Proyecto PDI

Homero Meneses Vázquez
Jesús Alejandro Guzmán Cordero
Enrique Ramirez Pérez
Roberto Misael Reyes

June 2023

1 Introduction

Este proyecto es con el fin de demostrar lo aprendido durante el curso y enfocarlo en una función o tarea que nos agrada por su uso y construcción. El procesamiento digital de imágenes es el conjunto de técnicas que se aplican a las imágenes digitales con el objetivo de mejorar su calidad o facilitar su interpretación automática . En Python, el procesamiento digital de imágenes se puede realizar utilizando diversas bibliotecas, que son tecnologías relativamente nuevas como Pillow, OpenCV y Scikit-image, entre otras.

Sin embargo, el propósito de este proyecto es programar con python, pero contruyendo el codigo sin el uso de bibliotecas que ya hacen el trabajo por nosotros, esto para la comprensión y un desarrollo más cercano al origen matemático de esta area. Como aplicar diversas técnicas de procesamiento de imágenes, como filtrado, segmentación, detección de bordes, entre otras. También se pueden utilizar técnicas de aprendizaje automático para realizar tareas más complejas, como el reconocimiento de objetos o rostros. En general, Python es una herramienta poderosa para el procesamiento digital de imágenes debido a su facilidad para trabajar con matrices , lo que permite un tratamiento eficiente y fácil de las imágenes como matrices numéricas en 2D o 3D.

2 Desarrollo

Para este proyecto usamos tres filtros con el propósito de tener información más clara y precisa de una imagen, el primero siendo el filtro homomórfico el cual combina aspectos de dos o más dominios, como una función de mapeo no lineal y un filtro de frecuencia espacial. Si suponemos un modelo de imagen en el que las variaciones de alta frecuencia espacial se atribuyen a los componentes reflectantes de la imagen y las variaciones de baja frecuencia están asociadas a los componentes de iluminación, entonces un filtro de frecuencia espacial que opera sobre estos elementos de forma independiente puede efectuar cambios drásticos

en cualquiera de los componentes. En imágenes reales existe un problema porque los componentes de iluminación y reflectancia son multiplicativos. El modelo básico viene dado por: f(x,y) =I(x,y)R(x,y)

El programa asume que la imagen original es una imagen de 256 niveles de gris × IMAGEN-¿Filas × IMAGEN-¿Cols pixel almacenada en la estructura IMAGEN. El tipo de detección de línea deseada se pasa al programa dentro de la matriz. Luego, el programa calcula la imagen de línea detectada usando la máscara 3 × 3.

Una operación de filtro de mediana en una imagen elimina el ruido de cola larga, como el ruido exponencial negativo y el ruido de sal y pimienta, de una imagen con un mínimo desenfoque de la imagen. El filtro de mediana se define como la mediana de todos los píxeles dentro de una región local de una imagen. Los píxeles que se incluyen en el cálculo de la mediana se especifican mediante una máscara. El filtro de mediana funciona mucho mejor que el filtro de media aritmética para eliminar el ruido de sal y pimienta de una imagen y preservar los detalles espaciales contenidos en la imagen. El filtro de mediana puede eliminar fácilmente el ruido atípico de imágenes que contienen menos del 50 por ciento de sus píxeles como valores atípicos. La definición de un filtro de mediana en términos de una imagen A es: Mediana(A) = Mediana — A((x + i, y + j)], donde la coordenada x + i, y + j está definida en la imagen A y la coordenada i, j está definida en la máscara M. La máscara M determina qué píxeles se incluirán en el cálculo de la mediana.

Explicación del codigo

Este código en Python utiliza las librerías NumPy y PIL (Python Imaging Library) para aplicar dos filtros a una imagen de entrada: el filtro homomórfico y el filtro de detección de líneas. También incluye un filtro de mediana para suavizar la imagen.

La función homomorphic filter implementa el filtro homomórfico. En primer lugar, aplica el logaritmo a los píxeles de la imagen de entrada para mejorar el contraste. Luego, realiza la transformada de Fourier sobre la imagen logarítmica. Multiplica la transformada de Fourier por una imagen de filtro predefinida. A continuación, aplica la transformada inversa de Fourier para obtener la imagen filtrada. Finalmente, aplica la función exponencial a los píxeles de la imagen filtrada y realiza una normalización lineal para ajustar los valores entre 0 y 255.

La función line detector filter aplica el filtro de detección de líneas. Primero, convierte la imagen a escala de grises. Luego, calcula la magnitud del gradiente en la dirección horizontal de la imagen utilizando el método de diferencias finitas. Normaliza los valores entre 0 y 255 y los convierte a tipo entero de 8 bits. Esta operación resalta los bordes y líneas en la imagen.

La función median filter implementa el filtro de mediana. Convierte la imagen a escala de grises y aplica el filtro de mediana en cada píxel, utilizando una ventana de tamaño especificado para calcular el valor mediano de los píxeles vecinos. El resultado es una imagen suavizada.

En la función 'main', se carga una imagen de entrada utilizando la librería PIL. Luego, se crea una imagen de filtro para el filtro homomórfico y se aplican los tres filtros mencionados a la imagen de entrada. Finalmente, se muestra la

imagen original y las imágenes filtradas.

3 Conclusiones

Durante el transcurso de este proyecto debatimos sobre las diferentes ideas y los retos que se nos presentaban, en conclusión, este proyecto ha proporcionado un código funcional y eficiente para el procesamiento digital de imágenes, aplicando técnicas de filtrado como el filtro homomórfico, el filtro de detección de líneas y el filtro de mediana. Estas técnicas son útiles para mejorar la calidad de las imágenes, resaltar características relevantes y eliminar el ruido no deseado. El código puede ser utilizado como base para futuros proyectos y aplicaciones en el campo del procesamiento de imágenes.

Mejora de la calidad de la imagen: al comprender los efectos de varias operaciones y ecuaciones de procesamiento de imágenes, podemos elegir las técnicas apropiadas para mejorar la calidad visual de las imágenes. Esto incluye técnicas de procesamiento de imágenes como la reducción de ruido, la mejora del contraste, la nitidez, la corrección de color y otras características que pueden cambiar la forma en que se percibe y analiza una imagen.

Extracción de información pertinente: el procesamiento de imágenes digitales tiene usos más allá de la mejora estética, incluida la extracción de datos pertinentes de las imágenes. La comprensión de las ecuaciones y metodologías subyacentes nos permite reconocer y extraer características particulares de interés, como bordes, formas, texturas o patrones, que pueden ser esenciales en aplicaciones como el reconocimiento de objetos, el análisis de imágenes médicas o la detección de anomalías.

Desarrollo de algoritmos y técnicas personalizadas: Comprender los fundamentos y las ecuaciones detrás del procesamiento de imágenes nos permite desarrollar algoritmos y técnicas personalizadas según las necesidades específicas de una aplicación. Esto implica adaptar y combinar diferentes métodos existentes, así como proponer nuevos enfoques en retos nuevos.

Evaluación: Existen múltiples métodos y enfoques para abordar un mismo problema. Comprender las ecuaciones y los efectos de cada método nos permite evaluar y comparar su desempeño en términos de calidad, velocidad, precisión y otros criterios relevantes. Esto nos ayuda a seleccionar el método más adecuado para una tarea específica y a tomar decisiones informadas durante el proceso de desarrollo.

Solución de problemas y depuración: Estamos mejor equipados para identificar y solucionar problemas que puedan surgir durante la implementación de algoritmos y técnicas. Esto incluye la depuración de errores, la optimización del rendimiento, etectera.