Київський національний університет імені Тараса Шевченка
Факультет комп'ютерних наук та кібернетики
Кафедра моделювання складних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему:

Виявлення контурів зображення зі згладжуванням поверхнею другого порядку

Студентки 4 курсу кафедри МСС:
Анни ПІДЛЕТЕЙЧУК

Науковий керівник:
доцент, кандидат фізико-математичних наук
Володимир МАТВІЄНКО

Робота заслухана на засіданні кафедри моделювання складних систем та рекомендовано до захисту в ЕК, протокол № 10 від 5 червня 2023р.

Завідувач кафедри МСС _____ д.т.н., доцент Дмитро ЧЕРНІЙ

Київ -2023

Анотація

Випускна кваліфікаційна робота бакалавра: 46 сторінок, 18 рисунків, 11 інформаційних джерел.

Актуальність роботи: Перетворення зображення в набір кривих є важливим етапом, оскільки ці криві відображають суттєві особливості зображення та дозволяють скоротити обсяг інформації для подальшого аналізу. Крім того, контурні зображення можуть бути використані у різних галузях, таких як медична діагностика, відеоспостереження, графічний дизайн тощо.

Об’єкт дослідження: Застосування різноманітних методів цифрової обробки, а саме тих, які дозволяють знаходити контури в зображенні, а також тих , які дозволяють покращити якість знайдених контурів.

Мета роботи: Розглянути різні відомі методи виділення контурів зображення, проблему виникнення фіктивних границь, з якою можна зустрітися під час цього та основні способи боротьби з нею. Також розробити алгоритм виявлення контурів зображення зі згладжуванням поверхнею другого порядку.

Предмет дослідження: Програмна реалізація градієнтних методів: Робертса, Превітта, Собела, Кірша, методів для проведення згладжування: прямокутного фільтра, фільтра Гауса, методу застосування порогової функції, більш складних методів таких, як використання детекторів Марра-Хілдрета та Кенні . Також програмна розробка алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку і програмна реалізація застосування його на експериментальних зображеннях.

Ключові слова: ПОШУК КОНТУРІВ, ГРАДІЄНТ, ОПЕРАТОР, ЗГЛАДЖУВАННЯ, ПОРОГОВА ФУНКЦІЯ, ПОВЕРХНЯ ДРУГОГО ПОРЯДКУ.

Зміст

Вступ

Контури - це криві на зображенні, які відображають різку зміну яскравості та інших видів неоднорідностей. Ці зміни можуть бути спричинені різними факторами: зміна освітлення, кольору, глибини сцени (орієнтації поверхні). Перетворення зображення в набір кривих є важливим етапом, оскільки ці криві відображають суттєві особливості зображення та дозволяють скоротити обсяг інформації для подальшого аналізу. Крім того, контурні зображення можуть бути використані у різних галузях, таких як медична діагностика, відеоспостереження, графічний дизайн тощо. Лікарі можуть використовувати їх, наприклад, для виявлення та оцінки пухлин на зображеннях медичних сканів, таких як КТ, МРТ або рентгенограм. Також контурні зображення корисні при створенні тривимірних моделей органів та тканин для планування хірургічних втручань та процедур лікування. Процес знаходження і виділення контурів на відеопотоці допомагає в визначенні та аналізі форми і розміру об'єктів, їх ідентифікації (наприклад, автомобілі, люди або тварини), виявленні особливостей руху. Це може допомогти в системах контролю, управління, охорони (наприклад, в ситуаціях, де хтось входить до обмеженої зони, про це стане відомо). В дизайні пошук контурів теж займає важливе місце. Вони допомагають в більш точній редакції , дають можливість виділяти певні об’єкти, наприклад, при видаленні фону, також дають можливість для створення векторних об'єктів на основі растрових зображень (після цього зображення зможуть бути масштабовані без втрати якості). Отже, через можливість широкого застосування в різних галузях пошук контурів в зображеннях є досить актуальною темою.

На практиці існує багато різних методів виділення контурів, в основному це градієнтні методи. Але , використовуючи їх, можна стикнутися з проблемами . Наприклад, можна побачити виникнення фіктивних контурів, відсутність необхідної зв’язності між контурами, сильний вплив шуму, тощо.

Метою роботи було розглянути різні відомі методи виділення контурів зображення (наприклад, методи Робертса, Превітта, Собела, Кірша), проблему виникнення фіктивних границь, з якою можна зусрітися під час цього та основні способи боротьби з нею (згладжування, порогова обробка), всі алгоритми реалізувати програмно. Також метою роботи було розробити алгоритм виявлення контурів зображення зі згладжуванням поверхнею другого порядку, реалізувати його програмно мовою програмування python. Оскільки, цей метод ніде не був розроблений раніше, то спробувати порівняти його практичні результати з іншими відомими та популярними методами.

Розділ 1. Виявлення контурів зображення

1.1 Цифрове зображення. Будова. Види

Цифрове зображення визначається за допомогою дискретизації неперервних аналогових даних в просторовій області. Воно складається з прямокутного масиву пікселів (x, y, u). Кожен з них є комбінацією розташування $(x, y) \in \mathbb{Z}^2$ і значення u , яке описує рівень яскравості в точці (x, y). Тут $\mathbb{Z}$ - множина всіх цілих чисел. Точки $(x, y) \in \mathbb{Z}^2$ утворюють регулярну сітку.

Вікном $W_p^{m,n}(I)$ називається частина зображення I, що має розмір mxn і позиційонована щодо початкової точки p (тобто деякого пікселя). За замовчуванням беруть m = n - непарне число, а p - центр вікна.

У бінарному зображенні кожен піксель може мати лише два значення: 0 - білий та 1 - чорний. У скалярному зображенні, яке називається напівтоновим, значення пікселів є цілими числами $u = \{0,1,\ldots,255\}$ і лінійно інтерполюються між 0 - чорним та 255 - білим кольорами. Різнокольорові зображення в відомій моделі RGB складаються з трьох каналів, що відповідають червоній, зеленій та синій компонентам. Кожен з цих каналів є напівтоновим зображенням, де значення пікселів належать множині $\{0,1,\ldots,255\}$. Далі в роботі будуть використовуватися напівтонові зображення. Детальніше про цифрове зображення можна почитати в джерелі [1].

1.2 Пошук контурів зображення за допомогою частинних похідних

Цифрове зображення представляє собою масив пікселів, який можна розглядати як функцію яскравості f(x,y), що залежить від двох змінних - координати x та координати y. Пошук контурів полягає у виявленні місць, де значення яскравості раптово змінюються. Контури на зображенні можуть бути

знайдені за допомогою різних методів (найчастіше контури визначають змінами локальних похідних).

Перепади у напрямках x та y для функції яскравості f(x,y) визначаються за допомогою частинних похідних $\frac{\partial f(x,y)}{\partial x}$ та $\frac{\partial f(x,y)}{\partial y}$, що в свою чергу пропорційні швидкостям зміни яскравості у відповідних напрямках. Контури, перпендикулярні до осі x, можна визначити за допомогою похідної $\frac{\partial f(x,y)}{\partial x}$, а контури, перпендикулярні до осі y, - за допомогою похідної $\frac{\partial f(x,y)}{\partial y}$.

У практичних завданнях потрібно виділяти контури, напрямок яких є довільним. Для цього можна використовувати модуль градієнта функції яскравості

$$|\nabla f(x,y)| = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2}, \qquad (1.2.1)$$

який пропорційний максимальній (за напрямом) швидкості зміни функції яскравості в даній точці і не залежить від напрямку контура. Часто використовуваними формулами ще є

$$|\nabla f(x,y)| = \left|\frac{\partial f(x,y)}{\partial x}\right| + \left|\frac{\partial f(x,y)}{\partial y}\right|, \qquad (1.2.2)$$

$$|\nabla f(x,y)| = max\{\left|\frac{\partial f(x,y)}{\partial x}\right|, \left|\frac{\partial f(x,y)}{\partial y}\right|\}. \qquad (1.2.3)$$

Модуль градієнта функції яскравості завжди має невід'ємні значення, тому на отриманому зображенні точки, що відповідають контурам, мають підвищений рівень яскравості. Отже, шукати контури потрібно, де модуль градієнта досягає локального максимума.

Шукати контури можна й за допомогою других частинних похідних. Потрібно знаходити точки, де лапласіан

$$\nabla^2 f(x,y) = \frac{\partial f^2(x,y)}{\partial x^2} + \frac{\partial f^2(x,y)}{\partial y^2} \qquad (1.2.4)$$

переходить через нуль.

Частинні похідні першого порядку відзначають на зображенні широкі перепади яскравості. Похідні другого порядку, натомість, можуть виявити більш дрібні деталі, такі як тонкі лінії, ізольовані точки та шум. При роботі з похилими та ступінчастими змінами яскравості другі похідні дають відгук у два рази більший. Напрямки перепадів яскравості (від світлого до темного або навпаки) можна визначити за допомогою знаків перших та других похідних. Цю та більш грунтовну інформацію про це можна знайти в джерелах [1] - [3].

1.3 Використання лінійних локальних операторів в зображеннях

Згідно джерел [1], [4] для певного даного зображення I розміру $N_{cols} \times N_{rows}$ розглядаються ковзні вікна W_p розміру (2k + 1)x(2k + 1) з центром p (початкова точка). Ця початкова точка обходить всі можливі позиції пікселів у зображенні I і у такий спосіб переміщує вікно по ньому. У кожній позиції вікна виконується локальна операція, результат якої визначає нове значення у точці p. Цей процес перетворює вихідне зображення I на нове зображення J.

Лінійний локальний оператор визначається в точці p = (x, y) шляхом згортки зображення I з ядром фільтра W наступним чином

$$J(p) = I*W(p) = \frac{1}{S} \sum_{i=-k}^{k} \sum_{j=-k}^{k} w_{(i,j)} I(x+i, y+j) \quad , \qquad (1.3.1)$$

де вагові коефіцієнти $w_{(i,j)} \in \mathbb{R}$ і масштабний коефіцієнт S > 0. Якщо вікно з центром у точці p не повністю міститься в I, то потрібно застосовувати “спеціальну стратегію обробки крайових пікселів”. Немає загальної згоди щодо такої стратегії. Одним з варіантів є виконання тієї ж локальної операції в меншому вікні. Ще одним варіантом є додавання додаткових пікселів навколо зображення, щоб утворити більше простору для вікна. Також часто обирають стратегію ігнорування крайових пікселів, тобто не виконування просторових операторів на пікселях, які знаходяться на краях зображення. Ця стратегія буде

використовуватися в прикладах в даній роботі. Ядра локальних лінійних операторів зазвичай описують як на Рис. 1.3.1 (приклад для оператора 3x3):

w1,1	w1,2	w1,3
w2,1	w2,2	w2,3
w3,1	w3,2	w3,3

/ S

Рис. 1.3.1 - Ядро локального лінійного оператора 3x3

1.4 Застосування різних операторів знаходження градієнта

Джерела [2], [3] дає багато інформації щодо застосування різних градієнтних методів для знаходження контурів в зображенні. Для підкреслення перепадів на зображенні можна використовувати такий найпростіший підхід, який полягає в наближеному обчисленні частинних похідних в точці (x, y) (використовують вікно 2x2:

$$\begin{bmatrix} f(x-1,y-1) & f(x-1,y) \\ f(x,y-1) & f(x,y) \end{bmatrix} \quad): \tag{1.4.1}$$

$$G_x = \frac{\partial f(x,y)}{\partial x} = \text{f(x,y)} - \text{f(x-1,y)}, \tag{1.4.2}$$

$$G_y = \frac{\partial f(x,y)}{\partial y} = \text{f(x,y)} - \text{f(x,y-1)} . \tag{1.4.3}$$

Ще одним простим способом є застосування оператора Робертса з використанням вікна 2x2 (маски на Рис. 1.4.1), який реалізовується наступними формулами: (розглядаються такі точки

$$\begin{bmatrix} f(x,y) & f(x,y+1) \\ f(x+1,y) & f(x+1,y+1) \end{bmatrix} \quad) \tag{1.4.4}$$

$$G_x = \frac{\partial f(x,y)}{\partial x} = \text{f(x+1,y+1)} - \text{f(x,y)}, \tag{1.4.5}$$

$$G_y = \frac{\partial f(x,y)}{\partial y} = \text{f(x+1,y)} - \text{f(x,y+1)} . \qquad (1.4.6)$$

Маски розміром 2×2 являються простими для розуміння, проте вони не такі зручні для виявлення напрямів перепадів, як маски, які є симетричними щодо центрального елемента (мінімальний розмір 3×3). Останні в свою чергу враховують значення з обох сторін від центральної точки. Це дає більше інформації про напрямок перепаду яскравості.

Одним з найбільш простих дискретних наближень частинних похідних у випадку використання масок 3x3 являється:

$$G_x = \frac{\partial f(x,y)}{\partial x} = (\text{ f(x+1,y-1) + f(x+1,y) + f(x+1,y+1) }) -$$
$$- (\text{ f(x-1,y-1) + f(x-1,y) + f(x-1,y+1) }), \qquad (1.4.7)$$
$$G_y = \frac{\partial f(x,y)}{\partial y} = (\text{ f(x+1,y+1) + f(x,y+1) + f(x+1,y+1) }) -$$
$$- (\text{ f(x-1,y-1) + f(x,y-1) + f(x+1,y-1) }) . \qquad (1.4.8)$$

Для реалізації цих формул використовується оператор , який називається оператором Превітта (маски на Рис. 1.4.2). Інтуїтивно можна очікувати, що контури будуть знайдені більш точно, ніж при використанні оператора Робертса.

Невелика видозміна останніх двох формул (1.4.7), (1.4.8) полягає у використанні вагового коефіцієнта 2 для середніх елементів:

$$G_x = \frac{\partial f(x,y)}{\partial x} = (\text{ f(x+1,y-1) + 2f(x+1,y) + f(x+1,y+1) }) -$$
$$- (\text{ f(x-1,y-1) +2f(x-1,y) + f(x-1,y+1) }), \qquad (1.4.9)$$
$$G_y = \frac{\partial f(x,y)}{\partial y} = (\text{ f(x+1,y+1) + 2f(x,y+1) + f(x+1,y+1) }) -$$
$$- (\text{ f(x-1,y-1) + 2f(x,y-1) + f(x+1,y-1) }). \qquad (1.4.10)$$

Цей ваговий коефіцієнт 2 для середніх елементів використовується для зменшення ефекту згладжування за рахунок надання більшої ваги середнім точкам. Для реалізації цих формул використовується оператор, який називається оператором Собела(маски на Рис. 1.4.3).

У оператора Собела вплив шуму кутових елементів трохи менший, ніж у оператора Превітта. Проте реалізовувати маски оператора Превітта трохи легше, ніж маски оператора Собела.

-1	0
0	1

0	-1
1	0

Рис. 1.4.1 - Оператор Робертса

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

Рис. 1.4.2 - Оператор Превітта

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Рис. 1.4.3 - Оператор Собела

Прикладом нелінійного просторового фільтра для виділення контурів є фільтр Кірша . В джерелі [5] розповідається, що при класичному використанні оператора Кірша необхідно знайти абсолютні значення перепадів яскравості за вісьмома напрямками. А потім знайти максимальне з отриманих восьми значень. Значення по одному напрямку , наприклад “з півночі на південь” буде шукатися, використовуючи маску на Рис.1.4.4. Далі потрібно обертати дану маску навколо центру ще 7 разів, так отримаємо маски для решти напрямків.

5	5	5
-3	0	-3
-3	-3	-3

Рис. 1.4.4 - Одна з масок, що застосовуються в фільтрі Кірша

Фільтр Кірша є інструментом, який дозволяє підкреслити контури з незначним перепадом яскравості.

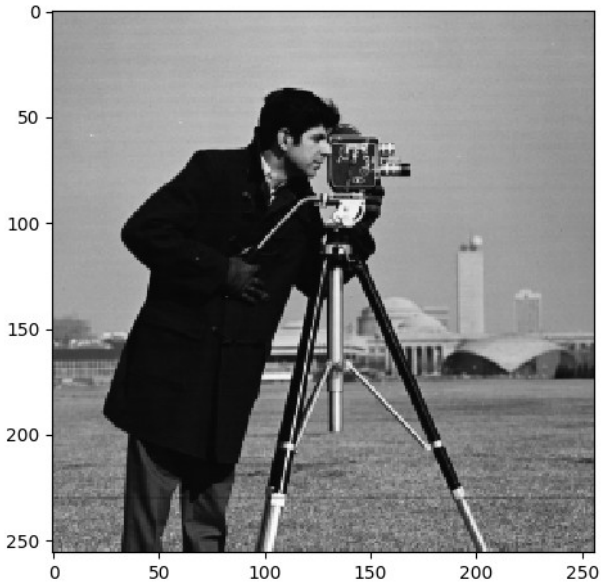
Також можна розглянути метод перетину нульового рівня. В цьому методі використовується оператор Лапласа(1.2.4) . Маскою цього оператора є маска на Рис. 1.4.5.

0	1	0
1	-4	1
0	1	0

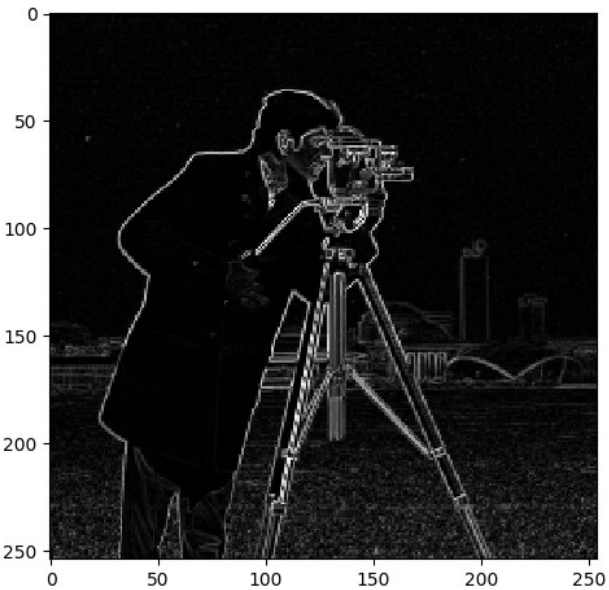
Рис. 1.4.5 - Маска оператора Лапласа

Для виділення контуру зображення, якщо виділяти лінії абсолютним значенням функції Лапласа, то товщина лінії подвоюється, тому краще використати тільки позитивні значення лапласіану[2].

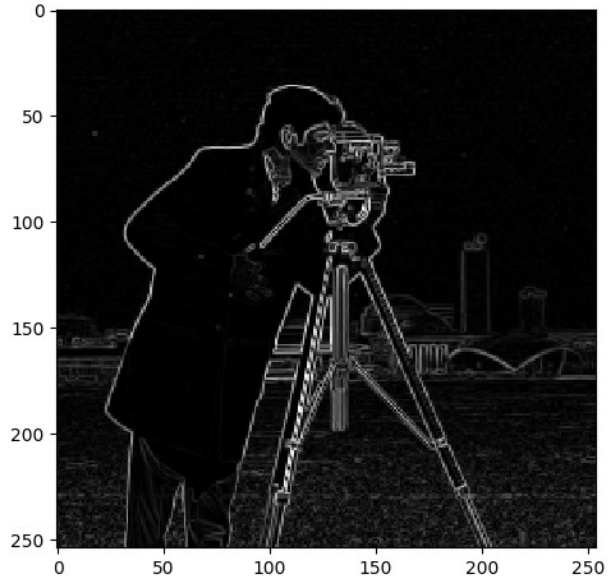
Розглянемо приклади роботи різних детекторів контурів, розглянутих вище, на Рис. 1.4.6.



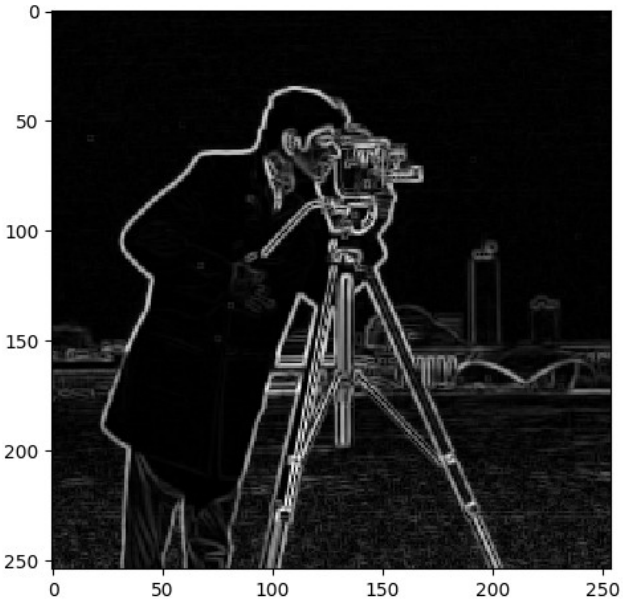
a



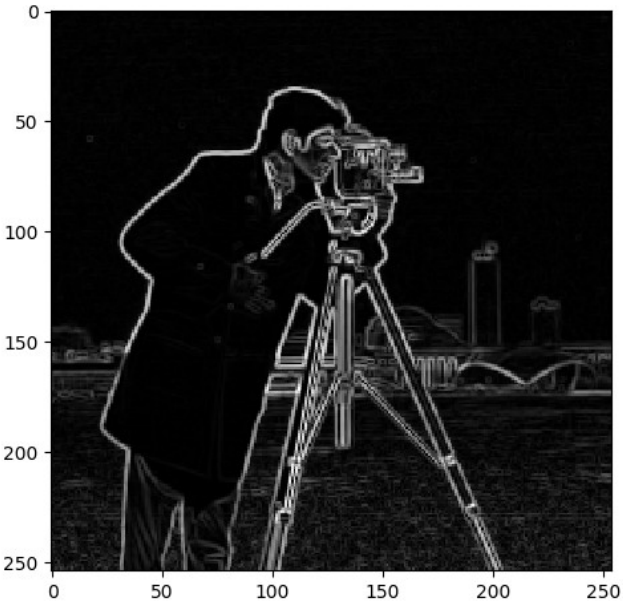
b


0
50
100
150
200
250
0
50
100
150
200
250

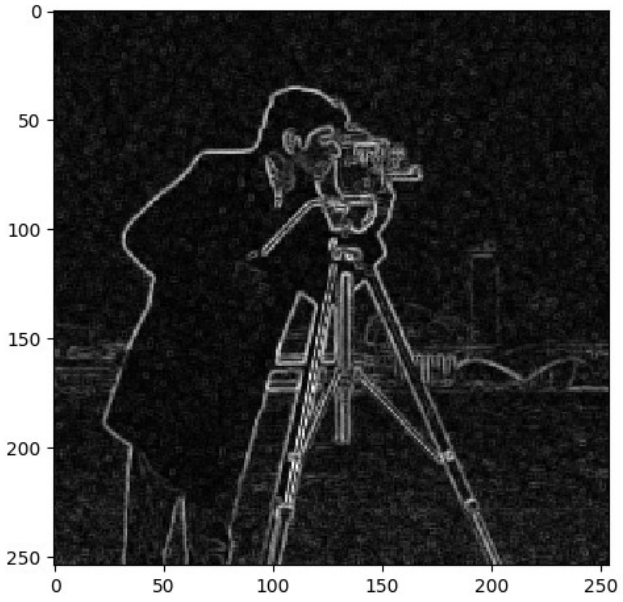
c


0
50
100
150
200
250
0
50
100
150
200
250

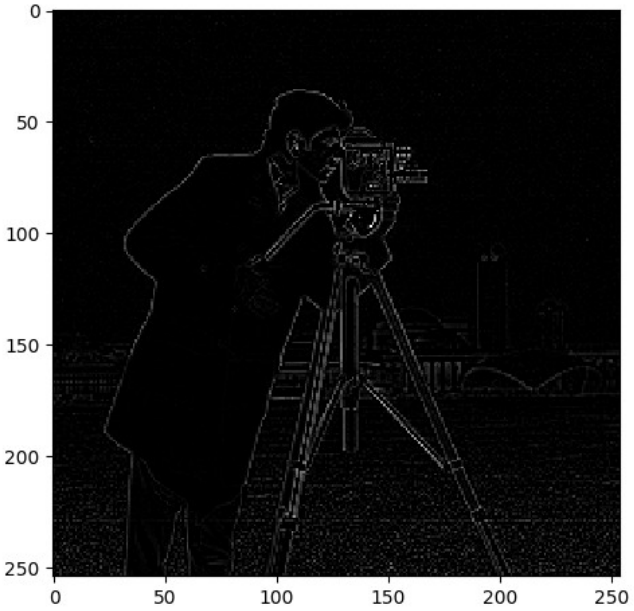
d


0
50
100
150
200
250
0
50
100
150
200
250

e



f



g

Рис. 1.4.6 - Застосування різних операторів для знаходження контурів:
a- Оригінальне зображення [10]; b- Дія наближеного обчислення похідних;
c - Дія оператора Робертса; d - Дія оператора Превітта;
e - Дія оператора Собела; f - Дія оператора Кірша;
g - Дія оператора Лапласа.

1.5 Необхідність згладжування при застосуванні градієнтних методів. Приклади операторів згладжування

Після застосування градієнтних методів до виявлення контурів, з'являються фіктивні границі в зображенні, оскільки градієнт є чутливим навіть до незначних змін яскравості. Тому перед застосуванням градієнтних операторів необхідно провести згладжування.

Наведемо приклади найпопулярніших операторів згладжування:

1. Прямокутний фільтр розміром (2k + 1)x(2k + 1) рахує локальне середнє по формулі

$$J(p) = \frac{1}{(2k+1)^2} \sum_{i=-k}^{k} \sum_{j=-k}^{k} w_{(i,j)} I(x+i, y+j) \quad . \qquad (1.5.1)$$

Він являється простим прикладом оператора згладжування. Після його використання помітно зменшується контрастність зображення, тому часто встачає використання ядер 3x3 або 5x5.

2. Фільтр Гауса представляє собою локальну згортку з ядром, визначеним вибірками з двовимірної гаусової функції. Ця функція є добутком двух одновимірних гаусових функцій

$$G_{\sigma,\mu_x,\mu_y} = \frac{1}{2\pi\sigma^2} e^{\frac{-(x-\mu_x)}{2\sigma^2}} e^{\frac{-(y-\mu_y)}{2\sigma^2}} \quad , \qquad (1.5.2)$$

де (μ_x , μ_y) - математичні сподівання по осям x та y, σ - стандартне відхилення (σ^2 - дисперсія), яке також називають радіусом цієї функції, а e - число Ейлера. Припускається, що гаусова функція є центрована (тобто μ_x = = μ_y = 0), тоді формула спрощується

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{\frac{-(x)^2}{2\sigma^2}} e^{\frac{-(y)^2}{2\sigma^2}} \quad . \qquad (1.5.3)$$

Використовуючи статистичне правило трьох сігм, для вибірки із G_σ потрібно ядро з разміром 6σ - 1. Чим більше σ , тим нечіткіше зображення ми отримаємо.

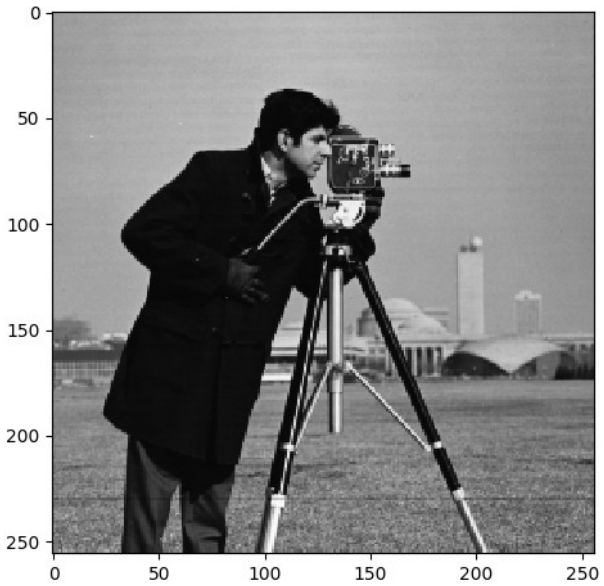
Нижче на Рис. 1.5.1 наведено приклад ядра гаусівського фільтра згладжування при $\sigma = 1$.

Про ці два метода та більше інформації про згладжування зображення можна знайти в джерелі [1].

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Рис. 1.5.1 - Якщо $\sigma = 1$, то ядро гаусівського фільтра згладжування

Розглянемо приклади роботи операторів згладжування , про які йшлось вище, на Рис. 1.5.2.



а

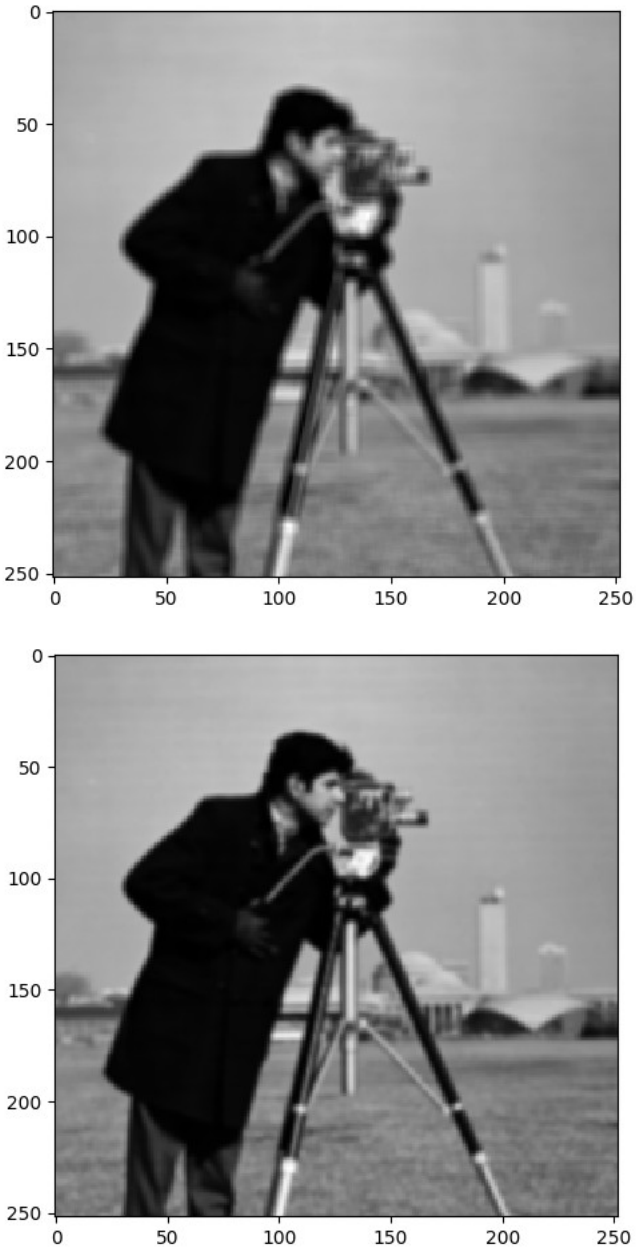
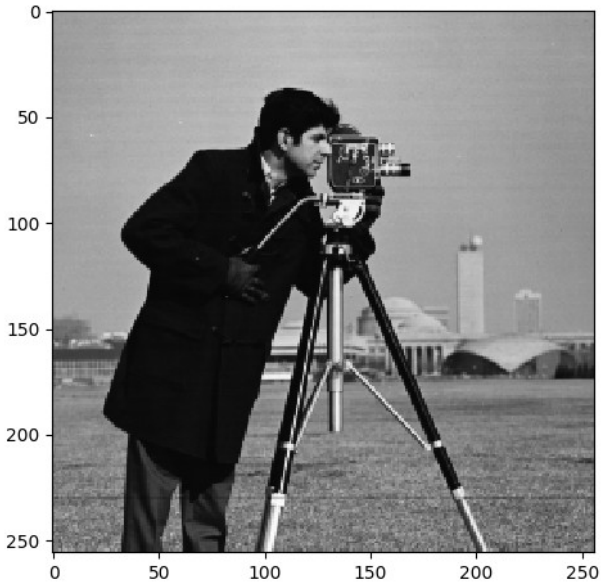


Рис. 1.5.2 - Застосування різних операторів згладжування:
a - Оригінальне фото[10]; b - Дія прямокутного фільтра;
c - Дія фільтра Гауса (при σ = 1).

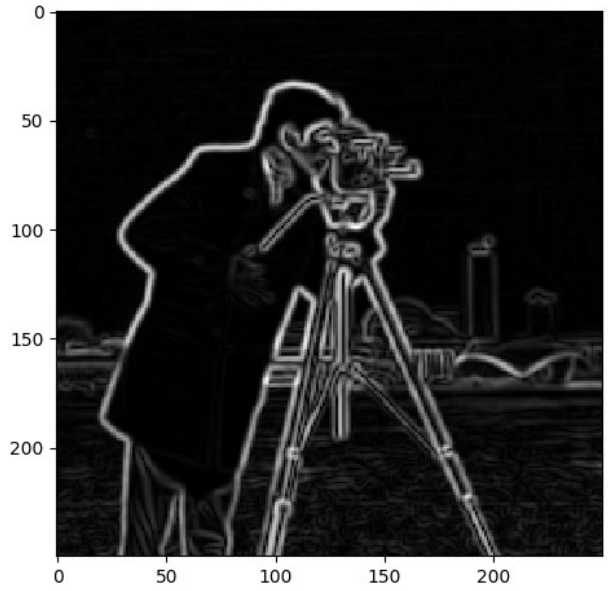
1.6 Порогова обробка (застосування порогової функції)

У джерелі [2] вибірковість детектора контурів можна підвищити шляхом порогової обробки градієнтного зображення. Суть метода полягає в тому, що всі пікселі, яскравість яких більша за якесь наперед вибране значення ε , відмічаються білим кольором, а решта – чорним. Вибір порогового значення

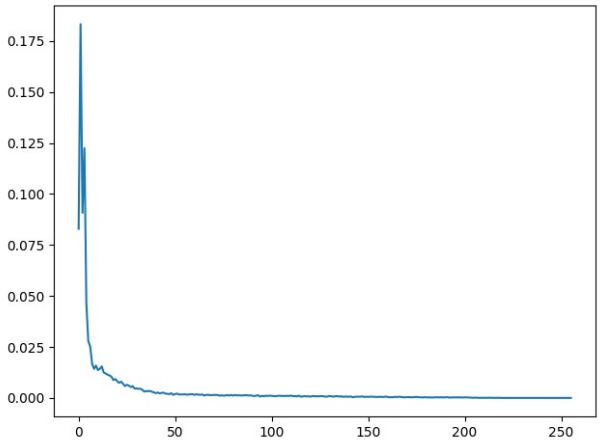
ε залежить від конкретної задачі та властивостей зображення, для цього існує багато різних методів, наприклад відомий метод Оцу. Після порогової обробки залишиться менше контурних ліній, які стануть різкішими. Але з’явиться такий недолік, як розриви на багатьох контурах. Зазвичай, коли ми хочемо виділити основні контури та водночас зберегти їх зв'язність, ми використовуємо згладжування та застосування порогової функції одне за одним.



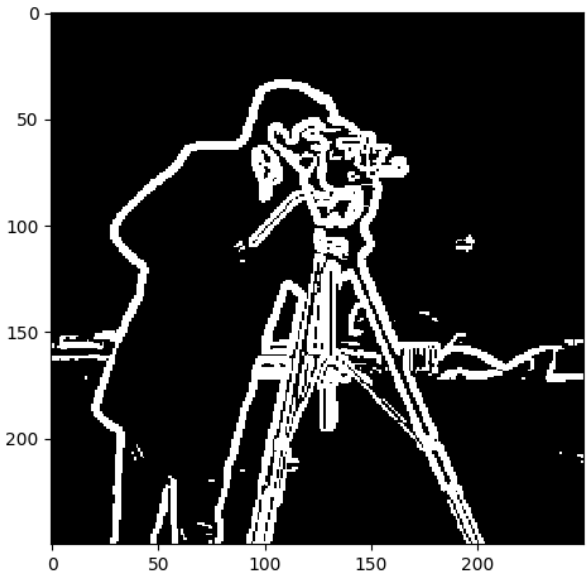
a



b



c



d

Рис. 1.6.1 - Застосування згладжування з пороговою функцією:
a - Оригінальне фото [10]; b - Застосування фільтра Гауса та оператора Собела ; c - Нормована гістрограма рисунку b (показує для кожного рівня яскравості від 0 до 255 число пікселів на зображенні цього рівня поділене на загальну кулькість пікселів зображення, оскільки ми шукаємо тільки найбільш світлі пікселі, то занулимо всі пікселі , значення яскравості яких менше 51, решту відмічаємо як 255; d - Застосування фільтра Гауса, оператора Собела та порогової функції 20% від максимального значення градієнтного зображення.

1.7 Знаходження контурів зображення детектором Марра-Хілдрета та детектором Кенні

Розглянемо детектор контурів Марра-Хілдрета (описаний в [2]), який є однією з перших успішних спроб впровадження більш складного аналізу в процес знаходження контурів. Марр і Хілдрет вирішили використати фільтр $\nabla^2 G$ (лапласіан гауссіана, ЛГ), де ∇^2 - оператор Лапласа, а G - двовимірна гаусова функція . Вибір цього оператора $\nabla^2 G$ базувався на двох основних ідеях.

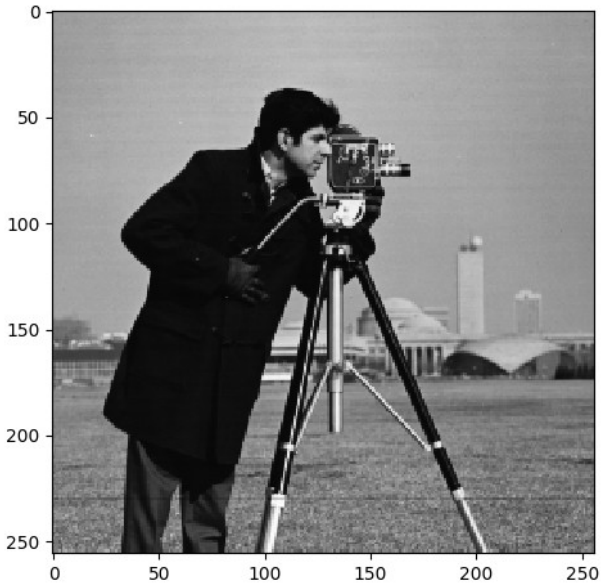
Перша ідея полягає в тому, що використання гаусової функції G в $\nabla^2 G$ забезпечує згладжування зображення. Завдяки цьому зменшується інтенсивність структур (включаючи шумові) з розмірами меншими за σ. Крім того, гаусова функція є гладкою в просторовій і в частотній областях. Через це ймовірність появи артефактів, які не були присутні на зображенні, знижується.

Друга ідея пов'язана з використанням другої похідної ∇^2 в фільтрі $\nabla^2 G$. Хоча для виявлення різких змін яскравості можна використовувати перші похідні, вони є спрямованими, а лапласіан є ізотропним (інваріантним до повороту). Це підходить для зорової системи людини та забезпечує однаковий відгук на зміни яскравості в будь-якому напрямку. Таким чином, можна уникнути необхідності застосування кількох масок для обчислення максимального відгуку.

Алгоритм Марра - Хілдрета складається зі згортки ЛГ-фільтра з вихідним зображенням $g(x,y)=[\nabla^2 G(x,y)] \bigstar f(x,y)$ і знаходження потім точок перетину нульового рівня функції g(x, y), щоб локалізувати контури на зображенні f(x, y). Зважаючи на те, що взяття другої похідної є лінійною операцією, рівняння можна переписати у вигляді $g(x,y)=\nabla^2[G(x,y) \bigstar f(x,y)]$. Отже, зображення можна спочатку згладити гаусовою функцією, а потім застосовувати оператор Лапласа (приклад такої обробки зображення можна побачити на Рис. 1.7.1(b)). Обидва останні рівняння дають ідентичні результати.

Один зі способів виявлення перетину функції g(x, y) з нульовим рівнем в пікселі p ґрунтується на перевірці значень у сусідніх пікселях (в околиці 3×3 з центром у p). Якщо щонайменше два з них, розташовані один проти одного , мають значення різних знаків, то це може свідчити про перехід через нуль. Треба превірити верхнього і нижнього, лівого і правого сусідів та тих, які відповідно по діагоналях. Зазвичай значення g(x, y) порівнюються з деяким порогом.Тобто треба, щоб не тільки щоб у протилежномих сусідів p були різні знаки, але щоб абсолютна величина різниці їх значення перевищувала заданий поріг. Якщо ці умови виконуються, то p вважається пікселем перетину нульового рівня (приклад такої обробки зображення можна побачити на Рис. 1.7.1(c)).

Марр і Хілдрет помітили, що ЛГ-фільтр можна апроксимувати різницею гауссіанів (РГ), де $\sigma 1 > \sigma 2$. Марр і Хілдрет рекомендували використовувати відношення сігм 1,6:1(це можна знайти також в [6]).



а

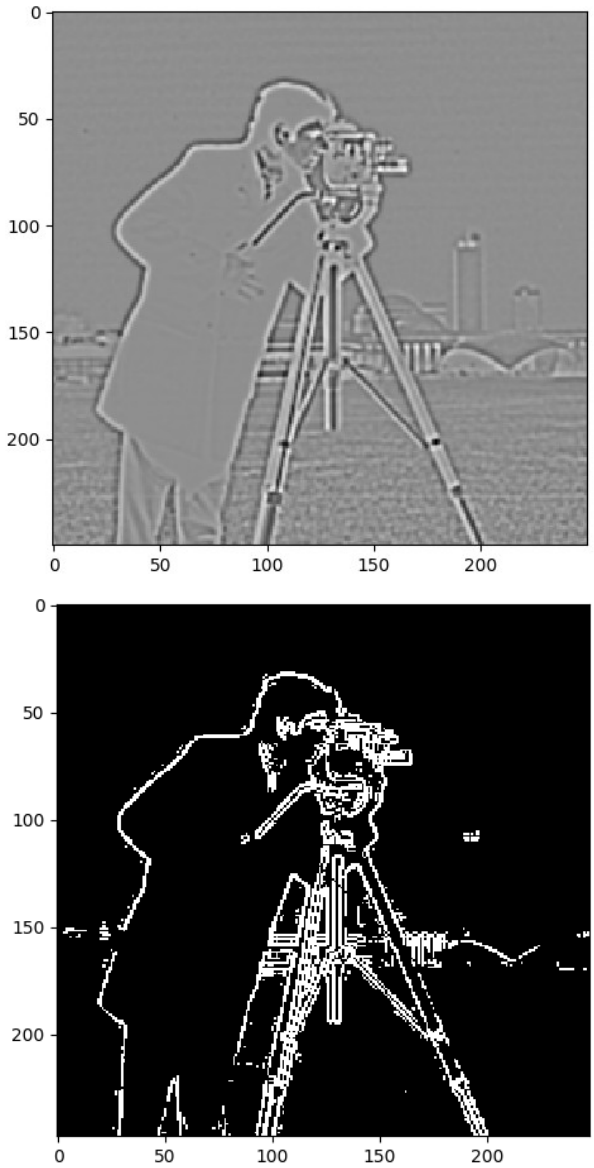


Рис. 1.7.1 - Застосування ЛГ алгоритму :
a - Оригінальне фото [10]; b - Застосування фільтра Гауса та оператора Лапласа ; c - На зображенні b знаходяться точки перетину нульового рівня з порогом 35.

Розглянемо ще однин детектор контурів - Кенні (описаний також в [2]). Він використовує більш складний алгоритм, якість його роботи в загальному випадку вища, ніж у розглянутих досі алгоритмів виділення контурів. Метод Кенні переслідує три основні цілі: низьку частоту помилок, хорошу

локалізацію контурних точок, одиночний відгук на точку контуру(це можна знайти також в [7]).

Алгоритм виділення контурів Кенні складається з наступних основних кроків:

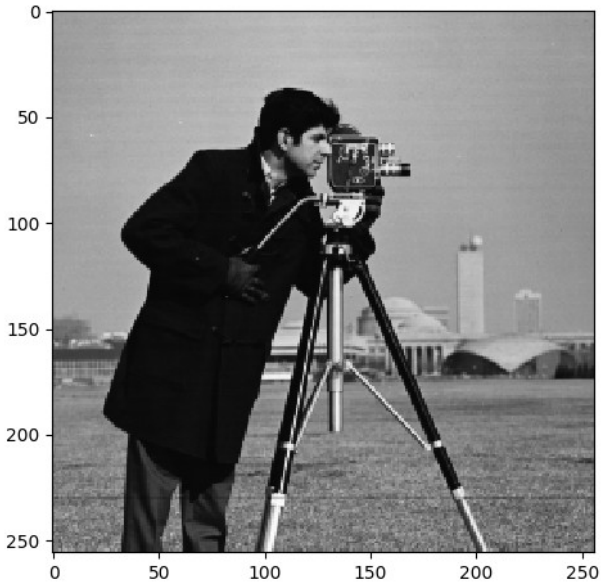
1. Згладити вихідне зображення фільтром Гауса.
2. Сформувати зображення модуля та напрямку градієнта.
3. Застосувати придушення немаксимальних точок зображення модуля градієнта.
4. Виконати перетворення з подвійним порогом та аналіз зв'язності для виявлення та зв'язування контурів.

Другий крок можна виконати багатьма способами, які були розглянуті в пункті 1.4 . Часто використовують оператор Собела з ядром 3x3 (формула для напрямку $\alpha(x,y) = arctg\frac{(\frac{\partial f(x,y)}{\partial y})}{(\frac{\partial f(x,y)}{\partial x})}$).

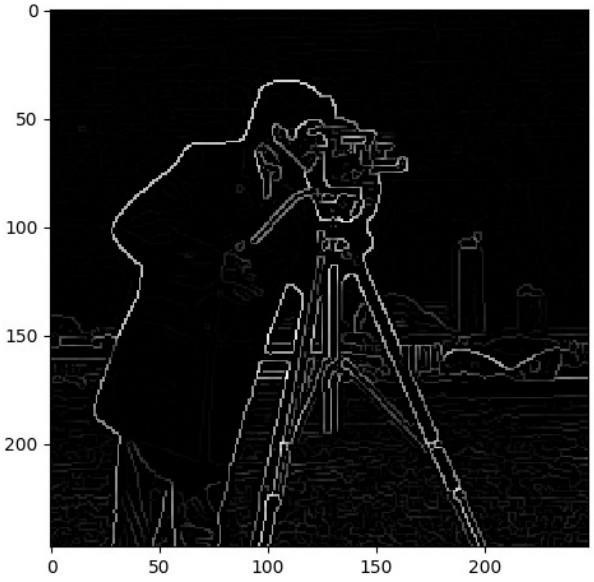
Третій крок починається з цього: в області розмірами 3×3 пікселя для контуру, що проходить через центральний елемент області, можна визначити чотири напрямки: горизонтальний, вертикальний, +45° і -45°. Потрібно привести всі можливі напрямки контуру до чотирьох фіксованих значень. Якщо кут нормалі до контуру знаходиться в інтервалі від -22,5° до +22,5° або від -157,5° до +157,5°, такий контур вважається горизонтальним, від +22,5° до +67,5° або від -112,5° до -157,5° - кут -45°, від +67,5° до +112,5° або від -67,5° до -112,5° - вертикальний кут, від +112,5° до +157,5° або від -22,5° до -67,5° - кут +45°. Тепер потрібно для кожного пікселя зображення знайти напрямок найближчий до $\alpha(x,y)$, перевірити модулі градієнтів сусідніх пікселів по цьому найближчому напрямку, якщо вони менші за модуль градієнта в пікселі (x,y), то в новому зображенні після третього кроку піксель $g_N(x,y)$ буде мати значення модуля градієнта пікселя (x, y) з другого кроку, в іншому

випадку буде мати значення 0. Після цього нове зображення буде містити тільки потоньшені контури, усі немаксимальні точки контурів подавлені (приклад такої обробки зображення можна побачити на Рис. 1.7.2(b)).

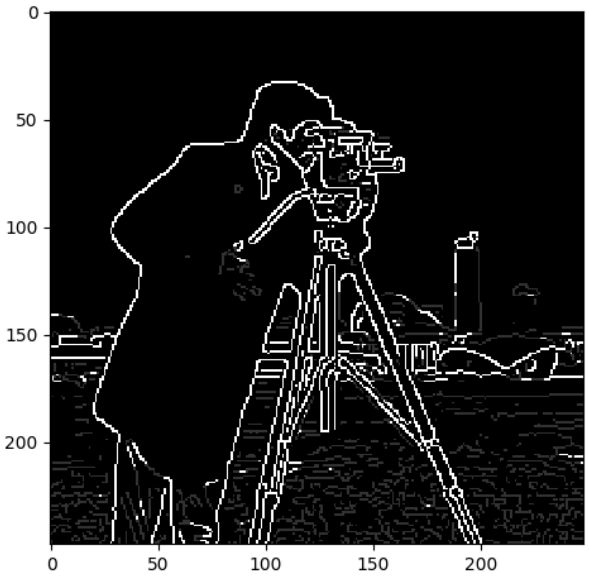
Четвертий крок починається з використання двох порогів T_H та T_L з відношенням величини верхнього порогу до нижньої величини від 2-3 до 1. Будується два зображення $g_{NH}=g_N(x,y)\geq T_H$, $g_{NL}=g_N(x,y)\geq T_L$, потім перетворюємо $g_{NL}=g_{NL}(x,y)-g_{NH}(x,y)$. Таким чином, в g_{NH} містяться “сильні контури”, а в g_{NL} - “слабкі”(приклад такої обробки зображення можна побачити на Рис. 1.7.2(c)). Після цього всі “сильні контури” вважаються точками істинних контурів, а “слабкі” перевіряються: береться піксель p з g_{NH} , пікселі з g_{NL} , які являються зв’язними з p, в сенсі 8-зв’язності(тобно розглядаються всі найближчі пікселі , що оточують p), позначаються, як істинні контури. Так перевіряються всі пікселі з g_{NH} , в кінці пікселі, які не позначені , як істинні зануляються(приклад такої обробки зображення можна побачити на Рис. 1.7.2(d)).



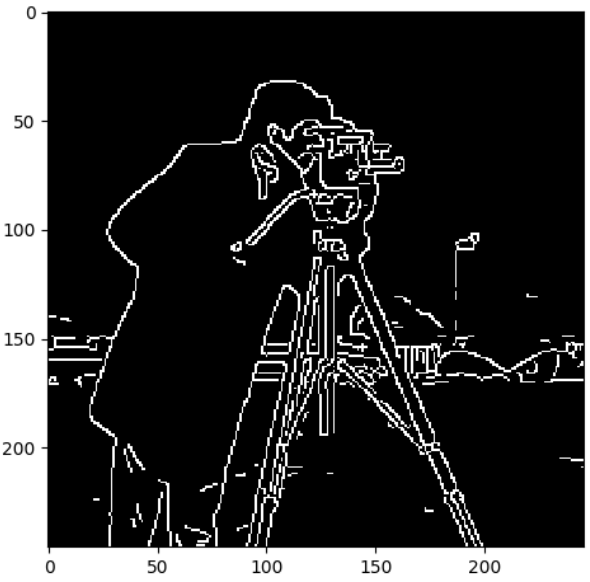
а


0
50
100
150
200
0
50
100
150
200

b


0
50
100
150
200
0
50
100
150
200

c


0
50
100
150
200
0
50
100
150
200

d

Рис. 1.7.2 - Застосування детектора Кенні :
a - Оригінальне фото[10]; b - Застосування придушення немаксимальних точок зображення модуля градієнта зображення a;
c - Перетворення з подвійним порогом (35 , 95) на зображенні b ;
d - Результат роботи детектора Кенні після зв'язування з "слабкими контурами".

Алгоритм Кенні має кращу якість виявлення контурів, але його реалізація більш складна та вимагає більше часу на обчислення. В деяких випадках, де потрібна швидкість обробки зображень у реальному часі, простіші методи, наприклад знаходження градієнту, а потім застосування порогового перетворення, можуть бути більш практичними. Однак, якщо головний пріоритет - якість виявлення контурів, то алгоритми Марра-Хілдрета та Кенні є хорошими альтернативами.

1.8 Застосування лінійного масштабування

Для того, щоб отримати після застосування різних градієнтних алгоритмів в результаті нове зображення, як було зроблено в усіх прикладах вище і застосовуватиметься надалі, необхідно провести, наприклад, лінійне масштабування. Воно порібне через те, що яскравості в зображенні повинні бути в діапазоні 0-255 , але під час проведення різних арифметичних операцій в алгоритмах для знаходження контурів значення яскравостей часто виходять за необхідний діапазон. Отже, відповідно до джерела [1] , якщо u -значення яскравості пікселя, u_{min} - мінімальне значення яскравості в зображенні, u_{max} - максимальне значення яскравості в зображенні, то треба скористатися такою формулою: $g(u)=b(u+a)$, де $a=-u_{min}$, $b=\frac{255}{u_{max}-u_{min}}$. І так проходитимо через кожен піксель.

Розділ 2. Виявлення контурів зображення зі згладжуванням поверхнею першого та другого порядку

2.1 Алгоритм виявлення контурів зображення зі згладжуванням поверхнею першого порядку

Для того, щоб хоча б частково усунути фіктивні контури в зображенні, можна провести згладжування значень яскравості в границях вікна поверхнями першого або другого порядку.

У джерелі [8] було розглянуто метод апроксимації яскравості функцією (поліномом першого порядку) для вікна 2x2

$$\begin{bmatrix} f(x-1,y-1) & f(x-1,y) \\ f(x,y-1) & f(x,y) \end{bmatrix} . \qquad (2.1.1)$$

Для елементів цього вікна побудуємо апроксимуючу площину

$$\hat{f} = ax + by + c . \qquad (2.1.2)$$

Для побудови площини скористаємося методом найменших квадратів[9]. При пошуку коефіцієнтів будемо мінімізувати величину

$$\varepsilon^2 = [\hat{f}(x,y) - f(x,y)]^2 + [\hat{f}(x-1,y) - f(x-1,y)]^2 + [\hat{f}(x,y-1) - f(x,y-1)]^2 + [\hat{f}(x-1,y-1) - f(x-1,y-1)]^2 . \qquad (2.1.3)$$

$$\begin{aligned} \varepsilon^2 = {} & [ax + by + c - f(x,y)]^2 + \\ & + [a(x-1)+by+c-f(x-1,y)]^2 + \\ & + [ax+b(y-1)+c-f(x,y-1)]^2 + \\ & + [a(x-1)+b(y-1)+c-f(x-1,y-1)]^2 . \end{aligned} \qquad (2.1.4)$$

З необхідних умов мінімуму (2.1.4), отримаємо

$$\frac{\partial \varepsilon^2}{\partial a} = 0 , \qquad \frac{\partial \varepsilon^2}{\partial b} = 0 , \qquad \frac{\partial \varepsilon^2}{\partial c} = 0 . \qquad (2.1.5)$$

Тепер знайдемо праві частини рівнянь (2.1.5):

$$\begin{aligned} \frac{\partial \varepsilon^2}{\partial a} = {} & 2[ax + by + c - f(x,y)]x + \\ & + 2[a(x-1) + by + c - f(x-1,y)](x-1) + \\ & + 2[ax+b(y-1)+c-f(x,y-1)]x + \\ & + 2[a(x-1) + b(y-1) + c - f(x-1,y-1)](x-1) . \end{aligned} \qquad (2.1.6)$$

$$\frac{\partial \varepsilon^2}{\partial b} = 2[ax + by + c - f(x,y)]y \; +$$
$$+ 2[a(x-1) + by + c - f(x-1,y)]y \; +$$
$$+ 2[ax + b(y-1) + c - f(x,y-1)](y-1) \; +$$
$$+ 2[a(x-1) + b(y-1) + c - f(x-1,y-1)](y-1) \; . \qquad (2.1.7)$$

$$\frac{\partial \varepsilon^2}{\partial c} = 2[ax + by + c - f(x,y)] \; +$$
$$+ 2[a(x-1) + by + c - f(x-1,y)] \; +$$
$$+ 2[ax + b(y-1) + c - f(x,y-1)] \; +$$
$$+ 2[a(x-1) + b(y-1) + c - f(x-1,y-1)] \; . \qquad (2.1.8)$$

Тепер розкриємо дужки і винесемо спільні множники в (2.1.6) – (2.1.8):

$$\frac{\partial \varepsilon^2}{\partial a} = a(8x^2 - 8x + 4) + b(8xy - 4x - 4y + 2) + c(8x - 4) -$$
$$- 2xf(x,y) - 2xf(x,y-1) + (2-2x)f(x-1,y) + (2-2x)f(x-1,y-1) \qquad (2.1.9)$$

$$\frac{\partial \varepsilon^2}{\partial b} = a(8xy - 4x - 4y + 2) + b(8y^2 - 8y + 4) + c(8y - 4) -$$
$$- 2yf(x,y) - 2yf(x-1,y) + (2-2y)f(x,y-1) + (2-2y)f(x-1,y-1) \qquad (2.1.10)$$

$$\frac{\partial \varepsilon^2}{\partial c} = a(8x - 4) + b(8y - 4) + 8c -$$
$$- 2f(x,y) - 2f(x,y-1) - 2f(x-1,y) - 2f(x-1,y-1) \qquad (2.1.11)$$

Тепер прирівняємо (2.1.9) – (2.1.11) до нулів відповідно до (2.1.5) , так отримуємо систему лінійних алгебраїчних рівнянь відносно невідомих коефіцієнтів a, b, c .Розв'язками цієї системи будуть такі коефіцієнти a, b, c:

$$a = 0.5[f(x,y) + f(x,y-1) - f(x-1,y) - f(x-1,y-1)] \; , \qquad (2.1.12)$$

$$b = 0.5[f(x,y) + f(x-1,y) - f(x,y-1) - f(x-1,y-1)] \; , \qquad (2.1.13)$$

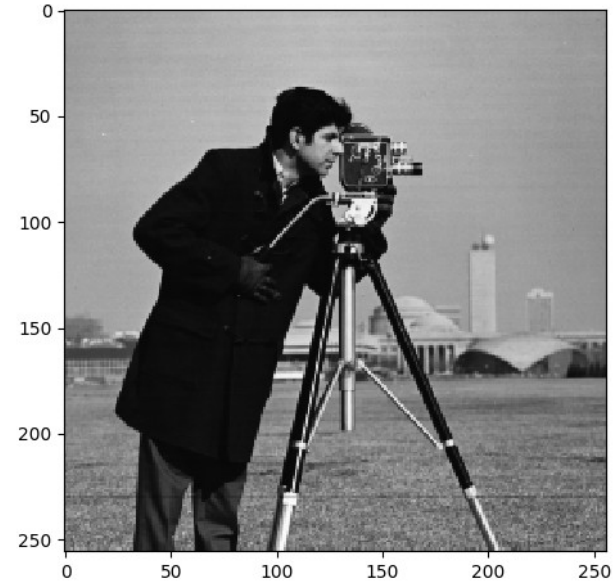
$$c = 0.25[-2xf(x,y) - 2xf(x,y-1) + 2xf(x-1,y) + 2xf(x-1,y-1) -$$
$$-2yf(x,y) + 2yf(x,y-1) - 2yf(x-1,y) + 2yf(x-1,y-1) + 3f(x,y) +$$
$$+f(x,y-1) + f(x-1,y) - f(x-1,y-1)] \quad . \qquad (2.1.14)$$

Якщо відомі a,b,c, тобто відома й площина,то можна знайти частинні похідні:

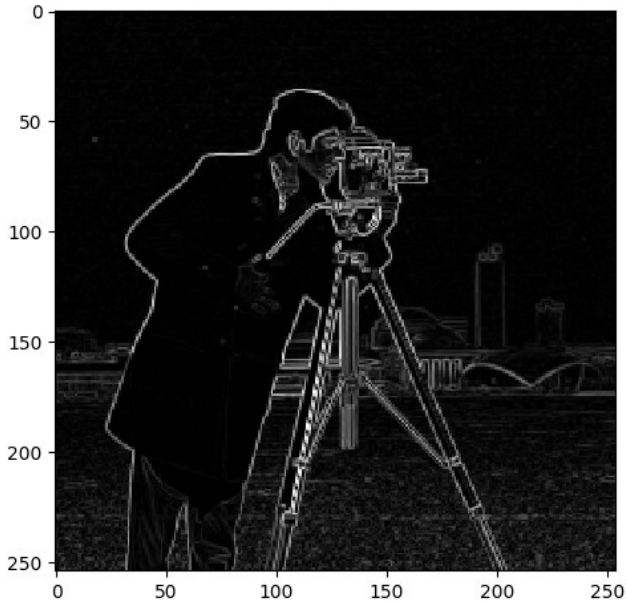
$$\frac{\partial \hat{f}}{\partial x} = a \quad , \qquad \frac{\partial \hat{f}}{\partial y} = b \quad . \qquad (2.1.15)$$

Зрештою, знаходимо модуль градієнта, щоб виявити локальні перепади яскравості:

$$\left|\nabla \hat{f}(x,y)\right| = \sqrt{a^2+b^2} \quad . \qquad (2.1.16)$$



a



b

Рис. 2.1.1 - Приклад використання методу для знаходження контурів

a - Оригінальне зображення [10]; b - Дія алгоритму виявлення контурів зображення зі згладжуванням поверхнею першого порядку.

2.2 Алгоритм виявлення контурів зображення зі згладжуванням поверхнею другого порядку

На основі попереднього алгоритму тепер розглянемо метод апроксимації яскравості поверхнею другого порядку $\hat{f} = ax^2 + by^2 + cxy + \alpha x + \beta y + \gamma$. (2.2.1) Для цього обирається вікно 3x3

$$\begin{bmatrix} f(x-1,y-1) & f(x-1,y) & f(x-1,y+1) \\ f(x,y-1) & f(x,y) & f(x,y+1) \\ f(x+1,y-1) & f(x+1,y) & f(x+1,y+1) \end{bmatrix} . \qquad (2.2.2)$$

Так само , як і при апроксимації площиною , скористаємося методом найменших квадратів. Мінімізуватимемо величину ε^2 .

$$\varepsilon^2 = [\hat{f}(x,y) - f(x,y)]^2 + [\hat{f}(x-1,y) - f(x-1,y)]^2 +$$
$$+ [\hat{f}(x,y-1) - f(x,y-1)]^2 + [\hat{f}(x-1,y-1) - f(x-1,y-1)]^2 +$$
$$+ [\hat{f}(x-1,y+1) - f(x-1,y+1)]^2 + [\hat{f}(x,y+1) - f(x-1,y+1)]^2 +$$
$$+ [\hat{f}(x+1,y-1) - f(x+1,y-1)]^2 + [\hat{f}(x+1,y) - f(x+1,y)]^2 +$$
$$+ [\hat{f}(x+1,y+1) - f(x+1,y+1)]^2 . \qquad (2.2.3)$$

Підставимо відповідні $\hat{f}$ в формулу (2.2.3):

$$\varepsilon^2 = [ax^2 + by^2 + cxy + \alpha x + \beta y + \gamma - f(x,y)]^2 +$$
$$+ [a(x-1)^2 + by^2 + c(x-1)y + \alpha(x-1) + \beta y + \gamma - f(x-1,y)]^2 +$$
$$+ [ax^2 + b(y-1)^2 + cx(y-1) + \alpha x + \beta(y-1) + \gamma - f(x,y-1)]^2 +$$
$$+ [a(x-1)^2 + b(y-1)^2 + c(x-1)(y-1) + \alpha(x-1) + \beta(y-1) + \gamma -$$
$$- f(x-1,y-1)]^2 + [a(x-1)^2 + b(y+1)^2 + c(x-1)(y+1) +$$
$$+ \alpha(x-1) + \beta(y+1) + \gamma - f(x-1,y+1)]^2 + [ax^2 + b(y+1)^2 +$$
$$+ cx(y+1) + \alpha x + \beta(y+1) + \gamma - f(x,y+1)]^2 +$$
$$+ [a(x+1)^2 + b(y-1)^2 + c(x+1)(y-1) + \alpha(x+1) + \beta(y-1) + \gamma -$$

$$- f(x+1,y-1)]^2 + [a(x+1)^2 + by^2 + c(x+1)y + \alpha(x+1) + \beta y +$$
$$+ \gamma + f(x+1,y)]^2 + [a(x+1)^2 + b(y+1)^2 + c(x+1)(y+1) + \alpha(x+1) +$$
$$+ \beta(y+1) + \gamma - f(x+1,y+1)]^2 \ . \qquad (2.2.4)$$

З необхідних умов мінімуму (2.2.4), отримаємо

$$\frac{\partial \varepsilon^2}{\partial a} = 0 \ , \qquad \frac{\partial \varepsilon^2}{\partial b} = 0 \ , \qquad \frac{\partial \varepsilon^2}{\partial c} = 0$$
$$\frac{\partial \varepsilon^2}{\partial \alpha} = 0 \ , \qquad \frac{\partial \varepsilon^2}{\partial \beta} = 0 \ , \qquad \frac{\partial \varepsilon^2}{\partial \gamma} = 0 \ . \qquad (2.2.5)$$

Порахуємо необхідні похідні програмно, запишемо результати для компактності запису з заміною $x-1=l, x+1=r, y-1=j, y+1=h$:

$$\frac{\partial \varepsilon^2}{\partial a} = 2*x^2*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*x^2*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*x^2*(\alpha*x+\beta*(j)+\gamma+a*x^2+b*(j)^2+c*x*(j)-f(x,j))+$$
$$+2*x^2*(\alpha*x+\beta*(h)+\gamma+a*x^2+b*(h)^2+c*x*(h)-f(x,h))+$$
$$+2*(l)^2*(\alpha*(l)+\beta*y+\gamma+a*(l)^2+b*y^2+c*y*(l)-f(l,y))+$$
$$+2*(l)^2*(\alpha*(l)+\beta*(j)+\gamma+a*(l)^2+b*(j)^2+c*(l)*(j)-f(l,j))+$$
$$+2*(l)^2*(\alpha*(l)+\beta*(h)+\gamma+a*(l)^2+b*(h)^2+c*(l)*(h)-f(l,h))+$$
$$+2*(r)^2*(\alpha*(r)+\beta*y+\gamma+a*(r)^2+b*y^2+c*y*(r)-f(r,y))+$$
$$+2*(r)^2*(\alpha*(r)+\beta*(j)+\gamma+a*(r)^2+b*(j)^2+c*(r)*(j)-f(r,j))+$$
$$+2*(r)^2*(\alpha*(r)+\beta*(h)+\gamma+a*(r)^2+b*(h)^2+c*(r)*(h)-f(r,h))$$

(2.2.6)

$$\frac{\partial \varepsilon^2}{\partial b} = 2*y^2*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*y^2*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*y^2*(\alpha*(l)+\beta*y+\gamma+a*(l)^2+b*y^2+c*y*(l)-f(l,y))+$$
$$+2*y^2*(\alpha*(r)+\beta*y+\gamma+a*(r)^2+b*y^2+c*y*(r)-f(r,y))+$$
$$+2*(j)^2*(\alpha*x+\beta*(j)+\gamma+a*x^2+b*(j)^2+c*x*(j)-f(x,j))+$$
$$+2*(j)^2*(\alpha*(l)+\beta*(j)+\gamma+a*(l)^2+b*(j)^2+c*(l)*(j)-f(l,j))+$$
$$+2*(j)^2*(\alpha*(r)+\beta*(j)+\gamma+a*(r)^2+b*(j)^2+c*(r)*(j)-f(r,j))+$$
$$+2*(h)^2*(\alpha*x+\beta*(h)+\gamma+a*x^2+b*(h)^2+c*x*(h)-f(x,h))+$$
$$+2*(h)^2*(\alpha*(l)+\beta*(h)+\gamma+a*(l)^2+b*(h)^2+c*(l)*(h)-f(l,h))+$$
$$+2*(h)^2*(\alpha*(r)+\beta*(h)+\gamma+a*(r)^2+b*(h)^2+c*(r)*(h)-f(r,h))$$

(2.2.7)

$$\frac{\partial \varepsilon^2}{\partial c} = 2*x*y*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*x*y*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*x*(j)*(\alpha*x+\beta*(j)+\gamma+a*x^2+b*(j)^2+c*x*(j)-f(x,j))+$$
$$+2*x*(h)*(\alpha*x+\beta*(h)+\gamma+a*x^2+b*(h)^2+c*x*(h)-f(x,h))+$$
$$+2*y*(l)*(\alpha*(l)+\beta*y+\gamma+a*(l)^2+b*y^2+c*y*(l)-f(l,y))+$$
$$+2*y*(r)*(\alpha*(r)+\beta*y+\gamma+a*(r)^2+b*y^2+c*y*(r)-f(r,y))+$$
$$+2*(l)*(j)*(\alpha*(l)+\beta*(j)+\gamma+a*(l)^2+b*(j)^2+c*(l)*(j)-f(l,j))+$$
$$+2*(l)*(h)*(\alpha*(l)+\beta*(h)+\gamma+a*(l)^2+b*(h)^2+c*(l)*(h)-f(l,h))+$$
$$+2*(r)*(j)*(\alpha*(r)+\beta*(j)+\gamma+a*(r)^2+b*(j)^2+c*(r)*(j)-f(r,j))+$$
$$+2*(r)*(h)*(\alpha*(r)+\beta*(h)+\gamma+a*(r)^2+b*(h)^2+c*(r)*(h)-f(r,h))$$

(2.2.8)

$$\frac{\partial \varepsilon^2}{\partial \alpha} = 2*x*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*x*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*x*(\alpha*x+\beta*(j)+\gamma+a*x^2+b*(j)^2+c*x*(j)-f(x,j))+$$
$$+2*x*(\alpha*x+\beta*(h)+\gamma+a*x^2+b*(h)^2+c*x*(h)-f(x,h))+$$
$$+(2*x-2)*(\alpha*(l)+\beta*y+\gamma+a*(l)^2+b*y^2+c*y*(l)-f(l,y))+$$
$$+(2*x-2)*(\alpha*(l)+\beta*(j)+\gamma+a*(l)^2+b*(j)^2+c*(l)*(j)-f(l,j))+$$
$$+(2*x-2)*(\alpha*(l)+\beta*(h)+\gamma+a*(l)^2+b*(h)^2+c*(l)*(h)-f(l,h))+$$
$$+(2*x+2)*(\alpha*(r)+\beta*y+\gamma+a*(r)^2+b*y^2+c*y*(r)-f(r,y))+$$
$$+(2*x+2)*(\alpha*(r)+\beta*(j)+\gamma+a*(r)^2+b*(j)^2+c*(r)*(j)-f(r,j))+$$
$$+(2*x+2)*(\alpha*(r)+\beta*(h)+\gamma+a*(r)^2+b*(h)^2+c*(r)*(h)-f(r,h))$$

(2.2.9)

$$\frac{\partial \varepsilon^2}{\partial \beta} = 2*y*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*y*(\alpha*x+\beta*y+\gamma+a*x^2+b*y^2+c*x*y-f(x,y))+$$
$$+2*y*(\alpha*(l)+\beta*y+\gamma+a*(l)^2+b*y^2+c*y*(l)-f(l,y))+$$
$$+2*y*(\alpha*(r)+\beta*y+\gamma+a*(r)^2+b*y^2+c*y*(r)-f(r,y))+$$
$$+(2*y-2)*(\alpha*x+\beta*(j)+\gamma+a*x^2+b*(j)^2+c*x*(j)-f(x,j))+$$
$$+(2*y-2)*(\alpha*(l)+\beta*(j)+\gamma+a*(l)^2+b*(j)^2+c*(l)*(j)-f(l,j))+$$
$$+(2*y-2)*(\alpha*(r)+\beta*(j)+\gamma+a*(r)^2+b*(j)^2+c*(r)*(j)-f(r,j))+$$
$$+(2*y+2)*(\alpha*x+\beta*(h)+\gamma+a*x^2+b*(h)^2+c*x*(h)-f(x,h))+$$
$$+(2*y+2)*(\alpha*(l)+\beta*(h)+\gamma+a*(l)^2+b*(h)^2+c*(l)*(h)-f(l,h))+$$
$$+(2*y+2)*(\alpha*(r)+\beta*(h)+\gamma+a*(r)^2+b*(h)^2+c*(r)*(h)-f(r,h))$$

(2.2.10)

$$\frac{\partial \varepsilon^2}{\partial \gamma} = 6*\alpha*x+6*\alpha*(l)+6*\alpha*(r)+6*\beta*y+6*\beta*(j)+6*\beta*(h)+$$
$$+18*\gamma+6*a*x^2+6*a*(l)^2+6*a*(r)^2+6*b*y^2+6*b*(j)^2+$$
$$+6*b*(h)^2+2*c*x*y+2*c*x*(j)+2*c*x*(h)+2*c*y*(l)+2*c*y*(r)+$$
$$+2*c*(l)*(j)+2*c*(l)*(h)+2*c*(r)*(j)+2*c*(r)*(h)-2*f(x,y)-2*f(x,j)-$$
$$-2*f(x,h)-2*f(l,y)-2*f(l,j)-2*f(l,h)-2*f(r,y)-2*f(r,j)-$$
$$-2*f(r,h)$$

(2.2.11)

Далі розглядаємо рівняння (2.2.6) – (2.2.11). Розкриваємо дужки і групуємо за змінними $a, b, c, \alpha, \beta, \gamma, f(l,j), f(l,y), f(l,h), f(x,j), f(x,y), f(x,h), f(r,j), f(r,y), f(r,h)$.

Отримуємо:

$$\frac{\partial \varepsilon^2}{\partial a} = \alpha*(18*x^3+36*x)+\beta*(18*x^2*y+12*y)+\gamma*(18*x^2+12)+$$
$$+a*(18*x^4+72*x^2+12)+b*(18*x^2*y^2+12*x^2+12*y^2+8)+$$
$$+c*(18*x^3*y+36*x*y)-2*x^2*f(x,y)-2*x^2*f(x,j)-$$
$$-2*x^2*f(x,h)+(-2*x^2-4*x-2)*f(r,y)+(-2*x^2-4*x-2)*f(r,j)+$$
$$+(-2*x^2-4*x-2)*f(r,h)+(-2*x^2+4*x-2)*f(l,y)+$$
$$+(-2*x^2+4*x-2)*f(l,j)+(-2*x^2+4*x-2)*f(l,h)$$

(2.2.12)

$$\frac{\partial \varepsilon^2}{\partial b} = \alpha*(18*x*y^2+12*x)+\beta*(18*y^3+36*y)+$$
$$+\gamma*(18*y^2+12)+a*(18*x^2*y^2+12*x^2+12*y^2+8)+b*(18*y^4+72*y^2+12)+$$
$$+c*(18*x*y^3+36*x*y)-2*y^2*f(x,y)-2*y^2*f(l,y)-$$
$$-2*y^2*f(r,y)+(-2*y^2-4*y-2)*f(x,h)+(-2*y^2-4*y-2)*f(l,h)+$$
$$+(-2*y^2-4*y-2)*f(r,h)+(-2*y^2+4*y-2)*f(x,j)+$$
$$+(-2*y^2+4*y-2)*f(l,j)+(-2*y^2+4*y-2)*f(r,j)$$

(2.2.13)

$$\frac{\partial \varepsilon^2}{\partial c} = \alpha*(18*x^2*y+12*y)+\beta*(18*x*y^2+12*x)+$$
$$+18*\gamma*x*y+a*(18*x^3*y+36*x*y)+b*(18*x*y^3+36*x*y)+$$
$$+c*(18*x^2*y^2+12*x^2+12*y^2+8)-2*x*y*f(x,y)-$$
$$-(-2*x*y-2*x)*f(x,h)+(-2*x*y+2*x)*f(x,j)+(-2*x*y-2*y)*f(r,y)+$$
$$+(-2*x*y+2*y)*f(l,y)+(-2*x*y-2*x-2*y-2)*f(r,h)+$$
$$+(-2*x*y-2*x+2*y+2)*f(l,h)+$$
$$+(-2*x*y+2*x-2*y+2)*f(r,j)+(-2*x*y+2*x+2*y-2)*f(l,j)$$

(2.2.14)

$$\frac{\partial \varepsilon^2}{\partial \alpha} = \alpha*(18*x^2+12)+18*\beta*x*y+18*\gamma*x+$$
$$+a*(18*x^3+36*x)+b*(18*x*y^2+12*x)+$$
$$+c*(18*x^2*y+12*y)-2*x*f(x,y)-2*x*f(x,j)-2*x*f(x,h)+$$
$$+(2-2*x)*f(l,y)+(2-2*x)*f(l,j)+(2-2*x)*f(l,h)+$$
$$+(-2*x-2)*f(r,y)+(-2*x-2)*f(r,j)+(-2*x-2)*f(r,h)$$

(2.2.15)

$$\frac{\partial \varepsilon^2}{\partial \beta} = 18*\alpha*x*y+\beta*(18*y^2+12)+18*\gamma*y+$$
$$+a*(18*x^2*y+12*y)+b*(18*y^3+36*y)+c*(18*x*y^2+12*x)-$$
$$-2*y*f(x,y)-2*y*f(l,y)-2*y*f(r,y)+(2-2*y)*f(x,j)+$$
$$+(2-2*y)*f(l,j)+(2-2*y)*f(r,j)+(-2*y-2)*f(x,h)+$$
$$+(-2*y-2)*f(l,h)+(-2*y-2)*f(r,h)$$

(2.2.16)

$$\frac{\partial \varepsilon^2}{\partial \gamma} = 18*\alpha*x+18*\beta*y+18*\gamma+$$
$$+a*(18*x^2+12)+b*(18*y^2+12)+18*c*x*y-2*f(x,y)-$$
$$-2*f(x,j)-2*f(x,h)-2*f(l,y)-2*f(l,j)-2*f(l,h)-$$
$$-2*f(r,y)-2*f(r,j)-2*f(r,h)$$

(2.2.17)

Тепер , знаючи похідні (2.2.12) – (2.2.17), можна приступити до системи (2.2.5). Знайти невідомі коефіцієнти a, b, c, α, β, γ можна різними методами. В даному випадку знайдемо їх програмно за допомогою бібліотеки

sympy мови програмування python. З неї використаємо метод solve(). Отримаємо такі результати:

$$a = -\frac{f(x,y)}{3} - \frac{f(x,j)}{3} - \frac{f(x,h)}{3} + \frac{f(l,y)}{6} + \\ + \frac{f(l,j)}{6} + \frac{f(l,h)}{6} + \frac{f(r,y)}{6} + \frac{f(r,j)}{6} + \frac{f(r,h)}{6}$$

(2.2.18)

$$b = -\frac{f(x,y)}{3} + \frac{f(x,j)}{6} + \frac{f(x,h)}{6} - \frac{f(l,y)}{3} + \\ + \frac{f(l,j)}{6} + \frac{f(l,h)}{6} - \frac{f(r,y)}{3} + \frac{f(r,j)}{6} + \frac{f(r,h)}{6}$$

(2.2.19)

$$c = \frac{f(l,j)}{4} - \frac{f(l,h)}{4} - \frac{f(r,j)}{4} + \frac{f(r,h)}{4}$$

(2.2.20)

$$\alpha = \frac{2*x*f(x,y)}{3} + \frac{2*x*f(x,j)}{3} + \frac{2*x*f(x,h)}{3} - \frac{x*f(l,y)}{3} - \frac{x*f(l,j)}{3} - \\ - \frac{x*f(l,h)}{3} - \frac{x*f(r,y)}{3} - \frac{x*f(r,j)}{3} - \frac{x*f(r,h)}{3} - \frac{y*f(l,j)}{4} + \frac{y*f(l,h)}{4} + \\ + \frac{y*f(r,j)}{4} - \frac{y*f(r,h)}{4} - \frac{f(l,y)}{6} - \frac{f(l,j)}{6} - \frac{f(l,h)}{6} + \frac{f(r,y)}{6} + \frac{f(r,j)}{6} + \frac{f(r,h)}{6}$$

(2.2.21)

$$\beta = -\frac{x*f(l,j)}{4} + \frac{x*f(l,h)}{4} + \frac{x*f(r,j)}{4} - \frac{x*f(r,h)}{4} + \frac{2*y*f(x,y)}{3} - \\ - \frac{y*f(x,j)}{3} - \frac{y*f(x,h)}{3} + \frac{2*y*f(l,y)}{3} - \frac{y*f(l,j)}{3} - \frac{y*f(l,h)}{3} + \frac{2*y*f(r,y)}{3} - \\ - \frac{y*f(r,j)}{3} - \frac{y*f(r,h)}{3} - \frac{f(x,j)}{6} + \frac{f(x,h)}{6} - \frac{f(l,j)}{6} + \frac{f(l,h)}{6} - \frac{f(r,j)}{6} + \frac{f(r,h)}{6}$$

(2.2.22)

$$\gamma = -\frac{x^2*f(x,y)}{3}-\frac{x^2*f(x,j)}{3}-\frac{x^2*f(x,h)}{3}+\frac{x^2*f(l,y)}{6}+\frac{x^2*f(l,j)}{6}+$$
$$+\frac{2*f(l,h)}{6}+\frac{x^2*f(r,y)}{6}+\frac{x^2*f(r,j)}{6}+\frac{x^2*f(r,h)}{6}+\frac{x*y*f(l,j)}{4}-\frac{x*y*f(l,h)}{4}-$$
$$-\frac{x*y*f(r,j)}{4}+\frac{x*y*f(r,h)}{4}+\frac{x*f(l,y)}{6}+\frac{x*f(l,j)}{6}+\frac{x*f(l,h)}{6}-\frac{x*f(r,y)}{6}-$$
$$-\frac{x*f(r,j)}{6}-\frac{x*f(r,h)}{6}-\frac{y^2*f(x,y)}{3}+\frac{y^2*f(x,j)}{6}+\frac{y^2*f(x,h)}{6}-\frac{y^2*f(l,y)}{3}+$$
$$+\frac{y^2*f(l,j)}{6}+\frac{y^2*f(l,h)}{6}-\frac{y^2*f(r,y)}{3}+\frac{y^2*f(r,j)}{6}+\frac{y^2*f(r,h)}{6}+\frac{y*f(x,j)}{6}-$$
$$-\frac{y*f(x,h)}{6}+\frac{y*f(l,j)}{6}-\frac{y*f(l,h)}{6}+\frac{y*f(r,j)}{6}-\frac{y*f(r,h)}{6}+\frac{5*f(x,y)}{9}+$$
$$+\frac{2*f(x,j)}{9}+\frac{2*f(x,h)}{9}+\frac{2*f(l,y)}{9}-\frac{f(l,j)}{9}-\frac{f(l,h)}{9}+\frac{2*f(r,y)}{9}-\frac{f(r,j)}{9}-\frac{f(r,h)}{9} \quad (2.2.23)$$

Тепер можна записати частинні похідні $\frac{\partial \hat{f}}{\partial x}$ та $\frac{\partial \hat{f}}{\partial y}$:

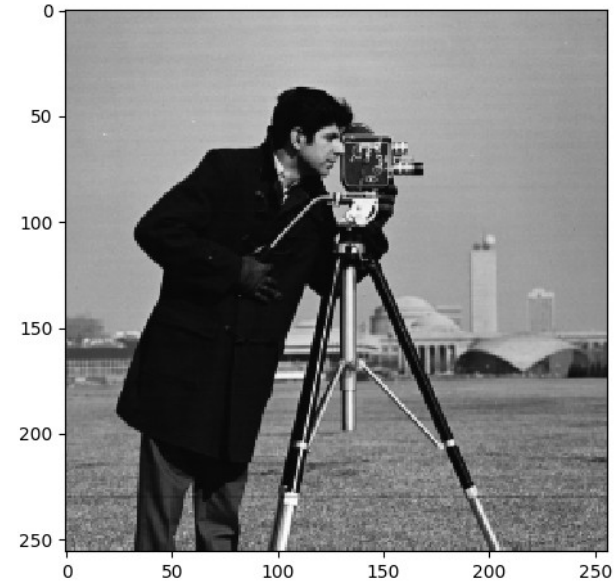
$$\frac{\partial \hat{f}}{\partial x} = 2ax + cy + \alpha \ , \qquad \frac{\partial \hat{f}}{\partial y} = 2by + cx + \beta \ . \quad (2.2.24)$$

І тоді модуль градієнта є ознакою локального перепаду яскравості:

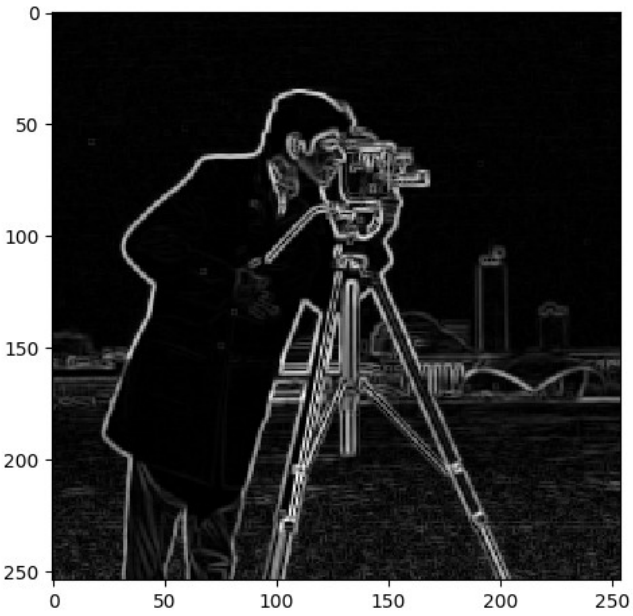
$$\left|\nabla \hat{f}(x,y)\right| = \sqrt{(2ax + cy + \alpha)^2 + (2by + cx + \beta)^2} \ . \quad (2.2.25)$$

Для реалізації цього методу програмно для знаходження контурів в зображенні будемо використовувати тільки знайдені коефіцієнти a, b, c, α, β (2.2.18) – (2.2.22) та, знаючи їх, будемо знаходити модуль градієнта (2.2.25). Ці операції будемо проводити у всіх пікселях зображення, окрім країв .

Розглянемо дію цього алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку на зображенні . (Рис. 2.2.1)



a



b

Рис. 2.2.1 - Приклад використання методу для знаходження контурів
a - Оригінальне зображення [10]; b - Дія алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку.

2.3 Результати застосування алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку

Для демонстрації роботи методу було проведено низку експериментів на різних зображеннях. Результати показують, що метод виявлення контурів зображення зі згладжуванням поверхнею другого порядку виявляє контури з

високою точністю. Він здатний виявляти контури різних форм та розмірів. Розглянемо приклади його роботи на Рис 2.3.1, Рис 2.3.2, Рис 2.3.3, Рис. 2.3.4.

a

b

Рис. 2.3.1 - Приклад використання методу для знаходження контурів
a - Оригінальне зображення [11]; b - Дія алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку.

a

b

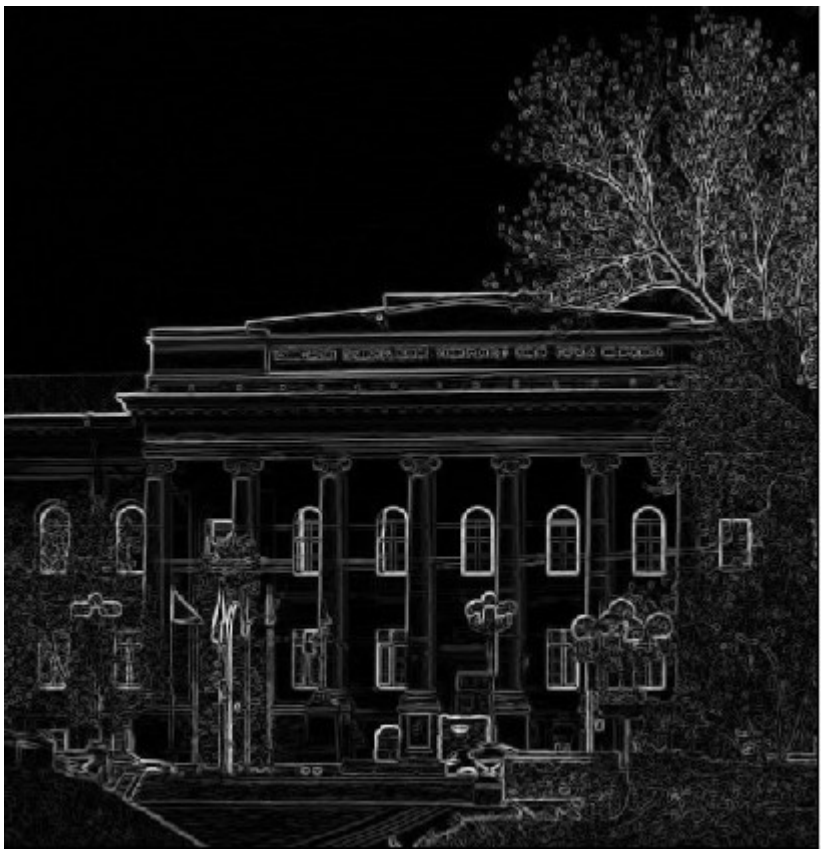
Рис. 2.3.2 - Приклад використання методу для знаходження контурів
a - Оригінальне зображення [11]; b - Дія алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку.

a

b

Рис. 2.3.3 - Приклад використання методу для знаходження контурів
a - Оригінальне зображення[11]; b - Дія алгоритму виявлення контурів
зображення зі згладжуванням поверхнею другого порядку.

a



b

Рис. 2.3.4 - Приклад використання методу для знаходження контурів
a - Оригінальне зображення[11]; b - Дія алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку.

Задача порівняння методів виділення контурів є зовсім не тривіальною та потребує свого окремого величезного дослідження. Зараз можна тільки, проводячи експерименти на різних фотознімках, виділити певні відмінності в результатах роботи методів. Спостерігаючи вихідні зображення після застосування алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку та інших відомих методів, можна сказати, що результати схожі до результатів, отриманих при використанні операторів Собела та Превітта. Всі три методи забезпечують високу чутливість до контурів об’єктів на зображеннях, часто дають повністю однакові результати. Часто на трохи зашумлених зображеннях помітний більший вплив згладжування при застосуванні методу виявлення контурів зображення зі згладжуванням поверхнею другого порядку, що призводить до виникнення меншої кількості фіктивних границь, в порівнянні з застосуванням методу Собела. Щодо порівння з роботою оператора Преввіта , можна сказати, що їх результати ще більш схожі, дуже часто повністю однакові. А загалом в більшості випадків, які потраплялися при експериментах і не були повністю однаковими, можна було найти поріг, при якому виділялася менша кількість елементів, що не є контурами. Отже, можна припустити і очікувати, що через особливості своєї побудови алгоритм виявлення контурів зображення зі згладжуванням поверхнею другого порядку в більшості випадків трохи сильніше згладжує зображення і тому виникне меньше фіктивних контурів. Проте це необхідно додатково перевірити на дуже великому наборі даних, знайти всі відомі методи порівнння методів виділення контурів, крім того ці методи ймовірно потребуватимуть еталонні контури з якими можна порівнювати, тобто знадобиться й великий набір зображень, для яких

втановлені справжні контури. Тому деякі з подальших досліджень можуть бути спрямовані на обгрунтування особливостей роботи даного алгоритму, детальне вивчення неочевидних ситуацій, коли найкраще його використати.

Варто зауважити, що в будь якому випадку окрім основних ліній на вихідному зображенні після застосування алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку , методів Собела та Превітта, з’являються зайві лінії, що не є контурами. Як і згадувалось раніше , можливо необхідно спочатку провести додатково згладжування, також можна застосувати порогову функцію. Більш того, в складніших та ефективніших методах знаходження контурів в такому, як детектор Кенні, проводять ще потоньшення контурів (придушення немаксимальних точок зображення модуля градієнта), застосовуються два пороги, а не один , також проводиться аналіз зв’язності виявленого і зв’язування контурів. Тобто, щоб отримати більш кращий результат методом виявлення контурів зображення зі згладжуванням поверхнею другого порядку, необхідно проводити ще додаткові відомі маніпуляції або ж, можливо, розробити нові. Тобто подальші дослідження також можуть бути спрямовані на оптимізацію методу та його адаптацію для конкретних випадків.

Висновки

В рамках цієї роботи **було програмно реалізовано** відомі методи пошуку контурів в зображенні: Робертса, Превітта, Собела, Кірша; згладжування зображення: прямокутний фільтр, фільтр Гауса; застосування порогової функції; більш складні методи пошуку контурів в зображенні: Марра-Хілдрета та Кенні .

Також в роботі було **розроблено та програмно реалізовано** алгоритм виявлення контурів зображення зі згладжуванням поверхнею другого порядку. Він полягав в тому, щоб апроксимувати значення яскравості в центральному пікселі вікна 3x3 поверхнею другого порядку, використовуючи значення яскравостей згаданого та сусідніх пікселів.

Далі в в роботі **було проаналізовано** результати застосування алгоритму виявлення контурів зображення зі згладжуванням поверхнею другого порядку. Він виявився здатним виявляти контури не гірше за інші популярні градієнтні методи, показавши результати схожі з результатами дії оператора Превітта або Собела. Але згідно спостережень за експериментальними даними та в силу його побудови очікується, що алгоритм виявлення контурів зображення зі згладжуванням поверхнею другого порядку трохи сильніше згладжує зображення і тому виникне меньше фіктивних контурів.

Подальші дослідження можуть бути спрямовані на детальне вивчення ситуацій, коли найкраще його використати, також на оптимізацію методу та його адаптацію для конкретних випадків.

Список використаних джерел

1. **Reinhard Klette**. Coincise Computer Vision. An Introduction into Theory and Algorithms.[Книга] ./ Springer, 2014. - 429 p.

2. **Rafael C. Gonzalez, Richard E. Woods**. Digital Image Processing / 3rd ed. [Книга], Person , 2008. - 976p.

3. **E. R. Davies.** Computer and Machine Vision: Theory, Algorithms, Practicalities / 4th ed.[Книга], Academic Press, 2012. - 832 p.

4. **Richard Szeliski.** Computer Vision: Algorithms and Applications. [Книга]./ Springer, 2010. - 979p.

5. **William K. Pratt.** Digital Image Processing / 3rd ed.[Книга], John Wiley & Sons, Inc., 2001. - 738p.

6. Theory of edge detection by D. Marr and E. Hildret.[Електронний ресурс]. URL:https://www.hms.harvard.edu/bss/neuro/bornlab/qmbc/beta/day4/marr-hildreth-edge-prsl1980.pdf

7. Edge Detection – The Canny Algorithm. A description of the Canny edge detection method. [Електронний ресурс] . URL: https://www.futurelearn.com/info/courses/introduction-to-image-analysis-for-plant-phenotyping/0/steps/302632#:~:text=Canny%20identified%20three%20requirements%20of%20an%20optimal%20edge,%E2%80%93%20each%20edge%20should%20be%20marked%20exactly%20once

8. Лекция 14. Методы выделения перепадов яркости с согласованием. [Електронний ресурс]. URL: https://helpiks.org/6-19185.html

9. **Christopher Dougherty**. Introduction to Econometrics . [Книга] / Oxford University Press, 1992. - 399p.

10. cameraman.tif. [Електронний ресурс]. URL: https://dome.mit.edu/handle/1721.3/195767

11. Pexels . [Електронний ресурс]. URL: https://www.pexels.com