



Tecnológico de Monterrey

Evidencia de Proyecto

Herramientas Computacionales: El Arte de la Programación

Equipo:

Jonathan Uziel Medina Rodríguez (A01255048)

Pablo Ernesto Moreno Cruz (A01255437)

Miguel de Jesús Degollado Macías (A01255388)

Docente: Baldomero Olvera Villanueva

Fecha de entrega:

Domingo 23 de marzo de 2025.

Investigación teórica:

La convolución es un método que filtra valores de los píxeles que componen a una imagen, lo cual permite que pueda ser más nítida o suavizada, que sus ejes y bordes sean detectados, etc.

Filtros de convolución:

Existen diferentes tipos de filtros o kernels para aplicar la convolución a una imagen. Incluso, el filtro puede ser desarrollado por un mismo usuario, de tal manera que se adecúe a sus necesidades. Algunos de estos filtros son:

- Nitidez: Permite que la diferencia entre el valor de un píxel y el de sus vecinos sea acentuada, resaltando límites entre cada entidad y hacer que las aristas de los objetos de la imagen sean más nítidas. Algunos tipos de filtros de nitidez son:

- Nitidez 3x3:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- Nitidez 5x5:

$$\begin{bmatrix} -1 & -3 & -4 & -3 & -1 \\ -3 & 0 & 6 & 0 & -3 \\ -4 & 6 & 21 & 6 & -4 \\ -3 & 0 & 6 & 0 & -3 \\ -1 & -3 & -4 & -3 & -1 \end{bmatrix}$$

- Sobel: Es utilizado para poder detectar las aristas en una imagen.

- Sobel vertical:

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- Sobel horizontal:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

- Detección de línea/borde: Este filtro permite que se realcen los bordes en una imagen.

- Norte a sur:

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- Noreste a suroeste:

$$\begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

- Este a oeste:

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

- Sureste a noroeste:

$$\begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}$$

- Scharr: Este filtro tiene buena simetría rotacional y permite que se detecten las aristas con más exactitud.

- Scharr horizontal (mejora la intensidad en la parte horizontal de la imagen):

$$\begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix}$$

- Scharr vertical (mejora la intensidad en la parte vertical de la imagen):

$$\begin{bmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix}$$

En el programa en el que trabajamos se hizo el uso de una versión modificada de cada uno de los filtros antes mencionados, de tal manera que se pudieran definir los bordes de la imagen de una radiografía y distinguir otros detalles.

Descripción del Programa:

Este programa consiste en un analizador de radiografías que hace uso de filtros de convolución para que se puedan ver mejor las anomalías que están presentes en una radiografía, ya cuerpos extraños como los objetos ingeridos por un niño pequeño, enfermedades respiratorias, etc. La principal innovación de este programa es que permite que el usuario pueda diferenciar los focos de consolidación para poder diferenciar una neumonía de foco único y de una de focos múltiples, áreas de consolidación de la trama bronquial, vascular y adenopatías (ganglios) parahiliares. También, este programa puede utilizarse si no hay una máquina de tomografía (TAC) disponible en un hospital.

Como ya se mencionó en la sección de la investigación, en este programa se hace el de versiones de los filtros de nitidez, Scharr, Sobel y de detección de bordes que fueron modificadas para obtener mejores resultados en este programa.

Requerimientos del programa:

- Python 3 (se recomienda la versión 3.13 al ser la más reciente).
- Tener instaladas las siguientes librerías:
 - Numpy
 - OS
 - OpenCV
 - Matplotlib
- La imagen de una radiografía que vaya a ser analizada (es recomendable que esté en el mismo directorio que el archivo del programa).

Funciones creadas:

- convolucion(): Aplica un filtro de convolución a una imagen en escala de grises. La precondition es tener una imagen que pueda ser leída por el programa. Las entradas que acepta son la matriz de la imagen, la matriz del filtro a aplicar y las medidas del padding horizontal y vertical. La salida resultante es una matriz de la imagen con el filtro aplicado. La postcondición

es que la matriz de la imagen tenga el filtro aplicado. La complejidad espacial y temporal de esta función es $O(n^2)$, ya que se recorren n filas de n columnas en la imagen.

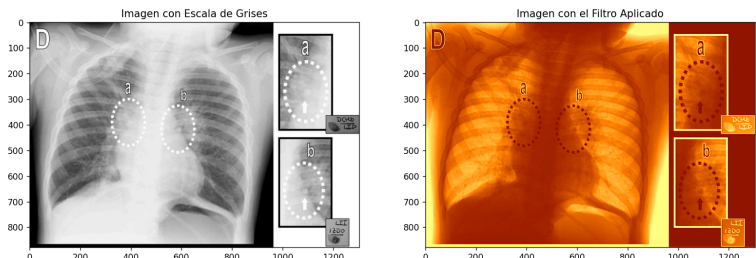

- guardar_imagen(): Guarda la imagen filtrada en una carpeta llamada "Fotos". La precondition es tener una imagen que pueda ser leída por el programa. Las entradas que acepta son la imagen a guardar y el nombre de la imagen. La función no tiene un dato de salida, pero la postcondición es que haya guardado la imagen exitosamente y que se haya creado la carpeta "Fotos" si no existía antes. La complejidad espacial y temporal de esta función es $O(1)$, ya que solo hay líneas de código que llaman funciones y que son condicionales. La función no cuenta con iteraciones.

Link del repositorio con los archivos del programa:

https://github.com/JonathanUzielMedina/Laboratorio_A01255048/tree/main

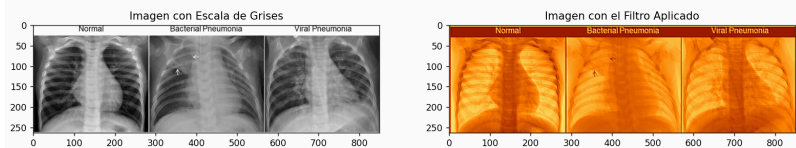
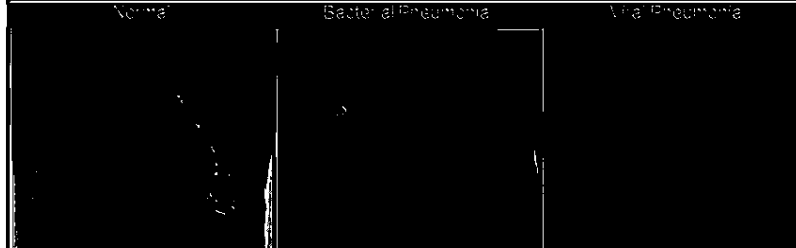
Casos de Prueba:

- **Caso 1:**

Nombre del archivo	rx2.jpg
Mapa de color	afmhot
Padding en el eje Y	*No se ingresó ninguno, por lo que su valor es 1 por defecto.
Padding en el eje X	*No se ingresó ninguno, por lo que su valor es 1 por defecto.
Resultado	<p style="text-align: center;">Graficación:</p>  <p style="text-align: center;">Archivo:</p> 

Error(es) y/o notas	La función cv2.imwrite no soportó la profundidad de la matriz con todos los filtros aplicados. Por eso la imagen exportada no es muy parecida a la que se mostró en la graficación.
Propósito	Se demuestra el funcionamiento normal del programa, donde se ingresa el nombre de una imagen de una radiografía de cuerpos extraños y que muestra el funcionamiento de los filtros aplicados en la graficación.

- **Caso 2:**

Nombre del archivo	pneumo.png
Mapa de color	<u>Intento 1:</u> afmho <u>Intento 2:</u> afmhot
Padding en el eje Y	2
Padding en el eje X	2
Resultado	<p>Graficación:</p>  <p>Archivo:</p> 
Error(es)	<u>Intento 1 [Mapa de color]:</u> “El mapa de color no es

	<p>válido. Ingrese uno de los solicitados.”. Si el usuario ingresa el nombre del mapa de color de manera incorrecta, el programa le dice que el nombre ingresado no forma parte de los mapas que maneja.</p> <p>La función cv2.imwrite no soportó la profundidad de la matriz con todos los filtros aplicados. Por eso la imagen exportada no es muy parecida a la que se mostró en la graficación.</p>
Propósito	En este caso se busca demostrar que el programa también sirve para diferenciar detalles en otro tipo de radiografías, donde en este caso se trata de enfermedades pulmonares, donde se puede distinguir mejor las condensaciones de las pulmonías.

- **Caso 3:**

Nombre del archivo	rx4.jpg
Mapa de color	*No aplica.
Padding en el eje Y	*No aplica.
Padding en el eje X	*No aplica.
Resultado	*No aplica.
Error(es)	“No se pudo encontrar la imagen.”.
Propósito	El programa necesita encontrar una imagen que esté dentro de la misma carpeta.

- **Caso 4:**

Nombre del archivo	rx2
Mapa de color	*No aplica.
Padding en el eje Y	*No aplica.
Padding en el eje X	*No aplica.
Resultado	*No aplica.
Error(es)	“No se pudo encontrar la imagen.”.

Propósito	Aunque haya una imagen llamada “rx2” en la misma carpeta donde se ubica el programa, es necesario especificar también la extensión del archivo.
------------------	---

Referencias:

Choosing Colormaps in Matplotlib. (s. f.). Matplotlib. Recuperado el 22 de marzo de 2025 de <https://matplotlib.org/stable/users/explain/colors/colormaps.html>

Fig. 2. (s. f.). Radiology Key. <https://radiologykey.com/body-aspiration-imaging-aspects/>

Figura 2. (2020). Elsevier. <https://www.elsevier.es/es-revista-radiologia-119-avance-resumen-aspectos-radiologicos-neumonia-covid-19-evolucion-S0033833820301661>

Figura 2. (s. f.). Anales de Pediatría. <https://www.analesdepediatría.org/es-cuerpo-extrano-bilateral-radiografia-toracica-articulo-S1695403314005347>

Figuras 7. RXs frontales en inspiración (a) y espiración (b). (s. f.). Sociedad Canaria de Pediatría de Santa Cruz de Tenerife. <https://scptfe.com/diagnostico-por-la-imagen-de-los-cuerpos-extranos-alojados-en-la-via-aerea-en-pediatria/>

FIGURE 11. (s. f.). ResearchGate. https://www.researchgate.net/figure/The-normal-chest-X-ray-left-panel-depicts-clear-lungs-without-any-areas-of-abnormal_fig5_361301975

Función de convolución. (s. f.). ArcGIS Desktop. Recuperado el 22 de marzo de 2025 de <https://desktop.arcgis.com/es/arcmap/latest/manage-data/raster-and-images/convolution-function.htm>

GeeksforGeeks. (2020a, 22 abril). *Matplotlib.pyplot.imshow() in Python*.
GeeksforGeeks. Recuperado el 20 de marzo de 2025 de
<https://www.geeksforgeeks.org/matplotlib-pyplot-imshow-in-python/>

GeeksforGeeks. (2022b, 10 febrero). *How to Fix: ValueError: setting an array element with a sequence*. GeeksforGeeks. Recuperado el 20 de marzo de 2025
de
<https://www.geeksforgeeks.org/how-to-fix-valueerror-setting-an-array-element-with-a-sequence/>

GeeksforGeeks. (2023c, 14 marzo). *Introduction to Convolutions using Python*.
GeeksforGeeks. Recuperado el 20 de marzo de 2025 de
<https://www.geeksforgeeks.org/introduction-to-convolutions-using-python/>

GeeksforGeeks. (2024d, 24 abril). *Python - How to Check if a file or directory exists*.
GeeksforGeeks. Recuperado el 20 de marzo de 2025 de
<https://www.geeksforgeeks.org/python-check-if-a-file-or-directory-exists/>

GeeksforGeeks. (2024e, 22 julio). *Types of convolution kernels*. GeeksforGeeks.
Recuperado el 20 de marzo de 2025 de
<https://www.geeksforgeeks.org/types-of-convolution-kernels/>

GeeksforGeeks. (2024f, 2 agosto). *Python OpenCV | cv2.imread() method*.
GeeksforGeeks. Recuperado el 20 de marzo de 2025 de
<https://www.geeksforgeeks.org/python-opencv-cv2-imread-method/>

GeeksforGeeks. (2024g, 28 agosto). *Numpy.sum() in Python*. GeeksforGeeks.
Recuperado el 21 de marzo de 2025 de
<https://www.geeksforgeeks.org/numpy-sum-in-python/>

GeeksforGeeks. (2024h, 25 noviembre). *How to display multiple images in one figure correctly in Matplotlib?* GeeksforGeeks. Recuperado el 20 de marzo de 2025
de
<https://www.geeksforgeeks.org/how-to-display-multiple-images-in-one-figure-correctly-in-matplotlib/>

GeeksforGeeks. (2025i, 24 enero). *Numpy.zeros() in Python*. GeeksforGeeks. Recuperado el 20 de marzo de 2025 de <https://www.geeksforgeeks.org/numpy-zeros-python/>

8.2 *Matriz de convolución.* (s. f.). GIMP. <https://docs.gimp.org/2.6/es/plugin-convmatrix.html>

Oliveres, J., & Escalante, B. (2011). *Convolución y Filtrado* [Presentación; Digital]. Universidad Nacional Autónoma de México. <https://lapi.fi-p.unam.mx/wp-content/uploads/6-Filtros-y-morfologia.pdf>

Olvera, B. (s. f.-a). *Convolución* [Diapositivas; Digital]. Tecnológico de Monterrey. https://experiencia21.tec.mx/courses/554652/discussion_topics/3503409

Olvera, B. (s. f.-b). *Procesamiento de Imágenes y Visión Computacional* [Diapositivas; Digital]. Tecnológico de Monterrey. https://experiencia21.tec.mx/courses/554652/discussion_topics/3503409

Ormesher, I. (2021, 14 diciembre). Convolution Filters. *Medium*. Recuperado el 20 de marzo de 2025 de <https://medium.com/@ianormy/convolution-filters-4971820e851f>

Basado en los siguientes códigos:

- "convolution.py" por Abhisek Jana. Recuperado de https://github.com/benjaminva/semena-tec-tools-vision/tree/master/Scripts/Ejem_plos
- "simple_conv.py" por Abhisek Jana y Benajmin Valdes. Recuperado de https://github.com/benjaminva/semena-tec-tools-vision/tree/master/Scripts/Ejem_plos
- "simple_sobel.py" por Abhisek Jana y Benajmin Valdes. Recuperado de https://github.com/benjaminva/semena-tec-tools-vision/tree/master/Scripts/Ejem_plos