

Informe de Laboratorio

Procesamiento de Video con Algoritmos Secuenciales y Paralelos

Transformación de video a escala de grises mediante algoritmos secuenciales y paralelos multicore.

2. Objetivos

- Implementar un algoritmo secuencial para convertir un video a escala de grises.
- Implementar un algoritmo paralelo multicore para optimizar el tiempo de procesamiento.
- Comparar el rendimiento entre ambos algoritmos mediante el cálculo del *speedup*.
- Reconstruir videos procesados a partir de sus fotogramas.

3. Marco Teórico

El procesamiento de video consiste en manipular secuencias de imágenes (frames) para transformarlas o analizarlas. Un video está compuesto por cientos de fotogramas que se muestran de manera continua para generar la ilusión de movimiento.

La conversión a escala de grises implica transformar cada píxel del espacio de color RGB a un valor único de intensidad luminosa usando operaciones como `cv2.cvtColor()`.

Los algoritmos secuenciales ejecutan cada instrucción una tras otra, mientras que los algoritmos paralelos distribuyen el trabajo entre múltiples núcleos del procesador, acelerando el tiempo total de ejecución. Esto permite medir el rendimiento comparativo usando:

Speedup:

$$\text{Speedup} = \text{TiempoSecuencial} / \text{TiempoParalelo}$$

4. Metodología

Hardware

- CPU multinúcleo (Google Colab)

Software

- Python 3
- OpenCV
- `concurrent.futures`

Procedimiento

1. Se cargó el video entregado por el estudiante.
2. Se extrajeron 261 fotogramas en color.
3. Se desarrollaron y ejecutaron dos algoritmos:
4. Conversión secuencial.

5. Conversión paralela multicore.
6. Se midieron los tiempos de ejecución de ambos métodos.
7. Se reconstruyeron dos videos en escala de grises.
8. Se calcularon métricas de rendimiento incluyendo el *speedup*.

5. Resultados

Proceso	Tiempo (s)
Secuencial	0.5087
Paralelo	0.4848

Speedup: 1.049

Ambos métodos producen la conversión correcta del video, con una ligera mejora en el tiempo al usar el algoritmo paralelo.

6. Algoritmos Desarrollados

Algoritmo Secuencial

- Procesa cada frame uno por uno.
- Utiliza un solo hilo de ejecución.
- Aplica transformación a escala de grises mediante OpenCV.

Algoritmo Paralelo

- Crea múltiples hilos con `ThreadPoolExecutor`.
- Divide el trabajo entre los núcleos disponibles.
- Acelera el tiempo total para grandes cantidades de frames.

Entradas

- Video original en color.

Salidas

- Video transformado a escala de grises usando:
- Método secuencial.
- Método paralelo multicore.

7. Cómo se usa el código

El proyecto funciona ejecutando el archivo `codigo.py`, el cual realiza automáticamente:

- La lectura del video original.
- La extracción de los frames.
- La conversión secuencial y paralela a escala de grises.
- El cálculo del tiempo y del speedup.
- La reconstrucción de los dos videos procesados.
- La creación del archivo .zip con toda la evidencia generada.

Si desea verificar el funcionamiento:

1. Solo debe ejecutar el código en Google Colab o Python con OpenCV instalado.
2. El video original debe estar en el mismo directorio donde se ejecuta el código.
3. El script generará automáticamente todas las carpetas y resultados.

7. Conclusiones

- El laboratorio demuestra la diferencia entre ejecución secuencial y paralela.
- El método paralelo ofrece una mejora ligera para videos pequeños.
- Para videos más grandes, el speedup puede aumentar significativamente.
- La práctica permite comprender el impacto del paralelismo en tareas multimedia.