UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

Laboratorio N 3: Diseño de filtros de imagenes

Integrante: Nicolás Gutiérrez

Curso: Redes de Computadores

Profesor(a): Carlos Gonzalez Cortes

17 de Mayo de 2019

Tabla de contenidos

1. Introducción

Una imagen natural capturada con una cámara, un microscopio o cualquier otro tipo de instrumento óptico presenta una variación de sombras y tonos, a este tipos de imágenes que no han pasado por un procesado se les conoce como imágenes analógicas. Cuando una imagen analógica necesita ser mostrada en un computador, esta debe ser procesada para ser admitida. A este tipo de imágenes se les conoce como imágenes digitales. Las imágenes digitales están discretizadas en los ejes X e Y ya que en cada punto tendrá un color. A cada punto se le conocerá como píxel.

Conociendo lo anterior, pocas veces se tiene noción de que ocurre cuando se aplica un filtro en alguna aplicación para manejar imágenes como Instagram. ¿Que tanto cambia la imagen al pasarla por un filtro?.

Este trabajo con imágenes se le conoce como procesamiento de imágenes digitales, el cual, en términos generales, envuelve al reconocimiento de imágenes 2D, 3D y secuencias de imágenes, análisis , manipulación, transmisión y otras áreas relacionadas.

En este laboratorio se verá un procesamiento de imágenes digitales con la aplicación de un filtro. El objetivo de este laboratorio es entender como es el proceso de la aplicación de un filtro, la aplicación de convoluciones en dos dimenciones y como afectan a la imagen la aplicación de distintos filtros o Kernels.

El documento está dividido de tal forma que se explican los conceptos y funciones a utilizar en la experiencia, pasando después a la parte de exposición de los resultados con su respecto análisis para finalmente llegar a la conclusión del informe.

2. Marco teórico

2.1. Kernel

En general se entiende Kernel como un núcleo. En particular para el caso de imágenes es una pequeña matriz que se utiliza para aplicar efectos como los que se pueden encontrar en Photoshop o Gimp, como desenfoque, nitidez, delineado, relieve ,etc.

outline

$$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Figura 1: Ejemplo de kernel

Dependiendo del kernel que se le aplique a una imagen será el resultado de esta misma. Cabe destacar que para aplicar un kernel se debe realizar una convolución de dos matrices.

2.2. Convolución

La convolución en general es un operador matemático que transforma dos funciones f y g en una tercera función. En palabras simples y para el caso particular del documento, es la superposición de las matrices, multiplicando los datos y sumándolos para obtener uno nuevo.

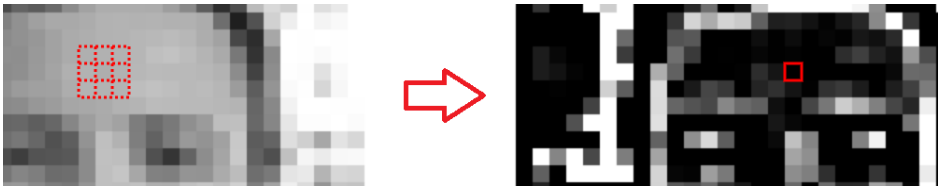

Figura 2: Ejemplo de convolución

Entonces como se ve en la Figura, de la imagen original los 9 píxeles fueron multiplicados por un kernel para luego sumarlos y obtener un valor de píxel.

3. Desarrollo de la experiencia

Para poder desarrollar la experiencia fue necesario utilizar Python 3 con las librerías $Numpy, Matplotlib, Scipy$ y PIL. Estas librerías son para el manejo de números complejos, imágenes, gráficas y funciones de conversión.

La idea de este laboratorio es abrir la siguiente imagen:

Figura 3: Imagen de ejemplo de Lenna

Con esta imagen se debe obtener la matriz de píxeles con los valores en escala de grises, además es necesario obtener la transformada de Fourier para luego hacer la comparación entre estas y las imágenes filtradas.

Con la matriz de valores, se debe hacer la convolución con los siguientes kernels matriz de valores, se debe hacer la convolución con los siguientes kernels

$$h = \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \qquad h = \begin{bmatrix} 1 & 2 & 0 & -2 & -1 \\ 1 & 2 & 0 & -2 & -1 \\ 1 & 2 & 0 & -2 & -1 \\ 1 & 2 & 0 & -2 & -1 \\ 1 & 2 & 0 & -2 & -1 \end{bmatrix}$$

Figura 4: Ejemplo de kernel

luego de aplicar los filtros de debe obtener la transforma de Fourier para ambos

filtros.

La forma en como se ha desarrollado esta experiencia se dividen en pasos o etapas las cuales son:

3.1. Obtención del arreglo de valores de la imagen.

Es importante considerar que obtener los datos de una imagen no es tan fácil ya que esta posee diferentes valores por cada píxel, como los valores en Rojo, Verde y Azul (colores presentes en un píxel.

Gracias a la función *open* del módulo PIL se pueden obtener la matriz en escala de grises de forma automática por lo que no es necesario trabajar a la matriz que resulta de la función para llevarla a una sola matriz. Si bien no es necesario trabajar con otras matrices, si es necesario pasar los datos leídos a una matriz de numpy, ya que con este módulo el manejo de matrices se hace mucho más fácil.

Es importante decir que solo las imágenes blanco y negro que fueron creadas en *bmp* son las que tienen esta propiedad, por lo que si se intenta procesar otra imágen, no funcionará.

Esta funcionalidad esta presente en la función getImage() creada en Python. Esta función tiene dos retornos, el primero es el arreglo de numpy con los valores de la imagen y el segundo es la imagen como tal leída.

3.2. Obtención de la Transformada de Fourier 2D

Para obtener transformada de Fourier en 2D se ha utilizado la función fft2() del módulo Scipy.

y para pasar modelar completamente en el dominio de las frecuencias en todos los ejes, se ha utilizado la función shiftfft2(). Sin embargo esta última función entrega la información a una escala que no puede ser apta para su muestreo, por lo que es necesario discretizar estos datos aplicando un logaritmo.

entonces luego de aplicar esto, se obtiene para la imagen original:

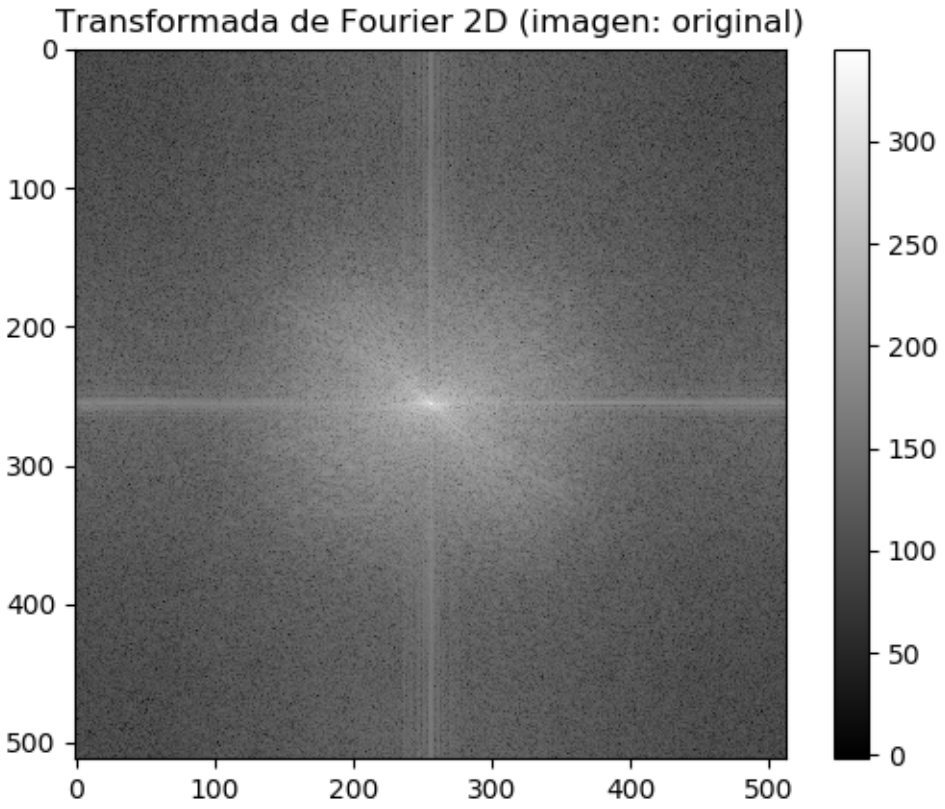

Figura 5: Transformada de la imagen original

3.3. Aplicación de filtro

Teniendo en cuenta como se abren las imágenes y que se tiene la matriz de datos correcta, es posible realizar la convolución como se describe en el apartado del marco teórico.

Para esto lo que se hace es hacer 4 ciclos for en le cual las dos primeras corresponden en a las iteraciones de filas y columnas de la matriz mas grande, las otras dos iteraciones iteran filas y columnas la matriz mas pequeña o el Kernel a implementar. Los limites de los primeros for están dados por la resta de las dimenciones de la matriz mas grande con la matriz de kernel. Los otros dos corresponden a las dimensiones del kernel.

Por lo que ocurre realmente es la superposición del kernel sobre la matriz mas grande pero el desplazamiento de arriba hacia abajo y cuando se completa, este movimiento, se hace un desplazamiento a la derecha.

Entonces haciendo la convolución con el kernel de suavizado, se obtiene la imagen:

Figura 6: Imagen con el filtro se suavizado aplicado

Como la convolución deja una imagen mas pequeña, es necesario aplicar una corrección para dejar la imagen en las mismas dimensiones que la imagen original, Esta parte se verá en el siguiente punto.

Luego la transformada de Fourier Obtenida para el suavizado fue :

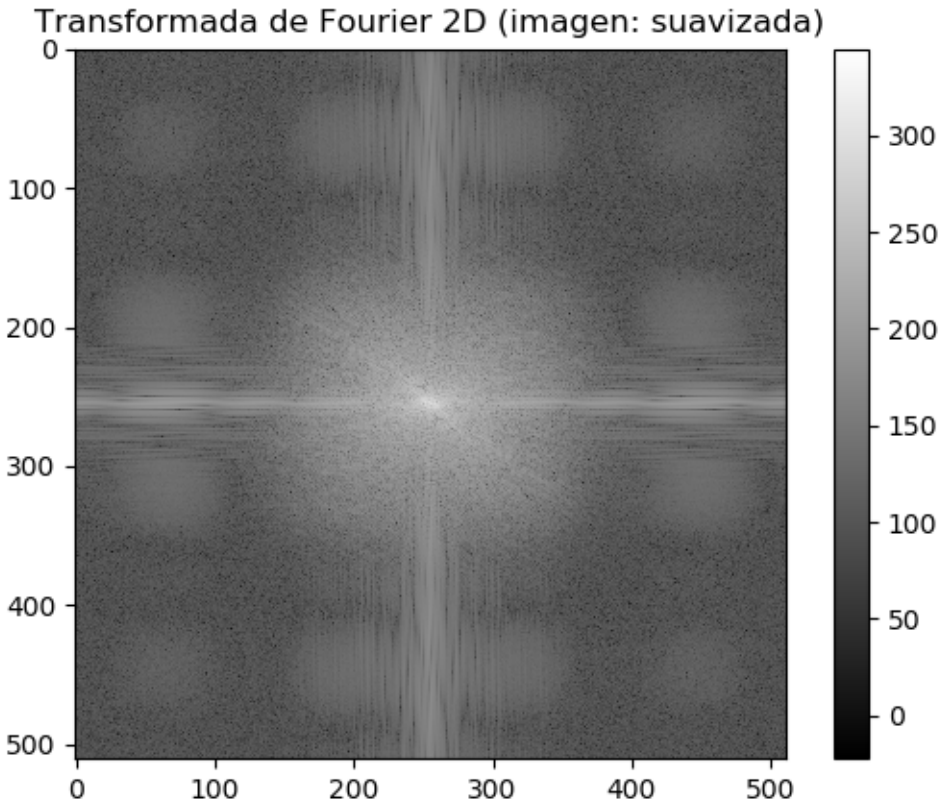

Figura 7: Transformada de Fourier para la imagen del filtro suavizado

Para la imagen con un kernel de detección de bordes , se obtiene la siguiente imagen:

Figura 8: Imagen con el filtro de detección de bordes aplicado

Cabe destacar que que para aplicar este filtro, es posible multiplicar una saturación; entre menor sea esta saturación mayor será la permanencia del tono negro, así entre mas cercano al 1 sea el valor, habrá mayor presencia del tono blanco

La transformada de Fourier para el filtro de bordes :

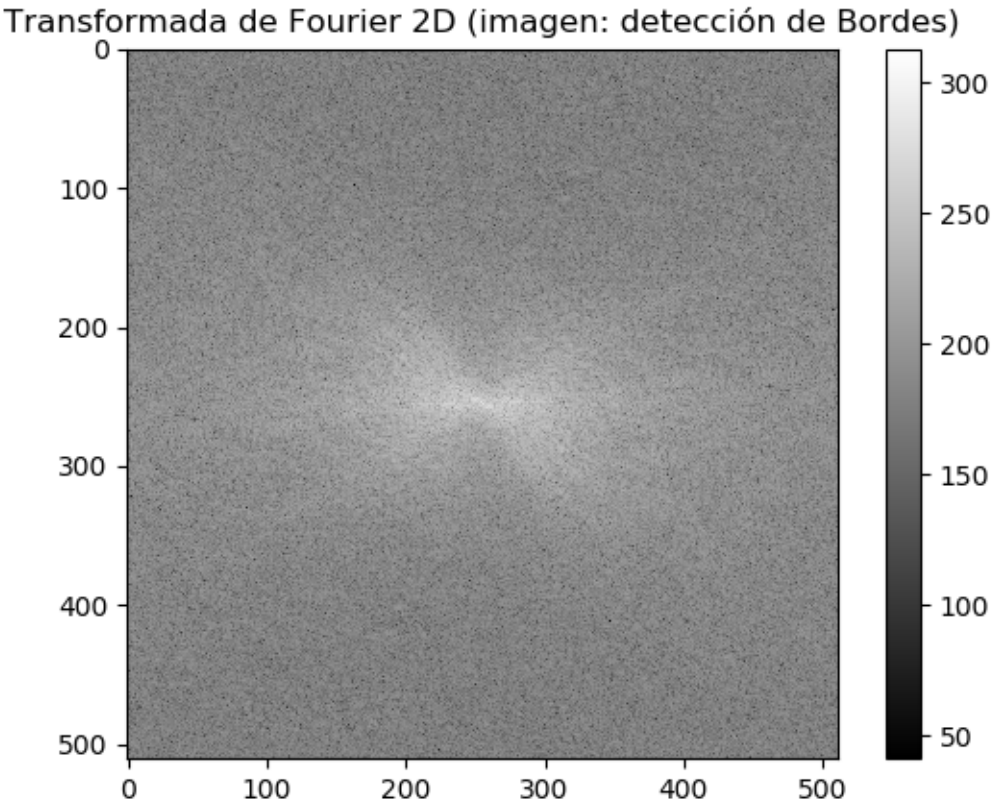

Figura 9: Transformada de Fourier para la imagen del filtro de detección de bordes

3.4. Corrección de la matriz

Para corregir la matriz se deben insertar filas y columnas de valores 0, correspondientes al tamaño del kernel menos 1. Entonces al insertar estas columnas la matriz de números originales queda rodeadas de valores 0.

Esto se se lleva a la imagen, se podrá ver que la imágenes un marco negro.

4. Análisis de los resultados

4.1. Transformada de Fourier Original

En cuanto a la Transformada de Fourier Se puede ver que en el centro resalta un destello blanco, esto quiere decir que las frecuencias con mayor amplitud se concentran en el centro.

Esto concuerda en que las imágenes posee un tono blanco, entendiendo el tono blanco como un tono que necesita varias frecuencias para producirlo.

Al concentrarse al centro, en ambos ejes, se producen estas lineas blancas que se pueden ver en la imagen de la transformada.

4.2. Filtro de suavizado

Para el filtro de suavizado aplicado, se puede ver que efectivamente se ha suavizado la imagen, esto se puede ver en el sombrero de Lanna, ya que las plumas que posee pasaron a ser un poco mas borroso y difuminado. Esto se debe a que se aplicó un kernel con valores muy simulares pero pequeños por lo que el resultado si bien, puede dar una figura mas pequeña, toma los valores cercanos por lo que la suma resulta en la predominancia del color que el kernel haya estado. Entonces se conservará los mismos valores pero con una pequeña perdida.

Como se puede ver en la imagen de la transforma de Fourier de la imagen con el filtro de suavizado, se puede ver que las lineas se ven un poco difusas, lo que claramente indica que se ha producido un cambio en la imagen.

En general la Transformada de Fourier se puede percibir de igual forma que la imagen original, pero con pequeños manchones con un tono mas blanco y las lineas del medio mas difusas.

Los manchones pueden deberse a que se perdieron ciertos valores de la imagen original por lo que al llevarla a la transformada de Fourier las frecuencias de tonos faltan por lo que se puede ver partes mas negras.

4.3. Filtro de Detección de Bordes

Para este caso el filtro fue aplicado correctamente, la imagen que se muestra del filtro en el apartado anterior, corresponde a una detección de bordes ya que, este detecta los cambios bruscos entre negro y blanco destacando la periferia de estos.

Es posible que la imagen esta con una saturación muy pequeña por lo que la presencia del tono negro está presente en toda la imagen, excepto en las partes donde se haya hecho un cambio brusco. Si se le da una saturación menor, lo que ocurre es que comienza a aparecer mayormente el tono blanco y el filtro se va haciendo mas detallado y no tan solo detecta los cambios bruscos sino que también los pequeños. Esto se puede ver como en el cambio de color en las plumas

La transformada de Fourier para este caso muestra mucho ruido y un pequeño destello casi imperceptible al centro. Esto quiere decir que los tonos blancos desaparecieron y solo hay algunos pocos, lo que corresponde con la imagen entregada. Las amplitudes disminuyeron considerablemente por lo que el filtro esta bien implementado.

5. Conclusiones

En conclusión y viendo el análisis, se pudo implementar los filtros de forma correcta ya que las imágenes entregadas corresponden con lo que se esperaba. El filtro de suavizado realmente suaviza la imagen difuminándola un tanto y el filtro de borde, genera una imagen con una detección de bordes pero este ultimo necesita una pequeña corrección con el tema de la saturación.

Con la transformada de Furier en 2D se tuvo problemas en temas de interpretar el gráfico. Esto luego se solución entendiendo que el tono blanco es el que necesita mas frecuencias en amplitud, por lo que se puede entender que el filtro de detección de bordes de una transformada de Fourier no tenga mucha amplitud.

Otros problemas que se tuvo y se mencionó antes, fue la corrección con la escala de amplitudes, ya que era necesario rectificar y normalizar los valores tan elevados que daba aplicando un logaritmo.

Con todo lo anterior se ha aprendido como se aplican los filtros y la aplicación que tiene la convolución con el procesamiento de imágenes digitales.

En síntesis, se cree que se hizo un buen laboratorio con el código bien documentada y funcional. Si bien se pueden tener errores en cuanto a la detección de bordes, este fue solucionado multiplicando una saturación.

Bibliografía

(2014). Procesamiento y análisis de imágenes digitales i. [Online] http://www.scian.cl/archivos/uploads/1149815218.5835.

(2015). Filtros electrónicos. [Online] http://mer1516bonilla.blogspot.com/2015/10/filtros-electronicos.html.

(2019). Image kernels. [Online] http://setosa.io/ev/image-kernels/.