**Video Capture and Playback Using OpenCV**

# Introduction

This tutorial will guide you through using two Python programs to capture video from your webcam, save it as a file, and later play it back. We will cover how to run these programs and explain the code so you can understand how everything works.

# Requirements

Before running these programs, ensure you have the required dependencies installed.

## Install OpenCV

```
pip install opencv-python
```

---

# Part 1: Capturing and Saving Video

The first program, `capture.py`, records video from your webcam and saves it to a file.

## Running the Capture Program

To start the capture program, run the following command in your terminal or command prompt:

```
python capture.py
```

## Controls:

- **Press 's'** to **start or resume** recording.
- **Press 'p'** to **pause** recording.
- **Press 'q'** to **stop and save** the recording.

## Code Breakdown

### 1. Importing OpenCV

```
import cv2
```

We use OpenCV (`cv2`) to handle video capture and saving.

### 2. Opening the Webcam

```
cap = cv2.VideoCapture(0)
```

This line opens the default camera (0). If you have multiple cameras, change the number (e.g., 1 for a secondary camera).

### 3. Setting Up Video Writer

```
frame_width = int(cap.get(3))
frame_height = int(cap.get(4))
fourcc = cv2.VideoWriter_fourcc(*'mp4v')  # Codec for Mac/Linux
out = cv2.VideoWriter('output.mp4', fourcc, 20.0, (frame_width,
frame_height))
```

- Retrieves the frame size from the camera.
- Defines the codec (`mp4v`) to save the video in MP4 format.
- Initializes `VideoWriter` to save the recorded frames.

### 4. Capturing and Saving Frames

```
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break

    cv2.imshow('Video', frame)
```

- Reads frames from the camera.
- Displays the current frame in a window.

### 5. Recording Control

```
    key = cv2.waitKey(1) & 0xFF

    if key == ord('s'):
        recording = True
        print("Recording started...")
    elif key == ord('p'):
        recording = False
        print("Recording paused.")
    elif key == ord('q'):
        print("Recording stopped and saved.")
        break
```

- Waits for key inputs to start, pause, or stop recording.

### 6. Writing Frames to the File

```
    if recording:
        out.write(frame)
```

- When recording is active, frames are saved to `output.mp4`.

### 7. Releasing Resources

```
cap.release()
out.release()
cv2.destroyAllWindows()
```

- Ensures all resources are properly released.

---

# Part 2: Playing the Saved Video

Once a video is recorded, you can play it back using `play-video.py`.

## Running the Playback Program

```
python play-video.py output.mp4
```

## Controls:

- **Press 'q'** to quit playback.

## Code Breakdown

### 1. Importing Dependencies and Handling Arguments

```
import cv2
import argparse
```

- `cv2` is used for video playback.
- `argparse` allows the user to specify which video file to play.

### 2. Opening the Video File

```
def play_video(video_path):
    cap = cv2.VideoCapture(video_path)
```

- Opens the specified video file.

### 3. Checking If the File Opened Successfully

```
    if not cap.isOpened():
        print(f"Error: Could not open video file {video_path}")
        return
```

- Displays an error message if the file cannot be opened.

### 4. Playing the Video

```
while True:
    ret, frame = cap.read()
    if not ret:
        break  # Exit when the video ends

    cv2.imshow("Video Player", frame)
```

- Reads and displays each frame from the video.
- Stops when the video ends.

### 5. Handling Quit Input

```
    if cv2.waitKey(25) & 0xFF == ord("q"):
        break
```

- Allows the user to exit playback by pressing 'q'.

### 6. Releasing Resources

```
cap.release()
cv2.destroyAllWindows()
```

- Ensures the video file is closed properly.

---

# Conclusion

With these two programs, you can:

1. **Capture video** from your webcam and save it as an `.mp4` file.
2. **Play back** the recorded video with controls for stopping playback.

These programs can be modified to add additional features such as different codecs, recording time limits, or saving videos in different formats.

**Try experimenting with different settings to understand how OpenCV handles video processing!**