МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

ВИКОРИСТАННЯ OPENCV ДЛЯ РЕЄСТРАЦІЇ РУХУ НА ВІДЕОСТРІМІ

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторної роботи № 3
з дисципліни «Віртуальна реальність»
для студентів бакалаврського рівня вищої освіти спеціальності 121
"Інженерія програмного забезпечення"

Львів -- 2025

Використання OpenCV для реєстрації руху на відеострімах: методичні вказівки до виконання лабораторної роботи №3 з дисципліни "Віртуальна реальність" для студентів першого (бакалаврського) рівня вищої освіти спеціальності 121 "Інженерія програмного забезпечення" . Укл.: О.Є. Бауск. -- Львів: Видавництво Національного університету "Львівська політехніка", 2025. -- 10 с.

Укладач: Бауск О.Є., к.т.н., асистент кафедри ПЗ

Відповідальний за випуск: Федасюк Д.В., доктор техн. наук, професор

Рецензенти: Федасюк Д.В., доктор техн. наук, професор

Задорожний І.М., асистент кафедри ПЗ

Тема роботи: Використання OpenCV для реєстрації руху на відеострімах.

Мета роботи: Ознайомитись з основами функціонування системи OpenCV, навчитися використовувати її для реєстрації руху на відеострімах.

Теоретичні відомості

Теоретичні відомості

Що таке OpenCV?

OpenCV (Open Source Computer Vision Library) — це бібліотека з відкритим вихідним кодом, яка містить понад 2500 алгоритмів для комп'ютерного зору та машинного навчання. Вона широко використовується для обробки зображень та відео, розпізнавання об'єктів, відстеження руху, аналізу сцен та інших задач, пов'язаних з комп'ютерним зором.

Відстеження особливих точок (Feature Tracking)

Відстеження особливих точок — це процес визначення та слідкування за рухом певних ключових точок (features) на послідовних кадрах відео. Це дозволяє аналізувати рух об'єктів, стабілізувати відео, розпізнавати жести та багато іншого.

Одним з популярних алгоритмів для відстеження особливих точок є алгоритм Лукаса-Канаде (Lucas-Kanade), який реалізований в OpenCV у функції `cv2.calcOpticalFlowPyrLK()`.

Алгоритм Лукаса-Канаде

Алгоритм Лукаса-Канаде базується на припущенні, що рух точок між двома послідовними кадрами є малим і приблизно однаковим у локальному околі точки. Він використовує піраміду зображень для ефективного відстеження руху точок на різних масштабах.

Основні етапи алгоритму Лукаса-Канаде:

1. Вибір ключових точок на першому кадрі (наприклад, за допомогою алгоритму Ші-Томасі або Харріса).
2. Відстеження цих точок на наступних кадрах за допомогою оптичного потоку.
3. Оновлення позицій точок та повторення процесу для наступних кадрів.

Щільний оптичний потік

Щільний оптичний потік — це метод, який обчислює оптичний потік для кожного пікселя зображення. Це дозволяє отримати повну картину руху на всьому зображенні.

Розріджений оптичний потік

Розріджений оптичний потік — це метод, який обчислює оптичний потік для деяких ключових точок на зображенні, а не для всіх пікселів. Це дозволяє зменшити обчислювальну складність і збільшити швидкість роботи алгоритму.

Особливі точки і алгоритм Ші-Томасі

Алгоритм Ші-Томасі (Shi-Tomasi) — це вдосконалена версія детектора кутів Харріса, що використовується для знаходження особливих точок на зображенні. Цей алгоритм оцінює "якість" кутів на основі мінімальних власних значень матриці градієнтів, що робить його більш надійним для відстеження точок між кадрами.

Основні особливості алгоритму Ші-Томасі:

1. Для кожного пікселя обчислюється матриця градієнтів у локальному вікні.
2. Обчислюються власні значення λ_1 та λ_2 цієї матриці.
3. Якщо $\min(\lambda_1, \lambda_2) >$ порогове_значення, то піксель вважається кутом (особливою точкою).

В OpenCV алгоритм Ші-Томасі реалізований у функції `cv2.goodFeaturesToTrack()`, яка знаходить "сильні" кути на зображенні. Ці точки потім можна відстежувати між кадрами за допомогою алгоритму Лукаса-Канаде (`cv2.calcOpticalFlowPyrLK()`).

Переваги алгоритму Ші-Томасі:

- Більш стабільне відстеження точок порівняно з детектором Харріса
- Краща продуктивність при зміні освітлення та невеликих деформаціях
- Ефективний для задач відстеження руху об'єктів на відеострімах

Цей алгоритм широко використовується в системах комп'ютерного зору для відстеження об'єктів, стабілізації відео, створення панорамних зображень та інших застосувань, де потрібно відстежувати рух між кадрами.

Висновок

Бібліотека OpenCV є потужним інструментом для обробки зображень та відео. Вона дозволяє ефективно відстежувати рух об'єктів на відеострімах, що є важливим для багатьох задач, таких як відстеження руху, розпізнавання жестів, аналіз сцен та ін.

Хід роботи

1. Підготовка середовища

1.1. Клонування репозиторію з кодом для лабораторної роботи:

Для початку роботи необхідно клонувати репозиторій з вихідним кодом. Відкрийте термінал (командний рядок) та виконайте наступну команду:

```
git clone git@github.com:bausk/ar-practice-2025.git
cd ar-practice-2025/lab-2025-03-01
```

1.2. Встановлення miniconda і залежностей.

Miniconda - це мінімальний інсталятор для Conda, який включає тільки Conda, Python та кілька інших необхідних пакетів. Це дозволяє створювати ізольовані середовища для різних проектів.

Встановіть miniconda за допомогою офіційного сайту.

https://www.anaconda.com/docs/getting-started/miniconda/install

1.3. Створення віртуального середовища та встановлення залежностей

Після встановлення Miniconda та клонування репозиторію, необхідно створити віртуальне середовище та встановити всі необхідні залежності. Переконайтеся, що ви знаходитесь у директорії проекту `lab-2025-03-01` і що нема активованих інших середовищ (командна строка не має показувати `(base)` чи щось подібне).

```
conda env create -n lab-2025-03 -f environment.yml
```

```
conda activate lab-2025-03
```

2. Запуск програми

Після налаштування середовища можна запустити програму для відстеження руху на відеострімі. Програма підтримує різні режими роботи та параметри.

2.1. Запуск з веб-камери

Для запуску програми з використанням веб-камери виконайте наступну команду:

```
python calculate.py
```

2.2. Запуск з відеофайлом

Для запуску програми з використанням відеофайлу (по бажанню і якщо є власне відео в форматі MP4) виконайте наступну команду:

```
python calculate.py --image_path=video.mp4
```

2.3. Додаткові параметри

Програма підтримує додаткові параметри для налаштування:

- `--step` - інтервал між захопленими кадрами в секундах
- `--crop` - відсоток горизонтального поля зору для збереження (за замовчуванням 100%)
- `--maxwidth` - максимальна ширина для зменшення розміру зображення (за замовчуванням без зміни розміру)

Приклад:

```
python calculate.py --step=50 --crop=0.5 --maxwidth=1024
```

3. Взаємодія з програмою

Під час роботи програми ви можете використовувати наступні клавіші для керування:

- `1` - перемикання на щільний оптичний потік з візуалізацією HSV кольорами (за замовчуванням)
- `2` - перемикання на щільний оптичний потік з візуалізацією лініями
- `3` - перемикання на метод Лукаса-Канаде з генераціямі якорних точок
- `s` - збереження поточного зображення
- `f` - горизонтальне відображення зображення
- `Пробіл` - пауза/продовження відтворення
- `ESC` - вихід з програми

4. Аналіз результатів

Під час роботи програми ви побачите візуалізацію руху на відеострімі. Залежно від обраного режиму, це може бути:

1. Кольорове зображення, де колір відповідає напрямку руху, а яскравість - швидкості
2. Лінії, що показують напрямок та величину руху
3. Деформоване зображення, що відображає рух
4. Точки та вектори, що відстежуються методом Лукаса-Канаде.

ХІД ВИКОНАННЯ РОБОТИ

1. Запустіть програму за допомогою команди `python calculate.py`.

Спостерігайте за помилкою, яка виникає при запуску програми.

2. Виправте помилку у коді. Додайте правильну ініціалізацію `create_default_lk_flow`.

```
    create_default_lk_flow = lambda: optical_flow.LucasKanadeOpticalFlow(
        feature_params=dict(
            maxCorners=100,
            qualityLevel=0.2,
            minDistance=15,
            blockSize=7,
            useHarrisDetector=False
        ),
        lk_params=dict(
            winSize=(15, 15),
            maxLevel=2,
            criteria=(
                cv2.TERM_CRITERIA_EPS | cv2.TERM_CRITERIA_COUNT,
                10,
                0.03
            )
        ),
    )
```

Запустіть виправлену програму. Виберіть режим роботи за допомогою клавіш `1`, `2`, `3`.

3. Використовуйте клавіші `s`, `f`, `Пробіл`, `ESC` для керування програмою.
4. Ознайомтеся з кодом програми. Зробіть скріншоти інтерфейсу програми в трьох режимах роботи.
5. Знайдіть, в якій саме частині коду проекту в режимі `3` використовується метод розрідженого оптичного потоку за Лукасом-Канаде, тобто викликається відповідна функція пакету машинного зору `cv2`. Занотуйте це в вашому звіті.
6. Проаналізуйте різницю між щільним оптичним потоком та розрідженим оптичним потоком. Опишіть в звіті, які переваги та недоліки щільного оптичного потоку та розрідженого оптичного потоку, який виконується швидше і так далі.
7. Подивіться документацію до OpenCV функції розрідженого оптичного потоку за Лукасом-Канаде (https://docs.opencv.org/3.4/dc/d6b/group__video__track.html або пошук в інтернеті за ключовим словом 'calcOpticalFlowPyrLK') та занотуйте в звіті, що означають параметри `lk_params` в ініціалізації оптичного потоку, яку ви реалізували в пункті 1.
8. Поміняйте код ініціалізації `create_default_lk_flow` вище наступним чином.

За ключовим словом 'goodFeaturesToTrack' знайдіть використання параметрів `feature_params`.

Це параметри для алгоритму Ші-Томасі, що використовується для відстеження "особливих точок" (features), які використовуються для **розрідженого відстеження руху**.

Оберіть різні значення feature_params.

Зверніть увагу:

Після знаходження особливих точок ми використовуємо це покоління точок для відстеження руху. Деяку кількість кадрів ми відстежуємо рух цих точок і зберігаємо їхні нові позиції, затім формується наступне покоління точок і процес повторюється.

Прослідкуйте, як програма припинить роботу, якщо в рамках одного покоління не вдасться знайти достатню кількість точок для відстеження руху.

Симулюйте різкий рух об'єкта на зображенні, такий, що виводить особливі точки за межі зображення.

Опишіть в звіті, які кожен з параметрів впливає на результат роботи алгоритму знаходження "особливих точок" (в режимі '3').

Запустіть алгоритм у режимі '3' з наступними значеннями параметрів `feature_params` (порахуйте значення параметру A та поміняйте значення параметрів у коді):

Параметр A = Номер в алфавіті першої літери вашого імені + номер в алфавіті першої літери вашої прізвища

```
feature_params=dict(
   maxCorners= A * 2, # кількість особливих точок
   qualityLevel= A / 100, # якість особливих точок, які братимуть участь у відстеженні
   minDistance= A * 5, # мінімальна відстань між особливими точками
   blockSize=A * 2, # розмір блоку для оцінки особливих точок
```

```
    useHarrisDetector=False # НЕ використовувати Harris детектор
)
```

Прослідкуйте, чи при даних параметрах алгоритм коректно відстежує особливі точки і чи програма припиняє роботу з помилкою при помірних рухах вашої камери або об'єкта.

Опишіть в звіті ваші результати і ваші висновки про те, покращилась чи погіршилась (і чому) якість відстеження руху при зміні параметрів `feature_params`.

УМОВА ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ

1. Виявити та виправитипомилку в коді програми, пов'язану з ініціалізацією функції `create_default_lk_flow`.
2. Досліджено три режими роботи програми для відстеження руху:
    - Режим 1: Щільний оптичний потік з візуалізацією HSV
    - Режим 2: Щільний оптичний потік з візуалізацією ліній
    - Режим 3: Метод розрідженого оптичного потоку за Лукасом-Канаде
3. Проаналізовано код програми та визначено місце виклику функції `cv2.calcOpticalFlowPyrLK()` для реалізації методу Лукаса-Канаде.
4. Порівняно переваги та недоліки щільного та розрідженого оптичного потоку:
    - Швидкість обчислення
    - Точність відстеження
    - Ресурсоємність
    - Стійкість до шумів
5. Вивчено параметри функції `calcOpticalFlowPyrLK` та їх вплив на результати роботи алгоритму.
6. Експериментально досліджено вплив різних значень параметрів `feature_params` на якість відстеження точок у режимі розрідженого оптичного потоку.
7. Отримано практичні навички роботи з алгоритмами комп'ютерного зору для відстеження руху в відеопотоці.

ІНДІВІДУАЛЬНІ ВАРІАНТИ ЗАВДАННЯ

Використати параметри оптичного потоку в залежності від імені та прізвища автора звіту:

Параметр A = Номер в алфавіті першої літери вашого імені + номер в алфавіті першої літери вашої прізвища

```
feature_params=dict(
   maxCorners= A * 2, # кількість особливих точок
   qualityLevel= A / 100, # якість особливих точок, які братимуть участь у відстеженні
   minDistance= A * 5, # мінімальна відстань між особливими точками
```

```
    blockSize=A * 2, # розмір блоку для оцінки особливих точок
    useHarrisDetector=False # НЕ використовувати Harris детектор
)
```

ЗМІСТ ЗВІТУ

1. Тема та мета роботи
2. Теоретичні відомості
3. Постановка завдання
4. Хід виконання роботи:
    - Скріншоти інтерфейсу програми
    - Скріншоти виправлених помилок
    - Скріншоти результатів роботи в різних режимах роботи, при щільному та розрідженому оптичному потоці
    - Скріншоти результатів роботи з різними параметрами `feature_params`
5. Результати роботи відповідно до того що було зазначено в ході роботи
6. Висновки

КОНТРОЛЬНІ ПИТАННЯ

1. Що таке оптичний потік і задача визначення власного руху?
2. Що таке щільний оптичний потік?
3. Що таке розріджений оптичний потік?
4. Що таке особливі точки?
5. Що таке алгоритм Ші-Томасі?
6. Опишіть своїми словами застосування алгоритму Лукаса-Канаде.

СПИСОК ЛІТЕРАТУРИ

1. https://en.wikipedia.org/wiki/Lucas%E2%80%93Kanade_method
2. https://docs.opencv.org/3.4/dc/d6b/group__video__track.html
3. https://en.wikipedia.org/wiki/Optical_flow
4. https://en.wikipedia.org/wiki/Corner_detection