Київський політехнічний інститут імені Ігоря Сікорського

Приладобудівний факультет

Кафедра автоматизації та систем неруйнівного контролю

Практична робота №3

з предмету "Комп'ютерний зір"

Студент: Погорєлов Богдан

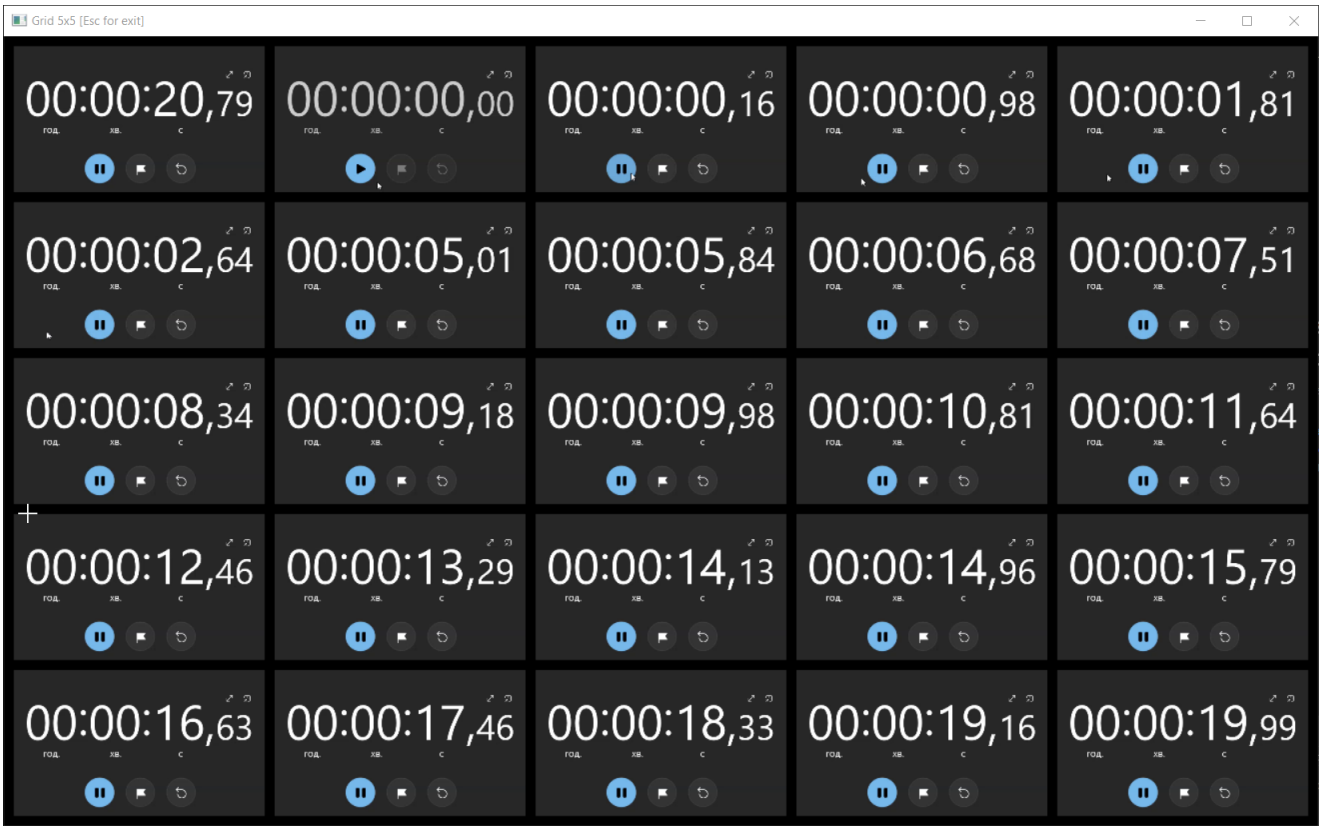
Група: ПК-51мп

2025 рік

Тема: Мозаїчне захоплення 25 кадрів

Мета роботи

Метою нашої практичної роботи була розробка програми на мові Python з використанням бібліотеки **OpenCV**. Ми ставили за ціль створити відеоплеєр, що відображає кадри з відеофайлу у вигляді сітки **5x5**, де кожен новий кадр, взятий через певний інтервал, послідовно оновлює одну з 25 комірок.



Опис реалізації

Для виконання завдання ми реалізували програму на Python, залучивши наступні бібліотеки:

- **OpenCV (`cv2`)**: для захоплення, читання та відображення відеокадрів.
- **NumPy**: для створення та маніпуляцій з багатовимірними масивами, які представляють зображення (полотно сітки).
- **Tkinter**: для визначення розширення екрана користувача, щоб адаптувати розмір вікна програми.

Алгоритм роботи програми:

1. **Визначення розширення екрана**: На початку програма отримує розміри екрана для коректного масштабування вікна.
2. **Створення полотна для сітки**: Далі обчислюються оптимальні розміри для комірок сітки на основі розмірів кадру відео та екрана. Створюється чорне фонове зображення (`grid_canvas`) і

розраховуються координати для кожної з 25 комірок.

3. **Головний цикл обробки відео**:
    - Ми відкриваємо відеофайл `video.mp4`.
    - У циклі програма зчитує кадри, але для відображення відбирає лише кожен **25-й кадр** (`n_frame_step=25`).
    - Відібраний кадр масштабується до розміру комірки і розміщується у наступній позиції в сітці (з 1-ї до 25-ї, а потім знову по колу).
    - Оновлене полотно з сіткою відображається у вікні `Grid 5x5 [Esc for exit]`.
    - Робота програми завершується натисканням клавіші **Esc**.
    - Коли відео добігає кінця, відтворення починається знову.

Як запустити

1. Переконайтеся, що у вас встановлено Python та необхідні бібліотеки:

```
pip install opencv-python numpy
```

2. Збережіть код у файл (наприклад, `main.py`).
3. Помістіть відеофайл з назвою `video.mp4` в ту саму директорію, де знаходиться скрипт.
4. Запустіть скрипт з терміналу:

```
python main.py
```

Висновок

В ході виконання практичної роботи **ми** успішно розробили програму для відображення відео у форматі сітки 5x5. **Ми** ознайомились з основними функціями бібліотеки **OpenCV** для захоплення та обробки відеокадрів, навчились динамічно створювати та оновлювати зображення за допомогою **NumPy**, а також адаптувати розмір вікна до розширення екрана. Програма виконує поставлене завдання, послідовно оновлюючи комірки сітки кадрами з відео, що демонструє отримані **нами** практичні навички з комп'ютерного зору.

Лістинг

```
import cv2
import numpy as np
import tkinter as tk
from typing import Tuple, List

def get_screen_resolution() -> Tuple[int, int]:
    '''Використовує tkinter для отримання розмірів екрана.'''
    try:
```

```
        root = tk.Tk()
        root.withdraw()
        screen_width = root.winfo_screenwidth()
        screen_height = root.winfo_screenheight()
        root.destroy()
        return screen_width, screen_height
    except Exception as e:
        print(f"Не вдалося визначити розширення екрана: {e}. Використовується
стандартне 1920x1080.")
        return 1920, 1080

def setup_grid_canvas(
    frame_shape: Tuple[int, int, int],
    screen_shape: Tuple[int, int],
    border_window: float = 0.85,
    border_frame: int = 10
) -> Tuple[np.ndarray, List[Tuple[int, int, int, int]], Tuple[int, int]]:
    '''Розраховує розміри, створює полотно з рамками та попередньо обчислює
координати комірок.'''
    frame_h, frame_w, _ = frame_shape
    screen_w, screen_h = screen_shape

    scale_factor = min((screen_w * border_window) / (frame_w * 5),
                       (screen_h * border_window) / (frame_h * 5), 1.0)

    scaled_cell_w, scaled_cell_h = int(frame_w * scale_factor), int(frame_h *
scale_factor)
    scaled_dim_single = (scaled_cell_w, scaled_cell_h)

    grid_canvas_w = (scaled_cell_w * 5) + (border_frame * 6)
    grid_canvas_h = (scaled_cell_h * 5) + (border_frame * 6)

    print(f"Розмір екрана: {screen_w}x{screen_h}")
    print(f"Розмір полотна сітки: {grid_canvas_w}x{grid_canvas_h}")

    grid_canvas = np.full((grid_canvas_h, grid_canvas_w, 3), 0, dtype=np.uint8)

    cell_coordinates = []
    for i in range(25):
        row, col = i // 5, i % 5
        y1 = (row * scaled_cell_h) + ((row + 1) * border_frame)
        y2 = y1 + scaled_cell_h
        x1 = (col * scaled_cell_w) + ((col + 1) * border_frame)
        x2 = x1 + scaled_cell_w
        cell_coordinates.append((y1, y2, x1, x2))

    return grid_canvas, cell_coordinates, scaled_dim_single

def create_video_grid_player(video_path: str, n_frame_step: int) -> None:
    '''Захоплює кожен n-ний кадр і динамічно оновлює сітку 5x5.'''
    cap = cv2.VideoCapture(video_path)
    if not cap.isOpened():
        print(f"Помилка: не вдалося відкрити відеофайл '{video_path}'")
```

```
            return

        ret, first_frame = cap.read()
        if not ret:
            print("Помилка: не вдалося прочитати перший кадр з відео.")
            cap.release()
            return
        cap.set(cv2.CAP_PROP_POS_FRAMES, 0)

        grid_canvas, cell_coords, scaled_dim = setup_grid_canvas(
            first_frame.shape, get_screen_resolution()
        )

        frame_count, update_index = 0, 0

        while cap.isOpened():
            ret, frame = cap.read()
            if not ret:
                print("Кінець відео. Повторне відтворення.")
                cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
                continue

            frame_count += 1
            if frame_count % n_frame_step == 0:
                y1, y2, x1, x2 = cell_coords[update_index]
                update_index = (update_index + 1) % 25

                grid_canvas[y1:y2, x1:x2] = cv2.resize(frame, scaled_dim)

                cv2.imshow('Grid 5x5 [Esc for exit]', grid_canvas)


            if cv2.waitKey(1) & 0xFF == 27: # 27 (код клавіші Escape)
                break

        cap.release()
        cv2.destroyAllWindows()


    if __name__ == '__main__':
        create_video_grid_player(video_path='video.mp4', n_frame_step=25)
```