Universidad de Guadalajara
Centro Universitario
de
Ciencias Exactas
e
Ingenierías
(CUCEI)

Materia: COMPUTACION TOLERANTE A FALLAS

Nombre:
Jonathan Aceves López

Código:

217481363

NRC: 179961

11/09/2023

Introducción
El procesamiento de imágenes es una tarea común en muchas aplicaciones, desde la edición de fotos hasta la visión por computadora. En este informe, se presenta un ejemplo de procesamiento de imágenes utilizando hilos en Python sin el uso de las bibliotecas cv2 ni Pillow. El objetivo es cargar imágenes desde una carpeta llamada "imagenes", invertir sus colores y guardarlas en una carpeta "imagenes_procesadas". Se utiliza la biblioteca threading para ejecutar estas tareas de forma concurrente.

Desarrollo
Configuración del Entorno
Antes de comenzar el procesamiento de imágenes, se realiza una configuración inicial del entorno:

Se especifica el directorio de entrada ("imagenes") y el directorio de salida ("imagenes_procesadas").
Se verifica si el directorio de salida existe; si no, se crea.

```python
import threading
import os
from io import BytesIO

# Directorio de entrada y salida
input_dir = "imagenes"
output_dir = "imagenes_procesadas"

# Crear la carpeta de salida si no existe
if not os.path.exists(output_dir):
    os.mkdir(output_dir)
```

Procesamiento de Imágenes
La función procesar_imagen se utiliza para cargar una imagen, invertir sus colores y guardarla en la carpeta de salida. Cada imagen se procesa en un hilo separado para aprovechar la concurrencia:

```python
# Función para cargar una imagen, invertir colores y guardarla
def procesar_imagen(filename):
    input_path = os.path.join(input_dir, filename)
    output_path = os.path.join(output_dir, filename)

    print(f"Procesando imagen: {filename}")

    try:
        # Cargar la imagen
        with open(input_path, 'rb') as file:
            image_data = file.read()

        image = Image.open(BytesIO(image_data))

        # Invertir los colores de la imagen
        inverted_image = Image.new("RGB", image.size)
        inverted_pixels = [(255 - r, 255 - g, 255 - b) for (r, g, b) in image.getdata()]
        inverted_image.putdata(inverted_pixels)

        # Guardar la imagen procesada
        with open(output_path, 'wb') as file:
            inverted_image.save(file, format="PNG")

        print(f"Imagen procesada y guardada en: {output_path}")
    except Exception as e:
        print(f"Error al procesar la imagen {filename}: {str(e)}")
```

Ejecución Concurrente

Se obtiene una lista de archivos de imagen en el directorio de entrada y se crea un hilo para procesar cada imagen. Luego, se inician los hilos y se espera a que todos terminen:

```python
# Lista de archivos en la carpeta de entrada
image_files = os.listdir(input_dir)

# Creamos un hilo para procesar cada imagen
threads = []
for filename in image_files:
    thread = threading.Thread(target=procesar_imagen, args=(filename,))
    threads.append(thread)
    thread.start()

# Esperamos a que todos los hilos terminen
for thread in threads:
    thread.join()

print("Procesamiento de imágenes completado.")
```