UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

LICENCIATURA EN MULTIMEDIA Y ANIMACIÓN DIGITAL

PROCESAMIENTO DE IMAGENES

Primer Avance

Docente: Jerson Aldair Gamez Castro

Grupo: 02

Nombre del alumno: Daniel García Mazatán

Matricula: 1663204

06 de septiembre de 2021

Propuesta de proyecto

Se desarrollará una aplicación de tres ventanas, cuya funcionalidad es darle al usuario las herramientas para tomar fotos y/o videos y editarlos con algunos filtros.

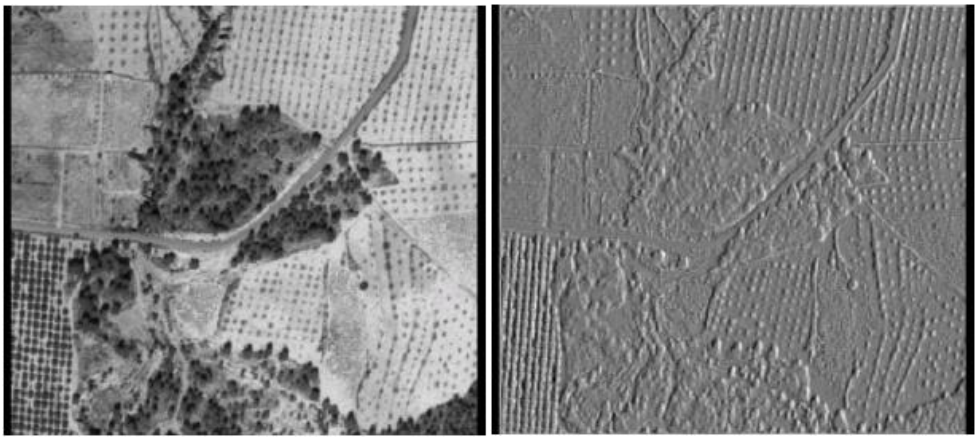
Los filtros a desarrollar son:

- **Filtro Norte-Sur**

  Se utiliza para detectar estructuras que siguen una determinada dirección en el espacio resaltando el contraste entre los píxeles situados a ambos lados de la estructura.

1	1	1
1	-2	1
-1	-1	-1

DIV=1



- **Filtro Y de Sobel**

  Se construye usando la derivada de la Gaussiana.

-1	-2	-1
0	0	0
1	2	1

  El operador Sobel calcula el gradiente de la intensidad de una imagen en cada píxel. Así, para cada punto, este operador proporciona la magnitud del gradiente, su dirección y sentido desde el más oscuro al más claro. La ventaja adicional que presentan estas máscaras sobre las anteriores es que además de estimar el valor del módulo del gradiente, al derivar sobre la Gaussiana, producen un alisamiento en la imagen que es beneficioso, dado el comportamiento ruidoso que presentan las estimaciones basadas en derivadas.

- **Filtro Laplaciano**

  La derivada de una función y = f(x) es el incremento de y para cada incremento infinitesimal de x. En el caso de Modelo Digital de Elevaciones la derivada es la pendiente. La segunda derivada es la derivada de la derivada, en el caso de un MDE nos da información acerca de la forma (ladera recta, cóncava o convexa, valle, cresta o cima) del terreno. En el caso de una imagen de satélite nos va a informar de como son los cambios, más o menos bruscos, que se producen entre pixeles contiguos. A continuación aparecen las ecuaciones de las derivadas respeto a x e y, las segundas derivadas respeto a x e y, la derivada compuesta y la derivada segunda compuesta denominada también laplaciana.

$$\Delta xp(i,j) = p(i,j) - p(i-1,j)$$

$$\Delta yp(i,j) = p(i,j) - p(i,j-1)$$

$$\Delta x_2 p(i,j) = \Delta xp(i+1,j) - \Delta xp(i,j) = p(i+1,j) + p(i-1,j) - 2p(i,j)$$

$$\Delta y_2 p(i,j) = \Delta yp(i,j+1) - \Delta yp(i,j) = p(i,j+1) + p(i,j-1) - 2p(i,j)$$

$$\Delta xyp(i,j) = sqr\Delta xp(i,j)^2 + \Delta yp(i,j)^2$$

$$\Delta xy_2 p(i,j) = [p(i+1,j) + p(i-1,j) + p(i,j+1) + p(i,j-1)] - 4p(i,j)$$

0	1	0
1	-4	1
0	1	0

DIV=1

- **Detector de bordes de Canny**

Uno de los métodos relacionados con la detección de bordes es el uso de la primera derivada, la que es usada por que toma el valor de cero en todas las regiones donde no varía la intensidad y tiene un valor constante en toda la transición de intensidad. Por tanto un cambio de intensidad se manifiesta como un cambio brusco en la primera derivada [1], característica que es usada para detectar un borde, y en la que se basa el algoritmo de Canny.
El algoritmo de Canny consiste en tres grandes pasos:

•Obtención del gradiente: en este paso se calcula la magnitud y orientación del vector gradiente en cada píxel.
•Supresión no máxima: en este paso se logra el adelgazamiento del ancho de los bordes, obtenidos con el gradiente, hasta lograr bordes de un píxel de ancho.
•Histéresis de umbral: en este paso se aplica una función de histéresis basada en dos umbrales; con este proceso se pretende reducir la posibilidad de aparición de contornos falsos.

Obtención del gradiente
Para la obtención del gradiente, lo primero que se realiza es la aplicación de un filtro gaussiano a la imagen original con el objetivo de suavizar la imagen y tratar de eliminar el posible ruido existente. Sin embargo, se debe de tener cuidado de no realizar un suavizado excesivo, pues se podrían perder detalles de la imagen y provocar un pésimo resultado final. Este suavizado se obtiene promediando los valores de intensidad de los píxels en el entorno de vecindad con una mascara de convolución de media cero y desviación estándar σ.

Una vez que se suaviza la imagen, para cada píxel se obtiene la magnitud y módulo (orientación) del gradiente, obteniendo así dos imágenes.

(a)

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

(b)

$$\frac{1}{115}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

Supresión no máxima al resultado del gradiente

Las dos imágenes generadas en el paso anterior sirven de entrada para generar una imagen con los bordes adelgazados. El procedimiento es el siguiente: se consideran cuatro direcciones identificadas por las orientaciones de 0º, 45º, 90º y 135º con respecto al eje horizontal. Para cada píxel se encuentra la dirección que mejor se aproxime a la dirección del ángulo de gradiente.

Algoritmo: Obtención de Gradiente

Entrada: imagen I
máscara de convolución H, con media cero y desviación estándar σ.

Salida: imagen E_m de la magnitud del gradiente
imagen E_o de la orientación del gradiente

1. Suavizar la imagen I con H mediante un filtro gaussiano y obtener J como imagen de salida.
2. Para cada píxel (i, j) en J, obtener la magnitud y orientación del gradiente basándose en las siguientes expresiones:
   El gradiente de una imagen $f(x,y)$ en un punto (x,y) se define como un vector bidimensional dado por la ecuación:

$$G[f(x,y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} f(x,y) \\ \frac{\partial}{\partial y} f(x,y) \end{bmatrix}$$

   siendo un vector perpendicular al borde, donde el vector G apunta en la dirección de variación máxima de f en el punto (x,y) por unidad de distancia, con la magnitud y dirección dadas por:

$$|G| = \sqrt{G_x^2 + G_y^2} = |G_x| + |G_y|,$$

$$\varphi(x,y) = \tan^{-1}\frac{G_y}{G_x}$$

3. Obtener E_m a partir de la magnitud de gradiente y E_o a partir de la orientación, de acuerdo a las expresiones anteriores.

Posteriormente se observa si el valor de la magnitud de gradiente es más pequeño que al menos uno de sus dos vecinos en la dirección del ángulo obtenida en el paso anterior. De ser así se asigna el valor 0 a dicho píxel, en caso contrario se asigna el valor que tenga la magnitud del gradiente.

La salida de este segundo paso es la imagen I_n con los bordes adelgazados, es decir, $E_m(i, j)$ después de la supresión no máxima de puntos de borde.

Histéresis de umbral a la supresión no máxima

La imagen obtenida en el paso anterior suele contener máximos locales creados por el ruido. Una solución para eliminar dicho ruido es la histéresis del umbral.

El proceso consiste en tomar la imagen obtenida del paso anterior, tomar la orientación de los puntos de borde de la imagen y tomar dos umbrales, el primero más pequeño que el segundo. Para cada punto de la imagen se debe localizar el siguiente punto de borde no explorado que sea mayor al segundo umbral. A partir de dicho punto seguir las cadenas de máximos locales conectados en ambas direcciones perpendiculares a la normal del borde siempre que sean mayo res al primer umbral. Así se marcan todos los puntos explorados y se almacena la lista de todos los puntos en el contorno conectado. Es así como en este paso se logra eliminar las uniones en forma de Y de los segmentos que confluyan en un punto.

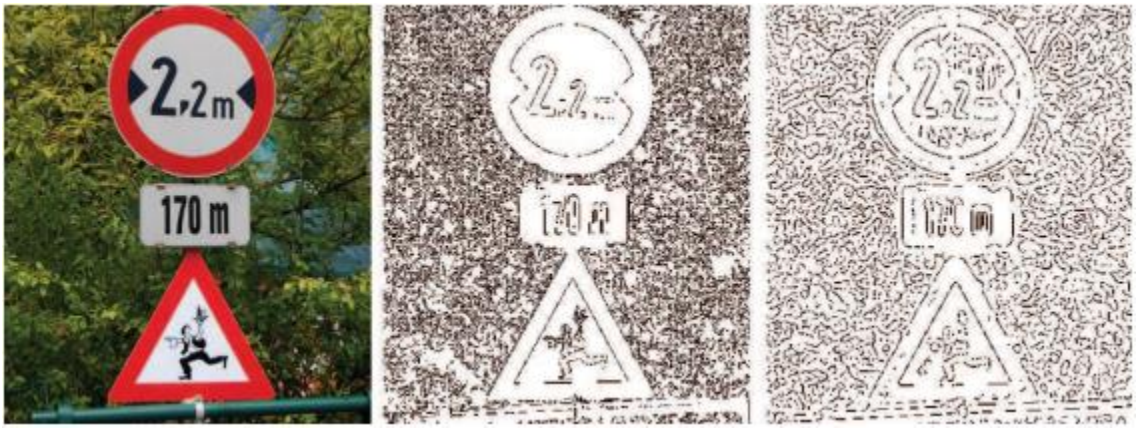
Algoritmo: Supresión no máxima

Entrada: imagen E_m de la magnitud del gradiente
imagen E_o de la orientación del gradiente

Salida: imagen I_n

Considerar: cuatro direcciones d_1, d_2, d_3, d_4 identificadas por las direcciones de 0°, 45°, 90° y 135° con respecto al eje horizontal

1. Para cada píxel *(i, j)* :
   1.1. Encontrar la dirección d_k que mejor se aproxima a la dirección E_o *(i, j)*, que viene a ser la perpendicular al borde.
   1.2. Si E_m *(i, j)* es más pequeño que al menos uno de sus dos vecinos en la dirección d_k , al píxel *(i, j)* de I_n se le asigna el valor 0, $I_n(i, j) = 0$ (supresión), de otro modo $I_n(i, j) = E_m(i, j)$.
2. Devolver I_n



- **Filtro menos laplaciano**

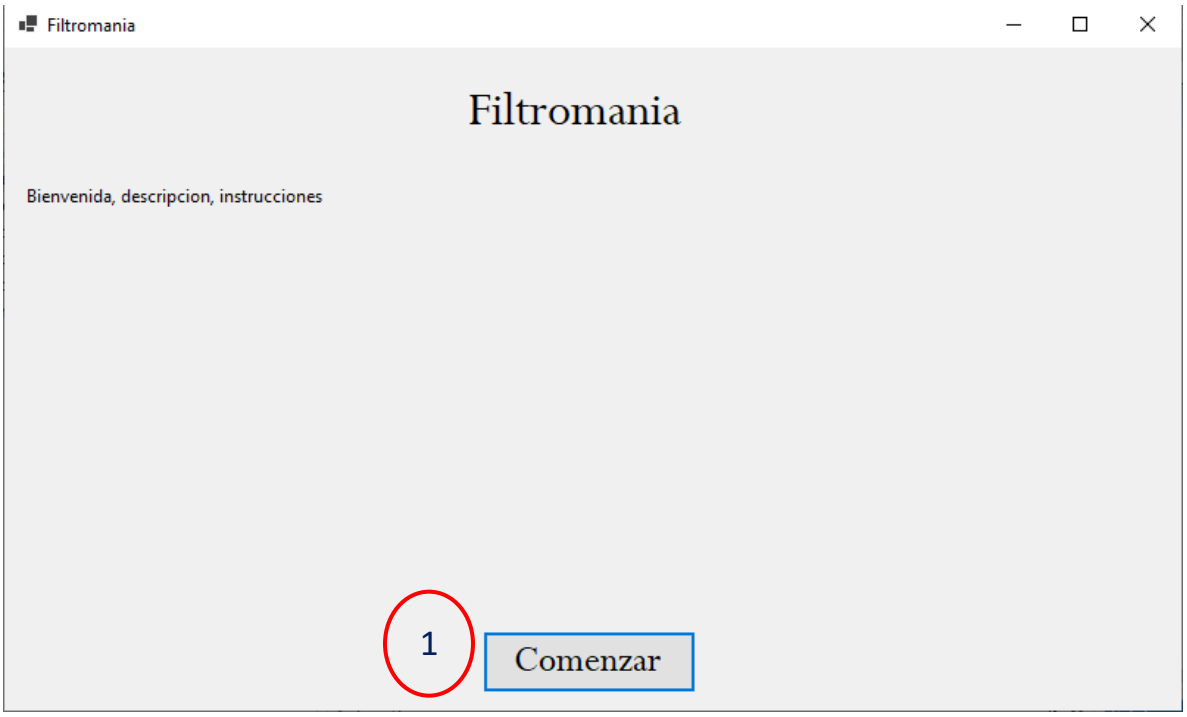
Es como el filtro laplaciano, pero negativo, es decir, sirve para delinear los bordes de las estructuras que se encuentren en la imagen sin perder de vista todos los demás elementos. Sirve para filtrar fotos satelitales para obtener una imagen clara.

0	-1	0
-1	5	-1
0	-1	0

DIV=1

Propuesta de interface

- Ventana 1



En esta ventana se mostrará un mensaje de bienvenida a la aplicación, así como una pequeña descripción y algunos pasos para seguir durante su ejecución.

1. Botón Comenzar: sirve para comenzar la aplicación.

➢ Ventana 2

En esta ventana se toma la foto o video.

2. Combo box Lista de cámaras: aquí se muestran las cámaras disponibles conectadas al dispositivo para tomar la foto.
3. Botón actualizar: sirve para actualizar la lista de cámaras.
4. Caja de imagen: aquí se muestra lo que esta captando la cámara en vivo.
5. Botón Encender cámara: sirve para encender la cámara seleccionada en la lista de cámaras.
6. Botón toma foto: sirve para capturar la foto al instante.
7. Botón grabar: sirve para grabar video.
8. Botón continuar: sirve para guardar la imagen para, enseguida, redirigirse a la ventana de edición.
9. Botón regresar: sirve para cancelar la ejecución de la toma de foto.

➢ Ventana 3

En esta ventana se aplican los filtros sobre la foto o video que se tomo en la ventana anterior.

10. Caja de imagen donde se muestra la imagen o video original tomada anteriormente.
11. Caja de imagen donde se muestra la foto o video tomado ya con los filtros aplicados.
12. Botones de filtros: cada uno corresponde a uno de los filtros a desarrollar, y se aplican sobre la foto o video.
13. Botón deshacer filtros: sirve para deshacer los procesos de filtrado de la foto o video, para retornarla a la original y volver a aplicar filtros.

14. Botón finalizar: sirve para dar por terminado el uso de la aplicación y devuelve al usuario a la primera ventana.
15. Botón regresar: cancela el proceso de edición y retorna a la ventana de toma de foto o video.