розділів, висновків та списку використаної літератури із 21 джерела. Загальний обсяг дипломної роботи складає: 30 сторінок, ілюстрацій – 25, таблиць – 4.

Ключові слова: сегментація, цифрове зображення, аерофотозйомка.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАУКОВА РОБОТА

для участі у Всеукраїнському конкурсі студентських наукових робіт з галузей знань і спеціальностей

Тема: Автоматизована система сегментації цифрових зображень на основі дискретних структур

Шифр роботи «РМ2019-1»

2019-2020 н.р.

ЗМІСТ

ВСТУП

Актуальність

Задачі сегментації і групування зображень залишаються складними для комп'ютерного зору. З часів гештальтпсихології було відомо, що сприйняття форми (перцептивне сприйняття) відіграє важливу роль у зоровому сприйнятті людини. Широкий спектр задач обчислювального бачення може використовувати сегментовані зображення, якщо такі надійно і ефективно обчислюються. Наприклад, задачі бачення середнього рівня складності, такі як стереобачення та оцінка руху потребують підтримки відповідних операцій. Просторово неоднорідні області можуть бути ідентифіковані за допомогою методів сегментації. Задачі бачення вищого рівня, такі як розпізнавання та індексація зображень, також можуть використовувати результати сегментації для розпізнавання по частинах.

В сучасному світі задача сегментації цифрових зображень використовується у:

- медичних дослідженнях, наприклад, для обробки зображень з сітківкою ока [1,2];
- розпізнавання жестів рук [3];
- трекінгу об’єктів на відеозйомці [4];
- інтерактивного визначення переднього плану на цифрових зображеннях [5];
- відокремленні доріг на супутникових знімках [6].

Хоча в задачах сегментації цифрових зображень все більше домінують згорткові нейронні мережі, методи, що засновані на використанні дискретних структур, також можуть давати гарні результати, при чому за менший час.

Метою роботи є розробка автоматизованої системи для сегментації цифрових зображень.

Об’єктом дослідження є процес сегментації цифрових зображень.

Предмет дослідження — метод сегментації за допомогою ґраткового графу.

Постановка задачі

Нехай задано цифрове зображення $I = \{ p_{ij} \}$, де кожен піксель p_{ij} характеризується координатами (i, j) та трьома кольоровими складовими.

Необхідно визначити розбиття I на класи еквівалентності за відношенням піксель $p_{i_1 j_1}$ «схожий на» піксель $p_{i_2 j_2}$ (рефлексивне і симетричне).

Іншими словами, утворимо скінченний набір множин $\{S_k\}_{k=1}^{N}$, які не перетинаються ($S_k \bigcap S_l = \varnothing$ при $k \neq l$), елементами яких є p_{ij}, такі, що для довільних пікселів $p_{i_1 j_1}$ «схожий на» $p_{i_2 j_2}$ тоді і лише тоді, коли існує єдиний індекс n такий, що $p_{i_1 j_1} \in S_n$ і $p_{i_2 j_2} \in S_n$.

Завдання:

1. вивчити метод ефективної сегментації на ґратковому графі;
2. розробити алгоритм сегментації цифрового зображення методом ефективної сегментації на ґратковому графі;
3. розробити обчислювальну схему для отримання сегментації цифрових зображень;
4. створити автоматизовану систему на основі розробленого алгоритму;
5. оцінити якість отриманої сегментації;
6. провести тестування автоматизованої системи на реальних даних.

РОЗДІЛ 1. ОСНОВНІ ПОНЯТТЯ ТА МЕТОДИ СЕГМЕНТАЦІЇ ЦИФРОВИХ ЗОБРАЖЕНЬ

В комп'ютерному зорі сегментація зображення - це процес поділу цифрового зображення на кілька сегментів (наборів пікселів, також відомих як супер-пікселі). Мета сегментації - спростити і / або змінити подання зображення на щось більш значуще і більш легке для аналізу [7].

Результатом сегментації зображення є набір сегментів, які в сукупності покривають все зображення, або набір контурів, витягнутих з зображення. Кожен з пікселів в області аналогічний за деякими характеристиками або обчислюваними властивостями, таких як колір, інтенсивність або текстура. Суміжні регіони істотно відрізняються один від одного по відношенню до однієї і тієї ж характеристики.

1.1. Огляд методів сегментації

Існують різні підходи для отримання сегментації цифрового зображення. Далі розглядаються основні з них з коротким викладенням ідеології роботи.

Побудова графу на основі зображення та подальша сегментація на його основі, що розглядаються в пункті 1.2., відповідають ідеології сімейства сегментацій на основі графів, тому їх загальна концепція не розглядається.

1.1.1. Методи, засновані на кластеризації

K – середніх [8] — це ітераційний метод, який використовується для того, щоб розділити зображення на K кластерів. Базовий алгоритм наведений нижче:

1. Вибрати K центрів кластерів, випадково або на основі деякої евристики;
2. Помістити кожен піксель зображення в кластер, центр якого найближче до цього пікселя;
3. Знову визначити центри кластерів, усереднюючи всі пікселі в кластері;
4. Повторювати кроки 2 і 3 до збіжності (наприклад, коли пікселі будуть залишатися в тому ж кластері).

Цей алгоритм гарантовано сходиться, але він може не привести до оптимального рішення. Якість рішення залежить від початкової множини кластерів і значення K.

1.1.2. Методи з використанням гістограми

Методи з використанням гістограми дуже ефективні порівняно з іншими методами сегментації оскільки вони вимагають тільки один прохід по пікселях. У цьому методі гістограма обчислюється за всіма пікселям зображення і її мінімуми і максимуми використовуються, щоб знайти кластери на зображенні [9]. Колір або яскравість можуть бути використані при порівнянні.

Покращення цього методу — рекурсивно застосовувати його до кластерів на зображенні для того, щоб поділити їх на дрібніші кластери. Процес повторюється з усе меншими і меншими кластерами до тих пір, коли перестануть з'являтися нові кластери [9, 10].

Один недолік цього методу — те, що йому може бути важко знайти значні мінімуми і максимуми на зображенні. У цьому методі класифікації зображень схожі метрика відстаней і зіставлення інтегрованих регіонів.

1.2. Метод ефективної сегментації на ґратковому графі

Використаємо заснований на графі підхід до сегментації [11, 12]. Нехай $G = (V, E)$ неорієнтований граф з вершинами $v_i \in V$ (рис. 1.1) - набір елементів для сегментації, і ребра $(v_i, v_j) \in E$ відповідають парам сусідніх вершин. Кожне ребро має відповідну вагу $w(v_i, v_j)$, яка є невід'ємної мірою відмінності між сусідніми елементами v_i і v_j. У разі сегментації зображення *I* елементи з *V* є пікселями, а вага ребра є деякою мірою відмінності між двома пікселями, пов'язаними цим ребром (наприклад, різниця в інтенсивності, кольорі, місцезнаходженні або якому-небудь іншому локальному атрибуті).

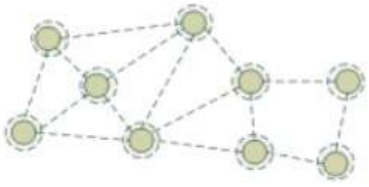

Рис. 1.1. Граф для сегментації

У графовому підході сегментація S являє собою розбиття V на компоненти таким чином, що кожен компонент (або область) $C \in S$ відповідає з'єднаним

компонентом у графі $G' = (V, E')$, де $E' \in E$. Іншими словами, будь-яка сегментація є підмножиною ребер в E (рис. 1.2). Існують різні способи вимірювання якості, але вцілому ми хочемо, щоб елементи в компоненті були схожі, а елементи в різних компонентах - несхожі. Це означає, що ребра між двома вершинами в одному компоненті повинні мати відносно низькі ваги, а ребра між вершинами в різних компонентах повинні мати більш високі ваги.

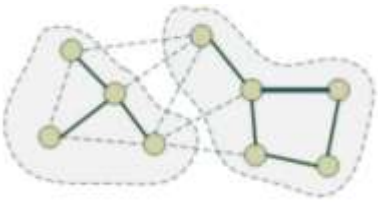

Рис. 1.2. Результат сегментації на графі

1.2.1. Предикат для попарного порівняння областей

У цьому розділі ми визначимо предикат D для оцінки необхідності розділення двох компонентів в сегментації (двох областей зображення I). Цей предикат заснований на вимірюванні відмінностей між елементами уздовж кордону двох компонентів, щодо міри відмінностей між сусідніми елементами всередині кожного з двох компонентів

Визначимо *внутрішню різницю* компонента $C \subseteq V$ як найбільшу вагу в $MST(C; E)$ [13](рис. 1.3), тобто

$$Int(C) = \max_{e \in MST(C, E)} (w(e)).$$

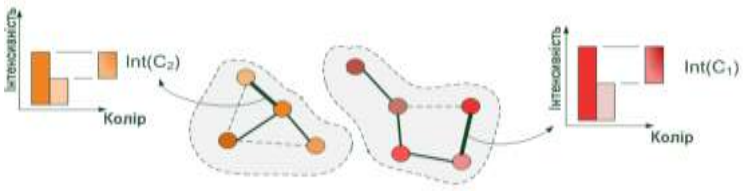

Рис. 1.3. Внутрішня різниця компонента

Ідея, що лежить в основі цієї міри полягає в тому, що компонент C залишається зв'язаним доти, доки ребра, що розглядаються, мають вагу меншу за $Int(C)$.

Визначимо різницю між двома компонентами $C_1,\ C_2 \subseteq V$ як мінімальну вагу з ребер, що з'єднує ці компоненти [13](рис. 1.4). Тобто:

$$Dif(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2, (v_i, v_j) \in E} (w(v_i, v_j)).$$

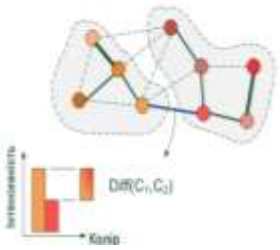

Рис. 1.4. Різниця між двома компонентами

Якщо немає ребра, що з'єднує C_1 і C_2, то нехай $Dif(C_1, C_2) = \infty$. Ця міра різниці в принципі може бути проблематичною, оскільки вона відображає тільки найменшу вагу ребра між двома компонентами. Але ця міра працює досить добре, незважаючи на це очевидне обмеження.

Предикат порівняння областей обчислює чи необхідно розділяти пару компонентів, перевіряючи, чи є різниця між компонентами $Dif(C_1, C_2)$ великою у порівнянні до внутрішньої різниці принаймні в одному з компонентів, $Int(C_1)$ і $Int(C_2)$. Порогова функція використовується для керування ступенем, в якій різниця між компонентами повинна бути більше мінімальної внутрішньої різниці. Визначимо предикат попарного порівняння[13](рис. 1.5) як,

$$D(C_1, C_2) = \begin{cases} \textit{істинний} & \textit{якщо } Dif(C_1, C_2) > MInt(C_1, C_2) \\ \textit{хибний} & \textit{в іншому випадку} \end{cases}$$

де мінімальна внутрішня різниця, $MInt$, визначається як,

$$MInt(C_1, C_2) = \min(Int(C_1) + \tau(C_1), Int(C_2) + \tau(C_2)).$$

Порогова функція τ керує ступенем, в якому відмінність між двома компонентами повинна бути більше за їх внутрішні відмінності, щоб між ними була межа (D повертав істинне значення). Для невеликих компонентів $Int(C)$ не є хорошою оцінкою локальних характеристик даних. В крайньому випадку, коли

$|C| = 1$, $Int(C) = 0$. Тому ми використовуємо порогову функцію, засновану на розмірі компонента $\tau(C) = k / |C|$.

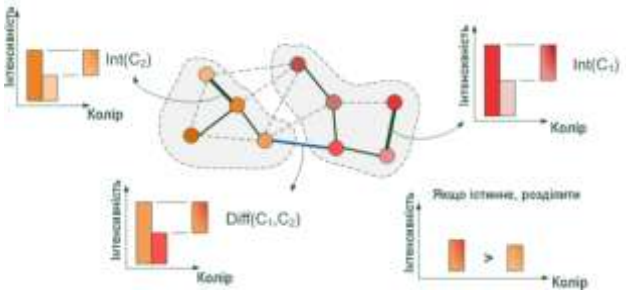

Рис. 1.5. Графічний зміст предиката порівняння областей

Де $|C|$ позначає розмір C, а k - деякий постійний параметр. Тобто для невеликих компонентів нам потрібні більш сильні докази кордону. На практиці k встановлює шкалу спостереження, в якій більше k викликає перевагу більших компонентів.

Будь-яка невід'ємна функція одного компонента може бути використана для τ без зміни алгоритму. Наприклад, метод сегментації може віддати перевагу компонентам певних геометричних форм, визначивши таку функцію τ, яка повертає великі значення для компонентів, які не відповідають деякій бажаній формі, і маленькі значення для тих, які відповідають. Це змусить алгоритм сегментації агресивно об'єднувати компоненти, які не мають бажаної форми.

1.2.2. Алгоритм і його властивості

У цьому розділі ми опишемо і проаналізуємо алгоритм отримання сегментації з використанням критерію прийняття рішення D, введеного вище.

Алгоритм 1 *Алгоритм сегментації [13].*

На вхід подається граф $G = (V, E)$, з n вершинами і m ребрами. Результатом є сегментація V на компоненти $S = (C_1, \ldots, C_r)$.

0. Сортуємо E $\pi = (o_1, \ldots, o_m)$ у порядку зростання за вагами;

1. Починаємо з сегментації S^0, де кожна вершина v_i знаходиться у своєму компоненті;

2. Повторюємо крок 3 для $q = 1, \ldots, m$;

3. Будуємо S^q за заданою S^{q-1}. Нехай v_i і v_j позначають вершини, з'єднані ребром q - й послідовності, тобто $o_q = (v_i; v_j)$. Якщо v_i і v_j знаходяться в системі неперетинних множин S^{q-1} і $w(o_q)$ мала, в порівнянні з внутрішньою різницею обох цих компонентів, то необхідно об'єднати ці два компоненти. В іншому випадку – нічого не робити. Більш формально, нехай C_i^{q-1} є компонентом з S^{q-1}, що містить v_i і C_j^{q-1} - компонент, що містить v_j. Якщо $C_i^{q-1} \neq C_j^{q-1}$ і $w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$, тоді при утворенні сегментації S^q компоненти об'єднуються. В іншому випадку компоненти C_i^{q-1} і C_j^{q-1} переходять до сегментації S^q без змін;

4. Повернути $S = S^m$.

Сегментація S, отримана алгоритмом 1, підпорядковується глобальним властивостями не бути ні надто доброю, ні надто грубою при використанні предиката порівняння областей D, визначеного в пункті 1.2.1 [13].

1.2.3. Оцінка якості кластеризації

Кількісним критерієм, що дозволяє віддати перевагу одному із розбиттів вихідної вібірки, є функціонал якості.

Нехай у результаті застосування методу кластеризації одержане розбиття

$$S = \{S_1, S_2, ..., S_K\}$$

вихідної вибірки на K кластерів

$$S_j = \left\{X_l^{(j)}; l = \overline{1, N_j}\right\} = \left\{\left(x_{1,l}^{(j)}, ..., x_{n,l}^{(j)}\right); l = \overline{1, N_j}\right\},\ j = \overline{1, K},$$

де

N_j — кількість об'єктів у кластері S_j;

$X_l^{(j)} = \left(x_{1,l}^{(j)}, x_{2,l}^{(j)}, ..., x_{n,l}^{(j)}\right)$ — l-й об'єкт кластера S_j, $l = \overline{1, N_j}$;

$x_{k,l}^{(j)}$ — значення k-ї ознаки об'єкта $X_l^{(j)}$, $k = \overline{1, n}$.

Для оцінки якості розбиття S відомо близько 50 функціоналів якості. Найбільш поширені такі [14]:

1. **Сума внутрішньокластерних дисперсій:**

$$Q_1(S) = \sum_{j=1}^{K} \sum_{l=1}^{N_j} d^2\left(X_l^{(j)}, \overline{X}^{(j)}\right),$$

де

$\overline{X}^{(j)}$ — центр кластера S_j.

2. **Сума попарних внутрішньокластерних відстаней**, що мінімізується:

$$Q_2(S) = \sum_{j=1}^{K} \sum_{l=1}^{N_j-1} \sum_{h=l+1}^{N_j} d\left(X_l^{(j)}, \overline{X}_h^{(j)}\right).$$

Мінімізація функціоналу забезпечує максимізацію суми міжкластерних відстаней.

3. **Загальна внутрішньокластерна дисперсія** має бути мінімальна:

$$Q_3(S) = \det\left(\sum_{j=1}^{K} N_j V_j\right),$$

де $V_j = \left\| v_{k,p}^{(j)},\ k,\ p = \overline{1,n} \right\|$ — матриця коваріацій кластера S_j.

РОЗДІЛ 2. ОБРОБКА ЗОБРАЖЕНЬ ТА КОЛЬОРОВІ МОДЕЛІ

Перед сегментуванням зображення часто застосовують первинну обробку зображення. У розділі описано основні аспекти обробки. Розмивання Гауса, що описано нижче є невід'ємною частиною ефективного сегментування зображення.

2.1. Обробка зображень

У цьому пункті розглядається теоретичне підґрунтя обробки цифрових зображень: описується згортка, як математична операція, та її використання на зображеннях. Також розглянуто декілька варіантів роботи на краях зображення, зокрема розширення, обгортання, віддзеркалення по краю та зріз ядра. В кінці описано розмивання зображення за допомогою фільтра на основі функції Гауса.

2.1.1. Згортка зображень та обробка країв

Згортка зображень

Згортка - це процес додавання кожного елемента зображення до його сусідів, зважених ядром. Важливо зауважити, що виконувана матрична операція - згортка - це не звичайне множення, хоча й позначається [15]. Загальним виразом згортки є:

$$p_{ij}^{*} = h * p_{ij} = \sum_{k=-a}^{a} \sum_{l=-b}^{b} h_{kl} * p_{i-k,y-l},$$

де p_{ij}^{*} — інтенсивність пікселя на відфільтрованому зображення, p_{ij} — інтенсивність пікселя на вхідному зображенні, h —ядро згортки. Розглядається кожен елемент ядра згортки $-a \le k \le a$ та $-b \le l \le b$ [16].

Приклад згортки для пікселя $p_{2,2}$ (без нормування):

$$\left(\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} * \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \right) = (a \cdot 1) + (b \cdot 2) + (c \cdot 3) + (d \cdot 4) + (e \cdot 5) +$$

$$+(f \cdot 6) + (g \cdot 7) + (h \cdot 8) + (i \cdot 9).$$

Обробка країв зображення

При використанні операції згортки на краях зображення, стають необхідними значення пікселів за його межами. Звідси з’являються різні варіанти обробки країв [17]:

- **розширення** (рис. 2.1)

  Найближчі до країв пікселі продовжуються настільки далеко, наскільки це необхідно для згортки. Кутові пікселі утворюють квадрати, а інші пікселі продовжуються у лінію.

Рис. 2.1. Тип обробки країв «розширення»

- **обгортання** (рис. 2.2)

  Зображення загорнуто саме в себе і значення беруться з протилежного краю або кута.

Рис. 2.2. Тип обробки країв «обгортання»

- **віддзеркалення по краю** (рис. 2. 3)

  При зверненні до пікселів за межами оригінального зображення беруться їх дзеркальні відображення відносно границь або кутів.

Рис. 2.3. Тип обробки країв «віддзеркалення по краю»

- **зріз ядра**

2.1.2. Розмивання Гауса

Розмивання Гауса — це метод фільтрації зображення за допомогою функції Гауса, який призводить до розмивання зображення. Даний ефект широко використовується в графічних програмах, як правило, для зменшення зашумленості зображення та зниження деталізації. Візуальний ефект цієї фільтрації розмивання аналогічний погляду на зображення крізь напівпрозорий екран.

Принцип роботи фільтра

Розмивання Гауса це тип фільтру розмивання зображення, що використовує функцію Гауса для розрахунку трансформації кожного пікселя у зображенні. Рівняння функції Гауса в одному вимірі:

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

Для двовимірного випадку:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

де x — це відстань від початку координат в осі абсцис, y — це відстань від початку координат у осі ординат, а σ — стандартне відхилення розподілу Гауса [18].

Коли метод застосовується у двох вимірах, отримується поверхня, контури якої є концентричними колами розподілу Гауса з центральної точки (рис. 2.4). Значення з цього розподілу використовуються для створення матриці згортки(початок координат у центрі матриці). Для кожного нового значення пікселя визначається середнє зважене в околі пікселя. Значення поточного оригінального пікселя має більшу вагу (найвище значення розподілу Гауса), а сусідні пікселі отримують все меншу вагу в залежності від того наскільки далеко вони знаходяться від поточного оригінального пікселя. Це надає ефект розмитості, яка зберігає кордони та краї краще, ніж інші, аналогічні фільтри розмиття.

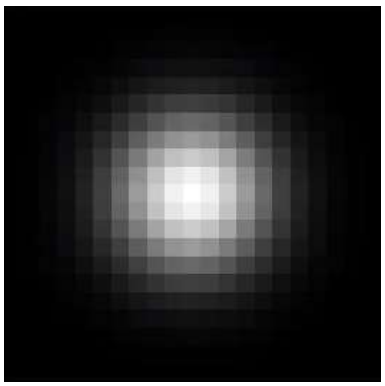

Рис. 2.4. Концентричні кола розподілу Гауса

Як правило, програми обробки зображень розрахувують матрицю з розмірами $[6\sigma+1]\times[6\sigma+1]$, щоб забезпечити результат, який є досить близьким до отриманого від усього розподілу Гауса.

2.2. Кольорові моделі

Існує чимало кольорових моделей та їх варіацій. Усі вони мають переваги та недоліки в різних сферах застосування. В пункті розглядаються наступні:

1. **RGB,** як найпопулярніша кольорова модель для відтворення зображень на екранах комп'ютерів;
2. **XYZ,** як проміжна модель у перетворенні RGB – LAB;
3. **LAB,** як апаратно-незалежна розширена кольорова модель, в основі якої лежить сприйняття відмінності кольорів людиною.

Основні кольорові моделі та шляхи, що використовуються на практиці. за якими з однієї моделі колір може бути перетворений в іншу, наведені на наступній схемі (рис. 2.5) [19, 20, 21].

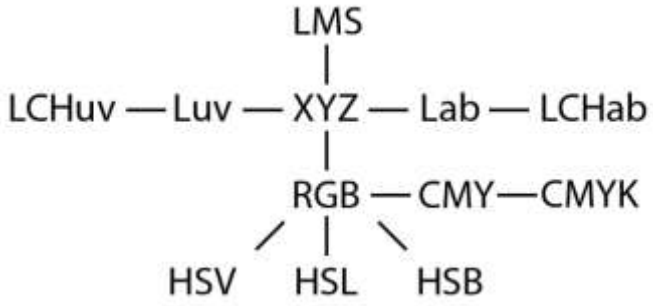

Рис. 2.5. Схема перетворення кольорових моделей

RGB — адитивна колірна модель, що описує спосіб синтезу кольору, за якою червоне, зелене та синє світло накладаються разом, змішуючись у різноманітні

кольори. Широко застосовується в техніці, що відтворює зображення за допомогою випромінення світла [19].

З метою уніфікації була розроблена міжнародна стандартна колірна модель. В результаті серії експериментів міжнародна комісія з освітлення (CIE) визначила криві складання основних (червоного, зеленого і синього) кольорів. У цій системі кожному кольору, що може побачити людина, відповідає певне співвідношення основних кольорів. Використовують уявні основні кольори: X (уявний червоний), Y (уявний зелений), Z (уявний синій)[20].

Lab — система задання кольорів, що використовує як параметри світлосилу, відношення зеленого до червоного та відношення синього до жовтого. Ці три параметри утворюють тривимірний простір, точки якого відповідають певним кольорам.

Основною метою при розробці CIE Lab було усунення нелінійності системи CIE XYZ з точки зору людського сприйняття. Під абревіатурою Lab зазвичай розуміється колірний простір CIE L*a*b*, яке на даний момент є міжнародним стандартом [21].

РОЗДІЛ 3. АВТОМАТИЗОВАНА СИСТЕМА ДЛЯ СЕГМЕНТАЦІЇ ЗОБРАЖЕНЬ

У розділі описано автоматизовану систему сегментації цифрових зображень та технічні засоби і структури даних, на яких вона реалізована.

3.1. Технічні засоби та структури даних

У пункті 3.1.1 розглядається структура даних «Система неперетинних множин», яка використовується у реалізованому програмному забезпеченні для представлення ґраткового графу, на основі якого проводиться сегментація цифрового зображення.

3.1.1. Система неперетинних множин

Система неперетинних множин— структура даних, яка дозволяє відстежувати множину елементів, розбиту на неперетинні підмножини (рис. 3.1). При цьому кожній підмножині призначається її представник — елемент цієї підмножини.

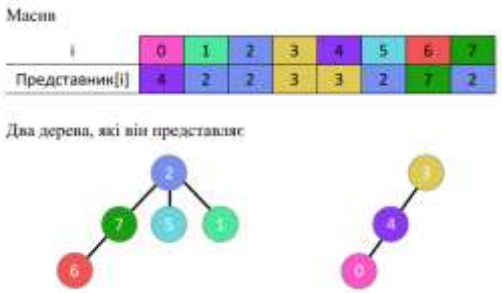

Рис. 3.1. Система неперетинних множин, яка реалізована на масивах

Спочатку є кілька елементів, кожен з яких знаходиться в окремій (своїй власній) множині. За одну операцію можна об'єднати дві будь-які множини, а також можна дізнатися, в якій множині зараз знаходиться зазначений елемент. У класичному варіанті вводиться ще одна операція — створення нового елемента, який поміщається в окрему множину.

Таким чином, базовий інтерфейс даної структури даних складається всього з трьох операцій:

- *MakeSet* — додання нового елемента; розміщення його в нову множину, що складається з одного нього;
- *Union* — об'єднання двох зазначених множин;
- *Find* — повернення значення, в якій множині знаходиться зазначений елемент.

Побудова ефективної структури даних

Множина елементів буде зберігатися у вигляді дерев: одне дерево відповідає одній множині. Корінь дерева — це представник (лідер) множини.

При реалізації це означає, що ми заводимо масив, в якому для кожного елемента зберігаємо посилання на його предка у дереві. Для коренів дерев будемо вважати, що їх предок — вони самі (тобто посилання зациклюється в цьому місці).

Наївна реалізація

Наведеної вище інформації вже достатньо, щоб написати першу реалізацію системи неперетинних множин. Вона буде досить неефективною, але потім буде покращена, за рахунок двох евристик.

Отже, вся інформація про множини елементів зберігається за допомогою масиву **parrent**.

Опис операцій

- Щоб створити новий елемент (операція *MakeSet(v)*), ми просто створюємо дерево з коренем у вершині, зазначаючи, що його предок — це він сам;
- Щоб об'єднати дві множини (операція *Union(a, b)*), ми спочатку знайдемо лідерів першої і другої множини. Якщо лідери збіглися, то нічого не робимо — це означає, що множини і так вже були об'єднані. В іншому випадку можна просто вказати, що предок першої вершини дорівнює другій (або навпаки) — тим самим приєднавши одне дерево до іншого;
- Реалізація операції пошуку лідера (*Find(v)*) проста: ми піднімаємося по предкам від вершини, поки не дійдемо до кореня. Цю операцію зручніше реалізувати рекурсивно.

Утім, така реалізація системи неперетинних множин дуже неефективна. Легко побудувати приклад, коли після кількох об'єднань множин вийде ситуація, що множина — це дерево, звиродніле в довгий ланцюжок.

Для уникнення подібних ситуацій вводяться евристики: «Евристика стиснення шляху» (рис. 3.2.) та «Евристика об'єднання за рангом» (рис. 3.3.).

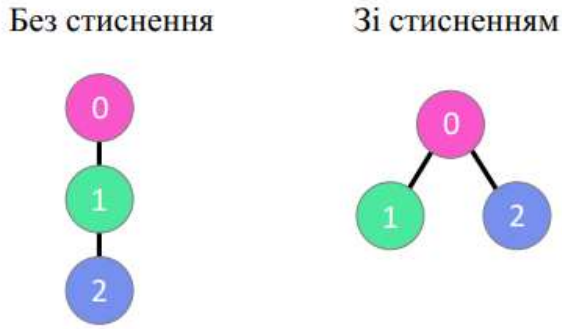

Рис. 3.2. Ілюстрація стиснення шляхів

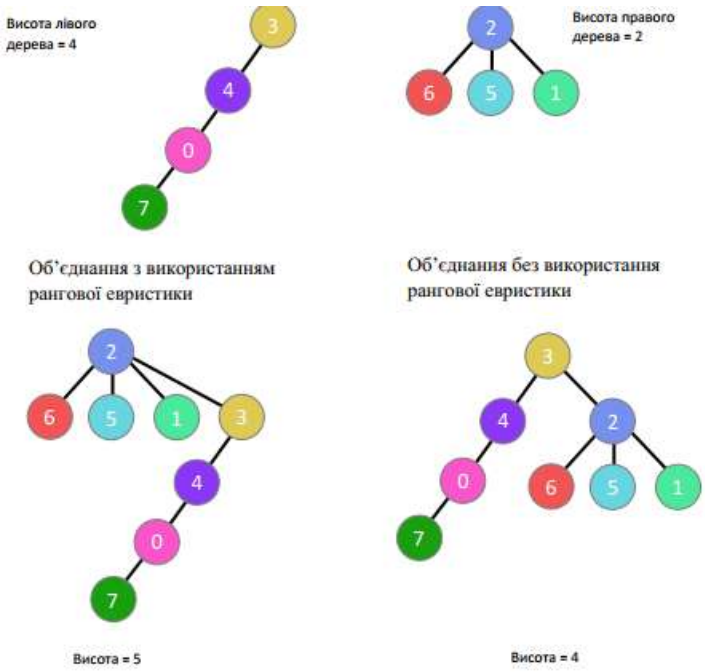

Рис. 3.3. Ілюстрація евристики об'єднання за рангом

3.2. Опис програмного забезпечення

Для виконання поставленої задачі було розроблено програмне забезпечення для обробки цифрових зображень на мові програмування C#.

Вхідні дані: цифрові зображення наступних форматів: JPEG (.jpg), BMP (.bmp), GIF (.gif), PNG (.png).

Вихідні дані: сегментовані цифрові зображення. Є можливість отримати всі сегменти на вхідному зображенні, позначені різним кольором, або масив сегментів.

Після зчитування зображення з файлу, програма проводить перетворення від растрового зображення до множини тривимірних точок. Кожна точка описується інтенсивністю компонент кожного з кольорових каналів. Далі є можливість змінити яскравість і/або контрастність зображення та перевірити наскільки параметр розмиття *sigma* впливає на зображення.

На основі множини тривимірних точок будується ґратковий граф і на ньому проводиться сегментація(є можливість швидкого порівняння з оригіналом). Результати можна оцінити за допомогою 3 функціоналів якості, а саме: суми внутрішньокластерних дисперсій, суми попарних внутрішньокластерних відстаней, загальної внутрішньокластерної дисперсії.

При побудові ґратковий графу, для визначення ваг ребер, можуть використовуватися 3 кольорові схеми: відтінки сірого, RBG, Lab.

У застосунку передбачена можливість роботи з великими зображеннями. Вони пропорційно зменшуються для розташування у робочій області. Слід зазначити, що всі маніпулювання будуть проведені над оригінальним (не зменшеним) зображенням.

На діаграмі діяльності (Додаток А) показані можливі сценарії роботи програми.

Побудова ґраткового графу

Для виконання сегментації на графі загальноприйнятим є з’єднання всіх пікселів з 4 або 8 сусідами. В даній роботі сусідами вважаться не тільки ті пікселі, що стоять поруч на горизонталі та вертикалі, а й на діагоналях. Таким чином піксель має 8 сусідів.

Для того, щоб побудувати ґратковий граф для зображення за однин прохід використовується шаблон (рис. 3.4). Результат його використання для цифрового зображення розмірами 5*4 показано на малюнку(рис. 3.5).

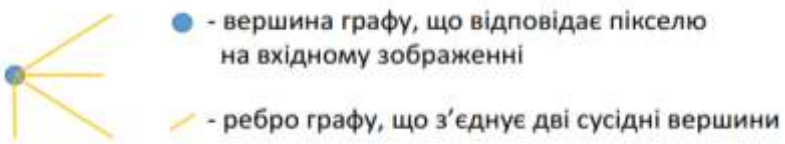

Рис. 3.4. Шаблон з'єднання пікселів

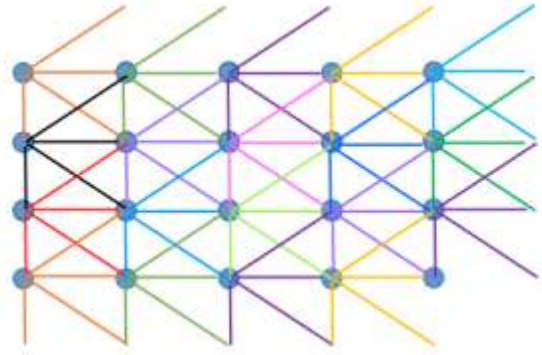

Рис. 3.5. Ґратковий граф для зображення 5*4.

3.2.1. Інструкція користувача

Після запуску програми на екрані з'являється початкове вікно (рис. 3.6).

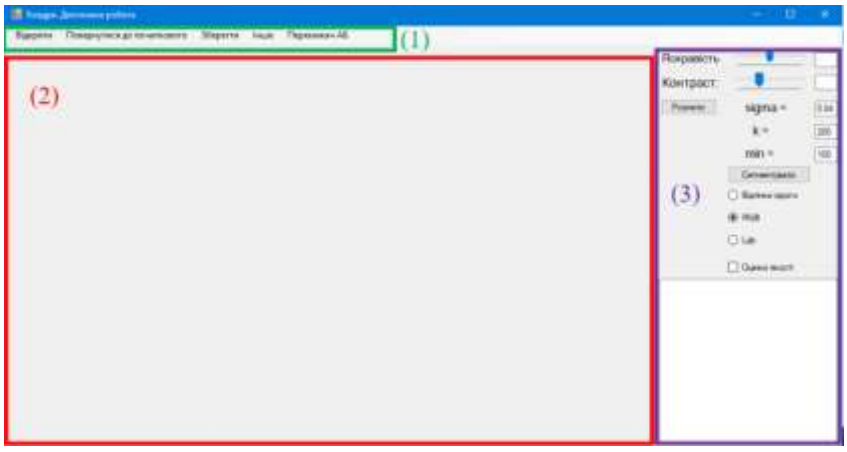

Рис. 3.6. Початкокий вигляд програми

Вікно застосунку складається з 3 основних областей:

1. меню програми;
2. область зображення;
3. область налаштувань.

Далі детально розглядаються кожна з областей.

Меню програми

Меню – частина інтерфейсу програми, яка маніпулює цифровими зображеннями, що подаються на вхід. З нього починається взаємодія користувача із застосунком.

В меню входять 5 пунків: «*Відкрити», «Повернутися до початкового», «Зберегти», «Інше» та «Перемикач АБ».*

Для початку роботи програми необхідно зчитати файл, що містить зображення формату BMP, GIF, JPEG, PNG, TIFF натиснувши на кнопку «Відкрити». Це призведе до відкриття діалогового вікна вибору файлу (рис. 3.7). Після вибору зображення, воно буде зчитано, перетворено на множину точок та виведено на екран в «область зображення».

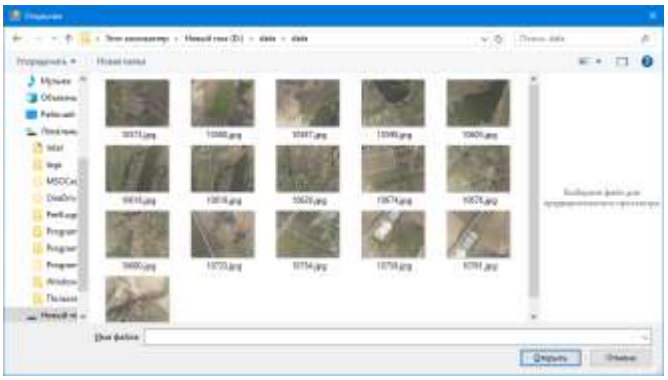

Рис. 3.7. Діалогове вікно вибору зображення для роботи

Кнопка «*Повернутися до початкового*» призначена для відміни всіх перетворень зображення.

Кнопка «Зберегти» призначена для збереження результатів сегментації цифрового зображення. Після її натискання буде відкрито діалогове вікно збереження файлу (рис. 3.8) для збереження останнього отриманого обробленого зображення. При збереженні користувачу буде дозволено обрати місце у його файловій системі, та ім’я файлу. За замовчуванням, застосунок пропонує встановити ім’я файлу, яке складається з початкового імені та дописаними до нього параметрами сегментації (*sigma, k, minSize*).

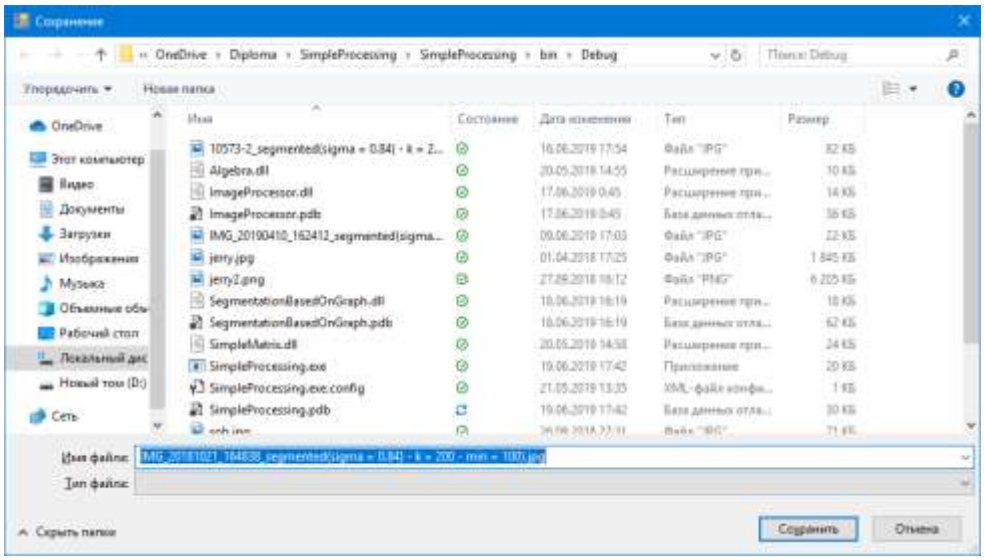

Рис. 3.8. Діалогове вікно збереження результатів сегментації

Кнопка «*Перемикач АБ*» призначена для порівняння результатів сегментації з початковим зображенням. Якщо ж після отримання результатів сегментації використати її, то вхідне зображення буде замінюватися результатами сегментації і навпаки. При чому, при виведенні результатів сегментації буде використовуватися кешована копія, тобто алгоритм заново виконуватися не буде.

Область налаштувань

Область налаштувань – частина інтерфейсу програми, яка відповідає за зміну яскравості та контрастності зображення, введення параметрів алгоритму сегментації (*sigma, k, minSize*), вибір кольорової схеми при виконанні алгоритму (відтінки сірого, RGB, Lab) та виведення деякої додаткової текстової інформації (кількість сегментів, та оцінки якості сегментації, якщо їх визначення було ввімкнене).

Обробка зображення

Для простої обробки цифрового зображення в додатку реалізовані зміни яскравості та контрастності.

Зміна контрастності відбувається за рахунок збільшення різниці між конкретним пікселем та середнім значенням кольору по всьому зображенні (рис. 3.9).

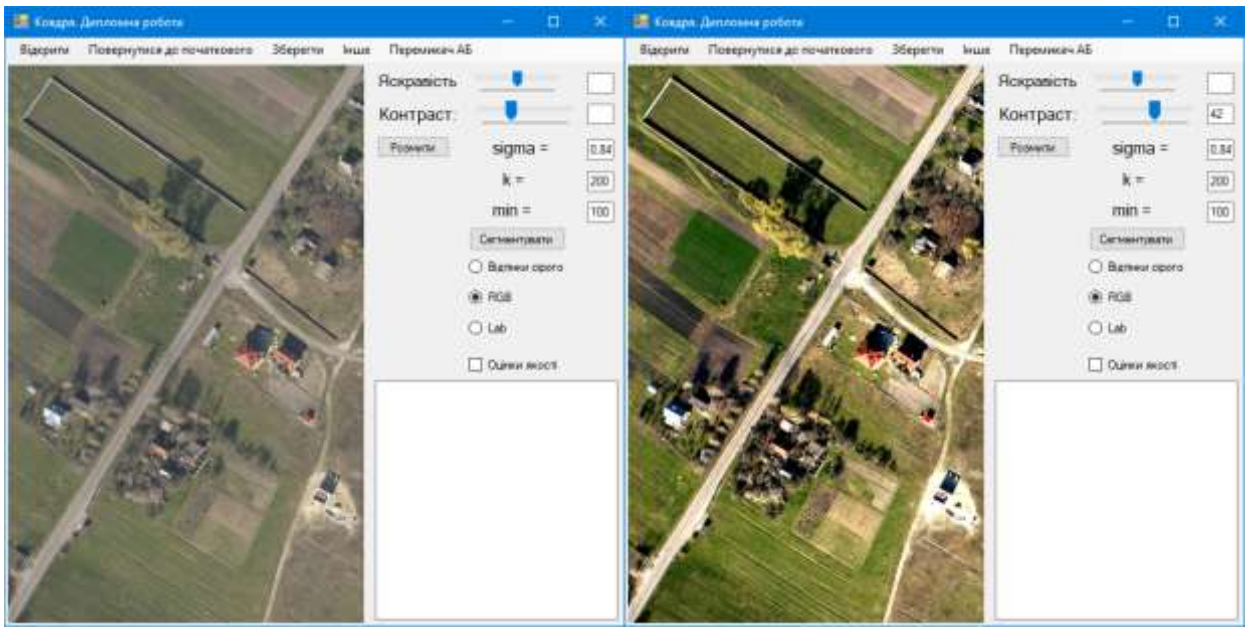

Рис. 3.9. Зміна контрастності

Параметри сегментації

Далі задаються параметри сегментації: *sigma* – параметр розмиття для згладження зображення, *k* –параметр що відповідає за ступінь відмінності між сегментами та *minSize* – розмір мінімального сегмента(на постобробці сегментації сегменти з розмірами меншими за цей параметр будуть об’єднуватися з сусідами). Дослідження залежності результатів сегментації від цих параметрів будуть показані у пункті 3.2.2.

За бажанням, користувач може подивитися як змінюється зображення під час розмиття з заданим параметром. Для цього поряд з параметром *sigma* є кнопка «Розмити». Але після підбору необхідного значення параметру, необхідно повернутися до вхідного зображення, оскільки розмиття включене в процедуру сегментації і буде виконано повторно,

Під кнопкою «сегментувати» надається можливість для вибору кольорового простору при виконанні сегментації. Доступні кольорові простори: відтінки сірого, RGB, Lab.

Під кольоровими просторами знаходиться перемикач для включення/виключення обрахунку оцінок якості отриманої сегментації.

Внизу області налаштувань знаходиться текстове поле для виведення кількості знайдених сегментів та оцінок якості сегментації, якщо їх обрахунок увімкнено.

Приклад виведення результатів сегментації з обрахованими оцінками якості показано на малюнку (рис. 3.10)

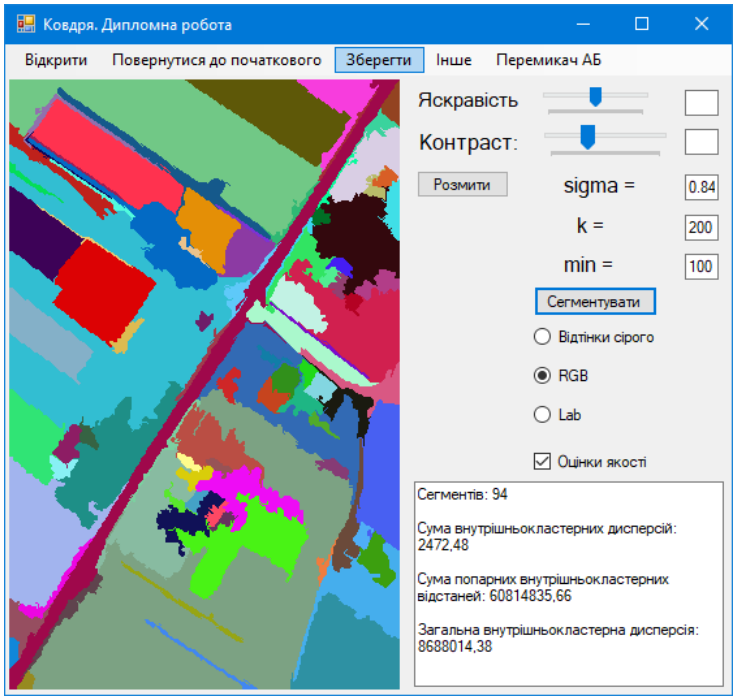

Рис. 3.10. Приклад виведення результатів сегментації

3.2.2. Дослідження результатів сегментації в залежності від значень параметрів

Аналіз впливу значень параметрів на результат сегментації було проведено на фотографії з аерофотозйомки (рис. 3.11). Розміри фото — 2000*1500 пікселів.

Для оцінювання якості сегментації було обрано суму внутрішньокластерних дисперсій. Функціонали «Сума попарних внутрішньокластерних відстаней» та «Загальна внутрішньокластерна дисперсія» — обчислювально складні, тому не розглядалися.

Для проведення дослідження було створено утиліту, яка зчитує зображення та проводить його сегментацію при різних значеннях параметрів (з визначенням оцінки якості). Результати аналізу зберігалися у файл MS Excel.

Кольоровий простір «Відтінки сірого» не розглядається через втрату інформації, яка відбувається при переході до нього.

Рис. 3.11. Фото, на якому проводились дослідження

Аналіз у кольоровому просторі RGB

У таблиці (табл. 3.1) наведено кількість сегментів, що отримані, в залежності від різних значень параметрів сегментації *k* та *minSize*.

Таблиця 3.1

Кількість сегментів при різних значеннях *k* та *minSize* (простір RGB)

k\ minSize	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**	**900**	**1000**	**1100**	**1200**
100	1398	759	542	433	362	317	282	255	237	218	203	192
200	762	390	260	204	171	145	123	112	101	98	88	87
300	564	301	201	155	133	108	95	87	78	74	71	67
400	426	240	160	121	102	82	72	68	62	57	51	51
500	358	211	150	112	92	78	68	66	63	58	52	49
600	308	190	130	98	81	70	62	59	54	52	44	41
700	302	180	126	95	80	68	58	56	51	49	44	42
800	226	130	90	68	59	47	41	35	31	30	29	28
900	203	112	76	57	44	35	33	30	28	28	28	28
1000	206	107	74	56	40	31	29	28	26	25	25	25
1100	197	102	73	53	39	32	29	27	26	25	25	25
1200	181	91	63	48	34	28	26	24	24	23	22	22

Тут і надалі параметр *sigma* дорівнює 0.83, оскільки він використовується, здебільшого лише для згладжування зображення.

На основі даних з таблиці 3.1 було побудовано залежність кількості сегментів від значень параметрів методу (рис. 3.12).

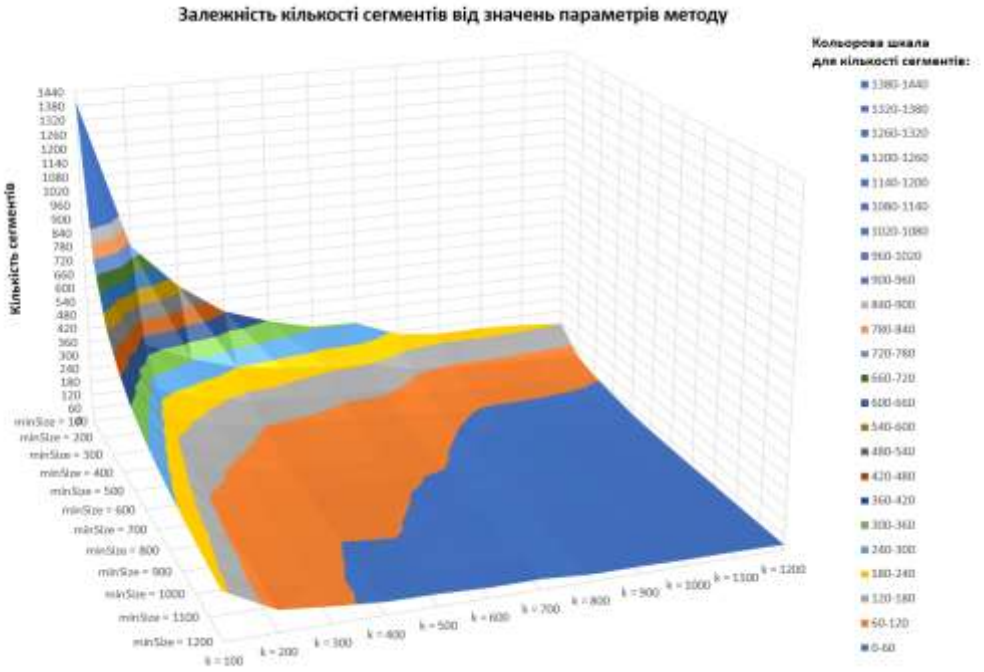

Рис. 3.12. Залежність кількості сегментів від значень параметрів методу (простір RGB)

Для отриманих результатів було пораховано оцінки якості. Дані представлені у таблиці(табл. 3.2). Значення округлено до цілих.

Таблиця 3.2

Оцінки якості результатів(простір RGB)

k\ minSize	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**	**900**	**1000**	**1100**	**1200**
100	32031	34051	35378	36235	37120	37989	38416	38992	39608	39955	40167	41442
200	36704	38114	39264	39954	40879	41653	42502	42764	43382	43676	44471	44480
300	39856	41031	42278	43080	43839	44685	44857	44886	45324	45653	45751	45835
400	49111	50032	50876	51512	52317	53142	53871	53990	54028	54345	54913	54913
500	64393	65187	65977	66472	66884	67164	67859	67907	68005	68169	68631	68647
600	68435	69110	69805	70174	70528	70855	71031	71071	71289	71408	71692	71743
700	68943	69565	70058	70730	70971	71365	71609	71614	71833	71951	72040	72361
800	77670	78321	78734	79043	79278	79648	79731	79818	80014	80154	80256	80564
900	74272	74766	75166	75427	75698	76056	76132	76189	76351	76351	76351	76351
1000	78393	78889	79142	79261	79445	79620	79620	79627	79705	79720	79720	79720
1100	78708	79187	79427	79551	79659	79784	79844	79854	80019	80034	80034	80034
1200	85285	85736	85932	86042	86139	86318	86430	86445	86445	86445	86508	86508

За даними з таблиці (табл. 3.2) було побудовано залежність суми внутнішньокластерних дисперсій від значень параметрів методу (рис. 3.13).

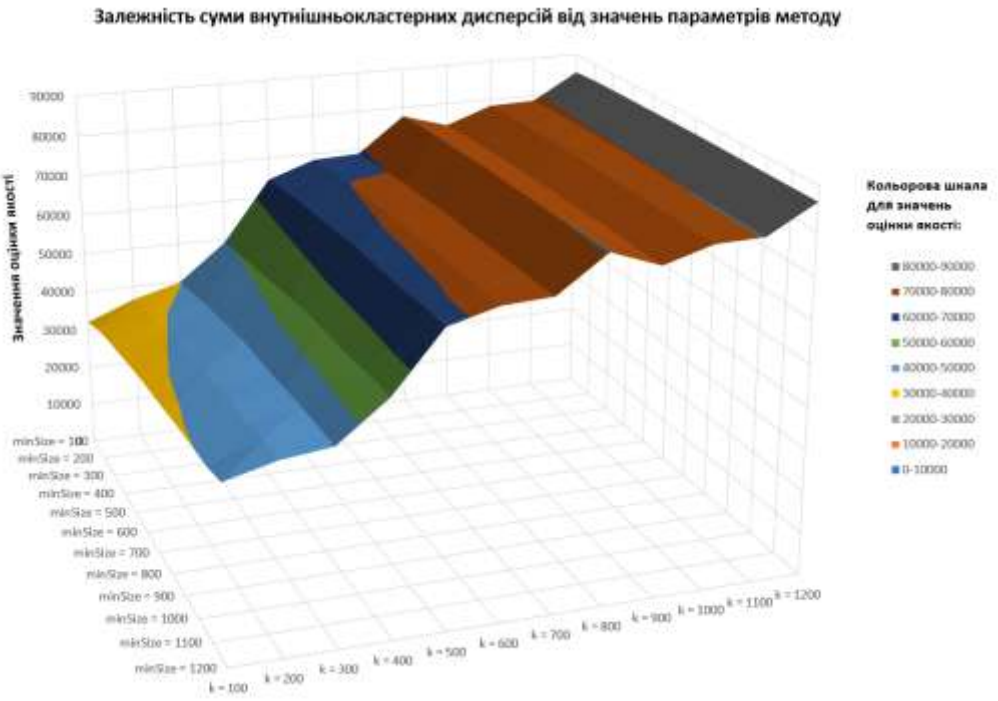

Рис. 3.13. Залежність суми внутнішньокластерних дисперсій
від значень параметрів методу (простір RGB)

З таблиці 3.1 видно, що при малих значеннях параметрів, відносно розміру зображення, утворюється багато сегментів (перелік йде на сотні), а при збільшенні значень параметрів кількість сегментів зменшується.

Згідно з таблицею 3.2, сума внутнішньокластерних дисперсій збільшується зі збільшенням значень параметрів (хоча і утворюються локальні максимуми).

Також за результатами сегментації, які представлені у додатку А, видно, що параметр *k* встановлює деталізацію (більші значення *k* призводять до утворення більших за розмірами «цільових» сегментів), в той час, коли *minSize* позбувається від «шумів» («нецільових» об’єктів з розмірами меншими за вказане значення).

Аналіз у кольоровому просторі Lab

Значення параметрів *k* та minSize використовувались аналогічні з RGB тестом.

У таблиці 3.3 наведено кількість отриманих сегментів при різних значеннях параметрів.

Таблиця 3.3.

Кількість сегментів при різних значеннях *k* та *minSize* (простір Lab)

k\ minSize	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**	**900**	**1000**	**1100**	**1200**
100	358	209	153	113	92	76	66	59	55	50	46	43
200	237	144	106	89	71	56	52	48	45	44	39	34
300	188	93	69	52	44	39	36	32	27	27	26	25
400	131	66	42	31	26	25	22	20	20	18	18	17
500	121	62	44	33	29	26	21	21	20	20	19	18
600	111	55	41	30	26	25	20	20	19	19	18	17
700	82	48	36	28	24	21	20	20	19	17	17	16
800	80	45	34	26	23	21	20	17	13	12	12	12
900	70	30	23	19	13	12	12	10	8	7	7	7
1000	59	27	21	18	13	11	9	8	8	7	7	6
1100	79	44	35	28	22	16	16	14	14	14	14	14
1200	74	44	35	30	24	18	18	16	14	14	14	14

На малюнку (рис. 3.14) показано отриману залежність кількості сегментів від значень параметрів методу.

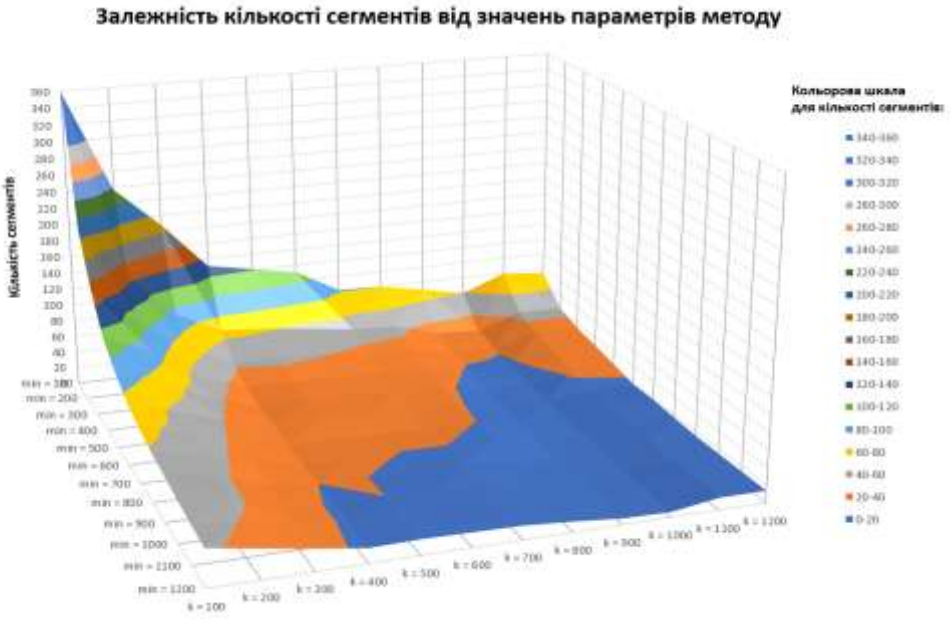

Рис. 3.14. Залежність кількості сегментів від значень параметрів методу (простір Lab)

Отримані оцінки якості представлені у таблиці 3.4.

Таблиця 3.4.

Оцінки якості результатів (простір Lab)

k\ minSize	100	200	300	400	500	600	700	800	900	1000	1100	1200
100	21221	21467	21665	21959	22091	22224	22282	22401	22415	22563	22803	22806
200	24161	24322	24443	24526	24590	24702	24720	24789	24863	24863	24995	25105
300	27043	27196	27273	27331	27380	27432	27453	27485	27534	27534	27563	27564
400	27990	28089	28167	28188	28221	28246	28256	28258	28258	28342	28342	28344
500	27250	27355	27413	27434	27471	27502	27514	27514	27565	27565	27610	27612
600	29679	29776	29809	29825	29866	29868	29880	29880	29930	29930	29975	29977
700	29552	29607	29630	29640	29678	29683	29684	29684	29737	29789	29789	29790
800	28072	28155	28184	28222	28237	28242	28243	28315	28402	28409	28409	28409
900	30272	30389	30442	30494	30526	30528	30528	30614	30629	30631	30631	30631
1000	30222	30309	30361	30389	30420	30463	30470	30548	30548	30550	30550	30577
1100	37198	37262	37298	37317	37343	37364	37364	37369	37369	37369	37369	37369
1200	37210	37268	37303	37318	37343	37365	37365	37370	37371	37371	37371	37371

Зобразивши оцінки якості і значення параметрів, при яких вони були отримані, маємо залежність (рис. 3.15).

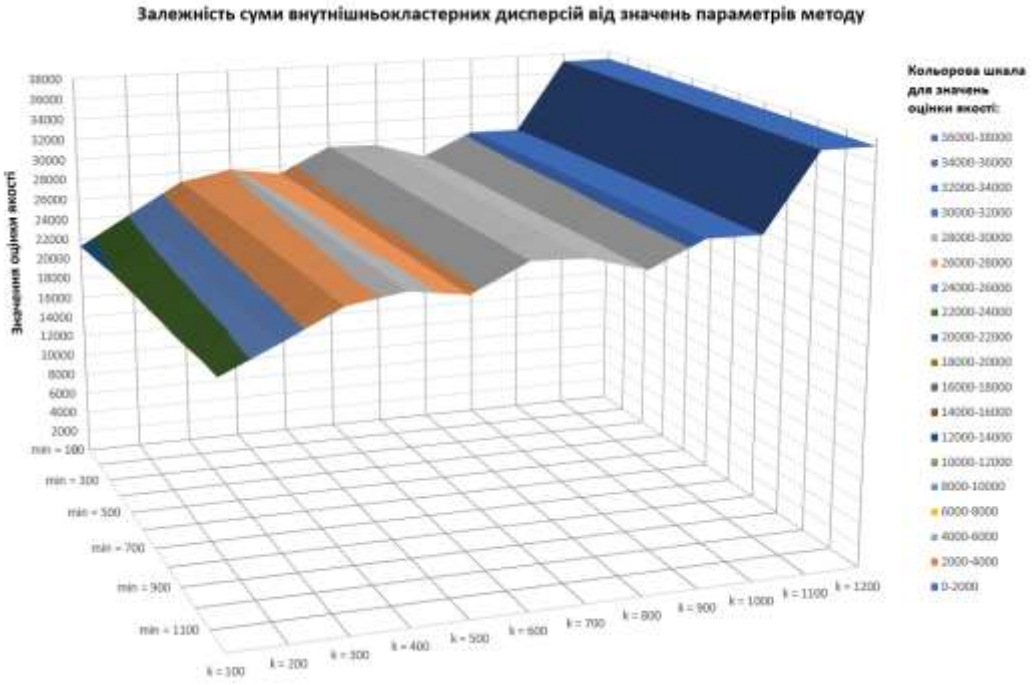

Рис. 3.15. Залежність суми внутнішньокластерних дисперсій від значень параметрів методу (простір Lab)

Сума внутрішньокластерних дисперсій зростає при збільшенні значень параметрів.

У порівнянні з простором RGB, у просторі Lab утворюється менша кількість сегментів при малих значеннях параметрів, відносно розмірів зображення.

ВИСНОВКИ

1. Проведено аналіз літератури та вивчено основні підходи до сегментації цифрових зображень, а саме методи, засновані на кластеризації, стисненні, диференціальних рівняннях з частинними похідними та методи з використанням гістограм;
2. Описано метод ефективної сегментації на ґратковому графі, починаючи з загального підходу сегментації цифрових зображень за їх представленнями у вигляді графів. Визначено предикат попарного порівняння сегментів та покроково описано алгоритм їх отримання, що засновується на методі Краскала для знаходження мінімального остовного дерева графу;
3. Розглянуто обробку зображень за допомогою операції «згортка» та проілюстровано 4 варіанти обробки країв: розширення, обгортання, віддзеркалення по краю та зріз ядра. Описано розмиття Гауса для отримання згладжених зображень. Розглянуто 3 кольорові моделі;
4. Розроблено автоматизовану систему для сегментації цифрових зображень з підтримкою наведених кольорових моделей;
5. Проведено дослідження залежності результатів сегментації в залежності від значень параметрів методу. Для зручного перегляду результатів сегментації та отриманих значень оцінок якості для них, було розроблено утиліту, яка призначена для збереження цих даних у форматі програми MS Excel. Виявлено, що при збільшенні значень одного з параметрів(*k* або *minSize*) кількість сегментів зменшується, а сума внутрішньокластерних дисперсій збільшується(можливі локальні екстремуми);
6. Перспективою даної роботи є розробка гібрида методів пірамідальної сегментації та ефективної сегментації на ґратковому графі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Popescu. D. Intelligent image processing system for detection and segmentation of regions of interest in retinal images / D. Popescu, L. Ichim // Department of control engineering and industrial informatics — 2018.
2. Li. Q. Retinal image segmentation using double-scale nonlinear thresholding on Vessel support regions / Q. Li, M. Zheng, F. Li, J. Wang, Y. Geng, H. Jiang // Department of ophthalmology — 2017.
3. Chen D. An interactive image segmentation method in hand gesture recognition / D. Chen, G. Li, Y. Sun, J. Kong, G. Jiang, H. Tang, Z. Ju // School of machinery and automation — 2017.
4. Yilmaz A. Object tracking: A survey / A. Yilmaz, O. Javed, M. Shah // Image processing and computer vision — 2006.
5. Rother C. Grabcut: Interactive foreground extraction using iterated graph cuts / C. Rother, A. Blake // Microsoft Research Cambridge — 2004.
6. Henry C. Road Segmentation in sar satellite images with deep fully- convolutional neural networks / C. Henry, S. M. Azimi, N. Merkle // German Aerospace Center — 2018.
7. Barghout L. Perceptual information processing system / L. Barghout, W. Lawrence // Department of electrical engineering and computer sciences — 2003.
8. Kanungo. T. An efficient k-means clustering algorithm: analysis and implementation / T. Kanungo, D. Mount, N. Netanyahu, C. Piatko // Department of Computer Science — 2002.
9. Shapiro L. Computer vision / L. Shapiro and G. Stockman, Seattle Washington 2001 — pp. 279–325.
10. Ohlander R. Picture segmentation using a recursive region splitting method / R. Ohlander, K. Price, D. Reddy, 1978 — pp 313–333
11. Caselles V. Geodesic active contours / V. Caselles, R. Kimmel, G. Sapiro — International Journal of Computer Vision , 61–79, 1997
12. Osher S. Geometric level set methods in imaging vision and graphics / S. Osher, N. Paragios — Springer, 2003.

13. Huttenlocher D. Efficient graph-based image segmentation / D. P. Huttenlocher, P.F. Felzenszwalb // Image processing — 2004.
14. Приставка П.О. Аналіз даних. Електронний посібник для студентів спеціальності «Прикладна математика» / П.О Приставка, О.М. Мацуга — 2010.
15. Damelin S. The mathematics of signal processing / S. Damelin; W. Miller — Cambridge University Press, 2012 — 462 P.
16. Young I. Fundamentals of image processing / I.T. Young, J.J. Gerbrands, L.J. van Vliet — Delft University of Technology, 1997 — 113 pp.
17. Hamey L. A functional approach to border handling in image processing / Leonard G. C. Hamey // Department of computing — 2015.
18. Szeliski R. Computer vision: algorithms and applications / Richard Szeliski — Springer, 2010 — 979 pp.
19. Tkalcic M. Colour spaces / Marko Tkalcic // Electrical engineering — 2003.
20. Ford A. Colour space conversions / A. Ford, A. Roberts // Image processing — 1998.
21. Bal A. Selected properties of perceptual colour spaces / A. Bal, H. Palus. P. Wolezyk // Department of automatic control — 2002.

ДОДАТОК А

Діаграма діяльності