МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чернігівський національний технологічний університет

РОЗПІЗНАВАННЯ ОБРАЗІВ

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт з дисципліни

“ Розпізнавання образів ”
для студентів спеціальності
123- "Комп'ютерна інженерія"

Обговорено і рекомендовано
на засіданні кафедри
інформаційних та комп'ютерних систем
Протокол №_7 від 28.02..2018 р

Чернігів 2018

.

Розпізнавання образів. Методичні вказівки до лабораторних робіт з дисципліни “Розпізнавання образів” для студентів спеціальності 123 “Комп'ютерна інженерія”. / Укл. доц. В.А. Бичко – Чернігів: ЧДТУ, 2017. – 52 с. Укр. мовою.

Укладач: Бичко В.А, кандидат фізико-математичних наук, доцент

Відповідальний за випуск: Зайцев С.В., завідувач кафедри інформаційних та комп'ютерних систем, доктор технічних наук, доцент.

Рецензент: Пріла О.А., кандидат технічних наук, доцент

Зміст

Вступ

Дисципліна "Розпізнавання образів" відноситься до розряду обов'язкових дисциплін професійно-орієнтованого напряму "Комп'ютерна інженерія".

Необхідною передумовою для освоєння даної дисципліни є знання студентами таких навчальних курсів, як "Програмування" та "Алгоритми та методи обчислень".

Вивчення дисципліни сприяє більш глибокому розумінню інженерних завдань у сфері розпізнавання образів і освоєнню сучасних методів їх застосування.

1 Лабораторна робота №1

ВИМІР ПАРАМЕТРІВ ГРАФІЧНИХ ОБ'ЄКТІВ

Мета. - Отримати теоретичні та практичні навички по роботі з графічними зображеннями, навчитися виявляти, фільтрувати і вимірювати параметри об'єктів на зображенні з використанням пакету Image Processing Toolbox середовища технічного моделювання MatLab

1.1 Теоретичні відомості

Інтегровані середовища для моделювання та виконання програм цифрової обробки зображень і сигналів містять потужні засоби для інженерно-наукових розрахунків і візуалізації даних. Більшість сучасних пакетів підтримує візуальне програмування на основі блок-схем. Це дозволяє створювати програми фахівцям, які не володіють технікою програмування. До таких пакетів відноситься Image Processing Toolbox системи MATLAB, розроблений фірмою MathWorks. Цей пакет володіє потужними засобами для обробки зображень. Вони мають відкриту архітектуру і дозволяють організовувати взаємодію з апаратурою цифрової обробки сигналів, а також підключати стандартні драйвера.

Система MATLAB і пакет прикладних програм Image Processing Toolbox (IPT) є хорошим інструментом розробки, дослідження і моделювання методів і алгоритмів обробки зображень. При вирішенні задач обробки зображень пакет IPT дозволяє йти двома шляхами. Перший з них полягає в самостійній програмній реалізації методів і алгоритмів. Інший шлях дозволяє моделювати рішення задачі за допомогою готових функцій, які реалізують найбільш відомі методи і алгоритми обробки зображень. І той, і інший спосіб виправданий.

Перш ніж використовувати для вирішення будь-яких завдань обробки зображень стандартні функції пакету IPT, розробник повинен досконало їх досліджувати. Для цього він повинен точно знати, який метод і з якими параметрами реалізує та чи інша функція. У тому чи іншому підході до вирішення задачі обробки відеоданих об'єктом дослідження є зображення. Для цього розглянемо коротко особливості подання зображень у IPT.

Зображення бувають векторними і растровими. Векторним називається зображення, описане у вигляді набору графічних примітивів. Растрові ж зображення являють собою двовимірний масив, елементи якого (пікселі) містять інформацію про колір. У цифровій обробці

використовуються растрові зображення. Вони в свою чергу діляться на типи - бінарні, напівтонові, палітровие, повнокольорові.

Елементи бінарного зображення можуть приймати тільки два значення - 0 або 1. Природа походження таких зображень може бути найрізноманітнішою. Але в більшості випадків, вони отримуються в результаті обробки напівтонових, палітрових або повнокольорових зображень методами бінарізаціі з фіксованим або адаптивним порогом. Бінарні зображення мають ту перевагу, що вони дуже зручні при передачі даних.

Півтонове зображення складається з елементів, які можуть приймати одне із значень інтенсивності якого-небудь одного кольору. Це один з найбільш поширених типів зображень, який застосовується при різного роду дослідженнях. У більшості випадків використовується глибина кольору 8 біт на елемент зображення.

У палітрових зображеннях значення пікселів є посиланням на клітинку карти кольорів (палітру). Палітра являє собою двовимірний масив, на шпальтах якого розташовані інтенсивності колірних складових одного кольору. На відміну від палітрових, елементи повнокольорових зображень безпосередньо зберігають інформацію про інтенсивність колірних складових.

Вибір типу зображення залежить від розв'язуваної задачі, від того, наскільки повно і без втрат потрібна інформація може бути представлена з заданою глибиною кольору. Також слід врахувати, що використання повнокольорових зображень вимагає великих обчислювальних витрат.

Залежно від типу зображення вони по-різному представляються в різних форматах. Цей момент буде дуже важливим при створенні програм в середовищі IPT. Найбільш зручно залежність способів подання елементів зображення від типу і формату представити у вигляді таблиці.

Таблиця 1.1

Тип зображення	double	uint8
Бинарне	0 и 1	0 и 1
Напівтоновое	[0, 1]	[0, 255]
Палітрове	[1, розмір палітри], де 1 – перший рядок палітри	[0, 255], где 0 - перший рядок палітри
Повнокольорове	[0, 1]	[0, 255]

Надалі, при розгляді методів обробки зображень, будемо вважати, що зображення представляється матрицею чисел (розмір матриці N × M), де значення кожного елемента відповідає певному рівню квантування його енергетичної характеристики (яскравості). Це так звана піксельна система координат. Вона застосовується в більшості функцій пакету IPT. Існує також просторова система координат, де зображення представляється безперервним числовим полем квадратів з одиничною величиною.

Кількість квадратів збігається з числом пікселів. Значення інтенсивності елемента в центрі квадрата збігається зі значенням відповідного пікселя в піксельній системі координат. При вирішенні практичних завдань, пов'язаних з вимірюваннями реальних геометричних розмірів об'єктів на зображенні, зручно використовувати просторову систему координат, так як вона дозволяє враховувати кількість пікселів на метр системи.

Маска фільтра (або апертура) являє собою матрицю розміру n × m. Вона накладається на зображення і здійснюється множенням елементів маски фільтра та відповідних елементів зображення з подальшою обробкою результату. Коли маска пересувається до кордону зображення, виникає так зване явище крайового ефекту. Щоб уникнути цього небажаного ефекту необхідно, коли маска вийшла за межі вихідного зображення, треба доповнити його елементами зображення, симетричними відносно його країв.

Обробка зображень здійснюється рекурсивними і нерекурсивними методами. Рекурсивні методи використовують результат обробки попереднього пікселя, нерекурсивні - не використовують. У більшості випадків використовуються нерекурсивні алгоритми обробки зображень.

1.2 Геометричні характеристики бінарних зображень

У лабораторній роботі розглядаються чорно-білі (бінарні) зображення. Їх легше отримувати, зберігати і обробляти, ніж зображення, в яких є багато рівнів яскравості. Нижче розглянемо способи розрахунку простих геометричних характеристик зображень, таких як: площа об'єкта, його положення і орієнтація. Подібні величини можуть використовуватися, наприклад, в процесі управління механічним маніпулятором при його роботі з деталями.

Оскільки зображення містять великий обсяг інформації, важливу роль починають грати питання її подання. Покажемо, що деякі геометричні характеристики можна витягти з проекцій бінарних зображень. Проекції набагато легше зберігати і обробляти. Також розглянемо безперервні бінарні зображення, характеристична функція яких дорівнює нулю або одиниці в кожній точці площини зображення. Це спрощує аналіз, однак при використанні ЕОМ зображення необхідно розбити на дискретні елементи.

1.2.1 Бінарні зображення

Почнемо з випадку, коли в полі зору знаходиться об'єкт, а усі інше вважається "фоном". Якщо об'єкт виявляється помітно темнішим (або світлішим), ніж фон, то легко визначити характеристичну функцію b (x, y),

яка дорівнює нулю для всіх точок зображення, відповідних фону, та одиниці для точок на об'єкті (рис.1) або навпаки.

Часто бінарне зображення отримують пороговим поділом звичайного зображення. До нього також можна прийти шляхом порогового поділу відстані на "зображенні", отриманому на основі вимірів відстаней.

Таку функцію, приймаючу два значення і звану бінарним зображенням, можна отримати пороговим поділом напівтонового зображення. Операція порогового поділу полягає в тому, що характеристична функція покладається рівною нулю в точках, де яскравість більше деякого порогового значення, і одиниці, де вона не перевершує його (або навпаки).

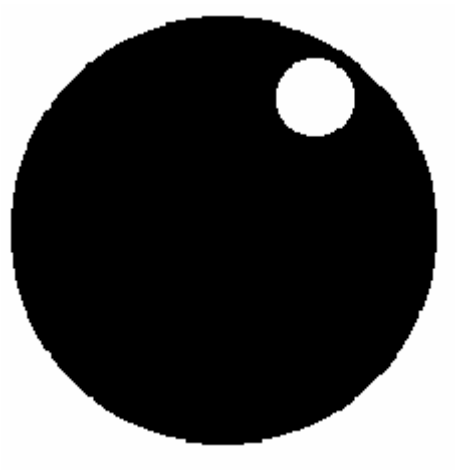


Рисунок 1.1 – Бінарне зображення, яке визначається характеристичною функцією f (x, y), яка приймає значення "нуль" і "одиниця ”

Іноді буває зручно розглядати компоненти зображення, а також отвори в них як множини точок. Це дозволяє комбінувати зображення за допомогою теоретико-множинних операцій, наприклад, об'єднання і перетин. В інших випадках зручно використовувати булеві операції для кожного пікселя. Насправді це лише два різні способи опису одних і тих же дій над зображеннями.

Оскільки кількість інформації, що містяться в бінарному зображенні, на порядок менше, ніж в співпадаючому з ним за розмірами напівтоновому зображенні, бінарне зображення легше обробляти, зберігати і пересилати. Цілком природно, що певна частина інформації при переході до бінарного зображення втрачається, і, крім того, звужується коло методів обробки таких зображень. В даний час існує досить повна теорія того, що можна і чого не можна робити з бінарними зображеннями.

Насамперед, ми можемо обчислити різні геометричні характеристики зображення, наприклад, розмір і положення об'єкта. Якщо в полі зору перебуває більше одного об'єкта, то можна визначити топологічні характеристики наявної сукупності об'єктів: наприклад,

різниця між числом об'єктів і числом отворів (число Ейлера). Цієї операції відповідає функція BWEULER - обчислення чисел Ейлера в пакеті.

Приклад:

Image Processing Toolbox:

L = imread ('test.bmp');

L = double (L);

imshow (L);

e = bweuler (L)

e = 1; %

На об'єкті дійсно один отвір.

Неважко також помітити окремі об'єкти і обчислити геометричні характеристики для кожного з них окремо. Нарешті, перед подальшою обробкою зображення можна спростити, поступово модифікуючи його ітеративним чином.

Обробка бінарних зображень добре зрозуміла, і її неважко пристосувати під швидку апаратну реалізацію, але при цьому потрібно пам'ятати про обмеження. Ми вже згадували про необхідність високого ступеня контрасту між об'єктом і фоном. Крім того, образ, який нас цікавить повинен бути істотно двовимірним. Адже усі, що ми маємо, - лише обриси чи силует об'єкта. При наяності такої інформації важко судити про його форму або просторове положення.

Характеристична функція b(x, y) визначена в кожній точці зображення. Таке зображення будемо називати безперервним. Пізніше ми розглянемо дискретні бінарні зображення, одержані шляхом відповідного розбиття поля зображення на елементи.

1.2.2 Прості геометричні характеристики

Припустимо знову, що в полі зору перебуває лише один об'єкт. Якщо відома характеристична функція b (x, y), то площа об'єкта обчислюється таким чином:

$$A = \iint Ib\ (x, y)\ dxdy, \qquad (1)$$

де інтегрування здійснюється по всьому зображенню I. При наявності більше одного об'єкта ця формула дає можливість визначити їх сумарну площу.

Приклад:

В системі Matlab цій операції відповідає функція BWAREA - обчислення площі об'єктів.

L = imread ('test.bmp');

L = double (L);

imshow (L);

S = bwarea (L (:,:, 1))

e = 24926; % Площа об'єкта в пікселах (розмір зображення 236x236).

1.2.3 Площа та положення

Як визначити положення об'єкта на зображенні? Оскільки об'єкт, як правило, складається не з однієї єдиної точки, ми повинні чітко визначити зміст терміну "становище". Звичайно як одну з характерних точок об'єкта вибирають його геометричний центр (рис. 2)

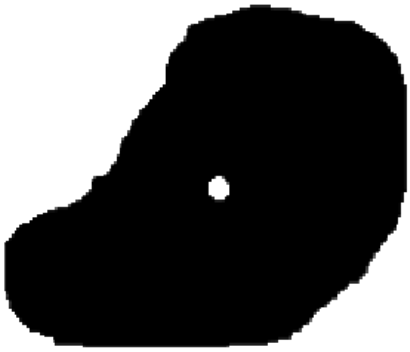


Рисунок 1.2 – Бінарне положення області на бінарному зображенні, яке можна визначити її геометричним центром. Останній являє собою центр мас тонкого листа матеріалу тієї ж форми

Геометричний центр - це центр мас однорідної фігури тієї ж форми. У свою чергу центр мас визначається точкою, в якій можна сконцентрувати всю масу об'єкта без зміни його першого моменту щодо будь-якої осі. У двовимірному випадку перший момент щодо осі x розраховується за формулою

$$x\iint Ib\,(x, y)\,dxdy = \iint I\,xb\,(x, y)\,dxdy, \quad (2)$$

а щодо осі y- за формулою ~

$$y\iint [b\,(x, y)\,dxdy = \iint [yb\,(x, y)\,dxdy, \quad (3)$$

де (x, y) - координати геометричного центру. Інтеграли в лівій частині наведених співвідношень - не що інше, як площа A, про яку йшлося вище. Щоб знайти величини x і y, необхідно припустити, що величина A не дорівнює нулю. Зауважимо принагідно, що величина A являє собою момент нульового порядку функції b (x, y).

1.2.4 Орієнтація

Ми також хочемо визначити, як розташований об'єкт в полі зору, тобто його орієнтацію. Зробити це дещо складніше. Припустимо, що об'єкт трохи витягнуть уздовж деякої осі; тоді її орієнтацію можна прийняти за орієнтацію об'єкта. Як точно визначити вісь, уздовж якої витягнуть об'єкт? Зазвичай вибирають вісь мінімального другого моменту. Вона являє собою двовимірний аналог осі найменшої інерції. Нам необхідно знайти пряму, для якої інтеграл від квадратів відстаней до точок об'єкта мінімальний; цей інтеграл має вигляд

$$E = \iint [r^2 b(x, y)\, dxdy, \quad (4)$$

де r - відстань уздовж перпендикуляра від точки з координатами (x, y) до шуканої прямий.

Інший шлях вирішення проблеми полягає в спробі знайти кут повороту θ, при якому матриця других моментів розміру 2 × 2 має діагональний вигляд.

(У пакеті Matlab операції визначення центру мас, орієнтації, а також інші морфометричні ознаки обчислюються за допомогою функції IMFEATURE.)

1.2.5 Проекції

Для обчислення положення та орієнтації об'єкта достатньо знати перші и другі моменти. (При цьом залишається двозначність у виборі напрямку на осі.) Щоб найти їх значення, немає необхідності шукати вихідне зображення: достатньо знати його проекції. Зазначений факт представляє інтерес, оскількі проекції описуються більш компактно и призводять до швидшого алгоритму.

1.2.6 Дискретні бінарні зображення

Дотепер ми розглядали безперервні бінарні зображення, визначені у всіх точках площини. Цілком природньо, що при переході до дискретних зображень інтеграли стають сумами. Наприклад, площа обчислюється (в одиницях площі елемента зображення) у вигляді суми

$$a = \sum_{i=1}^{n} \sum_{j=1}^{m} b_{i,j} \quad (5)$$

де bi- значення бінарного зображення в точці, що знаходиться в і -му рядку та j-му стовпці. Тут ми вважали, що поле зображення розбите на квадратну решітку з n стовпцями і m рядками.

Зазвичай зображення проглядається, як рядок за рядком в тій же самій послідовності, в якій телевізійний промінь біжить по екрану (якщо не враховувати того, що парні рядки зчитуються слідом за непарними). Як тільки отримано значення чергового елементу зображення, перевіряємо рівність bj = 1. Якщо воно виконується, додаємо 1, до лічильника значення площі, перших моментів і других моментів. По закінченню циклу сканування за допомогою цих значень легко знайти площу, положення й орієнтацію.

1.3 Функції отримання характеристики та трансформації бінарних зображень

Зображення містять великий обсяг інформації. Наведені далі методи можуть знайти застосування в задачах візуальної інспекції, виявлення і розпізнавання об'єктів.

У пакеті Image Processing Toolbox системи Matlab існує багато функцій, що здійснюють обробку бінарних зображень, зокрема, морфологічні операції. Серед них - BWMORPH, DILATE, ERODE, BWPERIM, MAKELUT, BWFILL, BWSELECT, IMFEATURE та інші.

При обробці, зображення значення кожного нового елемента можна визначити як результат локальної операції над відповідним елементом вихідного зображення.

Отримане бінарне зображення можна знову піддати обробці в наступному циклі обчислень. Це процес, який називають ітеративною модифікацією, вельми корисний, оскільки дозволяє поступово перевести важке для обробки зображення в таке, яке піддається раніше описаним методам.

1.3.1 Функція BWAREA

Ця функція виконує обчислення площі об'єктів.

Синтаксис:

a = bwarea (BW)

Функція a = bwarea (BW) обчислює сумарну площу всіх об'єктів на бінарному зображенні BW. Площа приблизно відповідає числу пікселів об'єктів на зображенні, але точно йому не дорівнює.

1.3.2 Функція BWAREAOPEN

Ця функція виконує видалення невеликих об'єктів на бінарному зображенні

синтаксис:

BW2 = bwareaopen (BW, P)

або

BW2 = bwareaopen (BW, P, CONN)

Функція BW2 = bwareaopen (BW, P) видаляє з бінарного зображення усі зв'язані компоненти (об'єкти), площа яких менше P пікселів, і поміщає результат обробки в зображення BW2. За замовчуванням зв'язність дорівнює 8 для двох вимірів, 26 - для трьох вимірів.

У функції BW2 = bwareaopen (BW, P, CONN) додатково вказується параметр зв'язності. Параметр CONN може приймати будь-яке з наведених нижче значень.

Таблиця 1.1

Значення параметра CONN	Опис
двовимірна зв'язність	
4	4-зв'язкова околиця
8	8-зв'язкова околиця
тривимірна зв'язність	
6	6-зв'язкова околиця
18	18-зв'язкова околиця
26	26-зв'язкова околиця

Можливості підключення може бути визначена і для інших вимірів за допомогою параметра CONN. Слід зазначити, що зв'язність повинна бути симетричною або мати центральний елемент.

Вимоги до вихідних даних.

- Початкове зображення повинно бути представлено логічним або числовим нерозрідженим масивом.
- Результат обробки, який поміщається в зображення BW2, є логічним масивом.

*Алгоритм (*Основні кроки*)*:

1. Визначення зв'язкових компонент.

L = bwlabeln (BW, CONN);

2. Обчислення площі кожного компонента.

S = regionprops (L, 'Area');

3. Видалення (переміщення) невеликих об'єктів.

bw2 = ismember (L, find ([S.Area]> = P));

Приклад:

Завнання. Необхідно видалити всі об'єкти зображення, які займають еквівалентну площу менше 40 пікселів.

Виконання.

1. Зчитування і візуалізація вихідного зображення:

bw = imread ('text.tif');

imshow (bw)

2. Видалення всіх об'єктів менше 40 пікселів.

bw2 = bwareaopen (bw, 40);

figure, imshow (bw2)

В результаті обробки було видалено кілька об'єктів (букв), площа яких менше 40 пікселів (рисунок 1.3).

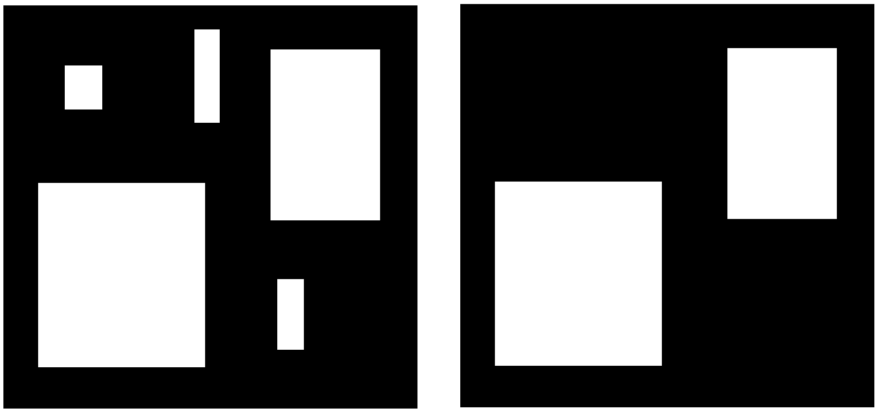


Рисунок 1.3 а)початкове зображення б) зображеня після видалення об‘єктів з площею меньшк порогового значення.

1.3.3 Функція BWLABEL

Ця функція виконує пошук об'єктів
синтаксис:ї
L = bwlabei (BW, n)
[L, num] = bwlabel (BW, n)
Функція L = bwlabel (BW, n) шукає на бінарному зображенні BW зв'язані області пікселів об'єктів і створює матрицю L, кожен елемент якої дорівнює номеру об'єкта, якому належить відповідний піксель зображення BW. Розмір матриці нумерованих об'єктів L дорівнює розміру BW. Об'єкти нумеруються по порядку починаючи з 1. Елементи, що мають значення 1, відносяться до першого об'єкту, що мають значення 2 відносяться до другого об'єкту і т.д. Якщо елемент в матриці L дорівнює 0, то це означає, що відповідний піксель вихідного зображення відноситься до фону. Параметр n вказує критерій зв'язності, який використовується для знаходження зв'язкових областей - об'єктів. Параметр n може приймати значення 4 або 8 (значення за замовчуванням).

Функція [L, num] = bwlabel (BW, n) додатково в параметр num повертає кількість об'єктів, знайдених на зображенні BW.

Матриця L має формат представлення даних double.

Матрицю номерів об'єктів L зручно відображати на екрані як кольорове зображення. Для цього слід разом з L використовувати палітру з кількістю кольорів більше або рівним num + 1. Якщо палітра містить різні кольори, то кожен об'єкт буде пофарбований в унікальний колір.

1.3.4 Функція BWLABELN

Ця функція визначає зв'язані компоненти бінарного зображення
Синтаксис:

L = bwlabeln (BW)
[L, NUM] = bwlabeln (BW)
[L, NUM] = bwlabeln (BW, CONN)

Функція L = bwlabeln (BW) повертає матрицю міток L, що містить мітки зв'язаних компонентів на бінарному зображенні BW. Зображення BW може бути будь-якої розмірності; розмірність матриці міток L збігається з розмірністю зображення BW. Елементи матриці L являють собою цілі числа більше або рівні 0. Пікселі, відмічені нулями, є фоном. Пікселі, відмічені 1, вказують на перший об'єкт; пікселі, відмічені 2, вказують на другий об'єкт, і т.д. За замовчуванням зв'язаність дорівнює 8 для двох вимірів, 26 - для трьох вимірів і conndef (ndims (BW), 'maximal') - для великих вимірювань.

Функція [L, NUM] = bwlabeln (BW) повертає в параметрі NUM число зв'язкових об'єктів, знайдених на основі аналізу бінарного зображення BW.

Функція [L, NUM] = bwlabeln (BW, CONN) визначає компоненти необхідної зв'язності. Параметр CONN може приймати будь-яке з нижче значень наведених у таблиці 1.1.

Функція bwlabel працює тільки з двовимірними вихідними зображеннями. Для обробки багатовимірних зображень слід використовувати функцію bwlabeln. Функції bwlabeln, відрізняється більш високою швидкодією. Якщо об'єкти вихідного двовимірного зображення є відносно великими по вертикалі, в цьому випадку рекомендується використовувати функцію bwlabel, в інших випадках, з метою збільшення швидкодії, слід використовувати функцію bwlabeln.

Вимоги до вихідних даних.

BW1 може бути логічним або цифровим масивом будь-якої розмірності, також він повинен бути не розрідженим. Матриця міток L має формат представлення даних double.

1.3.5 Функція BWDIST

Ця функція обчислює відстаней
Синтаксис:
D = bwdist (BW)
[D, L] = bwdist (BW)
[D, L] = bwdist (BW, METHOD)

Функція D = bwdist (BW) здійснює обчислення евклідових відстаней на бінарному зображенні BW. Для кожного пікселя на зображенні BW результат обчислень визначається числом, яке є відстанню між поточним пикселом і найближчим ненульовим пикселом зображення BW. За замовчуванням функція bwdist використовує метрику евклідового простору. Зображення BW може мати довільну розмірність. Розмірність масиву D збігається з розмірністю масиву BW.

Функція [D, L] = bwdist (BW) також здійснює обчислення для усіх пікселів і повертає результат в матрицю міток L, розмірність якої збігається з розмірністю масивів BW і D. Кожен елемент матриці L містить в собі лінійні індекси найближчих ненульових пікселів BW.

Функція [D, L] = bwdist (BW, METHOD) здійснює обчислення відстаней, де параметр METHOD визначає змінну метричну дистанцію. Параметр METHOD може набувати значень, які наведені у таблиці 1.2:

Таблиця 1,2

'Chessboard	У двовимірному вимірі, відстань типу "шахівниці" (chessboard) між пікселями (x1, y1) і (x2, y2) визначається як максимальна сума видстаней по вісям
'Cityblock'	У двовимірному вимірі, відстань типу "міських кварталів" (cityblock) між пікселями (x1, y1) і (x2, y2) визначається як сума видмтаней по вісям
'Euclidean'	У двовимірному вимірі, евклідова відстань між пікселями (x1,y1) та (x2,y2) визначається по теоремі Піфагора
'Quasi-euclidean'	У двовимірному вимірі, квазіевклідова відстань між пікселями (x1, y1) і (x2, y2).

Параметр METHOD може не вказуватися.

Функція bwdist використовує стійкий алгоритм обчислення евклідової відстані, який визначений для двовимірних масивів. Інші підходи базуються на обґрунтованих аналогіях цього методу. Однак, для альтернативних обчислень відстаней іноді використовують прискорювачі обробки багатовимірних зображень, особливо якщо вони (зображення) містять багато ненульових елементів.

Вимоги до вихідних даних.

- Початкове зображення BW має бути числовим або логічним нерозрідженним масивом.
- D і L дублюються матриці, розмірність яких дорівнює розмірності BW.

1.3.6 Функція BWFILL

Ця функція заповнює областей фону.

Синтаксис:

BWD = bwfill (BWS, c, г, n)

BWD = bwfill (BWS, n)
[BWD, IDX] = bwfill (...)
BWD = bwfill (x, y, BWS, XI, уг, n)
BWD = bwfill (BWS, 'дірки', n)
[BWD, IDX] = bwfill (BWS, 'дірки', n)

Функція BWd = bwfill (BWs, c, r, n) створює бінарне зображення BWd, що відрізняється від зображення BWs зв'язаноюий областю фону, що включає піксель з координатами (r, c), яка в BWd буде заповнена пікселями об'єкта. Піксель, з якого починається заповнення, називають затравочним. Якщо параметри r і c є векторами однакової довжини, то заповнюються всі області фону з затравочного пікселями (r(k), c(k)).

Параметр n для всіх розглянутих функцій bwfill задає критерій зв'язності для пікселів об'єкта. Цей параметр може приймати значення 4 або 8 (значення за замовчуванням). Слід пам'ятати, що критерій зв'язності для пікселів фону протилежний критерієм зв'язності для пікселів об'єкта, тобто якщо об'єкти 8-зв'язкові, то фон 4-зв'язний, і навпаки.

Функція BWd = bwfill (BWs, n) виводить зображення BWs на екран і надає користувачеві можливість інтерактивно задати координати затравочних пікселів. Бінарне зображення BWd створюється з вихідного зображення BWs, у якого заповнені зв'язкові області фону, що включають затравочние пікселі. Координати затравочних пікселів задаються одноразовим клацанням лівої кнопки миші. Попередній заданий затравочний піксель можна видалити, якщо натиснути клавішу Backspace або Delete. Останній затравочний піксель задається подвійним клацанням лівої кнопки миші або одноразовим клацанням правої клавіші миші. Натискання клавіші Enter завершує процес виділення затравочних пікселів без додавання ще одного затравочного пікселя. Відразу після натискання клавіші Enter або вказівки останньої затравочной точки починається заповнення відповідних пікселів фону.

Якщо при виконанні функції параметр BWs опущений, то функція використовує зображення з поточного вікна.

Функція BWd = bwfill (x, y, BWS, xi, yi, n) створює нове бінарне зображення BWd з вихідного зображення BWs, у якого зв'язанава область фону, що включає пікселі з координатами з векторів xi і yi, заданими в просторовій системі координат, заповнена пікселями об'єкта. Діапазони зміни координат просторової системи координат встановлюються для зображення BWd двокомпонентними векторами x і y.

Функція [x, y, BWd, idx, xi, yi] = bwfill (...) додатково повертає вектори координат xi і yi, відповідних координатах заповнених пікселів в просторовій системі координат. Діапазони зміни координат просторової системи координат, встановлені для вихідного зображення, повертаються в двокомпонентних векторах x і y.

Функція BWd = bwfill (BWs, 'holes', n) створює нове бінарне зображення BWd, у якого пікселями об'єкта заповнені дірки в об'єктах вихідного зображення BWd.

Якщо функція bwfill використовується без вихідних параметрів, наприклад bwfill (BWs, 'holes'), то вихідне зображення відображається на екрані в новому вікні.

1.3.7 Функція BWSELECT

Ця функція виділяє об'єкти.
Снтаксис:
BWd = bwselect (BWs, з, r, n)
BWd = bwselect (BWs, n)
[BWd, idx] = bwselect (...)
BWd = bwselect (x, y, BWs, xi, yi, n)

Функція BWd = bwselect (BWs, з, r, n) створює нове бінарне зображення BWd, що містить об'єкт вихідного зображення BWs, який включає в себе піксель з координатами (r, c). Піксель, з якого починається виділення об'єкта, називають затравочним.

Параметр n для всіх розглянутих функцій bwselect задає критерій зв'язності для пікселів об'єкта. Цей параметр може приймати значення 4 або 8 (значення за замовчуванням).

Функція BWd = bwselect (BWs, n) виводить зображення BWs на екран і надає користувачеві можливість інтерактивно відзначити затравочние пікселі. Всі об'єкти, які містять хоча б один із зазначених пікселів, переносяться з зображення BWs на зображення BWd. Координати затравочних пікселів задаються одноразовим клацанням лівої кнопки миші. Попередній заданий затравочний піксель можна видалити, якщо натиснути клавішу Backspace або Delete. Останній затравочний піксель задається подвійним клацанням лівої кнопки миші або одноразовим клацанням правої клавіші миші. Натискання клавіші Enter завершує процес виділення затравочних пікселів без додавання ще одного затравочного пікселя. Відразу після натискання клавіші Enter або вибрати останню затравочного пікселя створюється нове бінарне зображення BWd.

Функція BWd = bwselect (x, y, BWs, xi, yi, n) створює нове бінарне зображення BWd, що містить ті об'єкти вихідного зображення BWs, які включають в себе затравочние пікселі з координатами, переданими в векторах xi і yi. Координати в xi і yi вказуються в просторовій системі координат. Діапазони зміни координат просторової системи координат встановлюються для зображення BWd двокомпонентними векторами x і y.

Функція [x, y, BWd, idx, xi, yi] = bwselect (...) додатково повертає вектори координат xi і yi, відповідних координатах центрів пікселів

об'єктів з зображення BWd. Діапазони зміни координат просторової системи координат, встановлені для вихідного зображення, повертаються в двокомпонентних векторах x і y.

Якщо функція bwselect використовується без вихідних параметрів, наприклад bwselect (BWs, 10,30), то вийшло в результаті зображення відображається на екрані в новому вікні.

Приклад:

Припустимо, на бінарному зображенні, що містить 6 об'єктів, потрібно залишити тільки два з них. Це робиться шляхом передачі в функцію bwselect координат двох пікселів, кожен з яких належить одному з вибраних об'єктів.

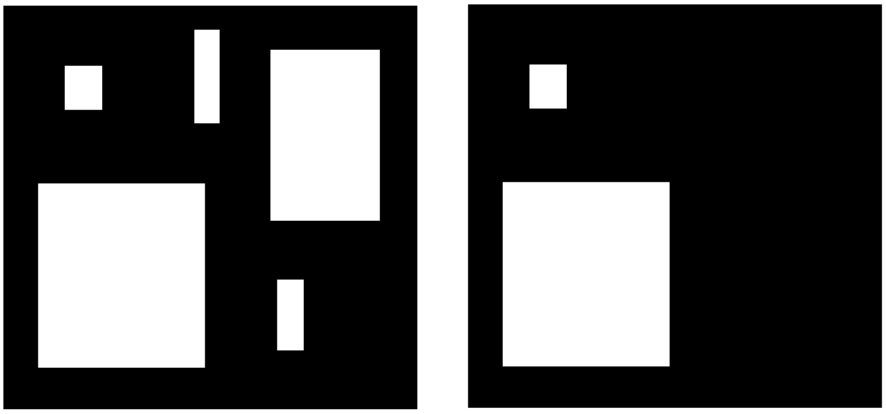


Рисунок 1.4 а)початкове зображення б) результат діїі функціі BWSELECT.

Синтаксис.

```
BW = imread ( 'circles.bmp');
imshow (BW);
% Вибір двох об'єктів.
BW = bwselect (BW, [30 30], [51 150], 8);
% Висновок результату на екран.
figure, imshow (BW)
```

1.3.8 Функція BWPERIM

Виділення меж бінарних об'єктів

синтаксис:

```
BWd = bwperim (BWs, n)
```

Функція BWd = bwperim (BWs, n) створює бінарне зображення BWd, у якого значення граничних пікселів об'єктів вихідного зображення налаштовані в 1, а значення усіх інших пікселів результуючого зображення налаштовані в 0. Піксель вважається граничним, якщо він дорівнює 1 і хоча б один з його n сусідів дорівнює 0. параметр n для всіх розглянутих

функцій bwperim задає критерій зв'язності для пікселів кордону і може приймати значення 4 або 8. Якщо при виконанні функції bwperim параметр n опущений, то перевіряються на рівність нулю всі 8 сусідів пікселя.

Ця функція не надто різниться від виклику функції bwmorph (BW, 'remove', Inf).

1.3.9 Функція BWMORPH

Ця функція здійснює морфологічні операції над бінарним зображенням

синтаксис:

BWd = bwmorph (BWs, operation)

BWd = bwmorph (BWs, operation, n)

Функція BWd=bwmorph (BWd, operation) створює бінарне зображення BWd, піддаючи обробці морфологічним фільтром вихідне бінарне зображення BWs. Тип використовуваного морфологічного фільтра визначається параметром operation. Морфологічні операції представляють собою нелінійний спосіб обробки зображень. Їх застосовують з метою зміни форми об'єктів. Розглядаючи морфологічні операції, будемо називати зв'язані області, значення пікселів яких дорівнюють одиниці, об'єктами, а області зображення, значення пікселів яких дорівнюють нулю, фоном.

Таблиця1.3. Значення параметру operation.

"Erode"	Ерозія об'єкта. Призводить до заміни значень граничних пікселів об'єкта на 0. Одноразове застосування ерозії призводить до видалення шару кордону товщиною в 1 піксель.
"Dilate"	Нарощення об'єкта. Призводить до заміни значень пікселів фону, що межують з об'єктом, на 1. Одноразове застосування нарощення призводить до додавання до об'єкту шару завтовшки в 1 піксель
"Open"	Відкриття. Являє собою послідовне застосування ерозії і нарощення. Призводить до з'єднання областей фону, раніше роз'єднаних вузькими ділянками пікселів об'єктів.
"Close"	Закриття. Являє собою послідовне застосування нарощення і ерозії. Призводить до видалення невеликих за площею фрагментів фону у середені об'єктів, наприклад "дірок" (замкнутих областей фону усіредині об'єкта).
"Tophat"	Перетворення типу "верх капелюха". Відповідає відніманню з вхідного зображення результату його відкриття.

"Bothat"	Перетворення типу "низ капелюхи". Відповідає відніманню вхідного зображення з результату його закриття.
"Clean"	Видалення ізольованих пікселів об'єктів. Пікселі рівні 1, усі 8 сусідів яких дорівнюють 0, замінюються на 0.
"Fill"	Заповнення ізольованих пікселів фону. Пікселі, рівні 0, всі 8 сусідів яких дорівнюють 1, замінюються на 1.
"Diag"	Знищення 8-зв'язності фону. Здійснюється додаванням необхідної кількості одиниць у фрагменти об'єктів, пов'язаних тільки по діагоналі.
"Bridge"	З'єднання пікселів об'єкта, роз'єднаних фрагментом фону товщиною в 1 піксель.
"Hbreak"	Видалення центрального пікселя в конфігураціях, схожих на букву "H":
"Remove"	Видалення внутрішніх пікселів об'єктів. Для цього в 0 встановлюються пікселі об'єкта, у яких 4 сусідніх по горизонталі і вертикалі пікселя були рівні 1, тобто теж були пікселями об'єкта. В результаті застосування цього морфологічного фільтра не скинутими залишаться тільки пікселі кордону об'єкта.
"Majority"	Ерозія і нарощення по переважанню в околиці пікселів фону або об'єкта. Здійснюється в такий спосіб: якщо в околиці пікселя розміру 3x3 знаходиться 5 або більше пікселів об'єкта, то розглянутий піксель встановлюється в 1, в іншому випадку - в 0.
"Skel"	Побудова кістяка (скелета) об'єкта. Операція виконує ерозію об'єкта з урахуванням ряду умов для збереження 8-зв'язності остова. В результаті послідовного застосування даної операції можна побудувати остов, який представляє собою зв'язну лінію товщиною в 1 піксель, що проходить по середині об'єкта (рис. 1в).
"Shrink"	Стиснення об'єкта. Операція виконує ерозію об'єкта з урахуванням ряду умов для збереження 8-зв'язності замкнутих ділянок кістяка. В результаті послідовного застосування даної операції об'єкти, що не містять дірок, перетворюються в точку, а об'єкти з дірками "стискаються" в 8-зв'язкові замкнуті ділянки остова, що проходять посередині і по зовнішньому кордоні об'єкта (рис. 1г).
"Thin"	Утоньшение об'єкта. Операція виконує ерозію об'єкта з урахуванням ряду умов для збереження 8-зв'язності ділянок кістяка. В результаті послідовного застосування даної операції об'єкти, що не містять дірок, перетворюються в одну або кілька зв'язкових ліній з мінімальною кількістю розгалуженні остова в порівнянні з

	остовом, що отримуються за допомогою оператора 'skel '', а об'єкти з дірками стискаються в 8-зв'язкові замкнуті ділянки кістяка, проходять посередині між кордонами дірок і зовнішнім кордоном об'єкта (рис. 1д).
"Thicken"	Потовщення об'єкта. Операція виконує нарощення об'єкта з урахуванням ряду умов для збереження 4-зв'язності ділянок фону. Дана операція може розглядатися як побудова кістяка фону і є за змістом зворотньою операції 'thin'. Результат даної операції буде незначно відрізнятися від результатів застосування оператора 'thin', якщо після її застосування інвертувати зображення.
"Spur"	Видалення відгалужень об'єкта товщиною в 1 піксель, тобто видалення пікселів, у яких тільки один з сусідніх пікселів встановлений в 1, а решта - в 0

.

1.3.10 Функція REGIONPROPS

Синтаксис:
STATS = regionprops (L, properties)

Функція STATS = regionprops (L, properties) вимірює набір характеристик для кожної області, зазначеної в матриці міток L. Позитивні елементи L відповідають різним областям. Наприклад, номер елементів L рівних 1 відповідає 1-й області, номер 2 відповідає 2-й області т.д. Результуюча змінна STATS має структуру у вигляді масиву длінойmax (L (:)). Поля структурного масиву позначають різні значення параметрів для кожного регіона (області) і визначають їх характеристики.

Характерістікі можут перераховуватися у вигляді рядка, осередок масиву містить рядок одиниць, рядок 'all' або 'basic'. У таблиці наведено список всіх строкових характерістік.

' *Таблиця1.4.* Характеристики properties.

'Area	'Extent	'PixelList
'EquivDiameter	Orientation	'ConvexImage
MajorAxisLength	ConvexArea'	FilledImage
Mino rAxisLength'	'Extrema	'Solidity
EulerNumber	'PixelIdxList	'Eccentricity
BoundingBox	'ConvexHull	'Image
Centroid	'FilledArea'	SubarrayIdx'

Коли характерістікі представлені рядком 'all', тоді обчислюються всі перечісленниевише вимірювання.

Коли характерістікі не визначені або визначені опцією 'basic', тоді проводятсятакіе вимірювання: 'Area', 'Centroid' і 'BoundingBox'.

Нваведем деякі визначення перерахованих вище параметрів.

'Area' - скаляр; дійсне число пікселів у зазначеній області. (Це значення можетнемного відрізнятися від значення, що видається функцією bwarea.)

'Centroid' - векторс розмірністю 1ndims (L); центр мас області. Відзначимо, що першим елементом характеристики Centroidявляется горизонтальна координата (або x-координата) центру мас, а вторимелементом є вертикальна координата (або y-координата).

'BoundingBox' вектор 1ndims (L) * 2; мінімальний прямокутник, що вміщає розглянуту локальну область.BoundingBox є вектором [ul_corner width], де ul_corner представляється вформе [x y z ...] і описує верхній-лівий кут огранічівающегопрямоугольніка; width представляється в формі [x_width y_width ...] і определяетшіріну уздовж кожної розмірності.

Малюнок внізуіллюстрірует центр мас і обмежує прямокутник. Область складається ізбелих пікселів, зеленим відзначений обмежує прямокутник і червоним отмеченцентр мас.

'MajorAxisLength'- скаляр; довжина (в пікселях) великий осі еліпса, який має той же другий моментчто і розглянута область. Ці властивості підтримуються тільки для двумернихісходних матриць міток.

'MinorAxisLength'- скаляр; довжина (в пікселях) меншої осі еліпса, який має той же другий моментчто і розглянута область. Ці властивості підтримуються тільки для двумернихісходних матриць міток.

'Eccentricity' -скаляр; ексцентриситет еліпса з головними моментами інерції, рівними главниммоментам інерції об'єкта. Ексцентриситет - це відношення відстаней междуфокусом і великою віссю еліпса. Це значення в діапазоні від 0 до 1. (0 і 1 длявирожденних випадків; коли ексцентриситет еліпса 0, тоді це коло, коли жеексцентрісітет еліпса 1, тоді це частина лінії.) Ці властивості поддержіваютсятолько для двовимірних вихідних матриць міток.

'Orientation' -скаляр; кут (положення) між максимальною віссю еліпса і x-віссю. Ці свойстваподдержіваются тільки для двовимірних вихідних матриць міток.

Малюнок внізу демонструє осі і орієнтацію еліпса. Ліва сторона фігури показує наізображеніі область і відповідний еліпс. Права сторона показує той жеелліпс з характеристиками, які відображені графічно; суцільна синя лініяобозначает осі, червоними плямами відзначені фокуси і орієнтація визначається какугол між горизонтальною пунктирною лінією і головною віссю.

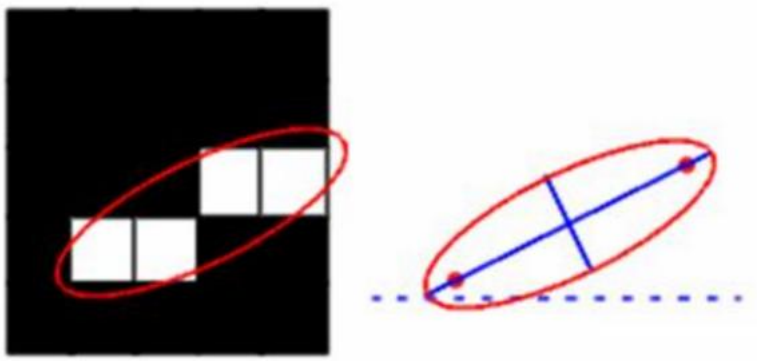


Рисунок 1.5 а)початкове зображення б) метод вимірювання кутанахилу за допомогою параметру «Orientation».

'Image' -бінарне (логічне) зображення, розміри якого збігаються з розмірамі прямокутніка, що обмежує область; всі пікселі поза регіоном видаляються.

'FilledImage' -бінарное (логічне) зображення, розміри якого збігаються з розмірами прямокутника, що обмежує область (зображення об'єкта з "залитими" дірками). Всі отвори в даній області заповнюються.

'FilledArea' -скаляр; повне число пікселів в FilledImage.

'ConvexHull' матриця з розмірністю p2; мінімальний опуклий багатокутник, який вміщує розглянуту область. Кожен рядок матриці містить x- та y-координати однієї вершини багатокутника. Еті властивості підтримуються тільки для двовимірних вихідних матриць міток.

'ConvexImage' -бінарное (логічне) зображення; багатокутника включно з усіма наповнюючим його пікселями. Зображення являє собою область, обмежену прямокутником. Ці властивості підтримуються тільки для двумерной матриці міток:

1.4 Завданння до виконання роботи

1. Ознайомитися з теоретичними відомостями, розглянути надані бібліотекою Image Processing Toolbox методи обробки бінарних, напівтонових, палітрових, повнокольорових зображень у розділі допомоги до пакету Matlab.

2. За допомогою будь-якого графічного редактора створити бінарне растрове зображення розміром 40x40 пікселів, що містить геометричні фігури (кола, прямокутники, трикутники) з накладенням і розміщенням один в одному.

3. Провести аналіз та обробку бінарного зображення в середовищі MatLab із застосуванням функцій бібліотеки Image Processing Toolbox: bwarea, bwareaopen, bwdist, bweuler, bwfill, bwlabel, bwlabeln, bwmorph, bwperim, bwselect, bwulterode, а так само додаткові imfeature, imread, imshow .

4. Привести опис та код програми використання по кожній з перерахованих функцій, Вивести створене зображення до та після застосування програми. Якщо результат роботи функції не можливо уявити у вигляді графічного зображення привести матрицю або числовий результат.

1.5 Вимоги до звіту

Звіт повинен містити:

- Титульну сторінку з даними про виконавця і перевіряючого.
- Порядковий номер, номер варіанта, тему і мету роботи.
- Опис кожної з перелічених у пункті 3 функції та, код програми її використання.
- Збільшене в 6 разів створене зображення до та після застосування програми (якщо результат роботи функції не можливо уявити у вигляді графічного зображення привести матрицю або числовий результат).
- Висновки по суті виконання роботи.

Звіт повинен бути оформлений відповідно до вимог СОКР.

2 Лабораторна робота №2

РЕАЛІЗАЦІЯ ФУНКЦІЙ ОБРОБКИ ГРАФІЧНИХ ЗОБРАЖЕНЬ

Мета: самостійно розробити алгоритм і провести його реалізацію, із застосуванням стандартних функцій середовища Matlab, без використання спеціалізованих методів пакету Image Processing Toolbox.

2.1 Теоретичні відомості

2.1.1 Складні об”єкти

Інколи в поле зору потрапляє більше одного об'єкта (рис. 2.1). У цьому випадку обчислення площі, геометричного центру та орієнтації призведе до значень, "усереднених" за всіма компонентами бінарного зображення. Як правило, це не те, що потрібно. Бажано якось помітити окремі компоненти зображення і обчислити значення площі, перших і других моментів для кожної компоненти окремо.

2.1.2 Розмітка компонент

Будемо вважати дві точки зображення пов'язаними, якщо існує шлях між ними, вздовж якого характеристична функція постійна

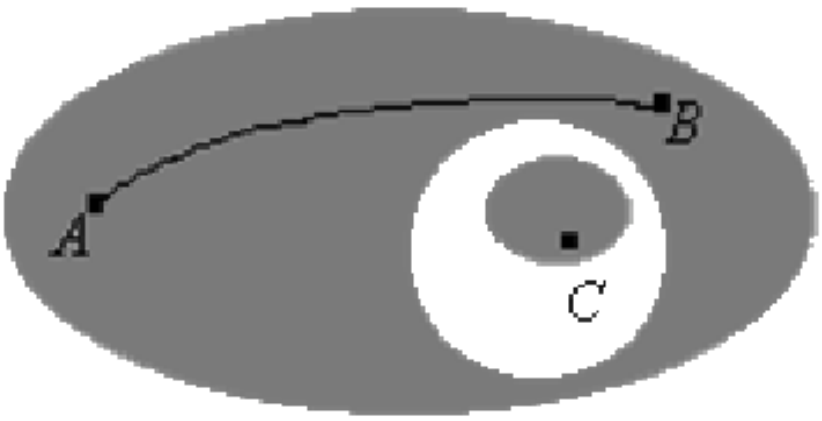


Рисунок 2.1 – Зображення, що складається з декількох областей, для кожної з яких необхідно проводити розрахунок положення і орієнтації .

Так, на рисунку 2.1 точка A пов'язана з точкою B, але не пов'язана з точкою C. Зв'язана компонента бінарного зображення є максимальною множиною пов'язаних точок, тобто множина, що складається з усіх тих точок, між двома з яких існує зв'язуючий їх шлях.

Елементи зображення необхідно помітити таким чином, щоб ті з них, які належать одній області були відмінні від інших. Для цього нам необхідно вирішити, які точки належать одній і тій же області. На рисунку 2.1 точка А вважається пов'язаною з точкою В, оскільки ми можемо знайти безперервну криву, яка цілком лежить в затіненій області та сполучає зазначені точки. Ясно, що точка А не пов'язана з точкою С, так як в цьому випадку не можливо знайти подібної кривої.

Один із способів розмітки об'єктів на дискретному бінарному зображенні полягає у виборі довільної точки, в якій bij = 1. Цій точці та її сусідам приписують певну мітку. На наступному кроці позначаються сусіди цих сусідів (крім уже помічених) і т.д. По завершенні цієї рекурсивної процедури, одна компонента буде повністю позначена, і процес можна буде продовжити, вибравши нову початкову точку та нову мітку для позначки. Щоб її відшукати, досить яким-небудь систематичним чином переміщуватися по зображенню до тих пір, поки не зустрінеться остання непомічена точка, в якій bij = 1. Коли на цьому етапі не залишиться жодного такого елемента, всі об'єкти зображення виявляться розміченими.

Ясно, що "фон" також можна розбити на зв'язкові компоненти, оскільки об'єкти можуть мати отвори. Їх можна помітити за допомогою тієї ж процедури, але при цьому необхідно звертати увагу не на одиниці, а на нулі.

2.1.3 Зв'язаність

Тепер потрібно розглянути зміст терміну сусід. Якщо ми маємо справу з квадратним растром, то, сусідами слід вважати чотири елементи зображення, що торкаються сторін даного елемента. Але як бути з тими, що торкаються його в кутах? Існують дві можливості:

• чотирьохзв'язність - сусідами вважаються тільки елементи, які примикають до сторін;

• восьмизв'язність - елементи, які торкаються в кутах, також вважаються сусідами. Зазначені можливості наведені на рисунку 2.2:

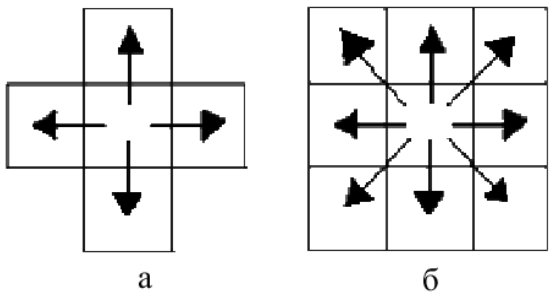


Рисунок 2.2 – Зв'язність елементів: а) чотирьохзв'язність, б) восьми зв'язність.

Виявляється, жодна з цих схем не є повністю задовільною. У цьому можна переконатися, якщо згадати, що фон також можна розбити на кілька зв'язаних компонент. Тут нам хотілося б застосувати наші інтуїтивні уявлення про зв'язність областей на безперервному бінарному зображенні. Так, наприклад, проста замкнута крива повинна розділяти зображення на дві зв'язані області (рис. 2.3). Це так звана теорема Жордана про криву.

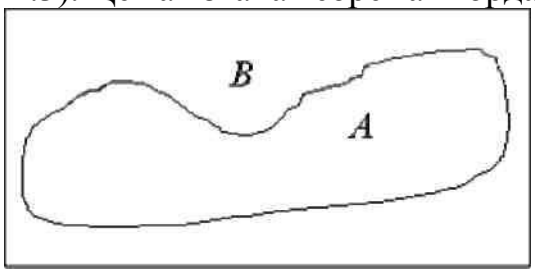


Рисунок 2.3 Проста замкнута крива, розбиває площину на дві зв'язні області.

Тепер розглянемо просте зображення, що містить чотири елементи зі значенням "одиниця", які примикають до центрального елементу зі значенням "нуль":

0 1 0
1 0 1
0 1 0

Це - хрест з викинутим центром. Якщо ухвалити угоду про чотирьохзв'язності, то на зображенні виявляться чотири різні компоненти (О1, О2, О3 і О4):

В1 О1 В1
О2 В2 О3
В1 О4 В1

Вони, природно, не утворюють замкнутої кривої, хоча центральний елемент, що відноситься до фону, і не пов'язаний з рештою фону. Незважаючи на відсутність будь-якої замкнутої кривої, у нас утворилися дві фонові області. Якщо ухвалити угоду про восьмизв‘язність, то, навпаки, чотири елементи растра будуть утворювати замкнуту криву, і в той же час центральний елемент виявиться пов'язаним з рештою фоном:

В О В
О В О
В О В

Отже, ми отримали замкнуту криву і тільки одну зв'язану компоненту фону.

Одне з рішень виниклої проблеми полягає у виборі чотирьохзв'язності для об'єкта і восьмизв'язності для фону (або навпаки). Така асиметрія в трактуванні об'єкта і фону часто небажана, і її можна уникнути шляхом введення іншого типу асиметрії. Будемо вважати

сусідами чотири елементи зображення, що примикають до даного по сторонах, а також два з чотирьох елементів, що торкаються в кутах:

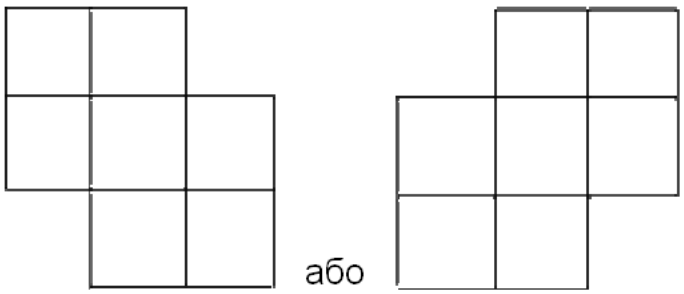


Рисунок 2.4 –Шестизв'язного елементів

Для забезпечення симетричності відносини зв'язності два кутові елемента повинні перебувати на одній і тій же діагоналі: якщо елемент А - сусід елемента В, то елемент В повинен бути сусідом елемента А. Найчастіше користуються першим з двох можливих варіантів, наведених вище. За допомогою шестизв'язності як об'єкт, так і фон можна трактувати одноманітно без будь-яких подальших непогодженостей. Така зв'язність використовується для зображень на квадратному растрі.

2.2 Локальні обчислення

Дотепер основна увага приділялася послідовній обробці інформації, що міститься в бінарному зображенні. Тепер ми розглянемо деякі алгоритми, які можуть використовувати паралельні обчислення.

2.2.1 Число Ейлера

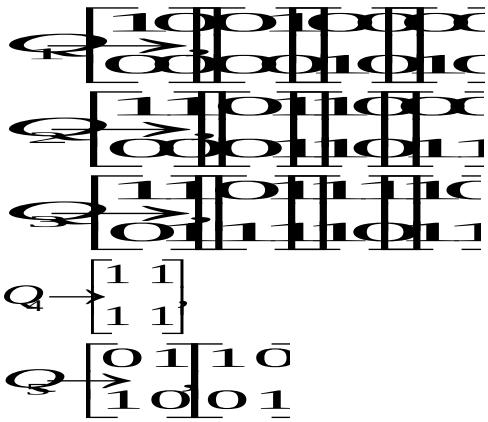
Число Ейлера визначається як різниця між кількістю об'єктів на зображенні та кількістю дірок в цих об'єктах. Параметр n визначає використовуваний критерій зв'язності. Він може бути рівним 4 або 8. При виконанні функції bweuler параметр n можна опустити, в цьому випадку об'єкти розглядаються як 8-зв'язкові.

Алгоритм:

Ідея алгоритму полягає у підрахунку кількості квадратних матриць 2x2 з різною кількостю ненульових елементів. Потім за допомогою формул, отриманих емпіричним шляхом підрахуємо число Ейлера.

Для спрощення підрахунків будемо використовувати наступний підхід. Введемо наступні позначення:

$$Q \rightarrow \begin{bmatrix} O & O \\ O & O \end{bmatrix}$$



Тоді для 4-зв'язаних об'єктів число Ейлера визначається за формулою з [1]:

$$E_u = \frac{1}{4}\left[n(Q_1) - n(Q_3) + 2n(Q_5)\right] \qquad (2.1)$$

а для 8-зв'язкових - за формулою

$$E_u = \frac{1}{4}\left[n(Q_1) - n(Q_3) - 2n(Q_5)\right] \qquad (2.2)$$

2.2.2 Обчислення периметру

Обчислення периметру являє собою лише приблизну оцінку, оскільки, як правило, дискретне бінарне зображення будується на основі безперервного, і при цьому межі об'єктів стають більш порізаними. Наприклад, оцінка довжини діагональної прямої в 2 разів більше "істинної":

Усереднення по всіх кутках нахилу дає середнє значення коефіцієнта, що показує, у скільки разів збільшено отримане значення. Воно складає 4 / π = 1,273 .... Розділивши на це число, можна поліпшити оцінку периметра.

2.2.3 Побудова остова (скелета) об'єкта

Операція побудова остова (скелета) об'єкта виконуєця із застосовуанням ерозії об'єкта з урахуванням ряду умов для збереження 8-зв'язності остова. У результаті послідовного застосування даної операції можна побудувати остов, що представляє собою зв'язну лінію товщиною в 1 піксель, що проходить по середині об'єкта:/

Бінарні зображення можна комбінувати різними шляхами. Можна здійснити операцію АБО. У результаті ми об'єднаємо два зображення в одне. Великий інтерес представляє те, як характеристики одержуваних подібними способами зображень співвідносяться з характеристиками вихідних зображень. Одна з причин такого інтересу пов'язана з надією

розбити зображення на велику кількість частин, одночасно обробити всі ці частини і потім об'єднати результат.

2.3 Завдання до виконання роботи

1. Ознайомитися з завданням, відповідно до вашого варіантом.
2. Провести аналіз поставленого завдання. Розробити блок схему реалізації поставленого завдання.
3. Реалізувати поставлену задачу у вигляді функції.
4. Привести приклад роботи реалізованої функції, пояснити отримані результати.

2.4 Варіанти завдань до виконання роботи

У всіх варіантах потрібно за замовчуванням приймати 4-х зв'язність елементів зображення або враховувати значення зв'язаності, введене при виклику функції.

1. Видалити всі об'єкти, периметр яких менше заданої кількості пікселів.
2. Розділити на n груп об'єкти зображення по площі, шляхом маркування, об'єктів з рівною площею однаковим індексом.
3. Побудувати скелет об'єктів зображення.
4. Побудувати контур об'єктів зображення.
5. Реалізувати функцію замикання розімкнутих кривих зображення.
6. Залити всі замкнуті елементи зображення кольором об'єкта.
7. Видалити (залишити) всі об'єкти зображення, які знаходяться усередині замкнутого об'єкта.
8. Реалізувати функцію розрахунку числа Ейлера.
9. Провести маркування всіх об'єктів зображення. Не зв'язані об'єкти маркувати різним, зорово сприйманим кольором.
10. Визначити діаметр кіл, розміщених на зображенні.
11. Визначити периметр об'єктів, розміщених на зображенні.
12. Реалізувати функцію вибору об'єкта по вказаній площі.
13. Реалізувати функцію вибору об'єкта по вказаній мітці.
14. Видалити всі об'єкти, площа яких менше заданої кількості пікселів.

2.5 Вимоги до звіту

Звіт повинен містити:

- Титульну сторінку з даними про виконавця і перевіряючого.
- Порядковий номер, номер варіанта.
- Тема і мета лабораторної роботи.
- Блок схема реалізованого завдання.
- Лістинг реалізованої функції.
- Результати застосування реалізованої функції до тестового зображення.
- Висновки про виконання роботи.

Звіт повинен бути оформлений відповідно до вимог СОКР

3 Лабораторна робота №3

ІДЕНТИФІКАЦІЯ ОБ’ЄКТІВ НА ЗОБРАЖЕННІ

Мета: навчитися розробляти і реалізовувати алгоритми ідентифікації об'єктів графічних зображень.

3.1 Теоретичні відомості

Під ідентифікацією об'єктів зображення розуміють, як визначення їх геометричних властивостей, так і віднесення об'єктів до якої-небудь, заздалегідь визначеної групи об'єктів.

У цій лабораторній роботі, необхідно навчитися виробляти виділення первинних ознак об'єктів графічного зображення з метою подальшої класифікації об'єктів в рамках певної предметної області.

Нижче розглянемо кілька прийомів, які дозволяють виміряти значення ознак об'єктів зображення. Алгоритми наводяться лише для ознайомлення.

3.1.1 Ідентифікація круглих об'єктів

Розглянемо задачу, основною метою якої буде ідентифікація округлих об'єктів.

Крок 1: Зчитування зображення.

Крок 2: Порогова обробка зображення.

Крок 3: Усунення шуму.

Крок 4: Пошук меж об'єктів зображення.

Крок 5: Визначення округлості об'єктів.

Розглянемо наведені кроки детальніше.

Крок 1: Зчитування зображення.

Завантажуємо зображення з файлу pills_etc.png.

RGB=imread('pillsetc.png'); imshow(RGB);

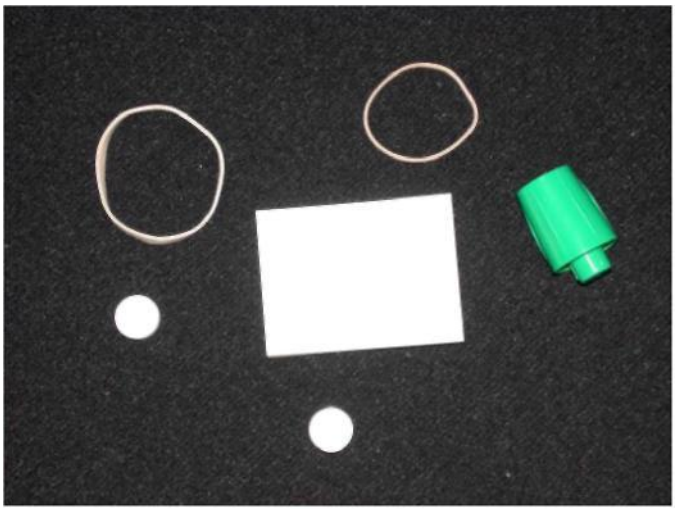


Рисунок 3.1 – Початкове зображення

Крок 2: Порогова обробка зображення.

Перетворимо, досліджуване зображення, в бінарне і таким чином підготуємо його для застосування функції bwboundaries, яка реалізує виділення меж об'єктів.

```
I=rgb2gray(RGB);
threshold=graythresh(I);
bw=im2bw(I, threshold);
imshow(bw)
```

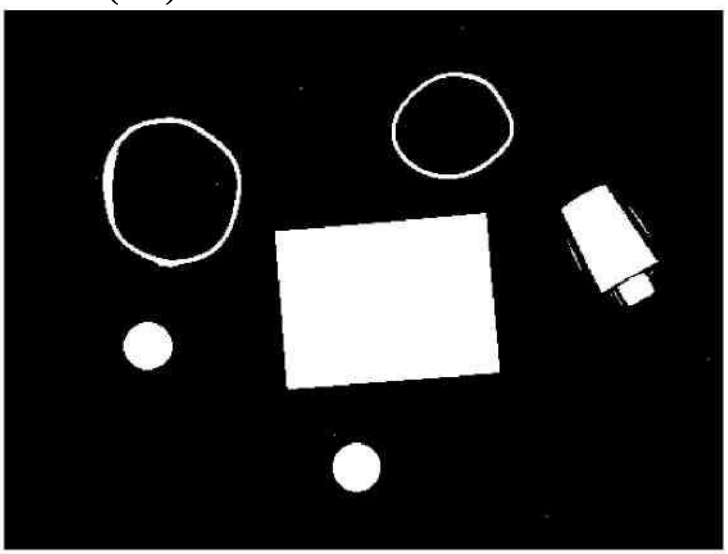


Рисунок 3.2 – Порогова обробка зображень

Крок 3: Усунення шуму.

З використанням морфологічних функцій усунемо об'єкти, які не є об'єктами нашого інтересу, тобто шум.

```
% Видалення всіх об'єктів, що містять менше ніж 30 пікселів
bw=bwareaopen(bw,30);
% заполнение пустот
se=strel('disk', 2);
bw=imclose(bw, se);
bw=imfill(bw,'holes');
```

imshow(bw)

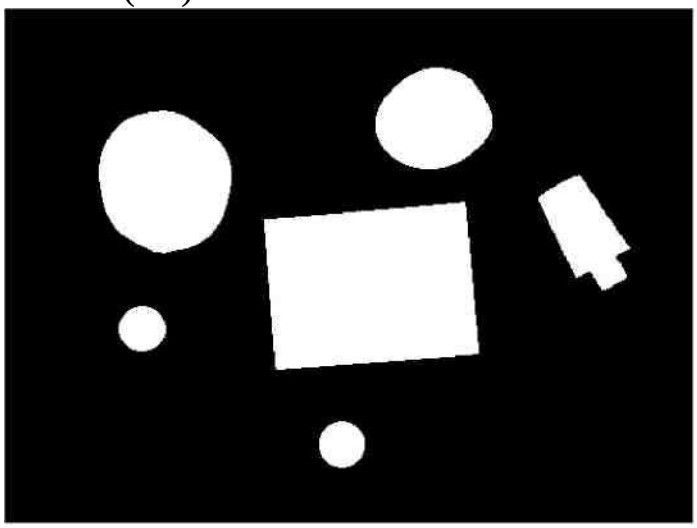


Рисунок 3.3 – усунення шуму

Крок 4: Пошук меж об'єктів зображення.

При вирішенні цього питання будемо розглядати тільки зовнішні кордони.

Опція 'noholes' призводить до прискорення обробки за допомогою функції bwboundaries і підвищує достовірність виділення меж об'єктів зображення.

```
[B, L] = bwboundaries (bw, 'noholes');
% Відображення матриці міток і витяг кордонів
imshow(label2rgb(L, @jet, [.5 .5 .5]))
hold on
for k=1:length(B) boundary=B{k}; plot(boundary(:, 2), boundary(:, 1), 'w', 'LineWidth', 2)
end
```

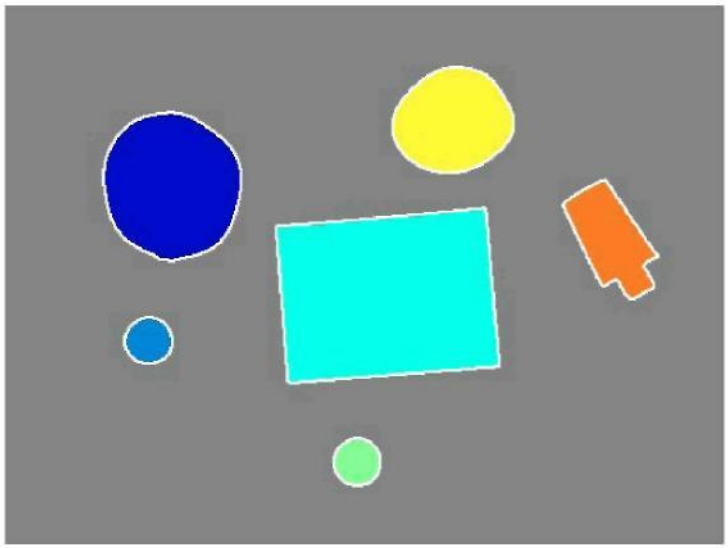


Рисунок 3.4–Пошук меж об'єктів зображення

Крок 5: Визначення округлості об'єктів.

Вимірюємо площу кожного об'єкта та його периметр. Ці дані будуть використані при обчисленні характеристики округлостей об'єктів:

metric = 4 * pi * area / perimeter ^ 2.

Параметр metric є достовірним тільки у випадку ідеальної окружності і є меншим при інших формах досліджуваного об'єкта. Крім того, при вирішенні завдань подібного роду, можна контролювати установку порога бінарізації, що призведе до поліпшення якості обробки. При обробці досліджуваних зображень використовувався поріг, рівний 0.94.

Використовуємо функцію regionprops для отримання вимірювань площі для всіх об'єктів. Відзначимо, що матриця міток, створена функцією bwboundaries, може повторно використовуватися у функції regionprops.

```
stats=regionprops(L, 'Area', 'Centroid');
threshold=0.94;
% Окружність кордонів
for k = 1: length (B)
% Отримання координат меж (X, Y), відповідних мітці 'k'
boundary = B {k};
% Обчислення вимірювань на основі периметра об'єктів
delta_sq = diff (boundary). ^ 2;
perimeter = sum (sqrt (sum (delta_sq, 2)));
% Отримання обчисленої площі, відповідної мітці 'k'
area = stats (k) .Area;
% Обчислення характеристики округлості metric
metric = 4 * pi * area / perimeter ^ 2;
% Відображення результатів
metric_string = sprintf ('% 2.2f', metric);
% Маркування об'єктів
if metric>threshold centroid=stats(k).Centroid; plot(centroid(1), centroid(2), 'ko');
end
text(boundary(1, 2)-35, boundary(1, 1)+13, metric_string,'Color', 'y',... 'FontSize', 14, 'FontWeight', 'bold'); end title(['Metrics closer to 1 indicate that ',.'the object is approximately round']);
```

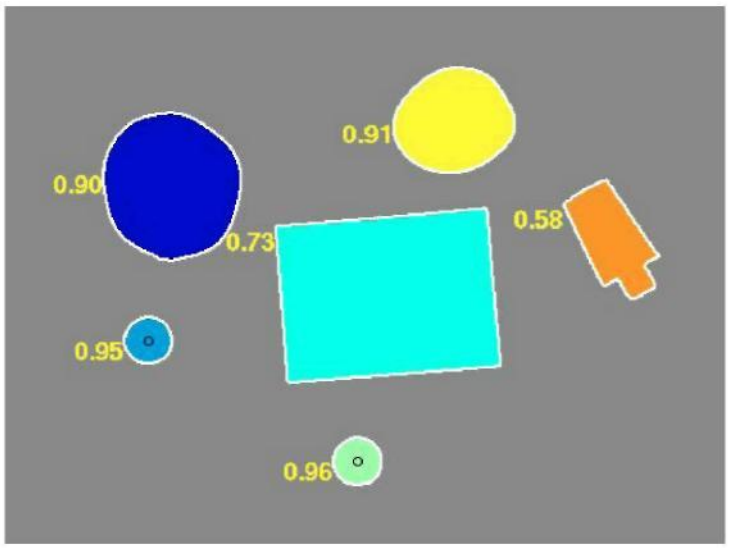


Рисунок 3.5 Визначення округлості об’ектsв

3.1.2 Вимірювання кутів перетину

Часто в машинному зорі застосовуються автоматичні вимірювання різних геометричних параметрів із застосуванням технології захоплення і обробки зображень. Вимірювання кута і визначення точки перетину двох елементів деякої конструкції виконується з використанням функції bwtraceboundary, яка реалізує також автоматичне відстеження кордонів.

Дії:

Крок 1: Зчитування зображення.

Крок 2: Визначення області інтересу.

Крок 3: Порогова обробка зображення.

Крок 4: Пошук початкових точок кожного кордону.

Крок 5: Відстеження кордонів.

Крок 6: Підгонка ліній кордонів.

Крок 7: Визначення кута перетину.

Крок 8: Пошук точки перетину.

Крок 9: Відображення результатів.

3.1.3 Віокремлення об‘єктів по кількості отворів

Часто треба розрізнити об‘экти на зображенні по кількості отворів, або інородних включень. для цього можна застосувати наступній алгоритм

Крок 1: Зчитування зображення.

Крок 2: Для кожного об‘єкту визначити його межі (координати мінімального прямокутника (Р), який він займає) та зкопіювати обасть Р для подальших дій.

Крок 3:. Відалити з області Р усі інші об‘єкти окрім досліджуваного

Крок 4: Підрахувати цисло єйлера для очищеної області Р.

Крок 5:В залежності від значення числп Ейлера прийняти рішення щодо позначення об‘кта символом.

3.1.4 Віокремлення об‘єктів по кількості кутів

Інколи треба розрізнити об‘экти на зображенні по кількості кутів. для цього можна застосувати наступній алгоритм

Крок 1: Зчитування зображення.

Крок 2: Для кожного об‘єкту визначити його граничні пікслі .

Крок 3:. Серед граничних пікселів знай такі, число сусідів, яких буде меньше половини (<4)/ Такі піксеклі й визначають кількість гострих кутів у об‘єкта.

Крок 4: Підрахувати кількість N кутових пікселів.

Крок 5:В залежності від N прийняти рішення щодо позначення об‘кта символом.

Інші варіанти завдань в даній лабораторній роботі потребують або конбінованого застосування наданих алгоритміх, або розробки інших евристичних алгоритмів.

3.2 Завдання до виконання роботи

1. Ознайомитися з завданням, відповідно до вашого варіанту.
2. Провести аналіз поставленого завдання.
3. Реалізувати поставлену задачу.
4. Привести приклад роботи та пояснити отримані результати.

3.3 Варіанти завдань до виконання роботи

№ варианту	Завдання
1	Реалізувати програму, що відрізняэ об‘экти з одним отвором від їнших об‘эктів та позначаэ їх сімволом «+» на зображенні.Тестове зображення створити самостійно.
2	Реалізувати програму, що відрізняэ об‘экти з двома отворами від їнших об‘эктів, підраховує їх кількість, та позначаэ їх сімволом «+» на зображенні.Тестове зображення створити самостійно.
3	Реалізувати програму, що відрізняэ трикутники від їнших об‘эктів та позначаэ їх сімволом «З» на зображенні.Тестове зображення створити самостійно.
4	Реалізувати програму, що відрізняэ чотирикутні об‘экти тільки з

	двома отворами від їнших об‘єктів та позначаэ їх символами «++» на зображенні.Тестове зображення створити самостійно.
5	Реалізувати програму, що відрізняэ прямокутники від їнших об‘єктів та позначаэ їх символом «3» на зображенні .Тестове зображення створити самостійно.
6	Реалізувати програму, що відрізняэ квадрати від їнших об‘єктів та позначаэ їх символом «4» на зображенні.Тестове зображення створити самостійно.
7	Реалізувати програму, що описує кола навколо усіх квадратів на зображенні. .Тестове зображення створити самостійно.
8	Реалізувати програму, що описує кола навколо усіх прямокутніх трикутників на зображенні. .Тестове зображення створити самостійно.
9	Реалізувати програму, що знаходить центри ваг об‘єктів та позначаэ їх символом «+» на зображенні.Тестове зображення створити самостійно.
10	Реалізувати програму, що відрізняэ трикутники з отвором на них від їнших об‘єктів та позначаэ їх символом «0» на зображенні.Тестове зображення створити самостійно.
11	Створити програму, що підраховує кількість монет кожного номіналу на зображенні а також підраховує їх загальну суму. Тестове зображення створити самостійно.
12	Реалізувати програму, що окреслює едентичні об‘єкти на другому зображенні по їх кольоровій гамі, отриманій з первого зображення. Для тесту використовувати 2 сусідні кадра, з однієї відеопослідовності .Тестові зображення створити самостійно.
13	Реалізувати програму, що помічає символами «+» всі кола на зображенні, та підраховує їх кількість. Тестове зображення створити самостійно.
14	Реалізувати програму, що відрізняэ прямокутники та трикутники від їнших об‘єктів, також позначаэ їх символами «+» та «-« та підраховує іх кількість на зображенні. Тестове зображення створити самостійно.
15	Реалізувати програму, що відмаркувати однотипні елементи зображення. Тестове зображення створити самостійно.
16	Реалізувати програму, що розраховує кут між 2-ма об'єктами зображення. Тестове зображення створити самостійно.

3.4 Вимоги до звіту

Звіт повинен містити:

- Титульну сторінку з даними про виконавця і перевіряючого.
- Номер варіанта, тему і мету роботи.
-Лістинг програми.
-Результати застосування створеної функції до тестового зображення.
-Висновки за результатами лабораторної роботи.

Звіт повинен бути оформлений відповідно до вимог СОКР

4 РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА

СЕГМЕНТАЦІЯ ОБ‘ЄКТІВ НА ВІДЕОЗОБРАЖЕННЯХ

Мета: засвоїти принципи створення процеду, призначених для маніпуляції об‘єктами на відеозображеннях.

4.1 Теоретичні відомості

На цей час актуальною залишається тема автоматизованої сегментації, комбінуванні, переносу та трансформації зорових об‘єктів на відеозображеннях, Такі процедури широко застосовуються у кінематографі, у сфері розваг, а також у новітніх методах відстежування, які застосовуються, слубами безпеки. Дана розрахунково-графічна робота присвяченя здійсненю автоматизованого сінтезу штучного зображення, який здійснюється шляхом переносу певного зорового об‘єкта на інший фон.

Як відомо, відеофайл , або відеопоток складається з окремих кадрів. Тож, щоб перемістити певний зоровий об‘єкт з однієї «сцени» на другу необхідно поперше розділити оба початкові відеофрагмента на кадри. Погтім потрібно віокремити об‘єкт, що підлягає переносу, створивши для нього маску. Далі потрібно перенести об‘ект на новий фон, використовучи процедури «зглажування» країв. І нарешті змонтувати послідовність зображень у відеофайл.

4.1.1 Створення контуру об‘єкта градиентним методом

Зазвичай завданням сегментації є поділ зображення на області, що складаються з точок, що мають приблизно однаковий рівень яскравості (в разі напівтонових зображень) або схожі колірні характеристики (в разі кольорових зображень). Відповідно, контурами, або межами, даних областей є сукупність точок на зображенні, в яких відбувається зміна рівня яскравості або кольору. Для визначення меж або контурів областей розроблено багато різних методів, основою для яких, в більшості випадків, є побудова градієнтного зображення (рис 4.1), тобто, застосування до вихідного зображення оператора першої похідної для дискретної функції, визначеної на площині. Оператор градієнта можна розглядати як лінійний фільтр зображення. Лінійні фільтри зображень, як правило, задаються квадратною матрицею коефіцієнтів або маскою фільтра, яка застосовується для кожної точки зображення. Розмір маски оператора першої похідної, або, іншими словами, його масштаб, може бути різним. Часто відповідний масштаб оператора градієнта, який дає в результаті найкращу картину контурів об'єктів, підбирається шляхом експериментальних перевірок. Але

зазвичай зображення, що зустрічаються на практиці, в більшості випадків містять контури з різними швидкостями зміни яскравості (для випадку напівтонових зображень) або кольору (для випадку кольорових зображень), тобто, як різкі, так і плавні. Одже, неможливо найкращим чином однозначно визначити всі існуючі на зображенні кордони, використовуючи лише оператор градієнта одного певного масштабу. Тож у більш складних випадках застосувують методи, що дозволяють побудувати картину контурів об'єктів зображення на основі інформації, одержуваної в результаті застосування оператора градієнта різних масштабів.

Багато методів сегментації, засновані на визначенні контурів об'єктів, наприклад, ватершед-перетворення, використовують в якості основи для проведення сегментації градиентного зображення.

Рисунок 4.1 –Перетворення зображення за допомогою градиентного методу

Нище наведений приклад процедури, яка дає можливість отриматий контури об‘єктік зображень представлених на рисунках 4.1 та 4.2.

```
procedure  GRADMETOD (Sender: TObject);
var
gs,gr,gb,gg,ii,jj,r,g,b,sr,i,j,blu,red,gre,
srsr:integer;
col:Tcolor;
begin // передача параметрів зображення
Image1.Width:=Image.Width;
```

```
Image1.Height:=Image.Height;
Image2.Width:=Image.Width;
Image2.Height:=Image.Height;
Image3.Width:=Image.Width;
Image3.Height:=Image.Height;
Image4.Width:=Image.Width;
Image4.Height:=Image.Height;
Image5.Width:=Image.Width;
Image5.Height:=Image.Height;
StringGrid2.ColCount:=640;
StringGrid2.RowCount:=480;
Progressbar1.Max:=Image.Width+1;
Progressbar2.Max:=Image.Width+1;
For i:=0 to Image.Width-1 do
begin
Progressbar1.Position:=i;
For j:=0 to Image.Height-1 do
  begin
   col:=Image.Canvas.Pixels[i,j];
   cr[i,j]:=ExRed (col); // отримання червоного кольору пікселя
   cb[i,j]:=ExBlu (col);
   cg[i,j]:=ExGre (col);
   StringGrid2.Cells[i,j]:=IntToStr(cr[i,j]);
  end;
end;

i:=2;
 Progressbar2.Visible:=true;
repeat
Progressbar2.Position:=i;
j:=2;
repeat
  gr:=0;
For ii:=-1 to 2 do
For jj:=-1 to 2 do
  Begin  // отримання модуля градієнт-вектора
  gr:=gr+abs(cr[i+ii,j+jj]-cr[i+ii-1,j+jj]);
  gb:=gb+abs(cb[i+ii,j+jj]-cb[i+ii-1,j+jj]);
  gg:=gg+abs(cg[i+ii,j+jj]-cg[i+ii-1,j+jj]);

  gr:=gr+abs(cr[i+ii,j+jj]-cr[i+ii,j+jj-1]);
  gb:=gb+abs(cb[i+ii,j+jj]-cb[i+ii,j+jj-1]);
  gg:=gg+abs(cg[i+ii,j+jj]-cg[i+ii,j+jj-1]);
```

```
  end;
gr:=gr div 4;
gg:=gg div 4;
gb:=gb div 4;
gs:=(gr+gg+gb)div 3;
StringGrid2.Cells[i,j]:=IntToStr(gr);
// створення кінцевого зображення
if gs>154 then gs:=154;
//Image1.Canvas.Pixels[i,j]:=RGB(2,2,2)
//else
Image1.Canvas.Pixels[i,j]:=RGB(255-gs,205-gs,155-gs);
j:=j+1;
until  j> Image.Height-3;
i:=i+1;
until  i> Image.Width-3;
  Progressbar2.Visible:=false;
end;
```

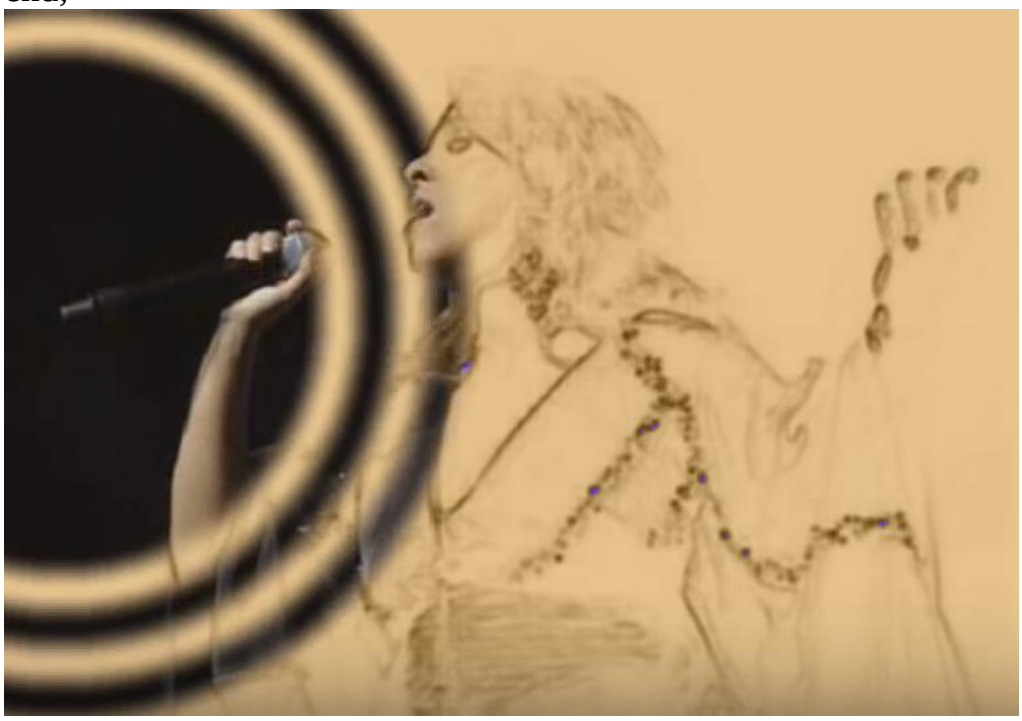


Рисунок 4.2 –Приклад застосування градиентного методу

Функція отримання червонного кольору в RGB-форматі

```
Function ExRed (col:Tcolor):integer;
var s,sr,sb,sg:string;
srr,sbb,sgg,x:integer;
begin
  s:=ColorToString(col);
    sr:=s;
```

```
    Delete(sr, 8, 2);
    Delete(sr, 4, 2);
    Delete(sr, 2, 2);
    Val(sr,sgg,x);
    sg:=s;
    Delete(sg, 4, 4);
    Delete(sg, 2, 2);
    Val(sg,srr,x);
    sb:=s;
    Delete(sb, 6, 4);
    Delete(sb, 2, 2);
    Val(sb,sbb,x);
    result:=srr;
end;
```

4.1.2 Створення маски

Функціональна суть створення маски об‘єкта полягає у необхідності відокремисті його від фону, на якому можуть міститися інші об‘єкти.

Рисунок 4.3 –Початкове зображення фону

Рисунок 4.4 –Початкове зображення об‘єкту

Для створення маски необхідно візначити ознаки, котрі необхідні для відокремлення об‘єкту. У кольорових зображеннях це можуть бути інтенсивності 3-х кольорів (RGB). У такому випадку потрібно визначити порові значення допустимих кольорів вибраного об‘єкту. Після цього створюється маска на допоміжному растрі.

Нище наведений фрагмент коду процедури, яка дає можливість отриматий маску об‘єкта зображення, представленого на рисунки 4.4. Маска об‘єкта зображеня на рисунку 4.5

```
procedure  GRADMETOD (Sender: TObject);
. . .
begin // передача параметрів зображення
. . .
Image2.Canvas.Brush.Color:=clWhite;
Image2.Canvas.Pen.Color:=clWhite;
Image2.Canvas.Rectangle(0,0,Image2.Width,Image2.Height); // допомізний растр
        delt:=20;
        cnt:=0;
        sumrgb:=0;
       For i:=0 to 639 do
       For j:=0 to 479 do
         begin
          col:=Image1.Canvas.Pixels[i,j];
         r:=ExBlu(col);
         b:=ExGre(col);
         g:=ExRed(col);
```

```
 cnt:=cnt+1;
 srrgbp:=r+g+b div 3;
 sumrgb:=sumrgb+srrgbp;
end;
 srrgb:=sumrgb div cnt;  ///  шукаємо порогове значення.
 Edit8.Text:=IntToStr(srrgb);
 Image2.Canvas.Brush.Color:=clBlack;
 Image2.Canvas.Pen.Color:=clBlack;
For i:=0 to 639 do
For j:=0 to 479 do
 begin
  col:=Image1.Canvas.Pixels[i,j];
 r:=ExBlu(col);
 b:=ExGre(col);
 g:=ExRed(col);
 tek:=r+g+b div 3;
if tek > srrgb then
 Image2.Canvas.Ellipse(i-delt,j-delt,i+delt,j+delt);  //  створюємо маску.
 end;
```

Рисунок 4.5 –Маска визначеного об‘єкту.

4.1.3 Розмивання меж об‘єкта

Розмивання меж об‘єкта потрібне для більш реалісчного сприняття згенерованого зобрження. Створюється ромиття шляхом підрахунку та застосування коефіціенту близості пікселя до межи маски обраного об‘єкта. Наведена нижче процедура створює розмиття контурів обєкта на

масці, яка потім використовується для перенесення об‘єта на новий фон (рисунок 4.6).

```
procedure TImageForm. EdgeSmooth;
 var
 col1,col2,colf:TColor;
 pix:TPoint;
  ok: boolean;
 x,y,xo,yo,i,ii,j,jj,step,cstep,delta,k,
 r,b,g,srrgb,srrgbp,sumrgb,tek,delt:integer;
 Begin
 // clin(Image3);
  delta:=20;
  colf:=RGB(254,254,254);
For ii:=0 to 639 do
 For jj:=0 to 479 do
 begin
if Image2.Canvas.Pixels[ii,jj] = clYellow then
   Image2.Canvas.Pixels[ii,jj] := clblack else
   Image2.Canvas.Pixels[ii,jj] := colf;
  end;
   col1:=colf;
  step:=255 div delta;
  For k:=1 to delta do
  begin
   cstep:=(delta-k)*step;

   col2:=RGB(cstep,cstep,cstep);
 For ii:=0 to 639 do
 For jj:=0 to 479 do
 if Image2.Canvas.Pixels[ii,jj] = clBlack then
begin
 For i:=-1 to 1 do
 For j:=-1 to 1 do
  begin
  x:=i+ii;
  y:=j+jj;
  if Image2.Canvas.Pixels[x,y] = col1 then
  Image2.Canvas.Pixels[ii,jj]:=col2 ; //
  end;

 end;
   col1:=col2;
```

```
 end;
For ii:=0 to 639 do
 For jj:=0 to 479 do
 begin
       r:=ExRed( Image2.Canvas.Pixels[ii,jj]);
   StringGrid7.Cells[ii+1,jj+1]:=IntToStr(r);
  end;

end;
```

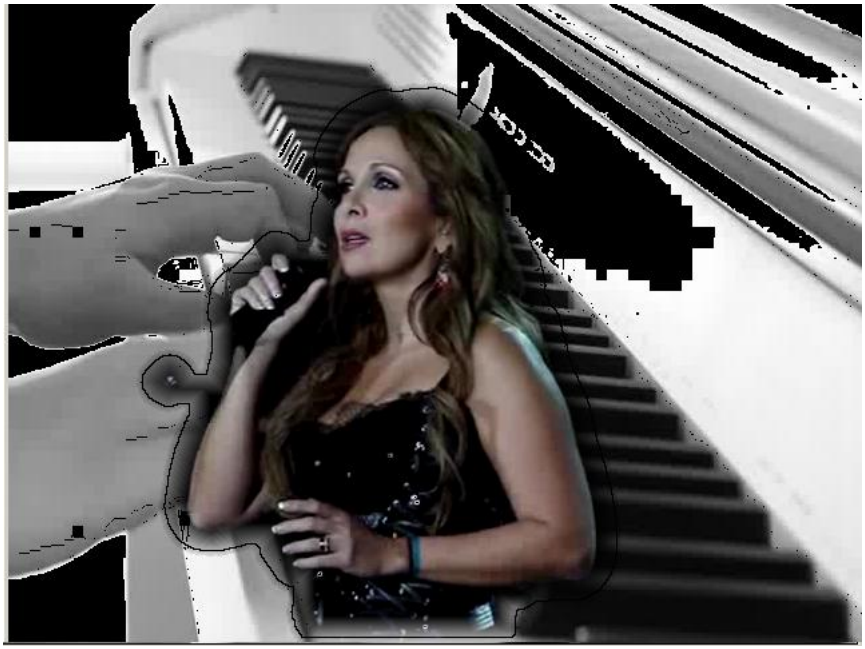


Рисунок 4.6 –кінцеве зображення об‘єктуна норвому фоні.

4.2 Завдання до виконання роботи

1. Створіть за допомогою веб-камери, або іншим шляхом 2 відеофрагмента, один з яких повинен містити рухомий об ‘єкт(або об‘єкти) який ви будите переносити на новий фон, а інший – майбутній фон. Один з цих відеофрагментів повинен містити ваше рухоме обличча.

2. Розбити ваш avi-відеофайл на посліфжовныcть кадрів bmp-формату 640x480 (не меньш ніж 100 кадрів за допомогою VirtualDub).

3. При необхідності за допомогою градіетного методу та інших характерних ознак виділти об‘єкти та іх контури на кадрі. При необхідності застосувати функцію замикання контурівоб‘єктів.

5. Створити бинарну маску для усіх замкнених об‘єктів.

6. При необхідності видалити усі непотрібні об‘єкти меньшої пощи.

7. Перенести обраний об‘єкт шляхом його переносу на новий фон

8. Застосувати функцію зглажування меж обраного об‘єкту на новому фоні.

9. Застосувати послідовність перевворень (п.з 2 по 8) до усієї послідовності кадрів.

10. За допомогою VirtualDub нову послідовності кадрів змонтувати avi-відеофайл.

4.3 Варіанти завдань до виконання роботи

РГР виконується без застосування варіантів. Оригінальність виконаної роботи забеспечується унікальністю вибраних початкових фрагментів відео та унікальністю послідовності функцій обробки, застосованих до цих фрагментів.

4.4 Вимги до звіту

Звіт повинен містити:

- Титульну сторінку з даними про виконавця і перевіряючого.
- Тема та мета розрахунково-графічної роботи.
- Блок-схема реалізованого завдання.
- Лістинг реалізованої функції.
- Початкові зображення фону .
- Початкові зображення об‘єкту.
- Результати застосування реалізованої функції до тестових зображень.
- Висновки про виконання роботи.

Звіт повинен бути оформлений відповідно до вимог СОКР

РЕКОМЕНДОВАНА ЛИТЕРАТУРА

1. Прэтт У. Цифровая обработка изображений. В 2 т. М.: Мир, 1982.
2. Ресурс [http://matlab.exponenta.ru/imageprocess/book3/index.php]
3. Ежов А.А., Шумский С.А. Нейрокомпьютинг и его применения в экономике и бизнесе. – М, 1998.
4. Загоруйко Н.Г. Прикладные методы анализа данных и знаний. – Новосибирск: ИМ СО РАН, 1999.
5. Закревский А.Д. Логика распознавания. – Минск: Наука и техника,1988.
6. Золотых Н.Ю. Машинное обучение. Курс лекций (ВМиК Нижегородского госуниверситета). – http://www.uic.nnov.ru/~zny/ml.
7. Журавлёв Ю.И., Рязанов В.В., Сенько О.В. «Распознавание». Мате- матические методы. Программная система. Практические применения. – М.: ФАЗИС, 2006.
8. Искусственный интеллект. – В 3 кн. Кн. 2. Модели и методы: Справочник// Под ред. Д.А. Поспелова. – М.: Радио и связь, 1990.
9. Кобзарь А.И. Прикладная математическая статистика. Для инженеров и научных работников. – М.: Физматлит, 2006.