МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут автоматики і електротехніки

Кафедра комп'ютерних технологій та інформаційної безпеки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних технологій та інформаційної безпеки, к.т.н., доцент
_____ С.М. Нужний
«19» ____12____ 2024 р.

Кваліфікаційна робота
на правах рукопису

КВАЛІФІКАЦІЙНА РОБОТА

Розробка програмного засобу зменшення розмитості зображення

Ступінь вищої освіти: магістр
Галузь знань: 14 Електрична інженерія
Спеціальність: 141 «Електроенергетика, електротехніка та електромеханіка»
Освітньо-професійна програма: Системи технічного захисту інформації, автоматизація її обробки

Виконав: студент 6 курсу групи 6367м
Дацко Олександр Юрійович _____

Кваліфікаційна робота містить результати самостійного виконання навчального завдання. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Керівник:
к.т.н., доцент кафедри КТІБ

Сірівчук Андрій Сергійович _____

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут автоматики і електротехніки

ЗАТВЕРДЖУЮ
Гарант освітньої програми, к.т.н., доцент, доцент кафедри КТІБ
_____ Турти М.В.
«_19_» ____12____ 2024 р.

ЗАВДАННЯ

на кваліфікаційну роботу

Студент: Дацко Олександр Юрійович

Курс:6

Група: 6366м

Ступінь вищої освіти: магістр

Галузь знань: 14 Електрична інженерія

Спеціальність: 141 «Електроенергетика, електротехніка та електромеханіка»

Освітньо-професійна програма: Системи технічного захисту інформації, автоматизація її обробки

1. Тема роботи: Розробка програмного засобу зменшення розмитості зображення

затверджена наказом НУК від «_14__» жовтня 2024 р. № 1075 - уч

2. Вихідні дані до роботи: об'єкт дослідження – програмний засіб для усунення розмитості зображення; тип розмитості – розмитість в русі та розфокусування.

3. Перелік питань, які необхідно розробити: описати причини розмиття зображення; визначити алгоритми усунення розмиття; провести аналіз алгоритму; створити програмну реалізацію визначеного алгоритму.

4. Терміни виконання завдання:
термін видачі завдання: «_04_» вересня 2024 р.
термін подання роботи: «_19_» грудня 2024 р.

6. План-графік виконання кваліфікаційної роботи

№ п/п	Заходи	Дата		Готовність	Відмітка про виконання
		Навч. тиждень	Контрольна дата		
1.	Наукове стажування	1 - 8	25.10.2024	Звіт з наукового стажування	
2.	Організаційні збори	9	31.10.2024	Попередній варіант змісту КР	
3.	Рубіжний контроль	10	05.11.2024	Попередній варіант оглядових розділів	
4.	Рубіжний контроль	13	28-29.11.2024	1. Попередній варіант повної КР. 2. Попередній варіант презентації.	
5.	Рубіжний контроль	15	12.12.2024	Доповідь на 12-ій Всеукраїнській науково-технічній конференції з міжнародною участю «СПІБТ–2024»	
6.	Перевірка на плагіат	15	12-13.12.2024	Електронний варіант кваліфікаційної роботи	
7.	КЕ на рівень « магістра»	16	17-18.12.2024	Здавання державного екзамену зі спеціальності на ступінь бакалавра	
8.	Кафедральний захист	16	19.12.2024	1. Оформлена КР (в форматі pdf) (передається студентом на кафедрі для внутрішньої рецензії). У разі отримання позитивної внутрішньої рецензії, КР подається на зовнішню рецензію. 2. Оформлена презентація (в форматі pdf).	
9.	Подання документів на захист	16	20.12.2024	1. Подання щодо захисту КР: тема, успішність, висновок керівника та висновок кафедри про КР. 2. Зовнішня рецензія (окремо від КР). Резюме на КР. 3. Електронний варіант та роздрукована презентація в кількості 5 примірників. 4. Електронний варіант КР на диску	
10.	Захист ДР	17	24,26,27, 28.12.2024	1. Приєднання студента до засідання кваліфікаційної комісії (конференції) у встановлений час для проведення процедури дистанційного захисту 2. Процедура захисту КР.	

ПРИМІТКА: Завдання додається до виконаної роботи та подається до атестаційної комісії.

Керівник
К.т.н., доцент _____ А.С. Сірівчук

Студент _____ О.Ю. Дацко

АНОТАЦІЯ

Дацко О. Ю. Розробка програмного засобу зменшення розмитості зображення: кваліфікаційна робота зі спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» (Освітньо-професійна програма "Системи технічного захисту інформації, автоматизація її обробки" – Миколаїв, НУК ім. адм. Макарова, 2024. – 60 с., 35 посилань.

Ключові слова: фільтр Віннера, зменшення розмитості зображення, алгоритм вододілу.

Актуальність роботи підтверджується необхідністю швидким розвитком технології комп’ютерного зору. Однією з задач комп’ютерного зору є зменшення розмитості зображень для задач ідентифікації об’єктів або загальних признаків неякісного зображення.

В рамках кваліфікаційної роботи були описані причини виникнення розмитостей зображення та методи покращення таких зображень. Практична частина роботи присвячена програмній реалізації алгоритмів зменшення розмитості зображень при розмитості в русі та розмитті Гаусса.

Ключові слова: фільтр Віннера, зменшення розмитості зображення, алгоритм вододілу.

ANNOTATION

Datsko O. Yu. Development of a software tool for reducing image blur: qualification work in specialty 141 "Electrical power engineering, electrical engineering and electromechanics" (Educational and professional program "Systems of technical information protection, automation of its processing" - Mykolaiv, NUK named after Adm. Makarov, 2024. - 60 p., 35 references.

Keywords: Winner filter, image blur reduction, watershed algorithm.

The relevance of the work is confirmed by the need for rapid development of computer vision technology. One of the tasks of computer vision is to reduce image blur for tasks of identifying objects or general features of a poor-quality image.

As part of the qualification work, the causes of image blur and methods for improving such images were described. The practical part of the work is devoted to the software implementation of algorithms for reducing image blur with motion blur and Gaussian blur.

Keywords: Wiener filter, image deblurring, watershed algorithm.

ЗМІСТ

ВСТУП

Відновлення спотворених зображень є однією з найцікавіших і важливіших проблем в завданнях обробки зображень – як з теоретичної, так і з практичної точок зору. Окремими випадками є розмиття внаслідок неправильного фокусу і зміщення – ці дефекти, з якими стикався кожен з користувачів, дуже складні у виправленні. З іншими спотвореннями (шум, неправильна експозиція, дисторсія) людство навчилося ефективно боротися, відповідні інструменти є в кожному фоторедакторі, який коректно реалізовано.

Вважається, що розмиття безповоротна операція і інформація безповоротно втрачається, оскільки кожен піксель перетворюється на пляму, усе змішується, а при великому радіусі розмиття так і зовсім отримаємо однорідний колір по усьому зображенню. Це не зовсім так – уся інформація просто перерозподіляється за деяким законом і може бути однозначно відновлена з деякими обмовками. Виключення складає лише краї зображення шириною в радіус розмиття – там повноцінне відновлення неможливе.

Метою кваліфікаційної роботи є реалізація програмного забезпечення для усунення розмитості цифрового зображення.

Для вирішення поставленої мети необхідно вирішити наступні задачі:

– описати причини появи розмитостей;

– провести аналіз методів усунення розмитостей;

– описати алгоритми усунення розмитостей;

– створити та протестувати програмну реалізацію.

1. ОБЛАСТЬ ЗАСТОСУВАННЯ СИСТЕМИ УСУНЕННЯ РОЗМИТОСТІ ЗОБРАЖЕННЯ

1.1 Обробка зображення, основні поняття і визначення

Будь-яка форма обробки інформації, для якої вхідні дані представлені зображенням, наприклад, фотографіями або відеокадрами називається обробкою зображень. Обробка зображень може здійснюватися як для отримання зображення на виході (наприклад, підготовка до поліграфічного тиражування, до телетрансляції тощо), так і для отримання іншої інформації (наприклад, розпізнавання тексту, підрахунок числа і типу клітин в полі мікроскопа тощо).

Світло, яке випромінюється або відбивається об'єктами та спроектовано на ділянку сітківки нашого ока, має складний спектральний розподіл. Сітківка ока складається з рецепторів трьох видів, чутливих до різних областей видимого світла. Тому набору з трьох чисел досить, щоб описати колір. Пласке зображення – це функції залежності кольору від координат, причому координати набувають дискретних значень. Відомі два основні підходи до формування і зберігання зображень: растрова графіка і векторна графіка, також існують їх комбінації.

Сьогодні редагування зображень проводиться в основному на комп'ютері растровими редакторами в цифровому виді. Сучасні редактори не позбавлені недоліків, проте грамотне їх використання дозволяє вирішити більшість завдань, що виникають при редагуванні зображень.

В якості джерел зображень у сучасному світі виступають: зображення з цифрового фотоапарата, негативні фотоплівки і слайди після оцифрування за допомогою сканера, фото банки, сервери файлообміну і пошукові системи. На цих ресурсах нерідко можна зустріти зображення без обмежень на використання.

Основні терміни, використовувані для виділення меж на зображенні, приведені в глосарії, який представлено в табл. 1.1.

Таблиця 1.1 – Глосарій предметної області

№ з / п	Термін	Визначення
1.	Обробка зображень	Будь-яка форма обробки інформації, для якої вхідні дані представлені зображенням, наприклад, фотографіями або відеокадрами
2.	Виділення меж	Термін в теорії обробки зображення і комп'ютерного зору, частково з області пошуку об'єктів і виділення об'єктів, грунтується на алгоритмах, які виділяють точки цифрового зображення, в яких різко змінюється яскравість або є інші види неоднорідностей
3.	Градієнт	Вектор, який своїм напрямом вказує напрям найшвидкого зростання деякої величини, значення якої міняється від однієї точки простору до іншої, а за величиною рівний швидкості росту цієї величини в цьому напрямі
4.	Точка	Простий графічний примітив, що має нульову розмірність. Точка характеризується тільки координатами свого місця розташування
5.	Згладжування	Технологія, що використовується в обробці зображень з метою зробити межі кривих ліній візуально гладшими, прибираючи « зубці », що виникають при растеризуванні на краях об'єктів
6.	Вектор	Впорядкована пара точок, одна з яких називається початком, друга – кінцем вектору

Продовження таблиці 1.1

7.	Піксель	Найменший логічний елемент двовимірного цифрового зображення в растровій графіці
8.	Растрове зображення	Сітка пікселів або кольорових точок на комп'ютерному моніторі, папері і інших пристроях, що відображають матеріал
9.	Графічний формат	Спосіб запису графічної інформації

Растрова графіка описує зображення з використанням кольорових точок, що називаються пікселями, розташованих на сітці. При редагуванні растрової графіки редагуються пікселі. Растрова графіка залежить від роздільної здатності, оскільки інформація, що описує зображення, прикріплена до сітки певного розміру. При редагуванні растрової графіки, якість її представлення може змінитися. Зокрема, зміна розмірів растрової графіки може привести до розмиття країв зображення, оскільки пікселі перерозподілятимуться на сітці. Виведення растрової графіки на пристрої з нижчою роздільною здатністю, ніж роздільна здатність самого зображення, знизить його якість.

Основою растрового представлення графіки є піксель (точка) з вказівкою її кольору. При описі, наприклад, червоного еліпса на білому фоні доводиться вказувати колір кожної точки як еліпса, так і фону. Зображення представляється у вигляді великої кількості точок – чим їх більше, тим візуально якісніше зображення і більше розмір файлу. Тобто одна точка і навіть картинка може бути представлена з кращою або гіршою якістю відповідно до кількості точок на одиницю довжини – так званою розподільною здатністю (зазвичай, точок на дюйм – *dpi* або пікселів на дюйм – *ppi*).

Крім того, якість характеризується ще і кількістю кольорів і відтінків, які може приймати кожна точка зображення. Чим великою кількістю відтінків характеризується зображення, тим більшу кількість розрядів потрібно для їх

опису. Червоний може бути кольором номер 001, а може і – 00000001. Таким чином, чим якісніше зображення, тим більше розмір файлу.

Колірна модель RGB — це адитивна колірна модель, у якій червоний , зелений і синій основні кольори світла додаються різними способами для відтворення широкого спектру кольорів . Назва моделі походить від ініціалів трьох додаткових основних кольорів : червоного, зеленого та синього.

Основне призначення колірної моделі *RGB* – сприйняття, представлення та відображення зображень в електронних системах, таких як телевізори та комп'ютери, хоча вона також використовується у звичайній фотографії та кольоровому освітленні . До епохи електроніки колірна модель *RGB* вже мала надійну теорію, засновану на людському сприйнятті кольорів .

RGB – це модель кольорів , що залежить від пристрою : різні пристрої виявляють або відтворюють задане значення *RGB* по-різному, оскільки елементи кольору (такі як люмінофори чи барвники) та їхня реакція на окремі рівні червоного, зеленого та синього відрізняються від виробника до виробника. або навіть в одному пристрої з часом. Таким чином, значення RGB не визначає той самий колір на різних пристроях без певного типу керування кольором .

Типовими пристроями введення *RGB* є кольорові телевізійні та відеокамери , сканери зображень і цифрові камери . Типовими пристроями виведення *RGB* є телевізори різних технологій (*CRT* , *LCD* , плазма , *OLED* , квантові точки тощо), дисплеї комп'ютерів і мобільних телефонів , відеопроектори , багатоколірні світлодіодні дисплеї та великі екрани, такі як Jumbotron . З іншого боку, кольорові принтери — це не RGB-пристрої, а субтрактивні кольорові пристрої, які зазвичай використовують колірну модель *CMYK* .

Щоб сформувати колір за допомогою *RGB*, необхідно накласти три світлові промені (один червоний, один зелений і один синій) (наприклад, шляхом випромінювання від чорного екрана або відбиття від білого екрана). Кожен з трьох променів називається компонентом цього кольору, і кожен з них

може мати довільну інтенсивність у суміші від повністю вимкненого до повністю включеного.

Колірна модель *RGB* є адитивною в тому сенсі, що якщо світлові промені різного кольору (частоти) накладаються в просторі, їхні спектри світла складаються, довжина хвилі за довжиною хвилі, щоб скласти кінцевий загальний спектр.Це, по суті, протилежне субтрактивній колірній моделі, зокрема колірній моделі *CMY* , яка застосовується до фарб, чорнила, барвників та інших речовин, колір яких залежить від відображення певних компонентів (частот) світла, під яким ми їх бачимо.

У адитивній моделі, якщо результуючий спектр, наприклад, накладання трьох кольорів, є плоским, білий колір сприймається людським оком при прямому попаданні на сітківку. Це різко контрастує з субтрактивною моделлю, де сприйманий результуючий спектр є тим, що випромінюють відбиваючі поверхні, такі як пофарбовані поверхні. Барвник відфільтровує всі кольори, крім свого; два змішані барвники відфільтровують усі кольори, крім спільного компонента кольору між ними, наприклад, зелений як загальний компонент між жовтим і блакитним, червоний як загальний компонент між пурпуровим і жовтим, і синьо-фіолетовий як загальний компонент між пурпуровим і блакитним. Серед пурпурового, блакитного та жовтого немає кольорових компонентів, тому спектр відтворюється з нульовою інтенсивністю: чорний .

Нульова інтенсивність для кожного компонента дає найтемніший колір (без світла, вважається чорним), а повна інтенсивність кожного дає білий ; якість цього білого залежить від природи основних джерел світла, але якщо вони належним чином збалансовані, результатом є нейтральний білий, який відповідає білій точці системи . Коли інтенсивність для всіх компонентів однакова, результатом є відтінок сірого , темніший або світліший залежно від інтенсивності. Коли інтенсивності відрізняються, результатом є кольоровий відтінок , більш-менш насичений залежно від різниці найсильнішої та найслабшої інтенсивності використовуваних основних кольорів.

Коли один із компонентів має найсильнішу інтенсивність, колір має відтінок, близький до основного кольору (червоний, зелений або блакитний), а коли два компоненти мають однакову найсильнішу інтенсивність, колір є відтінком вторинного кольору (відтінок блакитного , пурпурового або жовтого). Вторинний колір утворюється сумою двох основних кольорів однакової інтенсивності: блакитний – зелений+синій, пурпуровий – синій+червоний і жовтий – червоний+зелений. Кожен вторинний колір є доповненням до одного основного кольору: блакитний доповнює червоний, пурпурний доповнює зелений, а жовтий доповнює синій. Коли всі основні кольори змішуються в однаковій інтенсивності, в результаті виходить білий колір.

Колірна модель *RGB* сама по собі не визначає колориметрично, що мається на увазі під червоним , зеленим і синім , тому результати їх змішування вказуються не як абсолютні, а як відносні до основних кольорів. Коли визначено точну кольоровість основних кольорів червоного, зеленого та синього, модель кольорів стає абсолютним простором кольорів , наприклад *sRGB* або *Adobe RGB* .

Фізичні принципи вибору червоного, зеленого та синього кольорів.

Набір основних кольорів, наприклад *sRGB* , визначає трикутник кольорів; тільки кольори в цьому трикутнику можна відтворити шляхом змішування основних кольорів. Тому кольори поза кольоровим трикутником тут показані сірими. Показано основні параметри та білу точку *D*65 *sRGB*. Фоновим малюнком є діаграма кольоровості *CIE xy* .

Вибір основних кольорів пов'язаний з фізіологією людського ока ; Хороші первинні стимули - це стимули, які максимізують різницю між реакціями конусних клітин сітківки людини на світло різних довжин хвиль і, таким чином, утворюють великий кольоровий трикутник .

Звичайні три типи світлочутливих фоторецепторних клітин людського ока (колбочки) найбільше реагують на жовте (довга хвиля або L), зелене (середнє або *M*) і фіолетове (коротке або *S*) світло (пікові довжини хвилі

близько 570 нм). , 540 нм і 440 нм відповідно). Різниця в сигналах, отриманих від трьох типів, дозволяє мозку диференціювати широку гаму різних кольорів, залишаючись найбільш чутливим (загалом) до жовтувато-зеленого світла та до відмінностей між відтінками в області від зеленого до оранжевого.

Як приклад, припустімо, що світло в помаранчевому діапазоні довжин хвиль (приблизно від 577 нм до 597 нм) потрапляє в око та вражає сітківку. Світло цих довжин хвиль активує як середньо-, так і довгохвильові колбочки сітківки, але не однаково — довгохвильові клітини реагуватимуть сильніше. Різницю у реакції може виявити мозок, і ця різниця є основою нашого сприйняття апельсина. Таким чином, помаранчевий вигляд об’єкта є результатом того, що світло від об’єкта потрапляє в наше око та стимулює різні колбочки одночасно, але різною мірою.

Використання трьох основних кольорів недостатньо для відтворення всіх кольорів; лише кольори в межах колірного трикутника, визначеного кольоровістю основних кольорів, можуть бути відтворені шляхом адитивного змішування невід’ємних кількостей цих кольорів світла.

Колір у колірній моделі *RGB* описується за допомогою вказівки кількості кожного з червоного, зеленого та синього кольорів. Колір виражається триплетом *RGB* (r , g , b), кожен компонент якого може змінюватися від нуля до певного максимального значення. Якщо всі компоненти дорівнюють нулю, результат буде чорним; якщо всі максимальні, результатом буде найяскравіший білий колір, який можна представити.

Ці діапазони можна кількісно визначити кількома різними способами:

Від 0 до 1 з будь-яким дробовим значенням між ними. Це представлення використовується в теоретичному аналізі та в системах, які використовують представлення з плаваючою комою .

Значення кожного компонента кольору також можна записати у відсотках від 0% до 100%.

У комп’ютерах значення компонентів часто зберігаються як цілі числа без знаку в діапазоні від 0 до 255, діапазоні, який може запропонувати один 8-бітний байт . Вони часто представлені як десяткові чи шістнадцяткові числа.

Високоякісне цифрове обладнання для обробки зображень часто здатне працювати з більшими діапазонами цілих чисел для кожного основного кольору, такими як 0..1023 (10 біт), 0..65535 (16 біт) або навіть більше, розширюючи 24 біти (три 8-бітні значення) до 32-бітних , 48-бітних або 64-бітних одиниць (більш-менш незалежно від розміру слова конкретного комп’ютера).

У багатьох середовищах значення компонентів у діапазонах не керуються як лінійні (тобто числа нелінійно пов’язані з інтенсивністю, яку вони представляють), як, наприклад, у цифрових камерах і телевізійному мовленні та прийомі завдяки гамма-корекції. Лінійні та нелінійні перетворення часто розглядаються за допомогою цифрової обробки зображень. Представлення лише з 8 бітами на компонент вважаються достатніми, якщо використовується гамма-корекція.

Растрове представлення зазвичай використовують для зображень фотографічного типу з великою кількістю деталей або відтінків. На жаль, масштабування таких картинок у будь-яку сторону зазвичай погіршує якість. При зменшенні кількості точок втрачаються дрібні деталі і деформуються написи (правда, це може бути не так помітно при зменшенні візуальних розмірів самої картинки – тобто збереженні розподільної здатності). Додавання пікселів призводить до погіршення різкості і яскравості зображення, оскільки новим точкам доводиться давати відтінки, середні між двома і більше кольорами, що граничать. Поширені формати *.tif, .gif, .jpg, .png, .bmp*.

Виділення меж. Однією з основних і найважливіших цілей цифрової обробки зображень є розпізнавання присутніх на них об'єктів. Можливість розрізняти об'єкти закладена у високій інформативності зображення. Значною мірою саме це і залучає розробників різних інформаційних систем до використання зображень як способу представлення результатів спостереження.

Зображення, що в той же час пред'являються до обробки, містять багато надмірні мало інформативних відомостей, які займають, проте, великі об'єми пам'яті, і що вимагають виконання великої кількості обчислень при спробі використати їх для розпізнавання. У теорії розпізнавання образів існують методи, що дозволяють з'ясовувати міру інформативності тих або інших ознак. Проте нині ці методи не знайшли широкого застосування в розпізнаванні зображень.

Значно ширше застосовуються методи скорочення надмірності, що спираються на специфічні особливості зорового сприйняття зображень. Вважається, що суб'єктивне сприйняття спостережуваної сцени відбувається через її представлення у вигляді окремих однорідних областей (тобто сегментацію) і виділення контурних ліній. Контурні, або граничні, лінії розділяють на зображенні ділянки з різними властивостями, тому виділення контурів іноді розглядається як попередня обробка, спрямована на подальше виконання сегментації. В той же час, препарат, що утворюється в деяких випадках, може і самостійний ефективно використовуватися для розпізнавання, оскільки інформація, що міститься в нім, принаймні, з точки зору зорового сприйняття, цілком достатня для вирішення багатьох завдань такого типу.

Основною метою виявлення різких змін яскравості зображення є фіксація важливих подій і змін світу. Вони можуть відбивати різні припущення про модель формування зображення, зміни в яскравості зображення можуть вказувати на:

- зміни глибини;
- зміни орієнтації поверхонь;
- зміни у властивостях матеріалу;
- відмінність в освітленні сцени.

У ідеальному випадку, результатом виділення меж є набір пов'язаних кривих, що означають межі об'єктів, граней і відбитків на поверхні, а також криві які відображають зміни положення поверхонь. Таким чином,

застосування фільтру виділення меж до зображення може істотно зменшити кількість даних, які обробляються, через те, що відфільтрована частина зображення вважається менш значимою, а найбільш важливі структурні властивості зображення зберігаються. Проте не завжди можливо виділити межі в картинах реального світу середньої складності. Межі, виділені з таких зображень, часто мають такі недоліки як фрагментированость, відсутність меж або наявність неправдивих, таких, що не відповідають досліджуваному об'єкту, меж.

Межі виділені на двовимірному зображенні тривимірної сцени можуть бути підрозділені на залежні або не залежні від точки огляду. Незалежні від точки огляду межі зазвичай відбивають властивості, успадковані у об'єктів тривимірної сцени, такі як забарвлення поверхні і її форма. Залежні від точки зору межі можуть мінятися зі зміною точки огляду і відбивають геометрію сцени, як, наприклад, об'єкти, що перекриваються.

Звичайною межею може бути, наприклад, межа між блоками червоного і жовтого кольору. З іншого боку лінія може бути набором пікселів кольору, що відрізняється, на постійному фоні. Тому у лінії може бути по межі з кожного боку від неї.

Актуальність виділення контурів обумовлюється тим, що це початковий етап розпізнавання образів. У свою чергу, завдання розпізнавання образів вирішується в таких сферах, як системи технічного зору, застосовні для розпізнавання тексту, визначення дефектів вироблюваних виробів, пізнавання осіб, виявлення ДТП, управління автомобілем та ін.

1.2 Причини розмиття зображення

Розмитість даних зображення перешкоджає використанню методів отримання зображень у багатьох сферах застосування. Розмиття описує той факт, що інформація, що належить одній точці об’єкта, поширюється на певну область зображення замість того, щоб бути чітко локалізованою. Основними

джерелами розмиття є рух об’єктів під час запису зображення, рух камери під час запису, розфокусування, аберації та інші оптичні дефекти камери, а також атмосферні збурення.

Методи, які можуть усунути або зменшити розмитість і, таким чином, полегшити інтерпретацію розмитих зображень, називаються усуненням розмиття або деконволюцією . Однак деконволюція є дуже невірно поставленою проблемою. Посилення помилок у процедурах видалення розмиття може легко погіршити різкість зображення або навіть зробити його повністю непридатним для використання.

Величезну різноманітність методів деконволюції, розроблених протягом десятиліть досліджень у цій галузі, можна класифікувати за кількома критеріями:

Просторова структура розмиття: якщо всі частини зображення розмиті, по суті, однаково (тобто так звана функція розповсюдження точки, яка описує, як різка точка об’єкта зображена на спостережуваному зображенні, виглядає однаково в усіх місцях зображення)), один має просторово інваріантне розмиття. У цьому випадку математична модель для розмиття базується на простій згортці, що є вигідним для математичної та алгоритмічної обробки. В іншому випадку говорять про просторово-варіантне розмиття.

Знання про процес розмиття: у деяких контекстах процес розмивання відомий і може використовуватися як вхідна інформація для алгоритму розмивання. Такі алгоритми називають несліпими. Навпаки, алгоритми сліпої деконволюції спрямовані на оцінку функції розповсюдження точок разом із чітким зображенням із спостережуваного погіршеного зображення. Деякі методи сліпої деконволюції передбачають спочатку оцінку функції розподілу точок і застосування несліпого методу потім, тоді як інші підходи об’єднують обидва кроки в загальну процедуру. Так звані напівсліпі методи припускають часткове знання процесу розмиття, як правило, що функція розповсюдження точки належить до певного класу функцій з одним або декількома параметрами (як-от розмиття при русі чи дефокусуванні з невідомим параметром масштабу).

Рівень дискретизації: оскільки цифрові зображення базуються на дискретних вимірюваннях, отриманих із безперервної реальності (описаної законами оптики), існує два різних підходи до моделювання: по-перше, дискретні моделі, які описують дискретне розмите зображення як погіршення дискретного чіткого зображення, і тому може використовувати матричну алгебру та інші просторово-дискретні теорії. Тут процес розмиття має бути переформульований у дискретний спосіб. По-друге, безперервні моделі, які краще відображають безперервну природу самого процесу оптичного розмиття та його симетрії, і зазвичай включають (інтегро-) диференціальні рівняння та варіаційне числення. У цьому класі моделей дискретизація відкладена до числової оцінки.

В основному існує п’ять причин розмитих фотографій: рух камери, рух об’єкта зйомки, пропущений фокус, недостатня глибина різкості та м’якість об’єктива.

1.2.1 Рух камери

Якщо розмиття було спричинене рухом камери, ви часто побачите невеликі ореоли або подвійне зображення, коли дивитеся на зображення на 100%. Особливо це стосується зображень, встановлених на штативі, де розмиття спричинене вібрацією, а не безперервним рухом (рис. 1.1).

У цьому поданні видно ореоли (подвійне зображення), характерні для зображень, розмитих рухом камери. Це відбувається лише під час руху камери чи об’єкта.

У цьому випадку, оскільки вся фотографія виглядає таким чином, це не рух об’єкта, створений вітром, оскільки вітер рухав би листя та менші гілки більше, ніж стовбури. Отже, це має бути рух камери.

Якщо система стоїть на штативі то може бути важко визначити точну причину вібрації штатива після факту, тому важливо збільшити зображення на задній панелі камери, щоб перевірити різкість у полі. Тоді, якщо фотографія не різка, ви можете експериментувати, щоб знайти причину

Рисунок 1.1 – Приклад розмиття рухом камери

1.2.2 Рух обєкту

Під час руху об’єкта ви зазвичай бачите такий самий вид ореолів або подвійне зображення, що й під час руху камери, за винятком того, що ви зазвичай бачите це лише в окремих частинах зображення. Якщо, наприклад, квіти здув вітер, то одні квіти будуть гострішими за інші, а предмети, на які не впливає вітер, як-от каміння, будуть гострими (рис. 1.2).

Рисунок 1.2 – Приклад зображення з рухом об’єкту

Під час руху об’єкта, спричиненого вітром, деякі об’єкти будуть чіткішими за інші – навіть якщо вони можуть перебувати поблизу – і ви часто побачите ореоли (подвійне зображення).

У людей або тварин, що рухаються, кінцівки (руки, ноги, хвіст, кінчики крил) зазвичай більш розмиті, ніж тіло.

Методами захисту від розмиття руху об’єкта є або терпіння (щоб вщухнути вітер), або більш коротка витримка (зазвичай разом із вищим ISO).

1.3 Пропущений фокус

Незвично, коли фотографія повністю розфокусована. Для пейзажних фотографій ви зазвичай використовуєте середню або малу діафрагму, і більшість людей використовують автофокус, тому зазвичай щось у фокусі – хоча, можливо, не все або найважливіші речі.

Однак все ще можна повністю втратити фокус. Можливо, ви з якоїсь причини вимкнули автофокус, а потім забули його знов. Коли це станеться, ви не побачите ореол або подвійне зображення, викликане рухом камери; замість цього все зображення буде просто виглядати м’яким. Але одна підказка, яка вказує на втрату фокусу, полягає в тому, що зазвичай деякі частини зображення будуть трохи різкішими, ніж інші. Якщо об’єкти, розташовані ближче до камери, чіткіші, значить, ви сфокусувалися перед усіма. Якщо об’єкти, розташовані далі від камери, чіткіші, то ви сфокусувалися за всім (рис. 1.3).

Журавель (великий сірий птах) м'який. Але гуси (білі птахи) за ним гостріші. Насправді, чим далі птахи від камери, тим вони чіткіші, показуючи, що я зосередився позаду всього.

Можна сказати, що проблема була в недостатній глибині різкості, оскільки менша діафрагма зробила б кран гострішим. Але я використовував f/9,5, і всі птахи були далеко, тож я мав би сфокусувати їх усіх на f/9,5. Тож я просто пропустив фокус. (Я фокусувався вручну, тому що автофокус перестав працювати з моїм адаптером. Довга історія, але це те, що ви отримуєте, коли намагаєтеся сфокусуватися вручну на літаючих птахів!)

Рисунок 1.3 – Розмиття через пропущений фокус

Автофокус жодної камери не може ідеально відслідковувати рухомі об’єкти й постійно фокусуватися на них. Але якщо ви постійно пропускаєте фокус у таких ситуаціях із дзеркальною фотокамерою, спочатку спробуйте відкалібрувати автофокус камери (щось, що ви повинні мати можливість налаштувати самостійно в меню камери). Якщо це не допоможе, ви можете придбати камеру з кращою системою автофокусування.

1.2.3 Недостатня глибина різкості

Це типова проблема для пейзажних фотографій. Якщо частина зображення знаходиться у фокусі, але передній план, фон або обидва не у фокусі, то або вам потрібна була менша діафрагма, ви сфокусувалися на неправильній відстані, або неможливо отримати все у фокусі, незважаючи ні на що (рис. 1.4).

Очевидно, що фокус і глибина різкості пов’язані. Бувають випадки, коли, скажімо, діафрагма f/11 може бути достатньо малою, щоб отримати все у фокусі – якщо ви фокусуєтеся на правильній відстані. Але якщо ви сфокусуєтесь занадто близько до камери, фон буде розфокусований. І якщо ви сфокусуєтеся занадто далеко, передній план буде розфокусований.

Рисунок 1.4 – Зодраження з недостатньою глибиною різкості

Коли сцена має велику глибину, і отримати все у фокусі є складним завданням, вам потрібно сфокусуватися між переднім і заднім планом, але ближче до переднього плану, оскільки глибина різкості за точкою, на якій ви фокусуєтеся, більша, ніж перед ним.

Щоправда, фокусування «ближче до переднього плану» дещо розпливчасте. Фокусування для досягнення максимальної глибини різкості – складна тема, тому я рекомендую вам прочитати цю публікацію, якщо ви хочете зрозуміти деякі складності. Але якщо ви виявите, що багато ваших зображень мають розфокусовані передній або задній плани, тоді вам потрібно використовувати менші діафрагми, точніше фокусуватися або і те, і інше. І, можливо, це само собою зрозуміло, але використовуйте штатив! Штатив дозволить вам використовувати низьку витримку, необхідну для малих діафрагм, без потреби підвищувати ISO.

1.2.4 М'якість лінз

М’якість лінзи іноді робить все зображення нечітким, але зазвичай м’якими є лише кути. Оскільки середина чітка, ми можемо виключити рух камери або серйозну помилку фокусування. Рух об’єкта малоймовірний, якщо тільки вітер якимось чином не задував листя в кутах, а не в середині. Дерева посередині розташовані приблизно на тій самій відстані від камери, що й дерева в кутах, а діафрагма була достатньо малою (f/11), тому я сумніваюся, що проблема спричинена недостатньою глибиною різкості.

Іноді лінза може зробити все зображення м'яким. Якщо все трохи нечітко, і ви використовували f/16 або f/22, то причиною може бути дифракція. Наступного разу використовуйте ширшу діафрагму, якщо можливо (якщо ширша діафрагма забезпечить достатню глибину різкості). (Щоб дізнатися більше про діафрагму та дифракцію, перегляньте цю публікацію .)

З іншого боку, деякі об’єктиви можуть бути м’якими при широкій діафрагмі навіть у середині зображення. А деякі об’єктиви можуть бути м’якими при будь-якій діафрагмі. Особливо це стосується широкодіапазонного збільшення, наприклад 28-300 мм, або дешевих комплектних об’єктивів. Хоча деякі з новітніх широкодіапазонних зумів можуть бути відносно хорошими, вони, як правило, стають м’якими на великих фокусних відстанях, наприклад 200-300 мм. Телеконвертори також можуть зробити зображення м'яким.

Довгі об’єктиви також ускладнюють рух камери, вібрацію штатива та помилки фокусування. Отже, з телеоб’єктивами, як ви можете визначити, чи об’єктив (або комбінація об’єктив-телеконвертер) є м’яким, чи якась інша проблема спричинила нечіткість фотографії?

1.3 Огляд існуючого програмного забезпечення для ручного виділення меж на зображенні

Нині існує безліч графічних редакторів для обробки зображень від багатофункціональних редакторів до вузькоспеціалізованих. Розглянемо деякі сучасні редактори.

Adobe Photoshop – багатофункціональний графічний редактор, розроблений і поширюваний фірмою *Adobe Systems*. В основному працює з растровими зображеннями, проте має деякі векторні інструменти.

Підтримується обробка зображень, з глибиною кольору 8 біт, 16 біт, і 32 біт (використовуються числа одинарної точності з плаваючою комою). Можливе збереження у файлі додаткових елементів, як те: каналів, які спрямовують, шляхів обтравки шарів, що містять векторні і текстові об'єкти. Файл може включати колірні профілі (ICC), функції перетворення кольору (transfer functions). Допускаються неквадратні пікселі.

Переваги:

– *Photoshop* надає користувачеві велику кількість функцій для обробки зображень;

– можливість роботи з різними графічними форматами;

– підтримує роботу з шарами.

Недоліки:

– висока вартість;

– складність в освоєнні.

GIMP (*GNU Image Manipulation Program*) – растровий графічний редактор, програма для створення і обробки растрової графіки і частковою підтримкою роботи з векторною графікою.

Типові завдання, які можна вирішувати за допомогою *GIMP*, включають створення графіків і логотипів, масштабування і кадрування фотографій,

розфарбовування, комбінування зображень з використанням шарів, ретушування і перетворення зображень в різні формати.

Переваги:

– має достатній функціональний набір для обробки зображень;
– безкоштовний;
– відкритий початковий код;
– має можливість свертки із заданою матрицею.

Недоліки:

– не використовуються стандартні панелі управління *Windows*;
– інтерфейс всього редактора складається лише з вікон інструментів.

Paint.NET – безкоштовний растровий графічний редактор малюнків і фотографій для Windows, розроблений на платформі NET.Framework. Підтримується робота з шарами, є можливість створення різноманітних ефектів, відмінно працює відкат. Дозволяє використати різні завантажувані модулі. Програма має російський інтерфейс.

Переваги:

– програма безкоштовна для поширення і використання;
– має багато завантажуваних модулів;
– невелика вага.

Недоліки:

– потрібно час для освоєння інтерфейсу програмного продукту;
– виділення контура йде окремим завантажуваним модулем.

Висновки до розділу

Причинами розмиття зображення можуть бути, як неправильна зйомка кадру так і дефекти самого пристрою. Для відновлення зображення після розмиття можна використовувати спеціалізовані редактори цифроаих зображень але їх використання не надає змоги автоматизувати даний процес.

2. МАТЕМАТИЧНА МОДЕЛЬ ЗМЕНШЕННЯ РОЗМИТОСТІ ЗОБРАЖЕННЯ

2.1 Модель розмивання

Почнемо здалеку. Багато хто вважає, що розмиття - це незворотна операція і інформація в цьому випадку втрачається назавжди, тому що кожен піксель перетворюється на пляму, все змішується, і при великому радіусі розмиття ми отримаємо рівний колір по всьому зображенню. Але це не зовсім так - вся інформація просто перерозподіляється за певними правилами і може бути точно відновлена з певними припущеннями. Виняток становлять лише межі зображення, ширина яких дорівнює радіусу розмиття - повне відновлення тут неможливо.

Продемонструємо це на невеликому прикладі для одновимірного випадку - припустимо, що у нас є ряд пікселів з такими значеннями:

$$x_1 \mid x_2 \mid x_3 \mid x_4 \ldots$$ - Вихідне зображення

Після розмивання значення кожного пікселя додається до значення лівого: $x'_i = x_i + x_{i-1}$. Зазвичай його також потрібно розділити на 2, але ми опустимо це для простоти. У результаті ми маємо розмите зображення з такими значеннями пікселів:

$$x_1 + x_0 \mid x_2 + x_1 \mid x_3 + x_2 \mid x_4 + x_3 \ldots$$ - Розмите зображення

Тепер спробуємо його відновити, будемо послідовно віднімати значення за такою схемою - перший піксель з другого, результат другого пікселя з третього, результат третього пікселя з четвертого і так на, і ми отримаємо наступне:

$x_1 + x_0 \mid x_2 - x_0 \mid x_3 + x_0 \mid x_4 - x_0 \ldots$ - Відновлений образ

В результаті замість розмитого зображення ми отримали вихідне зображення з додаванням до кожного пікселя невідомої константи x_0 з альтернативним знаком. Це набагато краще - ми можемо вибрати цю константу візуально, ми можемо вважати, що вона приблизно дорівнює значенню x 1 , ми можемо автоматично вибрати її з таким критерієм, щоб значення сусідніх пікселів змінювалися якомога менше і т.д. Але все змінюється, як тільки ми додаємо шум (який завжди присутній на реальних зображеннях). У випадку описаної схеми на кожному етапі буде накопичуватися значення шуму в загальну величину, що в кінцевому підсумку може призвести до абсолютно неприйнятного результату, але, як ми вже знаємо, відновлення цілком можливо навіть таким примітивним методом.

Модель процесу розмивання

Тепер перейдемо до більш формального та наукового опису цих процесів розмиття та відновлення. Ми будемо розглядати лише зображення в градаціях сірого, припускаючи, що для обробки повнокольорового зображення достатньо повторити всі необхідні кроки для кожного з каналів кольорів RGB. Давайте введемо наступні визначення:

– $f(x, y)$ - вихідне зображення (нерозмите);

– $h(x, y)$ - функція розмиття;

– $n(x, y)$ - адитивний шум;

– $g(x, y)$ - зображення результату розмиття.

Ми сформуємо модель процесу розмивання у такий спосіб:

$$g(x, y) = h(x, y) * f(x, y) + n(x, y).$$

Завдання відновлення розмитого зображення полягає в знаходженні найкращого наближення $f'(x, y)$ до вихідного зображення. Розглянемо кожну складову докладніше. Що стосується функцій $f(x, y)$ і $g(x, y)$, то з ними все

цілком зрозуміло. Але щодо *h*(*x*, *y*) мені потрібно сказати пару слів - що це? У процесі розмиття кожен піксель вихідного зображення перетворюється на пляму у разі розфокусування та на відрізок лінії (або якийсь шлях) у разі звичайного розмиття через рух. Або ми можемо сказати інакше, що кожен піксель розмитого зображення «зібраний» з пікселів деякої сусідньої області вихідного зображення. Усі вони накладаються один на одного, що призводить до розмитого зображення. Принцип, за яким один піксель стає рознесеним, називається функцією розмиття. Інші синоніми - *PSF* (*Point spread function*), ядро та ін. Розмір цієї функції менше розміру самого зображення - наприклад, коли ми розглядали перший «демонстраційний» приклад, розмір функції був 2, тому що кожен піксель результату складався з двох пікселів.

Давайте подивимося, як виглядають типові функції розмивання. Далі ми будемо використовувати інструмент, який вже став стандартним для таких цілей - Matlab, він містить все необхідне для найрізноманітніших експериментів з обробкою зображень (крім іншого) і дозволяє зосередитися на алгоритмах, перекладаючи всю рутинну роботу на бібліотеки функцій. Однак це можливо лише за рахунок продуктивності.

PSF у випадку функцій Гауса (рис. 2.1) з використанням: *fspecial*('*gaussian*', 30, 8);

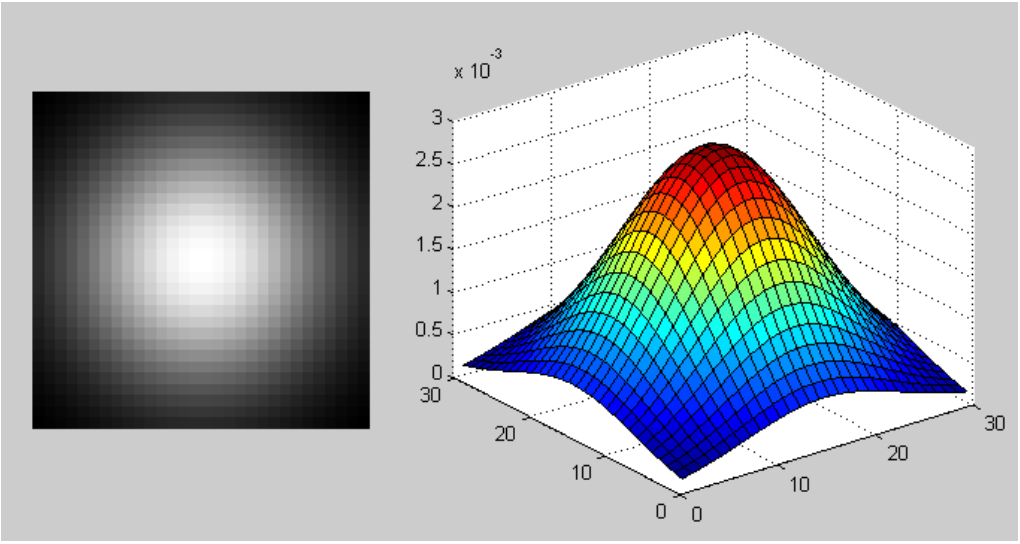


Рисунок 2.1 – Принцип роботи розмитості зображення

PSF у випадку функцій (рис. 2.2) розмиття за допомогою: *fspecial*('*motion*', 40, 45);

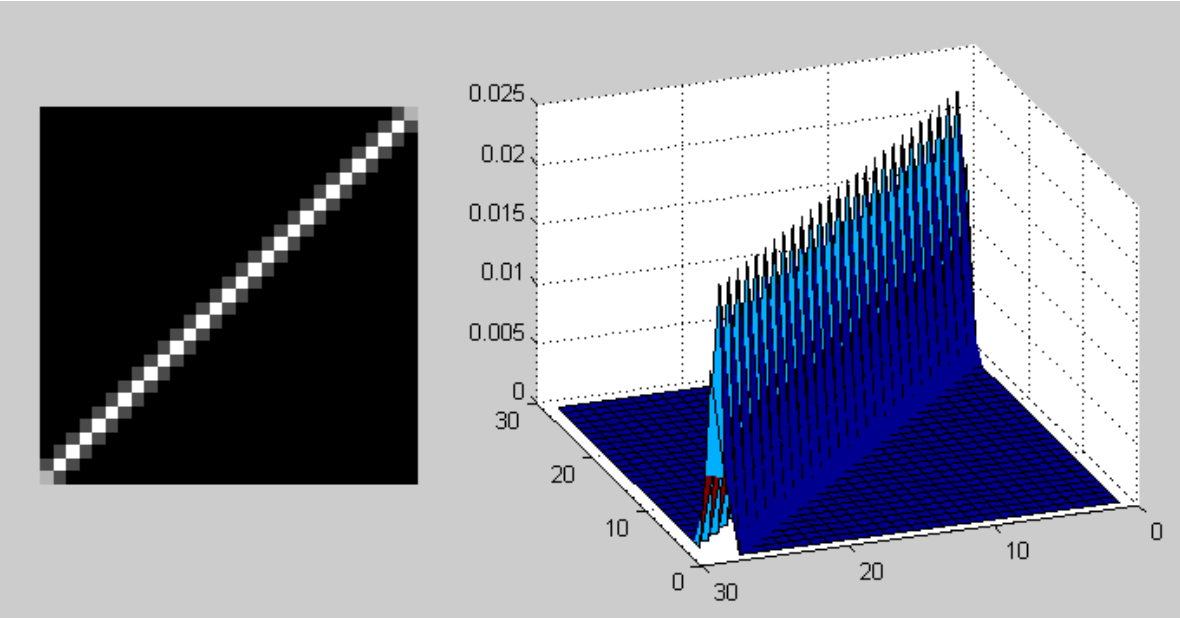


Рисунок 2.2 – Принцип розмиття рухом

Процес застосування функції розмиття до іншої функції (у його випадку до зображення) називається згорткою , тобто деяка область вихідного зображення згортається в один піксель розмитого зображення. Позначається через оператор «*», але не плутайте його з простим множенням! Математично для зображення f з розмірами M x N і функції розмиття h з розмірами mxn це можна записати таким чином:

$$g(x,y) = h(x,y) * f(x,y) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} h(i,j) f(x+i, y+j)$$

Де a=(m-1)/2, b=(n-1)/2 . Процес, протилежний згортці, називається деконволюцією , і вирішення такої задачі є досить рідкісним.

Залишилося лише розглянути останній доданок, що відповідає за шум, $n(x, y)$. Причин шуму в цифрових датчиках може бути багато, але

основними є теплові коливання (броунівський рух) і темновий струм. Гучність шуму також залежить від ряду факторів, таких як значення *ISO*, тип датчика, розмір пікселя, температура, величина магнітного поля і т. д. У більшості випадків присутній шум Гауса (який задається двома параметрами - середнім і дисперсією) , а також є адитивним, не корелює із зображенням і не залежить від координат пікселів. Останні три припущення дуже важливі для подальшої роботи.

2.2 Аналіз методів усунення розмиття

Проблема розмиття під час обробки зображень, яка знижує продуктивність і якість, є значною. Усунення розмиття усуває артефакти розмиття із зображень, зокрема розмиття від руху або розмиття через аберацію розфокусування. Розмиття часто описують як результат згортки (іноді змінної в просторі або часі) функції розповсюдження точки (*PSF*) із гіпотетичним чітким вхідним зображенням (яке потрібно отримати), де як різке вхідне зображення, так і функція поширення точок невідома.

Існує кілька методів видалення розмитості, які класифікуються як:

– алгоритм сліпої деконволюції. Якщо відомості про викривлення (розмиття та шум) невідомі, можна вважати ефективним алгоритм сліпої деконволюції. Метод одночасно відновлює зображення та функцію поширення точки. Кожна ітерація використовує прискорений демпфований алгоритм Річардсона-Люсі. Інші функції оптичної системи, наприклад камери, можна використовувати як вхідні параметри для покращення якості відновлення зображення. Обмеження PSF можуть передаватися функцією, визначеною користувачем;

– алгоритм Люсі-Річардсона. Коли відома функція розповсюдження точки PSF (оператор розмиття), але майже немає знань про шум, можна ефективно використовувати метод Люсі-Річардсона. Ітеративна, прискорена, демпфована техніка Люсі-Річардсона відновлює розмите та шумне зображення. Щоб

підвищити якість відновлення зображення, додаткову оптичну систему (наприклад, камеру) можна використовувати як вхідні параметри;

– регуляризований фільтр. Коли на відновлене зображення накладаються обмеження (такі як гладкість) і мало відомо про додатковий шум, можна ефективно використовувати регуляризовану деконволюцію. Регуляризований фільтр і обмежений алгоритм відновлення найменших квадратів використовуються для відновлення розмитого та шумного зображення;

– Фільтр Вінера. Деконволюція Вінера може бути корисною, коли функція розкиду точок і рівень шуму відомі або можуть бути передбачені.

– пари розмитих/зашумлених зображень. Цей метод використовує шумове зображення для зменшення розмитості вихідного зображення. Основна перевага цього методу полягає в тому, що він використовує як зашумлені, так і розмиті зображення, у результаті створюючи високоякісне реконструйоване зображення.

– функція щільності руху. За допомогою функції щільності руху в цьому підході досягається зменшення розмитості зображення. Покладення цього методу на оцінки недосконалих просторово інваріантних методів усунення розмиття для ініціалізації є одним із його обмежень.

– обробка викидів. Ця техніка аналізує різні викиди, включаючи насиченість пікселів і негаусівський шум, а потім пропонує метод деконволюції, який включає явний компонент для моделювання викидів. Внутрішні та зовнішні пікселі — це дві основні категорії, які використовуються для класифікації пікселів зображення.

– алгоритм *ADSD-AR*. Цей метод представляє схему *ASDS* (*Adaptive Sparse Domain Selection*), яка вивчає деякі компактні підсловники та адаптивно призначає підсловник як розріджений домен для кожного локального патча.

– гіперспектральний. Щоб декорелювати зображення *HS* і відокремити інформаційний вміст від шуму, цей підхід спочатку використовує *PCA*. Більшу частину загальної енергії або інформації зображення HS можна знайти в

перших k каналах *PCA*, тоді як решта *B-k* каналів *PCA* (де B — кількість спектральних смуг *HSI*, а *B k*) переважно містять шум.

– нейронні мережі. Тип багатопроцесорної комп'ютерної системи, відомий як нейронна мережа, має основні елементи обробки, високий рівень взаємозв'язку та адаптивну взаємодію між елементами. Завдяки своїй паралельній структурі нейронні мережі можуть функціонувати, навіть якщо один з їх елементів виходить з ладу.

– попереднє сліпе усунення розмиття. Це сліпа оцінка ядра та техніка видалення розмиття на основі попереднього градієнта *L*0. Підхід починається з оцінки ядра розмиття шляхом перемикання між чітким прогнозом зображення за допомогою використання l0 перед градієнтним зображенням і багатомасштабною оцінкою ядра. Після оцінки ядра проектується чітке зображення за допомогою стандартного несліпого процесу деконволюції з попередньо встановленим ядром.

2.2 Математичний опис алгоритму усунення розмитості

Так, існує так звана теорема згортки, згідно з якою операція згортки в просторовій області еквівалентна регулярному множенню в частотній області (де множення - поелементне, а не матричне). Відповідно, операція, протилежна згортці, еквівалентна діленню в частотній області, тобто це можна виразити наступним чином:

$$h(x,y) * f(x,y) \iff H(u,v)F(u,v)$$

де *H(u, v), F(u, v)* - функції Фур'є для *h(x,y)* і *f(x,y)*. Отже, процес розмивання можна записати в частотній області як:

$$G(u,v) = H(u,v)F(u,v) + N(u,v)$$

Тут у нас виникає спокуса розділити це рівняння на $H(u, v)$ і отримати наступну оцінку $F(u, v)$ вихідного зображення:

$$\hat{F}(u,v) = F(u,v) + \frac{N(u,v)}{H(u,v)}$$

Це називається зворотною фільтрацією, але на практиці вона майже ніколи не працює. чому так. Якщо функція $H(u, v)$ дає значення, близькі до нуля або дорівнює йому, то на вході цього доданка буде домінуючий. Це майже завжди можна побачити на реальних прикладах - щоб пояснити це, давайте згадаємо, як виглядає спектр після перетворення Фур'є.

Існуючі підходи до деконволюції

Існують підходи, які враховують наявність шуму на зображенні - одним з найпопулярніших і перших є фільтр Вінера. Він розглядає зображення і шум як випадкові процеси і знаходить таке значення f' для зображення f без спотворень , щоб середньоквадратичне відхилення цих значень було мінімальним. Мінімум такого відхилення досягається на функції в частотній області:

$$\hat{F}(u,v) = \left(\frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + S_\eta(u,v)/S_f(u,v)}\right) G(u,v)$$

Цей результат був знайдений Вінером у 1942 році. Ми не будемо давати його докладний висновок у цій статті, зацікавлені можуть знайти його тут . Функція S позначає тут енергетичний спектр шуму та зображення джерела відповідно - оскільки ці значення рідко відомі, то частка S n / S f замінюється деякою константою K, яку можна приблизно охарактеризувати як відношення сигнал/шум .

Наступний метод — «Обмежена фільтрація найменших квадратів», інші назви: «Тихоновська фільтрація», «Тихоновська регулярізація». Його ідея полягає у формуванні задачі у вигляді матриці з подальшим вирішенням відповідної задачі оптимізації. Результат рівняння можна записати таким чином:

$$\hat{F}(u,v) = \left(\frac{H^*(u,v)}{|H(u,v)|^2 + y|P(u,v)|^2}\right) G(u,v)$$

де y - параметр регуляризації, а $P(u, v)$ - Фур'є-функція оператора Лапласа (матриця 3 * 3).

Ще один цікавий підхід запропонували Річардсон (1972 рік) і Люсі (1974 рік) незалежно один від одного, тому цей підхід називають методом Люсі-Річардсона. Його відмінна риса полягає в тому, що він нелінійний, на відміну від перших трьох - потенційно це може дати кращий результат. Друга особливість - цей метод ітераційний, тому виникають труднощі з критерієм зупинки ітерацій. Основна ідея полягає у використанні методу максимальної правдоподібності, для якого передбачається, що зображення піддається розподілу Пуассона. Формули розрахунку досить прості, без використання перетворення Фур'є - все робиться в просторовій області:

$$\hat{f}_{k+1}(x,y) = \hat{f}_k(x,y)\left[h(-x,-y) * \frac{g(x,y)}{h(x,y) * \hat{f}_k(x,y)}\right]$$

Тут символ «*», як і раніше, позначає операцію згортки. Цей метод широко використовується в програмах для обробки астрономічних фотографій - використання в них деконволюції (а не нерізкої маски, як у фоторедакторах) є стандартом де-факто. Обчислювальна складність методу досить висока -

обробка середньої фотографії, в залежності від кількості ітерацій, може тривати багато годин і навіть днів.

Останній розглянутий метод, а точніше ціле сімейство методів, які активно не розвиваються, - це сліпа деконволюція. У всіх попередніх методах передбачалося, що функція розмиття *PSF* відома напевно, але на практиці це не так, зазвичай ми знаємо лише приблизну *PSF* за типом видимих спотворень. Сліпа деконволюція – це сама спроба врахувати це. Принцип досить простий, не заглиблюючись у подробиці – вибирається перше наближення *PSF*, потім виконується деконволюція одним із методів, після чого визначається ступінь якості за деяким критерієм, виходячи з цього ступеня *PSF*. функція налаштована, і ітерація повторюється до досягнення необхідного результату.

Для виділення меж на зображенні у відтінках сірого застосовується безліч алгоритмів. Одним з таких алгоритмів є оператор Собеля.

Оператор Собеля є неточним наближенням градієнта зображення, але підходить для практичного застосування у багатьох завданнях. Точніше, оператор використовує значення інтенсивності тільки в околиці 3×3 кожні пікселі для отримання наближення відповідного градієнта зображення, і використовує тільки цілочисельні значення вагових коефіцієнтів яскравості для оцінки градієнта.

Формула оператора Собеля (2.1) : *Gx* і *Gy* – дві матриці, де кожна точка містить наближені похідні по *x* і по *y*. Вони обчислюються таким чином шляхом множення матриці *Gx* і *Gy* та підсумовуванням обох матриць, в результаті отриманий результат записується в поточні координати *x* і *y* в нове зображення:

$$G = \sqrt{G_x^{\,2} + G^2{}_y}\,, \tag{2.1}$$

Матриці *Gy* і *Gx* (2.2):

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * A \tag{2.2}$$

Алгоритм оператора Прюитта подібний до алгоритму операторові Собеля, за винятком використання іншої матриці (2.3):

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} * A \tag{2.3}$$

Алгоритм оператора Шарра подібний до алгоритму операторові Собеля, за винятком використання іншої матриці (2.4) :

$$\begin{bmatrix} +3 & +10 & +3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix} \text{and} \begin{bmatrix} +3 & 0 & -3 \\ +10 & 0 & -10 \\ +3 & 0 & -3 \end{bmatrix} \tag{2.4}$$

Перехресний оператор Робертса – один з алгоритмів виділення меж, який обчислює суму квадратів різниць між діагонально суміжними пікселами (2.5). Це може бути виконано сверткой зображення з двома ядрами:

$$\begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix} \text{and} \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix} \tag{2.5}$$

Перетворення кожного піксела перехресним оператором Робертса може показати похідну зображення уздовж ненульової діагоналі, і комбінація цих перетворених зображень може також розглядатися як градієнт від двох верхніх

пікселів до двох нижнім. Оператор Робертса все ще використовується заради швидкості обчислень, але він програє порівняно з альтернативами з його значною проблемою чутливості до шуму. Він дає лінії тонше, ніж інші методи виділення меж.

В результаті використання операції дискретного двовимірного диференціювання виходить нове значення, яке записується в нове фото.

Оператор Кэнни використовує багатоступінчастий алгоритм для виявлення широкого спектру меж в зображеннях.

Алгоритм детектора меж не обмежується обчисленням градієнта згладженого зображення. У контурі межі оставляются тільки точки максимуму градієнта зображення, а не максимальні точки, що лежать поряд з межею, віддаляються. Тут також використовується інформація про напрям межі для того, щоб видаляти точки саме поряд з межею і не розривати саму межу поблизу локальних максимумів градієнта.

Потім за допомогою двох порогів віддаляються слабкі межі. Фрагмент межі при цьому обробляється як ціле. Якщо значення градієнта де-небудь на фрагменті, що простежується, перевищить верхній поріг, то цей фрагмент залишається також «допустимою» межею і в тих місцях, де значення градієнта падає нижче цього порогу, до тих пір, поки вона не стане нижча нижнього порогу. Якщо ж на усьому фрагменті немає жодної точки зі значенням великим верхнього порогу, то він віддаляється.

Такий гістерезис дозволяє зменшити число розривів у вихідних межах. Включення в алгоритм Кэнни шумозаглушування з одного боку підвищує стійкість результатів, а з іншої - збільшує обчислювальні витрати і призводить до спотворення і навіть втрати подробиць меж. У алгоритмі застосовується розмиття зображення для видалення шуму. Оператор Кэнни використовує фільтр (2.6) який може бути добре наближений до першої похідної гауссианы.

$$B_x = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 5 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * A \quad (2.6)$$

Межі відзначаються там, де градієнт зображення набуває максимального значення (2.7). Вони можуть мати різний напрям, тому алгоритм Кэнни використовує чотири фільтри для виявлення горизонтальних, вертикальних і діагональних ребер в розмитому зображенні.

$$G = \sqrt{G_x^2 + G_y^2}, \quad (2.7)$$

$$\Theta = \arctan\left(\frac{G_y}{G_X}\right) \quad (2.8)$$

Кут напряму вектору градієнта округляється і може набувати таких значень: 0, 45, 90, 135.

Перед застосуванням детектора, зазвичай перетворять зображення у відтінки сірого.

У вивченні обробки зображень вододілом є перетворення, визначене на зображенні в градаціях сірого . Назва метафорично стосується геологічного вододілу або дренажного вододілу, який розділяє суміжні дренажні басейни . Трансформація вододілу розглядає зображення, на якому воно працює, як топографічну карту , де яскравість кожної точки відображає її висоту, і знаходить лінії, які проходять уздовж вершин хребтів.

Існують різні технічні визначення вододілу. На графіках лінії вододілу можуть бути визначені на вузлах, на ребрах або гібридні лінії на вузлах і ребрах. Вододіли також можуть бути визначені в безперервній області. Існує також багато різних алгоритмів для обчислення вододілів. Алгоритми вододілу

використовуються в обробці зображень переважно для цілей сегментації об’єктів , тобто для поділу різних об’єктів на зображенні. Це дозволяє підрахувати об’єкти або провести подальший аналіз відокремлених об’єктів (рис. 2.3).

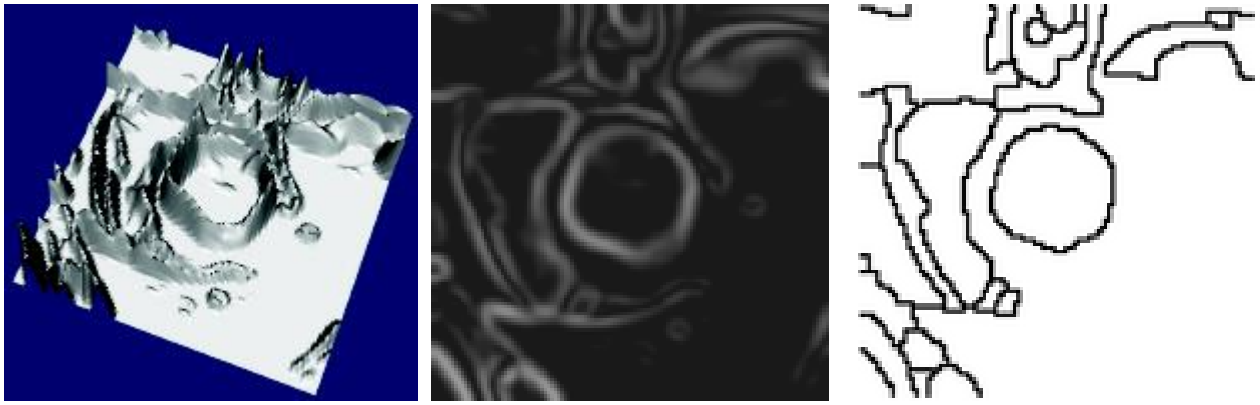


Рисунок 2.3 – Алгоритм вододілу

У геології вододіл - це вододіл, який розділяє суміжні водозбірні басейни.

Ідея була представлена в 1979 році Основна ідея полягала в розміщенні джерела води в кожному регіональному мінімумі рельєфу, щоб затопити весь рельєф із джерел і створити бар'єри при зустрічі різних джерел води. Отриманий набір бар’єрів утворює вододіл шляхом затоплення. Відтоді в цей алгоритм було внесено низку вдосконалень, які спільно називаються Priority-Flood.

Вододіл за топографічною відстанню

Інтуїтивно зрозуміло, що крапля води, що падає на топографічний рельєф, тече до «найближчого» мінімуму. «Найближчий» мінімум — це той мінімум, який знаходиться в кінці шляху найкрутішого спуску. З точки зору топографії, це відбувається, якщо точка лежить у водозбірному басейні цього мінімуму. Попереднє визначення не підтверджує цю умову.

Інтуїтивно зрозуміло, що вододіл – це відділення регіональних мінімумів, з яких крапля води може стікати до окремих мінімумів. Формалізація цієї

інтуїтивно зрозумілої ідеї була надана в для визначення вододілу реберно-зваженого графа.

Башер і Маєр представили алгоритмічну міжпіксельну реалізацію методу вододілу, враховуючи наступну процедуру:

– позначте кожен мінімум окремою міткою. Ініціалізуйте набір S позначеними вузлами.

– витягніть із S вузол x мінімальної висоти F , тобто $F(x) = \min\{F(y) | y \in S\}$. Присвойте мітку x кожному непоміченому вузлу y, що примикає до x , і вставте y S.

Попередні поняття зосереджені на водозбірних басейнах, але не на створеній розділовій лінії. Топологічний вододіл був введений М. Купрі та Г. Бертраном у 1997 р. і використав наступну фундаментальну властивість. Функція W є вододілом функції F тоді і тільки тоді, коли $W \leq F$ і W зберігає контраст між регіональними мінімумами F; де контраст між двома регіональними мінімумами M 1 і M 2 визначається як мінімальна висота, на яку необхідно піднятися, щоб перейти від $M1$ до $M2$.Ефективний алгоритм детально описано в статті.

Для використання принципу вододілу для сегментації зображення можуть бути використані різні підходи.

Локальні мінімуми градієнта зображення можуть бути обрані як маркери, у цьому випадку створюється надмірна сегментація, а другий крок включає об’єднання областей.

Перетворення вододілу на основі маркерів використовує конкретні позиції маркерів, які були або явно визначені користувачем, або визначені автоматично за допомогою морфологічних операторів чи іншими способами.

Алгоритм затоплення Мейєра. Один із найпоширеніших алгоритмів вододілу був представлений Ф. Мейєром на початку 1990-х років, хоча з тих пір у цей алгоритм було внесено ряд удосконалень, які спільно називаються Priority-Flood, включаючи варіанти, придатні для наборів даних, що складаються з трильйонів пікселів.

Алгоритм працює на зображенні в градаціях сірого. При послідовному затопленні сірого рельєфу утворюються вододіли з прилеглими водозбірними басейнами. Цей процес заливання виконується на градієнтному зображенні, тобто басейни повинні виникати по краях. Зазвичай це призведе до надмірної сегментації зображення, особливо для шумового матеріалу зображення, наприклад, даних медичної КТ. Або зображення має бути попередньо оброблено, або згодом регіони мають бути об’єднані на основі критерію подібності.

Вибирається набір маркерів, пікселів, з яких має початися заливання. Кожному надається окрема мітка. Сусідні пікселі кожної позначеної області вставляються в пріоритетну чергу з рівнем пріоритету, що відповідає величині градієнта пікселя.

Піксель із найнижчим рівнем пріоритету витягується з черги пріоритетів. Якщо всі сусіди виділеного пікселя, які вже були позначені, мають однакову мітку, тоді піксель буде позначено їх міткою. Усі непозначені сусіди, які ще не перебувають у черзі пріоритетів, поміщаються до черги пріоритетів.

Приклад перетворення вододілу, що підтримується маркером, для популяції фармацевтичних гранул. Лінії вододілу накладаються чорним кольором на стек зображень КТ.

Вододіли як оптимальні охоплюючі ліси були представлені Jean Cousty та ін. Вони встановлюють послідовність цих водозбірних басейнів: вони можуть бути еквівалентно визначені їхніми «водозбірними басейнами» (через властивість найкрутішого спуску) або «розділовими лініями», що розділяють ці водозбірні басейни (через принцип краплі води). Потім вони доводять через теорему про еквівалентність свою оптимальність з точки зору мінімальних охоплюючих лісів. Після цього вони вводять алгоритм лінійного часу для їх обчислення. Варто зазначити, що подібні властивості не перевіряються в інших фреймворках, і запропонований алгоритм є найефективнішим з існуючих алгоритмів як у теорії, так і на практиці.

Найважливішим методом усунення розмитості зображень через лінійний рух або несфокусовану оптику є фільтр Вінера. З точки зору обробки сигналу, розмиття внаслідок лінійного руху на фотографії є результатом поганої вибірки. Кожен піксель у цифровому представленні фотографії має відображати інтенсивність однієї нерухомої точки перед камерою. На жаль, якщо швидкість затвора надто повільна, а камера рухається, даний піксель буде амальграмою інтенсивності з точок уздовж лінії руху камери. Це двовимірна аналогія

$$G(u,v)=F(u,v).H(u,v)$$

де F — перетворення Фур'є «ідеальної» версії даного зображення, а H — функція розмиття. У цьому випадку H є функцією sinc: якщо три пікселі в рядку містять інформацію з однієї точки на зображенні, цифрове зображення буде здаватися згорнутим із триточковим вагоном у часовій області. В ідеалі можна було б провести зворотне проектування *Fest* або оцінки F, якщо G і H відомі. Ця техніка є зворотним фільтруванням.

Висновки до роздулі

Математичну модель розмиття можна представити у вигляді двох типів: Гаусове та розмиття в русі. Для усунення розмиття існує багато алгоритмів але виділення якогось особливого який буде підходити для всіх випадках немає. Найбільш поширеним є фільтр Віннера через його простоту та досить точні результати.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Огляд бібліотек косп’ютерного зору

Комп’ютерне бачення (CV) змінює те, як машини сприймають та інтерпретують візуальну інформацію. Комп’ютерний зір дозволяє машинам «бачити» і розуміти світ, подібно до системи зору людини. У центрі цієї трансформаційної сфери є спеціалізовані програмні засоби, відомі як бібліотеки комп’ютерного бачення. Бібліотеки комп’ютерного зору необхідні розробникам і дослідникам, яким потрібні візуальні дані в різних програмах.

Ці бібліотеки містять набір готових алгоритмів, функцій і інструментів, що спрощує складний процес аналізу зображень і відео. Їх значення полягає в їхній здатності вирішувати широкий спектр завдань, від розпізнавання обличчя до виявлення об’єктів, що робить комп’ютерний зір доступним для різних галузей.

Від систем спостереження, які підвищують безпеку, до автономних транспортних засобів, які пересуваються дорогами, комп’ютерний зір відіграє важливу роль у формуванні майбутнього технологій. Давайте дослідимо деякі популярні бібліотеки комп’ютерного зору, кожна з яких унікально сприяє розвитку візуального інтелекту. Ми вивчимо їхні функції, випадки використання та вплив, який вони мають на різні галузі, від охорони здоров’я до робототехніки.

3.1.1 *OpenCV*

OpenCV (Бібліотека комп’ютерного бачення з відкритим кодом) — це популярна бібліотека з відкритим кодом для задач комп’ютерного зору та обробки зображень. Він пропонує інструменти для виявлення обличчя, розпізнавання зображень і відстеження об’єктів.

Розробники використовують *OpenCV* для таких програм, як доповнена реальність, робототехніка та медична візуалізація. Він підтримує кілька мов

програмування, таких як *Python*, *C++* і *Java*, і працює на різних платформах, включаючи *Windows*, *macOS* і *Linux*.

Головні особливості:

– широкий набір функцій обробки зображень і відео;

– кросплатформна сумісність;

– інтеграція з фреймворками глибокого навчання;

– підтримує прискорення GPU для швидшої обробки.

Плюси:

– безкоштовний і з відкритим кодом;

– активна підтримка громади;

– велика документація;

– працює з кількома мовами програмування.

Мінуси:

– крута крива навчання для початківців;

– обмежена підтримка розширених завдань глибокого навчання;

Безкоштовне використання за ліцензією BSD.

3.1.2 *TensorFlow*

TensorFlow – це провідна платформа машинного навчання з відкритим кодом, розроблена *Google*. Він підтримує такі завдання, як розпізнавання зображень, обробка природної мови та навчання з підкріпленням.

TensorFlow широко використовується в проектах комп'ютерного бачення завдяки розширеним можливостям глибокого навчання та масштабованості. Він працює на різних платформах, включаючи мобільні пристрої та периферійні обчислення.

Головні особливості:

– попередньо навчені моделі для швидкого розгортання;

– *TensorFlow Lite* для мобільних і крайніх пристроїв;

– інструменти візуалізації з *TensorBoard*;

– підтримка процесорів і графічних процесорів.

Плюси:

– велика спільнота та екосистема;

– чудова документація та навчальні посібники;

– можливість масштабування для великих наборів даних.

Мінуси:

– висока крива навчання для початківці;.

– складний API порівняно з деякими альтернативами.

Безкоштовний із відкритим кодом за ліцензією Apache 2.0.

3.1.3. *BoofCV*

BoofCV — це легка та швидка бібліотека комп'ютерного зору для програм реального часу. Він написаний на Java та призначений для зручності використання.

BoofCV ідеально підходить для робототехніки, 3D-бачення та наукових досліджень. Він підтримує такі завдання, як виявлення функцій, калібрування камери та сегментація зображення.

Головні особливості:

– можливості обробки в реальному часі;

– вбудована підтримка геометричного зору;

– прості API для швидкої інтеграції.

Плюси:

– простий у використанні для розробників Java.

– легкий і ефективний.

– орієнтований на додатки в реальному часі.

Мінуси:

– обмежена підтримка спільноти порівняно з великими бібліотеками.

– менше додаткових функцій для глибокого навчання.

Безкоштовний і з відкритим кодом.

3.1.4 *SimpleCV*

SimpleCV – це платформа на основі *Python*, яка спрощує завдання комп'ютерного зору. Він зручний для початківців і надає інструменти для

обробки зображень, відстеження об’єктів і виявлення функцій. *SimpleCV* ідеально підходить для тих, хто починає вивчати комп’ютерний зір.

Головні особливості:

– прості для розуміння *API*;

– вбудовані функції для типових завдань обробки зображень;

– хороша інтеграція з бібліотеками *Python*.

Плюси:

– зручний для початківців;

– швидке налаштування та використання;

– чудово підходить для створення прототипів.

Мінуси:

– обмежені оновлення та підтримка.

– не підходить для розширених програм.

Безкоштовний і з відкритим кодом.

Відгуки G2:

3.1.5. *CAFFE*

CAFFE (*Convolutional Architecture for Fast Feature Embedding*) — це структура глибокого навчання, зосереджена на швидкості та модульності. *CAFFE*, розроблений *Berkeley Vision and Learning Center*, широко використовується для класифікації зображень і виявлення об’єктів.

Головні особливості:

– високошвидкісна обробка;

– модульна архітектура для гнучкості;

– попередньо навчені моделі для швидкого старту.

Плюси:

– дуже швидко для завдань, пов’язаних із зображеннями;

– чудово підходить для академічних і дослідницьких цілей;

– висока продуктивність на GPU.

Мінуси:

– обмежена підтримка НЛП і завдань без зображень;

– повільніший розвиток порівняно з новими фреймворками.

Безкоштовний і з відкритим вихідним кодом за ліцензією *BSD*.

3.1.6 *Detectron* 2

Detectron 2 — це передова бібліотека *Facebook AI* для виявлення та сегментації об’єктів. Він заснований на *PyTorch* і підтримує найсучасніші моделі. Розробники використовують його для таких завдань, як сегментація екземплярів, визначення ключових точок і панорамна сегментація.

Головні особливості:

– розширене виявлення та сегментація об’єктів;

– вбудована підтримка користувацьких наборів даних;

– прискорення GPU для швидшого навчання.

Плюси:

– підтримує новітні дослідницькі моделі;

– висока точність у задачах виявлення;

– гнучкий і настроюваний.

Мінуси:

– потрібні знання *PyTorch*;

– ресурсомісткість для навчання.

Безкоштовний із відкритим кодом за ліцензією *Apache* 2.0.

3.1.7 *OpenVINO*

OpenVINO (*Open Visual Inference and Neural Network Optimization*) — це набір інструментів *Intel* для розгортання моделей ШІ. Він оптимізує моделі глибокого навчання для периферійних пристроїв, пропонуючи високу продуктивність для таких завдань, як розпізнавання зображень і виявлення об’єктів.

Головні особливості:

– оптимізація під обладнання Intel;

– попередньо навчені моделі для різних застосувань;

– підтримує гетерогенні обчислення.

Плюси:

– підвищує швидкість висновку на пристроях *Intel*;

– легка інтеграція з периферійними програмами;

– комплексний набір інструментів для розгортання.

Мінуси:

– найкраща продуктивність обмежена обладнанням *Intel*;

– менш підходить для тренувань моделей.

Безкоштовний із відкритим кодом за ліцензією *Apache* 2.0.

3.2 Розробка модулю побудови фільтру розфукусування

Метою відновлення (усунення розмитості) є отримання оцінки вихідного зображення. Формула відновлення в частотній області:

$$U' = H_w \cdot S$$

де U' – спектр оцінки оригінального зображення U, H_w є фільтром відновлення, наприклад фільтр Вінера.

Фільтр Вінера — це спосіб відновлення розмитого зображення. Припустімо, що *PSF* є реальним і симетричним сигналом, спектр потужності оригінального справжнього зображення та шум невідомі, тоді спрощена формула Вінера:

$$H_w = \frac{H}{|H|^2 + \frac{1}{SNR}}$$

де *SNR* це відношення сигнал/шум.

Алгоритм відновлення розфокусованого зображення складається з генерації *PSF*, генерації фільтра Вінера та фільтрації розмитого зображення в частотній області:

```
// it needs to process even image only
   Rect roi = Rect(0, 0, imgIn.cols & -2, imgIn.rows & -2);
    //Hw calculation (start)
   Mat Hw, h;
   calcPSF(h, roi.size(), R);
   calcWnrFilter(h, Hw, 1.0 / double(snr));
   //Hw calculation (stop)
    // filtering (start)
   filter2DFreq(imgIn(roi), imgOut, Hw);
   // filtering (stop)
```

Функція *calcPSF*() формує круговий *PSF* відповідно до радіуса вхідного параметра:

```
void calcPSF(Mat& outputImg, Size filterSize, int R)
{
   Mat h(filterSize, CV_32F, Scalar(0));
   Point point(filterSize.width / 2, filterSize.height / 2);
   circle(h, point, R, 255, -1, 8);
   Scalar summa = sum(h);
   outputImg = h / summa[0];
}
```

Функція calcWnrFilter() синтезує спрощений фільтр Вінера за формулою, описаною вище:

```
void calcWnrFilter(const Mat& input_h_PSF, Mat& output_G, double nsr)
{
   Mat h_PSF_shifted;
   fftshift(input_h_PSF, h_PSF_shifted);
```

```
    Mat planes[2] = { Mat_<float>(h_PSF_shifted.clone()), Mat::zeros(h_PSF_shifted.size(), CV_32F) };
    Mat complexI;
    merge(planes, 2, complexI);
    dft(complexI, complexI);
    split(complexI, planes);
    Mat denom;
    pow(abs(planes[0]), 2, denom);
    denom += nsr;
    divide(planes[0], denom, output_G);
}
```

Функція *fftshift*() змінює *PSF*. Цей код щойно скопійовано з посібника Дискретне перетворення Фур’є :

```
void fftshift(const Mat& inputImg, Mat& outputImg)
{
    outputImg = inputImg.clone();
    int cx = outputImg.cols / 2;
    int cy = outputImg.rows / 2;
    Mat q0(outputImg, Rect(0, 0, cx, cy));
    Mat q1(outputImg, Rect(cx, 0, cx, cy));
    Mat q2(outputImg, Rect(0, cy, cx, cy));
    Mat q3(outputImg, Rect(cx, cy, cx, cy));
    Mat tmp;
    q0.copyTo(tmp);
    q3.copyTo(q0);
    tmp.copyTo(q3);
    q1.copyTo(tmp);
    q2.copyTo(q1);
```

```
    tmp.copyTo(q2);
}
```

Функція *filter2DFreq*() фільтрує розмите зображення в частотній області:

```
void filter2DFreq(const Mat& inputImg, Mat& outputImg, const Mat& H)
{
    Mat planes[2] = { Mat_<float>(inputImg.clone()), Mat::zeros(inputImg.size(), CV_32F) };
    Mat complexI;
    merge(planes, 2, complexI);
    dft(complexI, complexI, DFT_SCALE);

    Mat planesH[2] = { Mat_<float>(H.clone()), Mat::zeros(H.size(), CV_32F) };
    Mat complexH;
    merge(planesH, 2, complexH);
    Mat complexIH;
    mulSpectrums(complexI, complexH, complexIH, 0);

    idft(complexIH, complexIH);
    split(complexIH, planes);
    outputImg = planes[0];
}
```

Результати обробки зображені на рисунку 3.1. . Параметри відношення сигнал/шум та довжина зсуву підібрані експериментально. В даному випадку ці параметри становили *H*=25 а *SNR*=250.

а)

б)

Рисунок 3.1 – Результати обробки зображення

а) вхідне зображення; б) вихідне зображення

3.3 Розробка модулю зменшення розмиття в русі

Функція розповсюдження точки (PSF) лінійного спотворення розмиття рухом є сегментом лінії. Такий PSF визначається двома параметрами: є довжиною розмиття і є кутом руху. Щоб синтезувати фільтр Вінера для випадку розмиття руху, йому потрібно вказати співвідношення сигнал/шум SNR, LEN, THETA.

Алгоритм відновлення розмитості зображення складається з генерації PSF, генерації фільтра Вінера та фільтрації розмитого зображення в частотній області:

```
// it needs to process even image only
   Rect roi = Rect(0, 0, imgIn.cols & -2, imgIn.rows & -2);

   //Hw calculation (start)
   Mat Hw, h;
   calcPSF(h, roi.size(), LEN, THETA);
   calcWnrFilter(h, Hw, 1.0 / double(snr));
   //Hw calculation (stop)

   imgIn.convertTo(imgIn, CV_32F);
   edgetaper(imgIn, imgIn);

   // filtering (start)
   filter2DFreq(imgIn(roi), imgOut, Hw);
   // filtering (stop)
```

Функція calcPSF() формує PSF відповідно до вхідних параметрів (в градусах):

```
void calcPSF(Mat& outputImg, Size filterSize, int len, double theta)
{
    Mat h(filterSize, CV_32F, Scalar(0));
    Point point(filterSize.width / 2, filterSize.height / 2);
    ellipse(h, point, Size(0, cvRound(float(len) / 2.0)), 90.0 - theta, 0, 360, Scalar(255), FILLED);
    Scalar summa = sum(h);
    outputImg = h / summa[0];
}
```

Функція edgetaper() звужує краї вхідного зображення, щоб зменшити ефект дзвінка у відновленому зображенні:

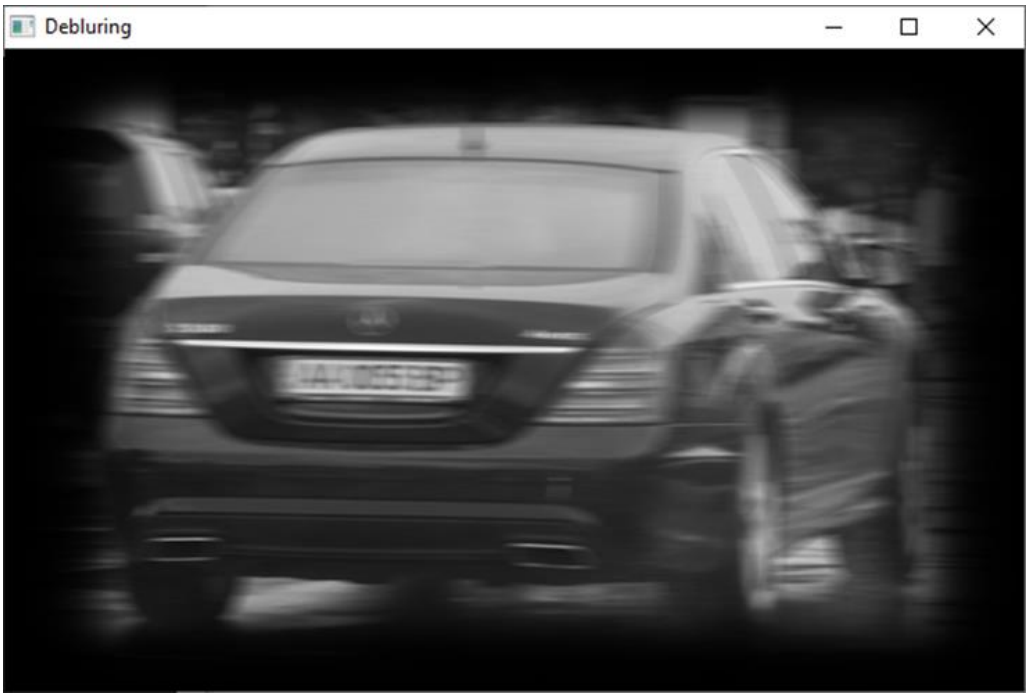
```
void edgetaper(const Mat& inputImg, Mat& outputImg, double gamma, double beta)
{
    int Nx = inputImg.cols;
    int Ny = inputImg.rows;
    Mat w1(1, Nx, CV_32F, Scalar(0));
    Mat w2(Ny, 1, CV_32F, Scalar(0));

    float* p1 = w1.ptr<float>(0);
    float* p2 = w2.ptr<float>(0);

    float dx = float(2.0 * CV_PI / Nx);
    float x = float(-CV_PI);

    for (int i = 0; i < Nx; i++)
    {
```

```
            p1[i] = float(0.5 * (tanh((x + gamma / 2) / beta) - tanh((x - gamma / 2) / beta)));
            x += dx;
        }
        float dy = float(2.0 * CV_PI / Ny);
        float y = float(-CV_PI);
        for (int i = 0; i < Ny; i++)
        {
            p2[i] = float(0.5 * (tanh((y + gamma / 2) / beta) - tanh((y - gamma / 2) / beta)));
            y += dy;
        }
        Mat w = w2 * w1;
        multiply(inputImg, w, outputImg);
    }
```

Функції *calcWnrFilter*(), *fftshift*() і *filter2DFreq*() реалізують фільтрацію зображення за вказаною PSF у частотній області. Функції скопійовано з пункту 3.2.

На слайді представлено відновлення зображення з розмитості в русі. Параметри відношення сигнал/шум та довжина зсуву підібрані експериментально. В даному випадку ці параметри становили *LEN*=15 а *SNR*=150, *THETA*=0

а)



б)

Рисунок 3.2 – Результати обробки зображення

а) вхідне зображення; б) оброблено зображення

Висновки до розділу

Представлені реалізації алгоритмів зменшення розмитості за допомогою бібліотеки OpenCV не дозволяють повністю відновити зображення але допомагають виділити ключові моменти, такі як джерело фотографії та номер автомобіля.

ВИСНОВОК

В ході кваліфікаційної роботи представлено види розмиття та причини їх виниктення. Розмиття зображенн вважається незворотньою дією але існують алгоритми для покращення зображень.

В рамкаж роботи було описано математичну модель появи розмитості та методів їх усунення. В практичній частині роботи було розроблено два модулі усунення розмитості на базі використання бібліотеки комп’ютерного зору OpenCV. Результати роботи показують, що алгоритми не можуть повністю відновити вхідне зображення але дозволяють отримати корисну інформацію з нього.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Quan, Qian. A Motion Blurred Image Restoration Method Based on Wiener Filtering. Advanced Materials Research. 2011/ 403-408. 1664-1667. 10.4028/www.scientific.net/AMR.403-408.1664.*

2. *M. M. Bronstein, A. M. Bronstein, M. Zibulevsky, and Y. Y. Zeevi, "Blind deconvolution of images using optimal sparse representations," IEEE Trans. Image Process., vol. 14, no. 6, pp. 726–736, Jun. 2005.*

3. *Dejee Singh, Mr R. K. Sahu "A survey of various image deblurring techniques", IJARCCE vol. 2, issue 12 december 2013.*

4. *D. G. Tzikas, A. C. Likas, and N. P. Galasanos, "Variational Bayesian sparse kernel-based blind imagedeconvolution with student's- priors," IEEE Trans. Image Process., vol. 18, no. 4, pp. 753–764, Apr. 2009.*

5. *Jian-Feng Cai, Hui Ji, Chaoqiang Liu and Zuowei Shen, "Framelet Based Blind Motion Deblurring From a Single Image", IEEE Transaction on the image processing Vol. 21.No.2. February 2012.*

6. *Mr. A. S. Mane, Prof. Mrs. M. M. Pawar "Removing blur from degraded image with blind deconvolution using canny edge detecting technique" vol. 1,issue 11 november 2014.*

7. *K. Sato, S. Ishizuka, A. Nikami, M. Saot, "Control techniques for optical image stabilizing system", IEEE Trans. Consum. Electron. 39 (3) (1993) 461–466.*

8. *M. Ben-Ezra and S. K. Nayar, "Motion-based motion deblurring,"IEEE Trans. PAMI, vol. 26, no. 6, pp. 689– 698, Jun. 2004.*

9. *Ashwini M. Deshpande and Suprava Patnaik "Uniform and Non-uniform single image deblurring based on spares representation and adaptive dictionary learning", IJMA, vol. 6, feb. 2014.*

10. *Rupali Patil, Sangeeta Kulkarni, "Blur removel from images using canny and blind deconvolution techniques", IJCTEE, 2011.*

11. R. Dash, P. K. Sa, and B. Majhi, "RBFN based motion blur parameter estimation," in Proc. IEEE International Conference on Advanced Computer Control, Singapore, Jan 2009, pp. 327-331.

12. Shamik Tiwari, V. P. Shukla, A. K. Singh, S. R. Biradar, "Review of motion blur estimation techniques" Journal of Image and Graphics Vol. 1, No. 4, December 2013.

13. R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, andW. T. Freeman "Removing camera shake from a single photograph" 25(3):787–794, 2006.

14. R. Vio, J. Nagy, and W. Wamsteker. "Blind motion deblurring using multiple images". Jrnl Comput. Phy., 228(14):5057–5071, 2009.

15. Aarpna Ashok, deepa, "handling noise and outliers in single image Deblurring using L0 Sparsity", vol 4, issue 7, july 2015.

16. Wenzhi Liao, Bart Goossens, Jan Aelterman, Hiep Quang Luong, Aleksandra Pizurica, Niels Wouters, Wouter Saeys, Wilfried Philips "Hyperspectral image deblurring with pca and total variation".

17. Swati Sharma, Shipra Sharma and Rajesh Mehra, " Image restoration using modified LR algorithm in the presence of Gaussian and motion blur" AEEE, vol 3, 2013.

18. L. Ilic, A. Pizurica, E. Vansteenkiste and W. Philips,"Image blur estimation based on the average cone of ratio in the wavelet domain," Proc. SPIE, Wavelet Applications in Industrial Processing VI, 72480F, pp. 1– 10, 2009.

19. Prodip Biswas, Abu Sufian Sarkar, Mohammed Mynuddin, "Deblurring images using weiner filter", IJCA vol.109, jan 2015.

20. H. Zhang, L. Zhang, H. Shen, "A super-resolution reconstruction algorithm for hyperspectral images," Signal Processing, vol. 92, no. 9, pp. 2082–2096, 2012.

21. *Neetin kumar, Dr. Manish shrivastva," Image deblurring using a neural network approaches", IJEIT, vol 2, September 2012*

22. M. Ben-Ezra, S. K. Nayar, "Motion-based motion deblurring," Pattern Analysis and Machine Intelligence, IEEE Transactions, vol. 26, no. 6,pp. 689 – 698.

23. J.-F. Caia, H. Jib, C. Liua, Z. Shenb, "Blind motion deblurring using multiple images," Journal of Computational Physics, vol. 228, no. 14, Aug., 2009, pp. 5057-5071.

24. L. Yuan, J. Sun, L. Quan, H.-Y. Shum, "Image deblurring with blurred/noisy image pairs," ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2007 TOG Homepage, vol. 26, no. 3, July 2007, article no. 1.

25. S. Dai, Y. Wu, "Motion from blur," Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference, 23-28 June 2008,pp. 1-8.

26. R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, W. T. Freeman, "Removing camera shake from a single photograph," ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2006 TOG Homepage, vol. 25, no. 3, July 2006, pp. 787 – 794.

27. L. Xu, J. Jia, "Two-phase kernel estimation for robust motion deblurring," ECCV'10 Proceedings of the 11th European conference on Computer vision: Part I, pp. 157-170.

28. Q. Shan, J. Jia, A. Agarwala, "High-quality motion deblurring from a single image," ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2008 TOG Homepage, vol. 27, no. 3, August 2008,

29. article no. 73.

30. L. Yuan, J. Sun, L. Quan, H.-Y. Shum, "Progressive inter-acale and intra-scale non-blind image deconvolution," ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2008 TOG Homepage, vol. 27, no. 3, Aug. 2008, article no. 74.

31. L. Zouy, H. Zhouz, S. Chengx, C. Heyy, "Dual range deringing for Nonblind image deconvolution," Image Processing (ICIP), 2010 17th IEEE International Conference, 26-29 Sept. 2010, pp. 1701 – 1704.

32. S. Cho, J. Wang, S. Lee, "Handling outliers in non-blind image deconvolution," Computer Vision (ICCV), 2011 IEEE International Conference, 6-13 Nov. 2011, pp. 495 – 502.

33. J.-H. Lee, Y.-S. Ho, "Non-blind image deconvolution with adaptive regularization," PCM'10 Proceedings of the 11th Pacific Rim conference on Advances in multimedia information processing: Part I, pp. 719-730.

34. A. Jalobeanu, L. Blanc-Féraud, J. Zerubia, "Satellite image deconvolution using complex wavelet packets," Image Processing, 2000. Proceedings. 2000 International Conference, 2000, vol. 3, pp. 809-812.

35. P. Marcelo, J. Daniel, "An iterative SNR estimation algorithm for wiener deconvolution of self-similar images distorted by camera shake blurring," 8th WSEAS International Conference on Signal, Speech and Image Processing (SSIP '08) Santander, Cantabria, Spain, September 23-25, 2008, pp.97-100.