Лабораторна робота №1
Робота з зображеннями: читання, візуалізація, базові операції

Мета роботи: ознайомлення з базовими операціями обробки цифрових зображень у середовищі Python, формування практичних навичок роботи з бібліотеками для комп'ютерного зору та підготовка до подальшого вивчення математичних методів аналізу зображень.

Короткі теоретичні відомості

Цифрове зображення – це дискретне представлення візуальної інформації у вигляді двовимірної або тривимірної матриці пікселів. Кожен піксель характеризується значенням інтенсивності або набором значень інтенсивностей для різних каналів.

У комп'ютерному зорі найпоширенішими є: градаційні (grayscale) зображення – один канал яскравості; кольорові зображення – зазвичай три канали (RGB або BGR).

Зображення в середовищі Python представляються у вигляді багатовимірних масивів типу numpy.ndarray.

Основні параметри цифрового зображення: розмірність (висота × ширина × кількість каналів); глибина кольору; тип даних (uint8, float32 тощо).

Базові операції обробки зображень є фундаментом для подальших математичних методів комп'ютерного зору, таких як фільтрація, виділення ознак, сегментація та розпізнавання образів.

Використовувані бібліотеки Python.

У лабораторній роботі використовуються такі бібліотеки:

NumPy – використовується для роботи з багатовимірними масивами, математичних операцій над пік селями:

```
import numpy as np
```

OpenCV (cv2) – основна бібліотека комп'ютерного зору для зчитування, обробки та перетворення зображень:

```
import cv2
```

Matplotlib – застосовується для візуалізації зображень та порівняння результатів обробки:

```
import matplotlib.pyplot as plt
```

За потреби може використовуватись Pillow (PIL) як альтернативний інструмент роботи з зображеннями.

Етапи виконання лабораторної роботи

Етап 1. Підготовка робочого середовища

Встановити та підключити необхідні бібліотеки Python для обробки та візуалізації зображень.

Приклад тексту програми:

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
```

Етап 2. Зчитування цифрового зображення з файлу Зчитати кольорове растрове зображення з локального каталогу та перевірити коректність його завантаження.

Приклад тексту програми:

```
image = cv2.imread("image.jpg")

if image is None:
    print("Помилка зчитування зображення")
else:
    print("Зображення успішно завантажене")
```

Етап 3. Аналіз структури зображення Дослідити розмірність, кількість каналів та тип даних масиву пікселів.

Приклад тексту програми:

```
print("Розмірність:", image.shape)
print("Тип даних:", image.dtype)
```

Етап 4. Візуалізація зображення Відобразити зображення з використанням бібліотеки Matplotlib з урахуванням особливостей колірної моделі OpenCV.

Приклад тексту програми:

```
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

plt.imshow(image_rgb)
plt.title("Оригінальне зображення")
plt.axis("off")
plt.show()
```

Етап 5. Перетворення зображення в відтінки сірого Виконати перетворення кольорового зображення у градаційне та порівняти результати.

Приклад тексту програми:

```
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

plt.imshow(gray, cmap="gray")
plt.title("Градаційне зображення")
plt.axis("off")
plt.show()
```

Етап 6. Зміна розміру зображення Зменшити або збільшити розмір зображення, використовуючи інтерполяцію.

Приклад тексту програми:

```
resized = cv2.resize(image, (300, 300))

plt.imshow(cv2.cvtColor(resized, cv2.COLOR_BGR2RGB))
plt.title("Змінений розмір")
plt.axis("off")
plt.show()
```

Етап 7. Обрізання фрагмента зображення (Crop) Виділити прямокутну область інтересу та візуалізувати її.

Приклад тексту програми:

```
crop = image[50:250, 100:350]

plt.imshow(cv2.cvtColor(crop, cv2.COLOR_BGR2RGB))
plt.title("Обрізане зображення")
plt.axis("off")
plt.show()
```

Етап 8. Інверсія яскравості зображення Реалізувати інверсію значень пікселів для отримання негативу.

Приклад тексту програми:

```
inverted = cv2.bitwise_not(image)

plt.imshow(cv2.cvtColor(inverted, cv2.COLOR_BGR2RGB))
plt.title("Інверсія яскравості")
plt.axis("off")
plt.show()
```

Етап 9. Зміна яскравості та контрастності Реалізувати лінійне перетворення інтенсивності пікселів.

Приклад тексту програми:

```
alpha = 1.5  # контраст
```

```
beta = 30    # яскравість

adjusted = cv2.convertScaleAbs(image, alpha=alpha, beta=beta)

plt.imshow(cv2.cvtColor(adjusted, cv2.COLOR_BGR2RGB))
plt.title("Зміна яскравості та контрастності")
plt.axis("off")
plt.show()
```

Етап 10. Порівняльний аналіз результатів Побудувати візуальне порівняння вихідного та оброблених зображень і зробити висновки щодо впливу кожної операції.

Приклад тексту програми:

```
plt.figure(figsize=(10,5))

plt.subplot(1,2,1)
plt.imshow(image_rgb)
plt.title("Оригінал")
plt.axis("off")

plt.subplot(1,2,2)
plt.imshow(gray, cmap="gray")
plt.title("Градації сірого")
plt.axis("off")

plt.show()
```

Етап 11. Дослідити, як зміна типу даних (uint8 → float32) впливає на точність та стабільність обчислень при обробці зображень.

Приклад тексту програми:

```
image_float = image.astype(np.float32) / 255.0
print(image_float.min(), image_float.max())
```

Завдання до лабораторної роботи

1. Ознайомитися з поняттям цифрового зображення та його поданням у пам’яті комп’ютера.
2. Навчитися зчитувати растрові зображення з файлу за допомогою бібліотек Python.
3. Реалізувати візуалізацію зображень у різних колірних просторах.
4. Дослідити структуру масиву пікселів (розмірність, тип даних).

5. Виконати базові операції над зображеннями: перетворення в відтінки сірого; зміну розміру зображення; обрізання (crop); інверсію яскравості; зміну яскравості та контрастності.
6. Проаналізувати вплив базових операцій на візуальні характеристики зображення.
7. Оформити звіт та завантажити до Системи електронного забезпечення навчання ЗНУ. Звіт повинен містити мету роботи, завдання роботи, тексти програм, висновки

Контрольні питання

1. Що таке цифрове зображення та як воно представляється в пам’яті комп’ютера?
2 Яка різниця між градаційним та кольоровим зображенням?
3. Чим відрізняються колірні моделі RGB та BGR?
4. Яку роль відіграє бібліотека NumPy в обробці зображень?
5. Які типи даних найчастіше використовуються для збереження пікселів і чому?
6. У чому полягає операція перетворення зображення в відтінки сірого?
7. Як зміна яскравості та контрастності впливає на значення пікселів?
8. Які переваги використання OpenCV у задачах комп’ютерного зору?